

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
вычислительной техники

Кафедра
МОиПЭВМ

Направление подготовки 09.03.04 «Программная инженерия»

Направленность Программная инженерия

БАКАЛАВРСКАЯ РАБОТА

на тему

**Бизнес-проект «Система для исторических квест-
экскурсий». Web-приложение для
конструирования квестов и анализа статистики**

Студент _____ Агапова Валерия Александровна
(подпись, дата) (ФИО полностью)

Руководитель _____ Такташкин Д.В.
(подпись, дата) (фамилия, инициалы)

Нормоконтролер _____ Такташкин Д.В.
(подпись, дата) (фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от _____ № _____)

Заведующий кафедрой _____ Козлов А.Ю.
(подпись) (фамилия, инициалы)

Работа защищена с отметкой _____ (протокол заседания ГЭК от _____ № _____)

Секретарь ГЭК _____ Попова Н.А.
(подпись) (фамилия, инициалы)

Пенза, 2024

Экспертное заключение
о согласовании темы ВКР в форме стартапа

Экспертная комиссия в составе: декан ФВТ д.т.н., профессор Л.Р. Фионова, доцент кафедры ВТ, к.э.н., И.В. Акифьев, главный разработчик ООО «Ай-теко ВЦ», к.т.н. М.П. Синев, рассмотрев стартап-проект «Система для исторических квест-экскурсий» Агаповой Валерии Александровны подтверждает его актуальность, практическую значимость и возможность коммерциализации.

Решение экспертной комиссии: выпускная квалификационная работа на тему «Бизнес-проект «Система для исторических квест-экскурсий». Web-приложение для конструирования квестов и анализа статистики» Агаповой Валерии Александровны может выполняться в форме стартапа (бизнес-проекта).

Председатель ЭК

д.т.н., профессор
декан ФВТ

Л.Р. Фионова

Члены ЭК

к.э.н., доцент кафедры ВТ

И.В. Акифьев

к.т.н., главный разработчик
ООО «Ай-теко ВЦ»

М.П. Синев

Федеральное государственное бюджетное образовательное учреждение
высшего образования
ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ
ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет
Вычислительной техники

Кафедра
МО и ПЭВМ

«Утверждаю»
Заведующий кафедрой
А.Ю. Козлов
«25» декабря 2023 г

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ В ФОРМЕ
СТАРТАПА

1. Студент Агапова Валерия Александровна гр. 20ВП1 факультета ВТ
<Фамилия, имя, отчество полностью>

2. Тема работы Бизнес-проект «Система для исторических квест-экскурсий». Web-
приложение для конструирования квестов и анализа статистики

Тема утверждена приказом ПГУ № _____ от " _____ " _____ 20__ г. _____

2. Руководитель работы к.т.н., доцент кафедры «МОиПЭВМ» Такташкин Д.В.

3. Общая часть ВКР: _____

Подготовка компьютерной презентации бизнес-проекта (стартапа).

4. Индивидуальная часть ВКР:

4.1. Задание на работу (назначение разработки, исходные данные и т.п.)

Назначение: программное средство предназначено для конструирования создания
исторических квест-экскурсий и анализа статистики

Основные функции:

1) Управление квест-экскурсиями

2) Управление пользователями

3) Просмотр статистики квестов

4) Просмотр статистики пользователей

Технологии разработки: ООП

Язык программирования: JavaScript, C#

Инструментальные средства разработки: Microsoft Visual Studio 2022 Community Edition,
Visual Studio Code

Среда исполнения: Linux, Docker

4.2. Перечень подлежащих разработке вопросов

1. Анализ предметной области и требований к системе. Постановка задачи
2. Проектирование программных средств
3. Реализация программных средств
4. Тестирование программных средств
5. Оформление пояснительной записки

5. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Разработка бизнес-плана проекта	10%	08.04.2024 – 10.04.2024	
2	Анализ предметной области и требований к системе	10%	11.04.2024 – 13.04.2024	
3	Проектирование программных средств	20%	15.04.2024 – 25.04.2024	
4	Реализация программных средств	30%	26.04.2024 – 12.05.2024	
5	Тестирование программных средств	10%	13.05.2024 – 22.05.2024	
6	Оформление пояснительной записки	20%	23.05.2024 – 03.06.2024	

Дата выдачи задания «21» декабря 2023г.

Руководитель бакалаврской работы _____
(подпись, дата)

Задание к исполнению принял студент _____
(подпись, дата)

Работу к защите допустить
Декан ФВТ _____ Л.Р. Фионова
(подпись, дата)

Реферат

Пояснительная записка содержит 198 листов, 47 рисунков, 27 таблиц, 3 листинга, 40 использованных источников, 5 приложений.

БИЗНЕС-ПРОЕКТ, WEB-ПРИЛОЖЕНИЕ, АДМИНИСТРИРОВАНИЕ СИСТЕМЫ, АВТОМАТИЗИРОВАННЫЕ КВЕСТ-ЭКСКУРСИИ, ДОПОЛНЕННАЯ РЕАЛЬНОСТЬ, АРХИТЕКТУРА НА ОСНОВЕ СЕРВИСОВ, POSTGRESQL, ASP.NET CORE, C#, REACT

Объект исследования: программные средства организации туристического досуга с использованием геймификации.

Целью выпускной квалификационной работы как стартапа является разработка бизнес-плана проекта «Система для исторических квест-экскурсий».

Целью индивидуальной части выпускной квалификационной работы как стартапа является разработка web-приложения для конструирования квестов и анализа статистики для системы исторических квест-экскурсий с применением технологии дополненной реальности.

Технология разработки: ООП.

Среда исполнения: Linux, Docker.

Область применения: организация туризма.

Разработка серверной части проводилась на языке программирования C# в среде разработки Microsoft Visual Studio 2022 Community, разработка клиентской части проводилась с использованием библиотеки React в среде разработки Visual Studio Code. Разработка производилась в операционной системе Windows 11.

					ПГУ 09.03.04 – 08BP201.01 ПЗ							
Изм.	Лист	№ докум.	Подпись	Дата	Бизнес-проект «Система для исторических квест-экскурсий». Web-приложение для конструирования квестов и анализа статистики. Пояснительная записка			Лит.	Лист	Листов		
Разраб.		Агапова В.А.									5	198
Провер.		Такташкин Д.В.						Группа 20ВП1				
Реценз.												
Н. Контр.		Такташкин Д.В.										
Утверд.		Козлов А.Ю.										

Содержание

Введение.....	8
1 Бизнес-проект «Система для исторических квест-экскурсий Questie»	10
1.1 Постановка цели и задач бизнес-проекта	10
1.2 Описание инновационной идеи	10
1.3 Описание продукта	13
1.4 Анализ рынка.....	15
1.5 Маркетинговый план	16
1.6 Финансовый план	18
1.7 Достигнутые результаты	19
2 Web-приложение для конструирования квестов и анализа статистики	21
2.1 Аналитическая часть.....	21
2.1.1 Анализ предметной области	21
2.1.2 Сравнительный анализ программных продуктов-аналогов	22
2.1.2.1 Анализ Ubego.....	23
2.1.2.2 Анализ SurpriseMe.....	25
2.1.2.3 Анализ Urban Quest.....	26
2.1.2.4 Результат проведенного анализа	28
2.1.3 Анализ требований к программным средствам	29
2.1.3.1 Анализ функциональных требований.....	29
2.1.3.2 Анализ нефункциональных требований	32
2.1.4 Постановка цели и задачи	33
2.2 Проектирование программных средств для проведения квестов с применением технологии дополненной реальности	34
2.2.1 Проектирование архитектуры серверной части.....	35
2.2.1.1 Выбор архитектуры серверной части	35
2.2.1.2 Разбиение серверной части на сервисы	40
2.2.1.3 Проектирование последовательности выполнения запросов.....	42

2.2.1.4 Проектирование выполнения основной логики обработки запросов.....	44
2.2.2 Проектирование логической модели базы данных.....	46
2.2.3 Проектирование клиентской части web-приложения для конструирования квестов и анализа статистики.....	48
2.2.4 Проектирование пользовательского интерфейса клиентской части web-приложения для конструирования квестов и анализа статистики.....	52
2.2.5 Планирование разработки и оценка бюджета.....	56
2.3 Реализация программных средств для проведения квестов с применением технологии дополненной реальности	60
2.3.1 Реализация серверной части	60
2.3.2 Реализация базы данных.....	62
2.3.3 Реализация клиентской части web-приложения для конструирования квестов и анализа статистики.....	64
2.3.4 Реализация пользовательского интерфейса	74
2.3.5 Тестирование и оценка качества кода.....	78
2.3.5.1 Функциональное тестирование.....	78
2.3.5.2 Структурное тестирование.....	81
2.3.5.3 Метрики качества кода.....	84
2.3.5.4 Модульное тестирование	87
Заключение	88
Список использованных источников	90
Приложение А (рекомендуемое) Результаты бизнес-проекта	95
Приложение Б (рекомендуемое) Код программы.....	102
Приложение В (рекомендуемое) Свидетельство о государственной регистрации программы для ЭВМ	167
Приложение Г (рекомендуемое) Тест-кейсы	169
Приложение Д (рекомендуемое) Исходный код модульного тестирования	190

Введение

Развитие туризма, как и большинства других отраслей, в настоящее время активно цифровизируется [1, 2], что позволяет внедрять новые технологии, направленные на улучшение качества услуг и персонализацию путешествий.

Бизнес-проект «Система для исторических квест-экскурсий» предназначен для предоставления туристам возможности проведения досуга в геймифицированной форме с познавательным сопровождением, система является эффективным инструментом популяризации исторических материалов и изучения географии городов в процессе ознакомления с их достопримечательностями.

Разрабатываемое программное обеспечение должно представлять собой:

- мобильное приложение, позволяющее проходить экскурсионные маршруты на основе квестов с элементами дополненной реальности;
- web-приложение для управления системой (конструирования квестов и анализа статистики).

Общую актуальность разрабатываемого продукта можно обосновать следующими фактами:

- Правительство России ведет активную политику по поддержке и развитию отечественного туризма;
- по данным ВЦИОМ 96% россиян считают важным знание истории своей страны и своего региона;
- турпоток по городам России за три квартала 2023 года по данным Росстата составил 134 955 161 поездок, что на 16,48% больше, чем за тот же период 2022 года;
- политическая обстановка привела к ограниченному выезду россиян за границу.

Происходит переориентация потока на внутренний туристический рынок. Около 5 миллионов туристов (примерно 150 млрд рублей в год в денежном

эквиваленте) аккумулируются внутри страны, что влечет за собой необходимость развития и повышения качества туризма в России.

Требуется развивать туризм в регионах и привлекать внимание к менее популярным направлениям, создавать туристические маршруты. Разрабатываемый продукт сможет разгрузить туристические потоки в крупных городах. Необходимость внедрения технологии дополненной реальности в проект объясняется увеличением благодаря ей уровня вовлеченности пользователей в экскурсионный процесс и повышением заинтересованности.

Основной целевой аудиторией являются туристы в возрасте от 18 до 50 лет, путешествующие по городам России, а также туристические агентства, заказывающие уникальные квесты под требования бизнеса.

1 Бизнес-проект «Система для исторических квест-экскурсий Questie»

1.1 Постановка цели и задач бизнес-проекта

Сформулированной в ходе анализа задания целью данной выпускной квалификационной работы как стартапа является разработка бизнес-плана проекта «Система для исторических квест-экскурсий».

Для достижения поставленной цели необходимо выполнить следующие задачи:

- выполнить детализированное описание инновационной идеи;
- сформировать описание предлагаемого продукта и его технико-экономические характеристики;
- провести анализ рынка и спрогнозировать развитие отрасли;
- разработать маркетинговый план, описывающий ценообразование продукта и способ его продажи;
- разработать финансовый план, содержащий этапы развития бизнес-проекта и описывающий ожидаемые финансовые показатели на каждом из этапов;
- найти партнеров проекта и привлечь финансирование на первый этап разработки.

1.2 Описание инновационной идеи

Экскурсионная деятельность является основой формирования туристского продукта. Современный рынок туристской индустрии требует внедрения инновационных технологий при организации экскурсионной деятельности. Высокий уровень конкуренции среди туристских компаний обуславливает необходимость постоянного развития, отслеживания тенденций потребительского спроса и внедрения новых форматов [1]. Одним из новых форматов, набирающих популярность среди путешественников, является квест-экскурсия.

Квест-экскурсия – это услуга по организации посещения специально отобранных объектов экскурсионного показа, предполагающая наличие сюжетной линии и препятствий в форме различных задач, головоломок, игр, разгадывая которые участники знакомятся и изучают конкретные объекты посредством использования имеющихся знаний, наблюдения и общения с другими субъектами [2].

Квест-экскурсия является комбинацией квеста и экскурсии, включающей в себя элементы обоих понятий, приведенные на рисунке 1.

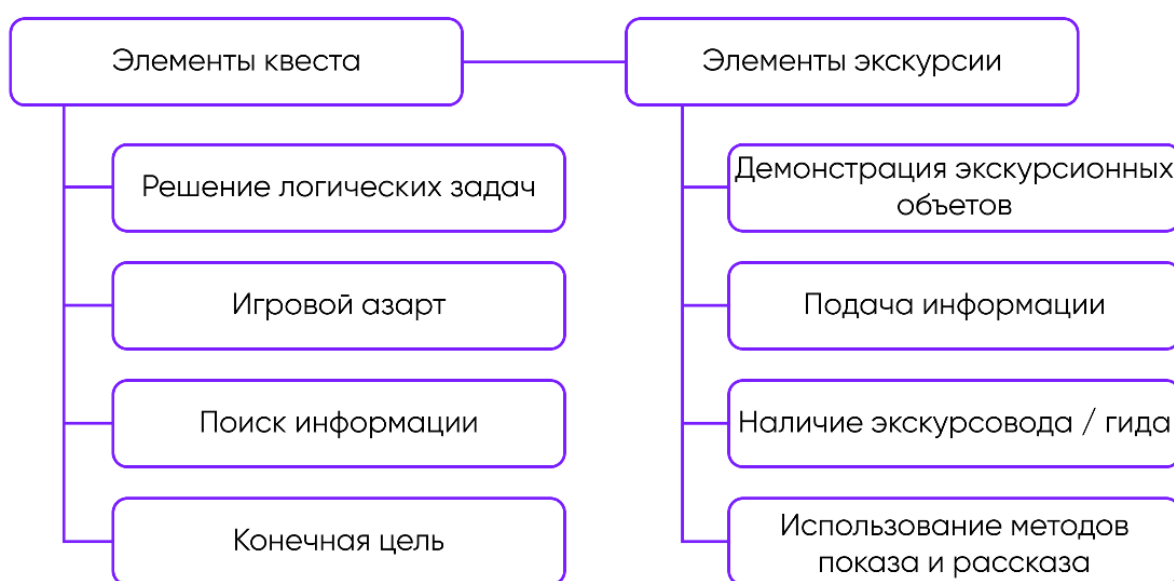


Рисунок 1 – Составляющие квест-экскурсии

Развитие цифровизации приводит к появлению различных мобильных приложений, покрывающих все сферы жизни человека. Туризм не стал исключением. Так, описанные выше квест-экскурсии переносятся в мобильные приложения, становясь удобным и интересным дополнением для путешествий, не требующим непосредственного контакта с гидом или экскурсоводом.

Повышает интерес пользователей к продукту также и использование технологии дополненной реальности (AR). Согласно исследованию компании Neuro-Insight Study, дополненная реальность почти втрое увеличивает уровень когнитивной активности мозга во время обучения, по сравнению с аналогичным

опытом, не связанным с AR. При этом, формируются устойчивые связи между объектом изучения (на котором размещена AR метка) и связанным с ним контентом (полученном в AR режиме с этого объекта) [3].

Разрабатываемая система должна предоставлять возможность прохождения квест-экскурсий с применением технологии дополненной реальности, а также обеспечивать необходимые административные функции, такие как управление пользователями, конструирование квестов и отслеживание статистики. Наиболее трудоемкая задача при разработке такой системы – наполнение контентом, самой сложной частью которого является создание 3D-объектов для генерации AR-таргетов. Этот процесс необходимо максимально автоматизировать для снижения стоимости продукта. Для этого планируется создать и применить комбинированный метод построения 3D-моделей, основанный на нейросетевом анализе и позволяющий использовать меньшее количество исходных данных.

Исходя из этого, можно сделать вывод, что система должна состоять из трех компонентов:

- мобильное приложение для проведения квестов с применением технологии дополненной реальности;
- приложение администратора (web-приложение для конструирования квестов и анализа статистики);
- нейросетевой программный модуль для генерации 3D-моделей.

Предлагается создать экскурсионные маршруты на основе квестов с элементами дополненной реальности в формате приложения для мобильных устройств. Приложение должно позволять пользователям выбирать квесты в городе своего расположения, получать баллы за выполнение заданий и доступ к историческим справкам, следить за достижениями друзей. Уникальность проекта заключается в использовании элементов дополненной реальности на станциях квестов.

Приложение администратора должно упрощать работу администраторов системы и создателей маршрутов квест-экскурсий, предоставляя им возможность конструировать квест-экскурсии, отслеживать статистику по квестам и пользователям, управлять пользователями системы.

Инновационная идея реализуется в несколько этапов. Первым этапом является исследование алгоритмов генерации 3D-моделей по изображениям, разработка программного модуля генерации 3D-моделей по изображениям, исследование зависимости качества сгенерированных 3D-моделей от входных параметров модуля генерации, а также исследование эффективности различных моделей баз данных в зависимости от объема выборки. Вторым этапом является разработка клиентского мобильного приложения с применением технологии дополненной реальности и приложения администратора. Далее будет проводиться апробация минимального жизнеспособного продукта и его вывод на рынок.

Приобретение аппаратных средств и неэксклюзивных прав на программное обеспечение не потребовало привлечения дополнительных финансов и было осуществлено из средств гранта УМНИК.

Ключевым фактором успешной реализации плана является научно-техническое превосходство, состоящее в результатах, достигнутых в ходе НИОКР в рамках реализации проекта УМНИК. Основными рисками реализации плана являются нехватка финансирования на реализацию маркетинговых кампаний, возможность возникновения несовместимости с некоторыми устройствами пользователей, не обладающими поддержкой ARCore (в случае с операционной системой Android), неудовлетворенность пользователей контентом и быстрое изменение их предпочтений.

1.3 Описание продукта

Описание продукта приведено в таблице 1.

Таблица 1 – Описание продукта

Характеристика	Описание
Наименование продукта	Программный комплекс для разработки и проведения городских квест-экскурсий с применением технологии дополненной реальности
Назначение и область применения	Предназначен для организации научно-развлекательного досуга групп людей в сфере самостоятельного туризма [4] путем проведения квест-экскурсий по туристическим маршрутам и историческим, культурным и природным достопримечательностям
Перспективы выпуска на рынке	Целевым рынком является рынок самостоятельного туризма. Потенциальные покупатели – туристы в возрасте от 18 до 50 лет, путешествующие по городам России (около 50 млн человек, из которых лишь 25 млн – платежеспособная аудитория); туристические агентства, заказывающие уникальные квесты под требования бизнеса. Способ поставки – мобильное приложение, устанавливаемое на мобильные телефоны пользователей и являющееся конечным продуктом.
Возможность замещения	В настоящее время подавляющая доля рынка приходится на отечественные приложения для проведения автоматизированных городских квест-экскурсий. Продукты иностранных компаний представлены лишь в нескольких городах. Замещение отечественных аналогов возможно за счет более качественной реализации программного обеспечения, внедрения инновационного функционала, а также лучшей проработанности сценариев квестов.
Технико-экономические характеристики	Разрабатываемая система должна представлять собой мобильное приложение (ПО) под управлением операционной системы Android версии 10.0 и новее. Телефон должен обладать не менее 200 Мб свободной памяти в хранилище и 2 Гб оперативной памяти. Обязательно наличие камеры, поддержка AR Core и доступ к сети интернет. Модуль генерации 3D-моделей по изображениям реализован с применением языка программирования Python 3, фреймворка машинного обучения PyTorch и библиотек NumPy для обработки многомерных массивов, PyOpenGL, trimesh для визуализации. Компьютер для запуска модуля генерации 3D-моделей (для администратора системы) должен обладать следующими рекомендуемыми техническими характеристиками: объем оперативной памяти: 64 Гб, объем памяти видеокарты: 24 Гб, объем свободной памяти в хранилище: 100 Гб. Оптимальное количество фотографий для генерации 3D-моделей – 50 шт., что позволяет производить генерацию в среднем за 5 минут на видеокарте RTX 3090 Ti.
Конкурентоспособность продукта	Основана на научно-техническом превосходстве, состоящем в использовании комбинированного метода построения 3D-моделей, используемых для построения AR-сцен, основанного на нейросетевом анализе. Данный метода позволяет сделать создание AR-контента менее трудозатратным и недорогим, а также обеспечивает возможность распознавания 3D-таргетов со всех сторон.

Продолжение таблицы 1

Характеристика	Описание
Потенциал повышения конкурентоспособности	Возможно дальнейшее проведение НИОКР с целью совершенствования методов работы над созданием AR-контента. Кроме того, повышению конкурентоспособности может способствовать качественная проработка сценариев квест-экскурсий.
Необходимость лицензирования	Отсутствует необходимость в приобретении сторонних лицензий. Права на РИД защищены.
Степень готовности	К началу выполнения выпускной квалификационной работы как стартапа командой проекта разработан комбинированный метод построения 3D-моделей, используемых для построения AR-сцен, основанный на нейросетевом анализе; разработано программное средство, реализующее данный метод; выполнено прототипирование пользовательского интерфейса мобильного приложения.
Безопасность и экологичность	Продукт не несет и не может нести угроз окружающей среде.

В данном разделе был описан разрабатываемый продукт и его основные характеристики.

1.4 Анализ рынка

Анализ рынка приведен в таблице 2.

Таблица 2 – Анализ рынка

Характеристика	Описание
Объем рынка	На основании данных Яндекс.Вордстат российский рынок экскурсий составляет около 15 млрд рублей в год. Оценка объема рынка по модели TAM/SAM/SOM. TAM – 15,3 млрд. рублей. SAM – 219.9 млн. рублей. SOM – 1,4 млн. рублей.
Темпы роста рынка	Рынок отечественного туризма в последние годы активно растет. Политическая обстановка привела к ограниченному выезду россиян за границу. Происходит переориентация потока на внутренний туристический рынок. Около 5 миллионов туристов аккумулируются внутри страны. Турпоток по городам России за три квартала 2023 года по данным Росстата составил 134 955 161 поездок, что на 16,48% больше, чем за тот же период 2022 года.
Производственные возможности	Производство продукта требует лишь разовых первоначальных затрат на разработку. Дальнейшие производственные работы сводятся к наращиванию базы квестов с целью масштабирования, а также к доработке и внедрению нового функционала с целью поддержания и увеличения уровня интереса аудитории.
Характер отрасли	Туристическая отрасль имеет сезонный характер, но в долгосрочной динамике стабильно развивается.

Продолжение таблицы 2

Характеристика	Описание
Динамика продаж за последние 5 лет	На динамику продаж за последние пять лет существенным образом повлияли такие факторы как пандемия, повлекшая за собой резкое падение отрасли, а также политическая ситуация, способствовавшая перенаправлению потока отечественных туристов на внутренний рынок. По результатам более стабильного 2023 года рост продаж отрасли составил по разным оценкам от 10% до 15%. В 2024 году предполагается рост продаж еще на 15%.
Прогнозы развития отрасли	Согласно Стратегии развития туризма от 20.09.2019 [5] к 2035 году планируется: увеличить количество внутренних туристских поездок на одного жителя более чем в два раза; увеличить экспорт туристских услуг более чем в три раза; увеличить инвестиции в сферу туризма в три раза. В 2020 году показатель количества туристов оценивался в 45,22 млн. человек. К 2025 году эту цифру планируется увеличить в 1,8 раза, до 83,78 млн. За следующие пять лет этот показатель должен вырасти еще более чем в полтора раза, до 140,29 млн. человек. Наконец, к 2035 году число туристов в России должно составить почти 144 млн человек, треть из которых – иностранцы [6].
Основные и потенциальные конкуренты	Основные конкуренты: Ubego – лидер рынка отечественных сервисов автоматизированных городских квестов; Quest Quiz – развивающийся сервис, автоматизирующий проведение квизов и викторин; SurpriseMe – развивающийся сервис, автоматизирующий проведение квизов; Urban Quest – развивающийся сервис, автоматизирующий проведение квизов. Потенциальные конкуренты: аудиогиды, например, izi.TRAVEL (более 1 млн. скачиваний), Sputnik8, YouTravel.Me, GetYourGuide, WeGoTrip, TravelRu и другие.
Уровень рентабельности отрасли	По оценкам рентабельности смежных направлений отрасли чистая рентабельность должна составить от 9% до 11% [7], что позволяет сделать вывод о высокой рентабельности проекта.

Таким образом, был проанализирован рынок сбыта продукта, выявлены прямые и косвенные конкуренты и сделаны прогнозы развития отрасли до 2035 года. Выявлено, что отрасль имеет высокую рентабельность и постоянно развивается.

1.5 Маркетинговый план

Для продвижения разрабатываемой системы для исторических квест-экскурсий с применением технологии дополненной реальности на туристическом рынке, привлечения и удержания целевой аудитории был разработан маркетинговый план, приведенный в таблице 3.

Таблица 3 – Маркетинговый план

Характеристика	Описание
Конечные потребители	Туристы в возрасте от 18 до 50 лет, путешествующие по городам России (около 50 млн чел., из которых лишь 25 млн – платежеспособная аудитория); туристические агентства, заказывающие уникальные квесты под требования бизнеса.
Уровень удовлетворения спроса	Представленные в данный момент на рынке продукты предоставляют квест-экскурсии лишь в 50 городах России, не перекрывая всю географию страны. В большинстве городов присутствует лишь по 1 – 2 квест-экскурсии.
Особенности сегмента рынка	Конкурентное преимущество продукта зависит от характеристик и разнообразия возможностей программного обеспечения для прохождения квестов. Также большое значение имеет качество проработанности сюжетов, что влияет на повторной использование продукта при выборе других квестов. Отрасль имеет сезонный характер.
Конкурентные преимущества товара (услуги) конкурентов	Ubego – большое количество квестов в разных городах; Quest Quiz – разнообразные форматы квестов; SurpriseMe – разнообразные форматы заданий, предложение для бизнеса.
Планируемая доля рынка	За 3-ий этап (с 05.2025 по 04.2026) реализации проекта планируется занять 0,64% рынка. За 4-ый этап (с 05.2026 по 04.2027) реализации проекта планируется занять 9,58% рынка. За 5-ый этап (с 05.2026 по 07.2028) реализации проекта планируется занять 31,96% рынка.
Патентная ситуация	Разработанное программное обеспечение будет зарегистрировано в качестве программы для ЭВМ в Федеральном институте промышленной собственности (ФИПС) с целью предотвращения прямого копирования. Копирование внутренних алгоритмов работы системы будет затруднительно выявить.
Обоснование цены на продукт	На начальном этапе развития проекта предполагается установить цену на квест-экскурсии ниже цен аналогичных продуктов, продающихся за 800 – 2600 рублей за квест. Во-первых, это даст дополнительное конкурентное преимущество. Во-вторых, снизит недовольство клиентов от возможных программных сбоев на стадии вывода первой версии продукта на рынок. С учетом необходимости покрытия операционных расходов, предполагаемая средняя цена может составлять 450 рублей за квест и увеличиваться на дистанции в процессе улучшения предлагаемого продукта.
Организация сбыта	Предполагается коммерциализировать продукт в двух направлениях. Основное направление коммерциализации – b2c: продажа доступа к платным квест-маршрутам пользователям мобильного приложения. По направлению b2b предполагается продажа рекламы в приложении компаниям из сферы услуг, расположенным на туристических маршрутах, а также сотрудничество с туристическими агентствами: создание эксклюзивных квестов под требования бизнеса.

Таким образом, был разработан маркетинговый план для программного комплекса для разработки и проведения городских квест-экскурсий с применением технологии дополненной реальности. Описана целевая аудитория продукта и особенности организации сбыта.

1.6 Финансовый план

Финансовый план приведен в таблице 4.

Таблица 4 – Финансовый план

Этап	Описание работ
Первый этап реализации (с 08.2023 по 05.2024)	Август 2023 – ноябрь 2023: разработка комбинированного метода построения 3D-моделей, используемых для построения AR-сцен, основанного на нейросетевом анализе и программного средства, реализующего данный метод. Декабрь 2023 – май 2024: разработка первой версии мобильного приложения для проведения исторических квест-экскурсии, а также административной панели. Июнь 2024: разработка первых туристических маршрутов по городу Пензе. Июль 2024: апробация минимального жизнеспособного продукта в некоммерческом формате на фокусных группах.
Второй этап реализации (с 08.2024 по 04.2025)	Август 2024: завершение обязательств по грантовой программе УМНИК. Сентябрь 2024 – декабрь 2024: исправление и доработка начального функционала в соответствии с результатами апробации минимального жизнеспособного. Январь 2025: регистрация юридического лица, поиск сценаристов исторических квестов (общая сумма около 100000 рублей). Февраль 2025: разработка туристических маршрутов и сценариев исторических квестов (60000 рублей). Март 2025: пробная маркетинговая компания (e-mail рассылка, целевая реклама в Интернет) (50000 рублей); разработка туристических маршрутов и сценариев исторических квестов (60000 рублей). Апрель 2025: подготовка заявки на конкурс Старт-1.
Третий этап реализации (с 05.2025 по 04.2026)	Май 2025 – апрель 2026: настройка маркетинговой компании (70000 рублей в месяц); разработка туристических маршрутов и сценариев исторических квестов (60000 рублей); первые продажи (предполагается 3000 продаж на 1,4 млн. рублей).
Четвертый этап реализации (с 05.2026 по 04.2027)	Май 2026: получение средств по программе Старт-1 (до 1 млн. рублей) Май 2026 – апрель 2027: работы над улучшением функционала системы; настройка маркетинговой компании; разработка туристических маршрутов и сценариев исторических квестов; увеличение среднего чека продажи; продажи (предполагается 30000 продаж на 21 млн. рублей).
Пятый этап реализации (с 05.2026 по 07.2028)	Май 2027: получение средств на реализацию второго этапа программы Старт-1. Май 2027 – июль 2028: поиск оптимальной модели реализации продукта, перевод исполнителей в штат; выход на самоокупаемость; поиск путей масштабирования бизнеса.

Таким образом, был разработан финансовый план на первые 5 лет реализации программного комплекса для разработки и проведения городских квест-экскурсий с применением технологии дополненной реальности. Выделены основные этапы разработки и продвижения продукта. Увеличение продаж запланировано за счет постепенного увеличения экскурсионных маршрутов. Финансовая модель 3 – 5 этапов плана приведена в таблице 5.

Таблица 5 – Финансовая модель 3 – 5 этапов плана развития проекта

	3 этап	4 этап	5 этап
Объем продаж, ед.	3000	30000	75000
Количество клиентов	1500	15000	25000
Средний чек, руб.	467	700	937
Выручка, тыс. руб.	1401	21000	70275
Расходы, тыс. руб., в т.ч.:	1600	15000	32775
Переменные, тыс. руб.	720	7200	12000
Постоянные, тыс. руб.	880	7800	20775
Операционная прибыль, тыс. руб.	-199	6000	37500
Маржа, %	-14,2%	25,6%	53,3%
Грант, тыс. руб.	3000	Отсутствует	Отсутствует
Инвестиции, займы, тыс. руб	Отсутствуют	Отсутствуют	Отсутствуют

Из составленной финансовой модели видно, что уже на четвертом этапе реализации проект должен выйти на самоокупаемость и начать приносить первую прибыль.

1.7 Достигнутые результаты

Реализуемому проекту одобрено грантовое финансирование в размере 500000 рублей (Договор №18446ГУ/2023 от 16.08.2023) по программе УМНИК-2022 (Пензенская область) [8].

Компания ООО «Открытые решения» выразила поддержку проекту и заявила о готовности к консультативной и информационной помощи. Кафедра «Маркетинг, коммерция и сфера обслуживания» Института экономики и управления выразила готовность к экспертной поддержке проекта в разработке содержательной части квест-экскурсий (туристических маршрутов и сюжетов для заданий квестов). Министерство культуры и туризма Пензенской области выразило заинтересованность в реализации и продвижении проекта. Все

организации выразили готовность поддержки проекта в своих информационных ресурсах. Письма поддержки приведены в приложении А.

Была проведена работа со СМИ, в выпуске №15 (240) от 24 октября 2023 молодежной газеты Пензенской области «Деловой» была опубликована статья «По Пензе минувших эпох» о проекте системы для исторических квест-экскурсий с применением технологии дополненной реальности [9].

Пройдена преакселерационная программа «Дотком инновации» [10] и акселерационная программа Пензенского государственного университета «Делать вместе новое» при поддержке стартап-студии «Открытые инновации» [11], позволившие качественнее проработать идею проекта и повысить компетенции членов команды. Сертификаты приведены в приложении А.

К началу реализации ВКР членами команды был разработан программный модуль генерации 3D-моделей по изображениям, проведено исследование алгоритмов генерации 3D-моделей, качества сгенерированных моделей в зависимости от количества входных изображений. Выявлено оптимальное количество входных параметров для разработанного программного модуля генерации 3D-моделей по изображениям, наилучшим образом подходящая для использования в программной системе.

2 Web-приложение для конструирования квестов и анализа статистики

2.1 Аналитическая часть

2.1.1 Анализ предметной области

В последние годы проблема развития внутреннего туризма в России становится все более актуальной. 24 декабря 2021 года было опубликовано постановление правительства РФ об утверждении государственной программы Российской Федерации «Развитие туризма». Данное постановление предусматривает развитие цифровой инфраструктуры в туризме, направленной на продвижение туристских продуктов, упрощение навигации и получения информации о направлениях и возможностях туризма [12].

Сегодня квест одна из популярных форм проведения досуга. На современном рынке услуг представлены самые разнообразные развлекательные квесты, ориентированные на самых разных потребителей [13].

Высокий уровень конкуренции среди туристских компаний обуславливает постоянный поиск новых форм предоставления услуг и изменение существующих предложений. К примеру, такая услуга как экскурсия, претерпела изменения и теперь может предоставляться в разных формах: классической экскурсии, аудиоэкскурсии, виртуальной экскурсии, квест-экскурсии и др.

Квест-экскурсия – это методически и методологически созданный и реализованный, интерактивный и нестандартный показ достопримечательностей (объекты природного, историко-культурного наследия и др.) в основе которого лежит анализ находящихся перед глазами экскурсантов объектов, а также качественный рассказ о событиях, связанных с ними, продолжительностью менее 24 часов без ночевки, посредством выполнения определенных заданий и прохождения маршрута.

Последние десятилетия происходит цифровизация всех сфер жизни человека и туризм не исключение. Широкое распространение получили

различные путеводители, аудиогиды, навигаторы и прочие необходимые туристу источники информации в формате мобильных приложений. Квест-экскурсии также становятся мобильными приложениями и занимают свое место в списке удобных и полезных для путешественников способов узнавать историю городов и необычные факты о них, осматривать памятники архитектуры и получать другие знания. Практичность использования таких приложений обусловлена отсутствием необходимости контактировать с гидами и экскурсоводами, а также возможность в игровой форме осматривать достопримечательности, ведь по данным агентства Gamification Now! использование в приложениях подхода геймификации увеличивает вовлеченность пользователей на 30% [14].

Трудоемкой задачей является разработка квест-экскурсий и наполнение подобных приложений контентом. Помимо того, что создателю квест-экскурсии необходимо разработать маршрут экскурсии и задания для нее, сопровождаемые интересными фактами о достопримечательностях, контент должен быть загружен в мобильное приложение, чтобы стать доступным для клиента. Кроме того, важным фактором успеха приложения на рынке является его продвижение. Для этого важно выстраивать различные маркетинговые стратегии, запускать рекламу в различных источниках, а также отслеживать успех реализации этих программ, используя различную статистику.

Удобным инструментом для этих целей может стать web-приложение для создания квест-экскурсий, позволяющие уменьшать трудоемкость наполнения контентом клиентских мобильных приложений.

2.1.2 Сравнительный анализ программных продуктов-аналогов

Рассмотрим существующие на рынке web-приложения для создания квест-экскурсий.

2.1.2.1 Анализ Ubego

Ubego – система, состоящая из клиентского мобильного приложения для прохождения квест-экскурсий, а также web-приложения для создания квест-экскурсий и других квестов. Web-приложение предоставляет возможность создавать авторские квесты: добавлять информацию о квесте, оформление квеста (изображения), задания квеста. Страница редактирования информации о квесте приведена на рисунке 2.

Ubego Личный кабинет создателя Квесты Тарифы Финансы Главная leeraag RU

Редактирование информации о квесте

Имя: Пенза

Описание: Общее описание квеста (предыстория, историческая справка и т.п.)

Краткое описание: Краткая версия описания квеста

Максимальное количество участников?: 5

Стоимость, Р?: 800

Расстояние, м?: 0

Теги:

- ☐ Призы
- ☐ Подходит для детей
- ☐ Задания в баре
- ☐ Тема эротического характера
- ☐ Семейный квест
- ☐ На свидание
- ☐ Вместе с друзьями

Специальные теги: 6+

Сохранить Отменить

Рисунок 2 – Страница редактирования информации о квесте Ubego

Первым шагом создания квеста является выбор города проведения квеста, а также добавление названия квеста. При редактировании информации о квесте пользователь должен ввести название, описание, максимальное количество

участников, стоимость квеста, расстояние, которое участники квеста должны будут преодолеть за время прохождения квеста, а также добавить различные теги. При выборе некоторых тегов наблюдается смысловое несоответствие.

Страница редактирования задания квест-экскурсии приведена на рисунке 3.

Рисунок 3 – Страница редактирования задания квест-экскурсии Ubego

При редактировании задания пользователь должен ввести текст задания, решения, текст после выполнения задания, подсказки, прочую информацию, местоположение прохождения задания. Кроме того, пользователь может указать, что задание будет выполняться в заведении либо при взаимодействии с людьми. Отсутствует выбор типа задания, добавление количества баллов за задания, нельзя добавить изображение отдельно для каждого задания.

При редактировании заданий есть возможность оставить все поля незаполненными, после добавления и сохранения задания не происходит переход к следующему шагу, а также встречаются другие неполадки в функционировании web-сайта. Система не предоставляет создателям квестов отслеживание статистики.

2.1.2.2 Анализ SurpriseMe

SurpriseMe – платформа для создания и проведения квестов, из клиентского мобильного приложения для прохождения квестов, а также web-приложения, которое предназначено как для прохождения квестов, так и для их создания. Web-приложение позволяет создать авторские квесты под любые потребности или заказать разработку квеста. После входа в аккаунт, интерфейс добавления квеста доступен на английском языке, что является существенным недостатком. Русскоязычная версия также есть, однако для этого необходимо изменить настройки языка в профиле.

Страница добавления вопроса квеста представлена на рисунке 4.

The screenshot displays the 'Welcome' event configuration page in the SurpriseMe web application. The interface is in English. On the left, there is a sidebar menu with options: Codes, Route Map, List of the tour events, 1 Welcome, 2 Task 1, 3 AR TEST, 4 TEST, and Event name. The main content area is titled 'Welcome' and includes the following sections:

- Pictures and videos:** A section with a blue header and a light blue box containing a camera icon and the text 'Drag photos or click to select. Up to photos or GIFs, 1 video, maximum 10 MB each'.
- Text:** A section with a blue header and a text input field. Above the input field are icons for bold, italic, underline, link, unlink, list, and image. Below the input field is a character count '0/1000'.
- Audio:** A section with a blue header and a light blue box containing a music note icon and the text 'Drag an audio or click to select'.
- You can skip the event:** A toggle switch that is currently turned on. Below it, there is a 'Bonus points' field with the value '10' and the text 'bonus points will get the user for passing the event'.
- How the next event starts «Задание 1»?** A toggle switch that is currently turned on. Below it, there is a text input field with the value 'далее' and the text 'Allow traveler to open the next step of your tour easily'.
- At the certain place:** A toggle switch that is currently turned off.
- After a question and answer:** A toggle switch that is currently turned off.
- By finding the artefact:** A toggle switch that is currently turned on.

At the bottom of the main content area is a blue button labeled 'Next event'. On the right side of the screen, there is a preview of the mobile app interface showing a 'Welcome' screen with a 'далее' button and a 'STEP 1/4' indicator.

Рисунок 4 – Страница добавления вопроса квеста SurpriseMe

При добавлении задания создатель квеста имеет возможность добавить изображение, а также количество баллов за прохождение конкретного задания. Возможности сервиса представлены рисунке 5.

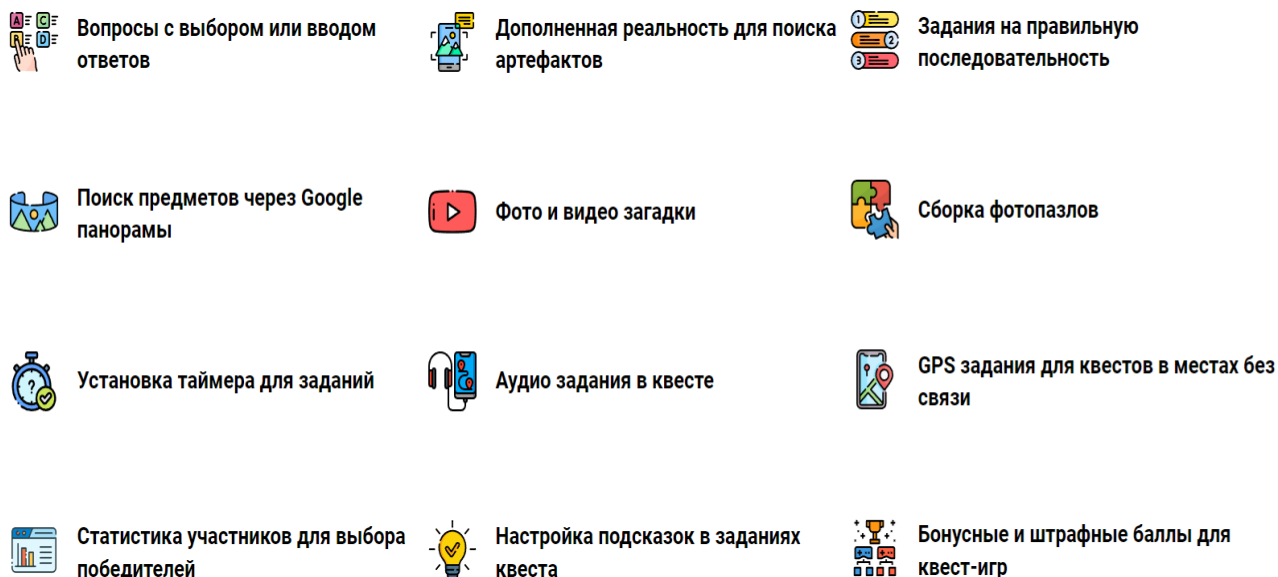


Рисунок 5 – Возможности SurpriseMe

При тестировании сервиса не была обнаружена заявленная функциональность в полном объеме: нет установки таймера для задания, добавления заданий на правильную последовательность, заданий с фотопазлами. Выбор города при добавлении квестов не предоставляется. Отслеживание статистики по квестам не предоставляется.

2.1.2.3 Анализ Urban Quest

Urban Quest также представляет собой комплексную платформу для создания и прохождения квестов, позволяющих путешественникам узнавать новые города. Экран добавления информации о квесте приведен на рисунке 6. Пользователю предоставляется возможность загрузить изображение, выбрать тип квеста, язык проведения квеста, добавить название, описание и текст начала квеста. При добавлении информации о квесте пользователь вводит адрес, выбирая таким образом город проведения квеста.

