

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет

Кафедра

вычислительной техники

«Вычислительная техника»

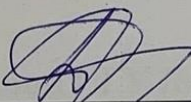
Специальность: 09.05.01 «Применение и эксплуатация
автоматизированных систем специального
назначения»
Специализация: «Эксплуатация вычислительных машин,
комплексов, систем и сетей специального
назначения»

ДИПЛОМНЫЙ ПРОЕКТ

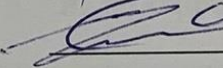
на тему

ПРИЛОЖЕНИЕ ДЛЯ СОЗДАНИЯ ОНЛАЙН-КУРСОВ ДЛЯ
ОБУЧЕНИЯ СПЕЦИАЛИСТОВ

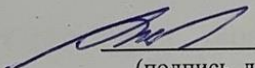
Студент


(подпись, дата) 15.06.24 (ФИО полностью) Степанова Дарья Павловна

Руководитель
(подпись, дата)


(подпись, дата) 15.06.24 Зинкин С.А.
(фамилия, инициалы)

Нормоконтролёр

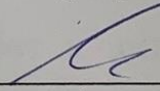

(подпись, дата) 17.06.24 Бычков А.С.
(фамилия, инициалы)

Рецензент

каф. вычислительной техники
17.06.24
(должность, место работы) (подпись, дата) (фамилия, инициалы) Абрамкин А.Ю.

Работа допущена к защите (протокол зас. кафедры от 27.05.2024 г. № 11)

Заведующий кафедрой


(подпись) Митрохин М.А.
(фамилия, инициалы)

Работа защищена с отметкой отлично

(протокол заседания ГЭК от 19.06.2024 № 12)

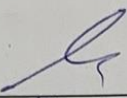
Секретарь ГЭК


(подпись)

Гурин Е.И.
(фамилия, инициалы)

Пенза 2024

«Утверждаю»
Заведующий кафедрой


Митрохин М.А.
«19» 04 2024 г.

ЗАДАНИЕ НА ДИПЛОМНЫЙ ПРОЕКТ

1. Студент Степанова Дарья Павловна гр.19ВОЭ1 факультета
вычислительной техники.

2. Тема работы: «Приложение для создания онлайн-курсов для обучения
специалистов».

Тема утверждена приказом ПГУ № 405/0 от 19.04.2024 г.

3. Руководитель работы д.т.н., профессор каф. ВТ «ПГУ» Зинкин С.А.

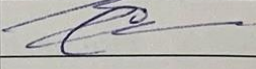
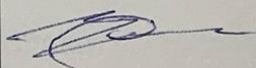
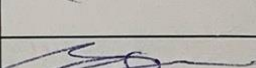
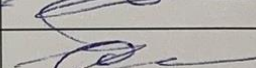
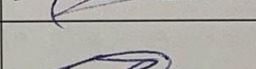
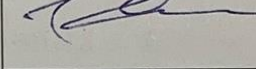
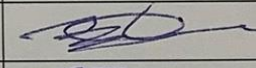
4. Задание на работу (назначение разработки, исходные данные и т.п.).

Разработать приложение для создания-онлайн курсов для обучения
специалистов. Приложение для создания онлайн-курсов для обучения
специалистов должна выполнять следующие функции: авторизация и
регистрация в приложении; создание и размещение курса; просмотр курса,
уроков; загрузка лабораторных работ и выполнение тестов; возможность
отслеживать прогресс каждого студента; доступ к курсу и его настройка;
просмотр видео для урока; возможность оставить комментарий. Приложение
для создания онлайн-курсов для обучения специалистов должна
разрабатываться с помощью среды разработки WebStorm 2023. База данных
должна разрабатываться для СУБД «PostgreSQL».

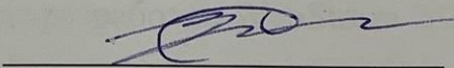
5. Перечень подлежащих разработке вопросов.

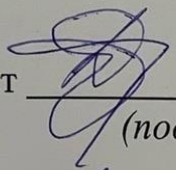
Требуется произвести анализ предметной области, определить функциональный состав программы, выбрать стек технологий для реализации программы, разработать алгоритм работы программы, разработать диаграммы: прецедентов, активностей, последовательностей, объектов, классов, с последующим написанием программы.

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1.	Анализ предметной области	10	15.03.2024	
2.	Проектирование программного приложения	10	09.04.2024	
3.	Разработка базы данных	20	21.04.2024	
4.	Отладка и тестирование	10	18.05.2024	
5.	Разработка экономической и бжд частей	10	20.05.2024	
6.	Оформление графической части	10	24.05.2024	
7.	Оформление выпускной работы	30	31.05.2024	

Дата выдачи задания 19.04 2024 года.

Руководитель дипломного проекта  Зинкин С.А.
(подпись, дата)

Задание к исполнению принял студент  Степанова Д.П.
(подпись, дата)

Работу к защите допустить
Декан факультета  Фионова Л.Р.
19.06.2024 (подпись, дата)

Реферат

Пояснительная записка содержит 214 листов, 88 рисунков, 12 таблиц, 11 источников и 3 приложения

ВЕБ-ПРИЛОЖЕНИЕ, ПЛАТФОРМА ДЛЯ КУРСОВ, ОНЛАЙН-ОБРАЗОВАНИЕ, ОБУЧЕНИЕ СПЕЦИАЛИСТОВ, БАЗА ДАННЫХ, REST API, CRUD, JAVASCRIPT, NODEJS, POSTGRESQL

Цель работы – разработать приложения для создания онлайн-курсов для обучения специалистов.

Разработанное приложение предназначено для создание курсов и обучения специалистов по выбранному направлению. Приложение позволяет вести регистрацию и авторизацию пользователей, размещать и создавать курсы, просматривать уроки, выполнять и загружать лабораторные или тестовые задания, оценивать результаты выполненных работ.

В результате проделанной работы разработано приложение для создания онлайн-курсов для обучения специалистов, состоящее из базы данных и клиент-серверного приложения для взаимодействия с базой данных.

Хранение данных о пользователях производится в базе PostgreSQL в связке с графическим клиентом pgAdmin4. Клиент-серверное приложение разработано на языке программирования JavaScript в среде разработки WebStorm 2023.

ПГУ 1.090501.10.001 ПЗ				
Изм	Лист	№ докум.	Подпись	Дата
Разраб.		Степанова Д.П.		15.06.24
Пров.		Зинкин С.А.		15.06.24
Н. Контр.		Бычков А.С.		17.06.24
Утв.		Митрохин М.А.		17.06.24

Литера	Лист	Листов
	2	214

Приложение для создания онлайн-курсов для обучения специалистов

Пояснительная записка

ФВТ, каф. ВТ
гр. 19ВОЭ1

Содержание

Введение.....	5
1 Приложение для создания онлайн-курсов для обучения специалистов	7
1.1 Анализ предметной области	7
1.1.1 Актуальность разработки.....	7
1.1.2 Анализ существующих аналогов	8
1.1.3 Постановка задачи	17
1.2 Выбор технологий и средств разработки приложения	19
1.2.1 Выбор языка программирования.....	19
1.2.2 Выбор СУБД.....	19
1.2.3 Клиентская часть.....	20
1.2.4 Серверная часть.....	23
1.3 Проектирование информационной системы	25
1.3.1 Диаграмма вариантов использования	25
1.3.2 Архитектура приложения.....	29
1.3.3 Проектирование базы данных	44
1.4 Разработка приложения	48
1.4.1 Разработка БД.....	48
1.4.2 Описание программы	52
1.4.3 Тестирование и отладка.....	65
2 Техничко-экономическое обоснование разработки	76
3 Безопасность жизнедеятельности.....	84
Заключение	86
Список использованных источников	87
Приложение А – Модели базы данных.....	88
Приложение Б – UML-диаграммы.....	91
Приложение В – Листинг программы.....	104
Приложение Г - Презентация.....	208

Перечень сокращений и обозначений

В настоящей работе применяют следующие обозначения и сокращения

БД – база данных

Веб-приложение – компьютерная программа, которая запускается в браузере

СУБД – система управления базами данных

API (Application Programming Interface) – описание способов, которыми одна компьютерная программа может взаимодействовать с другой программой

FK (Foreign key) – внешний ключ, понятие теории реляционных баз данных, относящееся к ограничениям целостности базы данных

GUI (Graphical User Interface) – графический интерфейс пользователя

HTML – язык гиперстрочковой разметки текста

ID (Identifier) – уникальный номер объекта

IDE (Integrated development environment) – интегрированная среда разработки

JS – язык программирования

JSON (JavaScript Object Notation) – текстовый формат обмена данными

LCMS – система управления учебным контентом

LMS – система управления обучением

PK (Primary key) – первичный ключ, понятие теории реляционных баз данных, относящееся к ограничениям целостности базы данных

SCORM – международный стандарт для определения технических характеристик

SQL (Structured Query Language) – язык структурированных запросов

UI – пользовательский интерфейс

UML (Unified Modeling Language) – унифицированный язык моделирования

Введение

В настоящее время онлайн-образование пользуется значительным спросом в обществе. Оно активно пропагандируется образовательными учреждениями и стало объектом пристального внимания государства. Эта тенденция способствует росту рынка образовательных услуг и повышению доступности образования для всех слоев общества[1].

Люди всегда стремились к получению новых знаний и передаче информации следующему поколению. Первоначально было достаточно элементарных рисунков, которые позже превратились в книги. По мере увеличения объема информации возникла необходимость не просто передавать знания, но и разъяснять их основополагающее значение. Точно так же, как со временем менялись методы передачи и хранения информации, образование адаптировалось к современным тенденциям. Отличительная черта современного онлайн-образования, в отличие от традиционных образовательных систем, заключается в его способности адаптировать процесс обучения к индивидуальным потребностям каждого учащегося.

Ведущие университеты по всему миру в настоящее время предлагают онлайн-курсы по широкому спектру дисциплин. Корпорации все чаще обращаются к онлайн-платформам для проведения обучения своих сотрудников. Более того, есть учреждения, которые полностью посвящают себя онлайн-курсам. Преимущества очевидны как для преподавателей, так и для учащихся: преподаватели избавлены от необходимости многократно преподавать один и тот же материал разным группам, а учащиеся больше не привязаны к физическому классу, что, помимо прочих преимуществ, позволяет маркетологам ориентироваться на более широкую клиентскую базу.

Резкий рост популярности онлайн-образования является реакцией на развитие технологий и рост удаленной работы, поскольку все больше людей стремятся получить образование, не выходя из дома. Этот сдвиг вызвал растущий спрос на онлайн-образовательные услуги. Платформы онлайн-обучения

потенциально могут сделать высшее образование более доступным для различных групп студентов. Кроме того, интеграция мультимедийных и интерактивных технологий в преподавание может повысить его эффективность и привести к повышению успеваемости студентов.

Целью данной выпускной работы является:

- изучение предметной области и анализ существующих решений на рынке;
- разработка приложения для создания и размещения курсов для обучения специалистов;
- проектирование и разработка пользовательского интерфейса;
- тестирование разработанного приложения;
- внедрение и развертывание приложения.

Приложение для создания онлайн-курсов позволит упростить и автоматизировать процесс обучения для специалистов, а также даст возможность обучаться и получать знания из любой точки.

					ПГУ 1.090501.10.001 ПЗ	Лист
						6
Изм.	Лист	№ докум.	Подпись	Дата		

1 Приложение для создания онлайн-курсов для обучения специалистов

1.1 Анализ предметной области

1.1.1 Актуальность разработки

Процесс дистанционного обучения предоставляет возможности передачи знаний и связь между преподавателем и студентом, включая в себя разнообразные технологии, такие как видеосвязь, онлайн-курсы, встроенные чаты. Суть обучения заключается в самостоятельном разборе материала с возможностью получения обратной связи от преподавателя. Дистанционное обучение позволяет ученикам иметь доступ к образованию в любое время и в любом месте. Дистанционное обучение подразумевает разработку ПО, а также разработку образовательных курсов.

Электронное обучение, используется для распространения части полной учебной программы, будь то в рамках учебных заведений или в рамках обязательных курсов, тренингов или комплексных программ дистанционного обучения. Используя компьютеры для всех видов деятельности, электронное обучение в широком смысле можно охарактеризовать как онлайн-обучение или виртуальное обучение в режиме реального времени. Сфера открытого дистанционного образования в России особенно обширна: различные платформы электронного обучения проводят пробные тесты для участия в ряде конкурсных экзаменов, в том числе в области инженерии, медицины, менеджмента, коммерции и других областях[2].

Онлайн-обучение обладает разнообразными преимуществами и областями применения, включая его эффективность в обучении студентов, его использование для профессионального развития, его экономическую эффективность в свете роста платы за обучение и его потенциал для предоставления образования в глобальном масштабе.

1.1.2 Анализ существующих аналогов

На данный момент существует ряд онлайн площадок для создания онлайн-курсов. Основная цель таких платформ является предоставление инструментов и ресурсов, которые помогут пользователям создавать качественные и результативные онлайн-курсы. Самыми популярными в России являются:

iSpring представляет собой комплексную экосистему, предназначенную для облегчения создания и распространения образовательного контента в режиме онлайн. Набор программных решений, предлагаемых компанией, разработан для совместной работы или в качестве автономных утилит, адаптированных к конкретным потребностям в обучении.

Ниже перечислены основные преимущества использования iSpring в качестве вашего партнера в области образовательных технологий:

- широкий спектр предлагаемого программного обеспечения: поскольку iSpring Learn и Page ориентированы на облачные технологии, они избавляют от необходимости локальной установки и работают непосредственно в вашем веб-браузере;
- интуитивно понятный пользовательский интерфейс;
- подробная аналитика прогресса в обучении: эксклюзивная для iSpring Learn функция позволяет создавать многоплановые отчеты с подробным описанием показателей прохождения курса и вовлеченности учащихся;
- строгие меры безопасности контента: для сред, где конфиденциальность имеет первостепенное значение, а подключение к Интернету ограничено, iSpring Learn может быть развернут на внутреннем сервере, эффективно изолированном от внешних сетей;
- бесплатное мобильное приложение iSpring Learn: приложение повышает гибкость обучения, позволяя пользователям загружать материалы для изучения в автономном режиме и позже синхронизировать свои результаты с платформой, как только подключение к Интернету будет восстановлено.

iSpring является оплотом инноваций, предлагая широкий спектр решений, отвечающих растущим требованиям цифрового образования, обеспечивая простоту использования, безопасность контента и всестороннюю аналитику обучения.

Недостатки iSpring:

- нет вебинарной комнаты;
- завышенная стоимость продуктов .

На рисунке 1 представлена главная страница системы ispring.

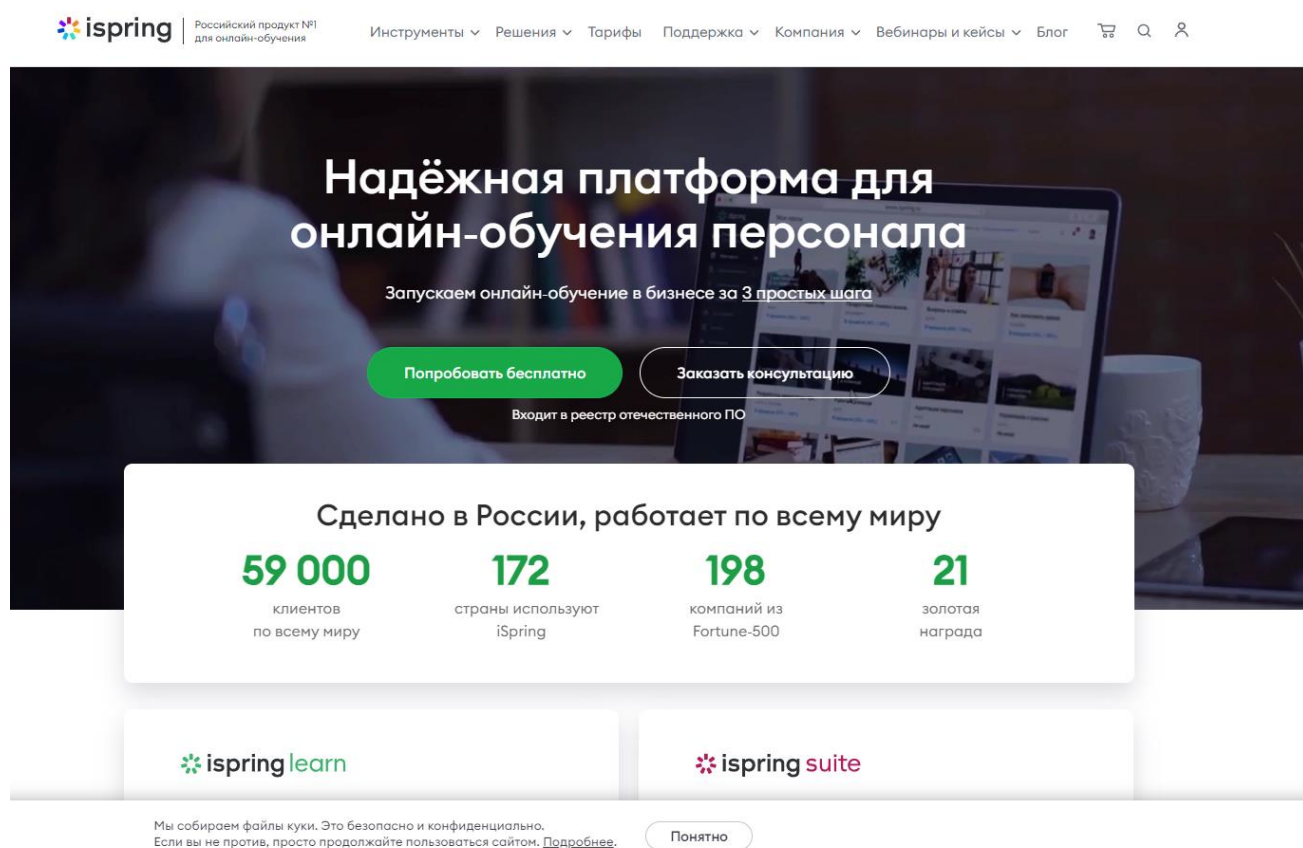


Рисунок 1 – Главная страница сайта iSpring

Moodle - востребованная платформа электронного обучения, используемая университетами по всему миру. Она переведена на более чем 100 языков и предлагает разнообразные возможности для создания и управления онлайн-курсами. Помимо локальной установки, moodle также предлагает облачное хранилище. Управление данной системой требует определенных навыков, поэтому

					ПГУ 1.090501.10.001 ПЗ	Лист
						9
Изм.	Лист	№ докум.	Подпись	Дата		

часто вводится отдельный специалист. С помощью различных модулей и плагинов можно настроить функционал и дизайн электронных курсов.

Moodle, знаменитая система управления обучением с открытым исходным кодом (LMS), предлагает множество преимуществ преподавателям и учреждениям, стремящимся обеспечить качественное онлайн-образование.

Преимущества:

- экономичное решение: Moodle доступен бесплатно, при условии, что вы выберете стандартную версию;
- систематизированная среда обучения: Moodle поддерживает последовательную структуру курса, которая включает в себя темы, мультимедийные учебные пособия, лекции и задания для всестороннего изучения;
- совместимость с SCORM: Moodle облегчает интеграцию предварительно разработанных курсов в формате SCORM, обеспечивая совместимость и возможность повторного использования контента;
- углубленная аналитика: встроенные инструменты отчетности позволяют создавать подробные отчеты о деятельности и успеваемости, помогая отслеживать прогресс учащихся;
- возможности сертификации: с помощью соответствующего плагина Moodle может выдавать сертификаты, повышая эффективность процесса обучения;
- неограниченный доступ пользователей: в своей бесплатной версии платформа не накладывает никаких ограничений на пользователей, что делает ее масштабируемой для любого числа учащихся.

Недостатки:

- сложность пользовательского интерфейса: бесплатная версия Moodle представляет собой сложный интерфейс, который требует знакомства с управлением сервером до создания курса, и для расширения функциональности часто требуются дополнительные плагины;
- отсутствие прямой технической поддержки. Несмотря на то, что у Moodle есть обширное сообщество специалистов по оказанию помощи, отсутствие

специализированной технической поддержки означает, что вам, возможно, потребуется нанять консультанта для получения персонализированной помощи;

– дополнительные расходы: несмотря на бесплатный характер платформы, могут возникнуть расходы, связанные с размещением сервера, приобретением плагинов и администрированием платформы, которые следует учитывать в общем бюджете.

Таким образом, Moodle предлагает гибкую и надежную платформу для онлайн-обучения с сильным акцентом на настройку и поддержку сообщества, хотя для полного использования ее возможностей могут потребоваться дополнительные инвестиции в техническую экспертизу и инфраструктуру.

На рисунке 2 представлена главная страница системы moodle.

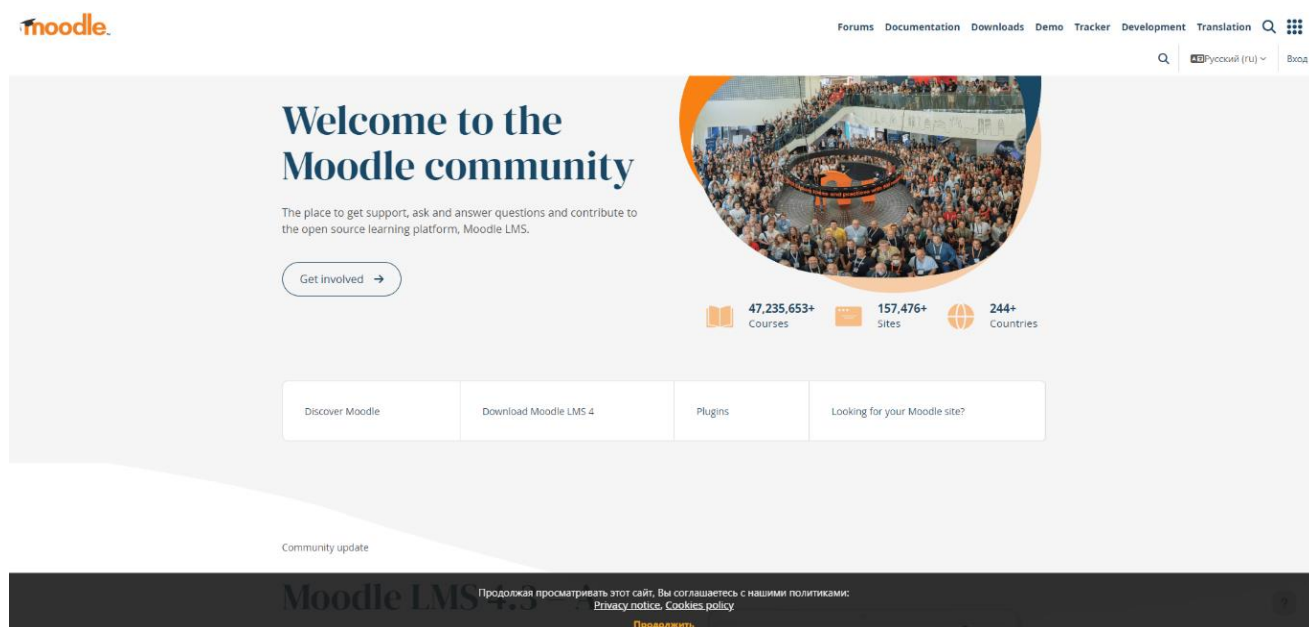


Рисунок 2 – Главная страница сайта Moodle

Stepik является универсальной платформой для разработки курсов онлайн. Сайт даёт возможность выбрать различные инструменты для создания курса, включая текстовые материалы, видеоуроки и практические задания.

Преимущества Stepik:

– установка не требуется: Stepik работает как веб-платформа, что исключает необходимость установки программы. Пользователи могут просто

					ПГУ 1.090501.10.001 ПЗ	Лист
						11
Изм.	Лист	№ докум.	Подпись	Дата		

зарегистрироваться и, при желании, оформить платную подписку для доступа к дополнительным функциям;

- разнообразные типы заданий: платформа предлагает широкий спектр заданий, включая тесты, упражнения на сортировку, сопоставление заданий, численные задачи, а также средства для обучения программированию и многое другое;

- полная статистика и отчетность: пользователи могут получить доступ к подробной информации о ходе прохождения курса, посещаемости и сроках завершения. Дополнительные функции доступны по тарифным планам;

- аккредитованные сертификаты: Stepik предоставляет возможность выдавать сертификаты государственного образца по окончании курса;

- мобильное приложение: Stepik предлагает мобильное приложение, совместимое как с платформами iOS, так и Android, что повышает доступность и гибкость для пользователей.

Недостатки Stepik:

- отсутствие поддержки;
- плохая защита контента: видеоуроки могут быть защищены только в рамках платных и частных курсов, и их нельзя загрузить. Кроме того, не предусмотрена установка водяных знаков или защита от записи с экрана.

Таким образом, Stepik предлагает полный набор инструментов для создания и развертывания курсов, поддерживаемых надежной статистикой и мобильным доступом. Однако отсутствие поддержки SCORM на платформе и ограниченные возможности защиты контента могут стать проблемой для некоторых пользователей, особенно для тех, у кого особые требования к функциональной совместимости и безопасности. Стоит отметить, что платформа не смотря на недостатки, является одной из ведущих на рынке онлайн-образования, но из-за отсутствия поддержки многие вынуждены переходить на другие платформы.

На рисунке 3 представлена главная страница системы stepik.

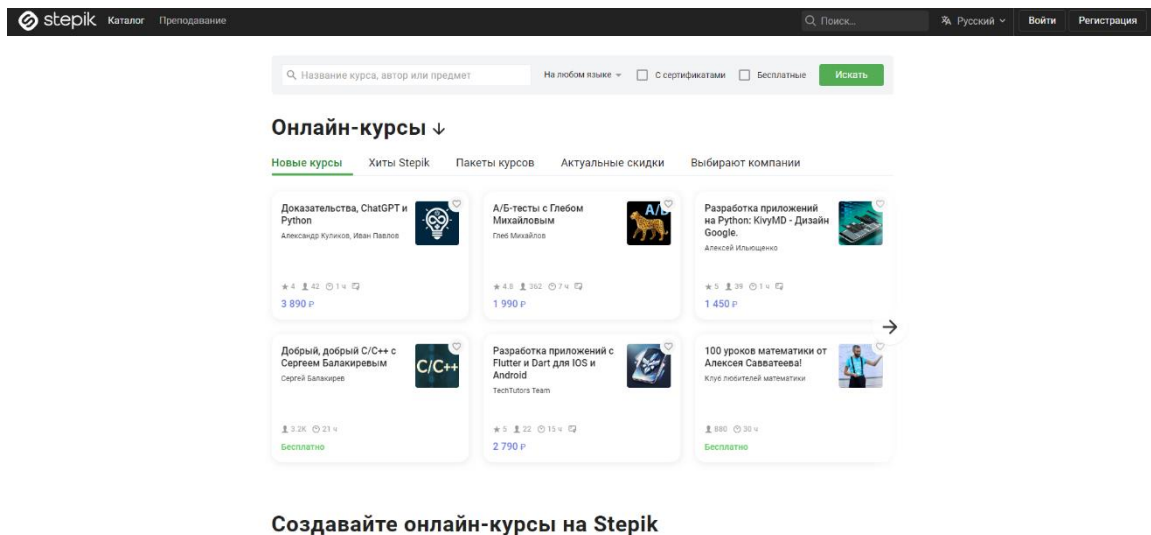


Рисунок 3 – Главная страница сайта Stepik

Anti-Trainings представляет собой универсальную платформу для создания онлайн-образовательных учреждений, предлагающую такие функции, как упрощение оплаты обучения, проведение вебинаров, инструменты оценки учащихся и каналы коммуникации с клиентами. Она предназначена для небольших проектов, которые не требуют сложных конфигураций и сложных маркетинговых возможностей.

Преимущества Anti-Trainings:

- удобный интерфейс: платформа может похвастаться интуитивно понятным интерфейсом, обеспечивающим беспрепятственный пользовательский опыт без сложных этапов обучения;
- последовательная структура курса: Anti-Trainings придерживается структурированного подхода к организации курса, способствуя согласованности и ясности в предоставлении образовательного контента;
- разнообразие способов оплаты: программа поддерживает различные способы оплаты, включая платежи по картам (как внутри страны, так и за рубежом), транзакции операторов связи, банковские чеки и даже варианты кредитов и рассрочки;

- аналитика и отчетность: пользователи могут создавать подробные отчеты о качестве уроков и вовлеченности учащихся, а платная версия предлагает возможность создавать до 14 отдельных отчетов;
- выдача сертификатов: платформа позволяет создавать индивидуальные сертификаты, позволяя пользователям настраивать внешний вид, указывая расположение названия, выбор шрифта и цветовую гамму;
- защита контента курса: защита от тренингов включает меры по защите контента курса от несанкционированного использования или распространения;
- бесплатное мобильное приложение: платформа предлагает бесплатное мобильное приложение, совместимое с устройствами iOS и Android, что повышает доступность и гибкость для пользователей.

Недостатки антитренингов:

- устаревший дизайн: возможности платформы в области дизайна ограничены, что обеспечивает минимальную гибкость при настройке внешнего вида уроков и курсов;
- отсутствие возможности создания целевых страниц: пользователи не могут создавать целевые страницы внутри платформы, что требует использования сторонних сервисов, таких как Tilda Publishing. В бесплатной версии Tilda Publishing может отображаться водяной знак, который может ухудшать эстетический вид страницы и потенциально приводить к дополнительным расходам для пользователей, выбирающих платные тарифные планы;
- зависимость от сторонних сервисов вещания: функции вещания ограничены интеграцией с внешними сервисами, такими как Zoom и YouTube, что требует от пользователей загрузки ссылок на свои личные аккаунты для размещения вебинаров.

Таким образом, Anti-Trainings представляет собой удобную в использовании платформу с разнообразными вариантами оплаты, надежными возможностями создания отчетов и функциями защиты контента, хотя для настройки и создания целевой страницы могут потребоваться дополнительные инвестиции, а для

некоторых функций требуются сторонние сервисы. На рисунке 4 представлена главная страница системы Антитренинги.

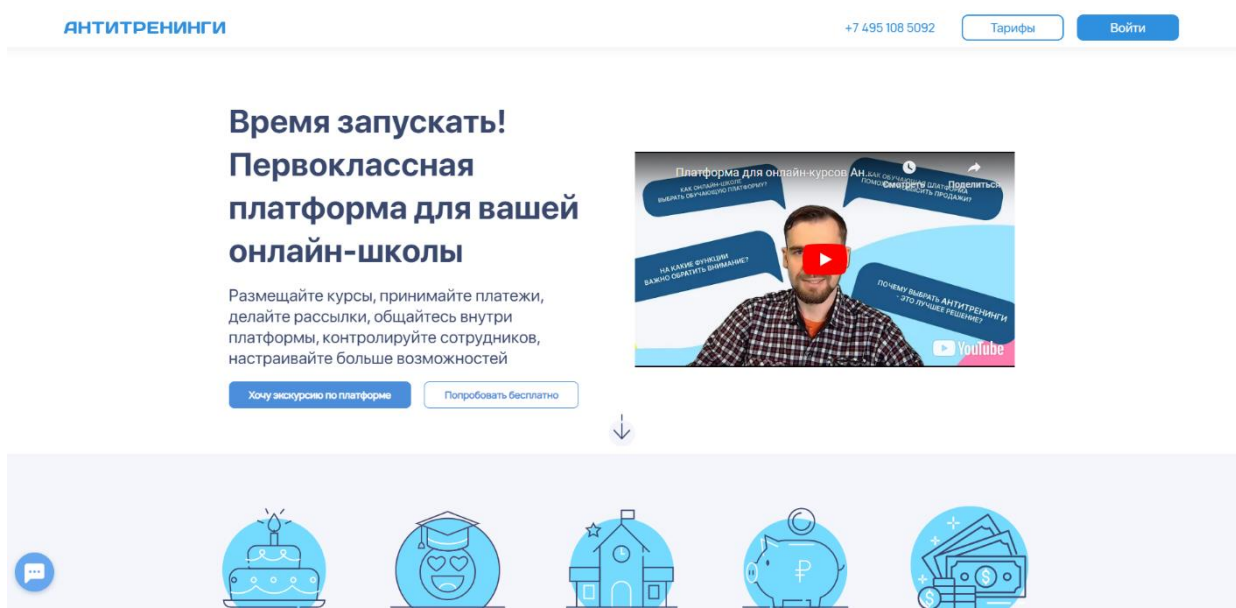


Рисунок 4 – Главная страница сайта Антитренинги

Adobe Captivate - это программное обеспечение профессионального уровня, предназначенное для создания интерактивных и захватывающих курсов. Его универсальный редактор предлагает широкий спектр функциональных возможностей, позволяя создавать программные симуляторы, интерактивные видеоролики и традиционные учебные материалы, что делает его особенно полезным для компаний, которым требуется всестороннее обучение сотрудников.

Преимущества Adobe Captivate:

- универсальные форматы курсов: Adobe Captivate поддерживает создание курсов в формате, близком к традиционным презентациям PowerPoint;
- поддержка дополнительных форматов;
- обширные элементы дизайна: Adobe Captivate предлагает богатый выбор ресурсов для дизайна, включая шаблоны, виджеты и инструменты для редактирования изображений, позволяющие создавать визуально привлекательный и интерактивный контент курса.

Недостатки Adobe Captivate:

- требования к установке: Adobe Captivate необходимо устанавливать на компьютер, что может создать неудобства для пользователей, ищущих более гибкие веб-решения;
- сложный интерфейс: сложность редактора может представлять проблему для начинающих разработчиков, поскольку для полного использования его возможностей может потребоваться обучение;
- недоступность в Российской Федерации: после ухода Adobe из России пользователи в этом регионе не могут продлить свои лицензии или совершить новые покупки, что приводит к отсутствию технической поддержки.

Таким образом, Adobe Captivate предлагает широкий набор функций для создания интерактивных курсов с поддержкой различных форматов курсов и совместимостью со SCORM. Однако необходимость локальной установки и сложность интерфейса могут создавать проблемы, а недоступность платформы в Российской Федерации ограничивает доступность и возможности поддержки для пользователей в регионе. На рисунке 5 представлена главная страница системы Adobe Captivate.

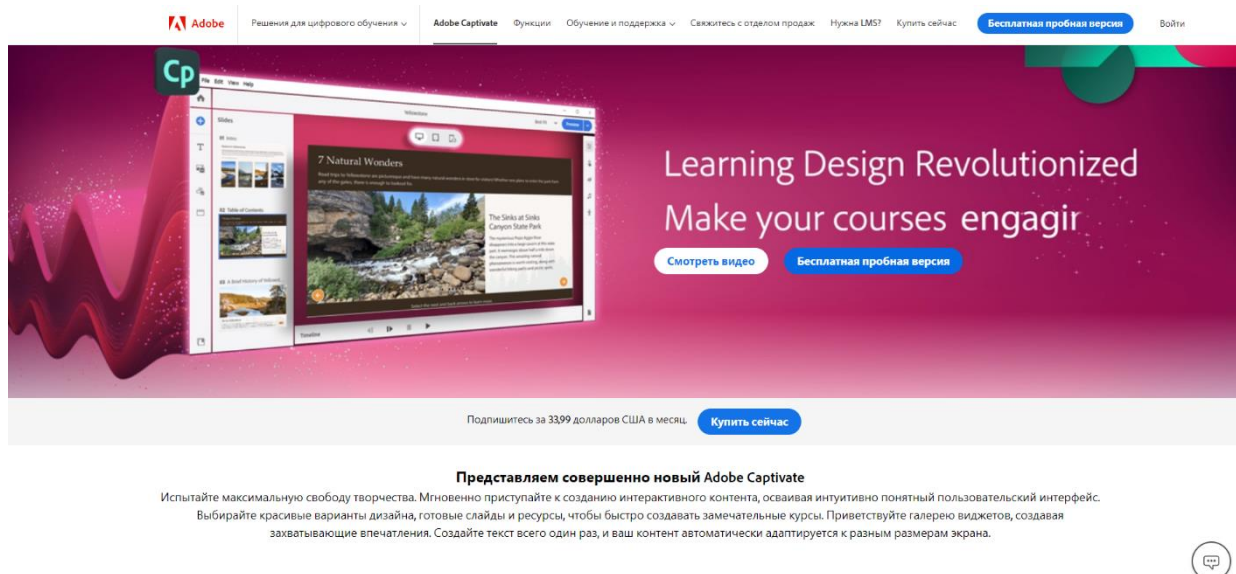


Рисунок 5 – Главная страница сайта Adobe Captivate

					ПГУ 1.090501.10.001 ПЗ	Лист
						16
Изм.	Лист	№ докум.	Подпись	Дата		

Анализ существующих платформ для онлайн-образования показывает, что они зачастую неудобны и предлагают ограниченный функционал. Поэтому возникла необходимость в создании простой системы для создания обучающих курсов с интуитивным интерфейсом. Необходимо разработать платформу под названием "Приложение для создания онлайн курсов для обучения специалистов" с целью обеспечить пользователям возможность создания и публикации курсов без специальных технических навыков. Функциональные требования включают создание контента, разработку тестов и заданий, создание обучающих материалов.