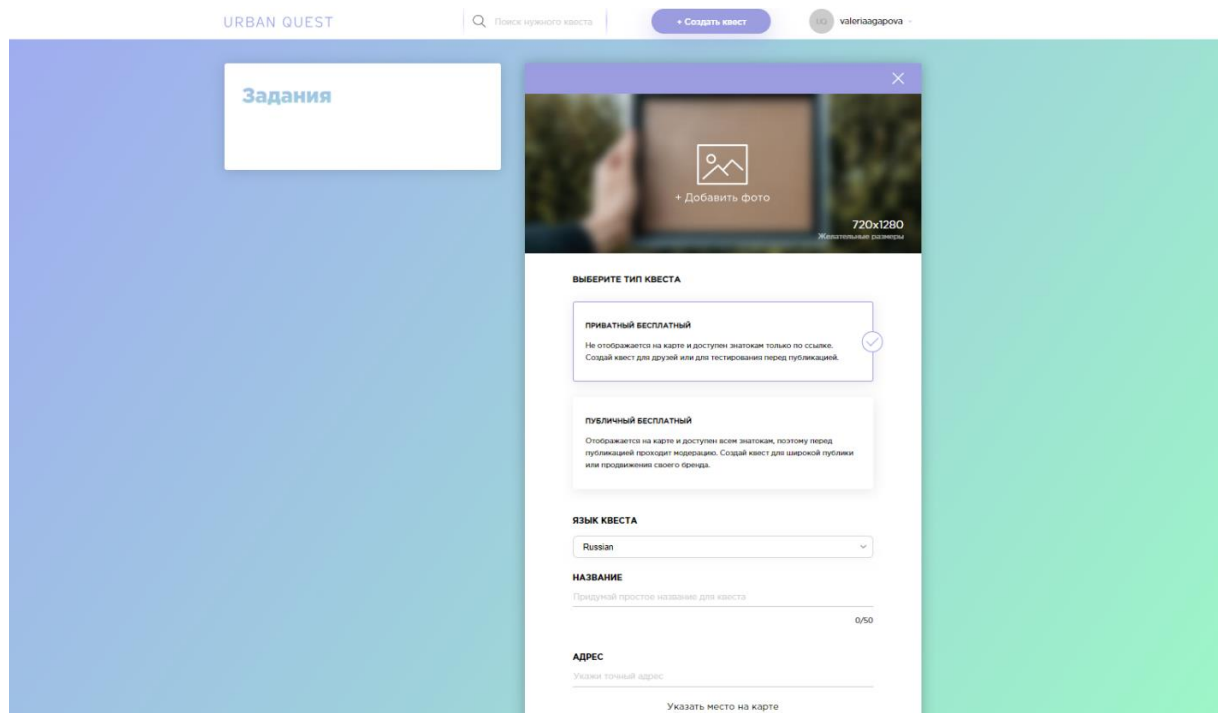


Рисунок 6 – Добавление информации о квесте Urban Quest

Экран добавления задания квеста представлен на рисунке 7. При добавлении задания доступен выбор типа задания, одним из которых является задание с механикой сканирования QR-кода. Также создателю квеста доступно добавление изображения для каждого задания. Привязка задания к геопозиции является опциональной.

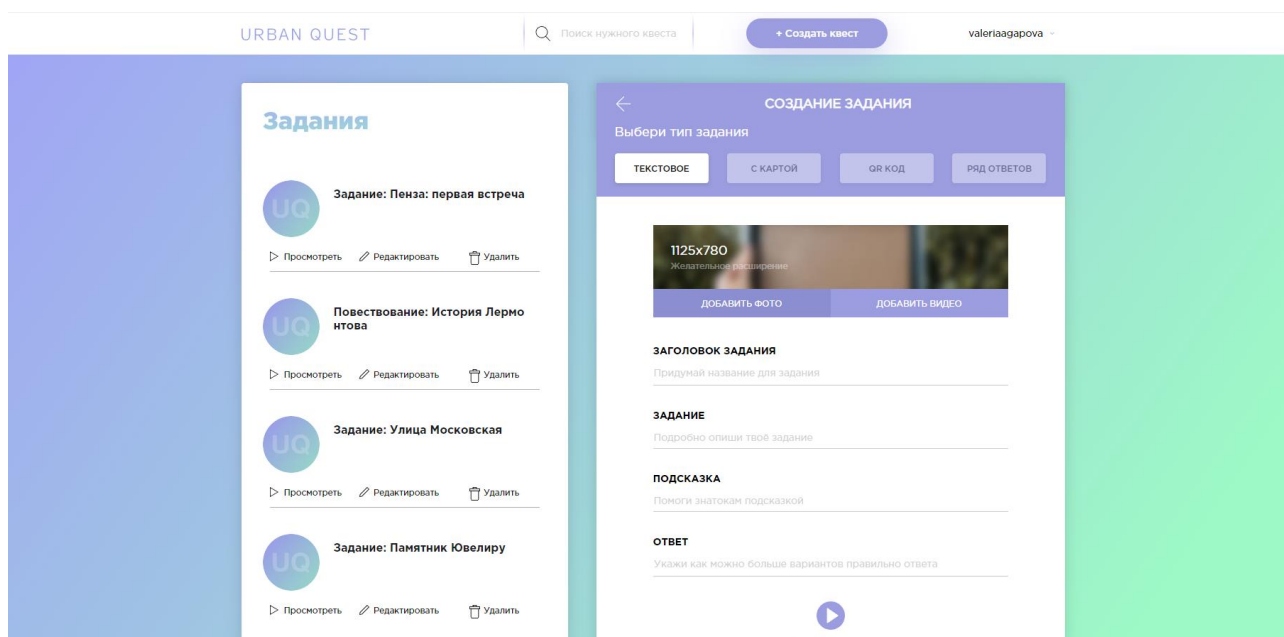


Рисунок 7 – Экран добавления задания Urban Quest

Выбрать количество баллов, начисляемых за правильный ответ нельзя. Отслеживание статистики прохождения квеста для создателя не предоставляется. Критичных сбоев в работе web-приложения для создания квестов обнаружено не было.

2.1.2.4 Результат проведенного анализа

Для обобщения проведенного анализа аналогов была составлена сводная таблица по ключевым характеристикам систем для создания городских квест-экскурсий (таблица 6).

Таблица 6 – Ключевые характеристики аналогичных систем для создания городских квест-экскурсий

Характеристика	Ubego	SurpriseMe	Urban Quest
Выбор города проведения квеста	Есть	Нет	Есть
Выбор типа задания	Нет	Есть	Есть
Добавление изображений к каждому заданию	Нет	Есть	Есть
Добавление заданий с AR	Нет	Есть	Нет (доступны задания на сканирование QR-кодов)
Добавление задания в привязке к геопозиции	Обязательно	Опционально	Опционально
Добавление баллов за прохождение задания	Нет	Есть	Нет
Отслеживание статистики	Нет	Нет	Нет
Качество технической реализации	Наличие сбоев в работе приложения	Редкие сбои в неосновном функционале	Критичных сбоев не обнаружено

Для создания конкурентоспособного приложения необходимо учесть ключевые преимущества аналогов (возможность выбрать город проведения квеста, выбрать тип задания, добавить изображения к каждому заданию, добавить задания с использованием технологии дополненной реальности, указать баллы за прохождение задания и др.), избежать технических

неисправностей, а также реализовать возможность отслеживать статистику прохождения квестов создателями квестов и управлять пользователями системы.

2.1.3 Анализ требований к программным средствам

2.1.3.1 Анализ функциональных требований

Разрабатываемое web-приложение для конструирования квестов и анализа статистики должно предоставлять пользователям следующий функционал:

- управлять историческими квестами (добавлять, удалять, редактировать);
- управлять пользователями (редактировать данные пользователей, изменять роли, удалять пользователей);
- просматривать статистику квестов (средняя дистанция, среднее количество заданий, количество добавленных квестов (с возможностью отслеживания динамики по дням/неделям/месяцам/ в виде графика), общее количество квестов (с возможностью отслеживания динамики по дням/неделям/месяцам в виде графика), количество прохождений квеста (с возможностью отслеживания динамики по дням/неделям/месяцам в виде графика));
- просматривать статистику пользователей (регистрации пользователей (с возможностью отслеживания динамики по дням/неделям/месяцам в виде графика), общее количество пользователей (с возможностью отслеживания динамики по дням/неделям/месяцам в виде графика), распределение пользователей по родным городам).

Доступ к функционалу web-приложение для конструирования квестов и анализа статистики должны иметь только авторизованные пользователи с ролями «Администратор» и «Создатель квестов». Администратору доступен полный функционал приложения, создатель квеста имеет доступ ко всему функционалу, кроме управления пользователями.

Для наглядного описания поведения разрабатываемого web-приложения была составлена диаграмма вариантов использования (рисунок 8), состоящая из актеров и выполняемых им прецедентов.



Рисунок 8 – Диаграмма вариантов использования

Для двух вариантов использования были описаны их спецификации. Спецификация прецедента «Удалить квест» представлена на рисунке 9.

Прецедент: Удалить квест
ID: 1
Краткое описание: Пользователь удаляет из системы выбранный квест
Главные актеры: Пользователь (Администратор или Создатель квестов).
Второстепенные актеры: Нет.
Предусловия: 1. Пользователь авторизован в системе 2. Пользователь открыл страницу «Квесты»
Основной поток: 1. Пользователь выбирает квест, который необходимо удалить. 2. Пользователь нажимает кнопку «Удалить» 3. Система отправляет HTTP-запрос на сервер для удаления 4. Система возвращает ответ с сервера
Постусловия: 1. Квест удален или обработана ошибка
Альтернативные потоки: Нет.

Рисунок 9 – Спецификация прецедента «Удалить квест»

Спецификация прецедента «Редактировать роль пользователя» представлена на рисунке 10.

Прецедент: Редактировать роль пользователя
ID: 2
Краткое описание: Администратор изменяет роль выбранного пользователя
Главные актеры: Администратор.
Второстепенные актеры: Нет.
Предусловия: 1. Администратор авторизован в системе 2. Администратор открыл страницу «Пользователи»
Основной поток: 1. Администратор выбирает пользователя, для которого необходимо изменить роль 2. Администратор нажимает кнопку «Изменить роль» 3. Администратор выбирает новую роль из выпадающего списка 4. Администратор нажимает кнопку «Сохранить» 5. Система отправляет HTTP-запрос на сервер для изменения роли пользователя 6. Система возвращает ответ с сервера
Постусловия: 1. Роль изменена или обработана ошибка
Альтернативные потоки: Нет.

Рисунок 10 – Спецификация прецедента «Редактировать роль пользователя»

В результате анализа функциональных требований были выявлены и описаны прецеденты системы, а также построена диаграмма вариантов использования.

2.1.3.2 Анализ нефункциональных требований

К разрабатываемому web-приложению были выдвинуты следующие нефункциональные требования:

- простота и понятность пользовательского интерфейса;
- наличие обратной связи при взаимодействии пользователя с элементами системы;
- однородность элементов, выполняющих схожие функции;
- схожий графический стиль всех элементов;
- корректное отображение элементов в различных браузерах;
- возможность масштабирования при расширении спектра бизнес-задач в перспективе.

Системные требования для функционирования клиентской части:

- web-браузер, поддерживающий ES6 (Chrome 51+, Edge 14+, Safari 10+, Firefox 54+, Opera 38+ и др.);
- объем оперативной памяти (RAM): 512 Мб (для 32-х битной версии ОС) или 2 Гб (для 64-х битной версии ОС);
- минимальный объём дискового пространства – 350 Мб.

К оборудованию для развертывания серверной части были выдвинуты следующие нефункциональные требования:

- наличие установленного Docker для развёртывания приложения;
- процессор с минимальной частотой 1 ГГц;
- минимум 512 Мб оперативной памяти;
- минимальный объём дискового пространства – 6 Гб;
- наличие интернет-соединения со скоростью не менее 100 Мбит/с.

2.1.4 Постановка цели и задачи

Целью индивидуальной части выпускной квалификационной работы как стартап является разработка web-приложения для конструирования квестов и анализа статистики.

Для достижения поставленной цели необходимо выполнить следующие задачи:

- проанализировать предметную область, выявить функциональные и нефункциональные требования к системе;
- спроектировать архитектуру серверной части web-приложения и модель базы данных;
- спроектировать клиентскую часть web-приложения и пользовательский интерфейс;
- провести планирование разработки и оценку бюджета;
- реализовать серверную часть web-приложения и базу данных;
- реализовать клиентскую часть web-приложения и пользовательский интерфейс;
- провести тестирование и оценить качество кода с использованием различных метрик.

Назначением разрабатываемого web-приложения является облегчение деятельности создателей квестов и администраторов системы. Система должна позволять управлять квест-экскурсиями, пользователями, AR-таргетами и городами, а также отслеживать статистику прохождения квестов и статистику пользователей.

При решении поставленных задач важно учитывать привлекательность, простоту и легкость взаимодействия пользователя с интерфейсом приложения, а также предусмотреть возможность масштабирования приложения для добавления различного функционала при росте требований бизнеса.

Необходимо реализовать возможность при создании квест-экскурсии добавлять в задания AR-таргеты для предоставления пользователям

возможности проходить квест-экскурсий с использованием технологии дополненной реальности для повышения уровня вовлеченности клиентов в процесс.

2.2 Проектирование программных средств для проведения квестов с применением технологии дополненной реальности

Анализ предметной области и бизнес-требований позволил установить, что система для проведения исторических квест-экскурсий должна представлять собой программный комплекс, обеспечивающий возможность не только прохождения квестов, но и их формирования, а также поддержания прочих административных функций. Из этого следует, что программный комплекс должен иметь как минимум два программных компонента, оперирующих общими данными:

- мобильное приложение для проведения квестов с применением технологии дополненной реальности;
- приложение администратора (web-приложение для конструирования квестов и анализа статистики).

Исходя из описанной особенности на этапе проектирования всего программного комплекса было принято решение применить трехуровневую клиент-серверную архитектуру. Данная архитектурная концепция характеризуется выделением сервера базы данных, сервера приложений и взаимодействующих с ним клиентских приложений [15], что позволяет ей обладать хорошей масштабируемостью, гибкостью, конфигурируемостью, а также иметь высокую надежность, безопасность и низкие требования к техническим характеристикам клиентских устройств [16].

Обобщенное распределение компонентов программного комплекса, организованного по принципу клиент-серверной архитектуры, описано с помощью диаграммы развертывания, приведенной на рисунке 11.

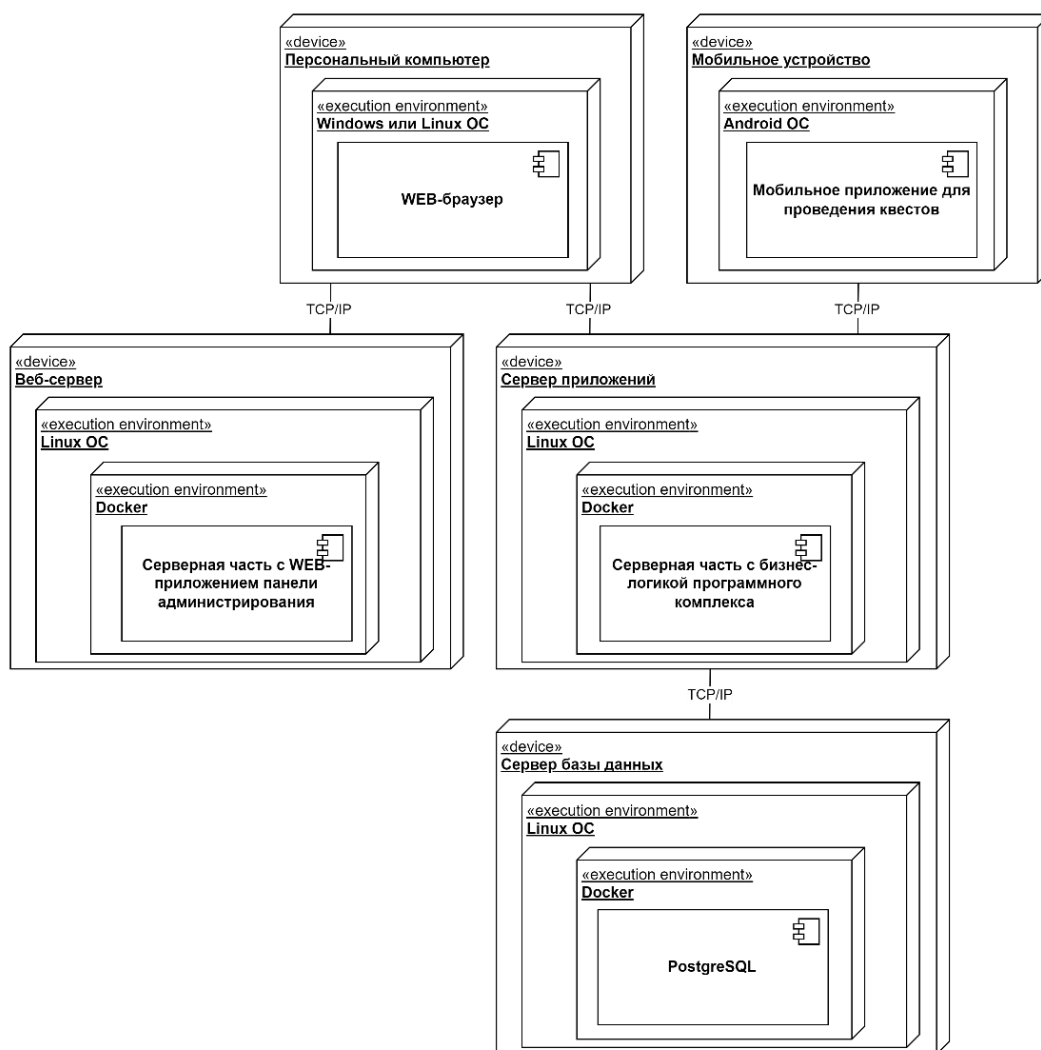


Рисунок 11 – Диаграмма развертывания программного комплекса

В рамках данной выпускной квалификационной работы производилась разработка лишь web-приложения для конструирования квестов и анализа статистики и части серверного программного обеспечения (ПО), необходимой для его функционирования.

2.2.1 Проектирование архитектуры серверной части

2.2.1.1 Выбор архитектуры серверной части

Первым этапом проектирования серверной части стал выбор архитектурного подхода, на котором должно базироваться приложение. При выборе архитектуры необходимо учесть набор факторов, обусловленных особенностями предметной области и спецификой проекта. Разработка

программного комплекса с целью создания стартап-проекта предполагает возможность быстрого роста числа пользователей, что влечет за собой необходимость быстрого и легкого масштабирования. Кроме того, при развитии проекта планируется постоянное внедрение большого количества нового функционала. Также стоит учитывать устойчивость к сбоям и сложность инфраструктурного окружения программного обеспечения. Наконец, важными факторами будут являться скорость и стоимость разработки.

Правильный выбор архитектуры серверной части приложения позволит найти оптимальное соотношение между качеством программного кода и затраченными на разработку ресурсами. Для выбора архитектуры воспользуемся методами теории принятия решений. Для этого введем кодовые обозначения сравниваемых архитектур (таблица 7) и опишем критерии для их сравнения (таблица 8).

Таблица 7 – Кодовые обозначения архитектур

Архитектура	Кодовое обозначение
Монолитная архитектура	B1
Архитектура на основе сервисов	B2
Микросервисная архитектура (MSA)	B3

Таблица 8 – Описание критериев сравнения архитектур

Критерий	Код критерия	Описание
Масштабируемость	K1	Работоспособность системы при росте числа пользователей или ужесточении уровня технических требований [17]. В том числе при возможности наращивания дополнительных ресурсов без структурных изменений главных компонентов серверной части программного комплекса
Поддерживаемость	K2	Потенциал простоты последующей модификации и внедрения нового функционала
Отказоустойчивость	K3	Способность сохранять надежность при отказе отдельных компонентов или сбоях в подсистемах [18]
Производительность	K4	Скорость и эффективность выполнения запросов
Развертываемость	K5	Простота и скорость выполнения действий, которые делают программную систему готовой к использованию
Тестируемость	K6	Мера того, насколько просто создать критерий проверки для системы и ее компонентов и выполнить эти тесты [19]
Общие затраты	K7	Уровень временных и экономических затрат, необходимых для разработки серверной части программного комплекса

Для выполнения сравнения качественных критериев необходимо перевести их значения в количественные показатели. Для этого введем шкалы количественных показателей для каждого из критериев (таблицы 9 – 15).

Таблица 9 – Вербально-числовая шкала для критерия К1

Критерий	Числовой эквивалент критерия
Очень высокая масштабируемость. Отсутствует зависимость между компонентами при масштабировании	1,0
Высокая масштабируемость. Присутствует зависимость между некоторыми компонентами при масштабировании	0,9
Низкая масштабируемость. Присутствует зависимость между многими компонентами при масштабировании	0,8
Очень низкая масштабируемость. Необходимо масштабировать все компоненты вместе	0,7

Таблица 10 – Вербально-числовая шкала для критерия К2

Критерий	Числовой эквивалент критерия
Очень быстрое внедрение нового и изменение существующего функционала	1,0
Быстрое внедрение нового и изменение существующего функционала	0,9
Требуется время для внедрения нового и изменение существующего функционала	0,8
Очень долгое внедрение нового и изменение существующего функционала	0,7

Таблица 11 – Вербально-числовая шкала для критерия К3

Критерий	Числовой эквивалент критерия
Возникновение сбоя локально и не влияет на работоспособность системы	1,0
Возникновение сбоя локально и влияет лишь на работоспособность конкретного компонента системы	0,9
Возникновение сбоя локально и влияет на работоспособность компонента системы и зависящих от него компонентов. Остальная часть системы остается работоспособной	0,8
Возникновение сбоя приводит к общесистемному сбою	0,7

Таблица 12 – Вербально-числовая шкала для критерия К4

Критерий	Числовой эквивалент критерия
Очень высокая производительность	1,0
Высокая производительность	0,9
Средняя производительность	0,8
Низкая производительность	0,7

Таблица 13 – Вербально-числовая шкала для критерия К5

Критерий	Числовой эквивалент критерия
Процедура развертывания обновленных компонентов не затрагивает работоспособность системы	1,0
Процедура развертывания обновленных компонентов выполняется при остановке обновляемых компонентов	0,9
Процедура развертывания обновленных компонентов выполняется при остановке обновляемых компонентов и компонентов, зависящих от них	0,8
Процедура развертывания обновленных компонентов выполняется при полной остановке системы	0,7

Таблица 14 – Вербально-числовая шкала для критерия К6

Критерий	Числовой эквивалент критерия
Максимально полная тестируемость	1,0
Почти полная тестируемость	0,9
Частичная тестируемость	0,8
Плохо поддается тестированию	0,7

Таблица 15 – Вербально-числовая шкала для критерия К7

Критерий	Числовой эквивалент критерия
Очень низкая стоимость и длительность разработки	1,0
Низкая стоимость и длительность разработки	0,9
Высокая стоимость и длительность разработки	0,8
Очень высокая стоимость и длительность разработки	0,7

Количественные показатели критериев для каждого из вариантов архитектуры на основе введенных вербально-числовых шкал перевода приведены в таблицу 16. Оценка критериев производилась на основе субъективного опыта и знаний, а также на основе оценок, данных архитектурам в книге «Фундаментальный подход к программной архитектуре» [17].

Таблица 16 – Количественные показатели критериев выбора архитектуры

Код критерия	Критерий	Значение критериев вариантов архитектуры		
		В1	В2	В3
K1	Масштабируемость	0,7	0,85	1,0
K2	Поддерживаемость	0,7	0,85	0,95
K3	Отказоустойчивость	0,7	0,8	0,85
K4	Производительность	0,8	0,9	0,75
K5	Развертываемость	0,7	0,8	0,9
K6	Тестируемость	0,9	0,9	0,8
K7	Общие затраты	0,95	0,9	0,7

При определении лучшего варианта архитектуры был использован метод ранжирования Борда [20]. Для этого составлена таблица рангов (таблица 17), содержащая ранги по всем критериям, зависящие от оценок всех вариантов по тому или иному критерию. Был проведен подсчет суммы рангов по формуле:

$$R_j = \sum_{i=1}^n r_{ij},$$

где R_j – сумма рангов j -того варианта;

r_{ij} – ранг i -того критерия j -того варианта;

n – количество критериев.

Таблица 17 – Ранжирование вариантов

Код критерия	Ранг критериев		
	B1	B2	B3
K1	3	2	1
K2	3	2	1
K3	3	2	1
K4	2	1	3
K5	3	2	1
K6	1,5	1,5	3
K7	1	2	3
Итого	16,5	12,5	13

Для определения лучшего варианта была применена формула:

$$R_l = \min [R_1, R_m],$$

где R_l – лучший вариант;

R_m – сумма рангов m -ого варианта;

m – количество вариантов.

Результатом применения ранжирования вариантов по методу Борда стало расположение приоритетности выбора архитектур в следующем порядке: B2 > B3 > B1. Лучшей архитектурой для выбранного набора критериев является архитектура на основе сервисов под кодовым обозначением B2. Именно эта архитектура позволит соблюсти оптимальное соотношение между качеством программного кода и затраченными на разработку ресурсами с учетом особенностей проекта. Стоит отметить, что при изменении приоритетов в формировании набора критериев выбора, их анализ может привести к другому наиболее подходящему под особенности типу архитектуры. Кроме того, с учетом погрешности, разница между рангами вариантов B2 и B3 позволяет сделать

выбор и в пользу микросервисной архитектуры. Однако с учетом ограниченности бюджета проекта и сроков разработки был выбран вариант архитектуры на основе сервисов.

2.2.1.2 Разбиение серверной части на сервисы

Типовая топология архитектуры на основе сервисов – это распределенная макроуровневая структура, состоящая из отдельно развернутого пользовательского интерфейса, отдельно развернутых удаленных крупномодульных сервисов и монолитной базы данных (рисунок 12).

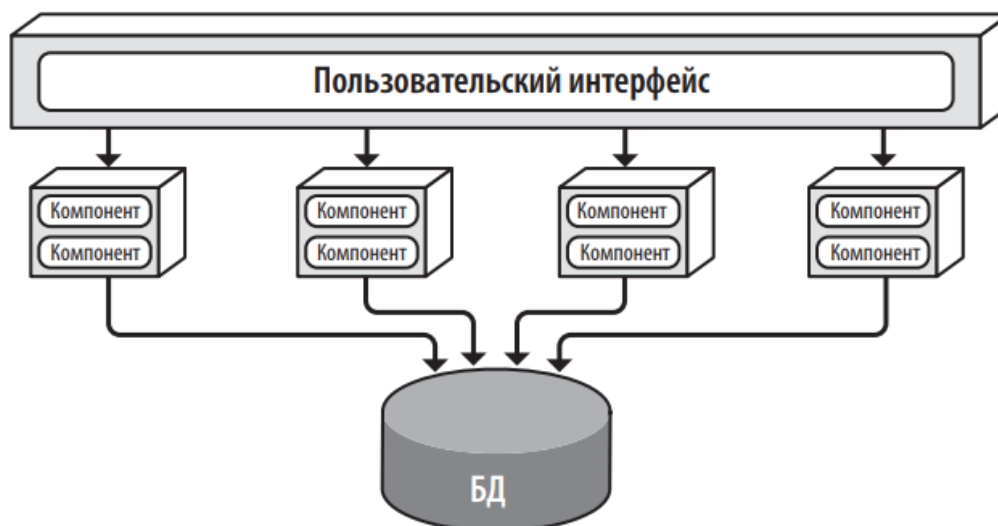


Рисунок 12 – Типовая топология архитектуры на основе сервисов

Сервисы в этом архитектурном стиле представлены, как правило, независимыми и отдельно развертываемыми крупномодульными «порциями приложения», которые обычно называют сервисами предметной области (domain services). Поскольку сервисы обычно совместно используют единую монолитную базу данных, их число в контексте приложения чаще всего варьируется от 4 до 12 единиц, то есть в среднем используется около 7 сервисов [17].

При проектировании серверной части для разрабатываемого web-приложения для конструирования квестов и анализа статистики было принято

решение выделить 4 сервиса, наименование и назначение которых приведено в таблице 18.

Таблица 18 – Сервисы серверной части

Наименование сервиса	Назначение сервиса
migrator	Инициализация базы данных и выполнение миграций
questie_backend	Обработка запросов, связанных с квестами и квестовыми заданиями
questie_backend_authorization	Обработка запросов на генерацию JSON Web Token (JWT)
questie_backend_users	Обработка запросов, связанных с пользователями

Наглядное представление распределения компонентов программного комплекса по различным узлам, отображающее связь сервисов с базой данных и клиентским web-приложением приведено в виде диаграммы развертывания на рисунке 13.

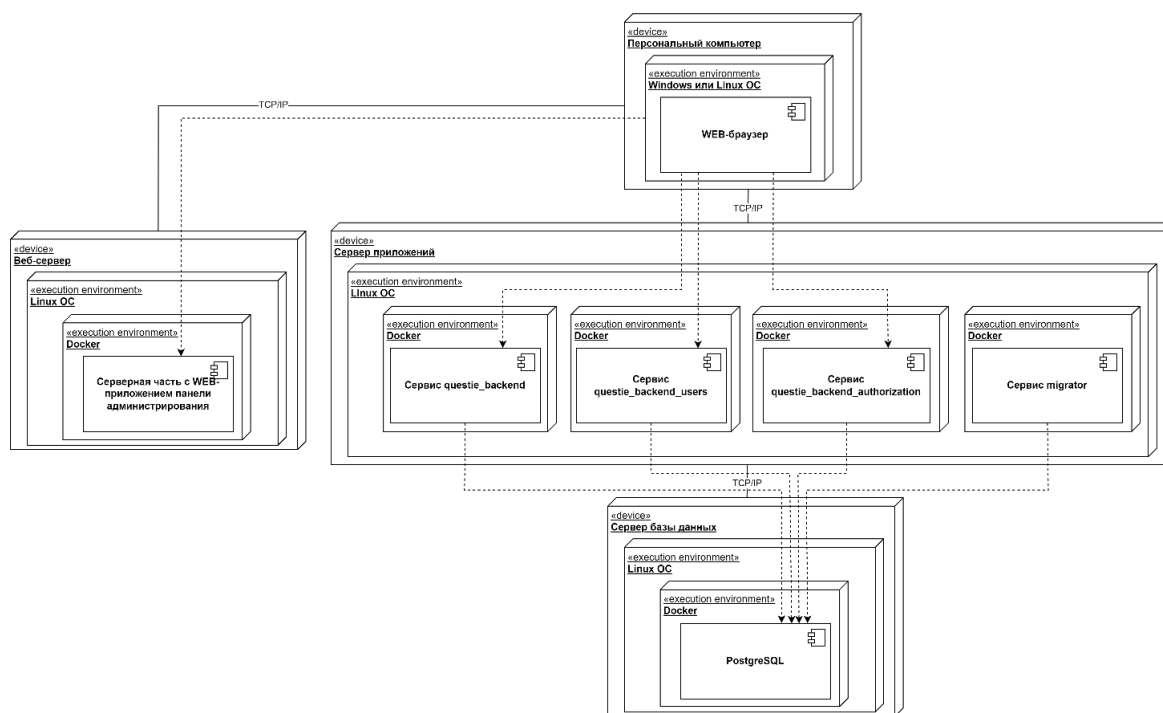


Рисунок 13 – Диаграмма развертывания части программного комплекса, необходимой для функционирования web-приложения

Из диаграммы видно, что все сервисы должны взаимодействовать с базой данных, однако сервис migrator не должен принимать запросы от web-браузера. Это обусловлено его назначением – инициализацией базы данных и выполнением миграций. При развертывании программного комплекса сервис

должен активироваться после развертывания базы данных (БД) и завершать свою работу до запуска остальных сервисов.

На приведенной диаграмме все 4 сервиса расположены на 1 физической машине, однако при необходимости увеличения вычислительных мощностей они могут быть разнесены по разным физическим машинам, что не вызовет затруднений благодаря использованию виртуализированной среды выполнения контейнеров Docker. При необходимости возможен и запуск нескольких экземпляров сервисов, обрабатывающих запросы. Однако в таком случае потребуется внедрение в серверную часть балансировщика нагрузки.

Взаимодействие между web-приложением и сервисами должно происходить на основе стека TCP/IP с применением протокола HTTPS на прикладном уровне и применением подхода REST API для описания интерфейсов взаимодействия.

2.2.1.3 Проектирование последовательности выполнения запросов

Обработка поступающих к сервису запросов в ASP.NET Core происходит по принципу конвейера, состоящего из набора последовательных компонентов, называемых промежуточным или связующим программным обеспечением (middleware) [21].

Обработка запросов к сервису `questie_backend` начинается с работы промежуточного ПО, отвечающего за настройку CORS для обеспечения поддержки кроссдоменных запросов. После чего в случае запуска сервиса в режиме разработки вызывается настройка инструмента Swagger и генерация графического интерфейса с документацией API, поддерживающей спецификацию OpenAPI. Далее управление передается в промежуточное ПО `StaticFiles`, которое предоставляет поддержку обработки статических файлов. Затем происходит настройка поддержки сессий, аутентификации, авторизации, вызов промежуточного ПО для обработки JWT и, наконец, сопоставление URL-адресов входящих запросов и действий по их обработке. После чего управление передается в соответствующий метод контроллера, где происходит обработка

запроса, не детализированная на диаграмме (вызов сервисных методов, выполняющих основную логику и при необходимости обращающихся к методам работы с данными для получения информации из базы данных).

Для наглядного представления последовательности обработки запросов была использована UML-диаграмма последовательности (рисунок 14), описывающая общий вид обработки запросов к сервису `questie_backend` в режиме Release.

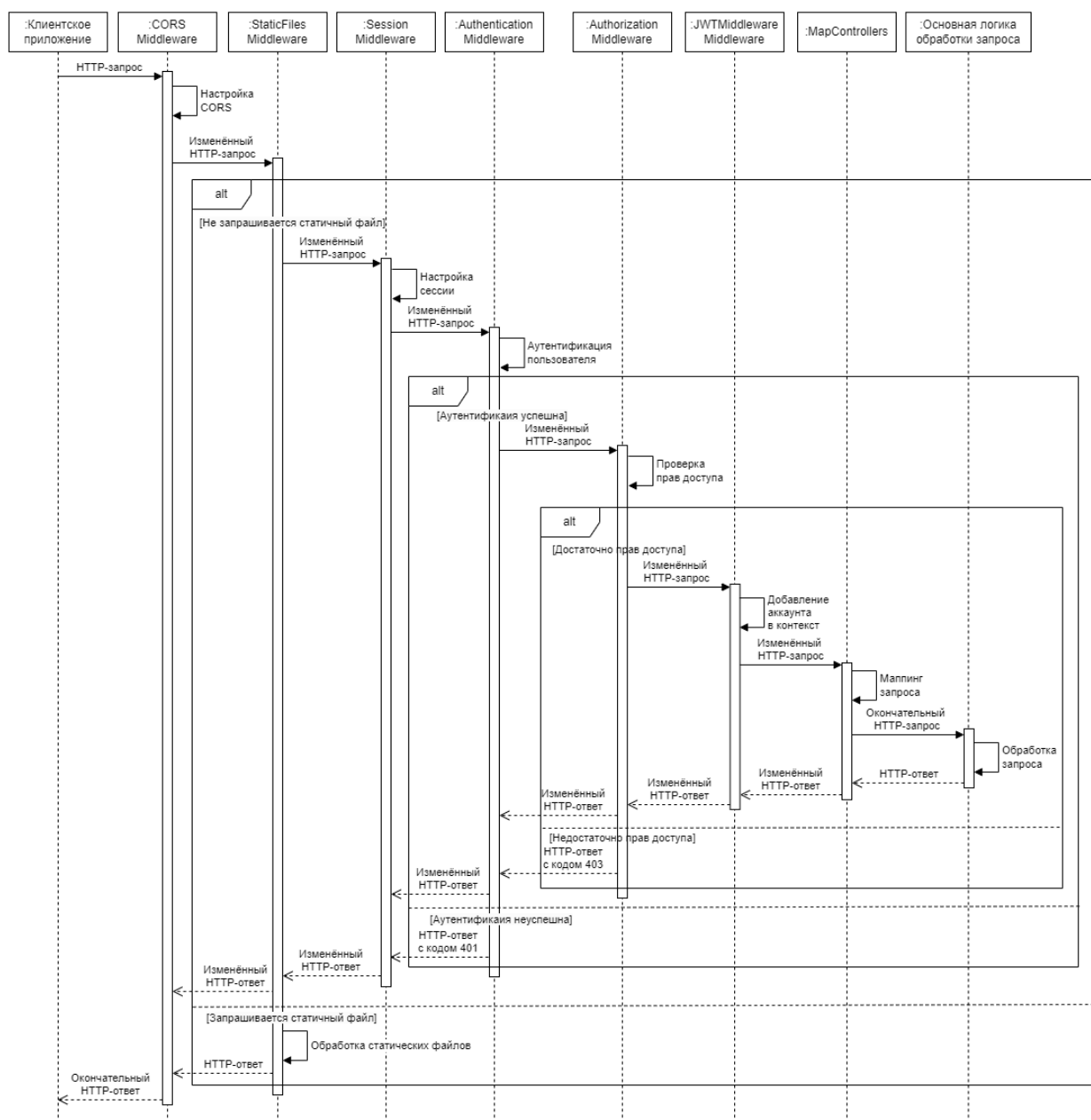


Рисунок 14 – Диаграмма последовательности запросов к сервису `questie_backend`

Обработка запроса к сервису `questie_backend_users` должна происходить аналогично, за исключением отсутствия промежуточного ПО `StaticFiles`.

Конвейер обработки запросов к сервису `questie_backend_authorization` также лишен промежуточного ПО `Authorization` и `JWTMiddleware`.

2.2.1.4 Проектирование выполнения основной логики обработки запросов

Поскольку сервисы предметной области в архитектуре на основе сервисов обычно крупномодульные, каждый из них разрабатывается с использованием многоуровневого стиля архитектуры, состоящего из уровня внешнего API-интерфейса, уровня бизнес-логики и уровня сохранения данных [17].

При проектировании сервисов серверной части программного комплекса для проведения квестов было принято решение также применить многоуровневый стиль архитектуры с техническим разбиением на слои.

Для проектирования обработки запросов, касающихся пользователей, был взят за основу шаблон проектирования «Стратегия», позволяющий определить семейство алгоритмов, инкапсулировать каждый из них и сделать их взаимозаменяемыми [22].

«Стратегия» применяется дважды (диаграмма классов на примере классов, выполняющих обработку запросов, касающихся пользователей, приведена на рисунке 15). В первом случае она обеспечивает возможность замены реализации сервисных классов и включает в себя следующие элементы:

- интерфейс `IUserService`, общий для всех реализаций сервисного класса;
- класс `UserService`, реализующий интерфейс `IUserService` (подобных классов-реализаций может быть несколько);
- класс `UserController`, хранящий ссылку на объект типа `IUserService` и связанный с интерфейсом `IUserService` отношением агрегации.

Во втором случае, представленном классами `UserService`, `UsersRepository` и интерфейсом `IUsersRepository`, применение данного шаблона проектирования

позволяет использовать различные реализации репозитория, что может быть актуально, например, в случае изменения ORM для взаимодействия с базой данных.

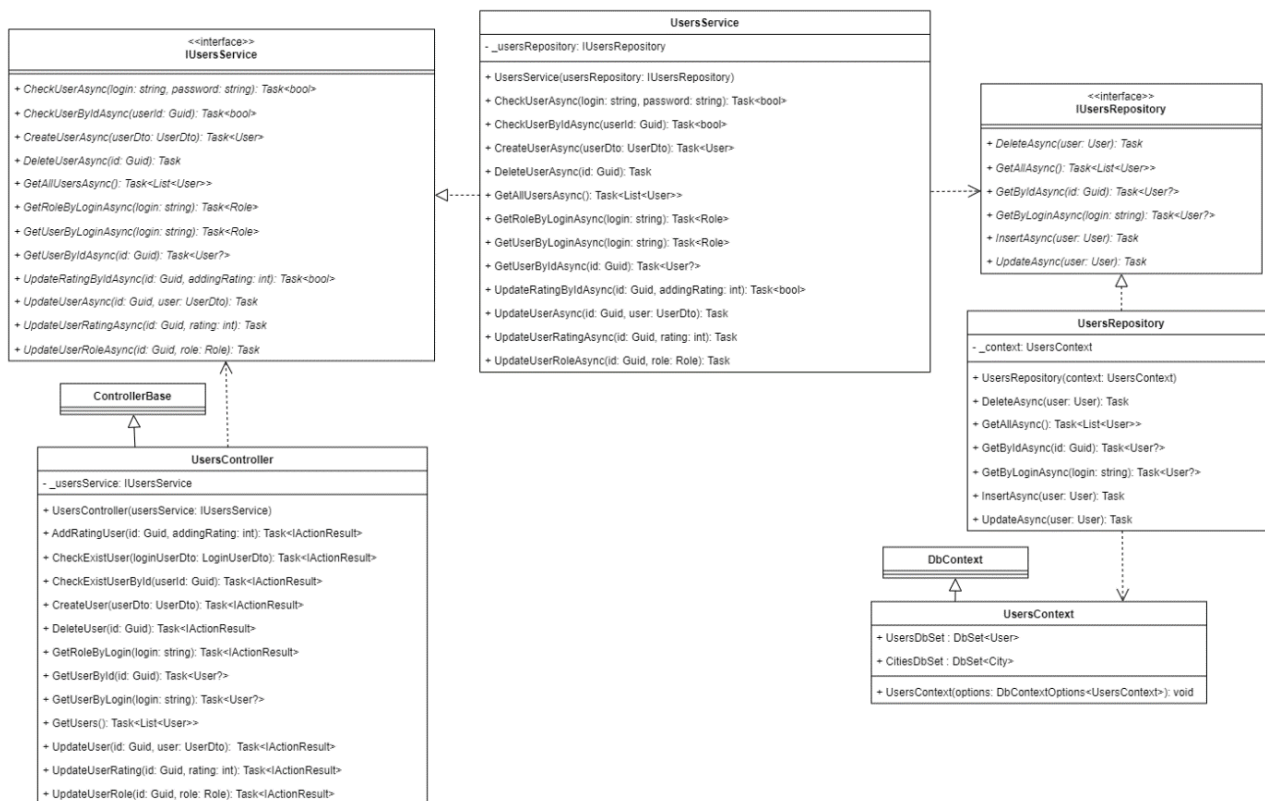


Рисунок 15 – Диаграмма классов, выполняющих обработку запросов, касающихся пользователей

Класс UsersContext предназначен для определения контекста доступа к базе данных и наследуется от базового класса контекста DbContext. Класс ControllerBase является базовым для всех контроллеров.

Связи с классами, описывающими сущности предметной области и объекты передачи данных (Data Transfer Object или DTO), не приведены на диаграмме для большей компактности.

Во избежание зависимости от низкоуровневых деталей и связи с другими конкретными классами был применен шаблон проектирования Dependency Injection или «Внедрение зависимостей» [23].

Внедрение зависимостей применяет принцип IoC (Inversion of Control или инверсия управления), чтобы гарантировать, что класс абсолютно никаким образом не участвует в создании экземпляра зависимого класса, и тем более не управляет его жизненным циклом (lifetime), то есть временем существования объектов [24].

В описываемом примере данный шаблон проектирования применяется для получения доступа к экземплярам классов Сервис, Репозиторий и QuestsContext, имеющих длительность жизненного цикла, ограниченную одним HTTP-запросом.

2.2.2 Проектирование логической модели базы данных

При работе с фреймворками объектно-реляционного отображения (ORM) имеется три подхода к моделированию структуры базы данных: Model-First, Database-First и Code-First [25]. Как правило, при необходимости создания базы данных с нуля используются Model-First или Code-First подход. Первый дает больше наглядности, второй позволяет экономить время разработки. При проектировании структуры базы данных серверной части для web-приложения для конструирования квестов и анализа статистики был выбран Code-First подход.

В Code-First работа начинается с определения модели предметной области, с использованием классов POCO (Plain Old CLR Object), не зависящих от Entity Framework. Code-First позволяет получить основную информацию о модели исключительно из описывающих ее классов. Однако существует возможность предоставить дополнительную конфигурацию для более полного описания модели или переопределить то, что изначально было создано на основе кода [26].

Таким образом, подход Code-First позволяет определять модель данных с помощью классов и свойств, а затем создавать базу данных на основе этой модели. Благодаря этому подходу разработчик может иметь большую гибкость при работе с моделью данных и управлении ее изменениями [27].

Поскольку серверная часть программного комплекса предназначена не только для функционирования web-приложения для конструирования квестов и анализа статистики, разработке которого посвящена данная работа, при проектировании логической модели базы данных была учтена необходимость создания атрибутов сущностей, предназначенных для функционирования мобильного приложения для прохождения исторических квест-экскурсий программного комплекса. Полученная логическая модель базы данных приведена на рисунке 16.

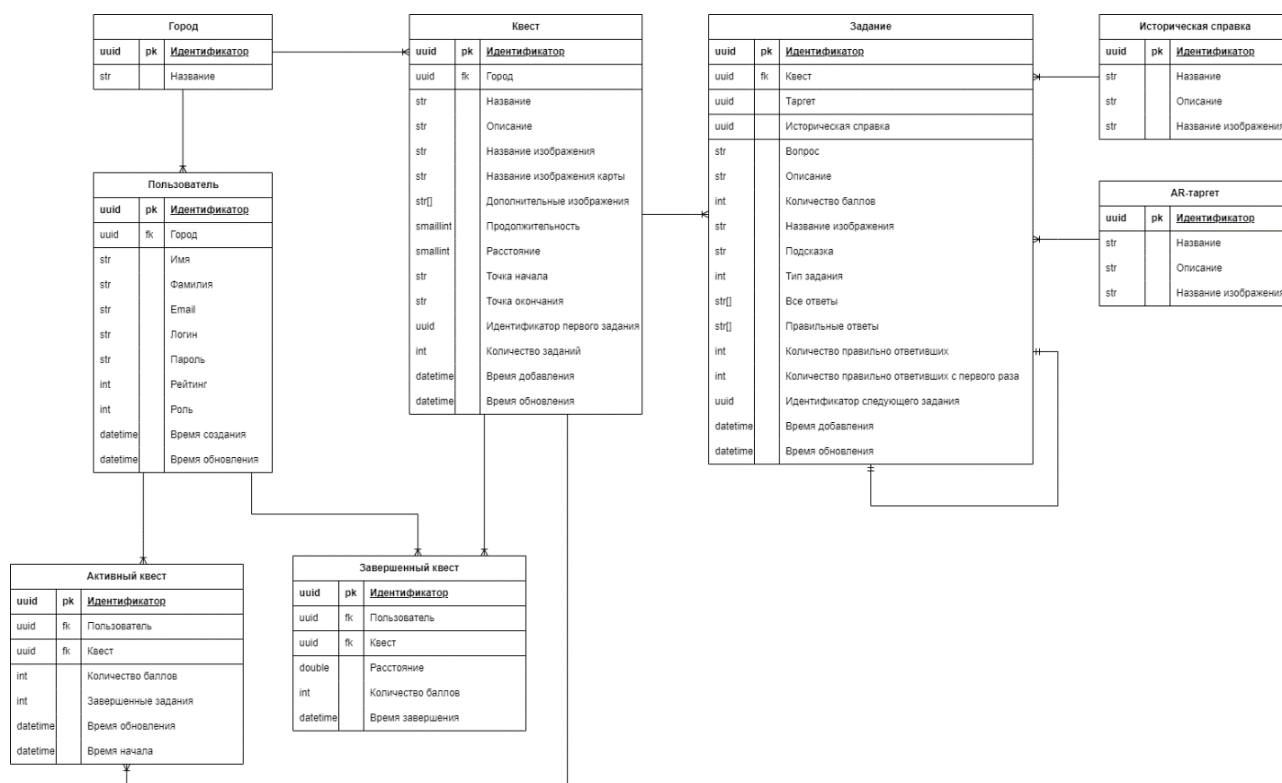


Рисунок 16 – Логическая модель базы данных

Назначение таблиц, необходимых для функционирования web-приложения для конструирования квестов и анализа статистики, разработанной логической модели данных приведено в таблице 19.

Таблица 19 – Таблицы логического уровня модели данных

Наименование сущности	Наименование таблицы	Назначение сущности
Города	CitiestDbSet	Список доступных городов
Завершенные квесты	CompletedQuestsDbSet	Информация о пройденных пользователями квестах (необходима для сбора статистики)
Исторические справки	NotesDbSet	Информация об исторических справках
Квесты	QuestsDbSet	Информация о доступных квестах
Квестовые задания	QuestionsDbSet	Информация о квестовых заданиях и их принадлежности к тем или иным квестам
Таргеты	TargetsDbSet	Информация об AR-таргетах
Пользователи	UsersDbSet	Информация о пользователе программного комплекса

Таким образом была спроектирована логическая модель базы данных, описывающая все необходимые для функционирования приложения сущности предметной области.

2.2.3 Проектирование клиентской части web-приложения для конструирования квестов и анализа статистики

На сегодняшний день в web-разработке существуют два основных архитектурных шаблона: одностраничные приложения, или SPA, и многостраничные приложения, или MPA [28].

С учетом специфики предметной области, а также необходимости легкого масштабирования проекта и необходимости быстрого внедрения нового функционала, наиболее подходящим архитектурным шаблоном для клиентской части web-приложения для конструирования квестов и анализа статистики является Single Page Application (SPA).

Согласно portalу npmtrends.com, наиболее популярным запросом является сравнение Angular-React-Vue. Это доказывает, что в настоящее время преимущественными технологиями для разработки web-приложений являются фреймворки Vue.js и Angular, а также библиотека React. На рисунке 17 приведена статистика, показывающая количество загрузок перечисленных npm-пакетов за последний год [29].

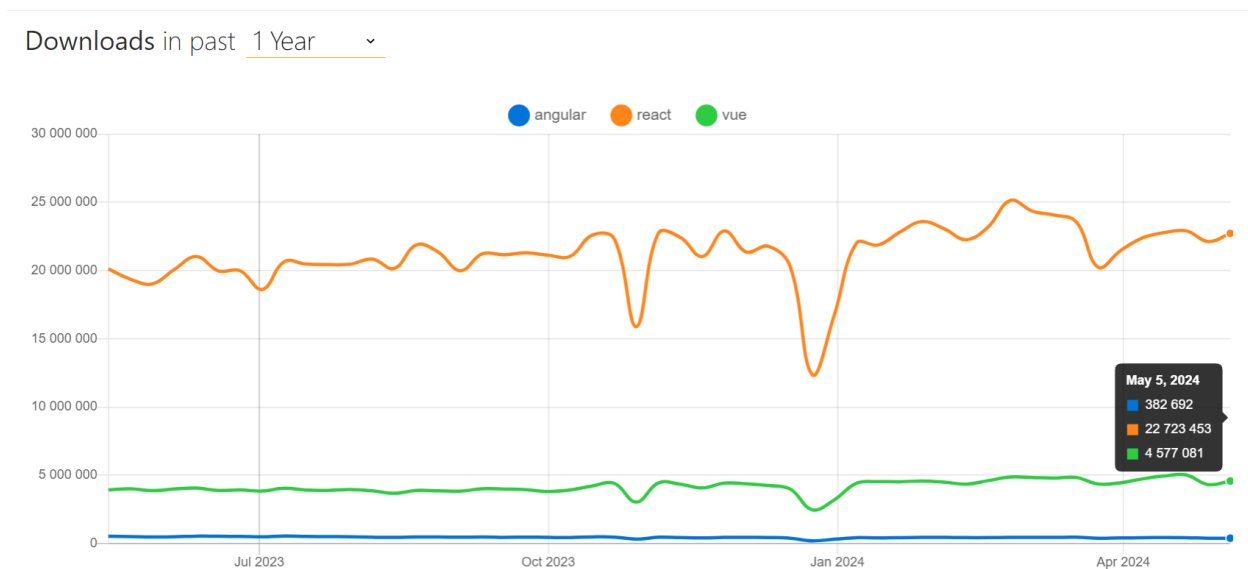


Рисунок 17 – Статистика количества загрузок прм-пакетов

Согласно данной статистике, можно сделать вывод о том, что на сегодняшний день React [30, 31] является наиболее предпочтительной (за счет большого сообщества разработчиков, что упрощает решение возникающих в процессе разработки проблем) технологией разработки клиентской части web-приложений.

Перед реализацией приложения, соответствующего выявленным ранее требованиям, необходимо спроектировать алгоритмы, согласно которым оно будет функционировать. При проектировании алгоритмов необходимо учесть возможные риски, например, некорректный ввод данных.

Для алгоритма добавления пользователя была разработана диаграмма деятельности, представленная на рисунке 18.

Для наглядного представления алгоритма добавления пользователя (функционал доступен только для администратора системы) и отображения динамических аспектов поведения системы была использована UML-диаграмма деятельности [32], представленная на рисунке 18.

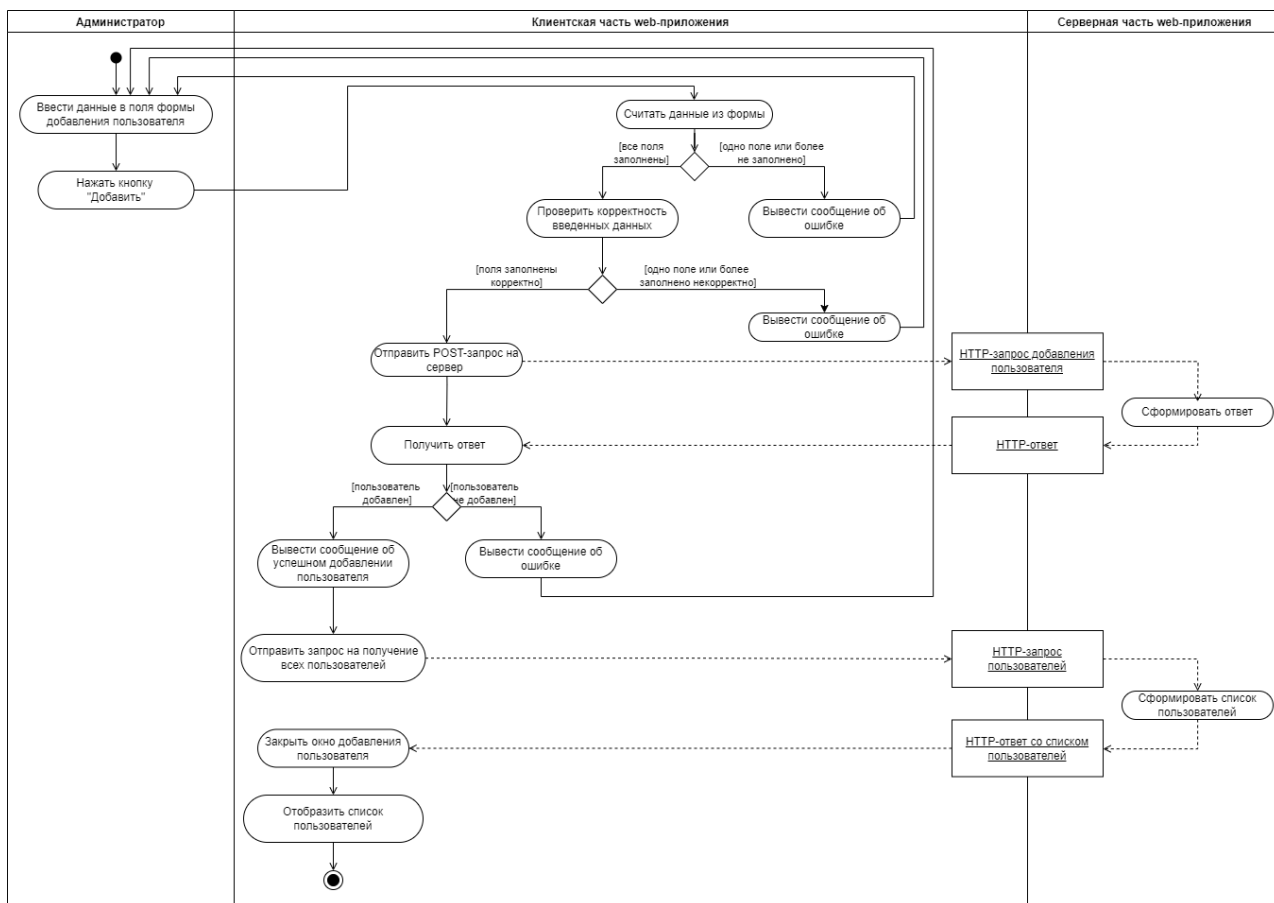


Рисунок 18 – Диаграмма деятельности алгоритма добавления пользователя

Как видно из диаграммы, в алгоритме добавления пользователя считываются данные, введенные пользователем, проверяется заполнение всех полей формы, если заполнены не все поля, то пользователь получает сообщение об ошибке и продолжает заполнение формы. Если все поля заполнены, проверяется корректность введенных данных (происходит сравнение значений с регулярными выражениями). Если какое-либо поле заполнено некорректно, выводится сообщение об ошибке и пользователь исправляет значение, введенное в поле. В случае корректного заполнения всех полей отправляется POST-запрос к серверу – попытка добавления пользователя. На серверной части web-приложения формируется ответ. В случае неудачного выполнения запроса, пользователь получает сообщение об ошибке. В случае удачного выполнения запроса, пользователь получает сообщение об успешном добавлении

пользователя, отправляется GET-запрос на получение всех пользователей, окно добавления пользователя закрывается, отображается список всех пользователей.

Для наглядного представления алгоритма добавления пользователя была использована UML-диаграмма деятельности [32], представленная на рисунке 14.

Для наглядного представления последовательности обработки запросов была использована UML-диаграмма последовательности [33], описывающая алгоритм добавления города, которая приведена на рисунке 19.

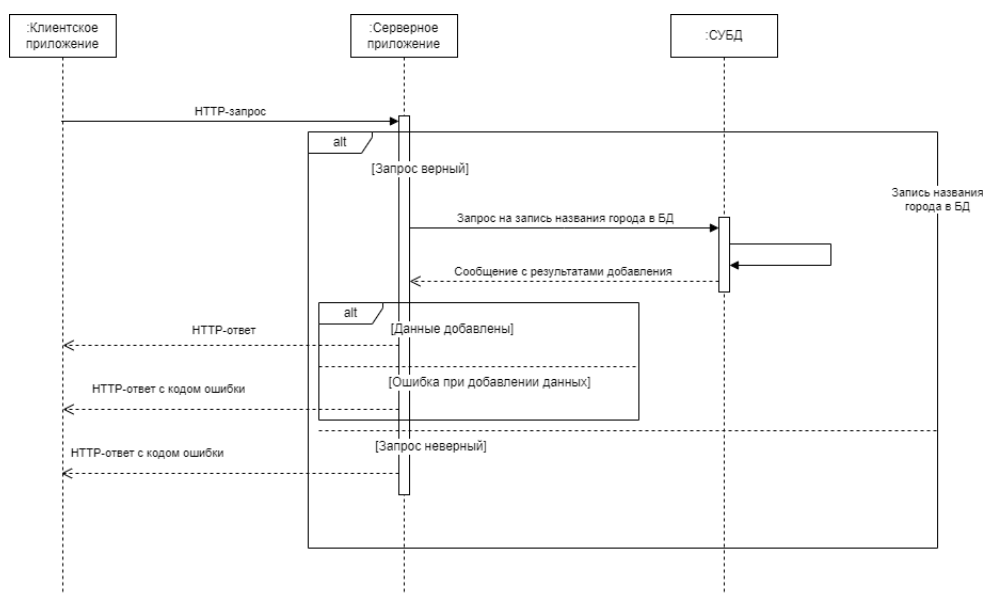


Рисунок 19 – Диаграмма последовательности алгоритма добавления города

Диаграмма описывает процесс добавления города. Она включает в себя три объекта: «Клиентское приложение», «Серверное приложение» и «СУБД», которые взаимодействуют в рамках сценария добавления города и обмениваются сообщениями. На клиенте создается и отправляется HTTP-запрос с названием города, введенным пользователем. Если запрос верный, на сервере формируется и отправляется запрос на запись информации в БД. Там название города сохраняются в базу, формируется и отправляется ответ серверу. Если данные были добавлены, то на клиент отправляется HTTP-ответ. В случае, если при добавлении данных возникла ошибка, то на клиент передается HTTP-ответ с ошибкой. Если HTTP-запрос изначально был неверным, на клиент также возвращается HTTP-ответ с ошибкой.

2.2.4 Проектирование пользовательского интерфейса клиентской части web-приложения для конструирования квестов и анализа статистики

Проектирование пользовательского интерфейса приложения администратора системы для исторических квест-экскурсий (web-приложения для конструирования квестов и анализа статистики) проводилось с использованием графического редактора для разработки интерфейсов и прототипирования Figma.

Руководство по стилю пользовательского интерфейса (UI Style Guide) – это набор правил, установленных для графических элементов, задействованных в web-приложении. В руководстве по стилю отражается типографика, цветовая схема, основные состояния полей ввода, основные компоненты.

Цветовая схема – набор основных оттенков, которые будут использоваться при разработке различных элементов web-приложения. Основной цвет – ярко-фиолетовый: #7A1EFF (RGB 122, 30, 255). Дополнительно в дизайне используются более приглушенные оттенки фиолетового, а также два нейтральных цвета: серый и черный (рисунок 20).



Рисунок 20 – Цветовая схема web-приложения для конструирования квестов и анализа статистики

Основными состояниями полей ввода являются: состояние по умолчанию; активное поле; поле, заполненное с ошибками; успешно заполненное поле; отключенное/недоступное поле (рисунок 21).

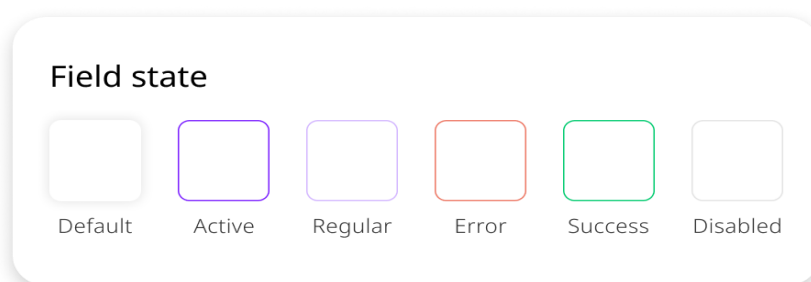


Рисунок 21 – Состояния полей ввода web-приложения для конструирования квестов и анализа статистики

Типографика – все, что связано с оформлением текстовых элементов в дизайне: шрифты, размер, цвет, начертание. Регламентировано написание заголовков и подзаголовков различных уровней, основного текста, подписей, текста кнопок. В дизайне приложения администратора системы для исторических квест-экскурсий был использован шрифт Noto Sans, практичная гарнитура без засечек, облегчающая восприятие большого объема текста (рисунок 22).

Typeface

Noto Sans

Type sizes	Size	Weight			
H1	32px	Medium	Body Large	16px	Regular
H2	28px	Semibold	Body	14px	Regular
H3	24px	Medium	Button	14px	Medium
H4	18px	Medium	Caption 1	12px	Regular
H5	16px	Medium	Caption 2	12px	Semibold
			Caption 3	10px	Regular
			Caption 4	8px	Regular

Рисунок 22 – Типографика web-приложения для конструирования квестов и анализа статистики

Также разработаны основные компоненты, используемые в дизайне web-приложения для конструирования квестов и анализа статистики: поля ввода, кнопки, выпадающие списки, чекбоксы, радиокнопки. Некоторые из них приведены на рисунке 23.

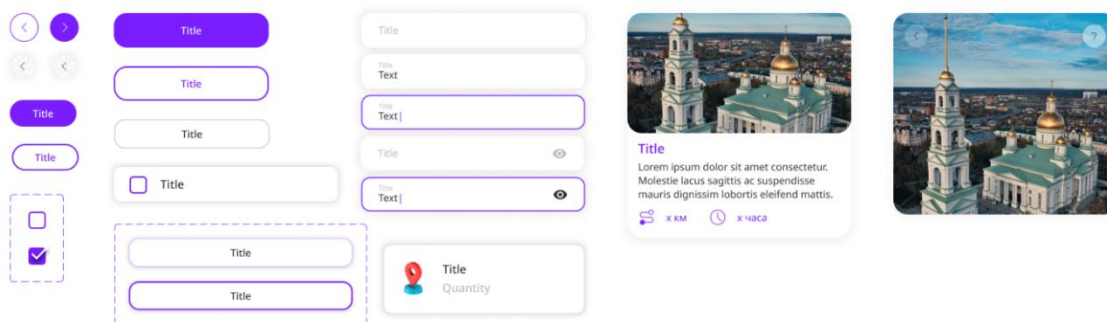


Рисунок 23 – Компоненты, используемые в дизайне web-приложения для конструирования квестов и анализа статистики

В процессе проектирования UI-дизайна используемые в дизайне web-приложения для конструирования квестов и анализа статистики были реализованы следующие основные страницы:

- главная страница;
- страница входа в систему, которая позволяет пользователю войти в уже существующий аккаунт;
- страница статистики, предоставляющая доступ к статистическим данным о пользователях и квестах за определенный период (день, неделя, месяц);
- страница квест-экскурсий, позволяющая просматривать перечень существующих квест-экскурсий, редактировать, просматривать статистику, добавлять новые квест-экскурсии;
- страница добавления квест-экскурсии, предоставляющий возможность добавлять название и текстовое описание квест-экскурсии, загружать изображения, выбирать различные типы заданий;
- страница с подробной информацией о квесте;
- профиль пользователя.

Макеты страниц приложения представлены на рисунке 24.

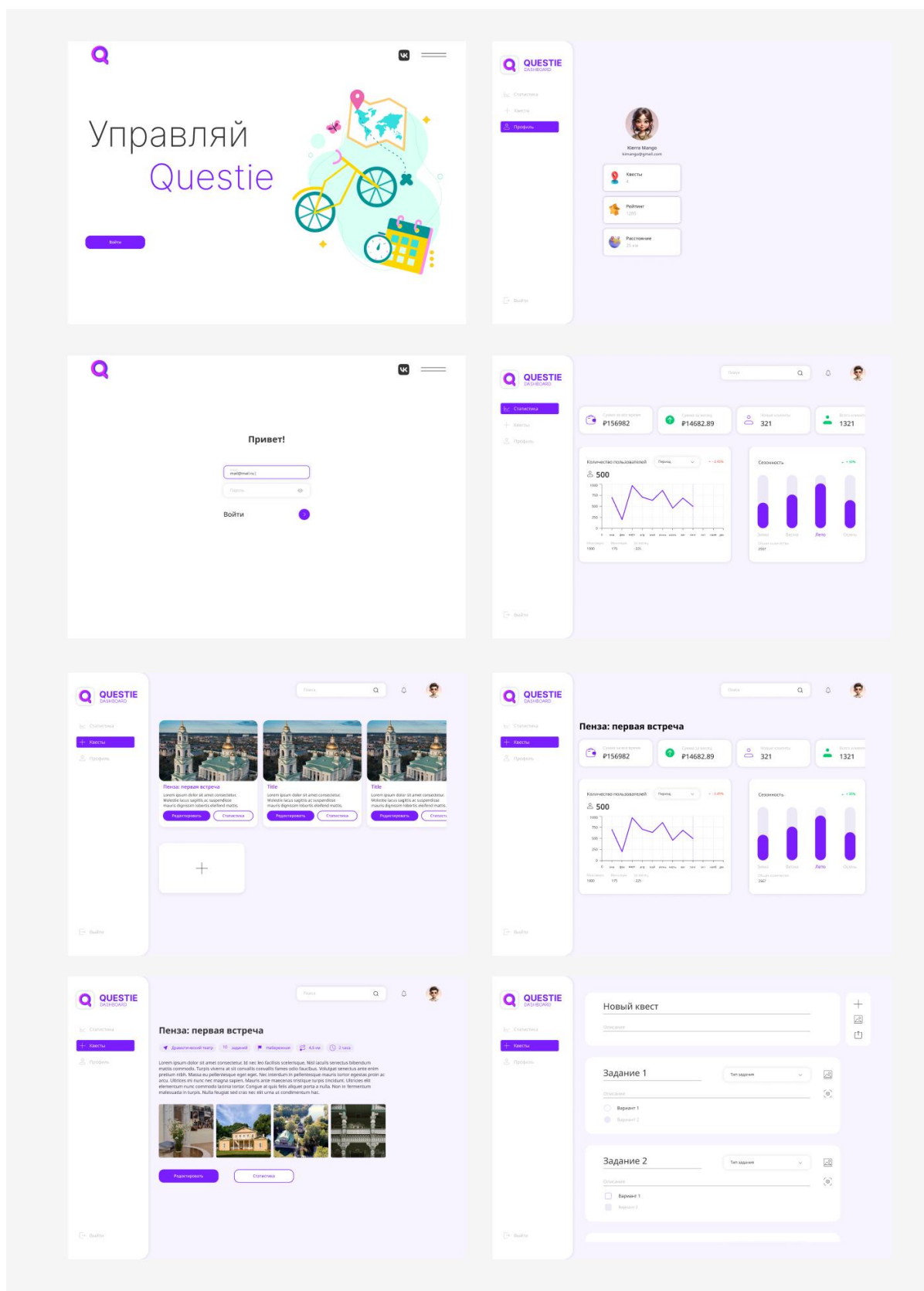


Рисунок 24 – Макеты страниц приложения

В ходе проектирования UI-дизайна административной панели были созданы рабочие функциональные прототипы, дающие возможность оценить удобство сценариев взаимодействия пользователя с системой. В прототипе реализованы все переходы между экранами, кнопки, прокрутки, всплывающие окна.

2.2.5 Планирование разработки и оценка бюджета

Для планирования разработки и оценки бюджета был применен программный продукт Microsoft Project, предназначенный для создания и управления различными проектами.

Для успешной реализации web-приложения для конструирования квестов и анализа статистики необходимо выделить следующие трудовые ресурсы:

- менеджер проекта;
- аналитик;
- проектировщик;
- backend-разработчик;
- frontend-разработчик;
- тестировщик.

Описание всех ресурсов проекта представлено на рисунке 25.

Название ресурса	Тип	Единицы измерения материалов	Макс. единиц	Стандартная ставка	Ставка сверхурочных	Начисление
Менеджер проекта	Трудовой		100%	р. 150,00/ч	р. 300,00/ч	Пропорциональное
Аналитик	Трудовой		100%	р. 150,00/ч	р. 250,00/ч	Пропорциональное
Проектировщик	Трудовой		100%	р. 210,00/ч	р. 400,00/ч	Пропорциональное
Backend-разработчик	Трудовой		100%	р. 170,00/ч	р. 500,00/ч	Пропорциональное
Frontend-разработчик	Трудовой		100%	р. 170,00/ч	р. 500,00/ч	Пропорциональное
Тестировщик	Трудовой		100%	р. 140,00/ч	р. 380,00/ч	Пропорциональное
Компьютер	Затраты					Пропорциональное
Принтер	Затраты					Пропорциональное
Бумага	Затраты					Пропорциональное
Электроэнергия	Материальный	кВт * ч		р. 4,24		По окончании
MS Visual Studio 2022 Professional	Затраты					Пропорциональное
MS Windows 11	Затраты					Пропорциональное
MS Office	Затраты					Пропорциональное

Рисунок 25 – Ресурсы проекта

Основными этапами разработки являются:

- разработка бизнес-плана;
- анализ разработки программных средств;
- проектирование программных средств;
- реализация программных средств;
- тестирование программных средств;
- составление и оформление документации.

Общий список задач проекта с распределением ресурсов по задачам представлен на рисунке 26.

Название задачи	Длительность	Начало	Окончание	Представитель	Названия ресурсов
Начало реализации проекта	127 дней	Пт 01.12.23	Пн 27.05.24		
Закупка оборудования и ПО	1 день	Пт 01.12.23	Пт 01.12.23		Компьютер[p.50 000,00];Принтер[p.8 100,00];MS Office [p.13 750,00];MS Visual Studio 2022 Pro
Разработка бизнес-плана	14 дней	Пн 04.12.23	Чт 21.12.23		
Описание резюме проекта	2 дней	Пн 04.12.23	Вт 05.12.23	2	Менеджер проекта;Электроэнергия[16 кВт * ч]
Описание инновационной идеи	2 дней	Ср 06.12.23	Чт 07.12.23	4	Менеджер проекта;Электроэнергия[16 кВт * ч]
Описание продукции	2 дней	Пт 08.12.23	Пн 11.12.23	5	Менеджер проекта;Электроэнергия[16 кВт * ч]
Анализ рынка	3 дней	Вт 12.12.23	Чт 14.12.23	6	Менеджер проекта;Электроэнергия[24 кВт * ч]
Составление маркетингового плана	3 дней	Пт 15.12.23	Вт 19.12.23	7	Менеджер проекта;Бумага[p.200,00];Электроэнергия[24 кВт * ч]
Составление финансового плана	2 дней	Ср 20.12.23	Чт 21.12.23	8	Менеджер проекта;Электроэнергия[16 кВт * ч]
Завершение разработки бизнес-плана	0 дней	Чт 21.12.23	Чт 21.12.23	9	
Разработка web-приложения для конструирования квестов	112 дней	Пт 22.12.23	Пн 27.05.24		
Анализ разработки программных средств	10 дней	Пт 22.12.23	Чт 04.01.24		
Анализ предметной области	2 дней	Пт 22.12.23	Пн 25.12.23	10	Аналитик[50%];Менеджер проекта[50%];Электроэнергия[16 кВт * ч]
Технический анализ конкурентов	2 дней	Вт 26.12.23	Ср 27.12.23	13	Аналитик[50%];Менеджер проекта[50%];Электроэнергия[16 кВт * ч]
Постановка задачи	1 день	Чт 28.12.23	Чт 28.12.23	14	Аналитик[50%];Менеджер проекта[25%];Проектировщик[25%];Электроэнергия[8 кВт * ч]
Анализ требований к программным средствам	3 дней	Пт 29.12.23	Вт 02.01.24	15	Аналитик[50%];Менеджер проекта[25%];Проектировщик[25%];Электроэнергия[24 кВт * ч]
Выбор технологий и средств разработки	2 дней	Ср 03.01.24	Чт 04.01.24	16	Аналитик[25%];Менеджер проекта[25%];Проектировщик[50%];Электроэнергия[16 кВт * ч]
Завершение анализа разработки	0 дней	Чт 04.01.24	Чт 04.01.24	17	
Проектирование программных средств	14 дней	Пт 05.01.24	Ср 24.01.24		
Проектирование серверной части	5 дней	Пт 05.01.24	Чт 11.01.24	18	Проектировщик;Электроэнергия[40 кВт * ч]
Проектирование структуры базы данных	2 дней	Пт 12.01.24	Пн 15.01.24	20	Проектировщик[75%];Аналитик[25%];Электроэнергия[16 кВт * ч]
Проектирование клиентской части web-приложения	3 дней	Вт 16.01.24	Чт 18.01.24	21	Проектировщик;Электроэнергия[24 кВт * ч]
Проектирование пользовательского интерфейса	4 дней	Пт 19.01.24	Ср 24.01.24	22	Проектировщик;Электроэнергия[32 кВт * ч]
Завершение проектирования	0 дней	Ср 24.01.24	Ср 24.01.24	23	
Реализация программных средств	70 дней	Чт 25.01.24	Ср 01.05.24		
Реализация серверной части	35 дней	Чт 25.01.24	Ср 13.03.24	24	Backend-разработчик;Электроэнергия[280 кВт * ч]
Реализация клиентской части	35 дней	Чт 14.03.24	Ср 01.05.24	26	Frontend-разработчик;Электроэнергия[280 кВт * ч]
Завершение реализации	0 дней	Ср 01.05.24	Ср 01.05.24	27	
Тестирование программных средств	11 дней	Чт 02.05.24	Чт 16.05.24		
Функциональное тестирование	3 дней	Чт 02.05.24	Пн 06.05.24	28	Тестировщик;Электроэнергия[24 кВт * ч]
Оценка метрик покрытия операторов и условий	2 дней	Вт 07.05.24	Ср 08.05.24	30	Тестировщик;Электроэнергия[16 кВт * ч]
Оценка метрик качества кода	2 дней	Чт 09.05.24	Пт 10.05.24	31	Тестировщик;Электроэнергия[16 кВт * ч]
Модульное тестирование	4 дней	Пн 13.05.24	Чт 16.05.24	32	Тестировщик[50%];Электроэнергия[32 кВт * ч];Backend-разработчик[50%]
Завершение тестирования	0 дней	Чт 16.05.24	Чт 16.05.24	33	
Составление и оформление документации	7 дней	Пт 17.05.24	Пн 27.05.24	34	Менеджер проекта;Электроэнергия[56 кВт * ч]
Завершение разработки web-приложения для конструирования квестов	0 дней	Пн 27.05.24	Пн 27.05.24	35	
Завершение реализации проекта	0 дней	Пн 27.05.24	Пн 27.05.24	36	

Рисунок 26 – Список задач проекта

Как видно из приведенного выше плана, вся разработка, включая составление документации, занимает 127 дней. По данным задачам была составлена диаграмма Ганта, приведенная на рисунке 27.

Реализация проекта запланирована на период с 1 декабря 2023 года по 27 мая 2024 года.



Рисунок 27 – Диаграмма Ганта

Отчет по проекту, составленный в Microsoft Project, представлен на рисунке 28.

Даты			
Начало:	Пт 01.12.23	Окончание:	Пн 27.05.24
Баз. начало:	Пт 01.12.23	Баз. окончание:	Пн 27.05.24
Факт. начало:	НД	Факт. окончание:	НД
Отклон. начала:	0 дней	Отклон. окончания:	0 дней
Длительность			
Текущий план:	127 дней	Оставшаяся:	127 дней
Базовый план:	127 дней	Фактическая:	0 дней
Отклонение:	0 дней	Процент завершени:	0%
Трудозатраты			
Текущий план:	1 008 ч	Оставшиеся:	1 008 ч
Базовый план:	1 008 ч	Фактические:	0 ч
Отклонение:	0 ч	Процент завершени:	0%
Затраты			
Текущий план:	р.266 053,92	Оставшиеся:	р.266 053,92
Базовый план:	р.266 053,92	Фактические:	р.0,00
Отклонение:	р.0,00		
Состояние задачи		Состояние ресурса	
Еще не начатые задачи	37	Трудовые ресурсы:	12
Выполняющиеся задачи	0	Трудовые ресурсы с прев. дос:	0
Завершенные задачи:	0	Материальные ресурсы:	1
Итого задач:	37	Итого ресурсов:	13

Рисунок 28 – Отчет по проекту

Отчет о бюджете проекта был также составлен в Microsoft Project. Данный отчет приведен на рисунке 29.

Ид.	Название задачи	Начисление фикс. затрат	Общие затраты
2	Закупка оборудования и ПО	Пропорциональное	p.93 500,00
26	Реализация серверной части	Пропорциональное	p.48 787,20
27	Реализация клиентской части	Пропорциональное	p.48 787,20
35	Составление и оформление докумен	Пропорциональное	p.8 637,44
20	Проектирование серверной части	Пропорциональное	p.8 569,60
23	Проектирование пользовательского	Пропорциональное	p.6 855,68
22	Проектирование клиентской части w	Пропорциональное	p.5 141,76
16	Анализ требований к программным	Пропорциональное	p.4 061,76
8	Составление маркетингового плана	Пропорциональное	p.3 901,76
33	Модульное тестирование	Пропорциональное	p.3 735,68
7	Анализ рынка	Пропорциональное	p.3 701,76
30	Функциональное тестирование	Пропорциональное	p.3 461,76
21	Проектирование структуры базы дан	Пропорциональное	p.3 187,84
17	Выбор технологий и средств разраб	Пропорциональное	p.2 947,84
4	Описание резюме проекта	Пропорциональное	p.2 467,84
5	Описание инновационной идеи	Пропорциональное	p.2 467,84
6	Описание продукции	Пропорциональное	p.2 467,84
9	Составление финансового плана	Пропорциональное	p.2 467,84
13	Анализ предметной области	Пропорциональное	p.2 467,84
14	Технический анализ конкурентов	Пропорциональное	p.2 467,84
31	Оценка метрик покрытия операторо	Пропорциональное	p.2 307,84
32	Оценка метрик качества кода	Пропорциональное	p.2 307,84
15	Постановка задачи	Пропорциональное	p.1 353,92
10	Завершение разработки бизнес-пла	Пропорциональное	p.0,00
18	Завершение анализа разработки	Пропорциональное	p.0,00
24	Завершение проектирования	Пропорциональное	p.0,00
28	Завершение реализации	Пропорциональное	p.0,00
34	Завершение тестирования	Пропорциональное	p.0,00
36	Завершение разработки web-прилож	Пропорциональное	p.0,00
37	Завершение реализации проекта	Пропорциональное	p.0,00
			p.266 053,92

Рисунок 29 – Отчет о бюджете проекта

Отчет о трудозатратах ресурсов приведен на рисунке 30.

Менеджер проекта 204 ч					
Ид.	Название задачи	Единицы	Трудозатраты	Начало	Окончание
2	Закупка оборудования и ПО	100%	8 ч	Пт 01.12.23	Пт 01.12.23
4	Описание резюме проекта	100%	16 ч	Пн 04.12.23	Вт 05.12.23
5	Описание инновационной идеи	100%	16 ч	Ср 06.12.23	Чт 07.12.23
6	Описание продукции	100%	16 ч	Пт 08.12.23	Пн 11.12.23
7	Анализ рынка	100%	24 ч	Вт 12.12.23	Чт 14.12.23
8	Составление маркетингового плана	100%	24 ч	Пт 15.12.23	Вт 19.12.23
9	Составление финансового плана	100%	16 ч	Ср 20.12.23	Чт 21.12.23
13	Анализ предметной области	50%	8 ч	Пт 22.12.23	Пн 25.12.23
14	Технический анализ конкурентов	50%	8 ч	Вт 26.12.23	Ср 27.12.23
15	Постановка задачи	25%	2 ч	Чт 28.12.23	Чт 28.12.23
16	Анализ требований к программным	25%	6 ч	Пт 29.12.23	Вт 02.01.24
17	Выбор технологий и средств разраб	25%	4 ч	Ср 03.01.24	Чт 04.01.24
35	Составление и оформление докумен	100%	56 ч	Пт 17.03.24	Пн 27.03.24
Аналитик 40 ч					
Ид.	Название задачи	Единицы	Трудозатраты	Начало	Окончание
13	Анализ предметной области	50%	8 ч	Пт 22.12.23	Пн 25.12.23
14	Технический анализ конкурентов	50%	8 ч	Вт 26.12.23	Ср 27.12.23
15	Постановка задачи	50%	4 ч	Чт 28.12.23	Чт 28.12.23
16	Анализ требований к программным	50%	12 ч	Пт 29.12.23	Вт 02.01.24
17	Выбор технологий и средств разраб	25%	4 ч	Ср 03.01.24	Чт 04.01.24
21	Проектирование структуры базы да	25%	4 ч	Пт 12.01.24	Пн 15.01.24
Проектировщик 124 ч					
Ид.	Название задачи	Единицы	Трудозатраты	Начало	Окончание
15	Постановка задачи	25%	2 ч	Чт 28.12.23	Чт 28.12.23
16	Анализ требований к программным сред	25%	6 ч	Пт 29.12.23	Вт 02.01.24
17	Выбор технологий и средств разработк	50%	8 ч	Ср 03.01.24	Чт 04.01.24
20	Проектирование серверной части	100%	40 ч	Пт 05.01.24	Чт 11.01.24
21	Проектирование структуры базы данны	75%	12 ч	Пт 12.01.24	Пн 15.01.24
22	Проектирование клиентской части web-п	100%	24 ч	Вт 16.01.24	Чт 18.01.24
23	Проектирование пользовательского ин	100%	32 ч	Пт 19.01.24	Ср 24.01.24
Backend-разработчик 288 ч					
Ид.	Название задачи	Единицы	Трудозатраты	Начало	Окончание
26	Реализация серверной	100%	280 ч	Чт 25.01.24	Ср 13.03.24
33	Модульное тестировани	50%	8 ч	Пн 13.05.24	Вт 14.05.24
Frontend-разработчик 280 ч					
Ид.	Название задачи	Единицы	Трудозатраты	Начало	Окончание
27	Реализация клиентской	100%	280 ч	Чт 14.03.24	Ср 01.05.24
Тестировщик 72 ч					
Ид.	Название задачи	Единицы	Трудозатраты	Начало	Окончание
30	Функциональное тестирование	100%	24 ч	Чт 02.05.24	Пн 06.05.24
31	Оценка метрик покрытия операторо	100%	16 ч	Вт 07.05.24	Ср 08.05.24
32	Оценка метрик качества кода	100%	16 ч	Чт 09.05.24	Пт 10.05.24
33	Модульное тестирование	50%	16 ч	Пн 13.05.24	Чт 16.05.24

Рисунок 30 – Отчет о трудозатратах

Таким образом, разработка проекта займет 127 рабочих дней, бюджет на разработку 266053,92 рублей.

2.3 Реализация программных средств для проведения квестов с применением технологии дополненной реальности

2.3.1 Реализация серверной части

Для разработки серверной части использовался язык C# и фреймворк ASP.NET Core.

Результат реализации серверной части web-приложения представлен диаграммой компонентов (рисунок 31) исходного кода сервиса `questie_backend`, выполняющего обработку основных запросов к серверной части. Стоит отметить, что для уменьшения объема диаграммы на ней не указаны компоненты сущностей предметной области и DTO.

Из диаграммы видно, что взаимодействие с сервисом `questie_backend` происходит через предоставляемый компонентами-контроллерами `NotesController.cs`, `QuestionsController.cs`, `QuestsController.cs`, `TargetsController.cs`, `StatisticController.cs` интерфейс REST API. Все контроллеры предоставляют доступ к своему функционалу через единый порт.

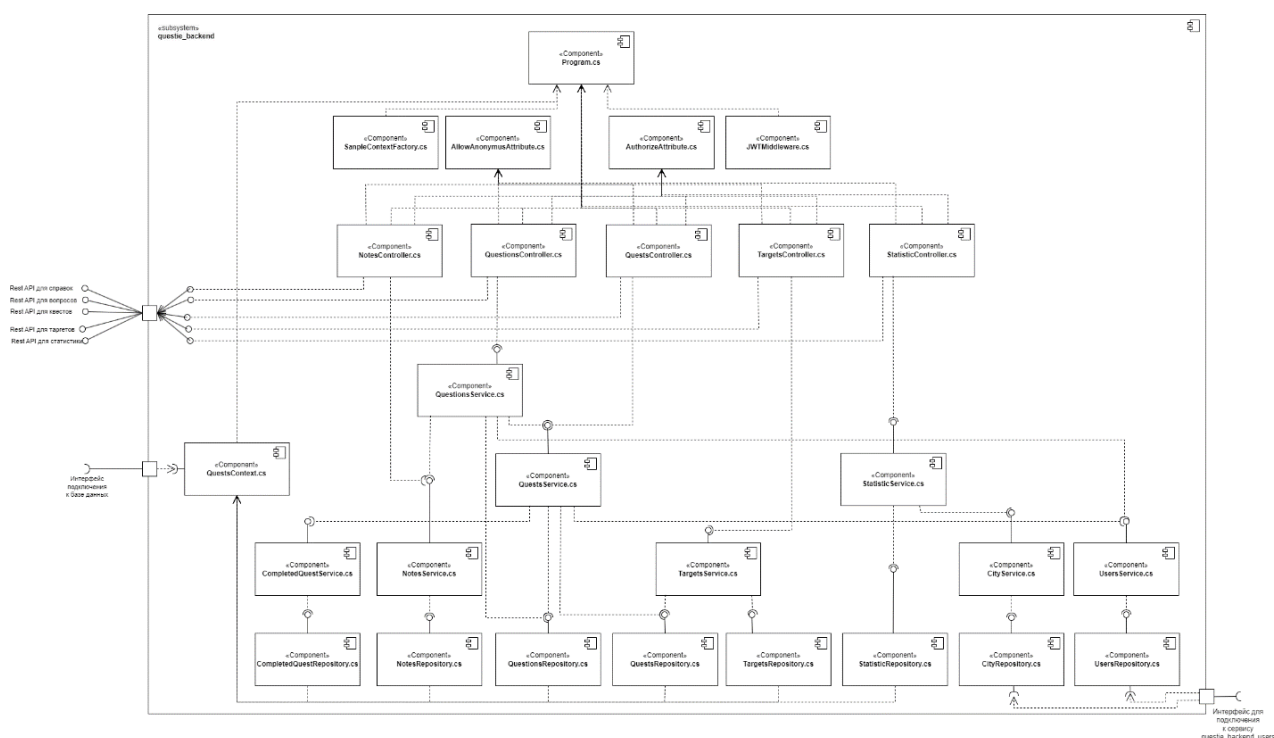


Рисунок 31 – Диаграмма компонентов сервиса `questie_backend`

Для функционирования контроллеры обращаются к интерфейсам, предоставляемым набором сервисных компонентов, в свою очередь также использующих интерфейсы соответствующих им компонентов-репозитория. Все компоненты-репозитории, кроме `UsersRepository.cs` зависят от компонента `QuestsContext.cs`, для функционирования которого требуется обращение к внешнему интерфейсу базы данных PostgreSQL 15.3. Компонент-репозиторий `UsersRepository.cs` обращается к внешнему интерфейсу для получения данных от сервиса `questie_backend_users`.