1.1.3 Постановка задачи

Необходимо разработать веб-платформу «Приложение для создания онлайн курсов для обучения специалистов».

Цель данного приложения: создать приложение, которое позволит пользователям без специальных технических знаний создавать и публиковать онлайн-курсы.

Функциональные требования:

- создание контента: позволяет интегрировать текстовые фрагменты, визуальные артефакты, анимационную графику и различные медиа-материалы, а также систематизировать указанный контент по разделам и педагогическим эпизодам;
- проверка и формулирование заданий: разработайте спектр оценочных тестов (множественный выбор, дихотомии "правда/ложь", открытые вопросы), дополните их практическими заданиями, автоматизируйте валидацию тестов и откалибруйте оценки заданий;
- создание педагогических материалов: разработка презентационных модулей и интерактивных компонентов, использование аудиальных и визуальных средств массовой информации, а также проведение викторин и опросов;
- управление курсом: настраивайте параметры доступа к образовательному курсу, создавайте группы пользователей и назначайте разрешения на основе ролей, отслеживайте образовательную траекторию учащихся и анализируйте аналитику, связанную с курсом.

Нефункциональные полномочия:

- интуитивно понятный интерфейс: интерфейс приложения должен быть интуитивно понятным и простым в освоении даже для тех, кто лишен технической хватки;
- универсальная доступность: возможности приложения должны распространяться на различные цифровые устройства (настольные компьютеры, планшеты, смартфоны, ноутбуки);
- масштабируемая архитектура: приложение должно обладать способностью поддерживать большое количество пользователей и множество курсов;
- неизменная надежность: приложение должно обеспечивать бесперебойную работу и гарантировать сохранность пользовательских данных;
- абсолютная безопасность: приложение должно соответствовать строгим требованиям по защите пользовательских данных.

Целевая демографическая группа: эксперты в своей области, виртуозы в различных дисциплинах, желающие поделиться своей эрудицией и опытом с помощью онлайн-курсов; академические учреждения, университеты и другие научные учреждения, стремящиеся интегрировать онлайн-курсы в свой педагогический репертуар.

Для преодоления обозначенных проблем первостепенное значение имеет создание интерфейса, который воплощает простоту и ориентированность на пользователя. Программное обеспечение должно включать в себя широкий набор логических модулей, способствующих легкому расширению новых функциональных возможностей в процессе разработки и быстрому выявлению и устранению аномалий во время тестирования приложений.

Программное обеспечение должно быть разработано таким образом, чтобы оно было совместимо с операционной системой Microsoft Windows 10.

1.2 Выбор технологий и средств разработки приложения

1.2.1 Выбор языка программирования

При создании онлайн-приложения для создания курсов в качестве языка был выбран JavaScript. Известный своей ключевой ролью в разработке веб-приложений, JavaScript отлично зарекомендовал себя в среде браузера. Это язык сценариев, который является неотъемлемой частью не только разработки интерфейса, но и создания серверной архитектуры. Цифровой ландшафт изобилует веб-сайтами на JS, дополненными множеством учебных материалов и туториалов для тех, кто стремится запустить свое цифровое предприятие.

JavaScript - это синоним простоты и доступности; его скрипты удобны для начинающих, а веб - это сокровищница ресурсов для разъяснения или поиска альтернатив его элементам и функциям. HTML и CSS объединяют усилия с JavaScript, чтобы материализовать опыт работы на стороне клиента, гарантируя, что веб-страницы будут не только точно отображаться, но и поражать эстетическим видом. В то время как CSS привносит стиль или, скорее, создает шаблоны для дизайна веб-сайта, сам JavaScript-код оживает в браузере, в то время как серверные операции иницируются в самом приложении.

Универсальность JavaScript очевидна, поскольку он позволяет создавать множество программ, адаптированных к различным функциональным возможностям. Экосистема JavaScript постоянно пополняется множеством фреймворков, предназначенных для оптимизации процессов программирования и упрощения определенных задач, что свидетельствует об эволюции подхода разработчиков к проектам на основе JavaScript.

1.2.2 Выбор СУБД

Основой всех приложений является надежная система управления базами данных (СУБД). Прежде чем приступить к разработке программы, необходимо получить полное представление о предполагаемых данных и требованиях к их хранению.

При создании приложения в качестве базы данных была выбрана PostgreSQL. PostgreSQL, которую называют одной из передовых баз данных, славится своей производительностью и эффективностью, обеспечивая упрощенную обработку информации и при этом обладая масштабируемостью. Широкое внедрение базы данных многими предприятиями свидетельствует о ее надежности в хранении данных, ведении журнала и аварийном восстановлении, что делает ее лидером в своем сегменте. База данных оснащена таким образом, чтобы противостоять аппаратным и программным сбоям, что еще больше укрепляет ее репутацию в области устойчивости.

Отличаясь от других реляционных баз данных, PostgreSQL предлагает широкий спектр возможностей для хранения сложных данных, подкреплённых современной документацией и превосходной расширяемостью. Благодаря поддержке множества языков программирования PostgreSQL становится универсальным инструментом в области разработки, что дополняется его совместимостью с различными операционными системами.

1.2.3 Клиентская часть

В настоящее время существует огромное количество библиотек и фреймворков, предназначенных для реализации визуального аспекта программы, включая Angular, Svelte и React.

Например, Angular хорошо подходит разработчикам, переходящим от серверной разработки к клиентской. Несмотря на то, что его хвалят за простоту и удобство в использовании, хранение данных в разрозненных файлах и папках делает невозможным быстрое создание кода и проверку во время разработки. Требование регистрировать компоненты во время их создания создает потенциальный источник ошибок. По сути, Angular использует синтаксис, напоминающий HTML, обеспечивая совместимость во всех браузерах.

Отличие svelte заключается в том, что не помещает целиком все приложения для выполнения в браузере, как делают другие, а на каждый запрос запускает компилятор на сервере, что приводит к неэффективности данного фреймворка. У

него есть встроенная анимация, что отличает сильно его от аналогов, но при этом у него нет основных многих функций, например, валидатора.

Для написания клиентской части был выбран react. Её отличие заключается в том, что она описана как библиотека, а значит можно контролировать последовательность программы. С помощью функций можно запускать когда угодно нужную часть кода в любом файле программы, как для проверки и поиска ошибок, так и для полноценной работы программы.

Главная особенность react – компоненты. Благодаря компонентам можно возвращать html-элементы. При освоении данной библиотеки вряд ли возникнут трудности, так как есть поддержка и очень большой набор библиотек, которые можно использовать под любые функции. При разработке приложения можно выбирать когда и где будет интегрирована та или иная библиотека в коде, то есть приходится отвечать за поток приложения.

React был выбран для создания клиентской части из-за возможности серверного рендеринга и наличия большого количества готовых компонентов.

React, которую часто называют React.js, - это библиотека JavaScript, разработанная Facebook для создания интерактивных и отзывчивых пользовательских интерфейсов. Используя подход Angular, React разбивает приложение на отдельные компоненты. Каждый компонент может сохранять свое собственное состояние, а родительские компоненты имеют возможность передавать свое состояние дочерним компонентам. В свою очередь, эти дочерние компоненты могут передавать изменения обратно родительскому с помощью функций обратного вызова. Компоненты React обычно создаются с использованием JSX (JavaScript XML) - расширения для JavaScript, которое позволяет разработчикам встраивать синтаксис HTML непосредственно в свой код, обеспечивая плавную интеграцию разметки и логики.

При выборе библиотеки для клиентской части важно также решить, какую графическую библиотеку использовать. Для React существует множество решений, включая Ant Design, Mantine и Material UI.

					ПГУ 1.090501.10.001 ПЗ	Лист
						21
Изм.	Лист	№ докум.	Подпись	Дата		

Ant Design — удобная библиотека компонентов пользовательского интерфейса, разработанная Alibaba Group. Компоненты обладают множеством настроек и опций, что делает их адаптивными. Преимущества включают в себя богатый выбор компонентов, простоту использования и постоянное развитие.

Преимущества Ant Design Library:

- обширная библиотека компонентов: Ant Design может похвастаться широким спектром компонентов, включающих как базовые элементы, так и усовершенствованные для создания сложных и эстетически приятных пользовательских интерфейсов;
- удобство в использовании: библиотека доступна разработчикам с различным уровнем квалификации благодаря интуитивно понятному API и хорошо структурированной документации;
- непрерывная эволюция: Ant Design выигрывает от активного сообщества разработчиков, которое активно участвует в развитии и обслуживании библиотеки.

Недостатки Ant Design Library:

- большой размер: обилие компонентов приводит к увеличению размера библиотеки, что потенциально может повлиять на производительность;
- специфика платформы: дизайн Ant разработан специально для React, что может ограничить его совместимость с другими платформами.

Плюсы библиотеки Mantine:

- ясность версий: Mantine четко указывает на степень готовности версий, предупреждая пользователей о возможных ошибках в недостаточно разработанных версиях;
- открытый исходный код: исходный код библиотеки находится в открытом доступе для ознакомления сообщества;
- универсальность: Mantine разработан таким образом, чтобы быть совместимым с различными платформами React;

					ПГУ 1.090501.10.001 ПЗ	Лист
						22
Изм.	Лист	№ докум.	Подпись	Дата		

– регулярные обновления: библиотека часто обновляется и активно представлена на GitHub.

Недостатки библиотеки Mantine:

– процесс обучения: новичкам может быть сложно освоить Mantine;
– проблемы с интеграцией: иногда могут возникать трудности при интеграции Mantine с другими библиотеками React.

Плюсы Material UI:

– готовые компоненты: Material UI предлагает набор компонентов, созданных в соответствии с рекомендациями Google по дизайну материалов, чтобы обеспечить единообразный внешний вид на разных платформах;

– подход, ориентированный на мобильные устройства: изначально разработанный для мобильных устройств, Material UI был эффективно адаптирован для использования на настольных компьютерах;

– простота тематизации: настройка темы приложения включает в себя простое изменение цвета в файле JavaScript;

– строгая документация и лаконичность: Material UI выбран для создания пользовательского интерфейса из-за его хорошо документированной природы и лаконичности, которые он предлагает, в соответствии с потребностями разрабатываемой веб-платформы.

1.2.4 Серверная часть

Инструменты серверной разработки были выбраны с акцентом на производительность, масштабируемость и возможности быстрой разработки. (таблица 1).

Ext JS - это фреймворк, используемый для разработки приложений React. Примечательно, что Ext JS поддерживает рендеринг на стороне сервера, позволяя генерировать исходные данные на сервере. Этот механизм обеспечивает быстрый запуск страницы браузера и увеличивает производительность.

Таблица 1 – Инструменты разработки

Наименование инструмента	Причина выбора
NodeJS	Фреймворк, позволяющий писать многопоточные серверные приложения
NextJS	Фреймворк, который позволяет генерировать структуру страницы на стороне клиента со стороны сервера
Nodemon	Утилита, позволяющая повысить производительность разработки, путём избавления от перезапуска сервера при каждом изменении
Express	Фреймворк, позволяющий создавать веб часть для nodeJs

Node.js это платформа для асинхронного выполнения JavaScript, управляемая событиями. Она использует движок JavaScript Google V8, соответствующий стандарту ES2015. Оснащенный множеством модулей, Node.js облегчает внедрение веб-приложений и поддерживает сетевые протоколы, такие как HTTP и IP. Его основная сила заключается в однопоточной модели программирования, которая соответствует потоковой модели, используемой в браузерах. Такой подход возможен благодаря трем фундаментальным концепциям Node.js события, асинхронные API и неблокирующий ввод-вывод.

Server-Side Rendering – это способность приложения вносить свой вклад, отображая веб-страницу на сервере, а не отображая ее в браузере. Для того, чтобы доставить клиенту готовую страницу, необходим рендеринг на стороне сервера. Это не значит, что код не будет выполняться на клиенте, он все остаётся и функционирует, но первая команда на сервере происходит. Чтобы предотвратить потерю данных, браузер должен проверить собранную объектную модель

документа (DOM), чтобы убедиться в ее соответствии с отображением клиента. При обнаружении несоответствий браузер повторно отобразит страницу.

Nodemon служит автоматизированным инструментом, который запускает перезапуск приложения при обнаружении изменений файлов в каталоге.

Express является базовой платформой для разработки веб-компонентов Node.js. Известный своей быстротой и простотой, Express упрощает разработку веб-приложений и API. Его полезность распространяется на создание маршрутов, обработку запросов и ответов, управление сессиями и различные другие важные функции.

1.3 Проектирование информационной системы

1.3.1 Диаграмма вариантов использования

Для решения выявленной проблемы планируется создать цифровую платформу для размещения базы данных учебных материалов и разработки персонализированных онлайн-курсов, адаптированных к индивидуальным потребностям пользователей. Эта инициатива направлена на приведение в соответствие с современными стандартами дистанционного обучения.

Принимая во внимание сильные и слабые стороны вышеупомянутых приложений, были определены конкретные требования к предполагаемой платформе.

На рисунке 6 изображена схема вариантов использования, которая иллюстрирует различные взаимодействия с системой.

Полная функциональность системы представлена на диаграмме вариантов использования, которая позволяет наглядно представить пользователей приложения и связанные с ними функции. Ключевые компоненты диаграммы охватывают участников и варианты использования, с которыми они взаимодействуют. Подробная разбивка диаграммы вариантов использования приведена в таблице 2.

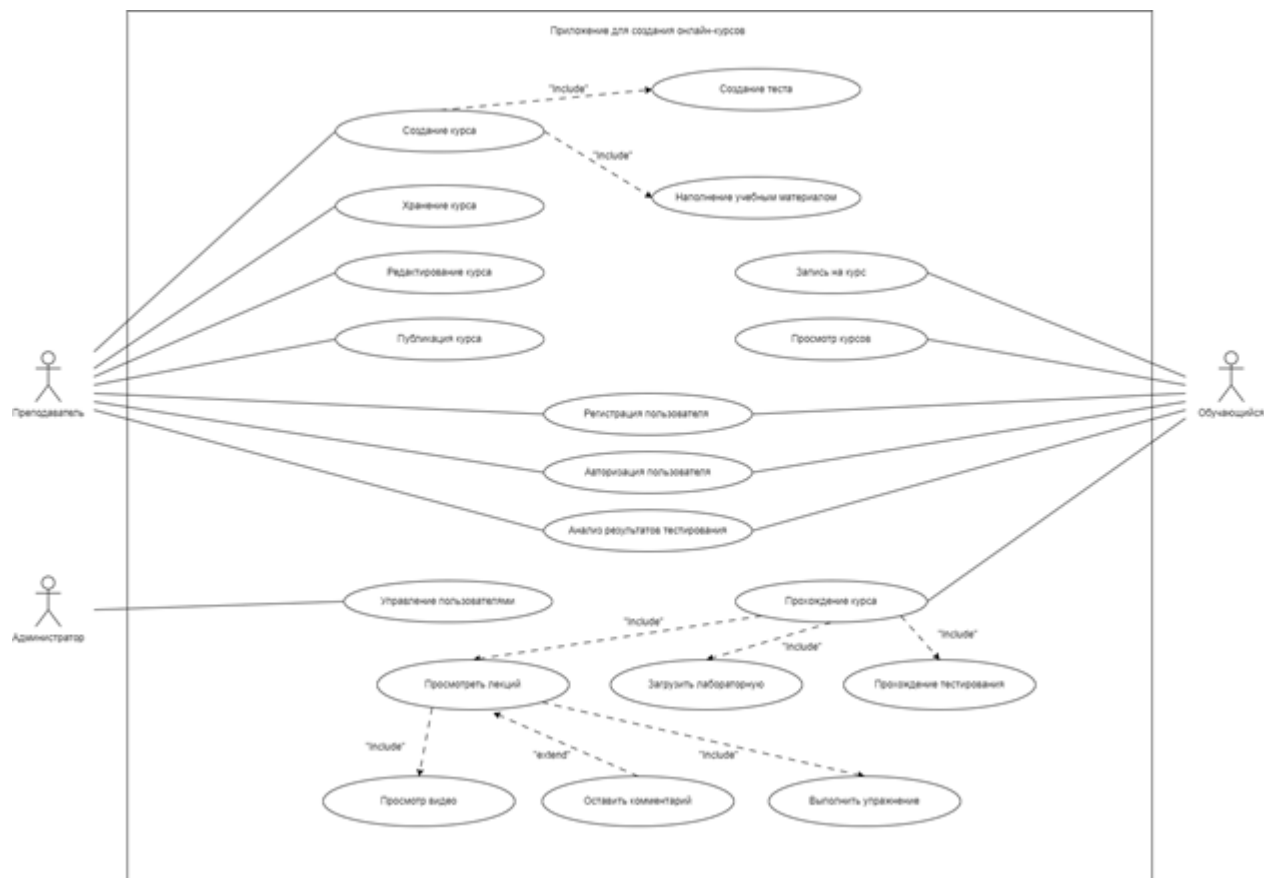


Рисунок 6 – Схема вариантов использования

Таблица 2 – Описание схемы вариантов использования

Актёр	Действие	Обозначение
Преподаватель	Авторизация	После ввода имени пользователя и пароля пользователю предоставляется доступ к системе, если предоставленные учетные данные подтверждены как точные.
	Регистрация	Пользователь регистрируется в системе, указывая свои имя, адрес электронной почты и пароль.
	Создание курса	Преподаватель создает новый курс, указывая его название, описание, категорию, тему.
	Создание теста	Преподаватель выбирает к какому из курсов будет создан тест. На основе

Актёр	Действие	Обозначение
		выбранного курса создаются вопросы и выбирается метод ответа на них, в виде выбора из заданных ответов или индивидуальный ответ.
	Наполнение учебным материалом	Преподаватель добавляет в курс лекции, видео, упражнения и тесты.
	Хранение курса	Перед публикацией преподаватель имеет возможность сохранить курс на сервере, что позволяет постепенно добавлять материалы с течением времени.
	Редактирование курса	Преподаватель имеет возможность в любой момент времени изменить курс, сделать правки.
	Публикация курса	Преподаватель по завершении создания курса может опубликовать свой курс.
	Анализ результатов тестирования	По завершении учеником тестирования, преподаватель может открыть результаты теста каждого обучающегося и оценить результат.
Обучающийся	Авторизация	Пользователь входит в систему, используя свои имя пользователя и пароль. В случае правильного ввода данных, выполняется вход.
	Регистрация	Пользователь регистрируется в системе, указывая свои имя, адрес электронной почты и пароль
	Запись на курс	Обучающийся записывается на курс, выбирая его из списка доступных курсов.
	Просмотр курсов	Обучающийся просматривает список доступных курсов, фильтруя их по категории, теме.

Актёр	Действие	Обозначение
Обучающийся	Прохождение курса	Обучающийся открывает курс, изучает информацию.
	Просмотр лекций	Обучающийся читает информацию из лекции.
	Просмотр видео	Обучающийся смотрит видео из курса.
	Оставить комментарий	Обучающийся имеет возможность оставить пожелания или задать вопрос в соответствующем окне после лекции.
	Выполнить упражнение	Обучающийся выполняет упражнение по пройденной лекции.
	Загрузить лабораторную	Обучающийся загружает файл, с выполненным заданием, указанным в лекции.
	Прохождение тестирования	Обучающийся по завершении курса проходит тестирование с целью оценки усвоения материала.
Администратор	Управление пользователями	Администратор создает учетные записи пользователей для тех, кто нуждается в доступе к системе и удаляет учетные записи пользователей, которые больше не используются. Также назначает права доступа и управление группами пользователей.

Преподаватель, помимо регистрации и авторизации в приложении, имеет возможность оформить курс, путём добавления в него различной информации, текста и медиафайлов. У преподавателя есть функция редактирования курса в любой момент времени.

Пользователь имеет не только стандартные функции регистрации и авторизации в информационной системе, но и возможность выбора интересующего его курса и записи на этот курс. Причем вариант использования «прохождения курса» включает в себя такие функции, как просмотр лекций, загрузка лабораторных по теме, а также прохождение тестирования после изучения курса. Варианты использования «просмотр видео», «оставить комментарий», «выполнить упражнение» включаются в «просмотр лекций».

Администратору доступна возможность управления пользователями информационной системы.

1.3.2 Архитектура приложения

На рисунке 7 представлена архитектура приложения для создания онлайн-курсов для обучения специалистов.

В процессе работы с программой пользователь обращается к элементам графического интерфейса, выполняя определенные действия на странице приложения. Взаимодействие происходит через компоненты, представленные в окнах приложения. Например, при заполнении формы регистрации пользователь вводит информацию в поля и затем активирует кнопку для завершения процесса.

После нажатия кнопки регистрации компонент Controller проводит проверку введенных данных. В случае соответствия требованиям, данные передаются для дальнейшей обработки Model-компоненту. Его задача - обработать информацию согласно установленной логике. После успешной обработки Model уведомляет View об успешной регистрации. В случае неудачной валидации данных Controller сообщит об этом View, который информирует пользователя об ошибке в заполнении формы.

Диаграмма развертывания приложения система состоит из трех ключевых компонентов (рисунок 8): клиент, база данных и сервер.

Система управления базами данных (СУБД) состоит из комплекта программных инструментов, которые позволяют управлять, создавать и администрировать базы данных. СУБД предлагает структурированный подход к организации, контролю и администрированию баз данных, способствуя

эффективному управлению данными. На рисунке 8 представлена схема развертывания приложения, которая наглядно демонстрирует конфигурацию и взаимодействие аппаратного и программного обеспечения системы.

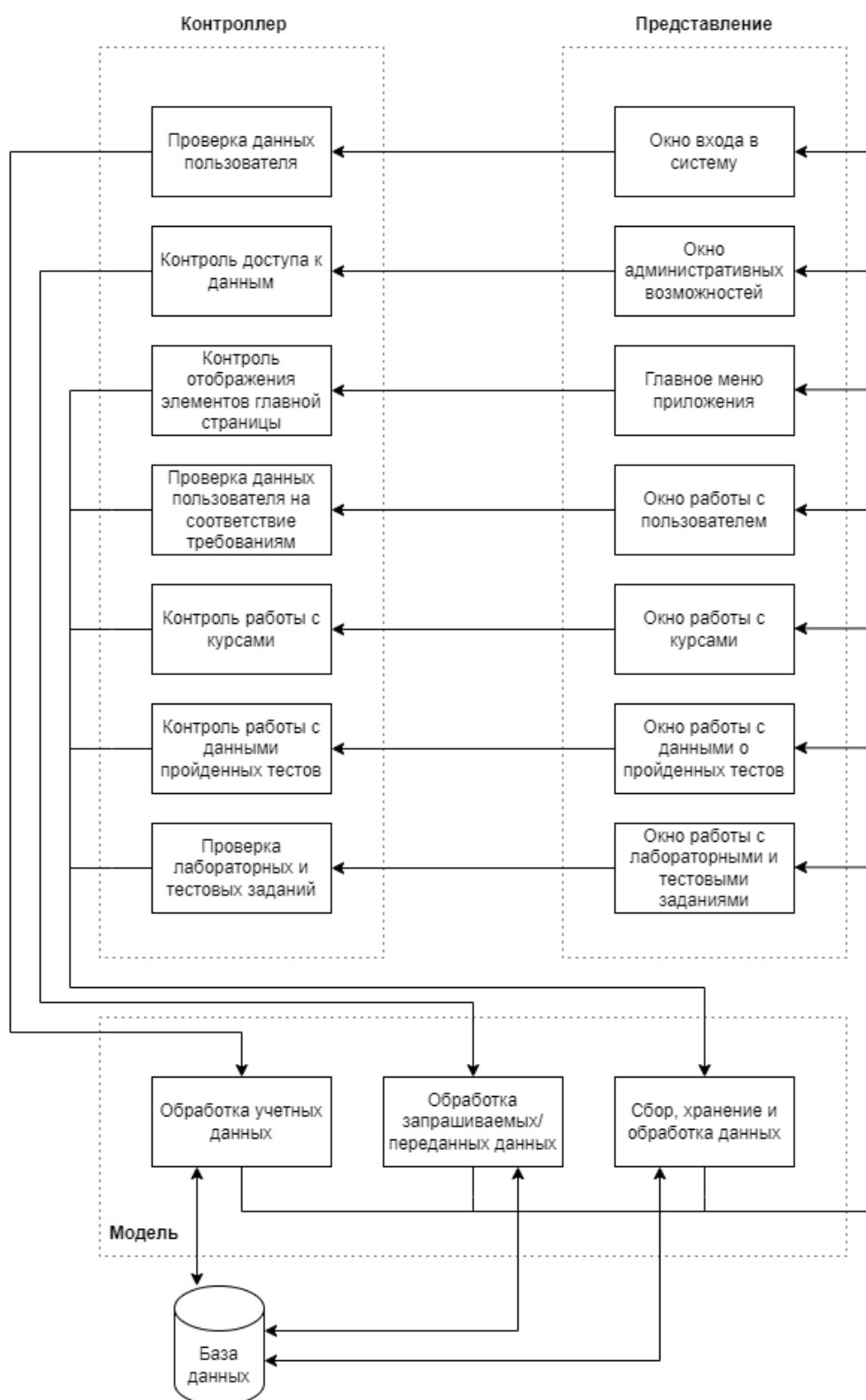


Рисунок 7 – Архитектура приложения для создания онлайн-курсов



Рисунок 8 – Схема развертывания приложения

На рисунке 9 показана схема последовательности действий, детализирующая порядок взаимодействий между объектами и процессами внутри системы с течением времени.

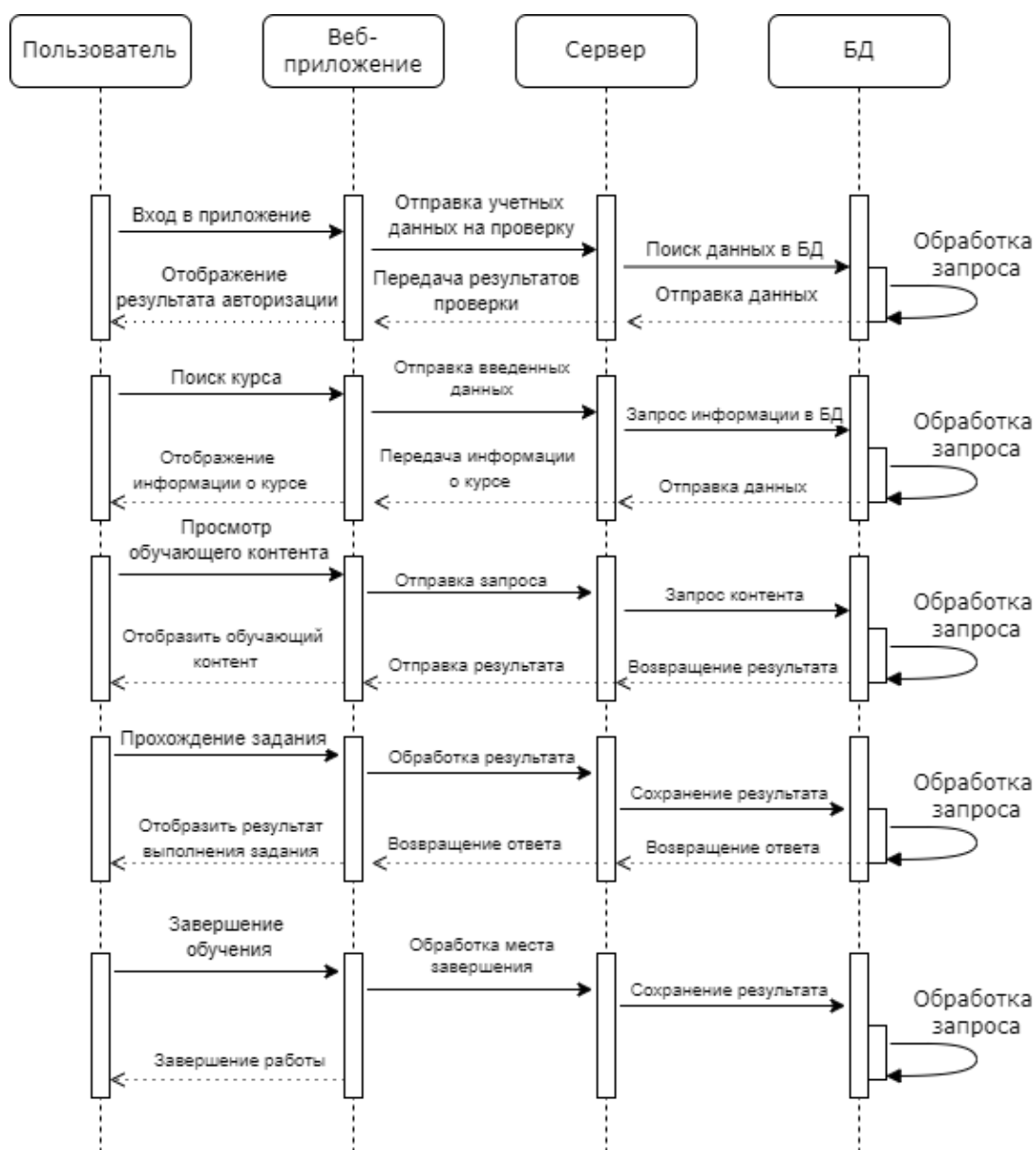


Рисунок 9 – Схема последовательности действий

Диаграмма класса проиллюстрирована на рисунке 10.

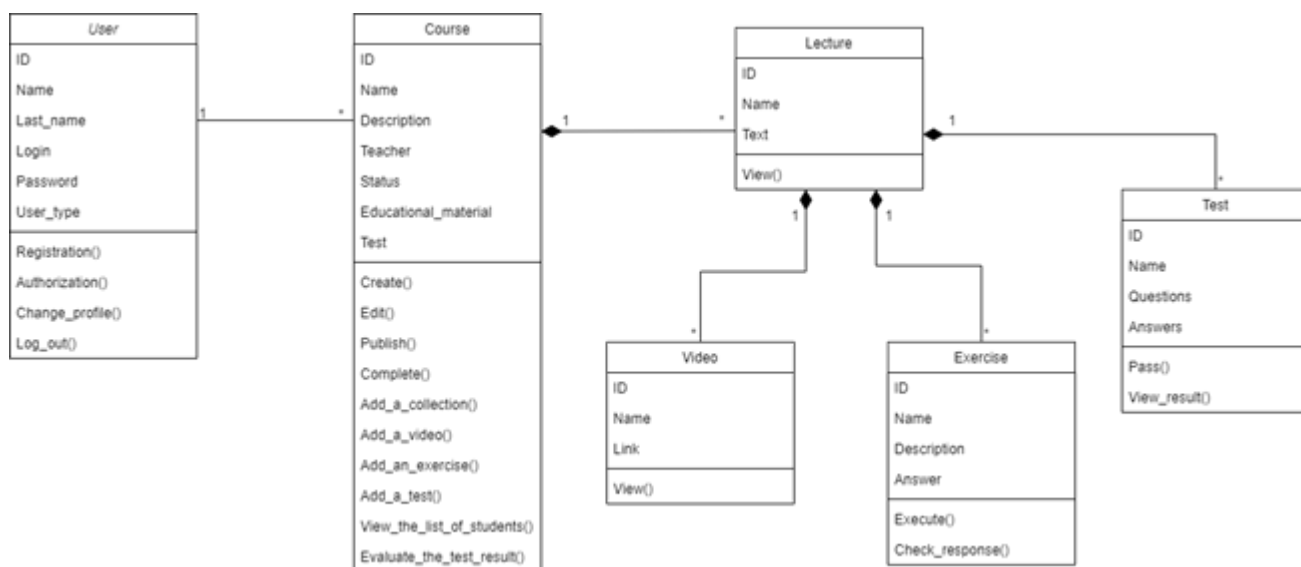


Рисунок 10 – Диаграмма классов

Класс "User" представляет пользователя системы онлайн-курсов и содержит информацию о нем, включая его имя, фамилию, логин, пароль и тип пользователя (преподаватель, студент или администратор). Класс "User" включает следующие атрибуты:

- уникальный идентификатор пользователя;
- имя пользователя;
- фамилия пользователя;
- логин пользователя (username);
- пароль пользователя;
- тип пользователя (например, преподаватель, студент, администратор).

В дополнение к атрибутам класс "User" также содержит методы для выполнения различных действий в системе: регистрация в системе, позволяет новому пользователю зарегистрироваться и создать учетную запись в системе онлайн-курсов; вход в систему, позволяет пользователю пройти аутентификацию и получить доступ к системе, используя свои учетные данные; изменить свой профиль, предоставляет пользователю функциональные возможности для изменения информации в своем профиле, такие как обновление имени

пользователя, логина или пароля, выход из системы, позволяет пользователю безопасно выйти из системы, завершив текущую сессию.

Класс "User" служит основой для создания пользовательских объектов в системе онлайн-курсов и управления ими, обеспечивая необходимую функциональность для управления пользователями и взаимодействия с системой.

Класс "User" также включает в себя такие функциональные возможности, как регистрация в системе, вход в систему, изменение своего профиля и выход из системы. Между тем, категория "Course" представляет собой курс, доступный в рамках онлайн-курса, и содержит подробную информацию о курсе, включая его название, описание, преподавателя, статус (черновик, опубликован, завершен), учебные материалы (лекции, видео, упражнения) и оценки.

Класс "Course" наделена следующими атрибутами:

- отдельным идентификатором курса;
- названием курса;
- описанием курса;
- преподаватель, ответственный за создание курса;
- статус курса (черновой, опубликованный, завершенный);
- учебные материалы, связанные с курсом;
- оценки, предназначенные для оценки понимания учащимися.

Кроме того, класс "Course" также включает в себя такие функции, как добавление упражнений к курсу, создание нового курса, интеграция тестов в курс, просмотр списка участников студентов обучающихся на этом курсе, редактирование сведений о курсе, публикация курса для студентов, оценка результатов тестирования студентов, добавление лекции в курс.

Класс "Лекция" описывает отдельную лекцию, которая является частью более широкой структуры курса. Оно включает в себя ключевые детали, относящиеся к лекции, включая ее название и сопровождающий текст.

В классе "Лекция" включены следующие атрибуты:

- уникальный идентификатор, который отличает лекцию;

- заголовок лекции;
- полный текст, который включает в себя лекция.

Кроме того, конструкция "Лекция" оснащена оперативными методами, в частности: функция для отображения текста, содержащего лекцию.

Классы «Test», «Video», «Exercise» являются неотъемлемыми компонентами курса, каждый из которых содержит конкретную информацию, относящуюся к их соответствующим областям. Класс "Видео" содержит такие сведения, как название и гиперссылка на видеоконтент, класс "Упражнение" содержит название, описание и правильный ответ на упражнение, в то время как класс "Тест" содержит название, набор вопросов и соответствующие ответы на них.

Атрибуты класса "Test" включают:

- уникальный идентификатор теста;
- заданное имя теста;
- набор запросов, составляющих тест;
- набор правильных ответов на вышеупомянутые вопросы.

В классе "Test" представлены следующие методы:

- процедура для проведения и завершения теста.;
- операция для просмотра результатов теста.