Все компоненты-контроллеры зависят от компонентов `AllowAnonymousAttribute.cs` и `AuthorizeAttribute.cs`, реализующих атрибуты проверки авторизации пользователей в системе.

Общая структура реализации сервисов `questie_backend_authorization` и `questie_backend_users` имеет похожий характер.

Классы, описывающие компоненты сущностей предметной области и DTO были вынесены в отдельные пакеты и разбиты следующим образом:

- `common_entities` – классы, описывающие перечисления и модели сущностей предметной области и применяющиеся во всех четырех сервисах серверной части приложения;
- `questie_entities` – классы, описывающие DTO сущностей, относящихся к квестам и применяющихся в сервисе `questie_backend`;
- `questie_users` – классы, описывающие DTO сущностей, относящихся к пользователям и применяющихся в сервисах `questie_backend`, `questie_backend_authorization`, `questie_backend_users`.

Описанные пакеты преобразуются в исполняемые файлы `common_entities.dll`, `questie_entities.dll` и `questie_users.dll` соответственно.

Сервисы `questie_backend`, `questie_backend_authorization` и `questie_backend_users` предоставляют для взаимодействия интерфейс REST API, описание которого соответствует спецификации OpenAPI. В качестве примера

на рисунке 32 представлен графический интерфейс Swagger UI, выполняющий PUT-запрос на изменение роли пользователя к сервису `questie_backend_users`.

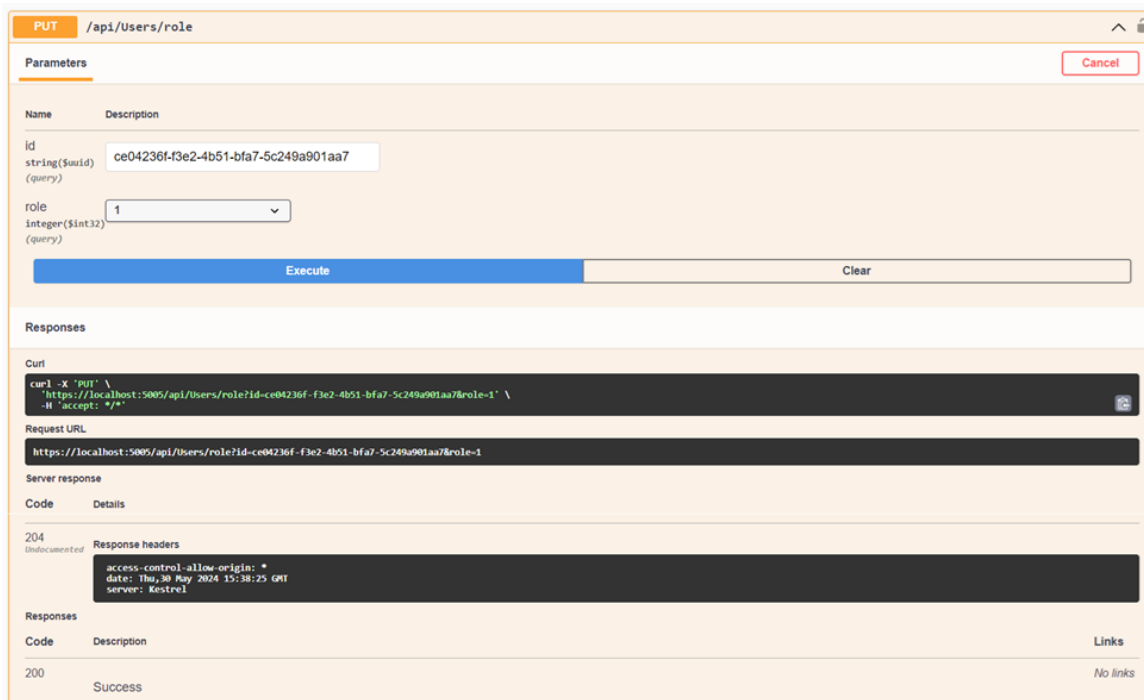


Рисунок 32 – PUT-запрос на изменение роли пользователя

Таким образом был разработан программный код серверной части web-приложения для конструирования квестов и анализа статистики. Фрагменты программного кода приведены в приложении Б. Подана заявка на свидетельство о государственной регистрации программы для ЭВМ «Серверное приложение управления системой для исторических квест-экскурсий с применением технологии дополненной реальности».

2.3.2 Реализация базы данных

Для формирания базы данных был разработан сервис `migrator`, проводящий создание схемы базы данных PostgreSQL из классов, описывающих сущности предметной области, на основе последовательного применения разработанного набора автоматически сгенерированных миграций. Диаграмма компонентов исходного кода сервиса `migrator` приведена на рисунке 33.

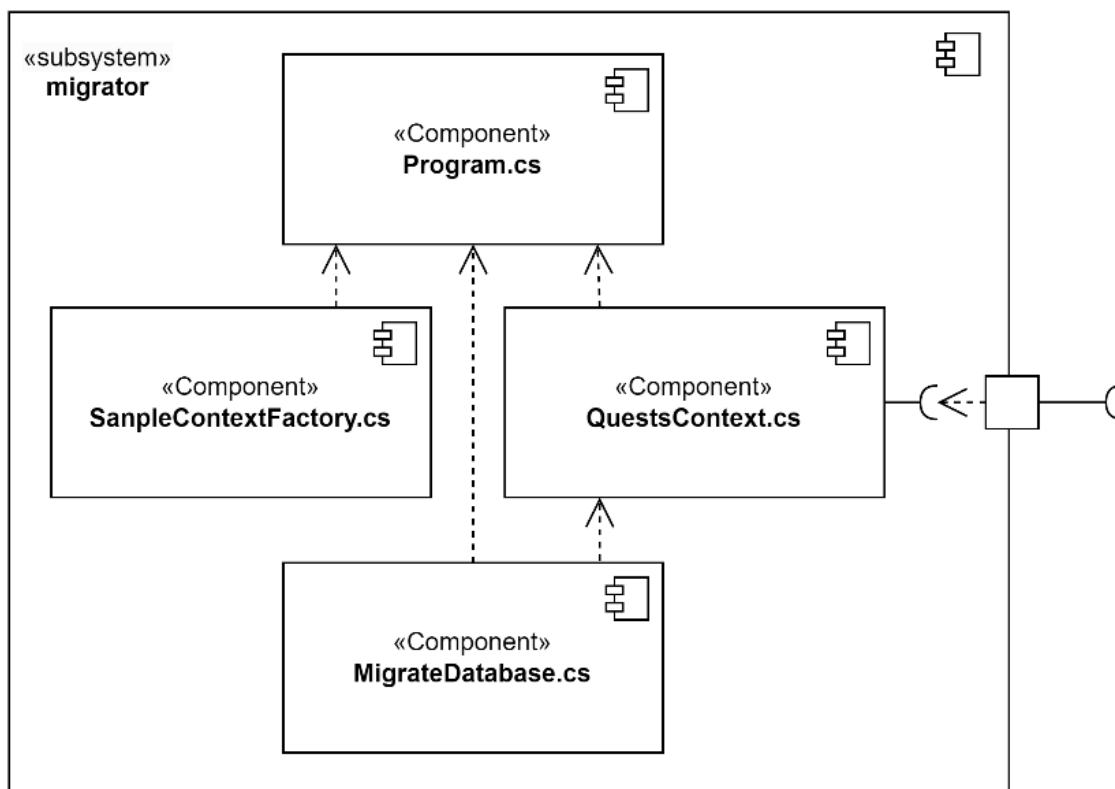


Рисунок 33 – Диаграмма компонентов сервиса migrator

Компонент QuestsContext.cs представляет собой контекст базы данных и отвечает за сопоставление программных классов и таблиц базы данных. Компонент SampleContextFactory.cs реализует шаблон проектирования «Фабрика» и отвечает за создание контекста базы данных с заданной конфигурацией подключения. Компонент MigrateDatabase.cs предназначен для запуска миграций. Реализация компонентов Program.cs и MigrateDatabase.cs приведена в листингах Б.15 и Б.16 соответственно.

Физическая модель сформированной сервисом migrator базы данных приведена на рисунке 34.

Таблица __EFMigrationsHistory является системной, генерируется автоматически и необходима для обеспечения выполнения миграций. Остальные таблицы базы данных предназначены для описания сущностей предметной области.

устройство системы, а также отражает зависимости между программными компонентами. Для уменьшения объема диаграммы на ней не указаны UI-компоненты.

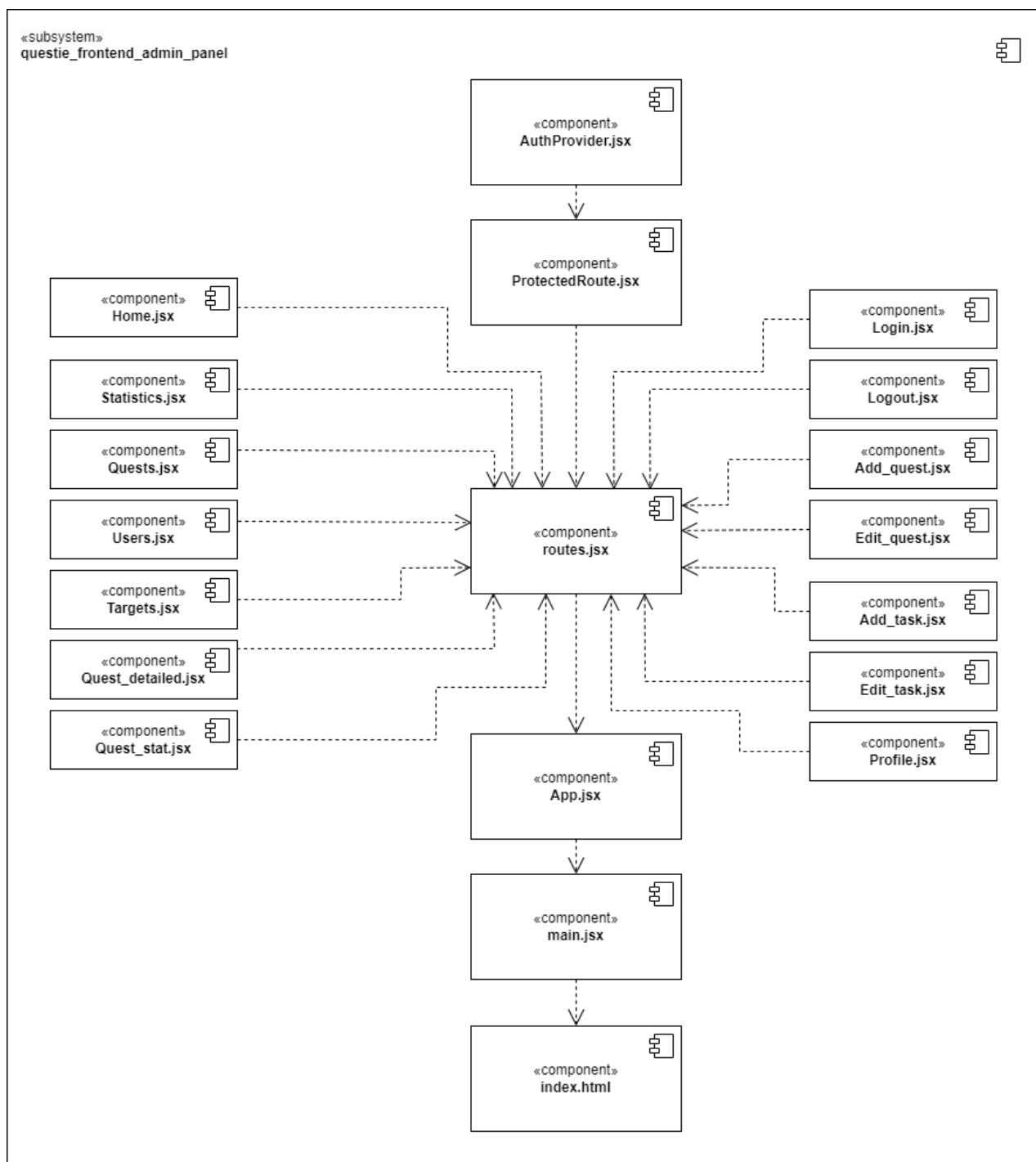


Рисунок 35 – Диаграмма компонентов клиентской части web-приложения

Описание компонентов, отображенных на диаграмме, приведено в таблице 20.

Таблица 20 – Компоненты клиентской части web-приложения

Название компонента	Описание компонента
AuthProvider.jsx	Провайдер аутентификации, контекст для доступа к пользовательскому токену
ProtectedRoute.jsx	Оболочка для аутентифицированных маршрутов
routes.jsx	Маршрутизация страниц приложения в зависимости от роли пользователя
Home.jsx	Главная страница
Login.jsx	Страница авторизации пользователя
Logout.jsx	Логика выхода из приложения
Statistics.jsx	Страница статистики системы
Quests.jsx	Страница управления квестами
Users.jsx	Страница управления пользователями
Targets.jsx	Страница управления AR-таргетами
Quest_detailed.jsx	Страница с подробной информацией об одном квесте
Quest_stat.jsx	Страница статистики одного квеста
Add_quest.jsx	Страница добавления информации о новом квесте
Edit_quest.jsx	Страница редактирования добавленного квеста
Add_task.jsx	Страница добавления задания добавляемого квеста
Edit_task.jsx	Страница редактирования задания добавленного квеста
Profile.jsx	Страница профиль пользователя
App.jsx	Корневой компонент приложения
main.jsx	Точка входа в приложение
index.html	Содержит базовую структуру и точку входа в приложение

Маршрутизация в web-приложении реализована с использованием библиотеки react-router-dom (листинг 1).

Листинг 1 – Файл routes.jsx

```
import { RouterProvider, createBrowserRouter } from "react-router-dom";
import { useAuth } from '../provider/AuthProvider';
import { ProtectedRoute } from "../ProtectedRoute";
import Home from "../components/screens/home/Home";
import Statistics from "../components/screens/statistics/Statistics";
import Quests from "../components/screens/quests/Quests";
import Users from "../components/screens/users/Users";
import Quest_detailed from
"../components/screens/quest_detailed/Quest_detailed";
import Add_quest from "../components/screens/add_quest/Add_quest";
import Targets from "../components/screens/targets/Targets";
import Login from "../components/screens/login/Login";
import Logout from "../components/screens/logout/Logout";
import Add_task from "../components/screens/add_task/Add_task";
import Profile from "../components/screens/profile/Profile";
import { jwtDecode } from 'jwt-decode'
import Edit_quest from "../components/screens/edit_quest/Edit_quest";
import Quest_stat from "../components/screens/quest_stat/Quest_stat";

const Routes = () => {
  const { token } = useAuth();
  const routesForPublic = [
    {
      path: "/home",
      element: <Home/>,
    },
  ],

  const routesForAdmin = [
    {
      path: "/",
      element: <ProtectedRoute />,
      children: [
        {
          path: "/stat",
          element: <Statistics/>,
        },
        {
          path: "/quests",
          element:<Quests/>,
        },
        {
          path: "/profile",
          element:<Profile/>,
        },
        {
          path: "/users",
          element: <Users/>,
        },
      ],
    },
  ],
}
```

```

        path: "/quest/:id",
        element: <Quest_detailed/>,
      },
      {
        path: "/add_quest",
        element: <Add_quest/>,
      },
      {
        path: "/add_task",
        element: <Add_task/>,
      },
      {
        path: "/edit_quest/:id",
        element: <Edit_quest/>,
      },
      {
        path: "/quest_stat/:id",
        element: <Quest_stat/>,
      },
      {
        path: "/targets",
        element: <Targets/>,
      },
      {
        path: "/logout",
        element: <Logout/>,
      },
    ],
  },
];

```

```

const routesForCreator = [
  {
    path: "/",
    element: <ProtectedRoute/>,
    children: [
      {
        path: "/stat",
        element: <Statistics/>,
      },
      {
        path: "/quests",
        element: <Quests/>,
      },
      {
        path: "/profile",
        element: <Profile/>,
      },
      {
        path: "/quest/:id",
        element: <Quest_detailed/>,
      },
      {
        path: "/add_quest",
        element: <Add_quest/>,
      },
    ],
  },
];

```

```

        path: "/add_task",
        element: <Add_task/>,
      },
      {
        path: "/edit_quest/:id",
        element: <Edit_quest/>,
      },
      {
        path: "/quest_stat/:id",
        element: <Quest_stat/>,
      },
      {
        path: "/targets",
        element: <Targets/>,
      },
      {
        path: "/logout",
        element: <Logout/>,
      },
    ],
  },
];

const routesForNotAuthenticatedOnly = [
  {
    path: "/login",
    element: <Login/>,
  },
];

const router = createBrowserRouter([
  ...routesForPublic,
  ...(!token ? routesForNotAuthenticatedOnly : []),
  ...((token &&
jwtDecode(token) ["http://schemas.microsoft.com/ws/2008/06/identity/claims
/role"] === 'Admin') ? routesForAdmin : routesForCreator)
]);
return <RouterProvider router={router} />;
};

export default Routes;

```

С помощью маршрутизации реализовано разделение ролей в приложении. Так, для неавторизованных пользователей доступна лишь главная страница и страница авторизации. Для администраторов системы доступен полный функционал, для создателей квестов недоступно управление пользователями. После авторизации декодируется JWT токен для получения роли пользователя и в зависимости от его роли выбираются доступные ему маршруты.

Для страницы добавления задания квеста была составлена диаграмма компонентов, отображающая зависимости от внешних библиотек и UI-компонентов (рисунок 36).

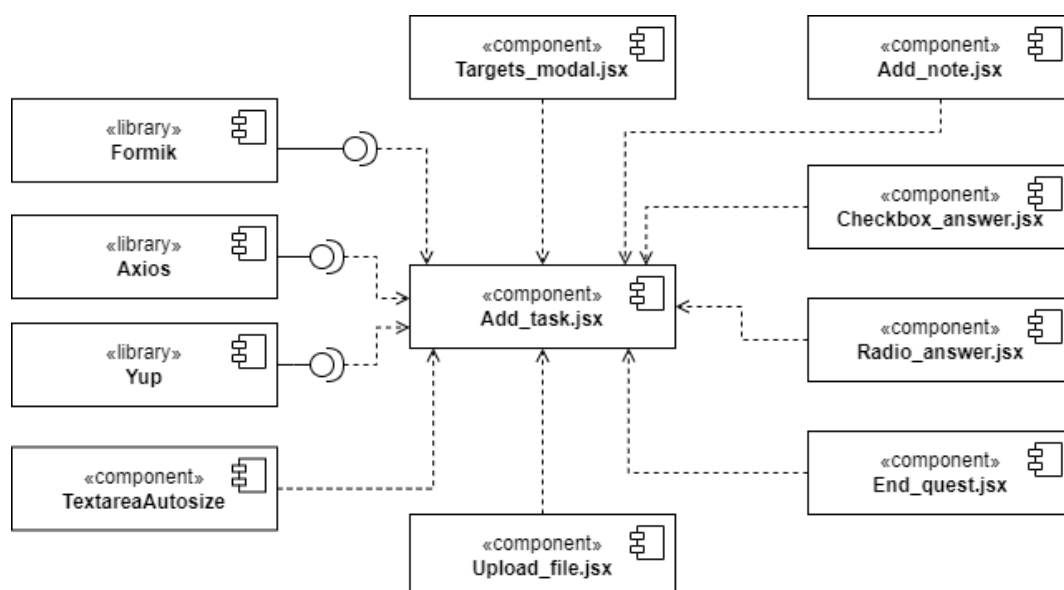


Рисунок 36 – Диаграмма компонентов страницы добавления задания квеста

Описание компонентов, отображенных на диаграмме, приведено в таблице 21.

Таблица 21 – Компоненты страницы добавления задания квеста

Название компонента	Описание компонента
Add_task.jsx	Страница добавления задания добавляемого квеста
Formik	Библиотека для создания форм
Axios	Библиотека для выполнения HTTP-запросов
Yup	Библиотека для валидации
TextareaAutosize	Подключаемый компонент для ввода текста, который автоматически изменяет размер области ввода при изменении размера содержимого
Targets_modal.jsx	UI-компонент для добавления AR-таргета в задание
Add_note.jsx	UI-компонент для добавления исторической справки
Checkbox_answer.jsx	UI-компонент для добавления механики выбора нескольких вариантов ответа
Radio_answer.jsx	UI-компонент для добавления механики выбора одного варианта ответа
Upload_file.jsx	UI-компонент для загрузки изображений
End_quest.jsx	UI-компонент для завершения добавления квеста

Визуализация зависимостей страницы добавления задания приведена на рисунке 37.

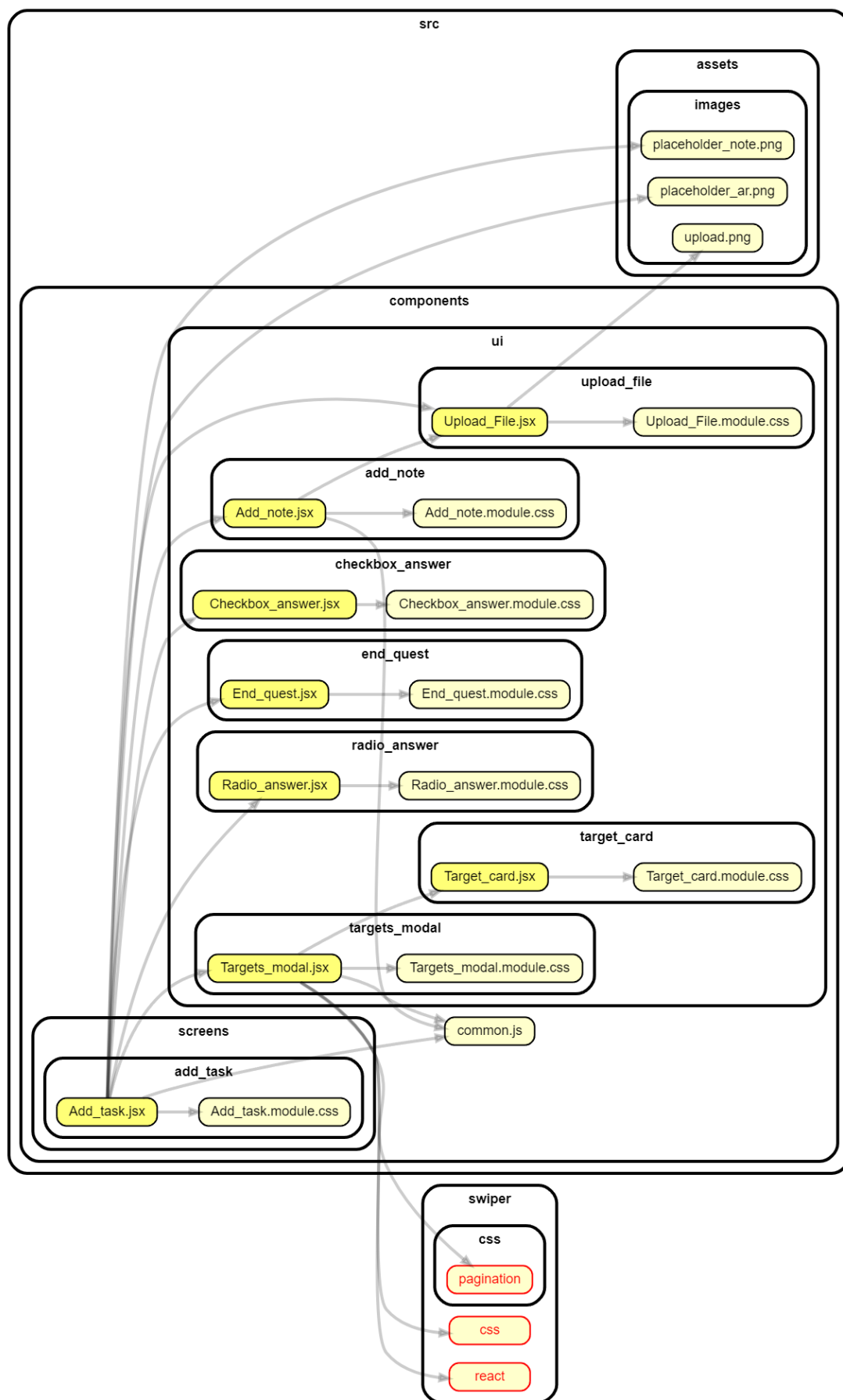


Рисунок 37 – Визуализация зависимостей страницы добавления задания квеста

При разработке web-приложения для конструирования квестов и анализа статистики применяли шаблоны проектирования React. Они помогают добиться модульности и переиспользуемости кода.

Шаблон «провайдер» (provider) обеспечивает глобальный обмен данными в приложении между различными компонентами [34]. React предоставляет встроенный Context API для управления состоянием, с помощью которого можно использовать любую переменную состояния, определенную внутри контекста, в любом компоненте приложения. Так, шаблон проектирования «провайдер» был использован в разрабатываемом приложении для реализации аутентификации пользователей. Для того, чтобы обеспечить доступ к определенным данным всем компонентам приложения, создается контекст аутентификации, а затем все приложение оборачивается в данный контекст. Провайдер предоставляет доступ к токеном аутентификации и функции сохранения токена всем компонентам приложения через контекст.

Шаблон проектирования «условный рендеринг» (conditional rendering) позволяет удобно реализовать отображение различных компонентов в зависимости от каких-либо условий. Пример условного рендеринга из добавления задания квеста приведен в листинге 2.

Листинг 2 – Условный рендеринг компонентов

```
{
  fieldValues.type === '' ? null : null
}
{
  fieldValues.type === 0 ? (
    <>
    <p className={styles.inputLabel}>Введите ответы</p>
    <Radio_answer
      answersRadio={answersRadio}
      setAnswersRadio={setAnswersRadio}
      correctAnswersRadio={correctAnswersRadio}
      setCorrectAnswersRadio={setCorrectAnswersRadio}/>
    </>
  ) : null
}
{
  fieldValues.type === 1 ? (
    <>
    <p className={styles.inputLabel}>Введите ответы</p>
```

```

        <Checkbox_answer
          answers={answers}
          setAnswers={setAnswers}/>
      </>
    ) : null
  }
  {
    fieldValues.type === 2 ? (
      <>
        <p className={styles.inputLabel}>Введите ответ</p>
        <Field name="textAnswer"
          placeholder="Введите ответ"
          type="text"
          onChange={ (e) => {
            handleChange(e)
            setAnswersFinal([e.target.value])
            setCorrectAnswers([0])
          }}
          onBlur={handleBlur}
          value={values.textAnswer}
          className={styles.answerField}>
        </Field>
      </>
    ) : null
  }
}

```

В данном случае, если пользователь не выбрал тип ответов, ничего не будет отрисовано. Если пользователь выбрал тип задания «С одним вариантом ответа», будет отображен компонент для добавления ответов с возможностью указать правильный ответ используя радиокнопку. Если пользователь выбрал тип задания «С несколькими вариантами ответа», будет отображен компонент для добавления ответов с возможностью указать правильные варианты используя чекбокс. В случае, когда пользователь выбрал тип задания «С вводом текстового ответа», будет отображено поле ввода для ответа.

Для стилизации компонентов применялись CSS-модули [35]. Для запросов к серверной части приложения использовалась библиотека *axios*.

Таким образом, был разработан программный код клиентской части web-приложения для конструирования квестов и анализа статистики. Исходный код реализации основного функционала приведен в приложении Б. На клиентскую часть web-приложения получено свидетельство о государственной регистрации программы для ЭВМ [36], представленное в приложении В.

2.3.4 Реализация пользовательского интерфейса

Основными экранами web-приложения, отвечающими за предоставление пользовательского функционала, являются:

- экран статистики;
- экран квестов;
- экран управления пользователями;
- экран AR-таргетов.

На рисунке 38 представлен экран статистики с отображением статистики по регистрации пользователей за день.

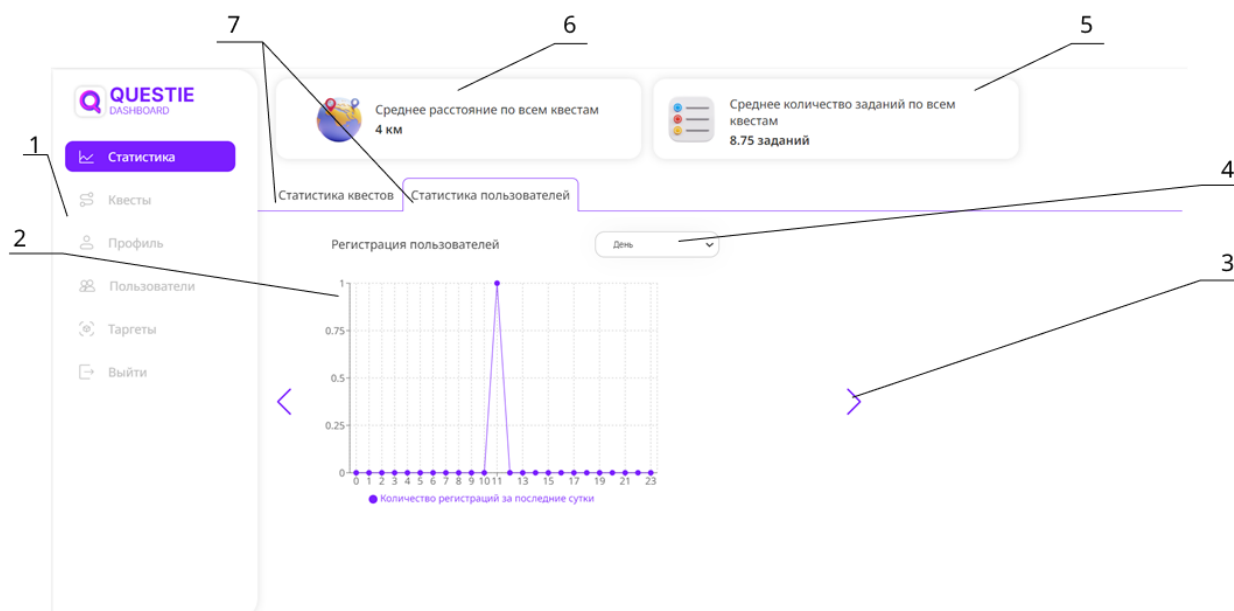


Рисунок 38 – Экран статистики

Рассмотрим основные элементы интерфейса экрана статистики разработанного приложения, представленные на рисунке 34:

- 1 – панель меню;
- 2 – график статистики по регистрации пользователей за месяц;
- 3 – кнопка переключения графиков статистики;
- 4 – выпадающее меню выбора периода, за который будет выведена статистика;
- 5 – карточка со статистикой среднего количества заданий по всем квестам;

6 – карточка со статистикой среднего расстояния по всем квестам.

На рисунке 39 представлен экран квестов.

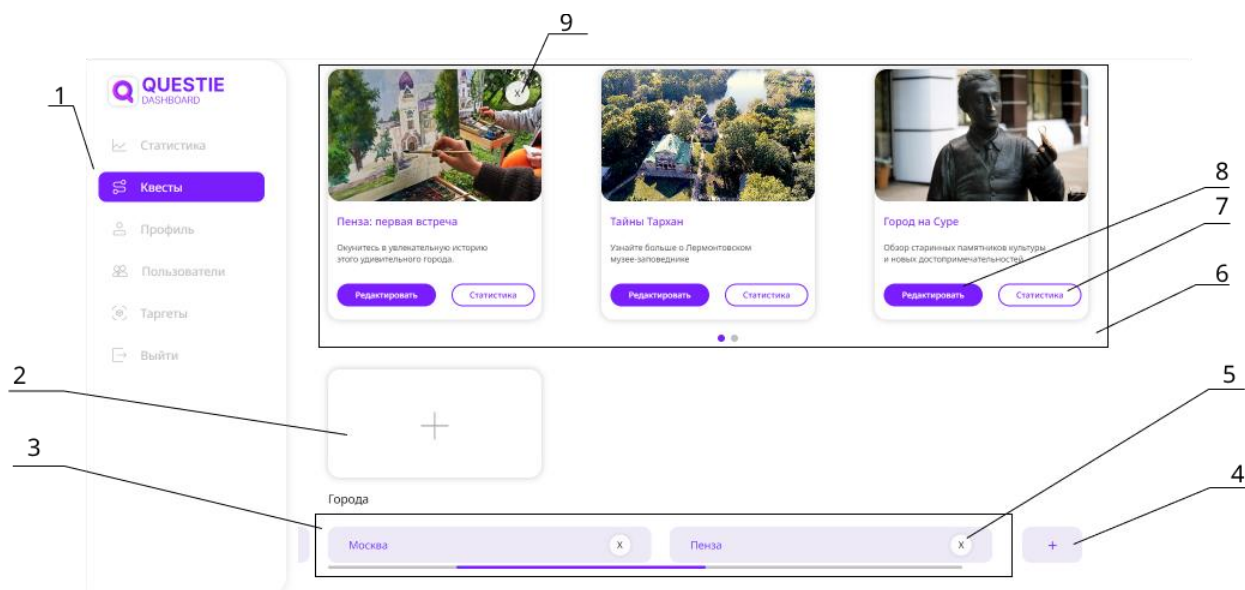


Рисунок 39 – Экран квестов

Рассмотрим основные элементы интерфейса экрана квестов разработанного приложения, представленные на рисунке 39:

- 1 – панель меню;
- 2 – кнопка добавления нового квеста;
- 3 – слайдер с существующими городами;
- 4 – кнопка добавления нового города;
- 5 – кнопка удаления города;
- 6 – слайдер с существующими квестами;
- 7 – кнопка статистики по одному квесту (количество прохождений квеста (с возможностью отслеживания динамики по дням/неделям/месяцам в виде графика));
- 8 – кнопка редактирования квеста;
- 9 – кнопка удаления квеста (появляется при наведении на карточку квеста).

Слайдеры реализованы с использованием библиотеки Swiper, которая позволяет задавать необходимый пользователю вариант навигации, менять оформление кнопок навигации, количество слайдов в области просмотра,

расстояние между слайдами и анимации при переходе. Каждая карточка квеста является ссылкой на страницу с подробной информацией о квесте.

На рисунке 40 представлен экран добавления информации о квесте.

Добавить квест

Название

Выберите город ▼

Описание

Основное изображение



Выберите файл или перетащите его сюда
(только для изображений)

Длительность квеста

Длина маршрута

Точка старта квеста

Рисунок 40 – Экран добавления квеста

При добавлении информации о квесте пользователь должен ввести название квеста, выбрать город, добавить описание квеста, основное изображение, длительность по времени, длину маршрута, добавить точку старта и окончания квеста, а также карту маршрута и дополнительные изображения.

На рисунке 41 представлен экран добавления задания квеста. При добавлении задания пользователь должен ввести вопрос, добавить описание, выбрать тип задания и добавить ответы, выбрав правильные, добавить подсказку, изображение, выбрать AR-таргет и добавить историческую справку, в которую входит название, описание и изображение.

Добавить задание

Вопрос

Описание

Выберите тип ответа

Выберите тип задания

Количество баллов за задание

Количество баллов за задание

Подсказка

Подсказка

Добавить изображение

Выберите файл или перетащите его сюда
(только для изображений)

Рисунок 41 – Страница добавления задания

Завершение добавления квеста реализовано путем нажатия кнопки «Завершить квест» после добавления всех необходимых заданий и выбора утвердительного ответа во всплывающем окне (рисунок 42).

Количество баллов за задание

Количество баллов за задание

Подсказка

Подсказка

Добавить изображение

Выберите файл или перетащите его сюда

Завершить добавление квеста?

Да Нет

Добавить таргет

Добавить историческую справку

Завершить добавление

Добавить вопрос

Рисунок 42 – Завершение добавления квеста

Экран управления пользователями с открытым всплывающим окном редактирования роли пользователя представлен на рисунке 43.

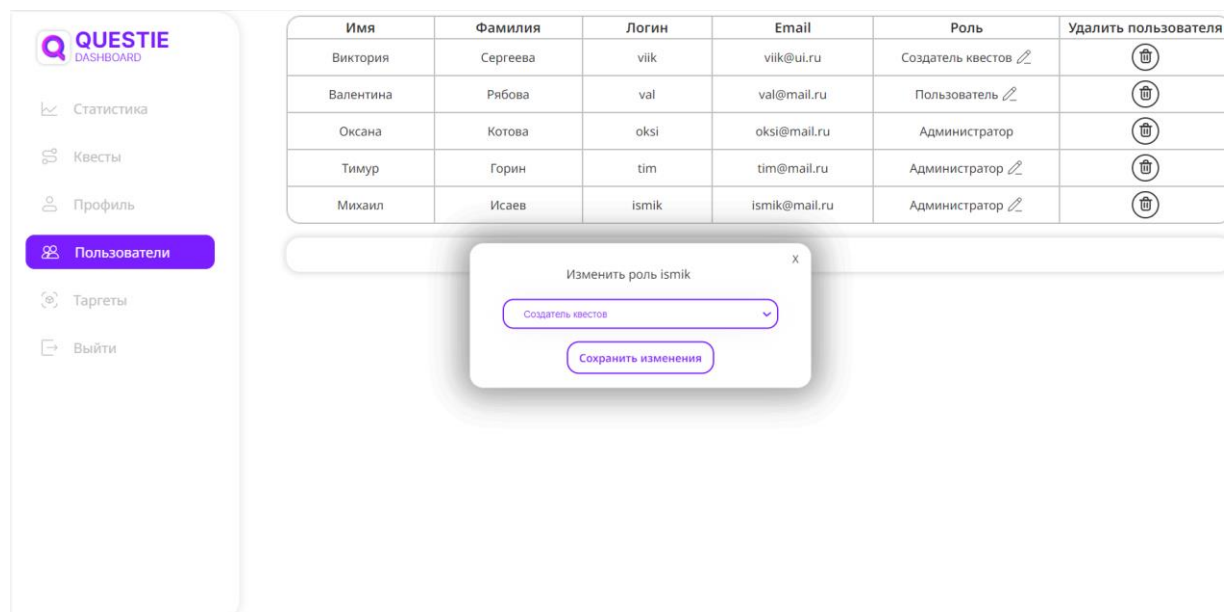


Рисунок 43 – Экран пользователей

Таким образом, был реализован пользовательский интерфейс в соответствии с выявленными функциональными требованиями и макетами интерфейса, разработанными при проектировании системы.

2.3.5 Тестирование и оценка качества кода

Для обеспечения качества web-приложения необходимо провести тестирование разрабатываемого программного продукта.

2.3.5.1 Функциональное тестирование

Для проведения функционального тестирования данного вида тестирования был использован метод функциональных диаграмм.

Функциональная диаграмма и таблица решений была составлена для прецедента «Добавить историческую справку».

Техническое задание на тестируемый участок web-приложения: пользователь должен иметь возможность добавить историческую справку к заданию при добавлении квеста. Для этого он должен ввести название и описание исторической справки, а также добавить изображение (опционально). Название может содержать от 4 до 20 символов, допускается ввод только русских

букв, цифр и знаков препинания. Описание может содержать от 4 до 100 символов и включать русские и латинские буквы, цифры и знаки препинания. Поля являются обязательными для заполнения, добавление изображения не обязательно, однако допустимыми форматами для изображений являются .jpeg, .jpg и .png.

Реакция программы на пользовательский ввод:

- программа должна сообщать о неверной длине названия и описания, если их размер не соответствует требуемому;
- программа должна сообщать об использовании некорректных символов в составе названия и описания, если таковые имеются;
- программа должна сообщать о том, что какое-либо из полей осталось незаполненным;
- программа должна сообщить, если формат изображения некорректен;
- программа должна сообщить об успешном добавлении исторической справки, если все введенные пользователем данные оказались корректными.

Были проанализированы поля формы добавления исторической справки и поведение системы на основе технического задания, выделены причины и следствия событий.

Причины:

- П1 – корректная длина названия исторической справки;
- П2 – корректная длина описания исторической справки;
- П3 – корректное содержание названия;
- П4 – корректное содержание описания;
- П5 – файл изображения представлен в допустимом формате или отсутствует.

Следствия:

- С1 – Сообщение: «Некорректная длина названия»;
- С2 – Сообщение: «Некорректная длина описания»;

- С3 – Сообщение: «Некорректные символы в составе названия»;
- С4 – Сообщение: «Некорректные символы в составе описания»;
- С5 – Сообщение: «Некорректный формат файла»;
- С6 – Успешное добавление исторической справки.

Причины и следствия событий приведены в виде функциональной диаграммы на рисунке 44.

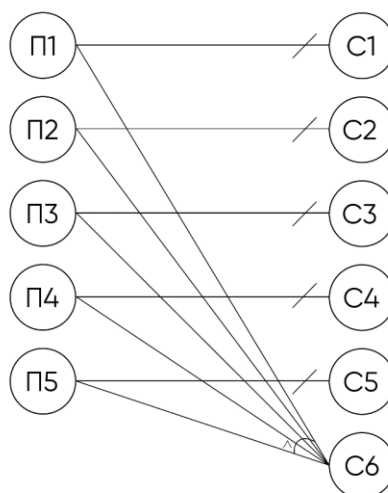


Рисунок 44 – Функциональная диаграмма добавления исторической справки

На основе функциональной диаграммы была составлена таблицы решений, отображающие комбинации причин и соответствующие им следствия. Решения для комбинаций 1 – 16 приведены в таблице 22.

Таблица 22 – Таблица решений тестирования авторизации пользователя

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
П1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
П2	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
П3	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
П4	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
П5	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
С1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
С2	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
С3	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0
С4	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
С5	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
С6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Решения для комбинаций 17 – 32 приведены в таблице 23.

Таблица 23 – Таблица решений тестирования авторизации пользователя

	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31	32
П1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
П2	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
П3	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
П4	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
П5	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
С1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
С2	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
С3	1	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0
С4	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0
С5	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0
С6	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

На основе таблицы решений был разработан набор тест-кейсов, по которым было проведено тестирование. Таблица с тест-кейсами и результаты функционального тестирования приведены в приложении Г.

Таким образом было проведено функциональное тестирование прецедента «Добавить историческую справку» методом функциональных диаграмм. На основе функциональной диаграммы и таблицы решений были составлены тест-кейсы. Проведенное тестирование показало успешный результат прохождения всех составленных тест-кейсов для прецедента «Добавить историческую справку».

2.3.5.2 Структурное тестирование

Структурное тестирование проводилось с использованием метода покрытия операторов и условий. Рассмотрим структурное тестирование функции добавления ответа для задания квеста с типом «Выбор одного варианта ответа». Программный код приведен в листинге 3.