Класс "Video" характеризуется такими атрибутами, как:

- уникальный идентификатор видео;
- название, присвоенное видео;
- URL-адрес, ведущий к видеоконтенту.

Методы в классе "Video" включают в себя:

- функцию для просмотра видео.

Наконец, класс "Упражнение" определяется следующими атрибутами:

- уникальный идентификатор упражнения;
- название упражнения;
- подробное объяснение упражнения;
- точное решение упражнения.

И класс "Упражнение" предлагает следующие методы:

- действие для выполнения упражнения;
- процесс проверки ответа, предоставленного в упражнении.

На рисунке 11 продемонстрирована диаграмма компонентов для приложения по созданию онлайн-курсов для обучения специалистов.

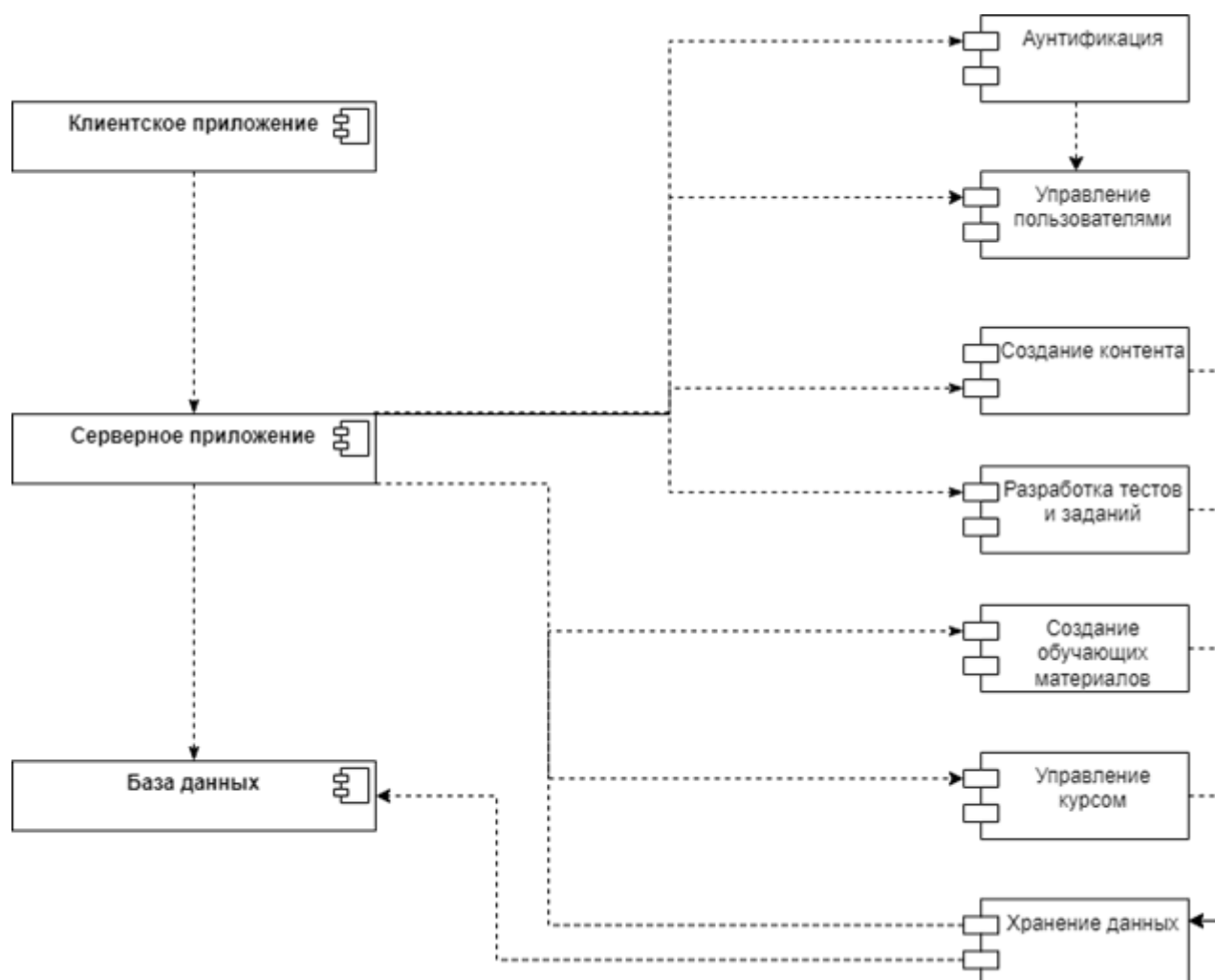


Рисунок 11 – Диаграмма компонент

Клиентское приложение может быть реализовано в различных формах, таких как веб-приложение, мобильное приложение или настольное приложение. Его цель - предоставить пользователям доступный интерфейс, облегчающий такие действия, как регистрация, вход в систему, просмотр курса, посещение лекций, выполнение упражнений, сдача тестов и просмотр результатов[3].

Серверу приложений поручено обрабатывать запросы клиентских приложений, выполнять основную логику системы и взаимодействовать с базой данных. Крайне важно, чтобы сервер был построен на надежной веб-платформе и был масштабируемым, чтобы соответствовать растущей базе пользователей[4].

База данных является хранилищем всех связанных с системой данных, включая профили пользователей, сведения о курсе, содержание лекций, видеоматериалы, упражнения, оценки и результаты тестов. Она должна обладать надежностью и масштабируемостью для управления расширяющимся набором данных.

Для управления этими данными необходимо создать структуру базы данных с конкретными таблицами.

Модуль аутентификации имеет решающее значение для проверки подлинности пользователя и предоставления разрешений. Он должен поддерживать множество механизмов аутентификации, включая имя пользователя и пароль, входы в социальные сети и двухфакторную аутентификацию.

Компоненты и артефакты модуля аутентификации включают:

- контроллер аутентификации: управляет обработкой запросов на вход в систему, регистрацию и сброс пароля;
- хранилище токенов: отслеживает данные сеанса пользователя;
- служба аутентификации: проверяет точность учетных данных пользователя;
- формы входа и регистрации: интерфейсы для ввода учетных данных пользователя;
- API аутентификации и авторизации: интерфейсы для взаимодействия с системой в отношении процессов аутентификации и авторизации;
- модели пользователей и сеансов: структуры данных, представляющие пользователей и информацию об их сеансах.

На рисунке 12 продемонстрирована диаграмма компонентов для модуля аутентификации.

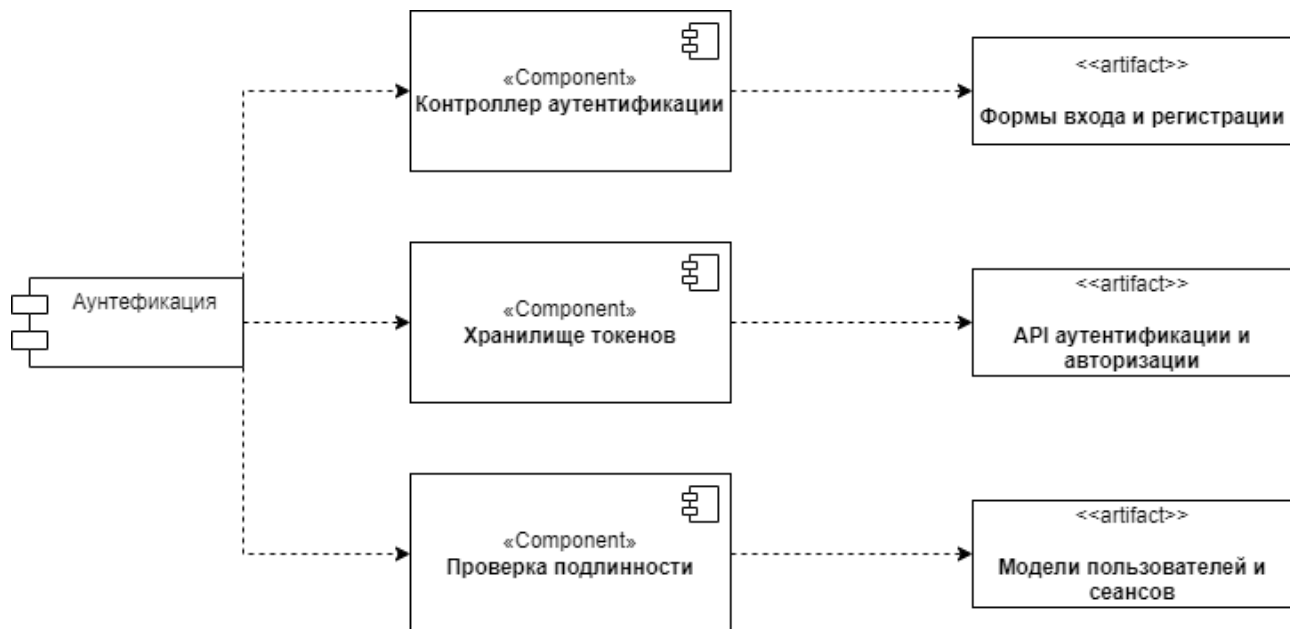


Рисунок 12 – Архитектура модуля аутентификации

Модуль управления пользователями разработан таким образом, чтобы служить основой для хранения информации о пользователях, контроля за операциями, связанными с пользователями, и предоставления API, адаптированного для администраторов приложений для эффективного управления пользователями.

Компоненты и артефакты модуля управления пользователями включают:

- менеджер профилей: этот компонент отвечает за работу с профилями пользователей, включая хранение пользовательской информации, ролей и разрешений;
- менеджер групп: отвечает за администрирование групп пользователей, включая контроль за членством в этих группах;
- служба уведомлений: система, предназначенная для рассылки уведомлений пользователям, улучшения коммуникации и вовлеченности;
- страницы профиля пользователя: пользовательские интерфейсы, позволяющие просматривать и редактировать отдельные профили пользователей;
- страницы управления группами: интерфейсы, предназначенные для администрирования групп пользователей и их членства;

– шаблоны уведомлений: predetermined formats, used by the notification service to ensure sequential and effective communication with users.

Together these elements work in tandem, ensuring a reliable system of user management, which is functional and convenient for the user, satisfying administrative needs of the application.

In Figure 13, a component diagram for the user management module is demonstrated.

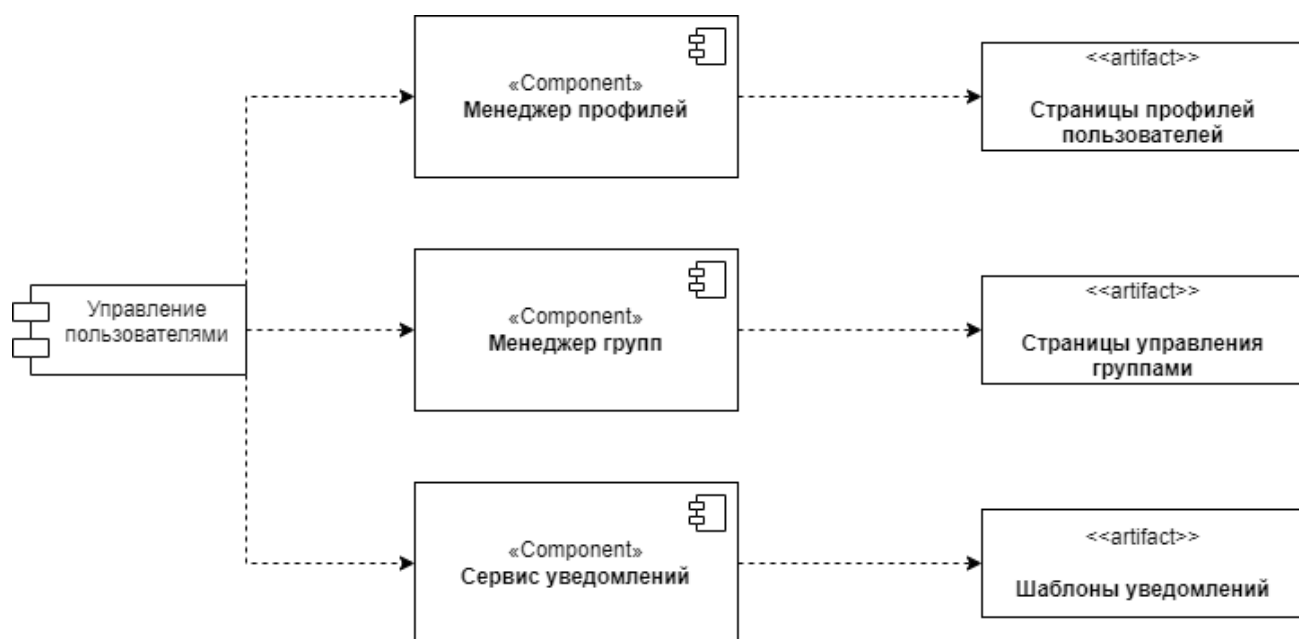


Рисунок 13 – Схема компонентов и артефактов модуля управления пользователями

The content creation module is designed to facilitate the creation of educational materials, allowing the use of different types of carriers and organizing content into a single course structure.

Key components and artifacts of the content creation module include:

– content editor: this instrument allows users to create and combine text, images, video and other multimedia elements into rich educational content;

- структура контента: предлагает структуру для организации контента в логические разделы и уроки, что облегчает пользователям навигацию и обучение;
- хранилище контента: хранилище для безопасного хранения всего созданного контента, обеспечивающее его доступность для дальнейшего использования и ссылок;
- редактор: пользовательский интерфейс, предоставляющий средства для создания и редактирования контента;
- инструменты форматирования: набор утилит в редакторе, которые позволяют пользователям форматировать и стилизовать контент для улучшения читаемости и вовлеченности;
- дерево структуры контента: визуальное представление или карта того, как организован контент, позволяющее создателям видеть иерархию разделов и уроков и вносить необходимые коррективы.

Вместе эти компоненты образуют комплексную систему, которая позволяет преподавателям эффективно создавать образовательный контент, управлять им и структурировать его, создавая динамичную и интерактивную среду обучения.

На рисунке 14 продемонстрирована диаграмма компонентов для модуля создания учебных материалов.

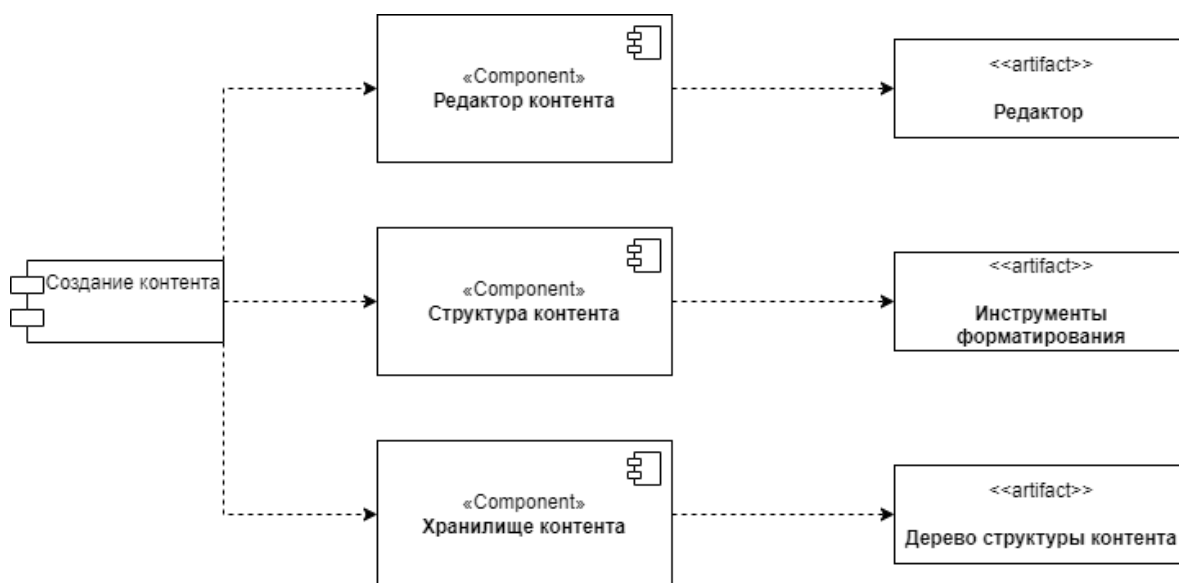


Рисунок 14 – Схема компонентов и артефактов модуля создания контента

Модуль разработки тестов и заданий предназначен для облегчения создания различных форматов тестов и практических заданий, а также для внедрения автоматизированной системы оценки.

Модуль разработки тестов и заданий оснащен несколькими ключевыми компонентами, предназначенными для упрощения создания оценок и заданий, которые включают в себя: функции создания тестов, что позволяет пользователям создавать различные типы тестов, в том числе вопросы с множественным выбором, правильные/ ложные вопросы и открытые запросы; инструменты разработки заданий, эти инструменты позволяют создавать практические задания, соответствующие конкретным целям обучения; система оценивания, включен автоматизированный механизм выставления оценок за тесты и задания, который не только упрощает процесс оценивания, но и оперативно предоставляет обратную связь пользователям; интерфейсы для создания тестов, эти удобные интерфейсы обеспечивают гибкость при разработке и персонализации тестов; интерфейсы создания заданий, специально разработанные для генерации и настройки практических заданий, эти интерфейсы обеспечивают преподавателям простоту и удобство в использовании; отчеты об оценке, эти отчеты генерируют выходные данные, которые демонстрируют результаты и аналитическую информацию, полученную от системы оценки, предлагая ценную обратную связь и показатели эффективности.

Объединяя эти элементы, модуль разработки тестов и заданий призван предоставить преподавателям надежную платформу для создания, сдачи и оценки экзаменов и заданий, тем самым способствуя созданию эффективной среды обучения.

На рисунке 15 продемонстрирована диаграмма компонентов для модуля разработки тестов и заданий. Этот модуль играет ключевую роль при создании курса. Благодаря модулю есть возможность создавать задания и внедрять в различные уроки.

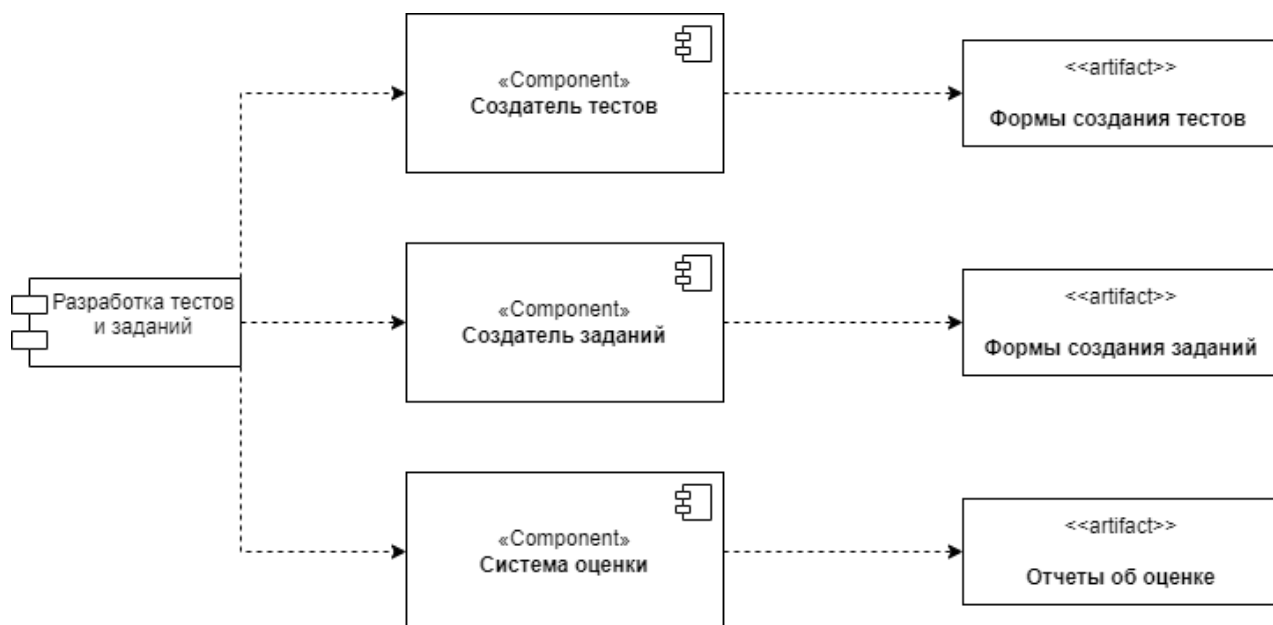


Рисунок 15 – Схема компонентов и артефактов модуля разработки тестов и заданий

Модуль создания учебных материалов специально разработан для облегчения разработки увлекательных интерактивных презентаций, которые могут включать различные мультимедийные элементы, такие как текст, изображения и видео. Кроме того, он предлагает API для упрощения доступа к этим учебным материалам через другие прикладные модули.

Ключевые компоненты и возможности модуля создания учебных материалов включают редактор презентаций, позволяющий пользователям создавать и персонализировать презентации, интегрируя различные типы контента для создания увлекательного процесса обучения. Медиабiblioteca служит хранилищем мультимедийных ресурсов, таких как аудио- и видеоматериалы, для улучшения качества презентаций и другого образовательного контента. Функция создания викторин позволяет создавать интерактивные тесты и опросы для оценки понимания учащимися и сбора отзывов. Шаблоны презентаций, мультимедийные библиотеки и редактор викторин, интерфейс для разработки и редактирования тестов, также являются неотъемлемыми компонентами.

Этот модуль играет ключевую роль в создании динамичных учебных материалов, которые не только эффективно передают информацию, но и увлекают учащихся и способствуют достижению общих образовательных целей приложения.

На рисунке 16 продемонстрирована диаграмма компонентов для модуля создания обучающих материалов.

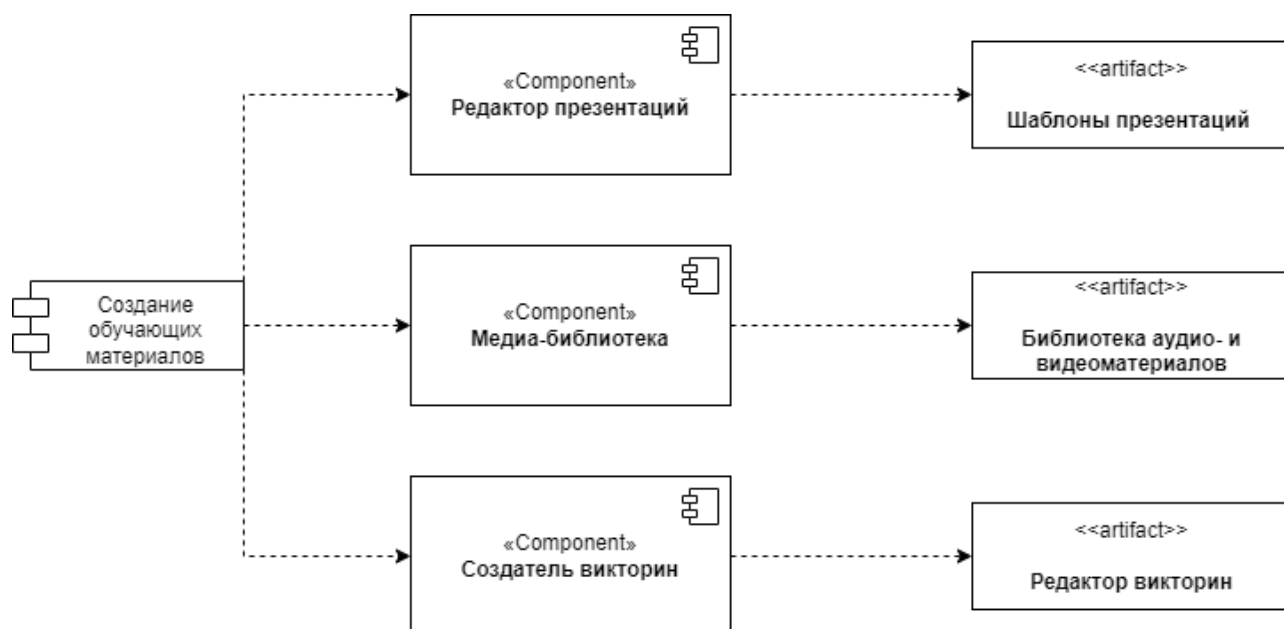


Рисунок 16 – Схема компонентов и артефактов модуля создания обучающих материалов

Модуль управления курсами предназначен для облегчения комплексного управления курсами, включая создание, редактирование, публикацию и контроль за содержанием курса. Кроме того, он позволяет преподавателям добавлять учебные материалы, управлять списками учащихся и отслеживать успеваемость учащихся.

Ключевые компоненты модуля управления курсами включают в себя:

- менеджер курсов: компонент, отвечающий за создание, настройку и управление онлайн-курсами, который предоставляет преподавателям инструменты для разработки и обеспечения эффективного процесса обучения;

– руководитель группы: отвечает за администрирование групп пользователей в рамках курсов, что позволяет организовывать группы студентов и управлять ими;

– система отслеживания успеваемости: система, которая отслеживает и фиксирует прогресс отдельных студентов на курсах, предоставляя ценную информацию об их учебном процессе;

– система аналитики: функция, которая генерирует статистику и аналитические данные об успеваемости и вовлеченности студентов в курс, предоставляя преподавателям ценные данные для оценки и совершенствования;

– страницы для создания и редактирования курсов: пользовательские интерфейсы, которые позволяют создавать, изменять содержимое курсов и управлять им, обеспечивая преподавателям удобство использования;

– страницы управления группами: интерфейсы, предназначенные для администрирования групп пользователей в рамках курсов, способствующие эффективному управлению и вовлечению студентов;

отчеты о проделанной работе: выходные данные, которые обобщают и

– отображают прогресс студентов в рамках курса, позволяя получить представление об их успеваемости и развитии;

– аналитические панели: Визуальные дисплеи и отчеты, в которых представлены статистические данные и аналитика по курсу, позволяющие преподавателям получать ценную информацию о вовлеченности и успеваемости учащихся.

Интегрируя эти компоненты, модуль управления курсами призван обеспечить преподавателям надежную платформу для создания, организации и мониторинга прогресса и результативности своих курсов, что в конечном итоге способствует созданию динамичной и эффективной среды обучения. Модуль хранения данных предназначен для хранения данных, информации о пользователях, курсах.

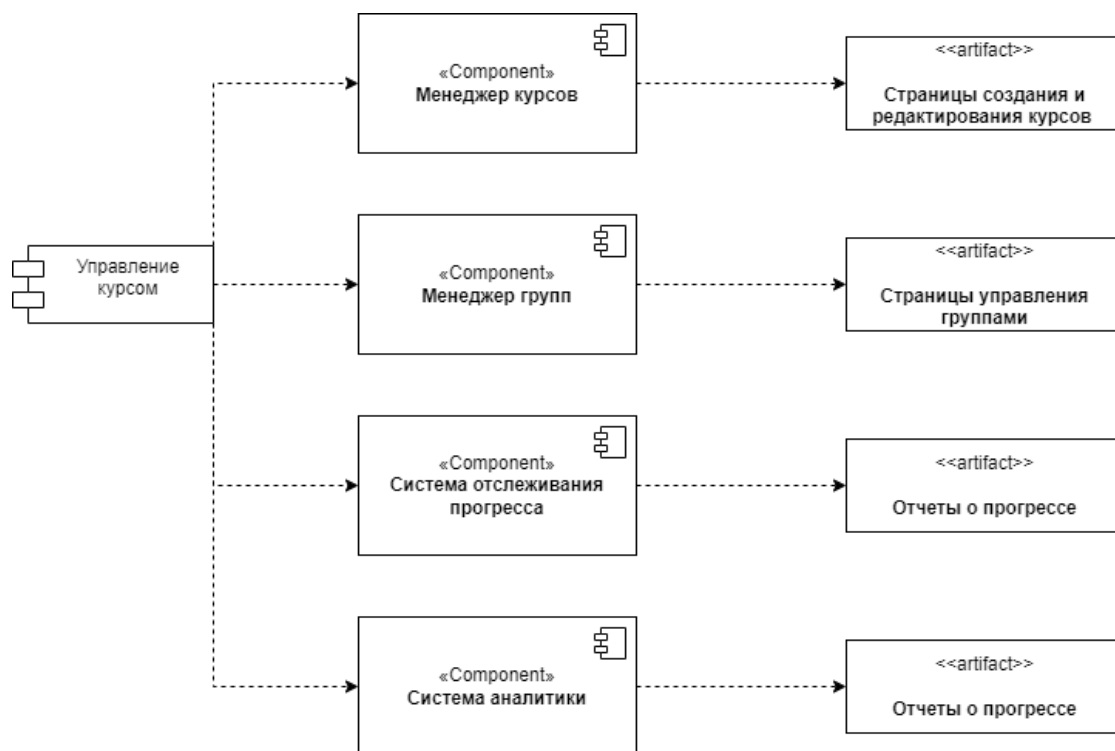


Рисунок 17 – Схема компонентов и артефактов модуля управлением курсом

1.3.3 Проектирование базы данных

Логическая схема структуры базы данных продемонстрирована на рисунке 18.

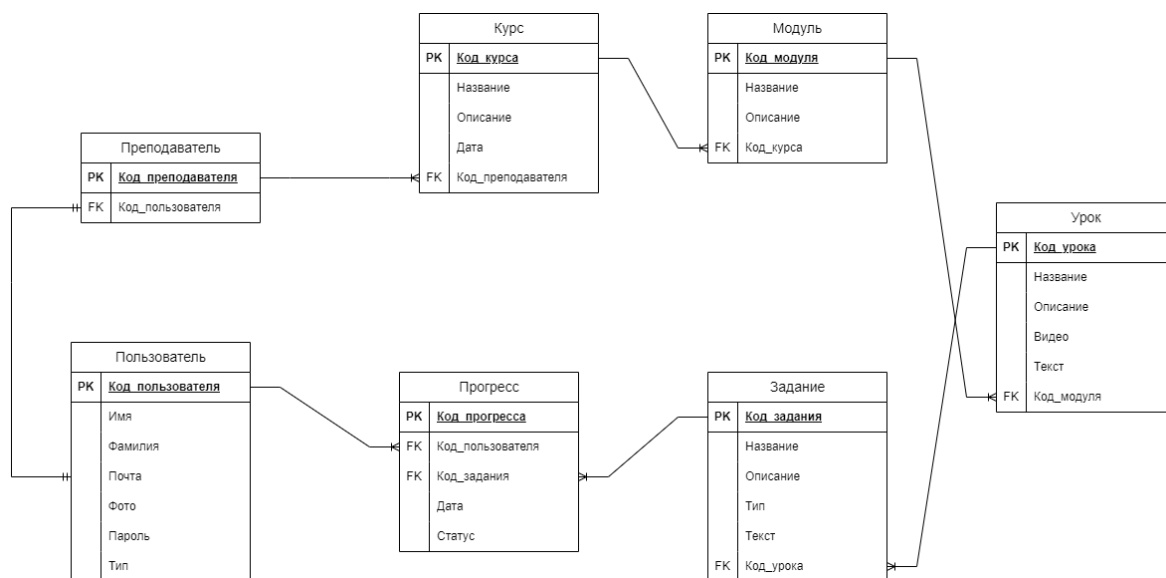


Рисунок 18 – Логическая модель структуры базы данных

На рисунке 19 продемонстрирована физическая модель структуры базы данных.

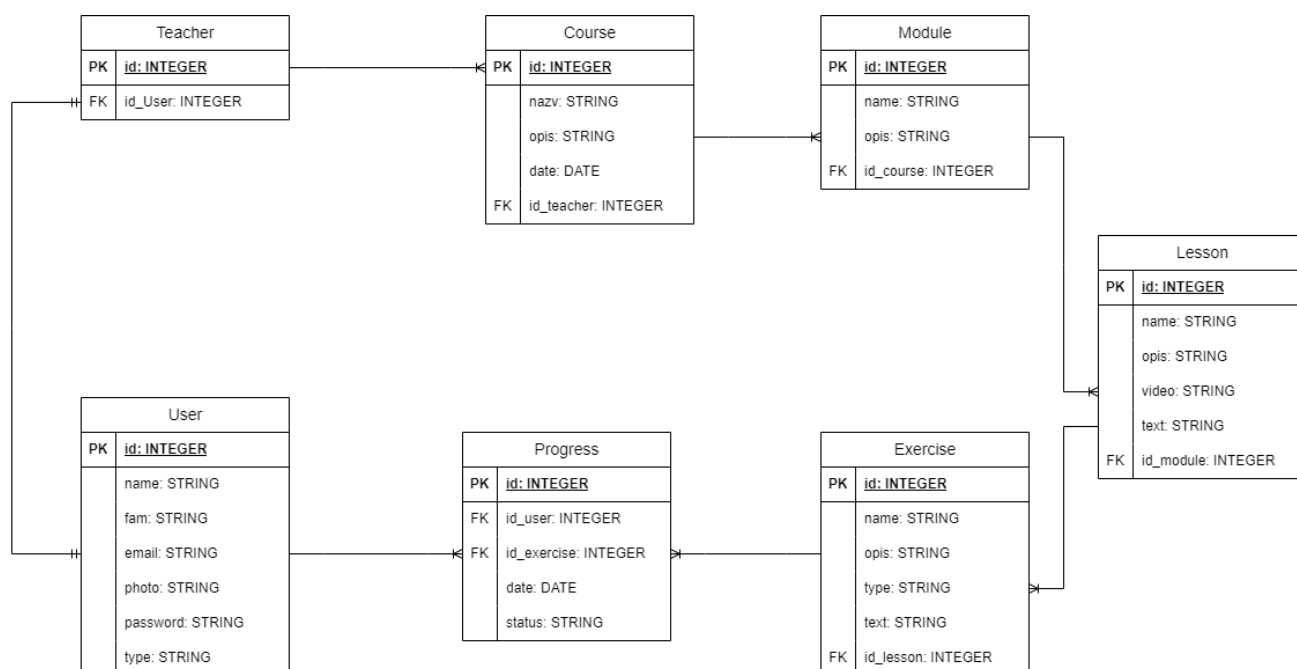


Рисунок 19 – Физическая модель структуры базы данных

Таблицы для хранения данных в базе данных представлены в таблицах 3-9.

Таблица 3 – Атрибуты таблицы users

Имя столбца	Тип данных	Описание	Роль атрибута	Пример
id	SERIAL	Идентификатор учетной записи	ПК	1
name	STRING	Имя		Daria
fam	STRING	Фамилия		Stepanova
email	STRING	Почта		stepadsh@mail.ru
photo	STRING	Фото		
passwd	STRING	Пароль от учетной записи		1234567890
type	STRING	Тип пользователя		Обучающийся

Таблица 4 – Атрибуты таблицы teacher

Имя столбца	Тип данных	Описание	Роль атрибута	Пример
id	INTEGER	Идентификатор преподавателя	РК	1
id_user	INTEGER	Идентификатор учетной записи пользователя	FK	1

Таблица 5 – Атрибуты таблицы course

Имя столбца	Тип данных	Описание	Роль атрибута	Пример
id	INTEGER	Идентификатор курса	РК	1
nazv	STRING	Название курса		Основы БД
opis	STRING	Описание курс		Курс для изучения базы данных
date	DATE	Дата загрузки		2024-01-01
id_teacher	INTEGER	Идентификатор преподавателя	FK	1

Таблица 6 – Атрибуты таблицы module

Имя столбца	Тип данных	Описание	Роль атрибута	Пример
id	INTEGER	Идентификатор модуля	РК	1
name	STRING	Название модуля		Работа с БД
opis	STRING	Описание модуля		Работа с БД
id_course	INTEGER	Идентификатор курса	FK	Петрович

Таблица 7 – Атрибуты таблицы lesson

Имя столбца	Тип данных	Описание	Роль атрибута	Пример
id	INTEGER	Идентификатор урока	PK	1
name	STRING	Название урока		Запросник
opis	STRING	Описание урока		Формирование запросов
video	STRING	Видео		
text	STRING	Текст лекции		Формирование запросов
id_module	INTEGER	Идентификатор модуля, куда входит урок	FK	1

Таблица 8 – Атрибуты таблицы exercise

Имя столбца	Тип данных	Описание	Роль атрибута	Пример
id	INTEGER	Идентификатор задания	PK	1
name	STRING	Название задания		Лабораторная №1
opis	STRING	Описание задания		Формирование регистрового отчёта
type	STRING	Тип задания		Лабораторное задание
text	STRING	Текст задания		2024-01-01 00:00
id_lesson	INTEGER	Идентификатор урока	FK	1

Таблица 9 – Атрибуты таблицы progress

Имя столбца	Тип данных	Описание	Роль атрибута	Пример
id	INTEGER	Идентификатор заказа	PK	1
id_user	INTEGER	Идентификатор пользователя, прошедшего задание	FK	1
id_exercise	INTEGER	Идентификатор выполненного задания	FK	1
date	DATE	Дата прохождения задания		2024-01-01 00:00
status	STRING	Статус задания		Выполнено

1.4 Разработка приложения

1.4.1 Разработка БД

На начальном этапе этой разработки мы внедрим класс DataTypes прямо из сердца библиотеки Sequelize. Этот шаг имеет первостепенное значение, поскольку он дает нам возможность с максимальной точностью определять типы данных, которые будут отображаться в каждом столбце таблиц нашей базы данных. После этого мы приступаем к тщательному процессу создания таблиц, каждый атрибут в которых определяется своим типом данных, и все они в конечном итоге должны находиться в пределах домена базы данных.