Листинг 3 – Функция добавления ответа

```
const addAnswer = (event) => {
  event.preventDefault();
  const trimmedUserInput = userInput.trim();
  if (trimmedUserInput) {
    if (!answersRadio.some(item => item === trimmedUserInput)) {
      setAnswersRadio((existingItems) => [
        ...existingItems,
        trimmedUserInput
      ])
      setUserInput("");
    }
  }
}
```

```

    } else {
        alert("Данное значение уже добавлено")
        setUserInput("");
    } else {
        alert("Введите значение")
    }
};

```

Схема алгоритма добавления ответа для задания квеста с типом «Выбор одного варианта ответа» представлена на рисунке 45.

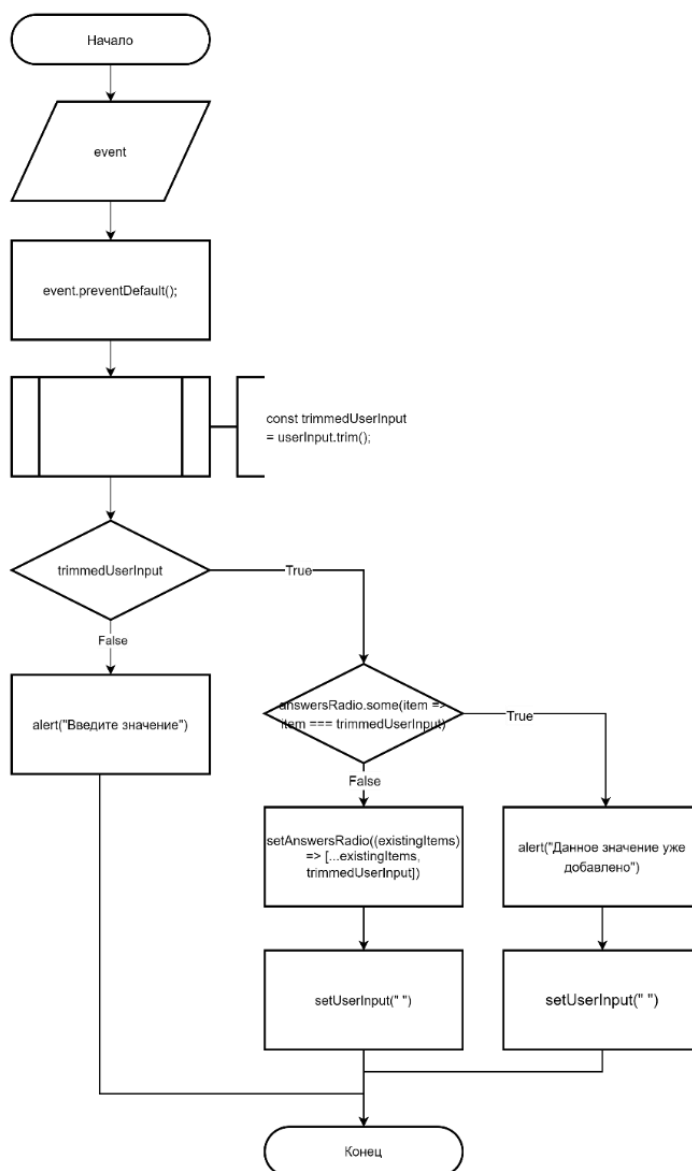


Рисунок 45 – Схема алгоритма добавления ответа для задания квеста с типом «Выбор одного варианта ответа»

В данной функции происходит проверка пользовательского ввода, если поле для добавления ответа является пустым, пользователь видит сообщение об

ошибке. В противном случае, происходит проверка на наличие введенного ответа в списке всех ответов. Если данный ответ не был обнаружен, он записывается в список, значение поля обнуляется. Если же ответ уже был добавлен пользователем, выводится сообщение об ошибке добавления, значение поля обнуляется.

На основании блок-схемы был составлен тест-кейс, покрывающий все ее операторы (таблица 24).

Таблица 24 – Тест-кейс покрытия операторов функции добавления ответа

Входные данные	Операторы	Условия	Ожидаемый результат
event	1, 2, 3, 7, 8, 11	4 (true), 6 (false)	Ответ добавлен в список ответов

Для покрытия условий были составлены три тест-кейса. Один из них повторяет тест-кейс, необходимый для покрытия операторов, два других приведены в таблице 25.

Таблица 25 – Тест-кейс покрытия условий функции добавления ответа

Входные данные	Операторы	Условия	Ожидаемый результат
event	1, 2, 3, 5, 11	4 (false)	Ответ не добавлен в список ответов
event	1, 2, 3, 9, 10, 11	4 (true), 6 (true)	Ответ не добавлен в список ответов

Результирующий список тест-кейсов с описаниями маршрутов, а также результат тестирования приведены в таблице 26.

Таблица 26 – Тест-кейс покрытия маршрутов функции добавления ответа

Входные данные	Маршрут	Ожидаемый результат	Полученный результат
event	1, 2, 3, 4, 5, 11	Ответ не добавлен в список ответов	Ответ не добавлен в список ответов
event	1, 2, 3, 4, 6, 9, 10, 11	Ответ не добавлен в список ответов	Ответ не добавлен в список ответов
event	1, 2, 3, 4, 6, 7, 8, 11	Ответ добавлен в список ответов	Ответ добавлен в список ответов

Таким образом, было проведено структурное тестирование функции добавления ответа для задания квеста с типом «Выбор одного варианта ответа» составлены и успешно пройдены тест-кейсы, обеспечивающие покрытие операторов, условий и маршрутов.

2.3.5.3 Метрики качества кода

Метрики качества кода позволяют выявить возможные проблемы в коде, такие как плохая читаемость кода или его дублирование, а также оценить уровень качества для того, чтобы разработчики могли принять меры для улучшения процесса разработки.

Для проверки качества реализации клиентской части web-приложения для конструирования квестов и анализа статистики было принято решение рассчитать метрики Холстеда, что позволит оценивать сложность разработанного программного кода.

Список операторов и операндов, необходимый для расчета метрик Холстеда по компоненту `Added_requests.jsx`, отображающему статистику добавленных квестов, приведены в таблице 27.

Таблица 27 – Списки операторов и операндов компонента `Added_requests.jsx`

№	Операторы		Операнды	
	Оператор	Количество	Операнд	Количество
1	(	28	period	9
2	)	28	quests	2
3	,	4	chartName	2
4	;	2	getQuestsAddedDefault	2
5	{	18	response	7
6	}	18	error	4
7	+	4	e	6
8	=	23	getQuestsAddedByPeriod	2
9	===	3	baseUrl_requests_service	2
10	setQuests	3		
11	setChartName	4		
12	if	1		
13	return	1		
14	Added_requests	2		
15	handleDropdown	2		
16	useEffect	1		
17	get	2		
18	then	2		
19	catch	2		
20	let	1		
21	=>	8		
	Всего	157	Всего	36

Метрики Холстеда основаны на следующих параметрах: N – длина программы; n – словарь программы.

Длина программы вычисляется по формуле:

$$N = N_1 + N_2 = 157 + 36 = 193,$$

где N – длина программы (общее количество операторов и операндов);

N_1 – общее количество операторов;

N_2 – общее количество операндов.

Для вычисления словаря программы применяется формула:

$$n = n_1 + n_2 = 21 + 9 = 30,$$

где n – словарь программы;

n_1 – количество уникальных операторов;

n_2 – количество уникальных операндов.

Объем программы вычисляется по следующей формуле:

$$V = N * \log_2 n = 193 * \log_2 30 = 947,051,$$

где V – объем программы по метрике Холстеда;

N – длина программы;

n – словарь программы.

Для вычисления сложности программы применяется формула:

$$D = (n_1 / 2) * (N_2 / n_2) = 2,625,$$

где D – сложность программы по метрике Холстеда;

n_1 – количество уникальных операторов;

N_2 – общее количество операндов;

n_2 – количество уникальных операндов.

Допустимым показателем является диапазон объема программы от 20 до 1000 для одного метода. Текст программы можно считать компактным.

Для проверки качества реализации серверной части web-приложения для конструирования квестов и анализа статистики был проведен автоматический подсчет метрик для сервиса `questie_backend_users`, выполненный с помощью интегрированной среды разработки Microsoft Visual Studio 2022 Community Edition (рисунок 46).

Иерархия	Индекс удобства подде...	Сложность организаци...	Глубина наследования	Взаимозависимость кл...	Строки исходного кода	Строки исполняемого кода
questie_backend_users (Debug)	84	205	4	156	1 213	299
questie_backend_users.Exception	100	1	4	1	7	0
NotFoundUserException	100	1	4	1	4	0
questie_backend_users	42	2	1	64	118	41
Program	42	2	1	64	115	41
questie_backend_users.Controllers	84	39	2	35	248	51
CitiesController	88	5	2	17	42	7
StatisticController	80	17	2	20	87	21
UserController	86	17	2	24	108	23
questie_backend_users.Data	81	10	2	24	33	9
SampleContextFactory	63	1	1	17	18	9
UsersContext	100	9	2	8	9	0
questie_backend_users.Middleware	77	15	2	45	126	29
AllowAnonymousAttribute	100	1	2	3	3	0
AuthorizeAttribute	67	8	2	23	49	11
IWTMiddleware	66	6	1	29	65	18
questie_backend_users.Repositories.Implementations	82	28	1	30	209	63
CitiesRepository	89	5	1	17	30	8
StatisticRepository	77	16	1	26	129	40
UsersRepository	81	7	1	18	41	15
questie_backend_users.Repositories.Interfaces	100	23	0	9	41	0
questie_backend_users.Services.Implementations	77	63	1	27	389	106
CitiesService	88	11	1	8	59	16
StatisticService	69	31	1	17	188	43
UserService	75	21	1	12	133	47
questie_backend_users.Services.Interfaces	100	24	0	14	42	0
ICitiesService	100	4	0	6	7	0
IStatisticService	100	8	0	8	11	0
IUserService	100	12	0	7	15	0

Рисунок 46 – Результат автоматического расчета метрик кода

Индекс удобства поддержки представляет относительную простоту обслуживания кода, имеет значения от 0 (низкий уровень поддержки) до 100 и рассчитывается по формуле [37]:

$$\text{Maintainability Index} = \text{MAX}(0, (171 - 5.2 * \ln(\text{Halstead Volume}) - 0.23 * (\text{Cyclomatic Complexity}) - 16.2 * \ln(\text{Lines of Code})) * 100/171),$$

где Maintainability Index – индекс удобства поддержки;

Halstead Volume – объем программы по метрике Холстеда;

Cyclomatic Complexity – цикломатическая сложность программы;

Lines of Code – количество строк кода.

Цикломатическая сложность измеряет структурную сложность кода. Программа, которая имеет сложный поток управления, требует больше тестов для достижения хорошего покрытия кода и менее поддерживается, соответственно, чем ниже показатель цикломатической сложности, тем качественнее написан код. Глубина наследования указывает количество различных классов, наследуемых друг от друга, вплоть до базового класса, чем выше данный показатель, тем глубже наследование [38].

Взаимозависимость классов – это мера того, сколько классов использует один класс [39]. При качественной архитектуре ПО типы и методы должны иметь низкую связь. Высокая связь ведет к трудностям с поддержкой кода.

2.3.5.4 Модульное тестирование

Для проверки корректности работы серверной части web-приложения для конструирования квестов и анализа статистики было проведено модульное тестирование, которое позволило в автоматическом режиме проверить работоспособность функций программы.

Реализация модульного тестирования проводилась с использованием фреймворков XUnit (непосредственно для разработки модульных тестов) и Moq (для имитации функциональности и создания мок-объектов).

В качестве примера в приложении Д приведен исходный код реализации модульных тестов для класса-контроллера исторических справок NotesController, результаты прохождения тестов представлены на рисунке 47.

Тестирование	Длительность	Сводка по группе
questie_backend.Tests.Controllers	146 мс	TestNotesController
TestNotesController (18)	72 мс	Тесты в группе: 18
		Общая длительность: 72 мс
		Результаты
		18 Пройден
CreateNote_BadRequestRes...	6 мс	
CreateNote_BadRequestRes...	< 1 мс	
CreateNote_BadRequestRes...	< 1 мс	
CreateNote_CreatedAtAction	1 мс	
DeleteNote_BadRequestRes...	< 1 мс	
DeleteNote_BadRequestRes...	< 1 мс	
DeleteNote_NoContentResul...	< 1 мс	
DeleteNote_NotFoundResult...	< 1 мс	
GetAllNotes_ShouldReturn2...	2 мс	
GetNoteById_BadRequestRe...	< 1 мс	
GetNoteById_BadRequestRe...	< 1 мс	
GetNoteById_NoteOutDto	59 мс	
GetNoteById_NotFoundResu...	< 1 мс	
UpdateNote_BadRequestRes...	< 1 мс	
UpdateNote_BadRequestRes...	< 1 мс	
UpdateNote_BadRequestRes...	< 1 мс	
UpdateNote_NoContentRes...	< 1 мс	
UpdateNote_NotFoundResul...	4 мс	

Рисунок 47 – Результаты модульного тестирования

Таким образом, модульное тестирование показало корректность функционирования разработанного программного обеспечения.

Заключение

В ходе выполнения выпускной квалификационной работы в формате стартапа был разработан бизнес-план системы для исторических квест-экскурсий с применением технологии дополненной реальности и выполнены следующие задачи:

- детально описана инновационная идея проекта, предлагаемый продукт и его технико-экономические характеристики;
- проведен анализ рынка и сделаны прогнозы развития отрасли;
- разработан маркетинговый план проекта, описано ценообразование продукта и способ его продажи;
- разработан финансовый план по этапам развития проекта, описаны ожидаемые на каждом этапе показатели;
- найдены партнеры проекта и привлечено финансирование на первый этап разработки.

В ходе выполнения выпускной квалификационной работы было реализовано web-приложение для конструирования квестов и анализа статистики. В процессе реализации были выполнены следующие задачи:

- проанализирована предметная область, выявлены функциональные и нефункциональные требования к системе;
- спроектирована архитектура серверной части web-приложения и модель базы данных;
- спроектирована клиентская часть web-приложения и пользовательский интерфейс;
- проведено планирование разработки и оценка бюджета;
- разработана серверная часть web-приложения и база данных;
- разработана клиентская часть web-приложения;
- проведено тестирование и оценка качества кода с использованием различных метрик.

По результатам выполнения выпускной квалификационной работы разработано web-приложение для конструирования квестов и анализа статистики, отвечающее всем требованиям технического задания.

Для реализации проекта одобрено грантовое финансирование в размере 500000 рублей (Договор №18446ГУ/2023 от 16.08.2023) по программе УМНИК-2022 (Пензенская область). Опубликовано 2 статьи, входящих в Российский индекс научного цитирования, 1 статья находится на стадии издания сборника. Получено свидетельство о государственной регистрации программы для ЭВМ «Web-клиент административной панели системы для исторических квест-экскурсий с применением технологии дополненной реальности», подана заявка на свидетельство о государственной регистрации программы для ЭВМ «Серверное приложение управления системой для исторических квест-экскурсий с применением технологии дополненной реальности».

Список использованных источников

1. Петухина Е.В., Ден В.Г. Методические и организационные аспекты разработки и проведения квестов в рамках экскурсионной деятельности // КиберЛенинка – Научная электронная библиотека [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/metodicheskie-i-organizatsionnye-aspekty-razrabotki-i-provedeniya-kvestov-v-ramkah-ekskursionnoy-deyatelnosti> (дата обращения: 06.12.2023).
2. Кургина С.О., Копцева М.Г., Суржигов В.И. Квест-экскурсия как инновационная форма экскурсионного продукта // КиберЛенинка – Научная электронная библиотека [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/kvest-ekskursiya-kak-innovatsionnaya-forma-ekskursionnogo-produkta> (дата обращения: 06.12.2023).
3. Воробьев А. Внедрение дополненной реальности (AR) в обучение. Часть #4 AR и геймификация // ARGUMENT [Электронный ресурс]. URL: <https://argument.digital/blog/vnedrenie-dopolnennoj-realnosti-ar-v-obuchenie-chas4-ar-i-gejmifikaciya/> (дата обращения: 06.12.2023).
4. Зырянова И.С. Самостоятельный туризм: определение понятия и синонимов // КиберЛенинка – Научная электронная библиотека. [Электронный ресурс]. URL: <https://cyberleninka.ru/article/n/samostoyatelnyy-turizm-opredelenie-ponyatiya-i-sinonimov> (дата обращения: 08.12.2023).
5. Стратегия развития туризма в Российской Федерации на период до 2035 года: распоряжение Правительства РФ от 20.09.2019 №2129-р // Правительство России [Электронный ресурс]. URL: <http://government.ru/docs/37906/> (дата обращения: 11.12.2023).
6. Феоктистов Д. Свежий взгляд: правительство поменяло ориентиры в развитии туризма // Национальные проекты России [Электронный ресурс]. URL: <https://национальныепроекты.рф/news/svezhiy-vzglyad-pravitelstvo-pomenyalo-orientiry-v-razvitii-turizma> (дата обращения: 12.12.2023).

7. Божко В.Е. Разработка и экономическое обоснование проекта создания мобильного приложения туристской фирмы (на примере ООО «Хэппи тур»): дипломный проект – Белгород, 2016. [Электронный ресурс]. URL: <https://nauchkor.ru/uploads/documents/5a402efc7966e104c6a3e5d1.pdf> (дата обращения: 13.12.2023).
8. Результаты региональных конкурсов по программе «УМНИК» // Фонд содействия инновациям [Электронный ресурс]. URL: <https://fasie.ru/press/fund/umnik-region-results-2023/> (дата обращения: 15.12.2023).
9. Грабова А. По Пензе минувших эпох / А. Грабова // Деловой – 2023, – №15 (240), с.10 – 11.
10. Преакселератор УМНИК // Дотком Энтерпрайз [Электронный ресурс]. URL: <https://dotcomgroup.ru/akselerator-umnik> (дата обращения: 15.12.2023).
11. Акселерационная программа Пензенского государственного университета // Институт Международного Сотрудничества [Электронный ресурс]. URL: https://ims.pnzgu.ru/international_cooperation_ims/projects_ims/accelator (дата обращения: 16.12.2023).
12. Агапова, В. А. Выявление функциональных особенностей мобильных приложений квест-экскурсий / В. А. Агапова, А. С. Кудашов, Е. А. Дзюба // Информационные технологии в науке и образовании. Проблемы и перспективы : сборник статей по материалам X Всероссийской научно-практической конференции, Пенза, 15 марта 2023 года. – Пенза: Пензенский государственный университет, 2023. – С. 200 – 204. – EDN DBACUY.
13. Киреева Ю.А., Полоцкая Н.А. Особенности организации квест-экскурсий // КиберЛенинка – Научная электронная библиотека. [Электронный ресурс] URL: <https://cyberleninka.ru/article/n/osobennosti-organizatsii-kvest-ekskursiy> (дата обращения: 23.12.2023).
14. Курылев И. Геймификация: новый уровень вашего маркетинга. Как повторить опыт «Сбера», «Якитории», Burger King и MIXIT // mindbox журнал

[Электронный ресурс] URL: <https://mindbox.ru/journal/experts/gejmifikaciya/> (дата обращения: 25.12.2023).

15. Лиманова, Н. И. Анализ эффективности клиент-серверной архитектуры / Н. И. Лиманова, И. А. Селезнев // Бюллетень науки и практики. – 2022. – Т. 8, № 7. – С. 392 – 396. – DOI 10.33619/2414-2948/80/37. – EDN AONLOB.

16. Прунов А. Архитектура «клиент-сервер»: основы и принципы // Айтиспектр [Электронный ресурс]. URL: <https://itspectr.ru/arhitektura-klient-server-osnovy-i-princzipy/> (дата обращения: 08.01.2024).

17. Ричардс, М. Фундаментальный подход к программной архитектуре: паттерны, свойства, проверенные методы / М. Ричардс, Н. Форд – СПб.: Питер, 2023. – 448 с.: ил. – (Серия «Для профессионалов»).

18. Рокатанский М. Инженерная надежность и отказоустойчивость распределенной системы // Хабр [Электронный ресурс]. URL: <https://habr.com/ru/companies/otus/articles/561630/> (дата обращения: 09.01.2024).

19. Сомасегар, С. Руководство Microsoft по проектированию архитектуры приложений / С. Сомасегар, С. Гатри, Д. Хилл. – Корпорация Майкрософт, 2-е изд-е, 2009. – 529 с.

20. Бешелев, С. Д. Математико-статистические методы экспертных оценок / С. Д. Бешелев, Ф. Г. Гурвич – 2-е изд., перераб. и доп. – М.: Статистика, 1980. – 263 с., ил. – (Матем. статистика для экономистов).

21. Sanctis, V. Building Web APIs with ASP.NET Core / V. Sanctis – NY: Manning Publications Co, 2023. – 821 с.

22. Эванс, Э. Предметно-ориентированное проектирование (DDD): структуризация сложных программных систем / Э. Эванс. – : Пер. с англ. – М.: 000 «И.Д. Вильямс», 2011. – 448 с. : ил. – Парал. тит. англ.

23. Fowler M. Inversion of Control Containers and the Dependency Injection pattern // [martinFowler.com](http://martinfowler.com) [Электронный ресурс]. URL: <https://www.martinfowler.com/articles/injection.html> (дата обращения: 11.01.2024).

24. Dependency Injection: Принцип. Паттерн. Контейнер // Calabonga SOFT [Электронный ресурс]. URL: <https://www.calabonga.net/blog/post/dependency-injection-principle-pattern-container> (дата обращения: 11.01.2024).

25. Code-First vs Model-First vs Database-First: Pros and Cons // Ryadel [Электронный ресурс]. URL: <https://www.ryadel.com/en/code-first-model-first-database-first-vs-comparison-orm-asp-net-core-entity-framework-ef-data/> (дата обращения: 15.01.2024).

26. Lerman, J. Programming Entity Framework: Code First / J. Lerman, R. Miller – Gravenstein Highway North, Sebastopol, CA: O'Reilly Media, Inc, 2011. – 177 с.

27. Киорогло Р. Подходы code first, database first и model first в создании базы данных используя entity framework // IBN Instrument Bibliometric National [Электронный ресурс]. URL: https://ibn.idsi.md/sites/default/files/imag_file/486-490_3.pdf (дата обращения: 16.01.2024).

28. Торопкин, Р. А. Технологии оптимизации работы сайта на примере аналитической системы публикационной активности Пензенского государственного университета / Р. А. Торопкин, Я. В. Зиновьев, Н. С. Рассказов, М. А. Митрохин // Модели, системы, сети в экономике, технике, природе и обществе. – 2020. – № 4 (36). – С. 71–78. – DOI 10.21685/2227-8486-2020-4-7

29. angular vs react vs vue // npm trends [Электронный ресурс]. URL: <https://npmtrends.com/angular-vs-react-vs-vue> (дата обращения: 10.03.2024).

30. React vs Vue vs Angular // Хабр [Электронный ресурс]. URL: <https://habr.com/ru/companies/auriga/articles/703836/> (дата обращения: 13.03.2024).

31. Aaron Xie Angular vs Vue vs React: Choose the best framework // educative [Электронный ресурс]. URL: <https://www.educative.io/blog/react-angular-vue-comparison> (дата обращения: 12.03.2024).

32. Activity Diagrams // The Unified Modeling Language [Электронный ресурс]. URL: <https://www.uml-diagrams.org/activity-diagrams.html> (дата обращения: 13.03.2024).
33. Леоненков, А. В. Самоучитель UML 2 / А.В. Леоненков – СПб.: БХВ-Петербург, 2007. – 576 с.: ил.
34. 7 Best React Design Patterns That Every Developer Should Know // GeeksforGeeks [Электронный ресурс]. URL: <https://www.geeksforgeeks.org/react-design-patterns/> (дата обращения: 14.04.2024).
35. Using CSS Modules with React // erzhtor [Электронный ресурс]. URL: <https://www.erzhtor.com/articles/using-css-modules-with-react> (дата обращения: 14.04.2024).
36. Свидетельство о государственной регистрации программы для ЭВМ № 2024660878 Российская Федерация. Web-клиент административной панели системы для исторических квест-экскурсий с применением технологии дополненной реальности : № 2024660119 : заявл. 04.05.2024 : опубл. 14.05.2024 / В. А. Агапова, А. С. Кудашов, Д. А. Дьячков, А. А. Левин. – EDN DVQUOO
37. Метрики кода – диапазон индексов удобства поддержки и их значения // Microsoft Learn [Электронный ресурс]. URL: <https://learn.microsoft.com/ru-ru/visualstudio/code-quality/code-metrics-maintainability-index-range-and-meaning?view=vs-2022&source=recommendations> (дата обращения: 15.05.2024).
38. Значения метрик кода // Microsoft Learn [Электронный ресурс]. URL: <https://learn.microsoft.com/ru-ru/visualstudio/code-quality/code-metrics-values?view=vs-2022> (дата обращения: 15.05.2024).
39. Метрики кода – связь классов // Microsoft Learn [Электронный ресурс]. URL: <https://learn.microsoft.com/ru-ru/visualstudio/code-quality/code-metrics-class-coupling?view=vs-2022> (дата обращения: 15.05.2024).

Приложение А
(рекомендуемое)
Результаты бизнес-проекта



**Открытые
Решения_**
разработка программно-
обеспечения

> + 7 (8412) 20-06-04
info@osinit.com
www.osinit.com

> ООО «Открытые решения»
Россия, 440000 г. Пенза,
ул. Московская, д. 29
ОГРН 1095835001712
ИНН 5835081619

Исх. № 894 от «31» октября 2022 г.

**В конкурсную комиссию
программы «УМНИК»
фонда содействия развития
малых форм предприятий
в научно-технической сфере**

Настоящим письмом компания ООО «Открытые Решения» в лице Исполнительного директора Бочкарева Игоря Вячеславовича выражает поддержку проекту Александра Кудашова «Разработка системы для исторических квест-экскурсий с применением технологии дополненной реальности».

Проект представляет собой мобильное Android – приложение для городских исторических квест экскурсий, предназначенное для изучения туристами культурно-исторических достопримечательностей и географии города с повышенным уровнем вовлеченности в процесс. В ходе реализации планируется разработка backend- и frontend-части приложения, формирования списка культурно-исторических мест города, а также апробация квестов, занесение исторических справок в приложение. Компания поддерживает участие Проекта в грантовом конкурсе программы «УМНИК» и рекомендует проект к рассмотрению конкурсной комиссией.

Компания готова оказать консультативную и информационную поддержку проекту, а также экспертную поддержку в лице своих сотрудников. Компания готова обеспечить освещение проекта на своих информационных ресурсах: сайт платформы подготовки IT-специалистов – software-academy.ru, группы в социальных сетях – vk.com/osinitcom; t.me/open_solutions.

Исполнительный директор



И.В. Бочкарев



МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Пензенский государственный университет»
(ФГБОУ ВО «ПГУ»)

Красная ул., д.40, г. Пенза, Россия, 440026. Тел/факс: (841-2) 66-62-85
 E-mail: cnit@pnzgu.ru; <http://www.pnzgu.ru>
 ОКПО 02069042, ОГРН 1025801440620, ИНН5837003736 КПП 583701001

Федеральное государственное бюджетное
 учреждение «Фонд содействия развитию малых
 форм предприятий в научно-технической сфере»
 (Фонд содействия инновациям)
 Генеральному директору
 Полякову Сергею Геннадьевичу

Уважаемый Сергей Геннадьевич!

Кафедра «Маркетинг, коммерция и сфера обслуживания» Института экономики и управления ФГБОУ ВО «Пензенский государственный университет» выражает заинтересованность и готовность участия в разработке проекта «Разработка системы для исторических квест-экскурсий с применением технологии дополненной реальности» в рамках программы «УМНИК» (далее Проект), а также его последующей коммерциализации.

Проект представляет собой мобильное Android-приложение для городских исторических квест-экскурсий, предназначенное для популяризации краеведения и формирования желания у туристов и местных жителей в игровой форме знакомиться с культурно-историческими достопримечательностями российских регионов. Пилотная версия мобильного приложения будет разработана для города Пенза. В перспективе возможно тиражирование и на другие населенные пункты Пензенской области и других субъектов РФ. В ходе реализации планируется разработка backend- и frontend- части приложения, формирования списка культурно-исторических мест города, а также квестов, занесение исторических справок в приложение.

Кафедра «Маркетинг, коммерция и сфера обслуживания» Института экономики и управления ФГБОУ ВО «Пензенский государственный университет» поддерживает участие Проекта в грантовом конкурсе программы «УМНИК» и рекомендует проект к рассмотрению конкурсной комиссией.

Кафедра готова оказать экспертную поддержку в лице учёных кафедры, занимающихся исследованием сферы туризма как социально-экономической системы; поддержку в лице студентов направления подготовки 43.03.02 «Туризм» (уровень бакалавриата) - в разработке содержательной части квест-экскурсии.

Кроме того, кафедра готова обеспечить информационную поддержку за счет освещения проекта на своих информационных ресурсах: сайт - https://dep_mkiso.pnzgu.ru/, группа в социальной сети ВКонтакте - <https://vk.com/mkiso>.

/ Директор «ИЭиУ»

В. М. Володин

Заведующий кафедрой «МКиСО»

Л. А. Гамидуллаева



Рисунок А.2 – Письмо поддержки кафедры «Маркетинг, коммерция и сфера обслуживания» Института экономики и управления



**МИНИСТЕРСТВО
КУЛЬТУРЫ И ТУРИЗМА
ПЕНЗЕНСКОЙ ОБЛАСТИ**

ул. Красная, д.71, г. Пенза, 440026
тел. (8412) 56-57-22, факс (8412) 56-63-09
e-mail: uka_po@mail.ru
http://minkult.pnzreg.ru

Генеральному директору
ФГБУ «Фонд содействия развитию
малых форм предприятий в научно-
технической сфере»
Полякову С.Г.
info@fasie.ru

от 14.11.2022 № 15-12/4940
На № _____ от _____

Уважаемый Сергей Геннадьевич!

Министерство культуры и туризма Пензенской области (далее – Министерство) выражает заинтересованность в реализации и продвижении проекта «Разработка системы для исторических квест-экскурсий с применением технологии дополненной реальности» в рамках программы «УМНИК».

Проект представляет собой мобильное Android-приложение для городских исторических квест-экскурсий, предназначенное для изучения туристами культурно-исторических достопримечательностей и географии города с повышенным уровнем вовлеченности в процесс. В ходе реализации планируется разработка backend- и frontend- части приложения, формирования списка культурно-исторических мест города, а также квестов, занесение исторических справок в приложение.

Общая актуальность разрабатываемого продукта обоснована следующими фактами: 2022-ой год – год культурного наследия народов России; Правительство России ведет активную политику по поддержке и развитию отечественного туризма; по данным ВЦИОМ 96% россиян считают важным знание истории своей страны и своего региона; турпоток по городам России по прогнозам на конец 2022 года составит 61 млн человек (на начало августа 2022 г. он уже составил 34 млн чел.). Необходимость внедрения технологии дополненной реальности в проект объясняется увеличением благодаря ей уровня вовлеченности пользователей в экскурсионный процесс и повышением заинтересованности.

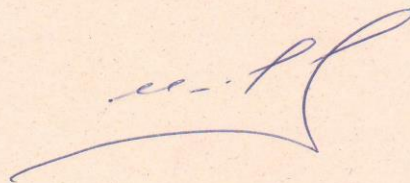
Министерство поддерживает участие Проекта в грантовом конкурсе

Рисунок А.3 – Письмо поддержки Министерства культуры и туризма
Пензенской области (часть 1)

программы «УМНИК» и рекомендует проект к рассмотрению конкурсной комиссией.

Кроме того, Министерство готово оказать консультативную и информационную поддержку проекту, а также обеспечить освещение проекта на официальном сайте Министерства и в социальных сетях.

Врио Министра



И.Н. Мавлюдов

Барышева Екатерина Михайловна
8(8412) 56-07-69

Рисунок А.4 – Письмо поддержки Министерства культуры и туризма
Пензенской области (часть 2)



Рисунок А.5 – Сертификат о прохождении преакселерационной программы
«Дотком инновации»



Рисунок А.6 – Сертификат о прохождении акселерационной программы
Пензенского государственного университета



Рисунок А.7 – Диплом победителя программы «УМНИК»

Приложение Б
(рекомендуемое)
Код программы

Листинг Б.1 – Add_quest.jsx (добавление квеста)

```

import styles from './Add_quest.module.css'
import { Link, useNavigate, redirect } from "react-router-dom"
import { useState, useEffect } from 'react'
import { Formik, Form, Field } from 'formik';
import * as Yup from 'yup';
import YupPassword from 'yup-password'
YupPassword(Yup)
import axios from 'axios'
import { baseURL_users_service, baseURL_quests_service } from
'../../common';
import TextareaAutosize from 'react-textarea-autosize';
import Upload_File from '../../ui/upload_file/Upload_File';

const defaultImageSrc = 'src/assets/images/placeholder_photo.png'

const infoSchema = Yup.object().shape({
  title: Yup.string()
    .min(2, 'Минимум 2 буквы')
    .max(50, 'Максимум 50 букв')
    .matches(/^[?!,.:()"\«\»\-\+а-яА-Яа-зА-З0-9\s]+$/, 'Неверное
название')
    .required('Обязательное поле'),
  description: Yup.string()
    .min(2, 'Минимум 2 буквы')
    .max(1000, 'Максимум 1000 символов')
    .matches(/^[?!,.:()"\«\»\-\+а-яА-Яа-зА-З0-9\s]+$/, 'Неверное
описание')
    .required('Обязательное поле'),
  duration: Yup.number()
    .min(1, 'Не меньше 1')
    .max(100, 'Не больше 100')
    .required('Обязательное поле'),
  distance: Yup.number()
    .min(1, 'Не меньше 1')
    .max(100, 'Не больше 100')
    .required('Обязательное поле'),
  startPoint: Yup.string()
    .min(2, 'Минимум 2 буквы')
    .max(1000, 'Максимум 30 символов')
    .matches(/^[?!,.:()"\«\»\-\+а-яА-Яа-зА-З0-9\s]+$/, 'Неверное
описание')
    .required('Обязательное поле'),
  endPoint: Yup.string()
    .min(2, 'Минимум 2 буквы')
    .max(1000, 'Максимум 1000 символов')
    .matches(/^[?!,.:()"\«\»\-\+а-яА-Яа-зА-З0-9\s]+$/, 'Неверное
описание')
    .required('Обязательное поле'),
  city: Yup.string()
    .required('Обязательное поле')
});

const getCities = baseURL_users_service + '/api/Cities'
const addQuest = baseURL_quests_service + '/api/Quests/create'

const Add_quest = () => {

```

```

const navigate = useNavigate()
const [fieldValues, setFieldValues] = useState({
  title: '',
  description: '',
  duration: 0,
  distance: 0,
  startPoint: '',
  endPoint: '',
  cityId: '',
  mainImageFile: null,
  questMapFile: null,
  additionalFiles: [],
})

const [mainImage, setMainImage] = useState([]);
const [questMapImage, setQuestMapImage] = useState([]);
const [additionalImages, setAdditionalImages] = useState([]);

const [cities, setCities] = useState([])
useEffect(() => {
  axios.get(getCities)
    .then(response => {
      setCities(response.data)
    })
}, [])
return (
  <div className={styles.container}>
    <Formik
      initialValues={{
        title: '',
        city: '',
        description: '',
        mainImageFile: null,
        duration: '',
        distance: '',
        startPoint: '',
        endPoint: '',
        questMapFile: null,
        additionalImages: null,
      }}
      validationSchema={infoSchema}
      onSubmit={(values, {resetForm}) => {
        console.log(fieldValues)
        const images = new FormData();
        images.append('ImgFile',
fieldValues.mainImageFile[0]);
        images.append('ImgMapFile',
fieldValues.questMapFile[0]);
        fieldValues.additionalFiles.forEach(file => {
          images.append('ImgAdditionalFiles',
file)
        })
        if (mainImage.length === 0) {
          alert("Добавьте основное изображение")
        } else if (questMapImage.length === 0) {
          alert("Добавьте карту")
        } else if (additionalImages.length === 0) {

```

```

        alert("Добавьте дополнительные
изображения")
    }
    else {
        axios.post(addQuest, images,
            {
                params: {
                    Title:
fieldValues.title,
                    Description:
fieldValues.description,
                    Duration:
fieldValues.duration,
                    Distance:
fieldValues.distance,
                    StartPoint:
fieldValues.startPoint,
                    EndPoint:
fieldValues.endPoint,
                    CityId:
fieldValues.cityId
                },
            })
        .then((response) => {
            console.log(response)
            const questId =
response.data.id

            sessionStorage.setItem('questId', questId);
            navigate('/add_task')
        })
        .catch((error) =>
console.log(error.response));
    }
    window.scrollTo({top: 0, behavior: 'smooth'})
  }}>
  {({
    values,
    errors,
    touched,
    handleBlur,
    handleChange,
    handleSubmit
  }) => (
    <Form
      onSubmit={handleSubmit}
      className={styles.form}>
      <div className={styles.input_container}>
      <h1 className={styles.addHeader}>Добавить
квест</h1>

      <div className={styles.textFields}>
        {errors.title && touched.title ? (
          <p className={styles.error}>{errors.title}</p>
        ) : <p className={styles.error}></p>}
        <Field name="title"
          placeholder="Название"
          type="text"

```

```

        onChange={ (e) => {
          handleChange(e)
          setFieldValues({
            ...fieldValues,
            title: e.target.value
          })
        }}
        onBlur={handleBlur}
        value={values.title}
        className={styles.questTitle}/>
{errors.city && touched.city ? (
<p className={styles.error}>{errors.city}</p>
) : <p className={styles.error}></p>}
<Field component="select"
  name="city"
  className={styles.dropdown}
  value={values.city}
  onBlur={handleBlur}
  onChange={ (e) => {
    handleChange(e)
    setFieldValues({
      ...fieldValues,
      cityId: e.target.value
    })
  }}>
  <option value='' disabled>Выберите
город</option>
    {cities.map(city => <option
key={city.id} value={city.id}>{city.name}</option>)}
  </Field>
  {errors.description && touched.description ? (
    <p
className={styles.error}>{errors.description}</p>
  ) : <p className={styles.error}></p>}
  <TextareaAutosize name="description"
  placeholder="Описание"
  onChange={ (e) => {
    handleChange(e)
    setFieldValues({
      ...fieldValues,
      description: e.target.value
    })
  }}
  onBlur={handleBlur}
  value={values.description}
  className={styles.questDescription}/>
</div>
<div className={styles.photoUpload}>
<p className={styles.inputLabel}>Основное
изображение</p>
  <Upload_File name="mainImage"
    onBlur={handleBlur}
    value={values.mainImageFile}
    accept=".jpg, .jpeg, .png"
    className={styles.dragNDrop}
    fileList={mainImage}
    setFileList={setMainImage}
  >

```

```

        multiple={false}/>
      </div>
      {errors.duration && touched.duration ? (
        <p
className={styles.error}>{errors.duration}</p>
        ) : <p className={styles.error}></p>}
      <Field name="duration"
        placeholder="Длительность квеста"
        type="number"
        onChange={ (e) => {
          handleChange(e)
          setFieldValues({
            ...fieldValues,
            duration: e.target.value
          })
        }}
        onBlur={handleBlur}
        value={values.duration}
        className={styles.answerField}/>
      {errors.distance && touched.distance ? (
        <p
className={styles.error}>{errors.distance}</p>
        ) : <p className={styles.error}></p>}
      <Field name="distance"
        placeholder="Длина маршрута"
        type="number"
        onChange={ (e) => {
          handleChange(e)
          setFieldValues({
            ...fieldValues,
            distance: e.target.value
          })
        }}
        onBlur={handleBlur}
        value={values.distance}
        className={styles.answerField}/>
      {errors.startPoint && touched.startPoint ? (
        <p
className={styles.error}>{errors.startPoint}</p>
        ) : <p className={styles.error}></p>}
      <Field name="startPoint"
        placeholder="Точка старта квеста"
        type="text"
        onChange={ (e) => {
          handleChange(e)
          setFieldValues({
            ...fieldValues,
            startPoint: e.target.value
          })
        }}
        onBlur={handleBlur}
        value={values.startPoint}
        className={styles.answerField}/>
      {errors.endPoint && touched.endPoint ? (
        <p
className={styles.error}>{errors.endPoint}</p>
        ) : <p className={styles.error}></p>}

```

```

<Field name="endPoint"
  placeholder="Точка окончания квеста"
  type="text"
  onChange={ (e) => {
    handleChange(e)
    setFieldValues({
      ...fieldValues,
      endPoint: e.target.value
    })
  }}
  onBlur={handleBlur}
  value={values.endPoint}
  className={styles.answerField}/>
<div className={styles.photoUpload}>
  <p className={styles.inputLabel}>Карта
маршрута</p>
  <Upload_File name="questMap"
    onBlur={handleBlur}
    value={values.questMapFile}
    accept=".jpg, .jpeg, .png"
    className={styles.dragNDrop}
    fileList={questMapImage}
    setFileList={setQuestMapImage}
    multiple={false}/>
  <p
className={styles.inputLabel}>Дополнительные изображения</p>
  <Upload_File name="additionalImages"
    onBlur={handleBlur}
    value={values.additionalImages}
    accept=".jpg, .jpeg, .png"
    className={styles.dragNDrop}
    fileList={additionalImages}
    setFileList={setAdditionalImages}
    multiple={true}/>
</div>
</div>
<div className={styles.buttons}>
  <button onClick={ () => navigate('/quests')}
className={styles.prevBtn}>Назад</button>
  <button type='submit'
    className={styles.nextBtn}
    onClick={ (e) => {
      setFieldValues({
        ...fieldValues,
        mainImageFile:
mainImage,
        questMapFile:
questMapImage,
        additionalFiles:
additionalImages,
      })
    }}>
    Далее
  </button>
</div>
</Form>
) }

```

```

        </Formik>
    </div>
  )
}

export default Add_quest

```

Листинг Б.2 – Add_task.jsx (добавление задания квеста)

```

import styles from './Add_task.module.css'
import { useState } from 'react'
import { Formik, Form, Field } from 'formik';
import * as Yup from 'yup';
import axios from 'axios'
import { baseURL_quests_service } from '../../common';
import TextareaAutosize from 'react-textarea-autosize';
import Upload_File from '../../ui/upload_file/Upload_File';
import Targets_modal from '../../ui/targets_modal/Targets_modal';
import placeholder_ar from '../../assets/images/placeholder_ar.png'
import placeholder_note from
'../../assets/images/placeholder_note.png'
import Radio_answer from '../../ui/radio_answer/Radio_answer';
import Add_note from '../../ui/add_note/Add_note';
import Checkbox_answer from '../../ui/checkbox_answer/Checkbox_answer';
import End_quest from '../../ui/end_quest/End_quest';

const addFirstQuestion = baseURL_quests_service + '/api/Quests/first'
const addNextQuestion = baseURL_quests_service + '/api/Questions/next'
let requestsCounter = 0

const taskSchema = Yup.object().shape({
  textQuestion: Yup.string()
    .min(2, 'Минимум 2 буквы')
    .max(50, 'Максимум 50 букв')
    .matches(/^[?!,.:()]"«»\-\+а-яА-Яа-зА-З0-9\s]+$/, 'Неверное
название')
    .required('Обязательное поле'),
  description: Yup.string()
    .min(2, 'Минимум 2 буквы')
    .max(1000, 'Максимум 1000 символов')
    .matches(/^[?!,.:()]"«»\-\+а-яА-Яа-зА-З0-9\s]+$/, 'Неверное
описание')
    .required('Обязательное поле'),
  type: Yup.string()
    .required('Обязательное поле'),
  rating: Yup.number()
    .min(1, 'Не меньше 1')
    .max(100, 'Не больше 100')
    .required('Обязательное поле'),
  hint: Yup.string()
    .min(2, 'Минимум 2 буквы')
    .max(100, 'Максимум 100 символов')
    .matches(/^[?!,.:()]"«»\-\+а-яА-Яа-зА-З0-9\s]+$/, 'Неверное
описание')
    .required('Обязательное поле'),
});

```

```

const Add_task = () => {
  const [showModalTarget, setShowModalTarget] = useState(false)
  const [showModalNote, setShowModalNote] = useState(false)
  const [showModalEnd, setShowModalEnd] = useState(false)
  const [photo, setPhoto] = useState([]);
  const [answers, setAnswers] = useState([])
  const [answersFinal, setAnswersFinal] = useState([])
  const [correctAnswers, setCorrectAnswers] = useState([])
  const [answersRadio, setAnswersRadio] = useState([])
  const [correctAnswersRadio, setCorrectAnswersRadio] = useState([])

  const [targetSelected, setTargetSelected] = useState({
    id: '',
    title: '',
  });
  const [note, setNote] = useState({
    id: '',
    title: '',
  });
  const [fieldValues, setFieldValues] = useState({
    textQuestion: '',
    description: '',
    rating: 0,
    imageFile: photo,
    hint: '',
    type: -1,
  })
}

const handleCorrectAnswers = () => {
  const results = []
  const allAnsw = []
  if (fieldValues.type === 1) {
    answers.forEach((item, index) => {
      if (item.correct === true) {
        results.push(index);
      }
      setCorrectAnswers(results)
      allAnsw.push(item.answer)
      setAnswersFinal(allAnsw)
    })
    console.log(allAnsw)
    console.log(results)
  } else if (fieldValues.type === 0) {
    setAnswersFinal(answersRadio)
    setCorrectAnswers(correctAnswersRadio)
  }
}

return (
  <div className={styles.container}>
    <Formik
      initialValues={{
        textQuestion: '',
        description: '',
        imageFile: null,
        type: '',

```

```

        rating: '',
        hint: '',
        textAnswer: '',
    }}
    validationSchema={taskSchema}
    onSubmit={ (values, {resetForm}) => {
        console.log(photo[0])
        const images = new FormData();
        images.append('ImgFile', photo[0]);
        if (correctAnswers.length < 1 ) {
            alert("Выберите правильные ответы")
        } else if (photo.length === 0) {
            alert("Выберите изображение")
        } else {
            const params = new URLSearchParams();

            params.append('TextQuestion', fieldValues.textQuestion)

            params.append('Description', fieldValues.description)

            params.append('Rating', fieldValues.rating)
            params.append('Hint', fieldValues.hint)
            params.append('Type', fieldValues.type)
            answersFinal.forEach(answer => {
                params.append('Answers', answer)
            })
            correctAnswers.forEach(answer => {
                params.append('CorrectAnswers',
answer)
            })

            params.append('QuestId', sessionStorage.getItem('questId'))
            if (Object.keys(targetSelected).length
!= 0) {

                params.append('TargetId', targetSelected.id)
            } else if (Object.keys(note).length !=
0) {

                params.append('NoteId', note.id)
            } else {
                null
            }
            console.log(params.toString())
            const postQuestion =
baseUrl_quests_service + '/api/Questions/create?' + params.toString()
            axios.post(postQuestion, images)
                .then(response => {
                    console.log(response)
                    requestsCounter++
                    const questionId = response.data.id
                    if (requestsCounter == 1) {
                        axios.put(addFirstQuestion,
null,
{params: {
                        questId:
sessionStorage.getItem('questId'),

```

```

                                firstQuestionId:
questionId
                                }
                                }
                                )
                                .then((response) => {
                                    console.log(response)
                                })
                                .catch((error) => {

                                console.log(error.response)

                                })
                                } else {
                                axios.put(addNextQuestion,
null,
                                {params: {
                                    id:
sessionStorage.getItem('prevQuestionId'),
                                    nextQuestionId:
questionId
                                }
                                }
                                )
                                .then((response) => {
                                    console.log(response)
                                })
                                .catch((error) =>
console.log(error.response))
                                }

                                sessionStorage.setItem('prevQuestionId', questionId)
                                })
                                .catch((error) =>
console.log(error.response));

                                }
                                resetForm({})
                                setPhoto([])
                                setTargetSelected({})
                                setNote({})
                                window.scrollTo({top: 0, behavior: 'smooth'})
                                }}>
                                {{
                                values,
                                errors,
                                touched,
                                handleBlur,
                                handleChange,
                                handleSubmit
                                }) => (
<Form
                                onSubmit={handleSubmit}
                                className={styles.form}>
                                <div className={styles.input_container}>
                                <h1 className={styles.addHeader}>Добавить
задание</h1>

                                <div className={styles.textFields}>

```

```

{errors.textQuestion && touched.textQuestion ?
(
    <p
className={styles.error}>{errors.textQuestion}</p>
) : <p className={styles.error}></p>
<Field name="textQuestion"
placeholder="Вопрос"
type="text"
onChange={(e) => {
handleChange(e)
setFieldValues({
...fieldValues,
textQuestion: e.target.value
})
}}
onBlur={handleBlur}
value={values.textQuestion}
className={styles.questTitle}/>
{errors.description && touched.description ? (
<p
className={styles.error}>{errors.description}</p>
) : <p className={styles.error}></p>
<TextareaAutosize name="description"
placeholder="Описание"
onChange={(e) => {
handleChange(e)
setFieldValues({
...fieldValues,
description: e.target.value
})
}}
onBlur={handleBlur}
value={values.description}
className={styles.questDescription}/>
{errors.type && touched.type ? (
<p className={styles.error}>{errors.type}</p>
) : <p className={styles.error}></p>
<p className={styles.inputLabel}>Выберите тип
ответа</p>
<Field component="select"
name="type"
className={styles.dropdown}
value={values.type}
onBlur={handleBlur}
onChange={(e) => {
handleChange(e);
if (e.target.value === '0') {
setFieldValues({
...fieldValues,
type: 0,
})
} else if (e.target.value === '1')
setFieldValues({
...fieldValues,
type: 1,
})
}}

```

```

        } else if (e.target.value === '2')
    {
        setFieldValues({
            ...fieldValues,
            type: 2,
        })
    }
    }}
    >
    <option value='' disabled>Выберите тип
задания</option>
    <option value='0'>Один вариант
ответа</option>
    <option value='1'>Несколько вариантов
ответа</option>
    <option value='2'>Текстовый
ответ</option>
</Field>
</div>
{
    fieldValues.type === '' ? null : null
}
{
    fieldValues.type === 0 ? (
        <>
        <p
className={styles.inputLabel}>Введите ответы</p>
        <Radio_answer
            answersRadio={answersRadio}

            setAnswersRadio={setAnswersRadio}

            correctAnswersRadio={correctAnswersRadio}

            setCorrectAnswersRadio={setCorrectAnswersRadio}/>
        </>
        ) : null
    }
    {
        fieldValues.type === 1 ? (
            <>
            <p
className={styles.inputLabel}>Введите ответы</p>
            <Checkbox_answer
                answers={answers}
                setAnswers={setAnswers}/>
            </>
            ) : null
        }
        {
            fieldValues.type === 2 ? (
                <>
                <p
className={styles.inputLabel}>Введите ответ</p>
                <Field name="textAnswer"
                    placeholder="Введите ответ"
                    type="text"

```

```

                                onChange={ (e) => {
                                    handleChange(e)

setAnswersFinal([e.target.value])

                                setCorrectAnswers([0])
                                }}
                                onBlur={handleBlur}
                                value={values.textAnswer}
                                className={styles.answerField}>
                                </Field>
                                </>
                                ) : null
                                }

                                {errors.rating && touched.rating ? (
                                <p
className={styles.error}>{errors.rating}</p>
                                ) : <p className={styles.error}></p>}
                                <p className={styles.inputLabel}>Количество
баллов за задание</p>

                                <Field name="rating"
                                placeholder="Количество баллов за
задание"

                                type="number"
                                onChange={ (e) => {
                                    handleChange(e)
                                    setFieldValues({
                                        ...fieldValues,
                                        rating: e.target.value
                                    })
                                }}
                                onBlur={handleBlur}
                                value={values.rating}
                                className={styles.answerField}/>
                                {errors.hint && touched.hint ? (
                                <p className={styles.error}>{errors.hint}</p>
                                ) : <p className={styles.error}></p>}
                                <p className={styles.inputLabel}>Подсказка</p>
                                <Field name="hint"
                                placeholder="Подсказка"
                                type="text"
                                onChange={ (e) => {
                                    handleChange(e)
                                    setFieldValues({
                                        ...fieldValues,
                                        hint: e.target.value
                                    })
                                }}
                                onBlur={handleBlur}
                                value={values.hint}
                                className={styles.answerField}/>
                                <div className={styles.photoUpload}>
                                <p className={styles.inputLabel}>Добавить
изображение</p>

                                <Upload_File name="photo"
                                onChange={handleChange}
                                onBlur={handleBlur}

```

```

        value={values.imageFile}
        accept=".jpg, .jpeg, .png"
        className={styles.dragNDrop}
        fileList={photo}
        setFileList={setPhoto}
        multiple={false}/>
      </div>
      <p className={styles.inputLabel}>Добавить
таргет</p>
        <button type='button'
className={styles.add_btn} onClick={() => setShowModalTarget(true)}>
          <img src={placeholder_ar}/>
        </button>
        {showModalTarget && <Targets_modal
closeModalTarget={() => setShowModalTarget(false)}
targetSelected={targetSelected} setTargetSelected={setTargetSelected}/>}
        {targetSelected !== '' ? (<p
className={styles.targetAdded}>{targetSelected.title}</p>) : null}

        <p className={styles.inputLabel}>Добавить
историческую справку</p>
        <button type='button'
className={styles.add_btn} onClick={() => setShowModalNote(true)}>
          <img src={placeholder_note}/>
        </button>
        {showModalNote && <Add_note
closeModalNote={() => setShowModalNote(false)} note={note}
setNote={setNote}/>}
        {note !== '' ? (<p
className={styles.targetAdded}>{note.title}</p>) : null}
      </div>
      <div className={styles.btns}>
        <button type='button' onClick={() =>
setShowModalEnd(true)} className={styles.prevBtn}>Завершить
добавление</button>
        {showModalEnd && <End_quest
closeModalEnd={() => setShowModalEnd(false)}>}
        <button type='submit'
className={styles.nextBtn}
onClick={handleCorrectAnswers}>Добавить вопрос</button>
      </div>
    </Form>
  )}
</Formik>
</div>

)
}

export default Add_task

```

Листинг Б.3 – Users.jsx (страница управления пользователями)

```

import styles from './Users.module.css'
import Menu_admin from '../../ui/menu_admin/Menu_admin'

```

```

import { useState, useEffect } from 'react'
import { baseURL_users_service } from '../../common.js'
import Add_user from '../../ui/add_user/Add_user.jsx'
import Edit_user from '../../ui/edit_user/Edit_user.jsx'
import axios from 'axios'
import Delete_user from '../../ui/delete_user/Delete_user.jsx'
import Edit_role from '../../ui/edit_role/Edit_role.jsx'

const getUsers = baseURL_users_service + '/api/Users'

const Users = () => {
  const [userData, setUserData] = useState([]);
  const [showModalAdding, setShowModalAdding] = useState(false)
  const [showModalEditing, setShowModalEditing] = useState(false)
  const [showModalDeleting, setShowModalDeleting] = useState(false)
  const [showModalRole, setShowModalRole] = useState(false)
  const [userState, setUserState] = useState({
    name: '',
    surname: '',
    email: '',
    login: '',
    password: '',
    role: -1,
  })
  const [flag, setFlag] = useState(false)

  useEffect(() => {
    axios.get(getUsers)
      .then(response => {
        setUserData(response.data)
        setFlag(true)
      })
      .catch((error) => console.log(error.response));
  }, [])

  const handleCallback = () => {
    axios.get(getUsers)
      .then(response => {
        setUserData(response.data)
      })
      .catch((error) => console.log(error.response));
  };

  return (
    <div className={styles.container}>
      <div className={styles.menu}>
        <Menu_admin />
      </div>
      <div className={styles.contentContainer}>
        {flag && userData.length ?
          <>
            <table className={styles.table}>
              <thead>
                <tr>
                  <th>Имя</th>
                  <th>Фамилия</th>
                  <th>Логин</th>

```

```

        <th>Email</th>
        <th>Пароль</th>
        <th>Роль</th>
        <th>Удалить пользователя</th>
    </tr>
</thead>
<tbody>
{userData.length ? userData.map((user) => {
    let userRole
    if (user.role === 0) {
        userRole = 'Администратор'
    } else if (user.role === 1) {
        userRole = 'Создатель квестов'
    } else if (user.role === 2) {
        userRole = 'Пользователь'
    }
    return (
    <tr className={styles.dataRow}
        key={user.id}
        onClick={() => {
            setShowModalEditing(true)
            setUserState({
                name: user.name,
                surname: user.surname,
                email: user.email,
                login: user.login,
                password: user.password,
                role: user.role,
            })
        }}>
        <td>{user.name}</td>
        <td>{user.surname}</td>
        <td>{user.login}</td>
        <td>{user.email}</td>
        <td>{user.password}</td>
        <td className={styles.editRoleCell}>
        {
            localStorage.getItem('login') !== user.login ? (
                <>
                <button className={styles.editRole}
                    onClick={() => {
                        setShowModalRole(true)
                        setUserState({
                            name: user.name,
                            surname: user.surname,
                            email: user.email,
                            login: user.login,
                            password: user.password,
                            role: user.role,
                        })
                    }}>
                {userRole}
                <svg width="24" height="24" viewBox="0 0 24
24" fill="none" xmlns="http://www.w3.org/2000/svg">
                <path d="M14.9412 4.55138L18.4706
8.07609M12.5882 21H22M3.17647 16.3004L2 21L6.70588 19.8251L20.3365
6.21269C20.7776 5.77204 21.0254 5.17447 21.0254 4.55138C21.0254 3.9283

```

```

20.7776 3.33073 20.3365 2.89007L20.1341 2.68799C19.6929 2.24747 19.0945 2
18.4706 2C17.8467 2 17.2483 2.24747 16.8071 2.68799L3.17647 16.3004Z"
stroke="#333333" strokeLinecap="round" strokeLinejoin="round"/>
    </svg>
  </button>
    {showModalRole && userState && <Edit_role
user={userState} closeModalRole={() => setShowModalRole(false)}
callback={handleCallback}/>}
  </>
  ) : <p>{userRole}</p>
}

</td>
<td>
  <button className={styles.delete_btn}
    onClick={() => {
      setShowModalDeleting(true)
      setUserState({
        name: user.name,
        surname: user.surname,
        email: user.email,
        login: user.login,
        password: user.password,
        role: user.role,
      })
    }}>
    <svg width="25px" height="25px" viewBox="0 0
24 24" fill="none" xmlns="http://www.w3.org/2000/svg">
      <path d="M10 11V17" stroke="#333333"
strokeWidth="2" strokeLinecap="round" strokeLinejoin="round"/>
      <path d="M14 11V17" stroke="#333333"
strokeWidth="2" strokeLinecap="round" strokeLinejoin="round"/>
      <path d="M4 7H20" stroke="#333333"
strokeWidth="2" strokeLinecap="round" strokeLinejoin="round"/>
      <path d="M6 7H12H18V18C18 19.6569 16.6569 21
15 21H9C7.34315 21 6 19.6569 6 18V7Z" stroke="#333333" strokeWidth="2"
strokeLinecap="round" strokeLinejoin="round"/>
      <path d="M9 5C9 3.89543 9.89543 3 11
3H13C14.1046 3 15 3.89543 15 5V7H9V5Z" stroke="#333333" strokeWidth="2"
strokeLinecap="round" strokeLinejoin="round"/>
    </svg>
  </button>
    {showModalDeleting && userState && <Delete_user
user={userState} closeModalDeleting={() => setShowModalDeleting(false)}
callback={handleCallback}/>}
  </td>
</tr>
)
}
)
: null
}
    {showModalEditing && userState && <Edit_user
user={userState} closeModalEditing={() => setShowModalEditing(false)}
callback={handleCallback}/>}
  </tbody>
</table>

```

```

        </>
        : <p className={styles.message}>Нет пользователей</p>
    }
    <button className={styles.add_btn} onClick={() =>
setShowModalAdding(true)}>+</button>
    {showModalAdding && <Add_user closeModalAdding={() =>
setShowModalAdding(false)} callback={handleCallback}/>}
    </div>
  </div>
)
}

export default Users

```

Листинг Б.4 – Add_user.jsx (Добавление пользователя)

```

import styles from './Add_user.module.css'
import { useState } from 'react';
import { baseURL_users_service } from '../../common.js'
import axios from 'axios'
import { Formik, Form, Field } from 'formik';
import * as Yup from 'yup';
import YupPassword from 'yup-password'
YupPassword(Yup)

const postUsers = baseURL_users_service + '/api/Users/user'

const userSchema = Yup.object().shape({
  name: Yup.string()
    .min(2, 'Минимум 2 буквы')
    .max(20, 'Максимум 20 букв')
    .matches(/^[А-Яа-я]+$/, 'Неверное имя (только буквы)')
    .required('Обязательное поле'),
  surname: Yup.string()
    .min(2, 'Минимум 2 буквы')
    .max(20, 'Максимум 20 букв')
    .matches(/^[А-Яа-я]+$/, 'Неверная фамилия (только буквы)')
    .required('Обязательное поле'),
  login: Yup.string()
    .min(2, 'Минимум 2 буквы')
    .max(20, 'Максимум 20 букв')
    .matches(/^[a-zA-Z][a-zA-Z0-9-_\.\.]{1,20}$/, 'Неверный логин
(латинские буквы, цифры, нижнее подчеркивание и дефис)')
    .required('Обязательное поле'),
  email: Yup.string()
    .email('Неверный email')
    .required('Обязательное поле'),
  password: Yup.string()
    .password()
    .min(8, 'Минимум 8 символов')
    .max(20, 'Максимум 20 символов')
    .minSymbols(1, 'Минимум 1 специальный символ')
    .minLowercase(1, 'Минимум 1 латинская буква в нижнем
регистре')
    .minUppercase(1, 'Минимум 1 латинская буква в верхнем
регистре')
    .minNumbers(1, 'Минимум 1 цифра')

```

```

        // .matches (/^[0-9-_\.] {1,20}$/, 'Минимум 1 цифра')
        .required('Обязательное поле'),
    role: Yup.string()
        .required('Обязательное поле')
    });

const Add_user = ({ closeModalAdding, callback }) => {
    const [roleRequest, setRoleRequest] = useState(-1)
    const [usersData, setUsersData] = useState({
        name: '',
        surname: '',
        email: '',
        login: '',
        password: '',
        role: -1,
    });

    return (
        <div className={styles.container}>
            <button onClick={closeModalAdding}
className={styles.closeBtn}>X</button>
            <p className={styles.header}>Добавить пользователя</p>
            <Formik
                initialValues={{
                    name: '',
                    surname: '',
                    login: '',
                    email: '',
                    password: '',
                    role: '',
                }}
                validationSchema={userSchema}
                onSubmit={(values, {resetForm}) => {
                    axios.post(postUsers, usersData)
                        .then((response) => {
                            console.log(response)
                            callback();
                            alert("Пользователь " + usersData.name + " " +
usersData.surname + " добавлен")
                            closeModalAdding()
                        })
                        .catch((error) => console.log(error.response));

                    resetForm({})
                }}
            >
                {({
                    values,
                    errors,
                    touched,
                    handleBlur,
                    handleChange,
                    handleSubmit,
                }) => (
                    <Form
                        onSubmit={handleSubmit}
                        className={styles.form}>
                        {errors.name && touched.name ? (

```

```

<p className={styles.error}>{errors.name}</p>
) : <p className={styles.error}></p>
<Field name="name"
  placeholder="Имя"
  type="text"
  onChange={handleChange}
  value={values.name}/>
{errors.surname && touched.surname ? (
<p className={styles.error}>{errors.surname}</p>
) : <p className={styles.error}></p>}
<Field name="surname"
  placeholder="Фамилия"
  type="text"
  onChange={handleChange}
  onBlur={handleBlur}
  value={values.surname}/>
{errors.login && touched.login ? (
<p className={styles.error}>{errors.login}</p>
) : <p className={styles.error}></p>}
<Field name="login"
  placeholder="Логин"
  type="text"
  onChange={handleChange}
  onBlur={handleBlur}
  value={values.login}/>
{errors.email && touched.email ? (
<p className={styles.error}>{errors.email}</p>
) : <p className={styles.error}></p>}
<Field name="email"
  placeholder="Email"
  type="text"
  onChange={handleChange}
  onBlur={handleBlur}
  value={values.email}/>
{errors.password && touched.password ? (
<p className={styles.error}>{errors.password}</p>
) : <p className={styles.error}></p>}
<Field name="password"
  placeholder="Пароль"
  type="password"
  onChange={handleChange}
  onBlur={handleBlur}
  value={values.password}/>
{errors.role && touched.role ? (
<p className={styles.error}>{errors.role}</p>
) : <p className={styles.error}></p>}
<Field component="select"
  name="role"
  className={styles.dropdown}
  value={values.role}
  onBlur={handleBlur}
  onChange={(e) => {
    handleChange(e)
    if (e.target.value === '0') {
      setRoleRequest(0)
    } else if (e.target.value === '1') {
      setRoleRequest(1)
    }
  }}
  >

```

```

        } else if (e.target.value === '2') {
            setRoleRequest(2)
        }
    }}
    >
    <option value='' disabled>Выберите роль</option>
    <option value='0'>Администратор</option>
    <option value='1'>Создатель квестов</option>
    <option value='2'>Пользователь</option>
</Field>
<button type="submit"
    className={styles.addBtn}
    onClick={() => {
        setUsersData({
            ...values,
            role: roleRequest
        })
    }}>
    Добавить
</button>
</Form>
)}
</Formik>
</div>
)
}
export default Add_user

```

Листинг Б.5 – Delete_target.jsx (удаление таргета)

```

import styles from './Delete_target.module.css'
import { useState, useEffect } from 'react';
import { baseURL_requests_service } from '../../common.js'
import axios from 'axios'

const Delete_target = ({target, closeModalDeleting, callback }) => {
    const [targetId, setTargetId] = useState('')

    const getTargetById = baseURL_requests_service + '/api/Targets/id/' +
target.id
    useEffect(() => {
        axios.get(getTargetById, {
            params: {
                id: target.id
            }
        })
        .then(response => {
            setTargetId(response.data.id)
        })
    }, [])

    const deleteTarget = baseURL_requests_service + '/api/Targets/' +
targetId
    const handleClick = () => {
        axios.delete(deleteTarget)
        .then(response => {
            console.log(response)

```

```

        callback();
        alert("Taprer " + target.title + " " + " удален")
        closeModalDeleting()
    })
    .catch((error) => console.log(error.response));
}
return (
    <div className={styles.container}>
        <button onClick={closeModalDeleting}
className={styles.closeBtn}>X</button>
        <p className={styles.header}>Удалить таргет
{target.title}?</p>
        <div className={styles.deletingBtns}>
            <button onClick={() => handleClick()}
className={styles.yesBtn}>Да</button>
            <button onClick={closeModalDeleting}
className={styles.noBtn}>Нет</button>
        </div>
    </div>
)
}
export default Delete_target

```

Листинг Б.6 – Edit_role.jsx (редактирование роли пользователя)

```

import styles from './Edit_role.module.css'
import { useState, useEffect } from 'react';
import { baseURL_users_service } from '../../common.js'
import axios from 'axios'
import { Formik, Form, Field } from 'formik';
import * as Yup from 'yup';
import YupPassword from 'yup-password'

const editRole = baseURL_users_service + '/api/Users/role'

const roleSchema = Yup.object().shape({
    role: Yup.string()
        .required('Обязательное поле')
});

const Edit_role = ({user, closeModalRole, callback }) => {
    const [userId, setUserId] = useState('')
    const [roleRequest, setRoleRequest] = useState(-1)
    const [usersData, setUsersData] = useState({});
    const [initialData, setInitialData] = useState(
        {
            id: '',
            role: user.role,
        }
    )

    const getUserByLogin = baseURL_users_service + '/api/Users/login/'
    + user.login
    useEffect(() => {
        axios.get(getUserByLogin, {
            params: {

```

```

        login: user.login
      }
    })
    .then(response => {
      setUserId(response.data.id)
    })
    .catch((error) => console.log(error.response));
  }, [])

  return (
    <div className={styles.container}>
      <button onClick={closeModalRole}
className={styles.closeBtn}>X</button>
      <p className={styles.header}>Изменить роль {user.login}</p>
      <Formik
        initialValues={{
          id: '',
          role: user.role,
        }}
        validationSchema={roleSchema}
        onSubmit={(values, {resetForm}) => {
          if (usersData.role !== initialData.role) {
            axios.put(editRole,
              null,
              {params: {
                id: userId,
                role: usersData.role
              }})
              .then((response) => {
                closeModalRole()
                callback();
              })
              .catch((error) => console.log(error.response))
            resetForm({
              values: {
                id: '',
                name: '',
                surname: '',
                login: '',
                email: '',
                password: '',
                role: '',
              },
            });
          } else {
            alert("Ничего не изменилось")
            closeModalRole()
          }
        }}>
      {{{
        values,
        errors,
        touched,
        handleBlur,
        handleChange,
        handleSubmit
      }}} => (

```

```

    <Form
      onSubmit={handleSubmit}
      className={styles.form}>
        {errors.role && touched.role ? (
          <p className={styles.error}>{errors.role}</p>
        ) : <p className={styles.error}></p>}
        <Field component="select"
          name="role"
          className={styles.dropdown}
          value={values.role}
          onBlur={handleBlur}
          onChange={(e) => {
            handleChange(e)
            if (e.target.value === '0') {
              setRoleRequest(0)
            } else if (e.target.value === '1') {
              setRoleRequest(1)
            } else if (e.target.value === '2') {
              setRoleRequest(2)
            }
          }}>
          <option value='' disabled>Выберите
роль</option>
          <option value='0'>Администратор</option>
          <option value='1'>Создатель квестов</option>
          <option value='2'>Пользователь</option>
        </Field>
        <button type="submit"
          className={styles.editBtn}
          onClick={() => {
            if (roleRequest !== -1) {
              setUsersData({
                id: userId,
                role: roleRequest
              }) else {
                setUsersData({
                  id: userId,
                  role: user.role
                })
              }
            }
          }}>
          Сохранить изменения
        </button>
      </Form>
    )}
  </Formik>
</div>
)
}
export default Edit_role

```

Листинг Б.7 – Registration_stat.jsx (статистика регистрации)

```

import styles from './Registration_stat.module.css'
import { useState, useEffect } from 'react'
import { baseURL_users_service } from '../../common'
import axios from 'axios'

```

```

import Scatter_chart from '../..charts/scatter_chart/Scatter_chart';

let period = 0
const Registration_stat = () => {
  const [periodAxis, setPeriodAxis] = useState('Час')
  const [registrations, setRegistration] = useState([])
  const [chartName, setChartName] = useState('Количество регистраций
за последние сутки')
  const getRegistrationStatDefault = baseURL_users_service +
'/api/Statistic/registrations?period=' + period
  useEffect(() => {
    axios.get(getRegistrationStatDefault)
      .then(response => {
        setRegistration(response.data)
      })
      .catch((error) => console.log(error.response));
  }, [])

  const handleDropdown = (e) => {
    if (e.target.value === '0') {
      period = 0
      setChartName('Количество регистраций за последние
сутки')
      setPeriodAxis('Час')
    } else if (e.target.value === '1') {
      period = 1
      setChartName('Количество регистраций за последнюю
неделю')
      setPeriodAxis('Дата')
    } else if (e.target.value === '2') {
      period = 2
      setChartName('Количество регистраций за месяц')
      setPeriodAxis('Дата')
    }
    const getRegistrationStatByPeriod = baseURL_users_service +
'/api/Statistic/registrations?period=' + period
    axios.get(getRegistrationStatByPeriod)
      .then(response => {
        setRegistration(response.data)
      })
      .catch((error) => console.log(error.response));
  }

  return (
    <div className={styles.container}>
      <div className={styles.period}>
        <p className={styles.statTitle}>Регистрация
пользователей</p>
        <select onChange={(e) => {handleDropdown(e)}}
className={styles.dropdown}>
          <option value="">Выберите период</option>
          <option value="0">День</option>
          <option value="1">Неделя</option>
          <option value="2">Месяц</option>
        </select>
      </div>
    </div>
  )
}

```

```

        <Scatter_chart param={registrations}
chartName={chartName} periodAxis={periodAxis}/>
      </div>

    </div>

  )
}

```

```
export default Registration_stat
```

Листинг Б.8 – Quests_stat.jsx (страница квестов)

```

import styles from './Quests_stat.module.css'
import { useState, useEffect } from 'react'
import { baseURL_requests_service } from '../../common'
import axios from 'axios'
import Line_chart from '../../charts/line_chart/Line_chart';

let period = 0
const Quests_stat = () => {
  const [quests, setQuests] = useState([])
  const [chartName, setChartName] = useState('Количество квестов за последние сутки')
  const getQuestsStatDefault = baseURL_requests_service +
  '/api/Statistic/quests?period=' + period
  useEffect(() => {
    axios.get(getQuestsStatDefault)
      .then(response => {
        setQuests(response.data)
        console.log(response)
      })
      .catch((error) => console.log(error.response));
  }, [])

  const handleDropdown = (e) => {
    if (e.target.value === '0') {
      period = 0
      setChartName('Количество квестов за последние сутки')
    } else if (e.target.value === '1') {
      period = 1
      setChartName('Количество квестов за последнюю неделю')
    } else if (e.target.value === '2') {
      period = 2
      setChartName('Количество квестов за месяц')
    }
    const getQuestsStatByPeriod = baseURL_requests_service +
    '/api/Statistic/quests?period=' + period
    axios.get(getQuestsStatByPeriod)
      .then(response => {
        setQuests(response.data)
      })
      .catch((error) => console.log(error.response));
  }

  return (
    <div className={styles.container}>

```

```

        <div className={styles.period}>
          <p className={styles.statTitle}>Все квесты</p>
          <select onChange={(e) => {handleDropdown(e)}}
className={styles.dropdown}>
            <option value="">Выберите период</option>
            <option value="0">День</option>
            <option value="1">Неделя</option>
            <option value="2">Месяц</option>
          </select>
        </div>
      <div>
        <Line_chart param={quests} chartName={chartName}/>
      </div>
    </div>

  )
}
export default Quests_stat

```

Листинг Б.9 – Area_chart.jsx (график с областью)

```

import styles from './Area_chart.module.css'
import React from "react";
import {
  AreaChart,
  Area,
  XAxis,
  YAxis,
  CartesianGrid,
  Tooltip,
  Legend
} from "recharts";

const Area_chart = ({ param, chartName }) => {
  return (
    <div className={styles.container}>
      <AreaChart
        width={600}
        height={400}
        data={param}
        margin={{
          top: 10,
          right: 30,
          left: 0,
          bottom: 0
        }}
        style={{
          fontSize: '1rem',
          fontFamily: 'Noto Sans',
          fontStyle: 'normal',
        }}
      >
        <CartesianGrid strokeDasharray="3 3" />
        <XAxis dataKey="name" />
        <YAxis />
        <Tooltip />
        <Legend />
      </AreaChart>
    </div>
  )
}

```

```

        <Area
          type="monotone"
          dataKey="uv"
          stroke="#8884d8"
          fill="#8884d8"
          name={chartName}/>
      </AreaChart>
    </div>
  );
}

```

```
export default Area_chart
```

Листинг Б.10 – Checkbox_answer.jsx (ответ с выбором нескольких вариантов)

```

import styles from './Checkbox_answer.module.css'
import { useState } from 'react'
import { Formik, Field } from 'formik';
import * as Yup from 'yup';

const answerSchema = Yup.object().shape({
  answer: Yup.string()
    .min(2, 'Минимум 2 буквы')
    .max(50, 'Максимум 50 букв')
    .matches(/^[?!,.:()\"'«»\-\+а-яА-Яа-зА-Z0-9\s]+$/, 'Неверное
название')
    .required('Обязательное поле'),
});

const Checkbox_answer = ({ answers, setAnswers }) => {
  const [userInput, setUserInput] = useState("");
  const addAnswer = (e) => {
    e.preventDefault();
    const trimmedUserInput = userInput.trim();
    if (trimmedUserInput) {
      if (!answers.some(item => item.answer === trimmedUserInput))
      {
        setAnswers((existingItems) => [
          ...existingItems,
          { answer: trimmedUserInput, correct: false },
        ]);
        setUserInput("");
      } else {
        alert("Такой уже есть")
        setUserInput("");
      } else {
        alert("Введите значение")
      }
    }
  };

  const toggleAnswer = (index) => {
    setAnswers((existingItems) =>
      existingItems.map((item, i) =>
        index === i ? { ...item, correct: !item.correct } : item
      )
    );
  };
};

```

```

const deleteAnswer = (index) => {
  setAnswers((existingItems) =>
    existingItems.filter((item, i) => index !== i)
  );
};

return (
  <div className={styles.container}>
    <Formik
      initialValues={{
        answer: '',
        checkbox: false,
      }}
      validationSchema={answerSchema}>
      {({
        values,
        errors,
        touched,
        handleBlur,
        handleChange,
      }) => (
        <div className={styles.input_container}>
          {errors.answer || touched.answer ? (
            <p className={styles.error}>{errors.answer}</p>
          ) : <p className={styles.error}></p>}
          <div className={styles.addAnswer}>
            <Field name="answer"
              placeholder="Ответ"
              value={userInput}
              type="text"
              onChange={(e) => {
                handleChange(e)
                setUserInput(e.target.value)
              }}
              onBlur={handleBlur}
              className={styles.answerField}/>
            <button type="button" onClick={addAnswer}
              className={styles.addBtn}> Добавить ответ </button>
          </div>
          {answers.map((item, index) => (
            <label key={index + item.answer}
              className={styles.answer}>
                <Field type="checkbox"
                  name="checkbox"
                  checked={item.correct}
                  className={styles.checkbox}
                  onChange={(e) => {
                    handleChange(e)
                    toggleAnswer(index)
                  }}/>
                <span className={styles.customCheckbox}/>
                <span className={styles.itemText}>
                  {item.answer}
                </span>
              </label>
            )
          )}
        </div>
      )}
    </Formik>
  </div>
)

```

```

        <button type='button'
            onClick={() => deleteAnswer(index)}

className={styles.removeBtn}>X</button>
        </label>
    )))
</div>
    ))
</Formik>
</div>
)
}

export default Checkbox_answer

```

Листинг Б.11 – Quest_card_small.jsx (карточка квеста)

```

import styles from './Quest_card_small.module.css'
import { Link } from "react-router-dom"
import { useState } from 'react'
import Delete_quest from '../delete_quest/Delete_quest'
import Quest_stat from '../screens/quest_stat/Quest_stat'

const Quest_card_small = ({ quest, handleCallback }) => {
const [showModalDeleting, setShowModalDeleting] = useState(false)
const [showModalStat, setShowModalStat] = useState(false)
const [questId, setQuestId] = useState('')

const handleDelete = () => {
    setShowModalDeleting(true)
    setQuestId(quest.id)
}

const handleStat = () => {
    setShowModalStat(true)
    setQuestId(quest.id)
}

return (
    <div className={styles.item}>
        <div className={styles.main}>
            <div
                className={styles.image}
                style = {{
                    backgroundImage: `url(${quest.imgName})`,
                }} />
            <div className={styles.itemDel}>
                <button
                    className={styles.itemDelBtn}
                    onClick={handleDelete}>
                    X
                </button>
                {showModalDeleting && <Delete_quest
questId={questId} closeModalDeleting={() => setShowModalDeleting(false)}
callback={handleCallback}/>}
            </div>

```

```

    </div>

    <h4 className={styles.questTitle}>
      <Link to={` /quest/${quest.id}`}>{quest.title}</Link>
    </h4>
    <p className={styles.questDescription}>{quest.description}</p>
    <div className={styles.buttons}>
      <button className={styles.edit_btn}>
        <Link to={` /edit_quest/${quest.id}`}>
          Редактировать
        </Link>
      </button>
      <button className={styles.stat_btn}>
        <Link to={` /quest_stat/${quest.id}`}>
          Статистика
        </Link>
      </button>
    </div>
  </div>
)
}

export default Quest_card_small

```

Листинг Б.12 – Menu_admin.jsx (боковое меню)

```

import './Menu_admin.css'
import logo_admin from '/app/src/assets/images/logo_admin.png'
import { NavLink } from 'react-router-dom'
import { useAuth } from '../../provider/AuthProvider'
import { jwtDecode } from 'jwt-decode'

const Menu_admin = () => {
  const { token } = useAuth();

  return (
    <div className="container" id="Container">
      <div className="logo">
        <img src={logo_admin} className="logo_img"/>
        <h1 className="questie">Questie</h1>
        <p>dashboard</p>
      </div>
      <NavLink to="/stat" className="menu_item">
        <svg width="24" height="21" viewBox="0 0 24 21"
fill="none" xmlns="http://www.w3.org/2000/svg">
          <path d="M1 1V15.25M23 20H1V15.25M1 15.25L6.78947
9.3125L12.5789 15.25L21.8421 6.9375" stroke="#C0C0C0" strokeWidth="1.5"
strokeLinecap="round" strokeLinejoin="round"/>
        </svg>
        Статистика
      </NavLink>
      <NavLink to="/quests" className="menu_item">
        <svg width="24" height="24" viewBox="0 0 24 24"
fill="none" xmlns="http://www.w3.org/2000/svg">
          <path d="M8.6 19.7C8.6 21.5225 7.12254 23 5.3 23C3.47746
23 2 21.5225 2 19.7C2 17.8775 3.47746

```

```

16.4 5.3 16.4C7.12254 16.4 8.6 17.8775 8.6
19.7ZM8.6 19.7H17.95C18.9711 19.7 19.9503
19.2944 20.6724 18.5724C21.3944 17.8503 21.8
16.8711 21.8 15.85C21.8 14.8289 21.3944
13.8497 20.6724 13.1276C19.9503 12.4056
18.9711 12 17.95 12H5.85C4.82892 12 3.84965
11.5944 3.12764 10.8724C2.40562 10.1503 2
9.17108 2 8.15C2 7.12892 2.40562 6.14965
3.12764 5.42764C3.84965 4.70562 4.82892 4.3
5.85 4.3H15.2M15.2 4.3C15.2 6.12254
16.6775 7.6 18.5 7.6C20.3225 7.6 21.8 6.12254
21.8 4.3C21.8 2.47746 20.3225 1 18.5
1C16.6775 1 15.2 2.47746 15.2 4.3Z"
stroke="#C0C0C0" strokeWidth="1.5"
strokeLinecap="round" strokeLinejoin="round"/>
</svg>
Квесты
</NavLink>
<NavLink to="/profile" className="menu_item">
<svg width="24" height="24" viewBox="0 0 24 24"
fill="none" xmlns="http://www.w3.org/2000/svg">
<path d="M2 19.5C2 18.1739 2.52678 16.9021 3.46447
15.9645C4.40215 15.0268 5.67392 14.5 7 14.5H17C18.3261 14.5 19.5979
15.0268 20.5355 15.9645C21.4732 16.9021 22 18.1739 22 19.5C22 20.163
21.7366 20.7989 21.2678 21.2678C20.7989 21.7366 20.163 22 19.5
22H4.5C3.83696 22 3.20107 21.7366 2.73223 21.2678C2.26339 20.7989 2
20.163 2 19.5Z" stroke="#C0C0C0" strokeWidth="1.5"
strokeLinejoin="round"/>
<path d="M11.9999 10.1184C14.4488 10.1184 16.4341
8.13316 16.4341 5.68421C16.4341 3.23526 14.4488 1.25 11.9999 1.25C9.55094
1.25 7.56567 3.23526 7.56567 5.68421C7.56567 8.13316 9.55094 10.1184
11.9999 10.1184Z" stroke="#C0C0C0" strokeWidth="1.5"/>
</svg>
Профиль
</NavLink>
{token &&
jwtDecode(token) ["http://schemas.microsoft.com/ws/2008/06/identity/claims
/role"] == 'Admin' ? (
<NavLink to="/users" className="menu_item">
<svg width="26" height="24" viewBox="0 0 26 24"
fill="none" xmlns="http://www.w3.org/2000/svg">
<path d="M25 19.5C25 21 23.5 21 23.5 21H11.5C11.5 21 10
21 10 19.5C10 18 11.5 13.5 17.5 13.5C23.5 13.5 25 18 25 19.5Z"
stroke="#C0C0C0" strokeWidth="1.5"/>
<path d="M21.6575 9.22208C21.8836 8.67611 22 8.09095 22
7.5C22 6.30653 21.5259 5.16193 20.682 4.31802C19.8381 3.47411 18.6935 3
17.5 3C16.3065 3 15.1619 3.47411 14.318 4.31802C13.4741 5.16193 13
6.30653 13 7.5C13 8.09095 13.1164 8.67611 13.3425 9.22208C13.5687 9.76804
13.9002 10.2641 14.318 10.682C14.7359 11.0998 15.232 11.4313 15.7779
11.6575C16.3239 11.8836 16.9091 12 17.5 12C18.0909 12 18.6761 11.8836
19.2221 11.6575C19.768 11.4313 20.2641 11.0998 20.682 10.682C21.0998
10.2641 21.4313 9.76804 21.6575 9.22208Z" stroke="#C0C0C0"
strokeWidth="1.5"/>
<path d="M9.559 13.5495C10.1853 13.6082 10.8036 13.7324
11.404 13.92C10.924 14.2905 10.4995 14.703 10.135 15.144C9.0655 16.437
8.5 17.985 8.5 19.5C8.49084 20.0183 8.60174 20.5317 8.824 21H2.5C1.5 21 1

```

```

20.5 1 19.5C1 18 2.5 13.5 8.5 13.5C8.85361 13.4987 9.20706 13.5152 9.559
13.5495Z" stroke="#C0C0C0" strokeWidth="1.5"/>
    <path d="M4.56802 5.06802C3.72411 5.91193 3.25 7.05653
3.25 8.25C3.25 9.44347 3.72411 10.5881 4.56802 11.432C5.41193 12.2759
6.55653 12.75 7.75 12.75C8.94347 12.75 10.0881 12.2759 10.932
11.432C11.7759 10.5881 12.25 9.44347 12.25 8.25C12.25 7.05653 11.7759
5.91193 10.932 5.06802C10.0881 4.22411 8.94347 3.75 7.75 3.75C6.55653
3.75 5.41193 4.22411 4.56802 5.06802Z" stroke="#C0C0C0"
strokeWidth="1.5"/>
    </svg>
    Пользователи
</NavLink>
) : null}

<NavLink to="/targets" className="menu_item">
    <svg width="24" height="24" viewBox="0 0 24 24"
fill="none" xmlns="http://www.w3.org/2000/svg">
        <g clipPath="url(#clip0_843_17730)">
            <path d="M6.42944 23C4.61839 22.8217 3.40029 22.4182
2.49043 21.5096C1.58058 20.5997 1.17826 19.3804 1 17.5706M6.42944
1C4.61839 1.17826 3.40029 1.58181 2.49043 2.49043C1.58058 3.40029 1.17826
4.61963 1 6.42944M17.5706 1C19.3816 1.17826 20.5997 1.58181 21.5096
2.49043C22.4194 3.40029 22.8217 4.61963 23 6.42944M17.5706 23C19.3816
22.8217 20.5997 22.4182 21.5096 21.5096C22.4194 20.5997 22.8217 19.3804
23 17.5706M7.04839 9.5242L12 12M7.04839 9.5242V14.4758L12 16.9516M7.04839
9.5242L12 7.04839L16.9516 9.5242M12 12L16.9516 9.5242M12 12V16.9516M12
16.9516L16.9516 14.4758V9.5242" stroke="#C0C0C0" strokeWidth="1.5"
strokeLinecap="round" strokeLinejoin="round"/>
        </g>
        <defs>
            <clipPath id="clip0_843_17730">
                <rect width="24" height="24" fill="white"/>
            </clipPath>
        </defs>
    </svg>
    Таргеты
</NavLink>
<NavLink to="/logout" className="menu_item">
    <svg width="24" height="24" viewBox="0 0 24 24"
fill="none" xmlns="http://www.w3.org/2000/svg">
        <g clipPath="url(#clip0_843_17736)">
            <path d="M16.7657 23H1.97411C1.71576 23 1.46799 22.9034
1.28531 22.7315C1.10263 22.5596 1 22.3264 1 22.0833V1.91667C1 1.67355
1.10263 1.44039 1.28531 1.26849C1.46799 1.09658 1.71576 1 1.97411
1H16.7657" stroke="#C0C0C0" strokeWidth="1.5" strokeLinecap="round"/>
            <path d="M8.13354 11.375C7.78837 11.375 7.50854 11.6548
7.50854 12C7.50854 12.3452 7.78837 12.625 8.13354 12.625L8.13354
11.375ZM23.442 12.4419C23.686 12.1979 23.686 11.8021 23.442
11.5581L19.4645 7.58058C19.2204 7.33651 18.8247 7.33651 18.5806
7.58058C18.3365 7.82466 18.3365 8.22039 18.5806 8.46447L22.1161
12L18.5806 15.5355C18.3365 15.7796 18.3365 16.1753 18.5806
16.4194C18.8247 16.6635 19.2204 16.6635 19.4645 16.4194L23.442
12.4419ZM8.13354 12.625L23 12.625L23 11.375L8.13354 11.375L8.13354
12.625Z" fill="#C0C0C0"/>
        </g>
        <defs>
            <clipPath id="clip0_843_17736">