Возьмем, к примеру, таблицу, которая предназначена для хранения данных о зарегистрированных пользователях. Это здание будет построено на основе

пользовательской модели, при этом каждый атрибут в нем будет тщательно проверен, чтобы определить, следует ли учитывать возможность его недействительности. Идентификатор пользователя будет иметь функцию автоматического увеличения - умный механизм, который гарантирует, что каждой новой записи, появляющейся в таблице, автоматически присваивается особый идентификатор.

Если пойти дальше, то, когда кто-то хочет сконструировать таблицу, чтобы сохранить детали, относящиеся к модулям, он должен сначала создать модель для таблицы "модуль".

Создание таблицы "Учитель", хранилища информации для преподавателей, зарегистрированных в системе управления приложениями, также требует определения с помощью Sequelize. Аналогичным образом, Sequelize - это наш инструмент для создания таблицы успеваемости, которая будет отражать студентов, проходящих тесты и лабораторные работы в рамках системы управления образованием.

Более того, чтобы создать таблицу "Заданий", предназначенную для хранения данных.. А для создания таблицы "Уроки", средства передачи данных об уроках, которые были тщательно подготовлены, Sequelize, без сомнения, является подходящим инструментом.

Чтобы сплести сеть взаимосвязей между этими таблицами, позволяющую базе данных автономно генерировать столбцы, наделенные внешними ключами, необходимо задействовать возможности выбранной системы управления базами данных для определения этих сложных связей.

Наконец, чтобы взаимодействовать с базой данных через сервер, необходимо экспортировать всю эту область базы данных целиком, каждая таблица должна быть точно помечена, используя специализированные инструменты, разработанные для этой цели.

На рисунках 20-29 представлена реализация ббазы данных.

```
const sequelize = require(`../db`)
const {DataTypes} = require(`sequelize`)
```

Рисунок 20 – Инициализация библиотеки Sequelize

```
const User = sequelize.define(`user`, {
  id: {type: DataTypes.INTEGER, primaryKey: true, autoIncrement: true},
  name: {type: DataTypes.STRING, allowNull: false},
  fam: {type: DataTypes.STRING, allowNull: false},
  email: {type: DataTypes.STRING, unique: true, allowNull: false},
  photo: {type: DataTypes.STRING},
  password: {type: DataTypes.STRING, allowNull: false},
  type : {type: DataTypes.STRING, defaultValue: "USER", allowNull: false}
})
```

Рисунок 21 – Таблица User

```
const Course = sequelize.define(`course`, {
  id: {type: DataTypes.INTEGER, primaryKey: true, autoIncrement: true},
  name: {type: DataTypes.STRING, allowNull: false, unique: true},
  opis: {type: DataTypes.STRING, allowNull: false},
  data: {type: DataTypes.DATE},
  img: {type: DataTypes.STRING}
//id_teacher: {type: DataTypes.INTEGER, foreignKey: true}
})
```

Рисунок 22 – Таблица Course

```
const Module = sequelize.define(`module`, {
  id: {type: DataTypes.INTEGER, primaryKey: true, autoIncrement: true},
  name: {type: DataTypes.STRING, allowNull: false},
  opis: {type: DataTypes.STRING},
// id_course: {type: DataTypes.INTEGER, foreignKey: true}
})
```

Рисунок 23 – Таблица Module

```
const Teacher = sequelize.define(`teacher`, {
  id: {type: DataTypes.INTEGER, primaryKey: true, autoIncrement: true},
})
```

Рисунок 24 – Таблица Teacher

```
const Progress = sequelize.define(`progress`, {
  id: {type: DataTypes.INTEGER, primaryKey: true, autoIncrement: true},
  data: {type: DataTypes.DATE},
  status: {type: DataTypes.STRING}
})
```

Рисунок 25 – Таблица Progress

```
const Exercise = sequelize.define(`exercise`,{
  id: {type: DataTypes.INTEGER, primaryKey: true, autoIncrement: true},
  name: {type: DataTypes.STRING, allowNull: false},
  opis: {type: DataTypes.STRING},
  type: {type: DataTypes.STRING, defaultValue: "EXERCISE", allowNull: true},
  text: {type:DataTypes.STRING},
  //id_lesson: {type: DataTypes.INTEGER, foreignKey: true}
})
```

Рисунок 26 – Таблица Exercise

```
const Lesson = sequelize.define(`lesson`, {
  id: {type: DataTypes.INTEGER, primaryKey: true, autoIncrement: true},
  name: {type: DataTypes.STRING, allowNull: false},
  opis: {type: DataTypes.STRING},
  video: {type: DataTypes.STRING},
  text: {type:DataTypes.STRING},
  //id_module: {type: DataTypes.INTEGER, foreignKey: true}
})
```

Рисунок 27 – Таблица Lesson

```
module.exports = {
  User,
  Course,
  Module,
  Teacher,
  Progress,
  Exercise,
  Lesson
}
```

Рисунок 28 – Выгрузка таблиц

```

Teacher.hasMany(Course)
Course.belongsTo(Teacher)

Course.hasMany(Module)
Module.belongsTo(Course)

Module.hasMany(Lesson)
Lesson.belongsTo(Module)

Lesson.hasMany(Exercise)
Exercise.belongsTo(Lesson)

Exercise.hasMany(Progress)
Progress.belongsTo(Exercise)

User.hasMany(Progress)
Progress.belongsTo(User)

User.hasOne(Teacher)
Teacher.belongsTo(User)

```

Рисунок 29 – Создание связей между таблицами в реляционной базе данных

1.4.2 Описание программы

Каталог src/store хранилище скриптов для взаимодействия с экосистемой MODX и защиты цифровой информации. В этой области можно столкнуться со сложными алгоритмами, предназначенными для согласования состояния приложения, инициирования диалогов с API и управления ими, а также тщательного управления хранением данных для цифрового инструмента. Каталог src/page связующее звено основополагающих компонентов, каждый из которых служит воротами к множеству аспектов цифровой платформы. Эти ключевые элементы, вероятно, являются ориентирами для различных сегментов, проводя пользователей по интерактивному пути в приложении. Каталог src/components здесь представлена тщательно отобранная коллекция модульных компонентов, каждый из которых предназначен для интеграции в цифровую среду. Эти фрагменты кода являются квинтэссенцией пользовательского интерфейса, тщательно разработанного для использования на различных страницах и в различных сегментах приложений.

File.js это сборник функций, каждая из которых ожидает вызова. В его рамках находится функция "создание курса", которая имеет огромное значение. Ее задача: передавать образовательный контент через эфир на сервер, обновлять пользовательский интерфейс после успешной передачи и вызывать уведомления о несчастьях, если цифровые духи не согласятся.

В сложной сфере современных веб-приложений клиентская кодовая база является свидетельством тщательной организации и модульной архитектуры, умело разделяя такие обязанности, как управление состоянием, компоненты страниц и создание повторно используемых элементов пользовательского интерфейса. Использование вызовов API для взаимодействия с сервером и эффективного управления ошибками является примером стандартной методологии разработки на стороне клиента.

Функция обновления курса берет на себя важную ответственность за передачу обновленных данных курса на сервер посредством вызова API. После успешного выполнения обновления курса пользовательский интерфейс корректно изменяется, отражая недавно внесенные изменения. Она организует асинхронный запрос на сервер, отправляя обновленные данные курса через соответствующую конечную точку API.

Функция удаления курсов является важнейшим компонентом в области управления удалением данных о курсах. Она взаимодействует с сервером, отправляя официальный запрос API с просьбой удалить определенный курс. После успешного удаления указанного курса пользовательский интерфейс корректируется таким образом, чтобы отразить отсутствие удаленного объекта. Кроме того, он обладает способностью быстро находить ошибки, обеспечивая обратную связь и соответствующий ответ, если удаление источника становится невозможным.

Функция загрузки курса берет на себя обязанность по извлечению данных курса с сервера, использованию этих данных для заполнения пользовательского интерфейса или выполнения других соответствующих действий, одновременно

устраняя любые потенциальные ошибки, которые могут возникнуть в ходе этого сложного процесса.

Реализация функций по взаимодействию с курсом представлена на рисунках 30-33.

```
async function createCourse(courseData) {
  try {
    const response = await fetch('/api/courses', {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json'
      },
      body: JSON.stringify(courseData)
    });

    if (response.ok) {
      const newCourse = await response.json();
      console.log('Course created:', newCourse);
      return newCourse;
    } else {
      throw new Error('Failed to create course');
    }
  } catch (error) {
    console.error('Error creating course:', error);
  }
}
```

Рисунок 30 – Создание курса

```
async function updateCourse(courseId, courseData) {
  try {
    const response = await fetch(`/api/courses/${courseId}`, {
      method: 'PUT',
      headers: {
        'Content-Type': 'application/json'
      },
      body: JSON.stringify(courseData)
    });

    if (response.ok) {
      const updatedCourse = await response.json();
      console.log('Course updated:', updatedCourse);
      return updatedCourse;
    } else {
      throw new Error('Failed to update course');
    }
  } catch (error) {
    console.error('Error updating course:', error);
  }
}
```

Рисунок 31 – Обновление курса

```

async function deleteCourse(courseId) {
  try {
    const response = await fetch(`/api/courses/${courseId}`, {
      method: 'DELETE'
    });

    if (response.ok) {
      console.log('Course deleted:', courseId);
    } else {
      throw new Error('Failed to delete course');
    }
  } catch (error) {
    console.error('Error deleting course:', error);
  }
}

```

Рисунок 32 – Удаление курса

```

async function loadCourse(courseId) {
  try {
    const response = await fetch(`/api/courses/${courseId}`);

    if (response.ok) {
      const courseData = await response.json();
      console.log('Course loaded:', courseData);
      return courseData;
    } else {
      throw new Error('Failed to load course');
    }
  } catch (error) {
    console.error('Error loading course:', error);
  }
}

```

Рисунок 33 – Загрузка курса

Чтобы создать функцию для ввода данных о курсах, необходимо определить метод, позволяющий пользователям вводить информацию о курсах в систему. Функция записи на курс (function enrollCourse) отправляет запрос на регистрацию на сервер.

Реализация функции ввода данных представлена на рисунке 34.

```

async function enrollInCourse(courseId) {
  try {
    const response = await fetch(`/api/courses/${courseId}/enroll`, {
      method: 'POST'
    });

    if (response.ok) {
      console.log('User enrolled in course:', courseId);
    } else {
      throw new Error('Failed to enroll in course');
    }
  } catch (error) {
    console.error('Error enrolling in course:', error);
  }
}

```

Рисунок 34 – Запись на курс

Функция получения информации о курсе, часто называемая "получение сведений о курсе", предназначена для отправки запроса на сервер с целью получения конкретных данных о курсе. Эта функция обычно включает в себя отправку асинхронного запроса, такого как HTTP-запрос, на сервер, где хранится информация о курсе.

Реализация функции получения информации представлена на рисунке 35.

```

async function fetchCourseDetails(courseId) {
  try {
    const response = await fetch(`/api/courses/${courseId}`);

    if (response.ok) {
      const courseData = await response.json();
      console.log('Course details fetched:', courseData);
      return courseData;
    } else {
      throw new Error('Failed to fetch course details');
    }
  } catch (error) {
    console.error('Error fetching course details:', error);
  }
}

```

Рисунок 35 – Получение информации о курсе

Функция отправки задания (function submitAssignmen) принимает и отправляет данные на сервер.

Реализация функции отправки задания на сервер представлена на рисунке 36.

```
async function submitAssignment(courseId, assignmentId, submissionData) {
  try {
    const response = await fetch(`/api/courses/${courseId}/assignments/${assignmentId}/submit`, {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json'
      },
      body: JSON.stringify(submissionData)
    });

    if (response.ok) {
      console.log('Assignment submitted:', assignmentId);
    } else {
      throw new Error('Failed to submit assignment');
    }
  } catch (error) {
    console.error('Error submitting assignment:', error);
  }
}
```

Рисунок 36 – Отправка задания

Функция fetchCourseProgress отслеживает прогресс пользователя по прохождению и выполнению курса.

Реализация функции отслеживания прогресса представлена на рисунке 37.

```
async function fetchCourseProgress(courseId) {
  try {
    const response = await fetch(`/api/courses/${courseId}/progress`);

    if (response.ok) {
      const progressData = await response.json();
      console.log('Course progress fetched:', progressData);
      return progressData;
    } else {
      throw new Error('Failed to fetch course progress');
    }
  } catch (error) {
    console.error('Error fetching course progress:', error);
  }
}
```

Рисунок 37 – Отслеживание прогресса

Функция `submitCourseFeedback` предоставляет возможность оставить комментарий после прохождения курса.

Реализация функции отправки комментария представлена на рисунке 38.

```
async function submitCourseFeedback(courseId, feedbackData) {  
  try {  
    const response = await fetch(`/api/courses/${courseId}/feedback`, {  
      method: 'POST',  
      headers: {  
        'Content-Type': 'application/json'  
      },  
      body: JSON.stringify(feedbackData)  
    });  
  }  
}
```

Рисунок 38 – Комментарий

Файлы со стилями и настройками приведены в приложении Б (листинг программы).

Описание серверной части.

В папке `server` находятся все файлы относящийся непосредственно к серверу. Папка `server/index.js` – основной файл сервера, где устанавливается связь с сервером и передача всех данных необходимых для функционирования приложения и отображение информации. В файле происходит объявление всех фреймворков и функции, которые будут использоваться в работе сервер, а также происходит подключение к базе данных через порт. В случае ошибки подключения, выводится соответствующее сообщению.

На рисунке 39 представлен главный файл программы.

В папке `error` хранится файл `ApiError` необходимый для обработки ошибок, возникающих при работе сервера. Для каждого типа ошибки выводится соответствующее сообщение. Конструктор принимает статус код и сообщение, которое будет возвращаться на клиент. Внутри класса созданы статические функции. Статические функции – это функции, которые не требуют создание объекта. На рисунке 40 представлен код файла обработки ошибок.

```

require('dotenv').config();
const express = require('express');
const sequelize = require('./db');
const models = require('./models/models');
const cors = require('cors');
const fileUpload = require('express-fileupload');
const PORT = process.env.PORT || 5000;
const router = require('./routes/index');
const errorHandler = require('./middleware/ErrorHandlerMiddleware');
const path = require('path');

const app = express();
app.use(cors());
app.use(express.json());
app.use(express.static(path.resolve(__dirname, 'static')))
app.use(fileUpload({}))
app.use('/api', router)

app.use(errorHandler)

const start = async () =>
{
  try {
    await sequelize.authenticate();
    await sequelize.sync();
    app.listen(PORT, () => console.log(`Listening on ${PORT}`))
  } catch (e)
  {
    console.log('error');
  }
}

start()

```

Рисунок 39 - Главный файл сервера index.js

```

class ApiError extends Error{
  constructor(status, message){
    super();
    this.status = status
    this.message = message
  }
  static badRequest(message) {
    return new ApiError(404, message)
  }
  static internal(message) {
    return new ApiError(500, message)
  }

  static forbidden(message) {
    return new ApiError(403, message)
  }
}

module.exports = ApiError

```

Рисунок 40 - Код файла обработки ошибок ApiError.js

В папке middleware хранится файл ErrorHandlerMiddleware.js, который необходим как промежуточное звено между функцией и запросом для проверки. Реализация проверки на ошибки представлена на рисунке 41.

```
const ApiError = require (`../error/ApiError`);

module.exports = function (err, req, res, next) {
  if (err instanceof ApiError){
    return res.status(err.status).json({message: err.message})
  }
  return res.status(500).json.json({message: "Непредвиденная ошибка"})
}
```

Рисунок 41 - Код файла ErrorHandlerMiddleware.js

Папка routes хранит в себе маршруты для методов. Каждый роутер хранит в себе метод добавления и получения данных. Реализация маршрутов представлена на рисунке 42.

```
const Router = require (`express`)
const router = new Router()
const userRouter = require(`./userRouter`)
const courseRoute = require(`./courseRouter`)
const moduleRouter = require(`./moduleRouter`)
const lessonRouter = require(`./lessonRouter`)
const exerciseRouter = require(`./exerciseRouter`)
const progressRouter = require(`./progressRouter`)
const teacherRouter = require(`./teacherRouter`)

router.use(`/user`, userRouter)
router.use(`/course`, courseRoute)
router.use(`/module`, moduleRouter)
router.use(`/lesson`, lessonRouter)
router.use(`/exercise`, exerciseRouter)
router.use(`/progress`, progressRouter)
router.use(`/teacher`, teacherRouter)

module.exports = router
```

Рисунок 42 - Код файла index.js

В moduleRouter описаны запросы на получение и отправку данных на сервер. Получать данные можно сразу все, либо по одному в зависимости от запроса. Реализация запросов на получение и отправку данных представлена на рисунке 43.

```
const Router = require(`express`)
const router = new Router()
const moduleController = require(`../controllers/moduleController`)

router.post(`/`, moduleController.create)
router.get(`/`, moduleController.getAll)
router.get(`/`, moduleController.getOne)

module.exports = router
```

Рисунок 43 - Код файла moduleRouter.js

В userRouter с помощью метода registration, происходит регистрация пользователя, с помощью метода login происходит вход посетителя сайта, а с помощью метода auth выполняется проверка пользователя на авторизованность пользователя в приложении. Реализация регистрации и авторизации представлена на рисунках 44-49.

```
const Router = require(`express`)
const router = new Router()
const userController = require(`../controllers/userController`)
const authMiddleware = require(`../middleware/authMiddleware`)

router.post(`/registration`, userController.registration)
router.post(`/login`, userController.login)
router.get(`/auth`, authMiddleware, userController.check)

module.exports = router
```

Рисунок 44 - Код файла userRouter.js

```

const Router = require (`express`)
const router = new Router()
const teacherController = require('../controllers/teacherController')

router.post(`/`, teacherController.create)
router.get(`/`, teacherController.getAll)

module.exports = router

```

Рисунок 45 - Код файла teacherRouter.js

```

const Router = require (`express`)
const router = new Router()
const progressController = require('../controllers/progressController')

router.post(`/`, progressController.create)
router.get(`/`, progressController.getAll)

module.exports = router

```

Рисунок 46 - Код файла progressRouter.js

```

const Router = require (`express`)
const router = new Router()
const courseController = require('../controllers/courseController')

router.post(`/`, courseController.create)
router.get(`/`, courseController.getAll)
router.get('/:id`, courseController.getOne)

module.exports = router

```

Рисунок 47 - Код файла courseRouter.js

```

const Router = require (`express`)
const router = new Router()
const exerciseController = require('../controllers/exerciseController')
const checkRole = require('../middleware/checkRoleMiddleware')

router.post(`/`,checkRole(`ADMIN`), exerciseController.create)
router.get(`/`, exerciseController.getAll)

module.exports = router

```

Рисунок 48 - Код файла exerciseRouter.js

```

const Router = require (`express`)
const router = new Router()
const lessonController = require('../controllers/lessonController')

router.post(`/`, lessonController.create)
router.get(`/`,lessonController.getAll)

module.exports = router

```

Рисунок 49 - Код файла lessonRouter.js

Папка controllers хранит в себе контроллеры для каждого роутера.

В файле userController используется одноименный класс, внутри которого описаны функции регистрации, авторизации и проверка на авторизованность. Перед описанием класса необходимо инициализировать генерацию токенов и задать секретный ключ для генерации, также время действия для того, чтобы в случае кражи данных по истечении 24 часов токен был не актуален.

В регистрации получаем данные из тела запроса, соответственно email, пароль и роль. Затем происходит проверка, если строка с emailом или паролем пустые, то возвращается ошибка. Затем происходит проверка на то, есть ли уже такой пользователь в системе. Если пользователь не найден, то хешируется пароль.

Авторизация происходит путем проверки наличия такого пользователя в бд, в случае не успеха, выдаётся ошибка и соответствующее сообщение.

Инициализация и функции регистрации, авторизации представлены на рисунках 50-52.

```
const bcrypt = require( bcrypt )
const {User, Exercise} = require(`../models/models`)
const ApiError = require (`../error/ApiError`);
const jwt = require(`jsonwebtoken`)
const generateJwt = (id, email, role) =>{
  return jwt.sign(
    {id: id, email, role},
    process.env.SECRET_KEY,
    {expiresIn: `24h`}
  )
}
```

Рисунок 50 - Инициализация функций userController.js

```
class UserController{
  async registration (req, res, next){
    const {email, password, role} = req.body
    if (!email || !password) {
      return next(ApiError.badRequest(`Некорректный email или password`))
    }
    const candidate = await User.findOne({where:{email}})
    if (candidate) {
      return next(ApiError.badRequest(`Пользователь с таким email уже существует`))
    }
    const hashPassword = await bcrypt.hash(password, 4)
    const user = await User.create({email, role, password: hashPassword})
    const exercise = await Exercise.create({userId: user.id})
    const token = generateJwt(user.id, user.email, user.role)
    return res.json({token})
  }
}
```

Рисунок 51 - Функция регистрации userController.js

```
async login(req,res, next){
  const {email, password} = req.body
  const user = await User.findOne({where: {email}})
  if (!user){
    return next(ApiError.internal(`Пользователь не найден`))
  }
  let comparePassword = bcrypt.compareSync(password, user.password)
  if (!comparePassword){
    return next(ApiError.internal(`Указан неверный пароль`))
  }
  const token = generateJwt(user.id, user.email, user.role)
  return res.json({token})
}
```

Рисунок 52 - Функция авторизации userController.js

Остальные контроллеры папки controllers приведены в приложении Б.

В папке middleware хранится файл authMiddleware.js, внутри которого проверяется происходит декодирование токена и проверка его на валидность. Если токен не валидный, то сразу возвращается соответствующая ошибка. На рисунке 53 представлена функция декодирования и проверки на валидность.

```
const jwt = require('jsonwebtoken')
module.exports = function (req, res, next) {
  if (req.method === "OPTIONS"){
    next ()
  }
  try {
    const token = req.headers.authorization.split(``)[1]
    if (!token){
      return res.status(401).json({message: "Не авторизован"})
    }
    const decoded = jwt.verify(token, process.env.SECRET_KEY)
    req.user = decoded
    next()
  } catch(e) {
    res.status(401). json({message: "Не авторизован"})
  }
}
```

Рисунок 53 - Функция декодирования и проверки на валидность
authMiddleware.js

1.4.3 Тестирование и отладка

После запуска приложения, отображается главная страница веб-приложения (рисунок 54). Чтобы создать или выбрать курс, необходимо зарегистрироваться или войти как «студент» или как «преподаватель». Для этого нужно в правом верхнем углу выбрать соответствующую кнопку (рисунки 55-57). После выбора соответствующей кнопки, пользователь переходит на окно авторизации для соответствующего типа пользователя. Если же пользователь ещё не зарегистрирован, ему необходимо ние под формой нажать на кнопку регистрации и заполнить все данные, затем вернуться к авторизации и войти в свой профиль.

Доступ к образованию в любом месте!

Широкий выбор курсов и возможность создания курсов для преподавателей.
Зарегистрируйся прямо сейчас и попробуй все возможности приложения!

Рисунок 54 – Отображение главной страницы приложения

Авторизация преподавателя

✉ Электронная почта

🔒 Пароль



Соглашаюсь с условиями пользования

Войти

[У меня нет аккаунта](#) [Регистрация](#)

Рисунок 55 – Авторизация преподавателя

					ПГУ 1.090501.10.001 ПЗ	Лист
						66
Изм.	Лист	№ докум.	Подпись	Дата		

Авторизация студента

 stepadsh@mail.ru



☒ Соглашаюсь с условиями пользования

Войти

[У меня нет аккаунта](#) [Регистрация](#)

Рисунок 56 – Авторизация студента

Авторизация студента

 stepadshkl...|

 Please include an '@' in the email address. 'stepadshkl...' is missing an '@'.

☒ Соглашаюсь с условиями пользования

Войти

[У меня нет аккаунта](#) [Регистрация](#)

Рисунок 57 – Авторизация студента с некорректным вводом данных

Если пользователь впервые заходит на сайт, ему доступна регистрация. Для этого необходимо войти на вкладку авторизации и снизу выбрать «Регистрация». На рисунке 58 представлена форма регистрации.

Регистрация студента

 Электронная почта

 Пароль

☒ Соглашаюсь с условиями пользования

Регистрация

Рисунок 58 – Форма регистрации

После регистрации, выводится окошко об успешной регистрации (рисунок 59).



Вы успешно зарегистрировались!

Рисунок 59 – Сообщение об успешной авторизации

После авторизации преподавателя, открывается домашняя страница преподавателя. На ней выполняется основная навигация по сайту (рисунок 60).

Вы зашли как преподаватель

Мой профиль	▼
Создать курс	▼
Мои курсы	▼
Отчет по курсам	▼

Рисунок 60 – Домашняя страница преподавателя

Вы зашли как преподаватель

Мой профиль	^
Перейти	
Создать курс	▼
Мои курсы	▼
Отчет по курсам	▼

Рисунок 61 – Выбор вкладки для перехода на страницу

Преподаватель может перейти на свой профиль, посмотреть и заполнить данные о себе (рисунок 62). Также может перейти на страницу создания курса (рисунок 63). Преподавателю доступна вкладка просмотра текущих курсов, созданных им (рисунок 64) и вкладка просмотра прогресса по каждому курсу (рисунок 70).

Редактирование профиля

Ваше имя *

Дарья

Сохранить

Фамилия *

Обо мне

Аватарка



Рисунок 62 – Личный профиль пользователя

Создание курса



Логотип

png-файл
с прозрачностью
230x230px

Название

Максимум 64 символа

15/64

Краткое описание ?

Для публикации нужно больше 100 символов

113/512

Создать

Рисунок 63 – Вкладка «создание курса»

Доступные курсы

Основы программирования	^
Перейти	
Проектирование систем	^
Перейти	
Микропроцессоры	^
Перейти	

Рисунок 64 – Вкладка просмотра текущих курсов

Для того, чтобы создать курс, необходимо перейти во вкладку «создать курс». Затем добавить краткое описание курса, чтобы студент мог ознакомиться с содержанием курса и название курса, загрузить картинку для курса и нажать создать (рисунок 65).

Создание курса



Название

Максимум 64 символа

15/64

Краткое описание ?

Изучение программно-управляемого устройства, предназначенного для обработки цифровой или коммерческой информации.

Для публикации нужно больше 100 символов

113/512

Рисунок 65 – Создание курса

После нажатия на кнопку создания курса, откроется страница с добавлением модулей и уроков. А также возможность после создания урока, его отредактировать и добавить задание (рисунки 66-69). На рисунке 70 продемонстрирована вкладка прогресса.

Онлайн-курсы для специалистов

Преподаватель

Выйти

Программа курса

В курсе пока нет ни одного урока.
Создайте первый модуль, чтобы добавить уроки

Новый модуль

Рисунок 66 – Окно наполнения курса

Онлайн-курсы для специалистов

Преподаватель

Выйти

Программа курса

1

Новый модуль

Всего баллов: 0

⋮

Дополнительное описание

Начало модуля ?

Новый модуль

<

Июнь

▼ 2024

>

Пн	Вт	Ср	Чт	Пт	Сб	Вс
27	28	29	30	31	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30

Этого урока и нажмите Enter.

Добавить урок

Рисунок 67 – Добавление модуля

Настройки урока

История

осталось 57 символов

+ Добавить новый шаг

Шаг 1: Текст ...

← → **B** *I* U `</>` Paragraph ▾ **A** ▾ Source

Разработка 8080 началась только в середине 1972 года, спустя шесть месяцев после того, как Федерико Фаджин начал убеждать руководство Intel в необходимости его создания. К тому времени начали зарождаться потенциальные рынки для микропроцессоров. Ранее в основном считалось, что микропроцессоры должны сосуществовать с доминирующими и более мощными мейнфреймами и миникомпьютерами. Компьютеры по-прежнему рассматривались как дорогой инструмент для бизнеса и исследований, а рынки для нового поколения относительно дешёвых персональных машин и промышленных контроллеров не существовали и во многих случаях никто и подумать не мог об их возможности.

body p

Рисунок 68 – Редактирование урока

Выберите тип задания и нажмите на кнопку добавить

- Текстовый отчет
- Выбор из списка
- Заполнение пробелов
- Загрузить файл

Рисунок 69 – Добавление тестового задания

Онлайн-курсы для
специалистов

Преподаватель

Выйти

Табель успеваемости курса Микропроцессоры

Поиск по ID, email, имени учащегося

Пользователь ▾		Итог	История
Мах:		0	
123450	Учащийся 1	0	
123451	Учащийся 2	0	
123452	Учащийся 3	0	

Рисунок 70 – Вкладка просмотра прогресса

После входа в систему у пользователя отображается список доступных курсов, а также есть возможность перейти во вкладку профиля, чтобы заполнить данные или обновить. На рисунке 71-72 представлена главная страница пользователя.

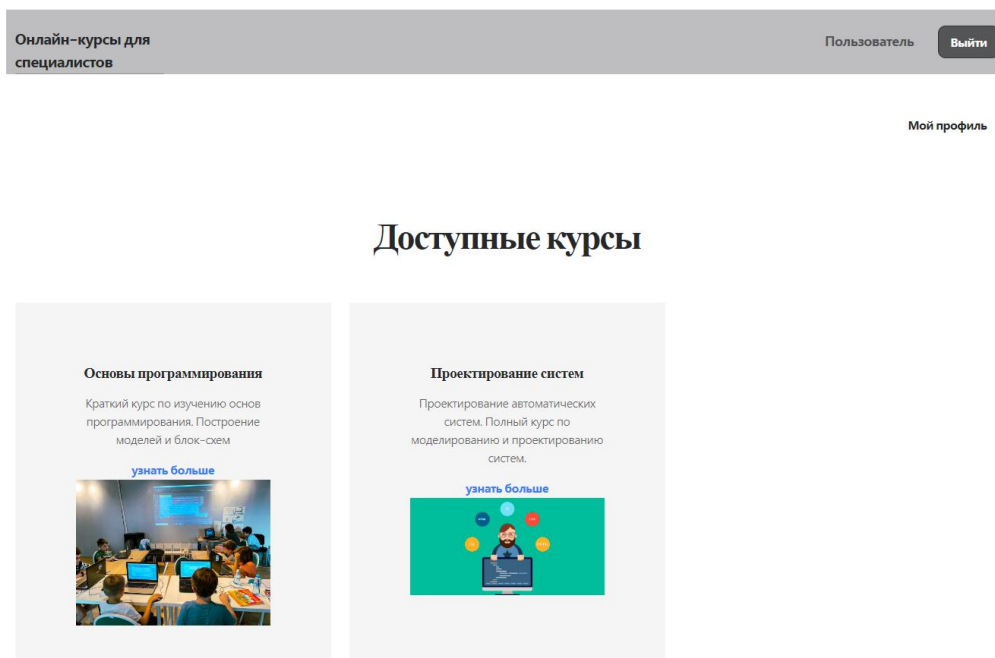


Рисунок 71 – Главная страница пользователя студента

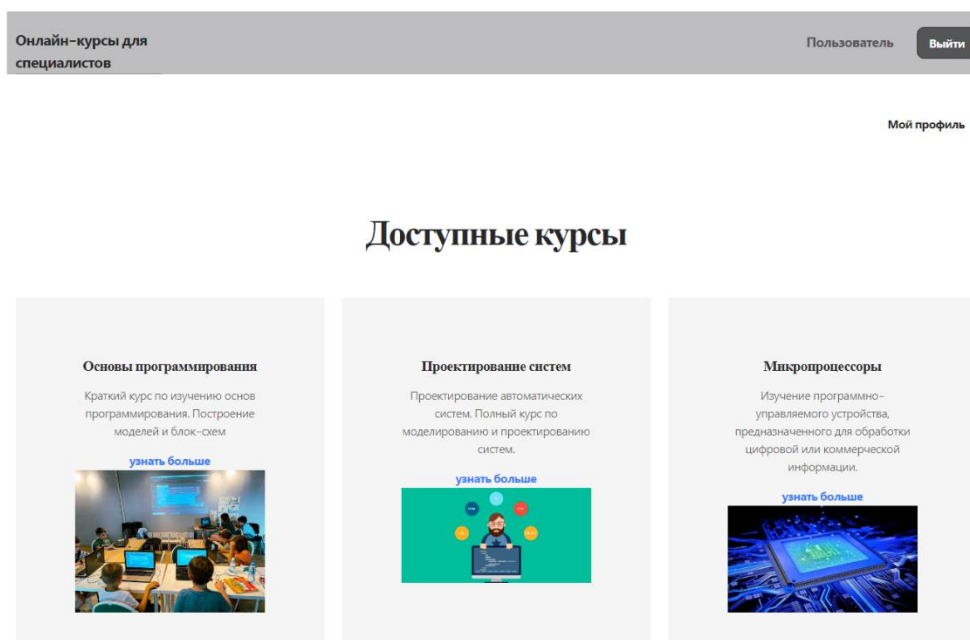


Рисунок 72 – Главная страница пользователя студента после добавления преподавателем нового курса

Пользователь может открыть любой курс по нажатии «узнать больше». После чего откроется описание курса и его модули. Пользователь может записаться на курс, после чего ему будут доступны уроки и задания. На рисунке 73 изображена страница курса, а на 74 рисунке представлен урок из курса.

Онлайн-курсы для специалистов
Пользователь
Выйти

Мой профиль

Микропроцессоры

Изучение программно-управляемого устройства, предназначенного для обработки цифровой или коммерческой информации.



Модули курса

История и архитектура микропроцессора

История: Прослеживает эволюцию микропроцессоров от простых 4-битных моделей до современных многоядерных монстров.

[Перейти](#)

Аппаратные средства защиты микропроцессора

Методы защиты: Описание различных аппаратных методов защиты микропроцессоров от несанкционированного доступа, вирусов и других угроз.

[Перейти](#)

Вычислительный конвейер

Принцип работы: Определение прекращения работы конвейерной обработки данных. Позволяет повысить производительность микропроцессора.

[Перейти](#)

Организация

Структурные блоки: Детальное описание основных компонентов микропроцессора, таких как ALU, контроллеры, кэш-память и т.д.

[Перейти](#)

Взаимодействие элементов микропроцессорных систем

Шина: Объяснение режима работы в качестве основного средства передачи данных между компонентами системы.

[Перейти](#)

Управление развитием микропроцессорных систем

Многозадачность: Описание относится к использованию многоядерных микропроцессоров для повышения производительности.