```

```

        <rect width="24" height="24" fill="white"/>
      </clipPath>
    </defs>
  </svg>
  Выйти
</NavLink>
</div>
)
}
export default Menu_admin

```

Листинг Б.13 – Quests.jsx (страница квестов)

```

import styles from '../Quests.module.css'
import Menu_admin from '../../ui/menu_admin/Menu_admin'
import Quest_card_small from
  '../ui/quest_card_small/Quest_card_small.jsx'
import {baseURL_requests_service, baseURL_users_service} from
  '../../common.js'
import { useState, useEffect } from 'react'
import axios from 'axios'
import { Swiper, SwiperSlide } from 'swiper/react'
import { Link } from 'react-router-dom'
import Delete_city from '../../ui/delete_city/Delete_city.jsx'
import 'swiper/css';
import 'swiper/css/pagination';
import 'swiper/css/scrollbar'
import { Pagination, Scrollbar } from 'swiper/modules';
import Add_city from '../../ui/add_city/Add_city.jsx'

const getRequests = baseURL_requests_service + '/api/Requests/all-brief-info'
const getCities = baseURL_users_service + '/api/Cities'

const Quests = () => {
  const [requests, setRequests] = useState([])
  const [cities, setCities] = useState([])
  const [cityDeleting, setCityDeleting] = useState({})
  const [showModalDeleting, setShowModalDeleting] = useState(false)
  const [showModalAdding, setShowModalAdding] = useState(false)

  useEffect(() => {
    axios.get(getRequests)
      .then(response => {
        setRequests(response.data)
      })
      .catch((error) => console.log(error.response));
    axios.get(getCities)
      .then(response => {
        setCities(response.data)
        console.log(response.data)
      })
      .catch((error) => console.log(error.response));
  }, [])

  const handleCallback = () => {
    axios.get(getRequests)
      .then(response => {

```

```

        setQuests(response.data)
    })
    .catch((error) => console.log(error.response));
    axios.get(getCities)
    .then(response => {
        setCities(response.data)
        console.log(response.data)
    })
    .catch((error) => console.log(error.response));
};

return (
    <div className={styles.container}>
        <div className={styles.menu}>
            <Menu_admin />
        </div>
        <Swiper
            slidesPerView={2}
            spaceBetween={100}
            pagination={{
                clickable: true,
            }}

            modules={[Pagination]}
            className={styles.mySwiperQuests}
            style={{
                "--swiper-pagination-color": "#7A1EFF",
                "--swiper-pagination-top": "450px",
                "--swiper-pagination-bullet-inactive-color": "#C0C0C0",
                "--swiper-pagination-bullet-inactive-opacity": "1",
                "--swiper-pagination-bullet-size": "12px",
                "--swiper-pagination-bullet-horizontal-gap": "5px",
            }}
        >
            {quests.length ? quests.map(quest => {
                return (
                    <SwiperSlide key={quest.id}
                        className={styles.slide}><Quest_card_small quest={quest}
                            handleCallback={handleCallback}/></SwiperSlide>
                )
            })
                : <p>Нет квестов</p>
            }
        </Swiper>
        <Link to="/add_quest"> <button
            className={styles.add_btn}>+</button> </Link>
        <div className={styles.citiesTitle}>
            <p>Города</p>
        </div>
        <Swiper
            slidesPerView={2}
            spaceBetween={30}
            scrollbar={{
                hide: false,
                draggable: true,
            }}
            dragSize={5}

```

```

modules={[Scrollbar]}
className={styles.mySwiperCities}
style={{
  '--swiper-scrollbar-border-radius': '10px',
  '--swiper-scrollbar-bottom': '0px',
  '--swiper-scrollbar-sides-offset': '50px',
  '--swiper-scrollbar-bg-color': '#C0C0C0',
  '--swiper-scrollbar-drag-bg-color': '#7A1EFF',
  '--swiper-scrollbar-size': '5px',
}}>
{cities.length ? cities.map(city => {
  return (
    <SwiperSlide key={city.id}
      className={styles.slideCity}
    >
      <div className={styles.city}>
        <p className={styles.cityLabel}>{city.name}</p>
        <div className={styles.itemDel}>
          <button
            className={styles.itemDelBtn}
            onClick={ (e) => {
              setShowModalDeleting(true)
              setCityDeleting(city)
            }}
          >
            X
          </button>
        </div>
      </div>
    </SwiperSlide>
  )
})
: <p>Нет городов</p>
}
<SwiperSlide>
  <button className={styles.cityBtn}
    onClick={ (e) => {
      setShowModalAdding(true) }}>+</button>
</SwiperSlide>
</Swiper>

{showModalAdding && <Add_city closeModalAdding={() =>
setShowModalAdding(false)} callback={handleCallback}/>}

{showModalDeleting && <Delete_city city={cityDeleting}
closeModalDeleting={() => setShowModalDeleting(false)}
callback={handleCallback}/>}
</div>
)
}
export default Quests

```

Листинг Б.14 – docker-compose.yml

```

version: '3.4'

```

```

services:
  questie_backend:
    image: ${DOCKER_REGISTRY-}questiebackend
    build:
      context: .
      dockerfile: questie_backend/Dockerfile
    environment:
      - ASPNETCORE_ENVIRONMENT=Development
      - ASPNETCORE_HTTP_PORTS=8080
      - ASPNETCORE_HTTPS_PORTS=8081
    ports:
      - "5002:8080"
      - "5003:8081"
    volumes:
      -
    ${APPDATA}/Microsoft/UserSecrets:/home/app/.microsoft/usersecrets:ro
      - ${APPDATA}/ASP.NET/Https:/home/app/.aspnet/https:ro
    networks:
      - questienet
    depends_on:
      - db
      - migrator

  questie_backend_users:
    image: ${DOCKER_REGISTRY-}questiebackendusers
    build:
      context: .
      dockerfile: questie_backend_users/Dockerfile
    environment:
      - ASPNETCORE_ENVIRONMENT=Development
      - ASPNETCORE_HTTP_PORTS=8080
      - ASPNETCORE_HTTPS_PORTS=8081
    ports:
      - "5004:8080"
      - "5005:8081"
    volumes:
      -
    ${APPDATA}/Microsoft/UserSecrets:/home/app/.microsoft/usersecrets:ro
      - ${APPDATA}/ASP.NET/Https:/home/app/.aspnet/https:ro
      - questie-upload-images:/app/Images
    networks:
      - questienet
    depends_on:
      - db
      - migrator

  questie_backend_authorization:
    image: ${DOCKER_REGISTRY-}questiebackendauthorization
    build:
      context: .
      dockerfile: questie_backend_authorization/Dockerfile
    environment:
      - ASPNETCORE_ENVIRONMENT=Development
      - ASPNETCORE_HTTP_PORTS=8080
      - ASPNETCORE_HTTPS_PORTS=8081
    ports:
      - "5006:8080"

```

```

    - "5007:8081"
  volumes:
    -
  ${APPDATA}/Microsoft/UserSecrets:/home/app/.microsoft/usersecrets:ro
    - ${APPDATA}/ASP.NET/Https:/home/app/.aspnet/https:ro
  networks:
    - questienet
  depends_on:
    - db
    - migrator

migrator:
  image: ${DOCKER_REGISTRY-}migrator
  build:
    context: .
    dockerfile: migrator/Dockerfile
  environment:
    - ASPNETCORE_ENVIRONMENT=Development
    - ASPNETCORE_HTTP_PORTS=8080
    - ASPNETCORE_HTTPS_PORTS=8081
  ports:
    - "5008:8080"
    - "5009:8081"
  volumes:
    -
  ${APPDATA}/Microsoft/UserSecrets:/home/app/.microsoft/usersecrets:ro
    - ${APPDATA}/ASP.NET/Https:/home/app/.aspnet/https:ro
  networks:
    - questienet
  depends_on:
    - db

db:
  container_name: postgresql-db
  image: postgres:15.3-alpine3.18
  environment:
    - POSTGRES_DB=questie_db
    - POSTGRES_USER=questie_user
    - POSTGRES_PASSWORD=admin
  ports:
    - "5000:5432"
  volumes:
    - postgres-data:/var/lib/postgresql/data
  networks:
    - questienet

pgadmin:
  container_name: pgadmin
  image: dpape/pgadmin4:8.0
  environment:
    PGADMIN_DEFAULT_EMAIL: questie@yandex.ru
    PGADMIN_DEFAULT_PASSWORD: admin
    PGADMIN_LISTEN_PORT: 80
    VIRTUAL_HOST: pgadmin.local.net
    VIRTUAL_PORT: 5001
  ports:
    - 5001:80

```

```

    volumes:
      - pgadmin:/var/lib/pgadmin
    networks:
      - questienet
    depends_on:
      - db

networks:
  questienet:
    name: questie_network
    driver: bridge

volumes:
  postgres-data:
  pgadmin:
  questie-upload-images:

```

Листинг Б.15 – Program.cs программы migrator

```

using Microsoft.EntityFrameworkCore;
using migrator.Data;

namespace migrator
{
    public class Program
    {
        public static void Main(string[] args)
        {
            var builder = WebApplication.CreateBuilder(args);

            // Add services to the container.
            var configBuilder = new ConfigurationBuilder();
            // установка пути к текущему каталогу

            configBuilder.SetBasePath(Directory.GetCurrentDirectory());
            // получаем конфигурацию из файла appsettings.json
            configBuilder.AddJsonFile("appsettings.json");
            // создаем конфигурацию
            var config = configBuilder.Build();
            // получаем строку подключения
            string connectionString =
            config.GetConnectionString("DefaultConnection");

            // Add services to the container.
            builder.Services.AddDbContext<QuestsContext>(options =>
            options.UseNpgsql(connectionString));

            var app = builder.Build();

            app.MigrateDatabase();
            Environment.Exit(0);
        }
    }
}

```

Листинг Б.16 – MigrateDatabase.cs программы migrator

```

using Microsoft.EntityFrameworkCore;

```

```

namespace migrator.Data
{
    public static class MigrationManager
    {
        public static IHost MigrateDatabase(this IHost host)
        {
            using (var scope = host.Services.CreateScope())
            {
                using (var appContext =
scope.ServiceProvider.GetRequiredService<QuestsContext>())
                {
                    try
                    {
                        appContext.Database.Migrate();
                    }
                    catch (Exception ex) { }
                }
            }

            return host;
        }
    }
}

```

Листинг Б.17 – Program.cs программы questie_backend

```

using Microsoft.AspNetCore.Authentication.JwtBearer;
using Microsoft.EntityFrameworkCore;
using Microsoft.Extensions.FileProviders;
using Microsoft.IdentityModel.Tokens;
using Microsoft.OpenApi.Models;
using questie_backend.Data;
using questie_backend.Middleware;
using questie_backend.Repositories.Implimentations;
using questie_backend.Repositories.Interfaces;
using questie_backend.Services.Implimentations;
using questie_backend.Services.Interfaces;
using System.Text;

internal class Program
{
    private static void Main(string[] args)
    {
        var builder = WebApplication.CreateBuilder(args);
        ConfigurationManager configuration = builder.Configuration;

        var configBuilder = new ConfigurationBuilder();
        // установка пути к текущему каталогу
        configBuilder.SetBasePath(Directory.GetCurrentDirectory());
        // получаем конфигурацию из файла appsettings.json
        configBuilder.AddJsonFile("appsettings.json");
        // создаем конфигурацию
        var config = configBuilder.Build();
        // получаем строку подключения
    }
}

```

```

        string connectionString =
config.GetConnectionString("DefaultConnection");

        // Add services to the container.
        builder.Services.AddDbContext<QuestsContext>(options =>
options.UseNpgsql(connectionString), ServiceLifetime.Transient,
ServiceLifetime.Transient);

        builder.Services.AddControllers();
        // Learn more about configuring Swagger/OpenAPI at
https://aka.ms/aspnetcore/swashbuckle
        builder.Services.AddEndpointsApiExplorer();
        builder.Services.AddSwaggerGen(opt =>
        {
            // To Enable authorization using Swagger (JWT)
            opt.AddSecurityDefinition("Bearer", new
OpenApiSecurityScheme()
            {
                Name = "Authorization",
                Type = SecuritySchemeType.ApiKey,
                Scheme = "Bearer",
                BearerFormat = "JWT",
                In = ParameterLocation.Header,
                Description = "Используется JWT авторизация.
\r\n\r\n Введите 'Bearer' [пробел] и свой токен в текстовом поле
ниже.\r\n\r\nНапример: \"Bearer 12345abcdef\"",
            });
            opt.AddSecurityRequirement(new
OpenApiSecurityRequirement
            {
                {
                    new OpenApiSecurityScheme
                    {
                        Reference = new OpenApiReference
                        {
                            Type =
ReferenceType.SecurityScheme,
                            Id = "Bearer"
                        },
                        Array.Empty<string>()
                    },
                },
            });
        });

        builder.Services.AddAuthorization();
        // Adding Authentication
        builder.Services.AddAuthentication(options =>
        {
            options.DefaultAuthenticateScheme =
JwtBearerDefaults.AuthenticationScheme;
            options.DefaultScheme =
JwtBearerDefaults.AuthenticationScheme;
        })
        // Adding Jwt Bearer
        .AddJwtBearer(options =>
        {

```

```

        options.SaveToken = true;
        options.RequireHttpsMetadata = false;
        options.TokenValidationParameters = new
TokenValidationParameters()
        {
            ValidateAudience = true,
            ValidAudience = configuration["Jwt:Audience"],
            ValidateIssuer = true,
            ValidIssuer = configuration["Jwt:Issuer"],
            ValidateIssuerSigningKey = true,

            IssuerSigningKey = new
SymmetricSecurityKey(Encoding.UTF8.GetBytes(configuration["Jwt:Key"]))
        };

    });

    builder.Services.AddDistributedMemoryCache();
    builder.Services.AddSession();

    builder.Services.AddHttpContextAccessor();

    builder.Services.AddScoped<IActiveQuestRepository,
ActiveQuestRepository>();
    builder.Services.AddScoped<IActiveQuestService,
ActiveQuestService>();
    builder.Services.AddScoped<ICityService, CityService>();
    builder.Services.AddScoped<ICompletedQuestRepository,
CompletedQuestRepository>();
    builder.Services.AddScoped<ICompletedQuestService,
CompletedQuestService>();
    builder.Services.AddScoped<IQuestRepository,
QuestRepository>();
    builder.Services.AddScoped<IQuestService, QuestService>();
    builder.Services.AddScoped<IQuestionsRepository,
QuestionsRepository>();
    builder.Services.AddScoped<IQuestionsService,
QuestionsService>();
    builder.Services.AddScoped<INotesRepository,
NotesRepository>();
    builder.Services.AddScoped<INotesService, NotesService>();
    builder.Services.AddScoped<IStatisticRepository,
StatisticRepository>();
    builder.Services.AddScoped<IStatisticService,
StatisticService>();
    builder.Services.AddScoped<ITargetsRepository,
TargetsRepository>();
    builder.Services.AddScoped<ITargetsService, TargetsService>();
    builder.Services.AddScoped<IUsersService, UsersService>();

    var app = builder.Build();
    var env = app.Environment;
    app.UseCors(builder => builder
        .AllowAnyHeader()
        .AllowAnyMethod()
        .AllowAnyOrigin()
    );

```

```

        // Configure the HTTP request pipeline.
        if (app.Environment.IsDevelopment())
        {
            app.UseSwagger();
            app.UseSwaggerUI();
        }

        List<string> staticFilePaths = new List<string>()
    {
        "Images/Notes",
        "Images/Quests",
        "Images/Questions",
        "Images/Targets"
    };

        foreach (var path in staticFilePaths)
        {
            app.UseStaticFiles(new StaticFileOptions
            {
                FileProvider = new
PhysicalFileProvider(Path.Combine(env.ContentRootPath, path)),
                RequestPath = $"/{path}"
            });
        }

        app.UseSession();

        app.UseAuthentication();
        app.UseAuthorization();

        app.UseMiddleware<JWTMiddleware>();

        app.UseAuthorization();

        app.MapControllers();

        app.Run();
    }
}

```

Листинг Б.18 – StatisticController.cs программы questie_backend

```

using entitys.DTO;
using entitys.Enums;
using Microsoft.AspNetCore.Mvc;
using questie_backend.Services.Interfaces;

namespace questie_backend.Controllers{
    [ApiController]
    [Produces("application/json")]
    [Route("api/[controller]")]
    public class StatisticController : ControllerBase{
        private IStatisticService _statisticService;
        public StatisticController(IStatisticService statisticService)
        {

```

```

        _statisticService = statisticService;
    }

    [HttpGet("avg-distance")]
    public async Task<IActionResult> GetAverageDistance() {
        double? distance = await
        _statisticService.GetAverageDistanceAsync();

        return distance != null ? Ok(new { distance }) :
        NotFound();
    }

    [HttpGet("avg-tasks")]
    public async Task<IActionResult>
    GetAverageQuantityTasksInQuests() {
        double? tasks = await
        _statisticService.GetAverageQuantityTasksAsync();

        return tasks != null ? Ok(new { tasks }) : NotFound();
    }

    [HttpGet("added-quests")]
    public async Task<List<StatisticPointDto>>
    GetQuantityAddedQuests(Period period) {
        return await
        _statisticService.GetQuantityAddedQuestsAsync(period);
    }

    [HttpGet("quests")]
    public async Task<List<StatisticPointDto>>
    GetQuantityQuests(Period period) {
        return await
        _statisticService.GetQuantityQuestsAsync(period);
    }

    [HttpGet("passed")]
    public async Task<List<StatisticPointDto>>
    GetQuantityPassedQuest(Period period, Guid questId) {
        return await
        _statisticService.GetQuantityPassedQuestsAsync(period, questId);
    }

    [HttpGet("passed-all")]
    public async Task<List<StatisticPointDto>>
    GetQuantityAllPassedQuest(Period period) {
        return await
        _statisticService.GetQuantityPassedQuestsAsync(period);
    }

    [HttpGet("cities")]
    public async Task<List<StatisticPointDto>>
    GetPopularityCities() {
        return await
        _statisticService.GetPopularityCitiesAsync();
    }
}

```

Листинг Б.19 – StatisticService.cs программы questie_backend

```

using entitys.DTO;
using entitys.Enums;
using entitys.Models;
using questie_backend.Repositories.Interfaces;
using questie_backend.Services.Interfaces;

namespace questie_backend.Services.Implimentations{
    public class StatisticService : IStatisticService{
        private readonly IStatisticRepository _statisticRepository;
        private readonly ICityService _cityService;

        public StatisticService(IStatisticRepository
statisticRepository, ICityService cityService) {
            _statisticRepository = statisticRepository;
            _cityService = cityService;
        }

        public async Task<double?> GetAverageDistanceAsync() {
            return await
            _statisticRepository.GetAverageDistanceAsync();
        }

        public async Task<double?> GetAverageQuantityTasksAsync() {
            return await
            _statisticRepository.GetAverageQuantityTasksAsync();
        }

        public async Task<List<StatisticPointDto>>
GetPopularityCitiesAsync(){
            int quantityPasses = await
            _statisticRepository.GetQuantityPassesAsync();
            List<StatisticPointDto> morePopularCities = await
            _statisticRepository.GetQuantityPassesByCitiesAsync();
            double morePopularPasses = 0;
            foreach (StatisticPointDto city in morePopularCities){
                if (city.Name is null){
                    morePopularCities.Remove(city);
                    continue;
                }
                morePopularPasses += city.Uv;

                Guid guid;
                Guid.TryParse(city.Name, out guid);
                City? cityObj = await
            _cityService.GetCityByIdAsync(guid);
                if (cityObj is null) {
                    morePopularCities.Remove(city);
                }

                city.Name = cityObj!.Name;
            }

            morePopularCities.Add(new StatisticPointDto() {
                Name = "Другие",

```

```

        Uv = quantityPasses - (int)morePopularPasses
    });
    return morePopularCities;
}

    public async Task<List<StatisticPointDto>>
GetQuantityAddedQuestsAsync(Period period){
    switch (period){
        case Period.Day:
            List<StatisticPointDto> pointsDay = await
            _statisticRepository.GetQuantityAddedQuestsInLastDayAsync();
            List<StatisticPointDto> resultDay = new
List<StatisticPointDto>();
            for (int i = 0; i < 24; ++i) {
                StatisticPointDto? point =
pointsDay.Where(u => u.Name == i.ToString()).FirstOrDefault();

                if (point is not null) {
                    resultDay.Add(point);
                }
                else{
                    resultDay.Add(new
StatisticPointDto()

                        {
                            Name = i.ToString(),
                            Uv = 0
                        });
                }
            }

            return resultDay;
        case Period.Week:
            List<StatisticPointDto> resultWeek = new
List<StatisticPointDto>();
            DateTime nowWeek = DateTime.UtcNow.Date;
            DateTime startWeek = nowWeek.AddHours(-144);
            for (DateTime time = startWeek; time <=
nowWeek; time = time.AddHours(24)) {
                resultWeek.Add(await
            _statisticRepository.GetQuantityAddedQuestsInLastMonthAsync(time));
            }

            return resultWeek;
        case Period.Month:
            List<StatisticPointDto> resultMonth = new
List<StatisticPointDto>();
            DateTime nowMonth = DateTime.UtcNow.Date;
            DateTime startMonth = nowMonth.AddHours(-720);
            for (DateTime time = startMonth; time <=
nowMonth; time = time.AddHours(24)) {
                resultMonth.Add(await
            _statisticRepository.GetQuantityAddedQuestsInLastMonthAsync(time));
            }
            return resultMonth;
        case Period.Year:
            throw new NotImplementedException();
    }
}

```

```

        return null;
    }

    public async Task<List<StatisticPointDto>>
    GetQuantityPassedQuestsAsync(Period period, Guid questId){
        switch (period){
            case Period.Day:
                List<StatisticPointDto> pointsDay = await
                _statisticRepository.GetQuantityPassedQuestsInLastDayAsync(questId);
                List<StatisticPointDto> resultDay = new
                List<StatisticPointDto>();
                for (int i = 0; i < 24; ++i) {
                    StatisticPointDto? point =
                pointsDay.Where(u => u.Name == i.ToString()).FirstOrDefault();

                    if (point is not null){
                        resultDay.Add(point);
                    }
                    else{
                        resultDay.Add(new
                StatisticPointDto()
                    {
                        Name = i.ToString(),
                        Uv = 0
                    });
                }
            }

            return resultDay;
            case Period.Week:
                List<StatisticPointDto> resultWeek = new
                List<StatisticPointDto>();
                DateTime nowWeek = DateTime.UtcNow.Date;
                DateTime startWeek = nowWeek.AddHours(-144);
                for (DateTime time = startWeek; time <=
                nowWeek; time = time.AddHours(24)) {
                    resultWeek.Add(await
                _statisticRepository.GetQuantityPassedQuestsInLastMonthAsync(time,
                questId));
                }
            return resultWeek;
            case Period.Month:
                List<StatisticPointDto> resultMonth = new
                List<StatisticPointDto>();
                DateTime nowMonth = DateTime.UtcNow.Date;
                DateTime startMonth = nowMonth.AddHours(-720);
                for (DateTime time = startMonth; time <=
                nowMonth; time = time.AddHours(24)) {
                    resultMonth.Add(await
                _statisticRepository.GetQuantityPassedQuestsInLastMonthAsync(time,
                questId));
                }
            return resultMonth;
            case Period.Year:
                throw new NotImplementedException();
        }
    }

```

```

        return null;
    }

    public async Task<List<StatisticPointDto>>
    GetQuantityPassedQuestsAsync(Period period) {
        switch (period)
        {
            case Period.Day:
                List<StatisticPointDto> pointsDay = await
                _statisticRepository.GetQuantityPassedQuestsInLastDayAsync();
                List<StatisticPointDto> resultDay = new
                List<StatisticPointDto>();
                for (int i = 0; i < 24; ++i)
                {
                    StatisticPointDto? point =
                    pointsDay.Where(u => u.Name == i.ToString()).FirstOrDefault();

                    if (point is not null) {
                        resultDay.Add(point);
                    }
                    else{
                        resultDay.Add(new
                        StatisticPointDto()
                        {
                            Name = i.ToString(),
                            Uv = 0
                        });
                    }
                }
                return resultDay;
            case Period.Week:
                List<StatisticPointDto> resultWeek = new
                List<StatisticPointDto>();
                DateTime nowWeek = DateTime.UtcNow.Date;
                DateTime startWeek = nowWeek.AddHours(-144);
                for (DateTime time = startWeek; time <=
                nowWeek; time = time.AddHours(24))
                {
                    resultWeek.Add(await
                    _statisticRepository.GetQuantityPassedQuestsInLastMonthAsync(time));
                }

                return resultWeek;
            case Period.Month:
                List<StatisticPointDto> resultMonth = new
                List<StatisticPointDto>();
                DateTime nowMonth = DateTime.UtcNow.Date;
                DateTime startMonth = nowMonth.AddHours(-720);
                for (DateTime time = startMonth; time <=
                nowMonth; time = time.AddHours(24))
                {
                    resultMonth.Add(await
                    _statisticRepository.GetQuantityPassedQuestsInLastMonthAsync(time));
                }
                return resultMonth;
            case Period.Year:
                throw new NotImplementedException();
        }
    }

```

```

        }

        return null;
    }

    public async Task<List<StatisticPointDto>>
    GetQuantityQuestsAsync(Period period)
    {
        switch (period)
        {
            case Period.Day:
                List<StatisticPointDto> resultDay = new
                List<StatisticPointDto>();
                DateTime now = DateTime.UtcNow;
                DateTime start = now.AddHours(-23);
                for (DateTime time = start; time <= now; time
                = time.AddHours(1))
                {
                    resultDay.Add(await
                    _statisticRepository.GetQuantityQuestsInHourAsync(time));
                }

                return resultDay;
            case Period.Week:
                List<StatisticPointDto> resultWeek = new
                List<StatisticPointDto>();
                DateTime nowWeek = DateTime.UtcNow.Date;
                DateTime startWeek = nowWeek.AddHours(-144);
                for (DateTime time = startWeek; time <=
                nowWeek; time = time.AddHours(24))
                {
                    resultWeek.Add(await
                    _statisticRepository.GetQuantityQuestsInDayAsync(time));
                }

                return resultWeek;
            case Period.Month:
                List<StatisticPointDto> resultMonth = new
                List<StatisticPointDto>();
                DateTime nowMonth = DateTime.UtcNow.Date;
                DateTime startMonth = nowMonth.AddHours(-720);
                for (DateTime time = startMonth; time <=
                nowMonth; time = time.AddHours(24))
                {
                    resultMonth.Add(await
                    _statisticRepository.GetQuantityQuestsInDayAsync(time));
                }

                return resultMonth;
            case Period.Year:
                throw new NotImplementedException();
        }

        return null;
    }
}

```

Листинг Б.20 – IStatisticService.cs программы questie_backend

```
using entitys.DTO;
using entitys.Enums;
using Microsoft.AspNetCore.Mvc;

namespace questie_backend.Services.Interfaces{
    public interface IStatisticService{
        Task<double?> GetAverageDistanceAsync();
        Task<double?> GetAverageQuantityTasksAsync();
        Task<List<StatisticPointDto>> GetPopularityCitiesAsync();
        Task<List<StatisticPointDto>>
GetQuantityAddedQuestsAsync(Period period);
        Task<List<StatisticPointDto>>
GetQuantityPassedQuestsAsync(Period period, Guid questId);
        Task<List<StatisticPointDto>>
GetQuantityPassedQuestsAsync(Period period);
        Task<List<StatisticPointDto>> GetQuantityQuestsAsync(Period
period);
    }
}
```

Листинг Б.21 – StatisticController.cs программы questie_backend_users

```
using entitys.DTO;
using entitys.Enums;
using Microsoft.AspNetCore.Mvc;
using questie_backend_users.Services.Interfaces;

namespace questie_backend_users.Controllers{
    [ApiController]
    [Produces("application/json")]
    [Route("api/[controller]")]
    public class StatisticController : ControllerBase{
        private IStatisticService _statisticService;
        public StatisticController(IStatisticService
statisticService){
            _statisticService = statisticService;
        }

        [HttpGet("quests/{userId}")]
        public async Task<IActionResult> GetQuestsByUserId(Guid
userId) {
            if (userId == Guid.Empty) {
                return BadRequest("Идентификатор пользователя не
может быть пустым");
            }

            int? quests = await
_statisticService.GetQuestsByUserIdAsync(userId);

            return quests != null ? Ok(new { quests }) : NotFound();
        }

        [HttpGet("rating/{userId}")]
        public async Task<IActionResult> GetRatingByUserId(Guid
userId) {
```

```

        if (userId == Guid.Empty) {
            return BadRequest("Идентификатор пользователя не
может быть пустым");
        }

        int? rating = await
_statisticsService.GetRatingByUserIdAsync(userId);

        return rating != null ? Ok(new { rating }) : NotFound();
    }

    [HttpGet("distance/{userId}")]
    public async Task<IActionResult> GetDistanceByUserId(Guid
userId) {
        if (userId == Guid.Empty) {
            return BadRequest("Идентификатор пользователя не
может быть пустым");
        }

        int? distance = await
_statisticsService.GetDistanceByUserIdAsync(userId);

        return distance != null ? Ok(new { distance }) :
NotFound();
    }

    /// Note: Получение количества пройденных квестов, дистанция и
рейтинг
    [HttpGet("all/{userId}")]
    public async Task<ActionResult<UserStatisticDto?>>
GetAllStatisticByUserId(Guid userId){
        if (userId == Guid.Empty){
            return BadRequest("Идентификатор пользователя не
может быть пустым");
        }

        UserStatisticDto? statistic = await
_statisticsService.GetAllStatisticByUserIdAsync(userId);
        return statistic != null ? statistic : NotFound();
    }

    [HttpGet("top")]
    public async Task<List<UserTopDto>> GetUsersTop(){
        return await _statisticsService.GetUsersTop();
    }

    [HttpGet("users")]
    public async Task<List<StatisticPointDto>>
QuantityUsers(Period period){
        return await
_statisticsService.GetQuantityUsersAsync(period);
    }

    [HttpGet("registrations")]
    public async Task<List<StatisticPointDto>>
QuantityRegistrations(Period period) {

```

```

        return await
        _statisticService.GetQuantityRegistrationsAsync(period);
    }

    [HttpGet("cities")]
    public async Task<List<StatisticPointDto>>
    QuantityUsersByCities() {
        return await
        _statisticService.GetQuantityUsersByCitiesAsync();
    }
}

```

Листинг Б.22 – StatisticService.cs программы questie_backend_users

```

using entitys.DTO;
using entitys.Enums;
using entitys.Models;
using questie_backend_users.Repositories.Interfaces;
using questie_backend_users.Services.Interfaces;
using System.Data;

namespace questie_backend_users.Services.Implementations {
    public class StatisticService : IStatisticService{
        private readonly IStatisticRepository _statisticRepository;

        public StatisticService(IStatisticRepository
        statisticRepository) {
            _statisticRepository = statisticRepository;
        }

        public async Task<UserStatisticDto?>
        GetAllStatisticByUserIdAsync(Guid userId) {
            int? distance = await
            _statisticRepository.GetDistanceByUserIdAsync(userId);
            int? rating = await
            _statisticRepository.GetRatingByUserIdAsync(userId);
            int? quests = await
            _statisticRepository.GetQuestsByUserIdAsync(userId);

            if (!distance.HasValue || !rating.HasValue ||
            !quests.HasValue) {
                return null;
            }

            UserStatisticDto userStatisticDto = new
            UserStatisticDto() {
                Id = userId,
                Distance = distance.Value,
                Rating = rating.Value,
                Quests = quests.Value,
            };

            return userStatisticDto;
        }
    }
}

```

```

        public async Task<int?> GetDistanceByUserIdAsync(Guid userId)
        {
            return await
            _statisticRepository.GetDistanceByUserIdAsync(userId);
        }

        public async Task<List<StatisticPointDto>>
        GetQuantityRegistrationsAsync(Period period) {
            switch (period)
            {
                case Period.Day:
                    List<StatisticPointDto> pointsDay = await
                    _statisticRepository.GetQuantityRegistrationsInLastDayAsync();
                    List<StatisticPointDto> resultDay = new
                    List<StatisticPointDto>();
                    for (int i = 0; i < 24; ++i) {
                        StatisticPointDto? point =
                        pointsDay.Where(u => u.Name == i.ToString()).FirstOrDefault();

                        if (point is not null) {
                            resultDay.Add(point);
                        }
                        else{
                            resultDay.Add(new
                            StatisticPointDto()

                                {
                                    Name = i.ToString(),
                                    Uv = 0
                                });
                        }
                    }

                    return resultDay;
                case Period.Week:
                    List<StatisticPointDto> resultWeek = new
                    List<StatisticPointDto>();
                    DateTime nowWeek = DateTime.UtcNow.Date;
                    DateTime startWeek = nowWeek.AddHours(-144);
                    for (DateTime time = startWeek; time <=
                    nowWeek; time = time.AddHours(24)) {
                        resultWeek.Add(await
                        _statisticRepository.GetQuantityRegistrationsInLastMonthAsync(time));
                    }

                    return resultWeek;
                case Period.Month:
                    List<StatisticPointDto> resultMonth = new
                    List<StatisticPointDto>();
                    DateTime nowMonth = DateTime.UtcNow.Date;
                    DateTime startMonth = nowMonth.AddHours(-720);
                    for (DateTime time = startMonth; time <=
                    nowMonth; time = time.AddHours(24)) {
                        resultMonth.Add(await
                        _statisticRepository.GetQuantityRegistrationsInLastMonthAsync(time));
                    }

                    return resultMonth;
            }
        }

```

```

        case Period.Year:
            throw new NotImplementedException();
        }
        return null;
    }

    public async Task<List<StatisticPointDto>>
    GetQuantityUsersAsync(Period period) {
        switch (period) {
            case Period.Day:
                List<StatisticPointDto> resultDay = new
                List<StatisticPointDto>();
                DateTime now = DateTime.UtcNow;
                DateTime start = now.AddHours(-23);
                for (DateTime time = start; time <= now; time
                = time.AddHours(1)) {
                    resultDay.Add(await
                    _statisticRepository.GetQuantityUsersInHourAsync(time));
                }

                return resultDay;
            case Period.Week:
                List<StatisticPointDto> resultWeek = new
                List<StatisticPointDto>();
                DateTime nowWeek = DateTime.UtcNow.Date;
                DateTime startWeek = nowWeek.AddHours(-144);
                for (DateTime time = startWeek; time <=
                nowWeek; time = time.AddHours(24)) {
                    resultWeek.Add(await
                    _statisticRepository.GetQuantityUsersInDayAsync(time));
                }

                return resultWeek;
            case Period.Month:
                List<StatisticPointDto> resultMonth = new
                List<StatisticPointDto>();
                DateTime nowMonth = DateTime.UtcNow.Date;
                DateTime startMonth = nowMonth.AddHours(-720);
                for (DateTime time = startMonth; time <=
                nowMonth; time = time.AddHours(24)) {
                    resultMonth.Add(await
                    _statisticRepository.GetQuantityUsersInDayAsync(time));
                }

                return resultMonth;
            case Period.Year:
                throw new NotImplementedException();
        }

        return null;
    }

    public async Task<List<StatisticPointDto>>
    GetQuantityUsersByCitiesAsync()
    {
        int quantityUsers = await
        _statisticRepository.GetQuantityUsersAsync();
    }

```

```

        List<StatisticPointDto> morePopularCities = await
        _statisticRepository.GetQuantityUsersByCitiesAsync();
        double morePopularUsers = 0;
        StatisticPointDto? nullPoint = null;

        foreach (StatisticPointDto city in morePopularCities){
            if (city.Name is null) {
                nullPoint = city;
                continue;
            }
            morePopularUsers += city.Uv;
        }

        if (nullPoint is not null) {
            morePopularCities.Remove(nullPoint);
        }

        morePopularCities.Add(new StatisticPointDto() {
            Name = "Другие",
            Uv = quantityUsers - (int)morePopularUsers
        });

        return morePopularCities;
    }

    public async Task<int?> GetQuestsByUserIdAsync(Guid userId) {
        return await
        _statisticRepository.GetQuestsByUserIdAsync(userId);
    }

    public async Task<int?> GetRatingByUserIdAsync(Guid userId) {
        return await
        _statisticRepository.GetRatingByUserIdAsync(userId);
    }

    public async Task<List<UserTopDto>> GetUsersTop() {
        List<User> users = await
        _statisticRepository.GetFirstHundredUsersByRatingAsync();
        List<UserTopDto> userTopDto = new List<UserTopDto>();

        foreach (var user in users) {
            userTopDto.Add(new UserTopDto() {
                Login = user.Login,
                Rating = user.Rating
            });
        }

        return userTopDto;
    }
}

```

Листинг Б.23 – IStatisticService.cs программы questie_backend_users

```

using entityys.DTO;
using entityys.Enums;
using Microsoft.AspNetCore.Mvc;

```

```

namespace questie_backend_users.Services.Interfaces
{
    public interface IStatisticService
    {
        Task<UserStatisticDto?> GetAllStatisticByUserIdAsync(Guid
userId);
        Task<int?> GetDistanceByUserIdAsync(Guid userId);
        Task<List<StatisticPointDto>>
GetQuantityRegistrationsAsync(Period period);
        Task<List<StatisticPointDto>> GetQuantityUsersAsync(Period
period);
        Task<List<StatisticPointDto>> GetQuantityUsersByCitiesAsync();
        Task<int?> GetQuestsByUserIdAsync(Guid userId);
        Task<int?> GetRatingByUserIdAsync(Guid userId);
        Task<List<UserTopDto>> GetUsersTop();
    }
}

```

Листинг Б.24 – TargetsController.cs программы questie_backend

```

using entitys.DTO;
using entitys.Models;
using Microsoft.AspNetCore.Mvc;
using questie_backend.Services.Interfaces;
using System;

namespace questie_backend.Controllers {
    [ApiController]
    [Produces("application/json")]
    [Route("api/[controller]")]
    public class TargetsController : ControllerBase{
        private ITargetsService _targetsService;
        public TargetsController(ITargetsService targetsService){
            _targetsService = targetsService;
        }

        // GET: api/<TargetsController>
        [HttpGet]
        public async Task<List<TargetOutDto>> GetAllTargets()
        {
            return await _targetsService.GetAllTargetsAsync();
        }

        // GET api/<TargetsController>/{id}
        [HttpGet("id/{id}")]
        public async Task<ActionResult<TargetOutDto>>
GetTargetById(Guid? id){
            if (id == Guid.Empty || id is null){
                return BadRequest("Идентификатор таргета не может
быть пустым");
            }

            TargetOutDto? target = await
_targetsService.GetTargetByIdAsync((Guid)id);
            return target != null ? target : NotFound();
        }
    }
}

```

```

        // GET api/<TargetsController>/{questId}
        [HttpGet("quest-id/{questId}")]
        public async Task<ActionResult<TargetOutDto>>
GetTargetByQuestId(Guid? questId){
            if (questId == Guid.Empty || questId is null){
                return BadRequest("Идентификатор квеста не может
        БЫТЬ ПУСТЫМ");
            }

            TargetOutDto? target = await
_targetetsService.GetTargetByQuestIdAsync((Guid)questId);
            return target != null ? target : NotFound();
        }

        // POST api/<TargetsController>
        [HttpPost]
        public async Task<IActionResult> CreateTarget([FromForm]
TargetDto targetDto) {
            if (targetDto.Title == null) {
                return BadRequest("Поле заголовка не может быть
        ПУСТЫМ");
            }
            if (targetDto.Description == null) {
                return BadRequest("Поле описания не может быть
        ПУСТЫМ");
            }
            if (targetDto.ImageFile == null) {
                return BadRequest("Изображение не может быть
        ПУСТЫМ");
            }

            Target target = await
_targetetsService.CreateTargetAsync(targetDto);

            return CreatedAtAction("GetTargetById", new { id =
target.Id }, target);
        }

        [HttpDelete("{id}")]
        public async Task<IActionResult> DeleteTarget(Guid? id){
            if (id == Guid.Empty || id is null){
                return BadRequest("Идентификатор таргета не может
        БЫТЬ ПУСТЫМ");
            }

            return await _targetetsService.DeleteTargetAsync((Guid)id)
? NoContent() : NotFound();
        }

        [HttpPut]
        public async Task<IActionResult> UpdateTarget(Guid? id,
[FromForm] TargetDto? target){
            if (id == Guid.Empty || id is null){
                return BadRequest("Идентификатор таргета не может
        БЫТЬ ПУСТЫМ");
            }

```

```

    }
    if (target == null){
        return BadRequest("Новый таргет не может быть
пустым");
    }

    return await _targetsService.UpdateTargetAsync((Guid)id,
target) ? NoContent() : NotFound();
}
}
}

```

Листинг Б.25 – TargetsService.cs программы questie_backend

```

using entitys.DTO;
using entitys.Models;
using questie_backend.Repositories.Implimentations;
using questie_backend.Repositories.Interfaces;
using questie_backend.Services.Interfaces;

namespace questie_backend.Services.Implimentations
{
    public class TargetsService : ITargetsService
    {
        private readonly ITargetsRepository _targetsRepository;
        private readonly IQuestionsRepository _questionsRepository;
        private readonly IWebHostEnvironment _webHostEnvironment;
        private readonly IHttpContextAccessor _httpContextAccessor;

        public TargetsService(ITargetsRepository targetsRepository,
                               IQuestionsRepository
questionsRepository,
                               IWebHostEnvironment
webHostEnvironment,
                               IHttpContextAccessor
httpContextAccessor)
        {
            _targetsRepository = targetsRepository;
            _questionsRepository = questionsRepository;
            _webHostEnvironment = webHostEnvironment;
            _httpContextAccessor = httpContextAccessor;
        }

        public async Task<List<TargetOutDto>> GetAllTargetsAsync()
        {
            List<Target> targets = await
_targetRepository.GetAllTargetsAsync();
            List<TargetOutDto> targetsOutDto = new
List<TargetOutDto>();

            foreach (var target in targets)
            {
                targetsOutDto.Add(new TargetOutDto()
                {
                    Id = target.Id,
                    Title = target.Title,