[Перейти](#)

Рисунок 73 – Страница курса

Онлайн-курсы для специалистов
Пользователь
Выйти

Мой профиль

Урок 1: История

Разработка 8080 началась только в середине 1972 года, спустя шесть месяцев после того, как Федерико Фаджин начал убеждать руководство Intel в необходимости его создания. К тому времени начали зарождаться потенциальные рынки для микропроцессоров. Ранее в основном считалось, что микропроцессоры должны сосуществовать с доминирующими и более мощными мейнфреймами и миникомпьютерами. Компьютеры по-прежнему рассматривались как дорогой инструмент для бизнеса и исследований, а рынки для нового поколения относительно дешевых персональных машин и промышленных контроллеров не существовали и во многих случаях никто и подумать не мог об их возможности.

Задание: найти и расписать историю развития микропроцессоров. Прикрепить файл.

Добавить файлы

Перейти к следующему уроку

Рисунок 74 – Страница урока со стороны пользователя

2 Технико-экономическое обоснование разработки

Суть проведения технико-экономического обоснования разработки заключается в установлении экономической целесообразности создания и усовершенствования веб-приложения. Это также включает в себя определение организационных и экономических предпосылок для его эффективного функционирования. Эффективность программного модуля зависит как от качества самого модуля, так и от эффективности процессов разработки и сопровождения.

Качество ПО определяется тремя основными аспектами:

- отзывы пользователей: мнения и уровень удовлетворенности конечных пользователей функциональностью и удобством использования программного обеспечения;
- эффективность использования ресурсов: эффективность использования программным обеспечением таких ресурсов, как оперативная память ПК, процессорное время, пропускная способность сети и других, во время использования, а также временных и финансовых ресурсов, затрачиваемых на разработку и эксплуатацию;
- соответствие требованиям: степень, в которой программное обеспечение соответствует заранее определенным функциональным и техническим характеристикам.

При разработке программного продукта важно учитывать оптимальное потребление ресурсов на этапе использования, а также минимизировать время разработки и эксплуатационные расходы.

Основные данные, которые нужны для расчета экономических показателей, представлены в таблице 11.

Таблица 11 - Основные данные для расчета экономических показателей

Обозначение	Наименование показателя	Единицы измерения	Значение
СЭВМ	Стоимость компьютерного оборудования и периферийных устройств	тыс. руб	140
P	Мощность, потребляемая ЭВМ	кВт	0,6
qI	Количество I-задач, решаемых потребителем	Зад. год	70
pн	Норма рентабельности	-	0,26
ТС	Срок службы разработанной программы	год	5
ωс	Коэффициент, учитывающий взносы	-	0,26
SD	Стоимость деталей, подлежащих замене во время ремонта	Руб.	2000
ωд	Коэффициент, учитывающий дополнительную заработную плату разработчика программы	-	0,2
tM.B.I	Время, затрачиваемое разработанной программой на решение i-й задачи	Маш.час	1
пп	Количество организаций, которые, как ожидается, приобретут программу	шт.	10

Обозначение	Наименование показателя	Единицы измерения	Значение
ZЭЛ	Тариф за 1 кВт/час	Руб.	4,13
tCM	Продолжительность смены	ч	8
НДС	Налог на добавленную стоимость	%	20
ТР	Количество рабочих дней в году	дн	264
NCM	Количество смен работы ЭВМ	-	1
ДМ	Среднее количество рабочих дней в месяц	дни	23
α	Простои ЭВМ	%	15
ωН	Коэффициент, учитывающий организационные накладные расходы	-	1, 34
NCP	Среднее количество ремонтов в год	-	3

Расчёт затрат на разработку ПО:

Суммарные затраты на разработку проекта рассчитываются по следующей формуле:

$$S_{\text{РП}} = S_{\text{ЗП}} + S_{\text{НАК}}, \quad (1)$$

где $S_{\text{ЗП}}$ – затраты по заработной плате инженера-программиста;

$S_{\text{НАК}}$ – накладные расходы.

Затраты по заработной плате инженера-программиста рассчитываются по формуле:

$$S_{\text{ЗП}} = O_{\text{ЗП}} * (1 + \omega_c) * (1 + \omega_d) * tp_i, \quad (2)$$

где $O_{3П}$ – основная заработная плата инженера-программиста за месяц (30 000 руб.).

Система разрабатывалась 35 дней, если учесть, что в одном месяце 23 рабочих дней, то:

$$tp_{12} = 35/23 = 1,5 \text{ (чел./Мес)}. \quad (3)$$

Таким образом, затраты по заработной плате инженера-программиста составят:

$$S_{3П} = 30\,000 * (1+0,26) * (1+0,2) * 1,5 = 68\,040 \text{ (руб.)}.$$

Накладные затраты рассчитываются по формуле:

$$S_{НАК} = O_{3П} * \omega_n * tp_{12}, \quad (4)$$

$$S_{НАК} = 30\,000 * 1,34 * 1,5 = 60\,300 \text{ (руб.)}.$$

Таким образом, суммарные затраты на разработку системы составляют:

$$S_{РП} = 68\,040 + 60\,300 = 128\,340 \text{ (руб.)}.$$

Оптовая цена разработанной программы определяется по следующей формуле:

$$Z_{П} = S_{РП} + П, \quad (5)$$

где $Z_{П}$ – оптовая цена (цена разработчика) (руб.);

$S_{РП}$ – суммарные затраты на разработку программы (руб.);

$П$ – прибыль.

Прибыль рассчитывается по формуле:

$$П = p_n * S_{РП}, \quad (6)$$

где p_n – норматив рентабельности, учитывающий прибыль организации, разрабатывающей данную программу в долях ко всем затратам данной организации на разработку программы.

Итак,

$$Z_{П} = S_{РП} * (1 + p_n), \quad (7)$$

$$Z_{П} = 128\,340 * (1 + 0,26) = 161\,708,4 \text{ (руб.)}.$$

Розничная цена программы рассчитывается с учетом налога на добавленную стоимость (НДС = 20%) по формуле:

$$Z_{Пр} = Z_{П} + H_{ДС} = Z_{П} + (Z_{П} * 20)/100 = Z_{П} * (1 + 0,20), \quad (8)$$

$$Z_{Пр} = 161\,708,4 * (1 + 0,20) = 194\,050,08 \text{ (руб.)}.$$

Выручка от продаж при условии $n_{п}$ – количество организаций, желающих приобрести web-приложение, составит:

$$B = Z_{Пр} * n_{п}, \quad (9)$$

$$B = 194\,050,08 * 10 = 1\,940\,500,8 \text{ (руб.)}.$$

Расчет капитальных вложений:

Капиталовложения, связанные с работой ЭВМ рассчитываются по формуле:

$$K_{ЭВМ} = C_{ЭВМ} + S_3, \quad (10)$$

где $C_{ЭВМ}$ – стоимость ЭВМ (руб.);

S_3 – стоимость запасных частей (руб.).

Итак, произведем расчет коэффициентов, входящих в формулу расчета величины капиталовложений:

$$S_3 = 0,15 * C_{ЭВМ}, \quad (11)$$

$$S_3 = 0,15 * 140\,000 = 21\,000 \text{ (руб.)}.$$

Капиталовложения в ЭВМ периферию составляют:

$$K_{ЭВМ} = 140\,000 + 21\,000 = 161\,000 \text{ (руб.)}.$$

Расчет эксплуатационных расходов:

Эксплуатационные расходы на ЭВМ рассчитываются по формуле:

$$E = (T_{МВ} * E_{ч}) + Z_{п}/T_{с}, \quad (12)$$

где $T_{МВ}$ – машинное время для решения задач с помощью разработанного web-приложения, (маш.час/год);

$E_{ч}$ – эксплуатационные расходы, приходящиеся на 1 час работы ЭВМ;

$Z_{п}$ – цена, по которой продается web-приложение (руб.);

$T_{с}$ – срок службы программы (г).

Полезный фонд времени работы ЭВМ рассчитывается по формуле:

$$T_{пол} = T_{общ} * t_{см} * N_{см} * (1 - \alpha/100), \quad (13)$$

где $T_{общ}$ – общий фонд времени работы ЭВМ (дни); $T_{общ} = T_{р}$;

Полезный фонд времени работы ЭВМ получим:

$$T_{пол} = 264 * 8 * 1 * (1 - 15/100) = 1795,2 \text{ (Маш. час /год)}.$$

Машинное время для решения задач с помощью данной программы рассчитывается по формуле:

$$T_{\text{МВ}} = qI * t_{\text{МВ}I}, \quad (14)$$

где qI – количество I -задач, решаемых потребителем в год(шт.);

$t_{\text{МВ}I}$ – время решения I -ой задачи, разработанной программой (Маш. час).

$$T_{\text{МВ}} = 70 * 1 = 70 \text{ (Маш. час / год)}.$$

Эксплуатационные расходы, приходящиеся на 1 час работы ЭВМ, оцениваются по формуле:

$$E_{\text{ч}} = (A_{\text{о}} + S_{\text{зп}} + S_{\text{эл}} + R_{\text{рм}}) / T_{\text{пол}}, \quad (15)$$

где $A_{\text{о}}$ – амортизационные отчисления (руб.);

$S_{\text{зп}}$ – затраты по заработной плате инженера в год (руб./год); $S_{\text{эл}}$ – стоимость потребляемой энергии (руб.);

$R_{\text{рм}}$ – затраты на ремонт ЭВМ (руб.);

$T_{\text{пол}}$ – полезный годовой фонд работы ЭВМ, (маш.час/год).

Амортизационные отчисления рассчитываются с учетом нормы амортизации ($A_{\text{н}} = 12,5 \%$);

$$A_{\text{о}} = A_{\text{н}} * K_{\text{ЭВМ}} = 0,125 * 161\,000 = 20\,125 \text{ (руб.)}. \quad (16)$$

Затраты по заработной плате инженера за год рассчитывается по формуле:

$$S_{\text{зп}} = (1 + \omega_{\text{с}}) * (1 + \omega_{\text{д}}) * O_{\text{зп}} * 1,2, \quad (17)$$

где $\omega_{\text{с}}$ – коэффициент, учитывающий начисления органам социального страхования на заработную плату разработчика программы, в долях к сумме основной заработной плате разработчика;

$\omega_{\text{д}}$ – коэффициент, учитывающий дополнительную заработную плату разработчика программы, в долях к сумме основной заработной платы;

$O_{\text{зп}}$ – основная заработная плата инженера за месяц 8-го разряда (25000 руб.).

Рассчитываем годовые затраты по заработной плате и социальным отчислениям для инженера:

$$S_{\text{зп}} = (1 + 0,26) * (1 + 0,2) * 30\,000 * 1,5 = 68\,040 \text{ (руб./ год)}.$$

Стоимость потребляемой энергии оценивается по формуле:

$$S_{\text{эл}} = P * T_{\text{пол}} * Z_{\text{эл}}, \quad (18)$$

где P – мощность, потребляемая ЭВМ (кВт);

$T_{\text{ПОЛ}}$ – полезный годовой фонд работы ЭВМ (маш.час/год);

$Z_{\text{ЭЛ}}$ – тариф за 1 кВт/час (руб. /кВт).

Итак, произведем расчет стоимости потребляемой энергии:

$$S_{\text{ЭЛ}} = 0,6 * 1\,795,2 * 4,13 = 4\,448,5 \text{ (руб.)}.$$

Затраты на ремонт ЭВМ вычисляются по формуле:

$$R_{\text{РМ}} = N_{\text{СР}} * S_{\text{Д}}, \quad (19)$$

где $N_{\text{СР}}$ – среднее количество ремонтов в год;

$S_{\text{Д}}$ – стоимость деталей, заменяемых при одном ремонте, в среднем.

$$R_{\text{РМ}} = 3 * 2\,000 = 6\,000 \text{ (руб.)}.$$

Произведем вычисление эксплуатационных расходов, приходящихся на 1 час работы ЭВМ:

$$E_{\text{ч}} = (20\,125 + 68\,040 + 4\,448,5 + 6000)/1795,2 = 55 \text{ (руб./час)}.$$

Далее вычислим затраты на ЭВМ:

$$E = (70 * 55) + 161\,708,4/5 = 36\,191,68 \text{ (руб.)}.$$

Для упрощения задач программиста и повышения производительности проект предусматривает использование всемирной паутины. В настоящее время на интернет-рынке Пензы работают многочисленные поставщики интернета. Цены, которую предлагают компании, сведены в таблицу 12, что позволяет провести анализ и сделать экономически обоснованный выбор.

Таблица 12 - Таблица компаний и предоставляемых ими услуг

Название провайдера	Предлагаемый тариф	Стоимость тарифа(руб.)	Скорость доступа(Мбит/с)
Ростелеком	Облачный	890	500
МТС	Тариф №3	700	500
ДОМ.РУ	Гига 500	950	500
Золотая линия	VIP 500	770	500

Учитывая, что вышеупомянутые провайдеры предлагают безлимитный интернет-трафик на одинаковой скорости, наиболее экономичным вариантом является выбор тарифного плана с наименьшей стоимостью трафика. Поэтому решено было выбрать тариф "Тариф №3", который предусматривает ежемесячную абонентскую плату в размере 700 рублей.

					ПГУ 1.090501.10.001 ПЗ	Лист
						83
Изм.	Лист	№ докум.	Подпись	Дата		

3 Безопасность жизнедеятельности

Внедрение передовых технологий существенно повлияло на условия труда программистов, что привело к повышению требований к работе и физической и умственной нагрузке. Это требует комплексного подхода к обеспечению рабочих мест, включая регулирование уровней шума, освещения и общих атмосферных условий.

Широкое использование персональных компьютеров и мобильных устройств подвергает людей воздействию потенциально вредных электромагнитных излучений (микроволновых, УВЧ, высокочастотных), шума, статических помех и других неблагоприятных факторов. Длительное использование таких устройств может привести к умственному переутомлению, зрительному переутомлению и напряжению опорно-двигательного аппарата, особенно в области шеи и спины. Важно обеспечить надлежащее оборудование и эргономичную организацию рабочих мест для пользователей и разработчиков.

Соблюдение правильного режима труда и отдыха имеет решающее значение, поскольку несоблюдение этого режима может привести к различным симптомам, таким как головные боли, нарушения сна, раздражительность, усталость и дискомфорт в нижней части спины, шее и руках.

Комфортность рабочих мест зависит от соблюдения определенных требований к качеству воздуха, уровню шума и вибрации, освещенности рабочего места и оборудованию. Требования к качеству воздуха требуют обеспечения циркуляции воздуха в рабочих помещениях, где используются компьютеры, для очистки от пыли, грязи и микроорганизмов, чтобы предотвратить присутствие вредной микрофлоры.

Система кондиционирования воздуха должна поддерживать заданные параметры микроклимата круглый год, очищать воздух от загрязняющих веществ и поддерживать достаточное давление в помещении, чтобы предотвратить попадание неподходящего воздуха. Кроме того, необходимо регулировать температуру, установленную на уровне не менее 19 градусов, в зависимости от

тепла, выделяемого оборудованием. Предпочтение отдается энергоэффективному оборудованию, при установке которого тепло не должно попадать на работников, поскольку количество устройств в помещении ограничено.

Напольное покрытие должно быть гладким, устойчивым к скольжению, легко моющимся и поддающимся влажной уборке, с антистатическими свойствами, которые сохраняются во время использования, что требует влажной уборки.

В рабочих зонах шум не должен быть громким и рекомендуется использовать малошумящее оборудование и звукопоглощающие материалы. Оборудование, производящее чрезмерный шум, должно располагаться за пределами рабочей зоны.

Требования к освещению имеют жизненно важное значение, что требует установки отдельной сети электропитания для обеспечения бесперебойной работы компьютера с акцентом на освещение рабочих мест с использованием сочетания естественного и искусственного света.

					ПГУ 1.090501.10.001 ПЗ	Лист
						85
Изм.	Лист	№ докум.	Подпись	Дата		

Заключение

В данной работе было тщательно разработано веб-приложение, призванное облегчить создание онлайн-курсов, направленных на повышение квалификации.

Был проведен тщательный анализ существующих систем дистанционного обучения и онлайн-курсов. В результате исследования была разработана концепция и создан прототип инновационного приложения, предназначенного для создания онлайн-курсов профессионального обучения. Это приложение предназначено для создания курсов с разнообразным содержанием и структурой, включающих различные типы учебных материалов (включая тексты, видео, аудиофайлы и изображения) и использующих широкий спектр педагогических подходов (таких как тесты, викторины и задания).

В ходе выполнения этой дипломной работы были определены критические спецификации приложения, разработан архитектурный проект приложения, а прототип прошел тщательное тестирование. Было принято решение создать веб-приложение с использованием языка программирования JavaScript, с использованием "React" на стороне клиента и серверных фреймворков, таких как "Nodemon", "Node.js" и "Next.js". В качестве системы управления базами данных был выбран PostgreSQL, работающий в синергии с графическим интерфейсом pgAdmin4 для взаимодействия с базами данных. В качестве предпочтительной интегрированной среды разработки (IDE) была выбрана "WebStorm".

Приложение обладает потенциалом для широкого применения в различных сферах образования и бизнеса. В целом, проведенное исследование является актуальным и имеет значительную практическую ценность. Разработанное приложение является ценным инструментом для создания эффективных онлайн-курсов, ориентированных на профессионалов.

Список использованных источников

1. Груздев В. С. Разработка приложений для обучения специалистов: теория и практика. - М.: Издательство "Питер", 2017. - 320 с.
2. Смирнов О. Д. Технологии онлайн обучения: практическое руководство. - Киев: НПУ, 2018. - 192 с.
3. Богданова Л. К. Инструменты и технологии для разработки образовательных онлайн курсов. - М.: Книга по Требованию, 2017. - 136 с.
4. Крылов А. С. Программирование приложений для онлайн обучения. - Киев: КНУ, 2018. - 200 с.
5. Куликов А. К. Разработка приложений для онлайн обучения специалистов: технологии и методы. - М.: Логос, 2018. - 220 с.
6. Федоров Д. Е. Разработка онлайн платформ для обучения специалистов. - Красноярск: Сибирское университетское издательство, 2017. - 176 с.
7. Роббинс Дженнифер HTML5, CSS3 и JavaScript. Исчерпывающее руководство (+ DVD-ROM); Эксмо - М., 2017. - 528 с.
8. Никсон Робин Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5; Питер - М., 2016. - 768 с.
9. Дэвид, Хэррон Node.js. Разработка серверных веб-приложений на JavaScript / Хэррон Дэвид. - М.: ДМК Пресс, 2016. - 667 с.
10. Официальная документация Introducing JSON. [Электронный ресурс] / Режим доступа: <https://www.json.org/json-en.html/>, свободный (дата обращения 15.04.2024).
11. Руководство по React Router [Электронный ресурс]/ – Режим доступа: <https://reacttraining.com/react-router/core/guides/quick-start> свободный (дата обращения 28.04.2024).

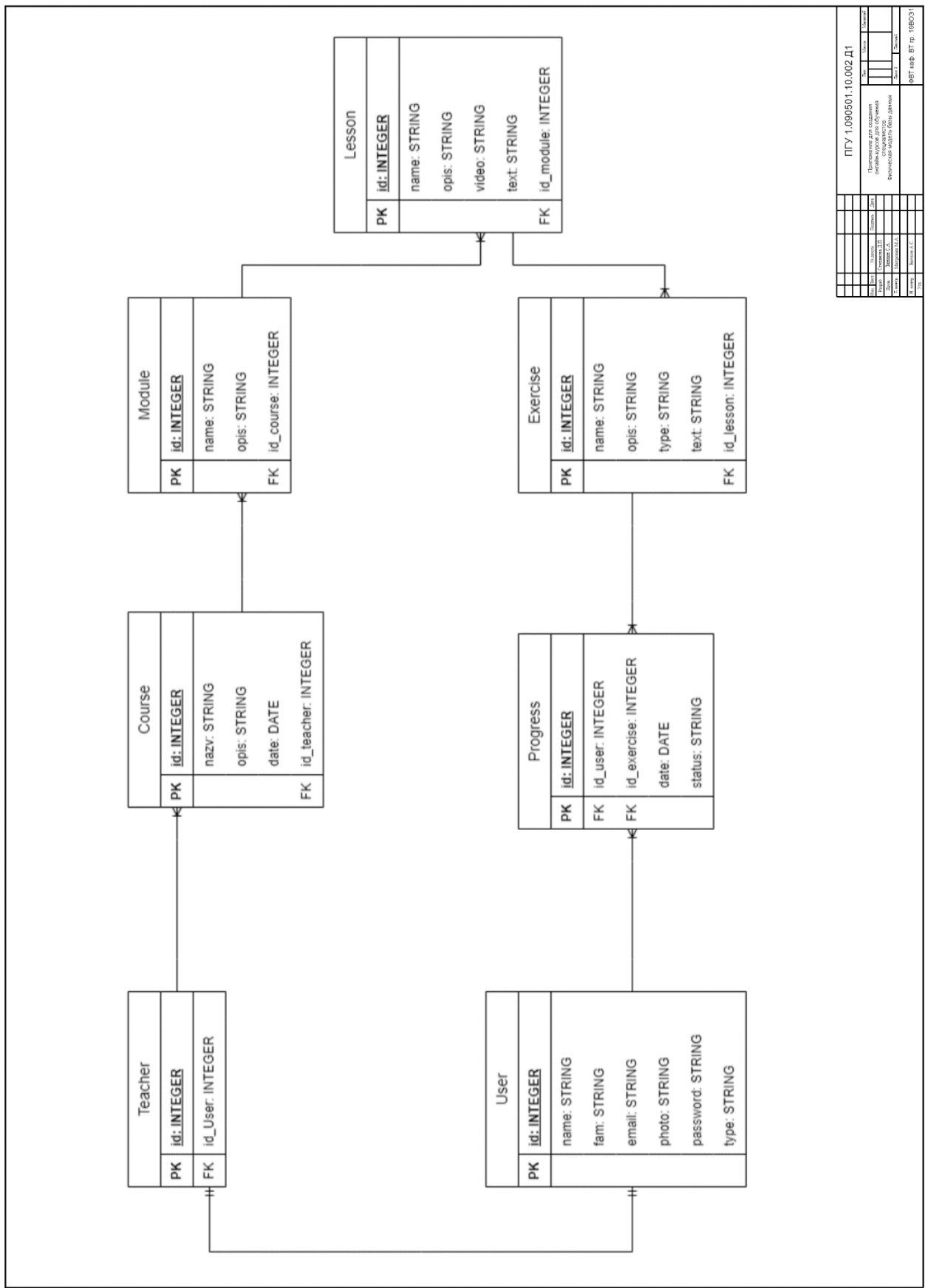
Приложение А

(Обязательное)

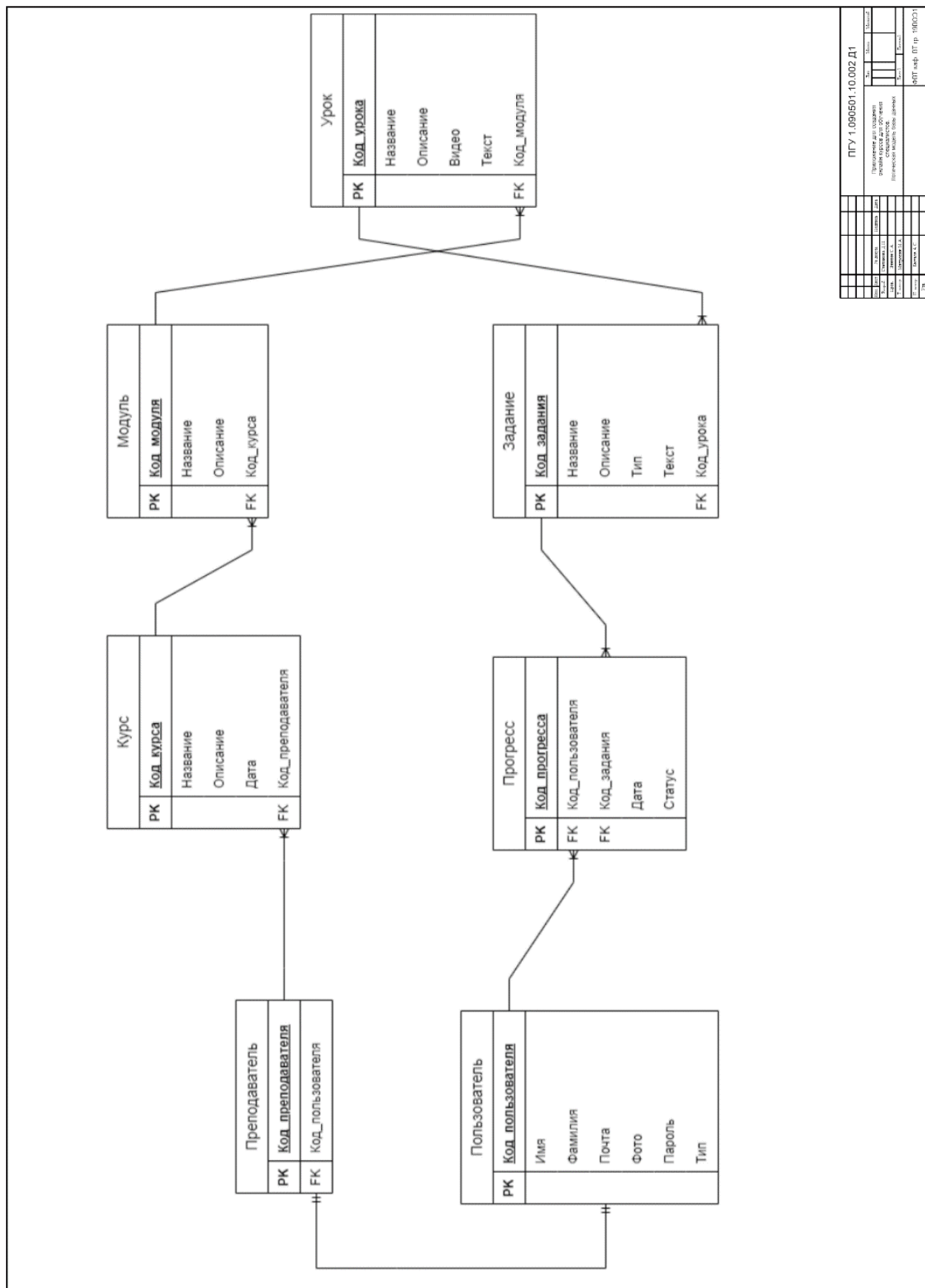
Приложение для создания онлайн-курсов для обучения специалистов
Модели базы данных

					ПГУ 1.090501.10.001 ПЗ	Лист
						88
Изм.	Лист	№ докум.	Подпись	Дата		

Приложение А.1 – Физическая модель данных



Приложение А.2 – Логическая модель данных



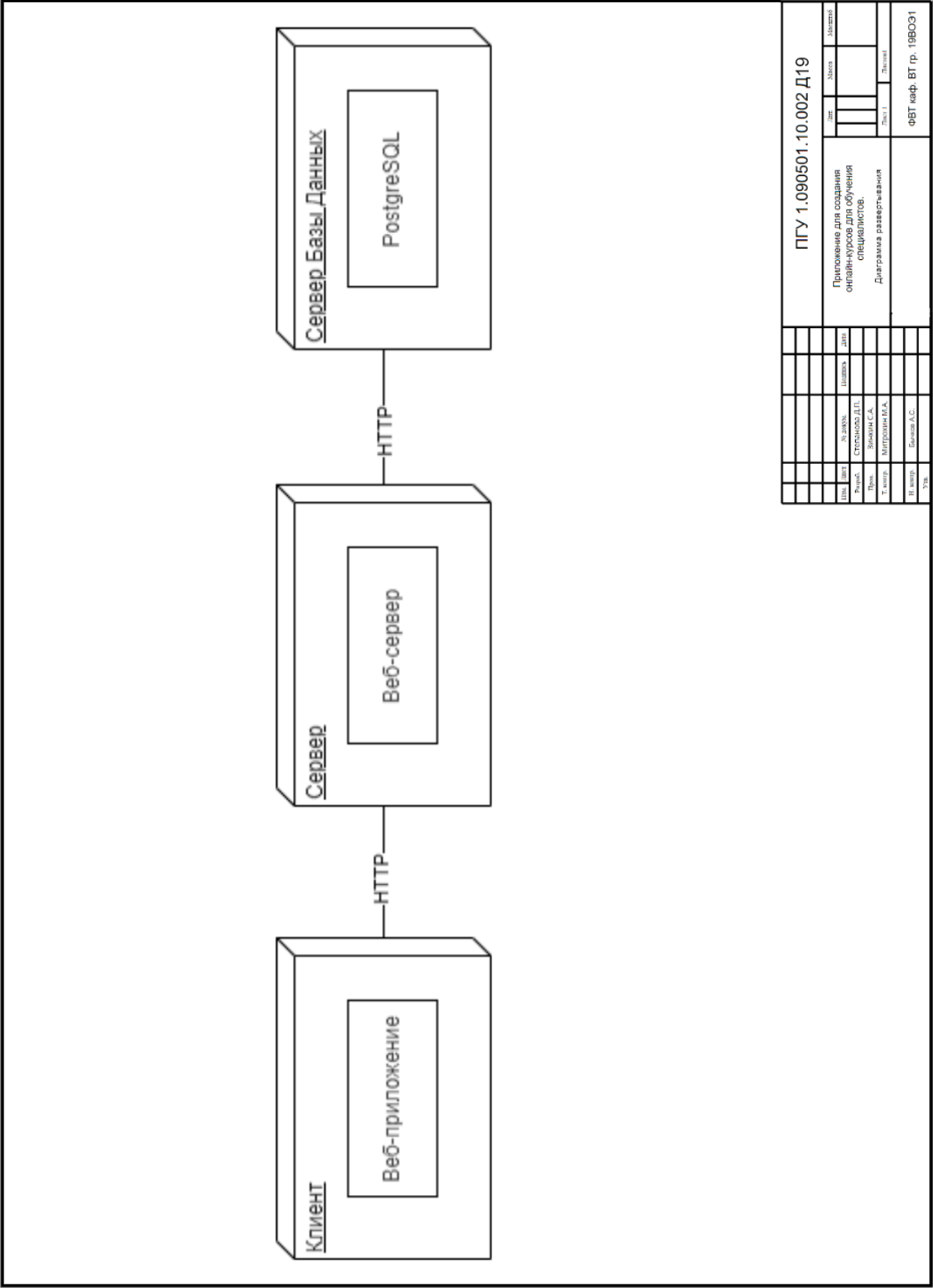
Приложение Б

(Обязательное)

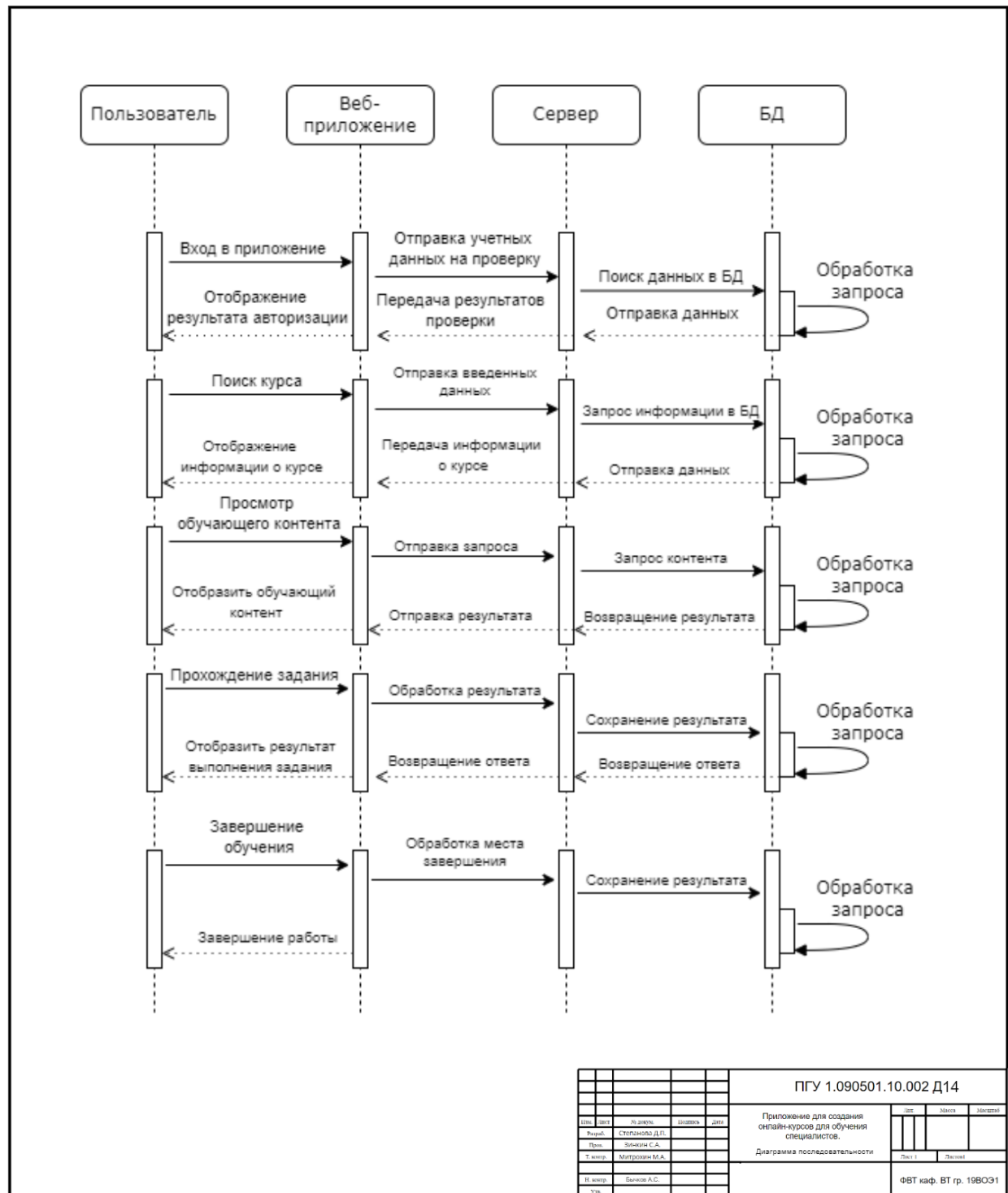
Приложение для создания онлайн-курсов для обучения специалистов
UML-диаграммы

					ПГУ 1.090501.10.001 ПЗ	Лист
						91
Изм.	Лист	№ докум.	Подпись	Дата		

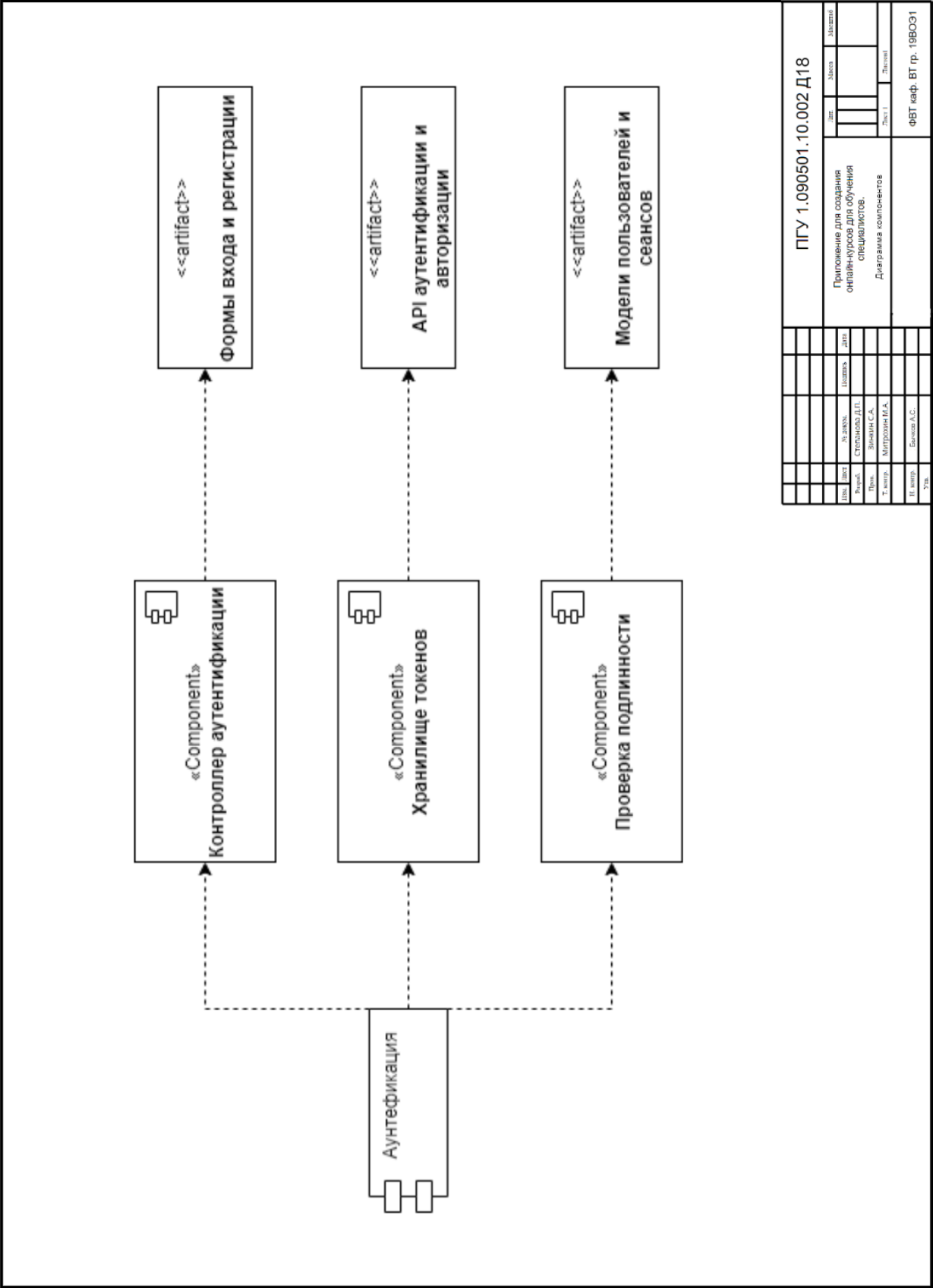
Приложение Б.1 – Диаграмма развертывания