```

```

        Description = target.Description,
        ImageSrc =
String.Format("${_httpContextAccessor.HttpContext!.Request.Scheme}://" +
"${_httpContextAccessor.HttpContext!.Request.Host}" +
"${_httpContextAccessor.HttpContext!.Request.PathBase}" +
$"/Images/Targets/{target.ImageName}")
        });
    }

    return targetsOutDto;
}

public async Task<TargetOutDto?> GetTargetByIdAsync(Guid id)
{
    Target? target = await
_targetsRepository.GetTargetByIdAsync(id);
    TargetOutDto? targetOutDto = null;

    if (target != null)
    {
        targetOutDto = new TargetOutDto();
        targetOutDto.Id = target.Id;
        targetOutDto.Title = target.Title;
        targetOutDto.Description = target.Description;
        targetOutDto.ImageSrc =
String.Format("${_httpContextAccessor.HttpContext!.Request.Scheme}://" +
"${_httpContextAccessor.HttpContext!.Request.Host}" +
"${_httpContextAccessor.HttpContext!.Request.PathBase}" +
$"/Images/Targets/{target.ImageName}");
    }

    return targetOutDto;
}

public async Task<TargetOutDto?> GetTargetByQuestIdAsync(Guid
questionId)
{
    Guid? targetId = await
_questionsRepository.GetTargetIdByQuestionIdAsync(questionId);
    return targetId != null ? await
GetTargetByIdAsync((Guid)targetId) : null;
}

public async Task<Target> CreateTargetAsync(TargetDto
targetDto)
{
    var imageName = await SaveImage(targetDto.ImageFile);
    Target target = new Target(targetDto.Title,
                                targetDto.Description,
                                imageName);
    await _targetsRepository.InsertTargetAsync(target);
}

```

```

        return target;
    }

    public async Task<bool> DeleteTargetAsync(Guid id)
    {
        Target? target = await
        _targetsRepository.GetTargetByIdAsync(id);
        if (target == null)
        {
            return false;
        }

        DeleteImage(target.ImageName);
        await _targetsRepository.DeleteAsync(target);

        return true;
    }

    public async Task<bool> UpdateTargetAsync(Guid id, TargetDto
target) {
        Target? oldTarget = await
        _targetsRepository.GetTargetByIdAsync(id);
        if (oldTarget == null){
            return false;
        }

        oldTarget.Title = target.Title != null ? target.Title :
oldTarget.Title;
        oldTarget.Description = target.Description != null ?
target.Description : oldTarget.Description;

        if (target.ImageFile != null){
            DeleteImage(oldTarget.ImageName);
            oldTarget.ImageName = await
SaveImage(target.ImageFile);
        }

        await _targetsRepository.UpdateAsync(oldTarget);

        return true;
    }

    private async Task<string> SaveImage(IFormFile imageFile)
    {
        string imageName = new
String(Path.GetFileNameWithoutExtension(imageFile.FileName).Take(10).ToAr
ray()).Replace(' ', '-');
        imageName += DateTime.Now.ToString("yymmssfff") +
Path.GetExtension(imageFile.FileName);
        var imagePath =
Path.Combine(_webHostEnvironment.ContentRootPath, "Images/Targets",
imageName);

        using (var fileStream = new FileStream(imagePath,
FileMode.Create)) {
            await imageFile.CopyToAsync(fileStream);

```

```

        }

        return imageName;
    }

    private void DeleteImage(string imageName){
        var imagePath =
Path.Combine(_webHostEnvironment.ContentRootPath, "Images/Targets",
imageName);

        if (File.Exists(imagePath))
        {
            File.Delete(imagePath);
        }
    }
}

```

Листинг Б.26 – NotesController.cs программы questie_backend

```

using entitys.DTO;
using entitys.Models;
using Microsoft.AspNetCore.Mvc;
using questie_backend.Services.Interfaces;

namespace questie_backend.Controllers{
    [ApiController]
    [Produces("application/json")]
    [Route("api/[controller]")]
    public class NotesController : ControllerBase{
        private INotesService _notesService;
        public NotesController(INotesService notesService){
            _notesService = notesService;
        }

        // GET: api/<NotesController>
        [HttpGet]
        public async Task<List<NoteOutDto>> GetAllNotes(){
            var t = Request.Host;
            return await _notesService.GetAllNotesAsync();
        }

        // GET api/<NotesController>/{id}
        [HttpGet("id/{id}")]
        public async Task<ActionResult<NoteOutDto>> GetNoteById(Guid
id) {
            NoteOutDto? note = await
            _notesService.GetNoteByIdAsync(id);
            return note != null ? note : NotFound();
        }

        // POST api/<NotesController>
        [HttpPost]
        public async Task<IActionResult> CreateNote([FromForm] NoteDto
noteDto) {
            if (noteDto.Title == null){

```

```

        return BadRequest("Поле заголовка не может быть
пустым");
    }
    if (noteDto.Description == null){
        return BadRequest("Поле описания не может быть
пустым");
    }
    if (noteDto.ImageFile == null){
        return BadRequest("Изображение не может быть
пустым");
    }
    Note note = await
_notesService.CreateNoteAsync(noteDto);

    return CreatedAtAction("GetNoteById", new { id = note.Id
}, note);
}

[HttpDelete("{id}")]
public async Task<IActionResult> DeleteNote(Guid id) {
    return await _notesService.DeleteNoteAsync(id) ?
NoContent() : NotFound();
}

[HttpPut]
public async Task<IActionResult> UpdateNote(Guid id,
[FromForm] NoteDto user) {

    return await _notesService.UpdateNoteAsync(id, user) ?
NoContent() : NotFound();
}
}
}

```

Листинг Б.27 – NotesService.cs программы questie_backend

```

using entitys.DTO;
using entitys.Models;
using questie_backend.Repositories.Interfaces;
using questie_backend.Services.Interfaces;

namespace questie_backend.Services.Implimentations{
    public class NotesService : INotesService{
        private readonly INotesRepository _notesRepository;
        private readonly IWebHostEnvironment _webHostEnvironment;
        private readonly IHttpContextAccessor _httpContextAccessor;

        public NotesService(INotesRepository notesRepository,
IWebHostEnvironment webHostEnvironment, IHttpContextAccessor
httpContextAccessor){
            _notesRepository = notesRepository;
            _webHostEnvironment = webHostEnvironment;
            _httpContextAccessor = httpContextAccessor;
        }

        public async Task<List<NoteOutDto>> GetAllNotesAsync() {

```

```

        List<Note> notes = await
_notesRepository.GetAllNotesAsync();
        List<NoteOutDto> notesOutDto = new List<NoteOutDto>();

        foreach (var note in notes){
            notesOutDto.Add(new NoteOutDto(){
                Id = note.Id,
                Title = note.Title,
                Description = note.Description,
                ImageSrc =
String.Format("${_httpContextAccessor.HttpContext!.Request.Scheme}://" +
"${_httpContextAccessor.HttpContext!.Request.Host}" +
"${_httpContextAccessor.HttpContext!.Request.PathBase}" +
$"/Images/Notes/{note.ImageName}")
            });
        }
        return notesOutDto;
    }

    public async Task<NoteOutDto?> GetNoteByIdAsync(Guid id){
        Note? note = await
_notesRepository.GetNoteByIdAsync(id);
        NoteOutDto? noteOutDto = null;

        if (note != null){
            noteOutDto = new NoteOutDto();
            noteOutDto.Id = note.Id;
            noteOutDto.Title = note.Title;
            noteOutDto.Description = note.Description;
            noteOutDto.ImageSrc =
String.Format("${_httpContextAccessor.HttpContext!.Request.Scheme}://" +
"${_httpContextAccessor.HttpContext!.Request.Host}" +
"${_httpContextAccessor.HttpContext!.Request.PathBase}" +
$"/Images/Notes/{note.ImageName}");
        }

        return noteOutDto;
    }

    public async Task<Note> CreateNoteAsync(NoteDto noteDto){
        var imageName = await SaveImage(noteDto.ImageFile);
        Note note = new Note(noteDto.Title, noteDto.Description,
                               imageName);
        await _notesRepository.InsertNoteAsync(note);
        return note;
    }

    public async Task<bool> DeleteNoteAsync(Guid id){
        Note? note = await
_notesRepository.GetNoteByIdAsync(id);
        if (note == null){
            return false;
        }

        DeleteImage(note.ImageName);
        await _notesRepository.DeleteAsync(note);
    }

```

```

        return true;
    }

    public async Task<bool> UpdateNoteAsync(Guid id, NoteDto note)
    {
        Note? oldNote = await
        _notesRepository.GetNoteByIdAsync(id);
        if (oldNote is null){
            return false;
        }

        oldNote.Title = note.Title is not null ? note.Title :
        oldNote.Title;
        oldNote.Description = oldNote.Description;

        if (note.ImageFile is not null){
            DeleteImage(oldNote.ImageName);
            oldNote.ImageName = await
        SaveImage(note.ImageFile);
        }
        await _notesRepository.UpdateAsync(oldNote);
        return true;
    }

    private async Task<string> SaveImage(IFormFile imageFile)
    {
        string imageName = new
        String(Path.GetFileNameWithoutExtension(imageFile.FileName).Take(10).ToAr
        ray()).Replace(' ', '-');
        imageName += DateTime.Now.ToString("yymmssfff") +
        Path.GetExtension(imageFile.FileName);
        var imagePath =
        Path.Combine(_webHostEnvironment.ContentRootPath, "Images/Notes",
        imageName);
        using (var fileStream = new FileStream(imagePath,
        FileMode.Create)){
            await imageFile.CopyToAsync(fileStream);
        }

        return imageName;
    }

    private void DeleteImage(string imageName){
        var imagePath =
        Path.Combine(_webHostEnvironment.ContentRootPath, "Images/Notes",
        imageName);

        if (File.Exists(imagePath)) {
            File.Delete(imagePath);
        }
    }
}

```

Приложение В
(рекомендуемое)

Свидетельство о государственной регистрации программы для ЭВМ

РОССИЙСКАЯ ФЕДЕРАЦИЯ



СВИДЕТЕЛЬСТВО

о государственной регистрации программы для ЭВМ

№ 2024660878

**Web-клиент административной панели системы для
исторических квест-экскурсий с применением
технологии дополненной реальности**

Правообладатель: *Агапова Валерия Александровна (RU)*

Авторы: *Агапова Валерия Александровна (RU), Кудашов
Александр Сергеевич (RU), Дьячков Даниил
Александрович (RU), Левин Андрей Алексеевич (RU)*



Заявка № 2024660119

Дата поступления 04 мая 2024 г.

Дата государственной регистрации

в Реестре программ для ЭВМ 14 мая 2024 г.

*Руководитель Федеральной службы
по интеллектуальной собственности*

ДОКУМЕНТ ПОДПИСАН ЭЛЕКТРОННОЙ ПОДПИСЬЮ
Сертификат: 429b6a0fe3853164baf96f83b73b4aa7
Владелец: **Зубов Юрий Сергеевич**
Действителен с 10.05.2023 по 02.08.2024

Ю.С. Зубов

Приложение Г
(рекомендуемое)
Тест-кейсы

Таблица Г.1 – Тест-кейсы тестирования прецедента «Добавить историческую справку»

№	Состав теста	Ожидаемый результат	Наблюдаемый результат	Вывод
1	Название: T@ Описание: D@& Файл: image.pdf	Некорректная длина названия, некорректная длина описания, некорректные символы в составе названия, некорректные символы в составе описания, недопустимый формат файла	Соответствует ожидаемым результатам (рисунок Г.1)	Тест пройден
2	Название: T@ Описание: D@& Файл: image.png	Некорректная длина названия, некорректная длина описания, некорректные символы в составе названия, некорректные символы в составе описания, корректный формат файла	Соответствует ожидаемым результатам (рисунок Г.2)	Тест пройден
3	Название: T@ Описание: Визит в Пензу – это возможность познакомиться с историей и архитектурой старинного русского города, съездить на родину Михаила Юрьевича Лермонтова, отдохнуть в ахунских сосновых борах Файл: image.pdf	Некорректная длина названия, некорректная длина описания, некорректные символы в составе названия, корректные символы в составе описания, некорректный формат файла	Соответствует ожидаемым результатам (рисунок Г.3)	Тест пройден
4	Название: T@ Описание: Визит в Пензу – это возможность познакомиться с историей и архитектурой старинного русского города, съездить на родину Михаила Юрьевича Лермонтова, отдохнуть в ахунских сосновых борах Файл: image.png	Некорректная длина названия, некорректная длина описания, некорректные символы в составе названия, корректные символы в составе описания, корректный формат файла	Соответствует ожидаемым результатам (рисунок Г.4)	Тест пройден
5	Название: Ли Описание: &Визит в Пензу @– это возможность познакомиться с историей и архитектурой старинного русского города, съездить на родину Михаила Юрьевича Лермонтова, отдохнуть в ахунских сосновых борах Файл: image.pdf	Некорректная длина названия, некорректная длина описания, корректные символы в составе названия, некорректные символы в составе описания, некорректный формат файла	Соответствует ожидаемым результатам (рисунок Г.5)	Тест пройден
6	Название: Ли Описание: &Визит в Пензу @– это возможность познакомиться с историей и архитектурой старинного русского города, съездить на родину Михаила Юрьевича Лермонтова, отдохнуть в ахунских сосновых борах Файл: image.png	Некорректная длина названия, некорректная длина описания, корректные символы в составе названия, некорректные символы в составе описания, корректный формат файла	Соответствует ожидаемым результатам (рисунок Г.6)	Тест пройден
7	Название: Ли Описание: Визит в Пензу – это возможность познакомиться с историей и архитектурой старинного русского города, съездить на родину Михаила Юрьевича Лермонтова, отдохнуть в ахунских сосновых борах Файл: image.pdf	Некорректная длина названия, некорректная длина описания, корректные символы в составе названия, корректные символы в составе описания, некорректный формат файла	Соответствует ожидаемым результатам (рисунок Г.7)	Тест пройден

Продолжение таблицы Г.1

№	Состав теста	Ожидаемый результат	Наблюдаемый результат	Вывод
8	Название: Ли Описание: Визит в Пензу – это возможность познакомиться с историей и архитектурой старинного русского города, съездить на родину Михаила Юрьевича Лермонтова, отдохнуть в ахунских сосновых борах Файл: image.png	Некорректная длина названия, некорректная длина описания, корректные символы в составе названия, корректные символы в составе описания, корректный формат файла	Соответствует ожидаемым результатам (рисунок Г.8)	Тест пройден
9	Название: T@ Описание: Памятник *ювелиру* Файл: image.pdf	Некорректная длина названия, корректная длина описания, некорректные символы в составе названия, некорректные символы в составе описания, некорректный формат файла	Соответствует ожидаемым результатам (рисунок Г.9)	Тест пройден
10	Название: T@ Описание: Памятник *ювелиру* Файл: image.png	Некорректная длина названия, корректная длина описания, некорректные символы в составе названия, некорректные символы в составе описания, корректный формат файла	Соответствует ожидаемым результатам (рисунок Г.10)	Тест пройден
11	Название: T@ Описание: Памятник ювелиру Файл: image.pdf	Некорректная длина названия, корректная длина описания, некорректные символы в составе названия, корректные символы в составе описания, некорректный формат файла	Соответствует ожидаемым результатам (рисунок Г.11)	Тест пройден
12	Название: T@ Описание: Памятник ювелиру Файл: image.png	Некорректная длина названия, корректная длина описания, некорректные символы в составе названия, корректные символы в составе описания, корректный формат файла	Соответствует ожидаемым результатам (рисунок Г.12)	Тест пройден
13	Название: Памятник ювелиру в Пензе Описание: Памятник *ювелиру* Файл: image.docx	Некорректная длина названия, корректная длина описания, корректные символы в составе названия, некорректные символы в составе описания, некорректный формат файла	Соответствует ожидаемым результатам (рисунок Г.13)	Тест пройден
14	Название: Памятник ювелиру в Пензе Описание: Памятник *ювелиру* Файл: image.jpg	Некорректная длина названия, корректная длина описания, корректные символы в составе названия, некорректные символы в составе описания, корректный формат файла	Соответствует ожидаемым результатам (рисунок Г.14)	Тест пройден
15	Название: Памятник ювелиру в Пензе Описание: Памятник ювелиру Файл: image.pptx	Некорректная длина названия, корректная длина описания, корректные символы в составе названия, корректные символы в составе описания, некорректный формат файла	Соответствует ожидаемым результатам (рисунок Г.15)	Тест пройден
16	Название: Памятник ювелиру в Пензе Описание: Памятник ювелиру Файл: image.png	Некорректная длина названия, корректная длина описания, корректные символы в составе названия и описания, корректный формат файла	Соответствует ожидаемым результатам (рисунок Г.16)	Тест пройден

Продолжение таблицы Г.1

№	Состав теста	Ожидаемый результат	Наблюдаемый результат	Вывод
17	Название: Памятник *ювелиру* Описание: &Визит в Пензу @– это возможность познакомиться с историей и архитектурой старинного русского города, съездить на родину Михаила Юрьевича Лермонтова. Файл: image.pdf	Корректная длина названия, некорректная длина описания, некорректные символы в составе названия, некорректные символы в составе описания, некорректный формат файла	Соответствует ожидаемым результатам (рисунок Г.17)	Тест пройден
18	Название: Памятник *ювелиру* Описание: &Визит в Пензу @– это возможность познакомиться с историей и архитектурой старинного русского города, съездить на родину Михаила Юрьевича Лермонтова, отдохнуть в ахунских сосновых борах	Корректная длина названия, некорректная длина описания, некорректные символы в составе названия, некорректные символы в составе описания, файл отсутствует	Соответствует ожидаемым результатам (рисунок Г.18)	Тест пройден
19	Название: Памятник *ювелиру* Описание: Визит в Пензу – это возможность познакомиться с историей и архитектурой старинного русского города, съездить на родину Михаила Юрьевича Лермонтова, отдохнуть в ахунских сосновых борах Файл: image.pdf	Корректная длина названия, некорректная длина описания, некорректные символы в составе названия, корректные символы в составе описания, некорректный формат файла	Соответствует ожидаемым результатам (рисунок Г.19)	Тест пройден
20	Название: Памятник *ювелиру* Описание: 123	Корректная длина названия, некорректная длина описания, некорректные символы в составе названия, корректные символы в составе описания, файл отсутствует	Соответствует ожидаемым результатам (рисунок Г.20)	Тест пройден
21	Название: Памятник ювелиру Описание: 12@ Файл: image.pdf	Корректная длина названия, некорректная длина описания, корректные символы в составе названия, некорректные символы в составе описания, некорректный формат файла	Соответствует ожидаемым результатам (рисунок Г.21)	Тест пройден
22	Название: Памятник ювелиру Описание: &9a	Корректная длина названия, некорректная длина описания, корректные символы в составе названия, некорректные символы в составе описания, файл отсутствует	Соответствует ожидаемым результатам (рисунок Г.22)	Тест пройден
23	Название: Памятник ювелиру Описание: 123 Файл: image.docx	Корректная длина названия, некорректная длина описания, корректные символы в составе названия, корректные символы в составе описания, некорректный формат файла	Соответствует ожидаемым результатам (рисунок Г.23)	Тест пройден
24	Название: Памятник ювелиру Описание: Визит в Пензу – это возможность познакомиться с историей и архитектурой старинного русского города, съездить на родину Михаила Юрьевича Лермонтова, отдохнуть в ахунских сосновых борах	Корректная длина названия, некорректная длина описания, корректные символы в составе названия, корректные символы в составе описания, файл отсутствует	Соответствует ожидаемым результатам (рисунок Г.24)	Тест пройден
25	Название: Памятник&^ Описание: <Памятник ювелиру> Файл: image.exe	Корректная длина названия и описания, некорректные символы в составе и описания, некорректный формат файла	Соответствует ожидаемым результатам (рисунок Г.25)	Тест пройден

Окончание таблицы Г.1

№	Состав теста	Ожидаемый результат	Наблюдаемый результат	Вывод
26	Название: Памятник&^ Описание: <Памятник ювелиру>	Корректная длина названия, корректная длина описания, некорректные символы в составе названия, некорректные символы в составе описания, файл отсутствует	Соответствует ожидаемым результатам (рисунок Г.26)	Тест пройден
27	Название: @Памятник# Описание: Памятник ювелиру Файл: image.exe	Корректная длина названия, корректная длина описания, некорректные символы в составе названия, корректные символы в составе описания, некорректный формат файла	Соответствует ожидаемым результатам (рисунок Г.27)	Тест пройден
28	Название: Памятник№ Описание: Памятник ювелиру Файл: image.png	Корректная длина названия, корректная длина описания, некорректные символы в составе названия, корректные символы в составе описания, корректный формат файла	Соответствует ожидаемым результатам (рисунок Г.28)	Тест пройден
29	Название: Памятник Описание: %Памятник ювелиру% Файл: image.docx	Корректная длина названия, корректная длина описания, корректные символы в составе названия, некорректные символы в составе описания, некорректный формат файла	Соответствует ожидаемым результатам (рисунок Г.29)	Тест пройден
30	Название: Памятник Описание: %Памятник ювелиру%	Корректная длина названия, корректная длина описания, корректные символы в составе названия, некорректные символы в составе описания, файл отсутствует	Соответствует ожидаемым результатам (рисунок Г.30)	Тест пройден
31	Название: Памятник Описание: Памятник ювелиру Файл: image.docx	Корректная длина названия, корректная длина описания, корректные символы в составе названия, корректные символы в составе описания, некорректный формат файла	Соответствует ожидаемым результатам (рисунок Г.31)	Тест пройден
32	Название: Памятник Описание: Памятник ювелиру Файл: image.jpeg	Корректная длина названия, корректная длина описания, корректные символы в составе названия, корректные символы в составе описания, корректный формат файла	Соответствует ожидаемым результатам (рисунок Г.32)	Тест пройден

Количество баллов за задание: 10

Подсказка: Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить историческую справку X

Некорректная длина названия
Некорректные символы в составе названия

Название: Tø

Некорректная длина описания
Некорректные символы в составе описания

Описание: Dф&

Выберите файл или перетащите его сюда (только для изображений)

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.1 – Результат прохождения тест-кейса 1 для добавления исторической справки

Количество баллов за задание: 10

Подсказка: Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить историческую справку X

Некорректная длина названия
Некорректные символы в составе названия

Название: Tø

Некорректная длина описания
Некорректные символы в составе описания

Описание: Dф&

Выберите файл или перетащите его сюда (только для изображений)

image.png X

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.2 – Результат прохождения тест-кейса 2 для добавления исторической справки

Количество баллов за задание: 10

Подсказка: Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить историческую справку X

Некорректная длина названия
Некорректные символы в составе названия

название: те

Некорректная длина описания

описание: Визит в Пензу – это возможность познакомиться с историей и архитектурой старинного русского города, съездить на родину Михаила Юрьевича Лермонтова, отдохнуть в ахунских сосновых борах

Выберите файл или петещите его сюда (только для изображений)

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.3 – Результат прохождения тест-кейса 3 для добавления исторической справки

Количество баллов за задание: 10

Подсказка: Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить историческую справку X

Некорректная длина названия
Некорректные символы в составе названия

название: те

Некорректная длина описания

описание: Визит в Пензу – это возможность познакомиться с историей и архитектурой старинного русского города, съездить на родину Михаила Юрьевича Лермонтова, отдохнуть в ахунских сосновых борах

Выберите файл или петещите его сюда (только для изображений)

image.png X

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.4 – Результат прохождения тест-кейса 4 для добавления исторической справки

Количество баллов за задание

10

Подсказка

Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить историческую справку

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.5 – Результат прохождения тест-кейса 5 для добавления исторической справки

Количество баллов за задание

10

Подсказка

Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить историческую справку

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.6 – Результат прохождения тест-кейса 6 для добавления исторической справки

Количество баллов за задание: 10

Подсказка: Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить историческую справку

Некорректная длина названия

Название: Ли

Некорректная длина описания

Описание: Визит в Пензу – это возможность познакомиться с историей и архитектурой старинного русского города, съездить на родину Михаила Юрьевича Лермонтова, отдохнуть в акунских сосновых борках

Выберите файл или петещите его сюда (только для изображений)

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.7 – Результат прохождения тест-кейса 7 для добавления исторической справки

Количество баллов за задание: 10

Подсказка: Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить историческую справку

Некорректная длина названия

Название: Ли

Некорректная длина описания

Описание: Визит в Пензу – это возможность познакомиться с историей и архитектурой старинного русского города, съездить на родину Михаила Юрьевича Лермонтова, отдохнуть в акунских сосновых борках

Выберите файл или петещите его сюда (только для изображений)

image.png

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.8 – Результат прохождения тест-кейса 8 для добавления исторической справки

Количество баллов за задание

10

Подсказка

Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить историческую справку X

Некорректная длина названия
Некорректные символы в составе названия

Название
T@

Некорректные символы в составе описания

Описание
Памятник *ювелиру*

Выберите файл или перетащите его сюда
(только для изображений)

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.9 – Результат прохождения тест-кейса 9 для добавления исторической справки

Количество баллов за задание

10

Подсказка

Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить историческую справку X

Некорректная длина названия
Некорректные символы в составе названия

Название
T@

Некорректные символы в составе описания

Описание
Памятник *ювелиру*

Выберите файл или перетащите его сюда
(только для изображений)

image.png

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.10 – Результат прохождения тест-кейса 10 для добавления исторической справки

Количество баллов за задание

10

Подсказка

Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить историческую справку x

Некорректная длина названия
Некорректные символы в составе названия

Название
т@

Описание
Памятник ювелиру

Выберите файл или перетащите его сюда
(только для изображений)

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.11 – Результат прохождения тест-кейса 11 для добавления исторической справки

Количество баллов за задание

10

Подсказка

Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить историческую справку x

Некорректная длина названия
Некорректные символы в составе названия

Название
т@

Описание
Памятник ювелиру

Выберите файл или перетащите его сюда
(только для изображений)

image.png

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.12 – Результат прохождения тест-кейса 12 для добавления исторической справки

Количество баллов за задание: 10

Подсказка: Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить историческую справку

Некорректная длина названия
Памятник ювелиру в Пензе

Некорректные символы в составе описания
Памятник *ювелиру*

Выберите файл или перетяните его сюда (только для изображений)

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.13 – Результат прохождения тест-кейса 13 для добавления исторической справки

Количество баллов за задание: 10

Подсказка: Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить историческую справку

Некорректная длина названия
Памятник ювелиру в Пензе

Некорректные символы в составе описания
Памятник *ювелиру*

Выберите файл или перетяните его сюда (только для изображений)

image.jpg

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.14 – Результат прохождения тест-кейса 14 для добавления исторической справки

Количество баллов за задание

10

Подсказка

Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.15 – Результат прохождения тест-кейса 15 для добавления исторической справки

Количество баллов за задание

10

Подсказка

Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить

image.png

Завершить добавление

Добавить вопрос

Рисунок Г.16 – Результат прохождения тест-кейса 16 для добавления исторической справки

Количество баллов за задание

10

Подсказка

Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить историческую справку X

Некорректные символы в составе названия

Памятник *ювелиру*

Некорректная длина описания

Некорректные символы в составе описания

Описание

&Визит в Пензу @- это возможность познакомиться с историей и архитектурой старинного русского города, съездить на родину Михаила Юрьевича Лермонтова, отдохнуть в ахунских сосновых борах

Выберите файл или петещите его сюда (только для изображений)

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.17 – Результат прохождения тест-кейса 17 для добавления исторической справки

Количество баллов за задание

10

Подсказка

Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить историческую справку X

Некорректные символы в составе названия

Памятник *ювелиру*

Некорректная длина описания

Некорректные символы в составе описания

Описание

&Визит в Пензу @- это возможность познакомиться с историей и архитектурой старинного русского города, съездить на родину Михаила Юрьевича Лермонтова, отдохнуть в ахунских сосновых борах

Выберите файл или петещите его сюда (только для изображений)

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.18 – Результат прохождения тест-кейса 18 для добавления исторической справки

Количество баллов за задание: 10

Подсказка: Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить историческую справку

Некорректные символы в составе названия

Название: Памятник *ювелиру*

Некорректная длина описания

Описание: Визит в Пензу – это возможность познакомиться с историей и архитектурой старинного русского города, съездить на родину Михаила Юрьевича Лермонтова, отдохнуть в ахунских сосновых борках

Выберите файл или петещите его сюда (только для изображений)

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.19 – Результат прохождения тест-кейса 19 для добавления исторической справки

Количество баллов за задание: 10

Подсказка: Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить историческую справку

Некорректные символы в составе названия

Название: Памятник *ювелиру*

Некорректная длина описания

Описание: 123

Выберите файл или петещите его сюда (только для изображений)

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.20 – Результат прохождения тест-кейса 20 для добавления исторической справки

Количество баллов за задание

10

Подсказка

Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.21 – Результат прохождения тест-кейса 21 для добавления исторической справки

Количество баллов за задание

10

Подсказка

Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.22 – Результат прохождения тест-кейса 22 для добавления исторической справки

Количество баллов за задание

10

Подсказка

Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.23 – Результат прохождения тест-кейса 23 для добавления исторической справки

Количество баллов за задание

10

Подсказка

Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.24 – Результат прохождения тест-кейса 24 для добавления исторической справки

Количество баллов за задание: 10

Подсказка: Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить историческую справку

Завершить добавление

Добавить вопрос

Добавить историческую справку

Некорректные символы в составе названия
Название: Памятник&^

Некорректные символы в составе описания
Описание: <Памятник ювелиру>

Выберите файл или перетащите его сюда (только для изображений)

Добавить

Рисунок Г.25 – Результат прохождения тест-кейса 25 для добавления исторической справки

Количество баллов за задание: 10

Подсказка: Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить историческую справку

Завершить добавление

Добавить вопрос

Добавить историческую справку

Некорректные символы в составе названия
Название: Памятник&^

Некорректные символы в составе описания
Описание: <Памятник ювелиру>

Выберите файл или перетащите его сюда (только для изображений)

Добавить

Рисунок Г.26 – Результат прохождения тест-кейса 26 для добавления исторической справки

Количество баллов за задание

10

Подсказка

Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.27 – Результат прохождения тест-кейса 27 для добавления исторической справки

Количество баллов за задание

10

Подсказка

Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.28 – Результат прохождения тест-кейса 28 для добавления исторической справки

Количество баллов за задание

10

Подсказка

Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить

Завершить добавление

Добавить вопрос

Добавить историческую справку

Памятник

Некорректные символы в составе описания

%Памятник ювелиру%

Выберите файл или перетащите его сюда (только для изображений)

Добавить

Рисунок Г.29 – Результат прохождения тест-кейса 29 для добавления исторической справки

Количество баллов за задание

10

Подсказка

Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить

Завершить добавление

Добавить вопрос

Добавить историческую справку

Памятник

Некорректные символы в составе описания

%Памятник ювелиру%

Выберите файл или перетащите его сюда (только для изображений)

Добавить

Рисунок Г.30 – Результат прохождения тест-кейса 30 для добавления исторической справки

Количество баллов за задание

10

Подсказка

Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить историческую справку

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.31 – Результат прохождения тест-кейса 31 для добавления исторической справки

Количество баллов за задание

10

Подсказка

Задание

Добавить изображение

Добавить таргет

Добавить историческую справку

Добавить историческую справку

Добавить

Завершить добавление

Добавить вопрос

Рисунок Г.32 – Результат прохождения тест-кейса 32 для добавления исторической справки

Приложение Д
(рекомендуемое)

Исходный код модульного тестирования

Листинг Д.1 – TestNotesController.cs

```

using entitys.DTO;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Moq;
using questie_backend.Controllers;
using questie_backend.Services.Interfaces;
using questie_backend.Tests.Mockdata;

namespace questie_backend.Tests.Controllers
{
    public class TestNotesController
    {
        /// <summary>
        /// GetAllNotes
        /// </summary>
        [Fact]
        public async Task GetAllNotes_ShouldReturn200Status()
        {
            /// Arrange
            var notesServiceMock = new Mock<INotesService>();
            notesServiceMock.Setup(_ => _.GetAllNotesAsync())
                .ReturnsAsync(NotesData.GetTestNotes());
            var notesControllerMock = new
NotesController(notesServiceMock.Object);
            /// Act
            var result = await notesControllerMock.GetAllNotes();
            /// Assert
            Assert.Equal(NotesData.GetTestNotes().Count(),
result.Count());
        }

        /// <summary>
        /// GetNoteById
        /// </summary>
        [Fact]
        public async Task GetNoteById_BadRequestResultWhenIdIsNull()
        {
            /// Arrange
            var notesServiceMock = new Mock<INotesService>();
            var notesControllerMock = new
NotesController(notesServiceMock.Object);
            /// Act
            var response = await
notesControllerMock.GetNoteById(null);
            /// Assert
            Assert.IsType<BadRequestObjectResult>(response.Result);
        }

        [Fact]
        public async Task GetNoteById_BadRequestResultWhenIdIsEmpty()
        {
            /// Arrange
            var notesServiceMock = new Mock<INotesService>();
            var notesControllerMock = new
NotesController(notesServiceMock.Object);

```

```

        /// Act
        var response = await
notesControllerMock.GetNoteById(Guid.Empty);
        /// Assert
        Assert.IsType<BadRequestObjectResult>(response.Result);
    }

    [Fact]
    public async Task GetNoteById_NotFoundResultWhenUserNotFound()
    {
        /// Arrange
        Guid testNoteId = new Guid("6c17bd3b-a853-44ea-a193-
359a2bfe5de4");
        var notesServiceMock = new Mock<INotesService>();
        notesServiceMock.Setup(_ =>
_.GetNoteByIdAsync(testNoteId)).ReturnsAsync(null as NoteOutDto);
        var controller = new
NotesController(notesServiceMock.Object);
        /// Act
        var response = await controller.GetNoteById(testNoteId);
        /// Assert
        Assert.IsType<NotFoundResult>(response.Result);
    }

    [Fact]
    public async Task GetNoteById_NoteOutDto()
    {
        /// Arrange
        Guid testNoteId = new Guid("6c17bd3b-a853-44ea-a193-
359a2bfe5de1");
        var notesServiceMock = new Mock<INotesService>();
        notesServiceMock.Setup(_ =>
_.GetNoteByIdAsync(testNoteId))

.ReturnsAsync(NotesData.GetTestNotes().FirstOrDefault(p => p.Id ==
testNoteId));
        var controller = new
NotesController(notesServiceMock.Object);
        /// Act
        var response = await controller.GetNoteById(testNoteId);
        /// Assert
        var note = Assert.IsType<NoteOutDto>(response.Value);
        Assert.Equal("Note One", note.Title);
        Assert.Equal("Note Description One", note.Description);
        Assert.Equal("https://localhost:5003/Images/Notes/note_one.jpg",
note.ImageSrc);
    }
    /// <summary>
    /// CreateNote
    /// </summary>
    [Fact]
    public async Task CreateNote_CreatedAtAction()
    {
        /// Arrange
        var fileMock = new Mock<IFormFile>();
        fileMock.Setup(f => f.FileName).Returns("example.jpg");
    }

```

```

        fileMock.Setup(f => f.Length).Returns(1024); // Set file
length as needed

        NoteDto noteDto = NotesData.GetNewNoteDto();
        noteDto.ImageFile = fileMock.Object;
        var notesServiceMock = new Mock<INotesService>();
        notesServiceMock.Setup(_ =>
            _.CreateNoteAsync(It.IsAny<NoteDto>()))
            .ReturnsAsync(NotesData.GetNewNote());

        var controller = new
NotesController(notesServiceMock.Object);
        /// Act
        var response = await controller.CreateNote(noteDto);
        /// Assert
        var note =
Assert.IsType<CreatedAtActionResult>(response);
        notesServiceMock.Verify(_ =>
            _.CreateNoteAsync(It.IsAny<NoteDto>()), Times.Exactly(1));
    }

    [Fact]
    public async Task
CreateNote_BadRequestResultWhenTitleFieldIsNull()
    {
        /// Arrange
        var fileMock = new Mock<IFormFile>();
        fileMock.Setup(f => f.FileName).Returns("example.jpg");
        fileMock.Setup(f => f.Length).Returns(1024); // Set file
length as needed
        NoteDto noteDto = NotesData.GetNullTitleNoteDto();
        noteDto.ImageFile = fileMock.Object;
        var notesServiceMock = new Mock<INotesService>();
        var controller = new
NotesController(notesServiceMock.Object);
        /// Act
        var response = await controller.CreateNote(noteDto);
        /// Assert
        Assert.IsType<BadRequestObjectResult>(response);
    }

    [Fact]
    public async Task
CreateNote_BadRequestResultWhenDescriptionFieldIsNull()
    {
        /// Arrange
        var fileMock = new Mock<IFormFile>();
        fileMock.Setup(f => f.FileName).Returns("example.jpg");
        fileMock.Setup(f => f.Length).Returns(1024);
        NoteDto noteDto = NotesData.GetNullDescriptionNoteDto();
        noteDto.ImageFile = fileMock.Object;
        var notesServiceMock = new Mock<INotesService>();
        var controller = new
NotesController(notesServiceMock.Object);
        /// Act
        var response = await controller.CreateNote(noteDto);

```

```

        /// Assert
        Assert.IsType<BadRequestObjectResult>(response);
    }

    [Fact]
    public async Task
CreateNote_BadRequestResultWhenFileFieldIsNull()
    {
        /// Arrange
        NoteDto noteDto = NotesData.GetNewNoteDto();
        var notesServiceMock = new Mock<INotesService>();
        var controller = new
NotesController(notesServiceMock.Object);
        /// Act
        var response = await controller.CreateNote(noteDto);
        /// Assert
        Assert.IsType<BadRequestObjectResult>(response);
    }

    /// <summary>
    /// DeleteNote
    /// </summary>
    [Fact]
    public async Task DeleteNote_BadRequestResultWhenIdIsNull()
    {
        /// Arrange
        var notesServiceMock = new Mock<INotesService>();
        var notesControllerMock = new
NotesController(notesServiceMock.Object);

        /// Act
        var response = await
notesControllerMock.DeleteNote(null);
        /// Assert
        Assert.IsType<BadRequestObjectResult>(response);
    }

    [Fact]
    public async Task DeleteNote_BadRequestResultWhenIdIsEmpty()
    {
        /// Arrange
        var notesServiceMock = new Mock<INotesService>();
        var notesControllerMock = new
NotesController(notesServiceMock.Object);
        /// Act
        var response = await
notesControllerMock.DeleteNote(Guid.Empty);
        /// Assert
        Assert.IsType<BadRequestObjectResult>(response);
    }

    [Fact]
    public async Task DeleteNote_NotFoundResultWhenNoteNotFound()
    {
        /// Arrange
        Guid noteId = new Guid("6c17bd3b-a853-44ea-a193-
359a2bfe5de4");

```

```

        var notesServiceMock = new Mock<INotesService>();
        notesServiceMock.Setup(_ =>
            _.DeleteNoteAsync(It.IsAny<Guid>()))
            .ReturnsAsync(false);
        var controller = new
NotesController(notesServiceMock.Object);
        /// Act
        var response = await controller.DeleteNote(noteId);
        /// Assert
        Assert.IsType<NotFoundResult>(response);
    }

    [Fact]
    public async Task DeleteNote_NoContentResultWhenAllOk()
    {
        Guid noteId = new Guid("6c17bd3b-a853-44ea-a193-
359a2bfe5de1");
        var notesServiceMock = new Mock<INotesService>();
        notesServiceMock.Setup(_ =>
            _.DeleteNoteAsync(It.IsAny<Guid>()))
            .ReturnsAsync(true);
        var controller = new
NotesController(notesServiceMock.Object);
        /// Act
        var response = await controller.DeleteNote(noteId);
        /// Assert
        Assert.IsType<NoContentResult>(response);
    }

    /// <summary>
    /// UpdateNote
    /// </summary>
    [Fact]
    public async Task UpdateNote_BadRequestResultWhenIdIsNull()
    {
        /// Arrange
        var notesServiceMock = new Mock<INotesService>();
        var notesControllerMock = new
NotesController(notesServiceMock.Object);
        /// Act
        var response = await
notesControllerMock.UpdateNote(null, NotesData.GetNewNoteDto());
        /// Assert
        Assert.IsType<BadRequestObjectResult>(response);
    }

    [Fact]
    public async Task UpdateNote_BadRequestResultWhenIdIsEmpty()
    {
        /// Arrange
        var notesServiceMock = new Mock<INotesService>();
        var notesControllerMock = new
NotesController(notesServiceMock.Object);
        /// Act
        var response = await
notesControllerMock.UpdateNote(Guid.Empty, NotesData.GetNewNoteDto());
        /// Assert
        Assert.IsType<BadRequestObjectResult>(response);
    }

```

```

[Fact]
public async Task UpdateNote_BadRequestResultWhenNoteIsNull()
{
    /// Arrange
    Guid noteId = new Guid("6c17bd3b-a853-44ea-a193-
359a2bfe5de1");
    var notesServiceMock = new Mock<INotesService>();
    var notesControllerMock = new
NotesController(notesServiceMock.Object);
    /// Act
    var response = await
notesControllerMock.UpdateNote(noteId, null);
    /// Assert
    Assert.IsType<BadRequestObjectResult>(response);
}

[Fact]
public async Task UpdateNote_NotFoundResultWhenNoteNotFound()
{
    /// Arrange
    Guid noteId = new Guid("6c17bd3b-a853-44ea-a193-
359a2bfe5de4");
    var notesServiceMock = new Mock<INotesService>();
    notesServiceMock.Setup(_ =>
_.UpdateNoteAsync(It.IsAny<Guid>(),
It.IsAny<NoteDto>())) .ReturnsAsync(false);
    var controller = new
NotesController(notesServiceMock.Object);
    /// Act
    var response = await controller.UpdateNote(noteId,
NotesData.GetNewNoteDto());
    /// Assert
    Assert.IsType<NotFoundResult>(response);
}

[Fact]
public async Task UpdateNote_NoContentResultWhenAllOk()
{
    /// Arrange
    Guid noteId = new Guid("6c17bd3b-a853-44ea-a193-
359a2bfe5de1");
    var notesServiceMock = new Mock<INotesService>();
    notesServiceMock.Setup(_ =>
_.UpdateNoteAsync(It.IsAny<Guid>(),
It.IsAny<NoteDto>())) .ReturnsAsync(true);
    var controller = new
NotesController(notesServiceMock.Object);
    /// Act
    var response = await controller.UpdateNote(noteId,
NotesData.GetNewNoteDto());
    /// Assert
    Assert.IsType<NoContentResult>(response);
}
}
}

```

Листинг Д.2 – NotesData.cs

```

using entitys.DTO;
using entitys.Models;

namespace questie_backend.Tests.Mockdata
{
    public class NotesData
    {
        public static List<NoteOutDto> GetTestNotes()
        {
            return new List<NoteOutDto>
            {
                new NoteOutDto
                {
                    Id = new Guid("6c17bd3b-a853-44ea-a193-
359a2bfe5de1"),
                    Title = "Note One",
                    Description = "Note Description One",
                    ImageSrc =
"https://localhost:5003/Images/Notes/note_one.jpg"
                },
                new NoteOutDto
                {
                    Id = new Guid("6c17bd3b-a853-44ea-a193-
359a2bfe5de2"),
                    Title = "Note Second",
                    Description = "Note Description Second",
                    ImageSrc =
"https://localhost:5003/Images/Notes/note_second.jpg"
                },
                new NoteOutDto
                {
                    Id = new Guid("6c17bd3b-a853-44ea-a193-
359a2bfe5de3"),
                    Title = "Note Third",
                    Description = "Note Description Third",
                    ImageSrc =
"https://localhost:5003/Images/Notes/note_third.jpg"
                },
            };
        }

        public static NoteDto GetNullTitleNoteDto()
        {
            return new NoteDto
            {
                Title = null,
                Description = "Note Description Fourth",
                ImageFile = null,
            };
        }

        public static NoteDto GetNullDescriptionNoteDto()
        {
            return new NoteDto
            {

```

```

        Title = "Note Fourth",
        Description = null,
        ImageFile = null,
    };
}

public static NoteDto GetNewNoteDto()
{
    return new NoteDto
    {
        Title = "Note Fourth",
        Description = "Note Description Fourth",
        ImageFile = null,
    };
}

public static Note GetNewNote()
{
    return new Note
    {
        Id = new Guid("6c17bd3b-a853-44ea-a193-
359a2bfe5de4"),
        Title = "Note Fourth",
        Description = "Note Description Fourth",
        ImageName = "example.jpg"
    };
}
}
}

```