Приложение Б.2 – Диаграмма последовательности

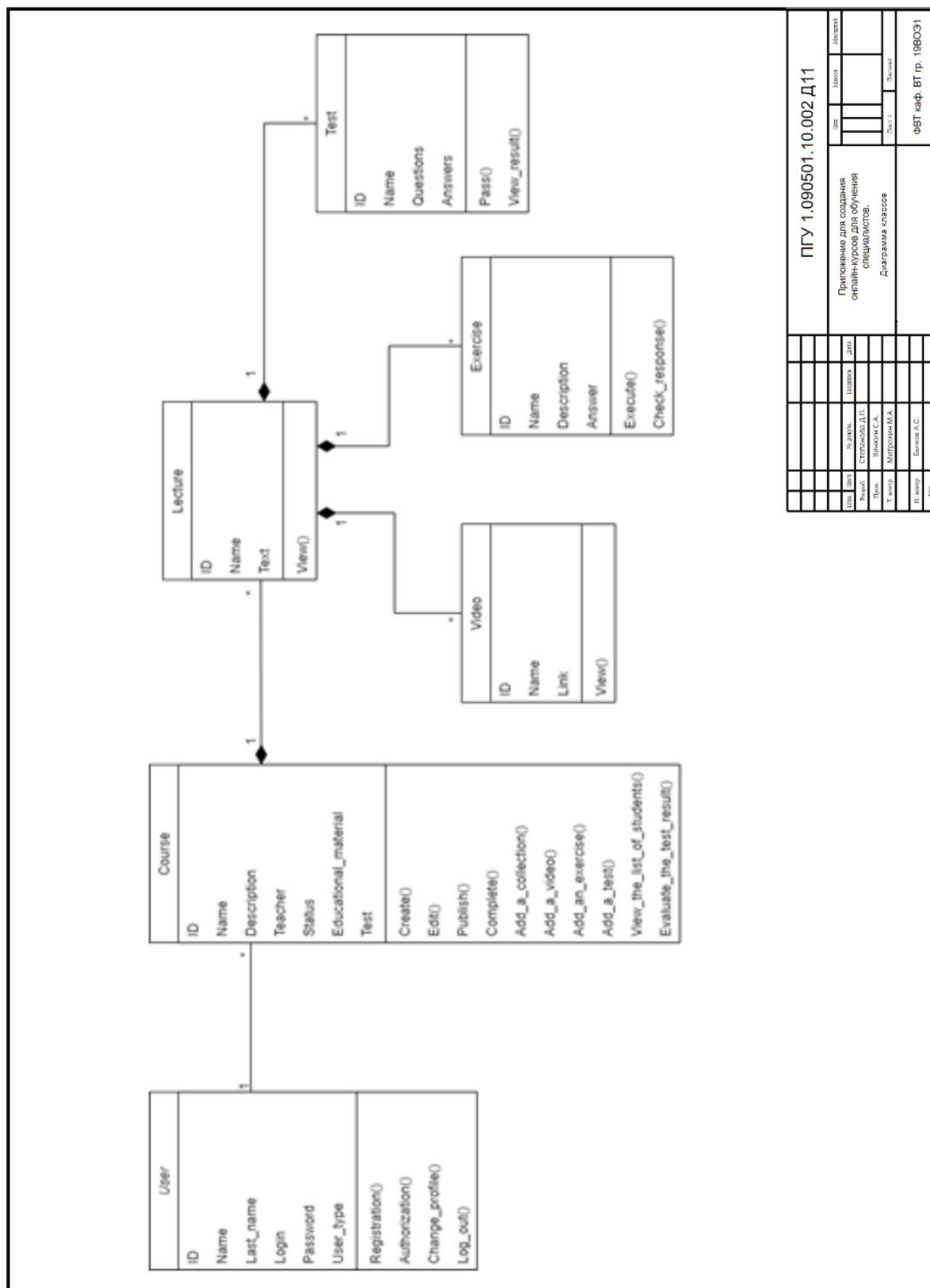


Приложение Б.3 – Диаграмма компонентов

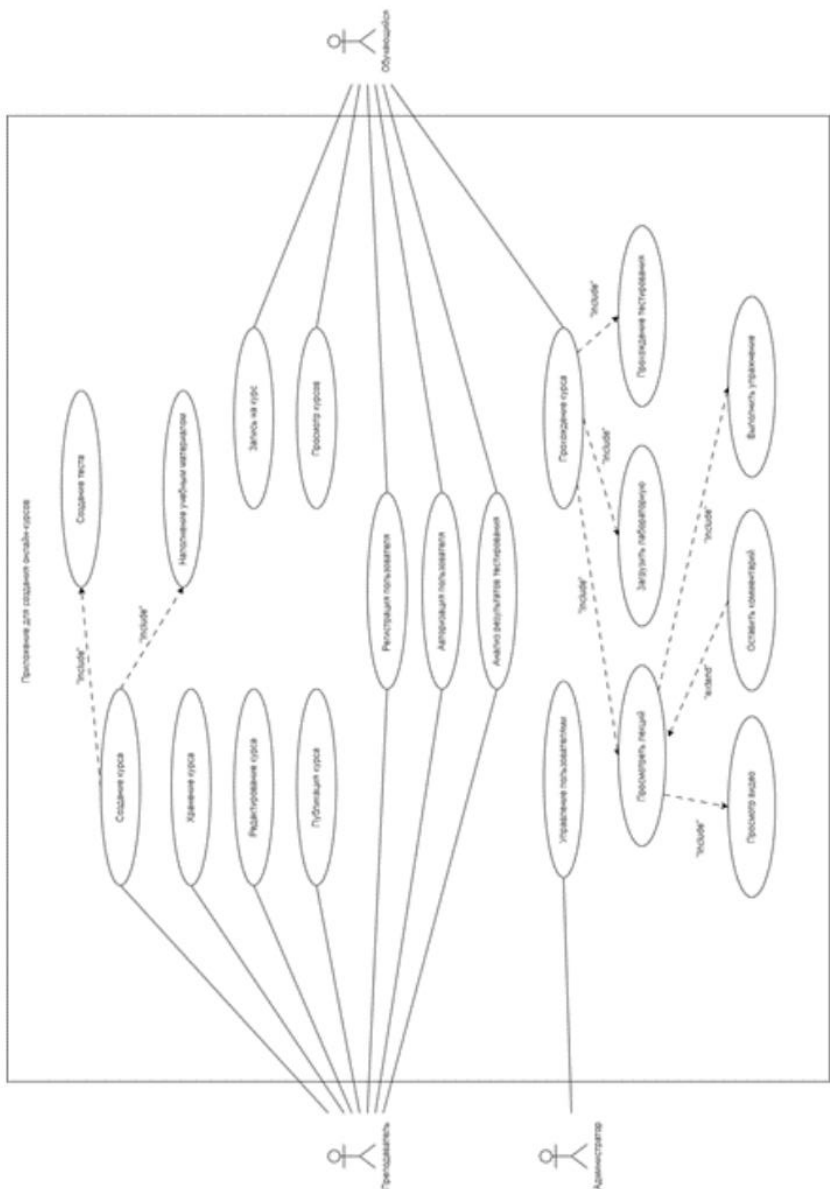


ПГУ 1.090501.10.002 Д18									
Приложение для создания онлайн-курсов для обучения специалистов.									
Диаграмма компонентов									
ФЕТ каф. ВТ гр. 18В031									

Приложение Б.4 – Диаграмма классов

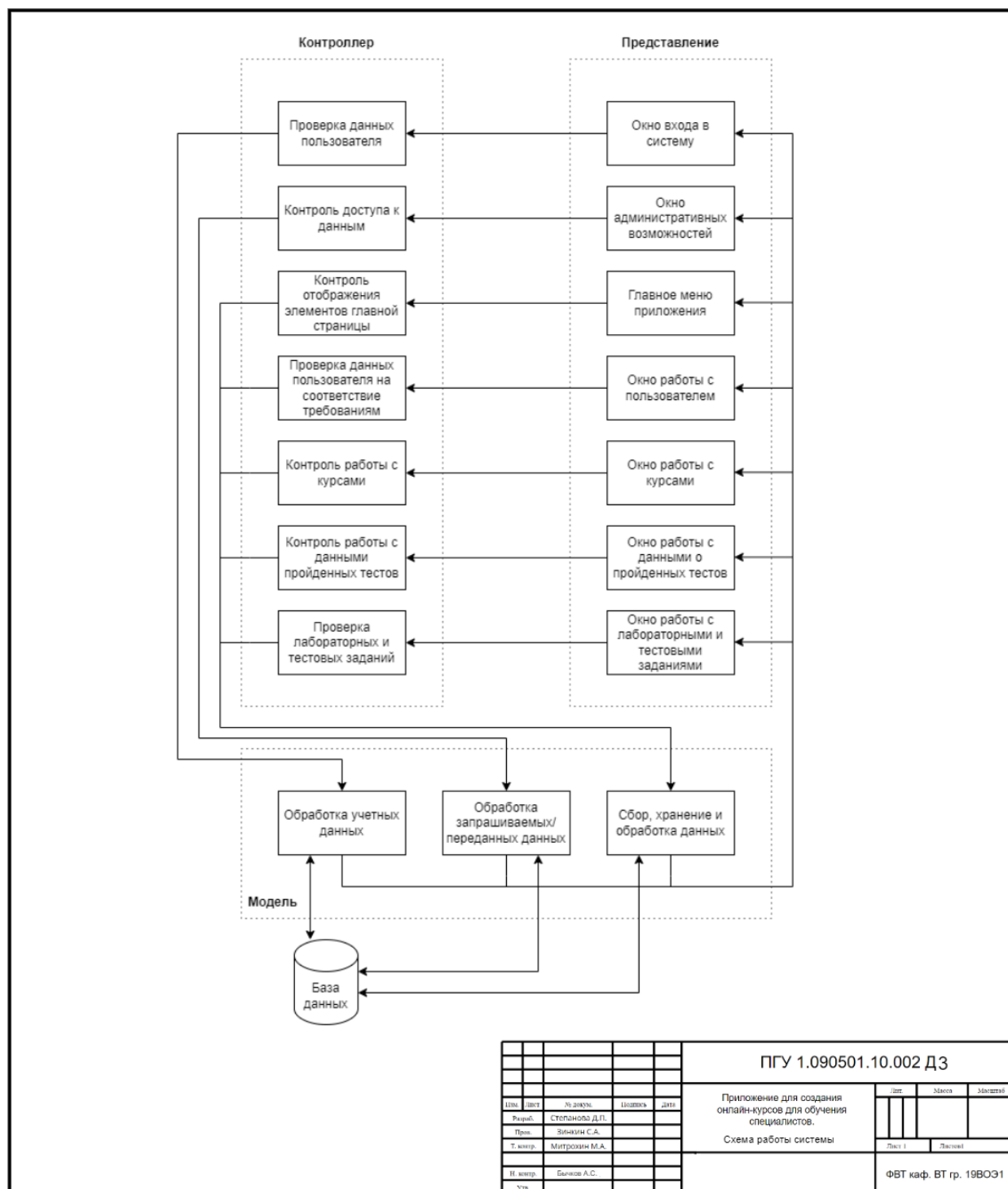


Приложение Б.5 – Диаграмма вариантов использования

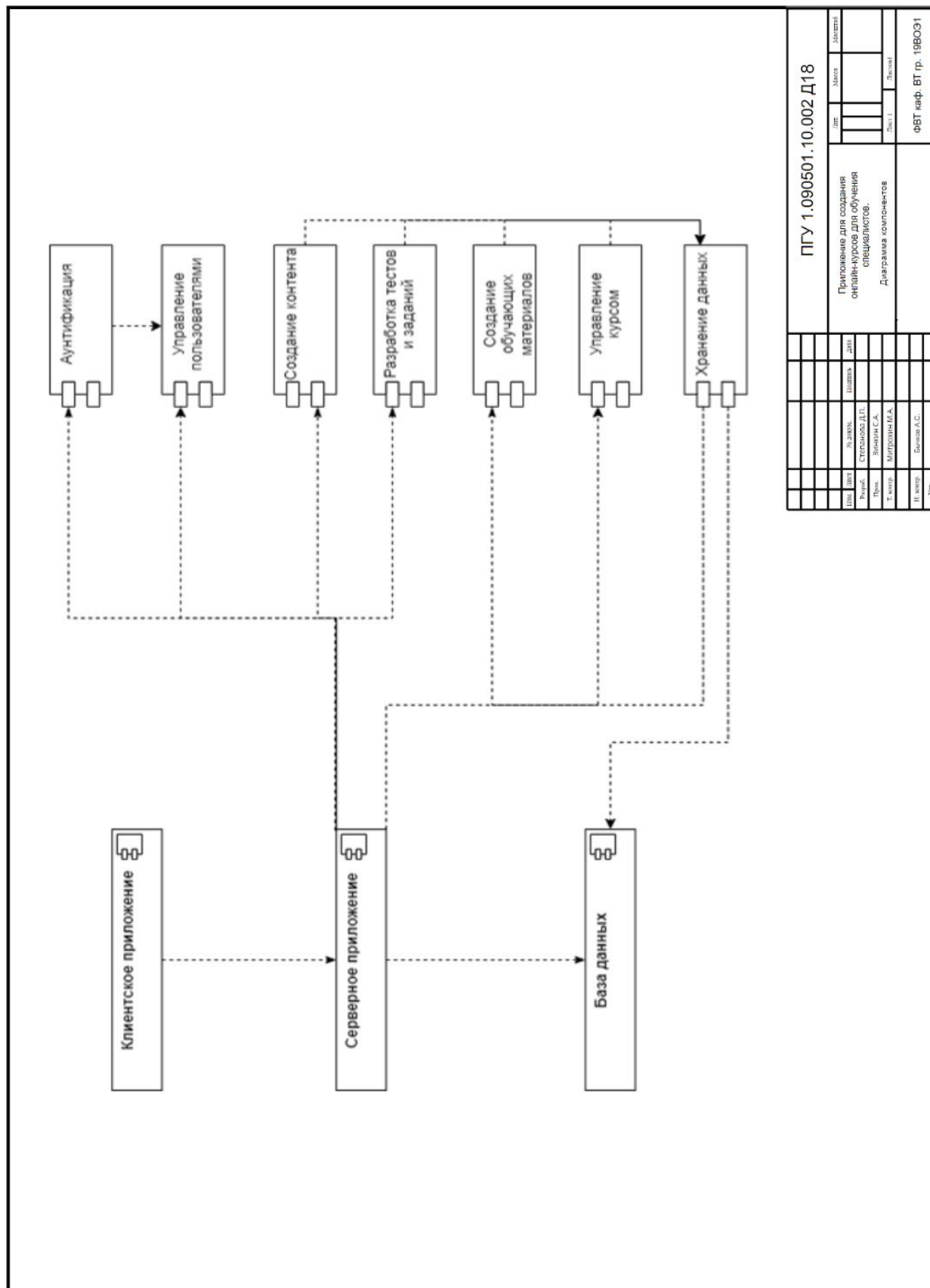


ПГУ 1.090501.10.002 Д13									
Приложение для создания онлайн-курсов для обучения специалистов. Диаграмма вариантов использования									
Изм.	Лист	№ докум.	Подпись	Дата	Изм.	Лист	№ докум.	Подпись	Дата
Лист 1									
ФВТ каф. ВТ гр. 19В031									

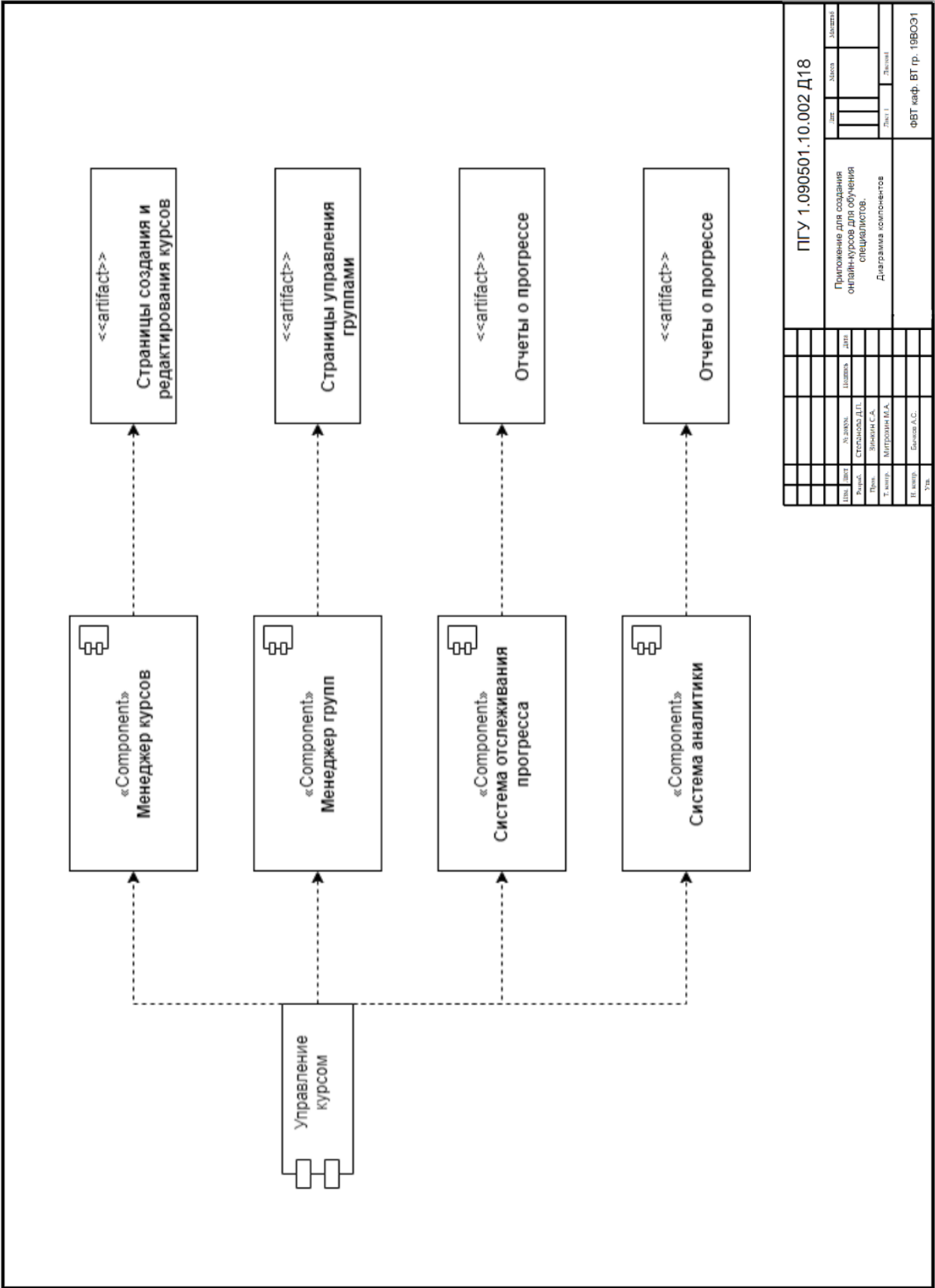
Приложение Б.6 – Схема работы системы



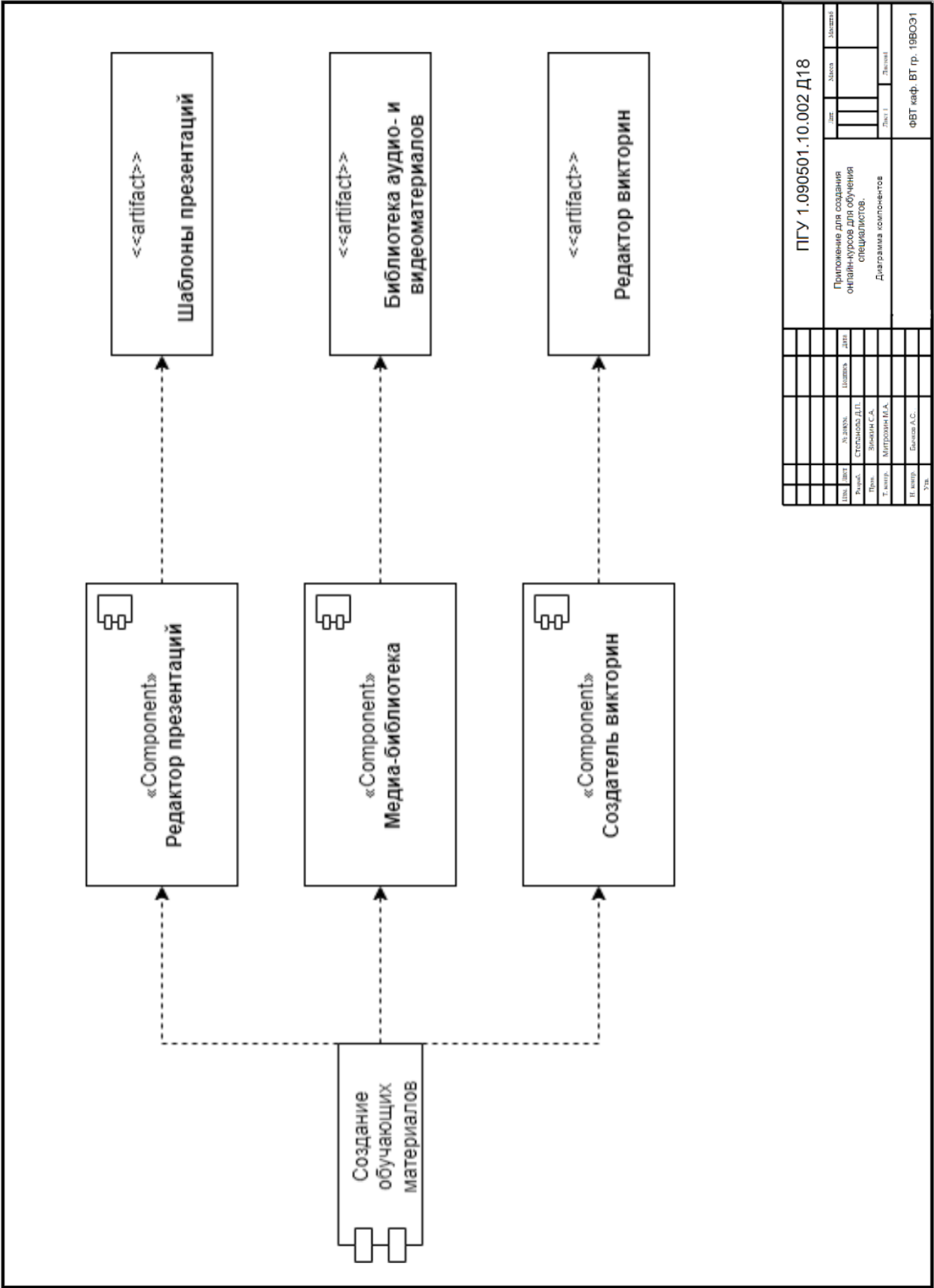
Приложение Б.7 – Диаграмма компонентов



Приложение Б.8 – Диаграмма компонентов

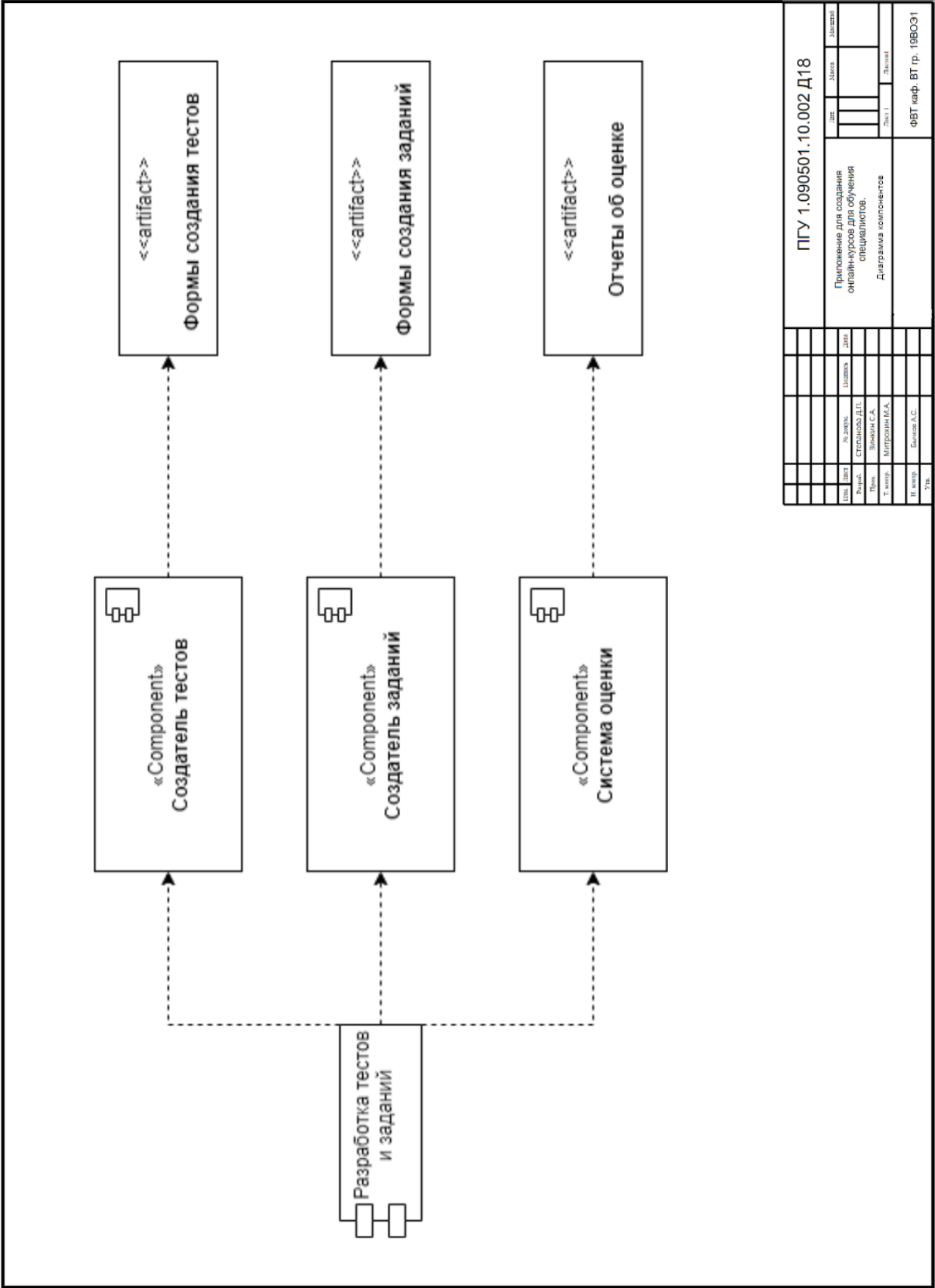


Приложение Б.9 – Диаграмма компонентов

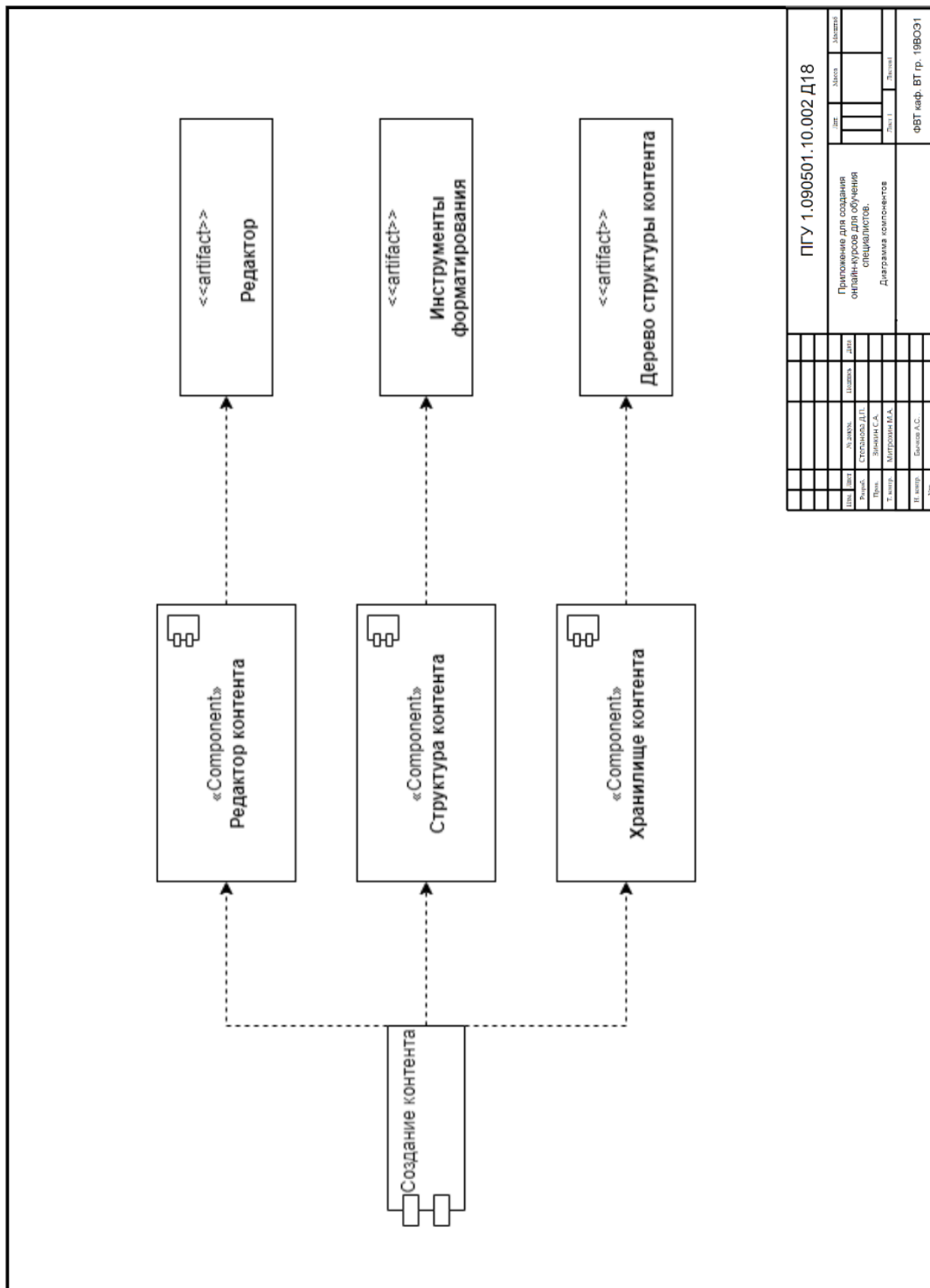


--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--	--

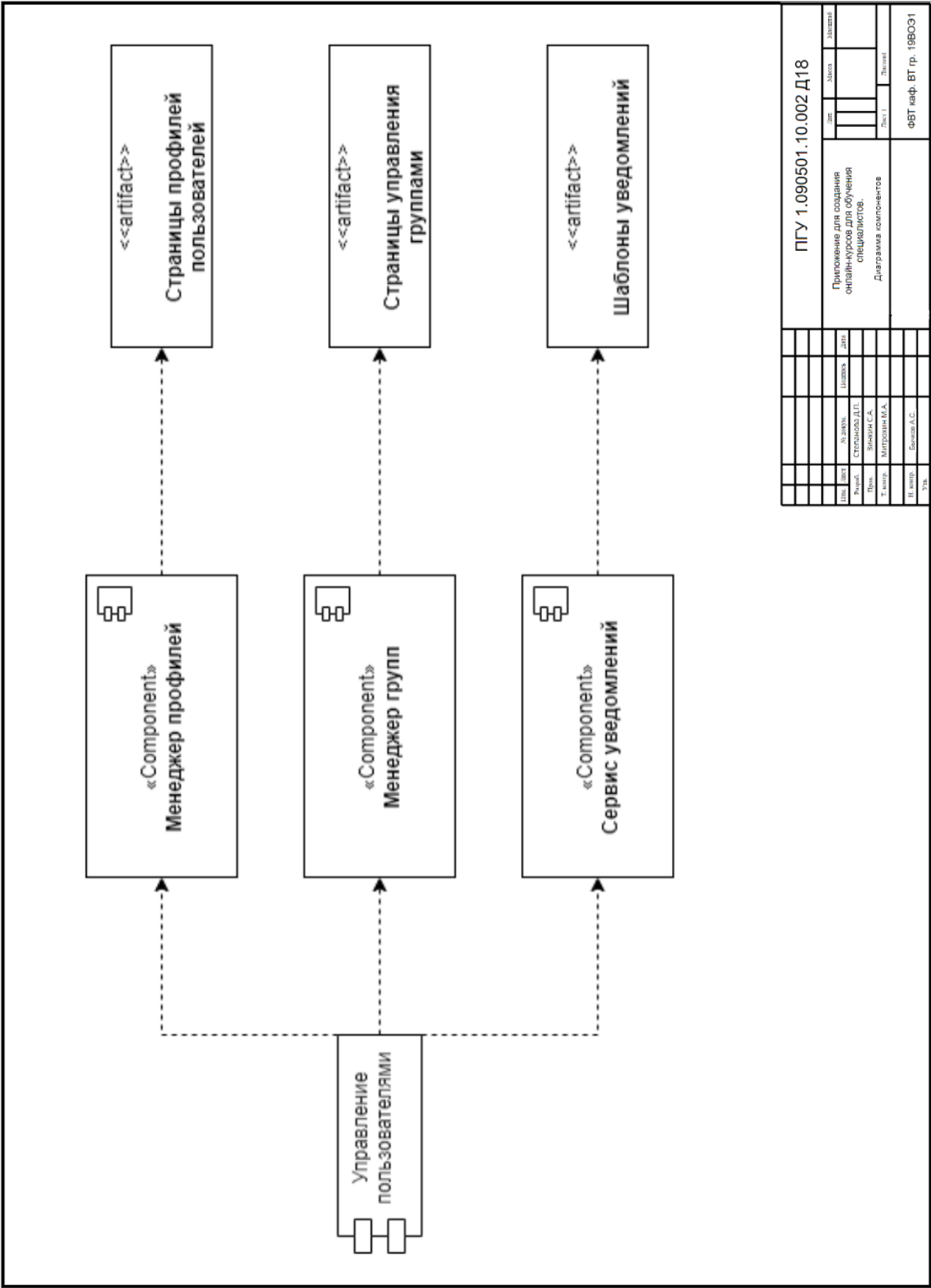
Приложение Б.10 – Диаграмма компонентов



Приложение Б.11 – Диаграмма компонентов



Приложение Б.12 – Диаграмма компонентов



Приложение В

(Обязательное)

Приложение для создания онлайн курсов для обучения специалистов

Листинги программы

					ПГУ 1.090501.10.001 ПЗ	Лист
						104
Изм.	Лист	№ докум.	Подпись	Дата		

Файл корневой на сервере Index.js:

```
require('dotenv').config();
const express = require('express');
const sequelize = require('./db');
const models = require('./models/models');
const cors = require('cors');
const fileUpload = require('express-fileupload');
const PORT = process.env.PORT || 5000;
const router = require('./routes/index');
const errorHandler = require('./middleware/ErrorHandlerMiddleware');
const path = require('path');

const app = express();
app.use(cors())
app.use(express.json())
app.use(express.static(path.resolve(__dirname, 'static')))
app.use(fileUpload({}))
app.use('/api', router)

app.use(errorHandler)

const start = async () =>
{
    try {
        await sequelize.authenticate();
        await sequelize.sync();
        app.listen(PORT, () => console.log(`Listening on ${PORT}`))
    } catch (e)
    {
        console.log('error');
    }
}
```

					ПГУ 1.090501.10.001 ПЗ	Лист
						105
Изм.	Лист	№ докум.	Подпись	Дата		

```

start()

const Sequelize = require('sequelize');

sequelize.authenticate().then(() => {
    console.log('Connection has been established successfully.');
```

```

    .catch(err => {
        console.log('Unable to connect to the database:', err);
    });
}

```

Файл db.js:

```

const {Sequelize} = require('sequelize');

module.exports = new Sequelize(
    process.env.DB_NAME,
    process.env.DB_USERNAME,
    process.env.DB_PASSWORD,
    {
        dialect: 'postgres',
        host: process.env.DB_HOST,
        port: process.env.DB_PORT
    }
)

module.exports = new Sequelize('testdb', 'postgres', '12345', {
    dialect: 'postgres',
    port: '5432',
    host: 'localhost'
});

```

Файл .env:

```
PORT=7000
```

					ПГУ 1.090501.10.001 ПЗ	Лист
						106
Изм.	Лист	№ докум.	Подпись	Дата		

```
DB_NAME=testdb
DB_USER=postgres
DB_PASSWORD=12345
DB_HOST=localhost
DB_PORT=5432
SECRET_KEY=12345
```

Файл courseController.js:

```
const uuid = require(`uuid`)
const path = require(`path`)
const {Course} = require(`../models/models`)
const ApiError = require(`../error/ApiError`)

class courseController{
  async create (req, res, next) {
    try {
      const { name, opis, data, courseId, teacherId} = req.body
      const {img} = req.files
      let filename = uuid.v4() + ".jpg"
      img.mv(path.resolve(__dirname, `..`, `static`, filename))
      const course = await Course.create({name, opis, data,
courseId, teacherId, img: filename})
      return res.json(course)
    } catch (e) {
      next(ApiError.badRequest(e.message))
    }
  }
  async getAll(req, res){
    const { courseId, teacherId} = req.query
    let courses;
    if (!courseId && !teacherId){
      courses = await Course.findAll()
    }
    if (courseId && !teacherId){
```

					ПГУ 1.090501.10.001 ПЗ	Лист
						107
Изм.	Лист	№ докум.	Подпись	Дата		

```

        courses = await Course.findAll({where:{courseId}})
    }
    if (!courseId && teacherId){
        courses = await Course.findAll({where:{teacherId}})
    }
    if (courseId && teacherId){
        courses = await Course.findAll({where:{courseId,
teacherId}})
    }
    return res.json(courses)
}
async getOne(req,res){
    const {id} = req.params
    const course = await Course.findOne(
        {
            where: {id},
            include: [{model: module, as: ``}]
        })
    return res.json(course)
}
}

module.exports = new courseController()

```

Файл moduleController.js:

```

const {Module} = require(`../models/models`)
const ApiError = require(`../error/ApiError`)

class ModuleController {
    async create(req, res) {
        const {name} = req.body
        const module = await Module.create({name})
        return res.json(module)
    }
}

```

```

    async getAll(req, res) {
        const modules = await Module.findAll()
        return res.json(modules)
    }

    async getOne(req, res) {
        const {id} = req.params
        const course = await Module.findOne(
            {
                where: {id},
                include: [{model: lesson, as: ``}]
            })
        return res.json(module)
    }
}

module.exports = new ModuleController()

```

Файл exerciseController.js:

```

const {Module} = require(`../models/models`)
const ApiError = require(`../error/ApiError`)

class exerciseController {
    async create(req, res) {
        const {name} = req.body
        const exercise = await exercise.create({name})
        return res.json(exercise)
    }

    async getAll(req, res) {
        const exercises = await exercise.findAll()
        return res.json(exercises)
    }
}

```

```

    async getOne(req, res) {
        const {id} = req.params
        const course = await Module.findOne(
            {
                where: {id},
                include: [{model: progress, as: ``}]
            })
        return res.json(exercise)
    }
}

module.exports = new exerciseController()

```

Файл lessonController.js:

```

const {lesson} = require(`../models/models`)
const ApiError = require(`../error/ApiError`)

class LessonController {
    async create(req, res) {
        const {name} = req.body
        const lesson = await lesson.create({name})
        return res.json(lesson)
    }

    async getAll(req, res) {
        const lessons = await lesson.findAll()
        return res.json(lessons)
    }

    async getOne(req, res) {
        const {id} = req.params
        const lesson = await lesson.findOne(
            {
                where: {id},

```

```

        include: [{model: lesson, as: ``}]
    })
    return res.json(lesson)

}

module.exports = new LessonController()

```

Файл userController.js:

```

const bcrypt = require(`bcrypt`)
const {User, Exercise} = require(`../models/models`)
const ApiError = require(`../error/ApiError`);
const jwt = require(`jsonwebtoken`)
const generateJwt = (id, email, role) =>{
    return jwt.sign(
        {id: id, email, role},
        process.env.SECRET_KEY,
        {expiresIn: `24h`}
    )
}

class UserController{
    async registration (req, res, next){
        const {email, password, role} = req.body
        if (!email || !password) {
            return next(ApiError.badRequest(`Некорректный email или password`))
        }
        const candidate = await User.findOne({where:{email}})
        if (candidate) {
            return next(ApiError.badRequest(`Пользователь с таким email уже существует`))
        }
        const hashPassword = await bcrypt.hash(password, 4)

```

```

        const user = await User.create({email, role, password:
hashPassword}))
        const exercise = await Exercise.create({userId: user.id})
        const token = generateJwt(user.id, user.email, user.role)
        return res.json({token})
    }
    async login(req, res, next) {
        const {email, password} = req.body
        const user = await User.findOne({where: {email}})
        if (!user) {
            return next(ApiError.internal(`Пользователь не найден`))
        }
        let comparePassword = bcrypt.compareSync(password,
user.password)
        if (!comparePassword) {
            return next(ApiError.internal(`Указан неверный пароль`))
        }
        const token = generateJwt(user.id, user.email, user.role)
        return res.json({token})
    }
    async check(req, res) {
        const token = generateJwt(req.user.id, req.user.email,
req.user.role)
        return res.json({token})
    }
}
module.exports = new UserController()

```

Файл ApiError.js:

```

class ApiError extends Error {

    constructor(status, message) {

        super();

        this.status = status
    }
}

```

```

        this.message = message

    }

    static badRequest(message) {

        return new ApiError(404, message)

    }

    static internal(message) {

        return new ApiError(500, message)

    }

    static forbidden(message) {

        return new ApiError(403, message)

    }

}

module.exports = ApiError

```

Файл authMiddleware.js:

```

const jwt = require(`jsonwebtoken`)

module.exports = function (req, res, next) {

    if (req.method === "OPTIONS"){

        next ()

    }

    try {

        const token = req.headers.authorization.split(``)[1]

```

```

    if (!token) {

        return res.status(401).json({message: "Не авторизован"})

    }

    const decoded = jwt.verify(token, process.env.SECRET_KEY)

    req.user = decoded

    next()

} catch(e) {

    res.status(401). json({message: "Не авторизован"})

}

}

```

Файл checkRoleMiddleware.js:

```

const jwt = require (`jsonwebtoken`)

module.exports = function(role){

    return function (req, res, next) {

        if (req.method === "OPTIONS") {

            next()

        }

        try {

            const token = req.headers.authorization.split(``)[1]

            if (!token) {

                return res.status(401).json({message: "Не
авторизован"})
            }

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						114
Изм.	Лист	№ докум.	Подпись	Дата		

```

    }

    const decoded = jwt.verify(token, process.env.SECRET_KEY)

    if (decoded.role !== role) {

        return res.status(401).json({message: "Нет доступа"})

    }

    req.user = decoded

    next()

} catch (e) {

    return res.status(401).json({message: "Не авторизован"})

}

}}

```

Файл errorHandlerMiddleware.js:

```

const ApiError = require (`../error/ApiError`);

module.exports = function (err, req, res, next) {

    if (err instanceof ApiError){

        return res.status(err.status).json({message: err.message})

    }

    return res.status(500).json.json({message: "Непредвиденная
ошибка"})

}

```

Файл models.js:

```

const sequelize = require(`../db`)

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						115
Изм.	Лист	№ докум.	Подпись	Дата		

```

const {DataTypes} = require(`sequelize`)

const User = sequelize.define(`user`, {

  id: {type: DataTypes.INTEGER, primaryKey: true, autoIncrement:
true},

  name: {type: DataTypes.STRING, allowNull: false},

  fam: {type: DataTypes.STRING, allowNull: false},

  email: {type: DataTypes.STRING, unique: true, allowNull: false},

  photo: {type: DataTypes.STRING},

  password: {type: DataTypes.STRING, allowNull: false},

  type : {type: DataTypes.STRING, defaultValue: "USER", allowNull:
false}

})

const Course = sequelize.define(`course`, {

  id: {type: DataTypes.INTEGER, primaryKey: true, autoIncrement:
true},

  name: {type: DataTypes.STRING, allowNull: false, unique: true},

  opis: {type: DataTypes.STRING, allowNull: false},

  data: {type: DataTypes.DATE},

  img: {type: DataTypes.STRING}

//id_teacher: {type: DataTypes.INTEGER, foreignKey: true}

})

```

```

const Module = sequelize.define(`module`, {

    id: {type: DataTypes.INTEGER, primaryKey: true, autoIncrement:
true},

    name: {type: DataTypes.STRING, allowNull: false},

    opis: {type: DataTypes.STRING},

// id_course: {type: DataTypes.INTEGER, foreignKey: true}

})

const Teacher = sequelize.define(`teacher`, {

    id: {type: DataTypes.INTEGER, primaryKey: true, autoIncrement:
true},

})

const Progress = sequelize.define(`progress`, {

    id: {type: DataTypes.INTEGER, primaryKey: true, autoIncrement:
true},

    data: {type: DataTypes.DATE},

    status: {type: DataTypes.STRING}

})

const Exercise = sequelize.define( `exercise`,{

    id: {type: DataTypes.INTEGER, primaryKey: true, autoIncrement:
true},

```

```

    name: {type: DataTypes.STRING, allowNull: false},

    opis: {type: DataTypes.STRING},

    type: {type: DataTypes.STRING, defaultValue: "EXERCISE",
allowNull: true},

    text: {type:DataTypes.STRING},

//id_lesson: {type: DataTypes.INTEGER, foreignKey: true}

})

const Lesson = sequelize.define(`lesson`, {

    id: {type: DataTypes.INTEGER, primaryKey: true, autoIncrement:
true},

    name: {type: DataTypes.STRING, allowNull: false},

    opis: {type: DataTypes.STRING},

    video: {type: DataTypes.STRING},

    text: {type:DataTypes.STRING},

//id_module: {type: DataTypes.INTEGER, foreignKey: true}

})

Teacher.hasMany(Course)

Course.belongsTo(Teacher)

Course.hasMany(Module)

Module.belongsTo(Course)

```

Module.hasMany (Lesson)

Lesson.belongsTo (Module)

Lesson.hasMany (Exercise)

Exercise.belongsTo (Lesson)

Exercise.hasMany (Progress)

Progress.belongsTo (Exercise)

User.hasMany (Progress)

Progress.belongsTo (User)

User.hasOne (Teacher)

Teacher.belongsTo (User)

module.exports = {

User,

Course,

Module,

Teacher,

Progress,

					ПГУ 1.090501.10.001 ПЗ	Лист
						119
Изм.	Лист	№ докум.	Подпись	Дата		

```
Exercise,  
  
Lesson  
  
}
```

Файл courseRouter.js:

```
const Router = require (`express`)  
  
const router = new Router()  
  
const courseController = require('../controllers/courseController')  
  
router.post(`/`, courseController.create)  
  
router.get(`/`, courseController.getAll)  
  
router.get('/:id`, courseController.getOne)  
  
module.exports = router
```

Файл exerciseRouter.js:

```
const Router = require (`express`)  
  
const router = new Router()  
  
const exerciseController =  
require('../controllers/exerciseController')  
  
const checkRole = require('../middleware/checkRoleMiddleware')  
  
router.post(`/`, checkRole(`ADMIN`), exerciseController.create)  
  
router.get(`/`, exerciseController.getAll)
```

```
module.exports = router
```

Файл index.js:

```
const Router = require (`express`)

const router = new Router()

const userRouter = require(`./userRouter`)

const courseRoute = require(`./courseRouter`)

const moduleRouter = require(`./moduleRouter`)

const lessonRouter = require(`./lessonRouter`)

const exerciseRouter = require(`./exerciseRouter`)

const progressRouter = require(`./progressRouter`)

const teacherRouter = require(`./teacherRouter`)


router.use(`/user`, userRouter)

router.use(`/course`, courseRoute)

router.use(`/module`, moduleRouter)

router.use(`/lesson`, lessonRouter)

router.use(`/exercise`, exerciseRouter)

router.use(`/progress`, progressRouter)

router.use(`/teacher`, teacherRouter)


module.exports = router
```

Файл lessonRouter.js:

					ПГУ 1.090501.10.001 ПЗ	Лист
						121
Изм.	Лист	№ докум.	Подпись	Дата		

```

const Router = require (`express`)

const router = new Router()

const lessonController = require('../controllers/lessonController')

router.post(`/`, lessonController.create)

router.get(`/`, lessonController.getAll)


module.exports = router

```

Файл moduleRouter.js:

```

const Router = require (`express`)

const router = new Router()

const moduleController = require('../controllers/moduleController')

router.post(`/`, moduleController.create)

router.get(`/`, moduleController.getAll)

router.get('/', moduleController.getOne)


module.exports = router

```

Файл progressRouter.js:

```

const Router = require (`express`)

const router = new Router()

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						122
Изм.	Лист	№ докум.	Подпись	Дата		

```

const progressController =
require('../controllers/progressController')

router.post(`/`, progressController.create)

router.get(`/`, progressController.getAll)

module.exports = router

```

Файл teacherRouter.js:

```

const Router = require (`express`)

const router = new Router()

const teacherController = require('../controllers/teacherController')

router.post(`/`, teacherController.create)

router.get(`/`, teacherController.getAll)

module.exports = router

```

Файл userRouter.js:

```

const Router = require (`express`)
const router = new Router()
const userController = require(`../controllers/userController`)
const authMiddleware = require(`../middleware/authMiddleware`)

router.post(`/registration`, userController.registration)
router.post(`/login`,userController.login)
router.get(`/auth`, authMiddleware, userController.check)

```

```
module.exports = router
```

Файл package.json:

```
{
  "name": "server",
  "version": "1.0.0",
  "main": "index.js",
  "scripts": {
    "dev": "nodemon index.js"
  },
  "keywords": [],
  "author": "Stepanova Daria",
  "license": "ISC",
  "description": "",
  "dependencies": {
    "bcrypt": "^5.1.1",
    "cors": "^2.8.5",
    "dotenv": "^16.4.5",
    "express": "^4.19.2",
    "express-fileupload": "^1.5.0",
    "jsonwebtoken": "^9.0.2",
    "pg": "^8.12.0",
    "pg-hstore": "^2.3.4",
    "sequelize": "^6.37.3",
    "uuid": "^10.0.0"
  },
  "devDependencies": {
    "nodemon": "^3.1.3"
  }
}
```

Файл function.js:

					ПГУ 1.090501.10.001 ПЗ	Лист
						124
Изм.	Лист	№ докум.	Подпись	Дата		

```

import React, { Component } from 'react';
import { connect } from 'react-redux';
import { firebase, helpers } from 'redux-react-firebase';
import classNames from 'classnames';

async function createCourse(courseData) {
  try {
    const response = await fetch('/api/courses', {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json'
      },
      body: JSON.stringify(courseData)
    });

    if (response.ok) {
      const newCourse = await response.json();
      console.log('Course created:', newCourse);
      return newCourse;
    } else {
      throw new Error('Failed to create course');
    }
  } catch (error) {
    console.error('Error creating course:', error);
  }
}

async function updateCourse(courseId, courseData) {
  try {
    const response = await fetch(`/api/courses/${courseId}`, {
      method: 'PUT',
      headers: {
        'Content-Type': 'application/json'
      },
    },

```

```

        body: JSON.stringify(courseData)
    });

    if (response.ok) {
        const updatedCourse = await response.json();
        console.log('Course updated:', updatedCourse);
        return updatedCourse;
    } else {
        throw new Error('Failed to update course');
    }
} catch (error) {
    console.error('Error updating course:', error);
}

}

async function deleteCourse(courseId) {
    try {
        const response = await fetch(`/api/courses/${courseId}`, {
            method: 'DELETE'
        });

        if (response.ok) {
            console.log('Course deleted:', courseId);
        } else {
            throw new Error('Failed to delete course');
        }
    } catch (error) {
        console.error('Error deleting course:', error);
    }
}

async function loadCourse(courseId) {
    try {

```

```

const response = await fetch(`/api/courses/${courseId}`);

if (response.ok) {
  const courseData = await response.json();
  console.log('Course loaded:', courseData);
  return courseData;
} else {
  throw new Error('Failed to load course');
}
} catch (error) {
  console.error('Error loading course:', error);
}
}

async function enrollInCourse(courseId) {
  try {
    const response = await fetch(`/api/courses/${courseId}/enroll`, {
      method: 'POST'
    });

    if (response.ok) {
      console.log('User enrolled in course:', courseId);
    } else {
      throw new Error('Failed to enroll in course');
    }
  } catch (error) {
    console.error('Error enrolling in course:', error);
  }
}

async function fetchCourseDetails(courseId) {
  try {
    const response = await fetch(`/api/courses/${courseId}`);

```

```

    if (response.ok) {
        const courseData = await response.json();
        console.log('Course details fetched:', courseData);
        return courseData;
    } else {
        throw new Error('Failed to fetch course details');
    }
} catch (error) {
    console.error('Error fetching course details:', error);
}

}

async function submitAssignment(courseId, assignmentId,
submissionData) {
    try {
        const response = await
fetch(`/api/courses/${courseId}/assignments/${assignmentId}/submit`, {
    method: 'POST',
    headers: {
        'Content-Type': 'application/json'
    },
    body: JSON.stringify(submissionData)
});

    if (response.ok) {
        console.log('Assignment submitted:', assignmentId);
    } else {
        throw new Error('Failed to submit assignment');
    }
} catch (error) {
    console.error('Error submitting assignment:', error);
}

}

```

```

async function fetchCourseProgress(courseId) {
  try {
    const response = await fetch(`/api/courses/${courseId}/progress`);

    if (response.ok) {
      const progressData = await response.json();
      console.log('Course progress fetched:', progressData);
      return progressData;
    } else {
      throw new Error('Failed to fetch course progress');
    }
  } catch (error) {
    console.error('Error fetching course progress:', error);
  }
}

async function submitCourseFeedback(courseId, feedbackData) {
  try {
    const response = await fetch(`/api/courses/${courseId}/feedback`,
    {
      method: 'POST',
      headers: {
        'Content-Type': 'application/json'
      },
      body: JSON.stringify(feedbackData)
    });
  }
}

```

Файл web.js:

```

require('dotenv').config();

const Path = require('path');

const HtmlWebpackPlugin = require('html-webpack-plugin');

```

```

const Webpack = require('webpack');

const ExtractTextPlugin = require('extract-text-webpack-plugin');

const CopyWebpackPlugin = require('copy-webpack-plugin');

const WebpackVersionFilePlugin = require('webpack-version-file-
plugin');

module.exports = (options) => {

  const ExtractSASS = new
ExtractTextPlugin(`styles/${options.cssFileName}`);

  const VENDOR_LIBS = [

    'history',

    'jquery',

    'lodash',

    'react',

    'react-dom',

    'react-ga',

    'react-helmet',

    'react-redux',

    'react-router',

    'react-router-redux',

    'redux',

    'redux-thunk'

  ];

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						130
Изм.	Лист	№ докум.	Подпись	Дата		

```

const webpackConfig = {

  devtool: options.devtool,

  entry: {

    app: Path.resolve(__dirname, '../src/app/index'),

    vendor: VENDOR_LIBS

  },

  output: {

    path: Path.resolve(__dirname, '../dist'),

    publicPath: process.env.HOSTING_URL || '/',

    filename: '[name].js'

  },

  resolve: {

    extensions: ['.js', '.jsx']

  },

  module: {

    rules: [

      {

        test: /\.jsx?$/,

        use: 'eslint-loader',

        enforce: 'pre',

        include: Path.resolve(__dirname, '../src'),

```

```

        exclude: /node_modules/

    }, {

        test: /\.jsx?$/,

        include: Path.resolve(__dirname, '../src/app'),

        use: 'babel-loader',

        exclude: /node_modules/

    }, {

        test: /\.css$/,

        use: [

            'style-loader', 'css-loader'

        ],

        exclude: /node_modules/

    }, {

        test: /\.svg$/,

        exclude: /node_modules/,

        use: `svg-sprite-loader?${JSON.stringify({ name:
'[name]_[hash]', prefixize: true })}`

    }

],

plugins: [

    new WebpackVersionFilePlugin({

        packageFile: Path.join(__dirname, '../package.json'),

        template: Path.join(__dirname, '../version.ejs'),

```

```

        outputFile: Path.join(__dirname, '../static/version.json')

    )),

    new Webpack.DefinePlugin({

        'process.env': {

            NODE_ENV: JSON.stringify(options.isProduction

                ? 'production'

                : 'development'),

            }

    )),

    new Webpack.optimize.CommonsChunkPlugin({

        names: ['vendor', 'manifest']

    })

]

};

if (options.isProduction) {

    webpackConfig.output.filename = 'scripts/[name].[chunkhash].js';

    webpackConfig.plugins.push(new HtmlWebpackPlugin({

        path: process.env.HOSTING_URL

            ? process.env.HOSTING_URL

            : '/',
    
```

```

        template: Path.resolve(__dirname, '../src/index.html')

    )), new CopyWebpackPlugin([

        {

            from: 'static',

            to: 'static'

        }

    ]), ExtractSASS);

webpackConfig.module.rules.push({

    test: /\.scss$/,

    use: ExtractSASS.extract(['css-loader?importLoaders=1',
'postcss-loader?config=webpack/postcss.config.js', 'sass-loader'])

});

webpackConfig.module.rules.push({

    test: /\.?(jpe?g|png|gif)$/i,

    use: [

        {

            loader: 'file-loader',

            query: {

                limit: 40000,

                name: process.env.HOSTING_URL

```

```

        ?
        `${process.env.HOSTING_URL}/static/img/[name].[hash].[ext]`

        : 'static/img/[name].[hash].[ext]'

    }

},

'image-webpack-loader'

]

});

} else {

    webpackConfig.plugins.push(new HtmlWebpackPlugin({

        path: '',

        template: Path.resolve(__dirname, '../src/index.html')

    }), new Webpack.HotModuleReplacementPlugin());

    webpackConfig.module.rules.push({

        test: /\.?(jpe?g|png|gif)$/i,

        use: ['file-loader', 'image-webpack-loader']

    });

    webpackConfig.module.rules.push({

        test: /\.scss$/,

        use: ['style-loader', 'css-loader?importLoaders=1', 'postcss-loader?config=webpack/postcss.config.js', 'sass-loader']
    });

```

```

    });

    webpackConfig.devServer = {

        contentBase: Path.resolve(__dirname, '../'),

        hot: true,

        inline: true,

        historyApiFallback: true

    };

}

return webpackConfig;

};

```

Файл auth.js:

```

import React, {useContext, useState} from 'react';

import {Container, Form} from "react-bootstrap";

import Card from "react-bootstrap/Card";

import Button from "react-bootstrap/Button";

import Row from "react-bootstrap/Row";

import {NavLink, useLocation, useHistory} from "react-router-dom";

import {LOGIN_ROUTE, REGISTRATION_ROUTE, SHOP_ROUTE} from
"../utils/consts";

import {login, registration} from "../http/userAPI";

import {observer} from "mobx-react-lite";

import {Context} from "../index";

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						136
Изм.	Лист	№ докум.	Подпись	Дата		

```

const Auth = observer(() => {

    const {user} = useContext(Context)

    const location = useLocation()

    const history = useHistory()

    const isLogin = location.pathname === LOGIN_ROUTE

    const [email, setEmail] = useState('')

    const [password, setPassword] = useState('')


    const click = async () => {

        try {

            let data;

            if (isLogin) {

                data = await login(email, password);

            } else {

                data = await registration(email, password);

            }

            user.setUser(user)

            user.setIsAuth(true)

            history.push(SHOP_ROUTE)

        } catch (e) {

            alert(e.response.data.message)

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						137
Изм.	Лист	№ докум.	Подпись	Дата		

```

    }

}

return (

    <Container

        className="d-flex justify-content-center align-items-
center"

        style={{height: window.innerHeight - 54}}

    >

        <Card style={{width: 600}} className="p-5">

            <h2 className="m-auto">{isLoggedIn ? 'Авторизация' :
"Регистрация"}</h2>

            <Form className="d-flex flex-column">

                <Form.Control

                    className="mt-3"

                    placeholder="Введите ваш email..."

                    value={email}

                    onChange={e => setEmail(e.target.value)}

                />

                <Form.Control

                    className="mt-3"

                    placeholder="Введите ваш пароль..."

```

```

        value={password}

        onChange={e => setPassword(e.target.value)}

        type="password"

    />

    <Row className="d-flex justify-content-between mt-
3 pl-3 pr-3">

        {isLogin ?

            <div>

                Нет аккаунта? <NavLink
to={REGISTRATION_ROUTE}>Зарегистрируйся!</NavLink>

            </div>

            :

            <div>

                Есть аккаунт? <NavLink
to={LOGIN_ROUTE}>Войдите!</NavLink>

            </div>

        }

        <Button

            variant={"outline-success"}

            onClick={click}

        >

            {isLogin ? 'Войти' : 'Регистрация'}

        </Button>

```

</Row>

</Form>

</Card>

</Container>

);

});

export default Auth;

Файл manifest.json:

{

"short_name": "diplom",

"name": "diplom",

"icons": [{

"src": "\\icons\\android-icon-36x36.png",

"sizes": "36x36",

"type": "image/png",

"density": "0.75"

},

{

"src": "\\icons\\android-icon-48x48.png",

					ПГУ 1.090501.10.001 ПЗ	Лист
						140
Изм.	Лист	№ докум.	Подпись	Дата		

```

        "sizes": "48x48",

        "type": "image/png",

        "density": "1.0"

    },

    {

        "src": "\\icons\\android-icon-72x72.png",

        "sizes": "72x72",

        "type": "image/png",

        "density": "1.5"

    },

    {

        "src": "\\icons\\android-icon-96x96.png",

        "sizes": "96x96",

        "type": "image/png",

        "density": "2.0"

    },

    {

        "src": "\\icons\\android-icon-144x144.png",

        "sizes": "144x144",

        "type": "image/png",

        "density": "3.0"

    },

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						141
Изм.	Лист	№ докум.	Подпись	Дата		

```

{
    "src": "\\icons\\android-icon-128x128.png",
    "sizes": "128x128",
    "type": "image\\png",
    "density": "3.0"
},
{
    "src": "\\icons\\android-icon-192x192.png",
    "sizes": "192x192",
    "type": "image\\png",
    "density": "4.0"
}
],
"start_url": "/",
"display": "standalone",
"orientation": "portrait"
}

```

Файл .css:

```

.btn {

    color: $white;

    border-width: 1px;

    border-style: solid;

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						142
Изм.	Лист	№ докум.	Подпись	Дата		

```

border-radius: 25px;

padding: 3px 13px;

@include scale-anim();

&:focus {

    outline: none !important;

}

&.btn-xs {

    padding: 2px 10px;

    font-size: 12px;

    font-weight: lighter;

    line-height: 1.1;

}

&.btn-primary {

    background-color: $green;

    border-color: $green;

    a {

        color: $white;

        text-decoration: none;

```

```

    }

}

&.btn-cancel {

    background-color: $red;

border-color: $red;

}

&.btn-outline {

    color: $grey;

    border-color: $grey;

}

&.disabled {

    opacity: .3;

}

}

.go-back {

    position: absolute;

    left: 5px;

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						144
Изм.	Лист	№ докум.	Подпись	Дата		

```

        .icon {

            fill: $white;

        }

    }

    .close {

        width: 16px;

        height: 20px;

        display: inline-block;

        position: absolute;

        right: 15px;

        top: 15px;

        z-index: 1;

    }

    $settings: ' 0.3s forwards';

    .fade-in {

        animation: fade-in .5s forwards;

        @keyframes fade-in {

            0% { opacity: 0; visibility: hidden; }

            1% { opacity: 0; visibility: visible; }

            100% { opacity: .9; visibility: visible; }

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						145
Изм.	Лист	№ докум.	Подпись	Дата		

```

    }

}

.fade-out {

    animation: fade-out .5s forwards;


    @keyframes fade-out {

        0% { opacity: .9; visibility: visible; }

        99% { opacity: 0; visibility: visible; }

        100% { opacity: 0; visibility: hidden; }

    }

}

.flyout {

    transform: translateX(-100%);


    &.opened { animation: slide-in-left + $settings; }

    &.closed { animation: slide-out-left + $settings; }


    @keyframes slide-in-left {

        0% { transform: translateX(-100%) }

        100% { transform: translateX(0%) }

```

```

    }

    @keyframes slide-out-left {

        0% { transform: translateX(0%) }

        100% { transform: translateX(-100%) }

    }

}

.push-menu, .flyout-right {

    transform: translateX(100%);

    &.opened { animation: slide-in-right + $settings; }

    &.closed { animation: slide-out-right + $settings; }

    @keyframes slide-in-right {

        0% { transform: translateX(100%) }

        100% { transform: translateX(0%) }

    }

    @keyframes slide-out-right {

        0% { transform: translateX(0%) }

        100% { transform: translateX(100%) }

    }

}

```

```

    }

}

.pushed-container {

    &.opened { animation: resize-in + $settings; }

    &.closed { animation: resize-out + $settings; }

    @keyframes resize-in {

        0% { max-width: 100%; }

        100% { max-width: 75%; }

    }

    @keyframes resize-out {

        0% { max-width: 75%; }

        100% { max-width: 100%; }

    }

}

.dropdown-panel {

    animation: slide-down + $settings;

    &.open {

```

```

        animation: slide-up + $settings;

    }

    @keyframes slide-up {

        0% { transform: translateY(100%) }

        100% { transform: translateY(0%) }

    }

    @keyframes slide-down {

        0% { transform: translateY(0%) }

        100% { transform: translateY(100%) }

    }

}

.cards-heading {

    text-align: center;

    text-transform: uppercase;

    font-size: 22px;

    font-weight: lighter;

    letter-spacing: 3px;

    margin-bottom: 30px;

}

```

```

.cards {

    text-align: center;

    margin-bottom: 50px;

    margin-left: auto;

    margin-right: auto;


.loader-small {

    display: block;

    margin: 0 auto;

}

}

.cards-list {

    list-style-type: none;

    width: fit-content;

    padding: 0 20px;


@include mobile() {

    padding: 0;

}

}

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						150
Изм.	Лист	№ докум.	Подпись	Дата		

```

.card {

    display: inline-block;

    min-width: 250px;

    max-width: 23%;

    vertical-align: top;

    background-color: white;

    text-align: left;

    margin: 10px;

    @include desktop() {

        @include scale-anim(1.05);

    }

    @media (max-width: 600px) {

        width: 100%;

        max-width: 100%;

        margin: 0 0 20px 0;

    }

    &:last-child {

        margin-right: 0;

    }

```

```

.card-thumb {

    width: 100%;

    text-align: center;

    height: 150px;

    border-top-left-radius: 5px;

    border-top-right-radius: 5px;

    margin-bottom: 10px;

    display: table;

    font-size: 70px;

    color: $white;

    background-color: #6ad0dc;


    @media (max-width: 600px) {

        border-top-left-radius: 0;

        border-top-right-radius: 0;

    }

    span {

        display: table-cell;

        vertical-align: middle;

    }

```

```

        &.card-image {

            background-size: cover;

            background-repeat: no-repeat;

        }

    }

```

```

.card-wrapper {

    padding: 0 10px;

    margin-bottom: 15px;

}

```

```

.card-title {

    font-size: 18px;

    line-height: 26px;

    font-weight: lighter;

    width: 100%;

    white-space: nowrap;

    overflow: hidden;

    text-overflow: ellipsis;

}

.course-icon {

```

```

        font-size: 12px;

        border: 1px solid;

        border-radius: 5px;

        vertical-align: middle;

        margin-right: 5px;

        width: 24px;

        line-height: 20px;

        padding: 2px;

        display: inline-block;

        text-align: center;

        height: 24px;

    }

}

```

```

.card-meta {

    font-size: 12px;

    .icon {

        margin-right: 5px;

        margin-bottom: 5px;

    }
}

```

```

        .card-date {

            font-weight: bold;

            margin-right: 10px;

        }

    }

    $font-size: 14px;

    $line-height: 1.5;

    $lines-to-show: 5;

    .card-content {

        display: block; /* Fallback for non-webkit */

        display: -webkit-box;

        height: $font-size*$line-height*$lines-to-show; /* Fallback
for non-webkit */

        margin: 0 auto;

        font-size: $font-size;

        line-height: $line-height;

        -webkit-line-clamp: $lines-to-show;

        -webkit-box-orient: vertical;

        overflow: hidden;

        text-overflow: ellipsis;

    }

```

```

.card-actions {

    margin: 10px 0;

    .btn-secondary {

        color: $blue;

    }

}

.card-info {

    margin-top: 10px;

    .card-enrolled {

        font-size: 14px;

        padding-top: 8px;

        display: inline-block;

    }

    .card-price {

        float: right;

        font-size: 22px;

        color: $dark-grey;

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						156
Изм.	Лист	№ докум.	Подпись	Дата		

```

        font-weight: bold;

    }

}

a, a:hover, a:active, a:visited, a:focus {

    color: $blue;

}

textarea, input, button {

    border: none;

    background: none;

    &:focus {

        outline: none !important;

    }

}

.input-field {

    border-bottom: 1px solid $white;

}

input:-webkit-autofill, textarea:-webkit-autofill, select:-webkit-
autofill {

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						157
Изм.	Лист	№ докум.	Подпись	Дата		

```

        -webkit-box-shadow: 0 0 0 1000px white inset !important;
    }

// Tables

table {

    width: 100%;

    th {

        background-color: $blue;

        color: $white;

    }

    th, td {

        border-bottom: 1px solid $light-grey;

        padding: 5px;

    }

    tr:hover { background-color: $very-light-gray }

    input { margin-right: 5px }

}

```

```

// Horizontal nav

.horizontal-nav {

    list-style-type: none;

    width: fit-content;

    margin: 20px auto 0 auto;

    padding: 0;

    .horizontal-nav-item {

        display: inline-block;

        padding: 0 10px;

        &.active {

            font-weight: bold;

        }

    }

}

// Date picker

.react-datepicker__tether-element {

    @include mobile() {

        width: 100%;

        height: 100%;

    }

}

```

```

        position: fixed !important;

        margin-top: 0 !important;

        transform: none !important;
    }
}

.react-datepicker__input-container {

    .react-datepicker__close-icon:after {

        background-color: transparent;

        color: $grey;

        border: 1px solid $grey;

        right: 0;

    }
}

.react-datepicker {

    border-radius: 0;

    font-family: inherit;

    cursor: pointer;

    @include mobile() {

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						160
Изм.	Лист	№ докум.	Подпись	Дата		

```

        width: 100%;

        height: 100%;

        font-size: 16px;

    }

    .datepicker__day {

        @include mobile() {

            width: 35px;

            line-height: 35px;

        }

    }

    .datepicker__day--selected,

    .datepicker__day:hover {

        border-radius: 0;

    }

    .react-datepicker__navigation {

        z-index: 1;

        @include mobile() {

            border-width: 10px;

```

```

    }

}

.react-datepicker__current-month {

    @include mobile() {

        font-size: 18px;

    }

}

}

// Select2

.page .select2-container--default.select2-container--focus .select2-
selection--multiple {

    border-color: $grey;

}

// Tooltip

.tooltip {

    position: relative;

    display: inline-block;

    &:hover .tooltip-text,

    &.active .tooltip-text {

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						162
Изм.	Лист	№ докум.	Подпись	Дата		

```

        visibility: visible;

        opacity: 1;
    }

    .tooltip-text {

        width: 120px;

        visibility: hidden;

        background-color: rgba(255,255,255,.7);

        color: $dark-grey;

        font-size: 13px;

        text-align: center;

        border-radius: 6px;

        padding: 5px 3px;

        position: absolute;

        opacity: 0;

        transition: opacity 1s;

        z-index: 9;

        &.inverted {

            background-color: rgba(0,0,0,.9);

            color: $white;

        }
    }

```

```

&.right {

    top: -25px;

    left: 105%;

    &::after {

        top: 50%;

        right: 100%; /* To the left of the tooltip */

        border-color: transparent rgba(255,255,255,.7)
transparent transparent;

    }

    &.inverted {

        &::after { border-color: transparent rgba(0,0,0,.9)
transparent transparent; }

    }

}

&.left {

    top: -25px;

    right: 105%;

    &::after {

        top: 50%;

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						164
Изм.	Лист	№ докум.	Подпись	Дата		

```

        left: 100%; /* To the right of the tooltip */

        border-color: rgba(255,255,255,.7) transparent
transparent transparent;

    }

    &.inverted {

        &::after { border-color: rgba(0,0,0,.9) transparent
transparent transparent; }

    }

    }

    &.top {

        bottom: 125%;

        left: 50%;

        margin-left: -60px;

        &::after {

            top: 100%;

            left: 50%;

            border-color: rgba(255,255,255,.7) transparent
transparent transparent;

        }

        &.inverted {

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						165
Изм.	Лист	№ докум.	Подпись	Дата		

```

        &::after { border-color: rgba(0,0,0,.9) transparent
transparent transparent; }

    }

}

&.bottom {

    top: 100%;

    left: 50%;

    margin-left: -60px; /* Use half of the width (120/2 =
60), to center the tooltip */

    &::after {

        bottom: 100%;

        left: 50%;

        border-color: transparent transparent
rgba(255,255,255,.7) transparent;

    }

    &.inverted {

        &::after { border-color: transparent transparent
rgba(0,0,0,.9) transparent; }

    }

}

span {

```

```

        font-size: 11px;

        display: block;
    }

    &::after {

        content: '';

        position: absolute;

        margin-left: -5px;

        border-width: 5px;

        border-style: solid;

    }

}

}

}

.select2-container {

    box-sizing: border-box;

    display: inline-block;

    margin: 0;

    position: relative;

    vertical-align: middle; }

.select2-container .select2-selection--single {

    box-sizing: border-box;

```

```

    cursor: pointer;

    display: block;

    height: 28px;

    user-select: none;

    -webkit-user-select: none; }

.select2-container .select2-selection--single .select2-
selection__rendered {

    display: block;

    padding-left: 8px;

    padding-right: 20px;

    overflow: hidden;

    text-overflow: ellipsis;

    white-space: nowrap; }

.select2-container[dir="rtl"] .select2-selection--single .select2-
selection__rendered {

    padding-right: 8px;

    padding-left: 20px; }

.select2-container .select2-selection--multiple {

    box-sizing: border-box;

    cursor: pointer;

    display: block;

    min-height: 32px;

    user-select: none;

```

```

    -webkit-user-select: none; }

    .select2-container .select2-selection--multiple .select2-
selection__rendered {

    display: inline-block;

    overflow: hidden;

    padding-left: 8px;

    text-overflow: ellipsis;

    white-space: nowrap; }

.select2-container .select2-search--inline {

    float: left; }

.select2-container .select2-search--inline .select2-search__field
{

    box-sizing: border-box;

    border: none;

    font-size: 100%;

    margin-top: 5px; }

.select2-container .select2-search--inline .select2-
search__field::-webkit-search-cancel-button {

    -webkit-appearance: none; }

.select2-dropdown {

    background-color: white;

    border: 1px solid #aaa;

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						169
Изм.	Лист	№ докум.	Подпись	Дата		

```
border-radius: 4px;
```

```
box-sizing: border-box;
```

```
display: block;
```

```
position: absolute;
```

```
left: -100000px;
```

```
width: 100%;
```

```
z-index: 1051; }
```

```
.select2-results {
```

```
    display: block; }
```

```
.select2-results__options {
```

```
    list-style: none;
```

```
    margin: 0;
```

```
    padding: 0; }
```

```
.select2-results__option {
```

```
    padding: 6px;
```

```
    user-select: none;
```

```
    -webkit-user-select: none; }
```

```
.select2-results__option[aria-selected] {
```

```
    cursor: pointer; }
```

					ПГУ 1.090501.10.001 ПЗ	Лист
						170
Изм.	Лист	№ докум.	Подпись	Дата		

```

.select2-container--open .select2-dropdown {

    left: 0; }

.select2-container--open .select2-dropdown--above {

    border-bottom: none;

    border-bottom-left-radius: 0;

    border-bottom-right-radius: 0; }

.select2-container--open .select2-dropdown--below {

    border-top: none;

    border-top-left-radius: 0;

    border-top-right-radius: 0; }

.select2-search--dropdown {

    display: block;

    padding: 4px; }

.select2-search--dropdown .select2-search__field {

    padding: 4px;

    width: 100%;

    box-sizing: border-box; }

.select2-search--dropdown .select2-search__field::-webkit-search-
cancel-button {

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						171
Изм.	Лист	№ докум.	Подпись	Дата		

```

        -webkit-appearance: none; }

.select2-search--dropdown.select2-search--hide {

    display: none; }

.select2-close-mask {

    border: 0;

    margin: 0;

    padding: 0;

    display: block;

    position: fixed;

    left: 0;

    top: 0;

    min-height: 100%;

    min-width: 100%;

    height: auto;

    width: auto;

    opacity: 0;

    z-index: 99;

    background-color: #fff;

    filter: alpha(opacity=0); }

.select2-hidden-accessible {

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						172
Изм.	Лист	№ докум.	Подпись	Дата		

```

border: 0 !important;

clip: rect(0 0 0 0) !important;

height: 1px !important;

margin: -1px !important;

overflow: hidden !important;

padding: 0 !important;

position: absolute !important;

width: 1px !important; }

```

```

.select2-container--default .select2-selection--single {

    background-color: #fff;

    border: 1px solid #aaa;

    border-radius: 4px; }

.select2-container--default .select2-selection--single .select2-
selection__rendered {

    color: #444;

    line-height: 28px; }

.select2-container--default .select2-selection--single .select2-
selection__clear {

    cursor: pointer;

    float: right;

    font-weight: bold; }

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						173
Изм.	Лист	№ докум.	Подпись	Дата		

```

    .select2-container--default .select2-selection--single .select2-
selection__placeholder {

    color: #999; }

    .select2-container--default .select2-selection--single .select2-
selection__arrow {

    height: 26px;

    position: absolute;

    top: 1px;

    right: 1px;

    width: 20px; }

    .select2-container--default .select2-selection--single .select2-
selection__arrow b {

    border-color: #888 transparent transparent transparent;

    border-style: solid;

    border-width: 5px 4px 0 4px;

    height: 0;

    left: 50%;

    margin-left: -4px;

    margin-top: -2px;

    position: absolute;

    top: 50%;

    width: 0; }

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						174
Изм.	Лист	№ докум.	Подпись	Дата		

```

.select2-container--default[dir="rtl"] .select2-selection--single
.select2-selection__clear {

    float: left; }

.select2-container--default[dir="rtl"] .select2-selection--single
.select2-selection__arrow {

    left: 1px;

    right: auto; }

.select2-container--default.select2-container--disabled .select2-
selection--single {

    background-color: #eee;

    cursor: default; }

.select2-container--default.select2-container--disabled .select2-
selection--single .select2-selection__clear {

    display: none; }

.select2-container--default.select2-container--open .select2-
selection--single .select2-selection__arrow b {

    border-color: transparent transparent #888 transparent;

    border-width: 0 4px 5px 4px; }

.select2-container--default .select2-selection--multiple {

    background-color: white;

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						175
Изм.	Лист	№ докум.	Подпись	Дата		

```

border: 1px solid #aaa;

border-radius: 4px;

cursor: text; }

.select2-container--default .select2-selection--multiple .select2-
selection__rendered {

    box-sizing: border-box;

    list-style: none;

    margin: 0;

    padding: 0 5px;

    width: 100%; }

.select2-container--default .select2-selection--multiple .select2-
selection__placeholder {

    color: #999;

    margin-top: 5px;

    float: left; }

.select2-container--default .select2-selection--multiple .select2-
selection__clear {

    cursor: pointer;

    float: right;

    font-weight: bold;

    margin-top: 5px;

    margin-right: 10px; }

.select2-container--default .select2-selection--multiple .select2-
selection__choice {

```

```

background-color: #e4e4e4;

border: 1px solid #aaa;

border-radius: 4px;

cursor: default;

float: left;

margin-right: 5px;

margin-top: 5px;

padding: 0 5px; }

.select2-container--default .select2-selection--multiple .select2-
selection__choice__remove {

    color: #999;

    cursor: pointer;

    display: inline-block;

    font-weight: bold;

    margin-right: 2px; }

.select2-container--default .select2-selection--multiple .select2-
selection__choice__remove:hover {

    color: #333; }

.select2-container--default[dir="rtl"] .select2-selection--multiple
.select2-selection__choice, .select2-container--default[dir="rtl"]
.select2-selection--multiple .select2-selection__placeholder,
.select2-container--default[dir="rtl"] .select2-selection--multiple
.select2-search--inline {

    float: right; }

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						177
Изм.	Лист	№ докум.	Подпись	Дата		

```
.select2-container--default[dir="rtl"] .select2-selection--multiple
.select2-selection__choice {
```

```
    margin-left: 5px;
```

```
    margin-right: auto; }
```

```
.select2-container--default[dir="rtl"] .select2-selection--multiple
.select2-selection__choice__remove {
```

```
    margin-left: 2px;
```

```
    margin-right: auto; }
```

```
.select2-container--default.select2-container--focus .select2-
selection--multiple {
```

```
    border: solid black 1px;
```

```
    outline: 0; }
```

```
.select2-container--default.select2-container--disabled .select2-
selection--multiple {
```

```
    background-color: #eee;
```

```
    cursor: default; }
```

```
.select2-container--default.select2-container--disabled .select2-
selection__choice__remove {
```

```
    display: none; }
```

					ПГУ 1.090501.10.001 ПЗ	Лист
						178
Изм.	Лист	№ докум.	Подпись	Дата		

```
.select2-container--default.select2-container--open.select2-container--above .select2-selection--single, .select2-container--default.select2-container--open.select2-container--above .select2-selection--multiple {
```

```
border-top-left-radius: 0;
```

```
border-top-right-radius: 0; }
```

```
.select2-container--default.select2-container--open.select2-container--below .select2-selection--single, .select2-container--default.select2-container--open.select2-container--below .select2-selection--multiple {
```

```
border-bottom-left-radius: 0;
```

```
border-bottom-right-radius: 0; }
```

```
.select2-container--default .select2-search--dropdown .select2-search__field {
```

```
border: 1px solid #aaa; }
```

```
.select2-container--default .select2-search--inline .select2-search__field {
```

```
background: transparent;
```

```
border: none;
```

```
outline: 0;
```

```
box-shadow: none; }
```

					ПГУ 1.090501.10.001 ПЗ	Лист
						179
Изм.	Лист	№ докум.	Подпись	Дата		

```

.select2-container--default .select2-results > .select2-
results__options {

    max-height: 200px;

    overflow-y: auto; }

.select2-container--default .select2-results__option[role=group] {

    padding: 0; }

.select2-container--default .select2-results__option[aria-
disabled=true] {

    color: #999; }

.select2-container--default .select2-results__option[aria-
selected=true] {

    background-color: #ddd; }

.select2-container--default .select2-results__option .select2-
results__option {

    padding-left: 1em; }

    .select2-container--default .select2-results__option .select2-
results__option .select2-results__group {

    padding-left: 0; }

    .select2-container--default .select2-results__option .select2-
results__option .select2-results__option {

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						180
Изм.	Лист	№ докум.	Подпись	Дата		

```

margin-left: -1em;

padding-left: 2em; }

.select2-container--default .select2-results__option .select2-
results__option .select2-results__option .select2-results__option {

margin-left: -2em;

padding-left: 3em; }

.select2-container--default .select2-results__option .select2-
results__option .select2-results__option .select2-results__option
.select2-results__option {

margin-left: -3em;

padding-left: 4em; }

.select2-container--default .select2-results__option .select2-
results__option .select2-results__option .select2-results__option
.select2-results__option .select2-results__option {

margin-left: -4em;

padding-left: 5em; }

.select2-container--default .select2-results__option
.select2-results__option .select2-results__option .select2-
results__option .select2-results__option .select2-results__option
.select2-results__option {

margin-left: -5em;

padding-left: 6em; }

.select2-container--default .select2-results__option--
highlighted[aria-selected] {

background-color: #5897fb;

```

```

color: white; }

.select2-container--default .select2-results__group {

    cursor: default;

    display: block;

    padding: 6px; }

.select2-container--classic .select2-selection--single {

    background-color: #f7f7f7;

    border: 1px solid #aaa;

    border-radius: 4px;

    outline: 0;

    background-image: -webkit-linear-gradient(top, white 50%, #eeeeee 100%);

    background-image: -o-linear-gradient(top, white 50%, #eeeeee 100%);

    background-image: linear-gradient(to bottom, white 50%, #eeeeee 100%);

    background-repeat: repeat-x;

    filter:
progid:DXImageTransform.Microsoft.gradient(startColorstr='#FFFFFF',
endColorstr='#FFEEEE', GradientType=0); }

.select2-container--classic .select2-selection--single:focus {

    border: 1px solid #5897fb; }

```

```

    .select2-container--classic .select2-selection--single .select2-
selection__rendered {

    color: #444;

    line-height: 28px; }

    .select2-container--classic .select2-selection--single .select2-
selection__clear {

    cursor: pointer;

    float: right;

    font-weight: bold;

    margin-right: 10px; }

    .select2-container--classic .select2-selection--single .select2-
selection__placeholder {

    color: #999; }

    .select2-container--classic .select2-selection--single .select2-
selection__arrow {

    background-color: #ddd;

    border: none;

    border-left: 1px solid #aaa;

    border-top-right-radius: 4px;

    border-bottom-right-radius: 4px;

    height: 26px;

    position: absolute;

    top: 1px;

    right: 1px;

```

```

width: 20px;

background-image: -webkit-linear-gradient(top, #eeeeee 50%,
#cccccc 100%);

background-image: -o-linear-gradient(top, #eeeeee 50%, #cccccc
100%);

background-image: linear-gradient(to bottom, #eeeeee 50%, #cccccc
100%);

background-repeat: repeat-x;

filter:
progid:DXImageTransform.Microsoft.gradient(startColorstr='#FFEEEEEE',
endColorstr='#FFCCCCC', GradientType=0); }

.select2-container--classic .select2-selection--single .select2-
selection__arrow b {

border-color: #888 transparent transparent transparent;

border-style: solid;

border-width: 5px 4px 0 4px;

height: 0;

left: 50%;

margin-left: -4px;

margin-top: -2px;

position: absolute;

top: 50%;

width: 0; }

```

```
.select2-container--classic[dir="rtl"] .select2-selection--single
.select2-selection__clear {

    float: left; }
```

```
.select2-container--classic[dir="rtl"] .select2-selection--single
.select2-selection__arrow {

    border: none;

    border-right: 1px solid #aaa;

    border-radius: 0;

    border-top-left-radius: 4px;

    border-bottom-left-radius: 4px;

    left: 1px;

    right: auto; }
```

```
.select2-container--classic.select2-container--open .select2-
selection--single {

    border: 1px solid #5897fb; }
```

```
.select2-container--classic.select2-container--open .select2-
selection--single .select2-selection__arrow {

    background: transparent;

    border: none; }
```

```
.select2-container--classic.select2-container--open .select2-
selection--single .select2-selection__arrow b {

    border-color: transparent transparent #888 transparent;
```

					ПГУ 1.090501.10.001 ПЗ	Лист
						185
Изм.	Лист	№ докум.	Подпись	Дата		

```
border-width: 0 4px 5px 4px; }
```

```
.select2-container--classic.select2-container--open.select2-container--above .select2-selection--single {
```

```
border-top: none;
```

```
border-top-left-radius: 0;
```

```
border-top-right-radius: 0;
```

```
background-image: -webkit-linear-gradient(top, white 0%, #eeeeee 50%);
```

```
background-image: -o-linear-gradient(top, white 0%, #eeeeee 50%);
```

```
background-image: linear-gradient(to bottom, white 0%, #eeeeee 50%);
```

```
background-repeat: repeat-x;
```

```
filter:
```

```
progid:DXImageTransform.Microsoft.gradient(startColorstr='#FFFFFFF', endColorstr='#FFEEEEEE', GradientType=0); }
```

```
.select2-container--classic.select2-container--open.select2-container--below .select2-selection--single {
```

```
border-bottom: none;
```

```
border-bottom-left-radius: 0;
```

```
border-bottom-right-radius: 0;
```

```
background-image: -webkit-linear-gradient(top, #eeeeee 50%, white 100%);
```

```
background-image: -o-linear-gradient(top, #eeeeee 50%, white 100%);
```

					ПГУ 1.090501.10.001 ПЗ	Лист
						186
Изм.	Лист	№ докум.	Подпись	Дата		

```

background-image: linear-gradient(to bottom, #eeeeee 50%, white
100%);

background-repeat: repeat-x;

filter:
progid:DXImageTransform.Microsoft.gradient(startColorstr='#FFEEEEEE',
endColorstr='#FFFFFF', GradientType=0); }

.select2-container--classic .select2-selection--multiple {

background-color: white;

border: 1px solid #aaa;

border-radius: 4px;

cursor: text;

outline: 0; }

.select2-container--classic .select2-selection--multiple:focus {

border: 1px solid #5897fb; }

.select2-container--classic .select2-selection--multiple .select2-
selection__rendered {

list-style: none;

margin: 0;

padding: 0 5px; }

.select2-container--classic .select2-selection--multiple .select2-
selection__clear {

display: none; }

.select2-container--classic .select2-selection--multiple .select2-
selection__choice {

```

```

background-color: #e4e4e4;

border: 1px solid #aaa;

border-radius: 4px;

cursor: default;

float: left;

margin-right: 5px;

margin-top: 5px;

padding: 0 5px; }

.select2-container--classic .select2-selection--multiple .select2-
selection__choice__remove {

    color: #888;

    cursor: pointer;

    display: inline-block;

    font-weight: bold;

    margin-right: 2px; }

.select2-container--classic .select2-selection--multiple .select2-
selection__choice__remove:hover {

    color: #555; }

.select2-container--classic[dir="rtl"] .select2-selection--multiple
.select2-selection__choice {

    float: right; }

```

```
.select2-container--classic[dir="rtl"] .select2-selection--multiple
.select2-selection__choice {

    margin-left: 5px;

    margin-right: auto; }
```

```
.select2-container--classic[dir="rtl"] .select2-selection--multiple
.select2-selection__choice__remove {

    margin-left: 2px;

    margin-right: auto; }
```

```
.select2-container--classic.select2-container--open .select2-
selection--multiple {

    border: 1px solid #5897fb; }
```

```
.select2-container--classic.select2-container--open.select2-container-
-above .select2-selection--multiple {

    border-top: none;

    border-top-left-radius: 0;

    border-top-right-radius: 0; }
```

```
.select2-container--classic.select2-container--open.select2-container-
-below .select2-selection--multiple {

    border-bottom: none;

    border-bottom-left-radius: 0;
```

```
border-bottom-right-radius: 0; }
```

```
.select2-container--classic .select2-search--dropdown .select2-  
search__field {  
  
    border: 1px solid #aaa;  
  
    outline: 0; }
```

```
.select2-container--classic .select2-search--inline .select2-  
search__field {  
  
    outline: 0;  
  
    box-shadow: none; }
```

```
.select2-container--classic .select2-dropdown {  
  
    background-color: white;  
  
    border: 1px solid transparent; }
```

```
.select2-container--classic .select2-dropdown--above {  
  
    border-bottom: none; }
```

```
.select2-container--classic .select2-dropdown--below {  
  
    border-top: none; }
```

					ПГУ 1.090501.10.001 ПЗ	Лист
						190
Изм.	Лист	№ докум.	Подпись	Дата		

```

.select2-container--classic .select2-results > .select2-
results__options {

    max-height: 200px;

    overflow-y: auto; }

.select2-container--classic .select2-results__option[role=group] {

    padding: 0; }

.select2-container--classic .select2-results__option[aria-
disabled=true] {

    color: grey; }

.select2-container--classic .select2-results__option--
highlighted[aria-selected] {

    background-color: #3875d7;

    color: white; }

.select2-container--classic .select2-results__group {

    cursor: default;

    display: block;

    padding: 6px; }

.select2-container--classic.select2-container--open .select2-dropdown
{

```

```

border-color: #5897fb; }

body {

    font-family: 'Fira Sans', sans-serif;

    font-weight: lighter;

    color: $dark-grey;

    background-color: #f4f3f2;

    margin: 0;

}

.main {

    &.has-hero {

        .main-background {

            height: 450px;

            background: linear-gradient(150deg, #53f 15%, #05d5ff
70%, #a6ffcb 94%);

            @include mobile() {

                height: 100vh;

            }

        }

    }

}

}

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						192
Изм.	Лист	№ докум.	Подпись	Дата		

```

.main-background {

    position: absolute;

    top: 0;

    width: 100%;

    height: 40px;

    overflow: hidden;

    //    transform: skewY(-8deg);

    //    transform-origin: 0;

    background: linear-gradient(150deg, #53f 15%, #05d5ff 70%, #5897fb
94%);

    &:before {

        content: '';

        position: absolute;

        z-index: 0;

        background-image: url('/static/img/triangles.jpg');

        background-size: cover;

        width: 100%;

        height: 100%;

        opacity: .2;

    }

}

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						193
Изм.	Лист	№ докум.	Подпись	Дата		

```

.flyout {

    width: 100%;

    min-height: 100vh;

    position: fixed;

    z-index: 3;

    top: 0;

    background-color: $white;

    border-right: 1px solid $light-grey;


    @include desktop() {

        width: 25%;

    }

}

```

```

.push-menu, .flyout-right {

    width: 100%;

    min-height: 100vh;

    position: fixed;

    z-index: 3;

    top: 0;

    right: 0;

    background-color: $white;

```

```

    @include desktop() {

        width: 25%;

        top: 70px;

    }

}

.overlay {

    position: fixed;

    top: 0;

    left: 0;

    width: 100%;

    height: 100vh;

    background-color: rgba(0,0,0,.5);

    z-index: 1;

    display: none;

    animation-duration: .3s;

}

.dropdown-panel {

    background-color: $white;

    position: fixed;

```

```

top: 40px;

width: 100%;

height: 100vh;

overflow: scroll;

z-index: 2;


@include mobile () {

    top: 70px;

}


.close {

    right: 20px;

    fill: $darkest-grey;

    cursor: pointer;

}

}


.icon {

    width: 20px;

    height: 20px;

    vertical-align: middle;

}

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						196
Изм.	Лист	№ докум.	Подпись	Дата		

```

@mixin desktop() {

    @media (min-width: 768px) { @content ; }

}

@mixin mobile() {

    @media (max-width: 768px) { @content ; }

}

// Adds an scaling animation. Good for CTAs

@mixin scale-anim($value: 1.1) {

    transition: all 200ms ease-in;

    transform: scale(1);

    &:hover {

        transition: all 200ms ease-in;

        transform: scale(#{ $value });

    }

}

@mixin truncate($lines: 4, $lineHeight: 25) {

    overflow: hidden;

    text-overflow: ellipsis;

```

```

display: -webkit-box;

line-height: #{$lineHeight}px;      /* fallback */

max-height: #{$lineHeight * $lines}px;      /* fallback */

-webkit-line-clamp: $lines; /* number of lines to show */

-webkit-box-orient: vertical;

}

@mixin toggle-switch($width: 50px, $height: 17px, $size: 13px) {

    width: $width;

    height: $height;

    background: $white;

    font-size: 12px;

    display: inline-block;

    border-radius: 25px;

    vertical-align: middle;

    position: relative;

    cursor: pointer;


    label {

        margin: 0;

    }

```

```
label:before,
```

```
label:after {
```

```
    text-align: center;
```

```
}
```

```
label:before {
```

```
    display: inline-block;
```

```
    width: $width;
```

```
    height: $height;
```

```
    border: none;
```

```
    color: $blue;
```

```
    padding-top: 1px;
```

```
    text-align: center;
```

```
}
```

```
input[type=checkbox]:not(:checked) + label:before {
```

```
    content: "";
```

```
    padding-left: 10px;
```

```
}
```

```
input[type=checkbox]:checked + label:before {
```

```
    content: "";
```

```
    padding-left: 0;
```

					ПГУ 1.090501.10.001 ПЗ	Лист
						199
Изм.	Лист	№ докум.	Подпись	Дата		

```

        padding-right: 10px;
    }

    input[type=checkbox]:not(:checked) + label:after,
    input[type=checkbox]:checked + label:after {

        content: '';

        width: $size;

        height: $size;

        background: $blue;

        display: block;

        border-radius: 50%;

        position: absolute;

    }

    input[type=checkbox]:not(:checked) + label:after {

        top: 2px;

        left: 2px;

    }

    input[type=checkbox]:checked + label:after {

        top: 2px;

        right: 12px;

    }

}

```

```

@mixin random-color($selector) {

    #{$selector}: unquote("rgba("#{random(256)} - 1}, #{random(256)}-1},
#{random(256)} - 1}}, #{random(100)/100})");

}

.page {

    min-height: calc(100vh - 60px);

    padding-top: 70px;

    width: 100%;

    &.listing-page {

        min-height: calc(100vh - 110px);

    }

    .loader-small {

        display: block;

        margin: 0 auto;

    }

    .page-wrapper {

        background-color: $white;

        margin: 0 20px;

        min-height: calc(100vh - 150px);

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						201
Изм.	Лист	№ докум.	Подпись	Дата		

```

        @include mobile() {

            margin: 0;

        }

    }

    .title {

        text-align: center;

        padding-top: 20px;

        font-weight: lighter;

    }

    .meta {

        text-align: center;

        font-size: 13px;

        color: $grey;

        padding: 0 20px;

        .icon { margin: 0 5px }

        span { font-weight: bold }

    }

```

```

.columns {

    padding: 10px 20px 20px 20px;

    display: flex;


    @include mobile() {

        display: block;

        padding: 10px 0 0 0;

    }


    &.single-column {

        .page-content {

            @include desktop() {

                width: 85%;

            }

            @media (min-width: 1200px) {

                width: 60%;

            }

        }

    }

}

.column {

```

```

width: 50%;

padding: 20px;


@include mobile() {

    width: 100%;

}

}

.page-content {

margin: 0 auto;

text-align: justify;


@include mobile() {

    width: 100%;

}


img {

    width: 100%;

margin-bottom: 20px;

}

}

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						204
Изм.	Лист	№ докум.	Подпись	Дата		

```

.page-sidebar {

    @include mobile() {

        width: 100%;

    }

    img {

        width: 100%;

        margin-bottom: 20px;

    }

}

.hidden {

    display: none !important;

}

.clearfix:after {

    visibility: hidden;

    display: block;

    font-size: 0;

    content: " ";

    clear: both;

    height: 0;

}

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						205
Изм.	Лист	№ докум.	Подпись	Дата		

```

.clearfix { display: inline-block; }

/* start commented backslash hack */
* html .clearfix { height: 1%; }
.clearfix { display: block; }
/* close commented backslash hack */

.float-right { float: right }

.float-left { float: left }

$black:                #333333;

$white:                #ffffff;

$border-light:         rgba(255,255,255,.5);

$border-dark:          rgba(0,0,0,.5);

$very-light-gray:     #f7f7f7;

$quite-light-gray:    #f2f2f2;

$grey-bg:             #EEEEEE;

$lighter-grey:        #e5e5e5;

$light-grey:          #BBBBBB;

$grey-top:            #a0a0a0;

$grey:                #888888;

```

					ПГУ 1.090501.10.001 ПЗ	Лист
						206
Изм.	Лист	№ докум.	Подпись	Дата		

```

$grey-hover:          #777777;

$very-dark-grey:      #666666;

$dark-grey:           #444444;

$darker-grey:         #393939;

$darkest-grey:        #2B2B2B;


$main:                $darker-grey;


$green:               #2ecc71;

$green-dark:          #239955;

$red:                 #e8303f;

$blue:                #295cb1;

$blue2:               #0c1f3e;

$havelock-blue:       #448cd3;

$yellow:              #ffdd00;

$orange:              #f0ad4e;

```

Файл package.json:

```
{
```

Файл package.json:

```
{
```

					ПГУ 1.090501.10.001 ПЗ	Лист
						207
Изм.	Лист	№ докум.	Подпись	Дата		

Приложение Г
(Обязательное)

Приложение для создания онлайн курсов для обучения специалистов
Презентация

					ПГУ 1.090501.10.001 ПЗ	Лист
						208
Изм.	Лист	№ докум.	Подпись	Дата		

Пензенский государственный университет
Дипломный проект на тему:

Приложение для создания онлайн-курсов для обучения специалистов

Автор работы: ст-ка гр.19ВОЭ1 Степанова Дарья Павловна
Руководитель проекта: д.т.н., профессор кафедры «ВТ» Зинкин Сергей Александрович

Цель работы

Целью данного проекта является разработка приложения, которое позволит преподавателям создавать свои собственные онлайн-курсы. Приложение будет простым в использовании и позволит создавать курсы с различными функциями, такими как:

- видеоуроки;
- текстовые материалы;
- тесты;
- лекции.

Актуальность разработки

В современном мире онлайн-обучение становится все более популярным. Это связано с его преимуществами:

- **Доступность:** к онлайн-курсам может получить доступ любой человек;
- **Гибкость:** обучающиеся могут проходить курсы в удобное для них время;
- **Интерактивность:** онлайн-курсы могут быть более увлекательными и эффективными;
- **Стоимость:** онлайн-курсы, как правило, дешевле, чем традиционные методы обучения.

Аналоги



Изм.	Лист	№ докум.	Подпись	Дата

ПГУ 1.090501.10.001 ПЗ

Лист

210

Используемые технологии



Структура базы данных

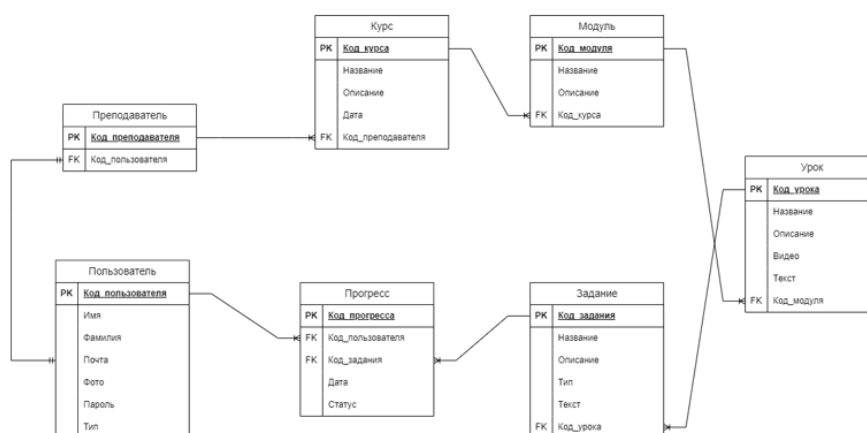
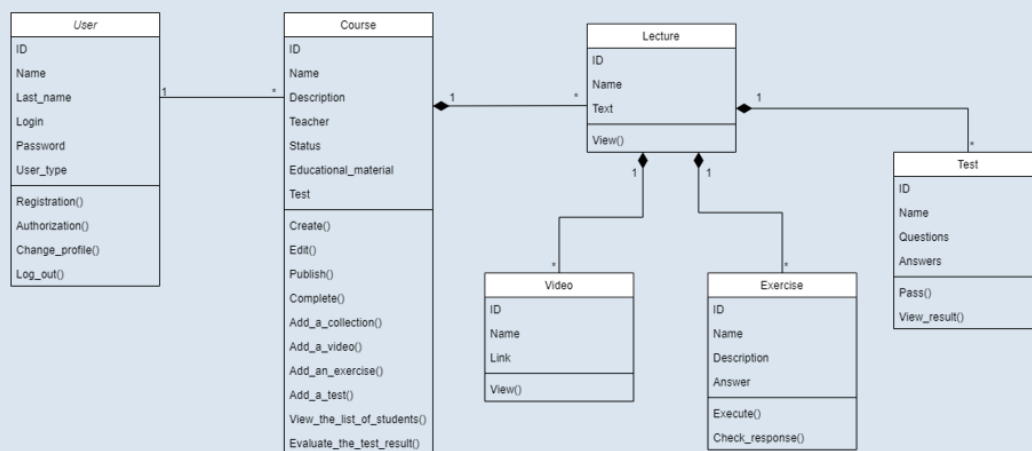
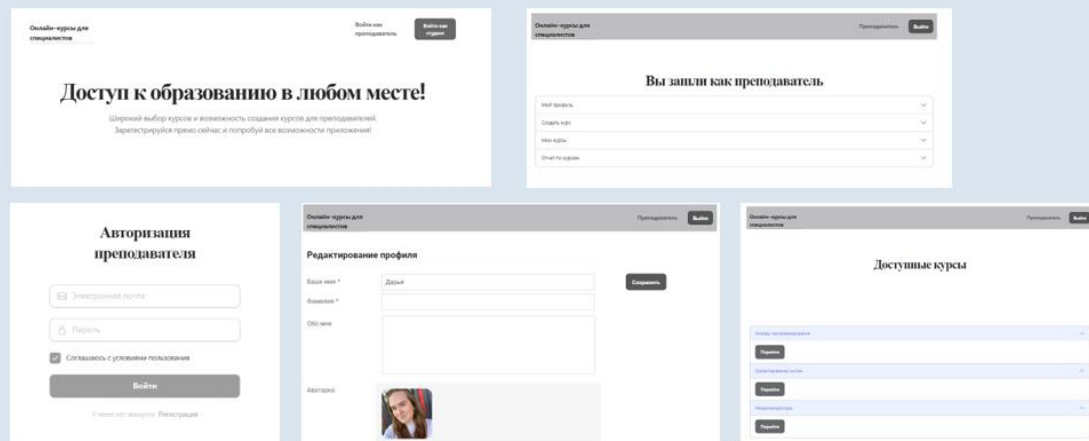


Диаграмма классов



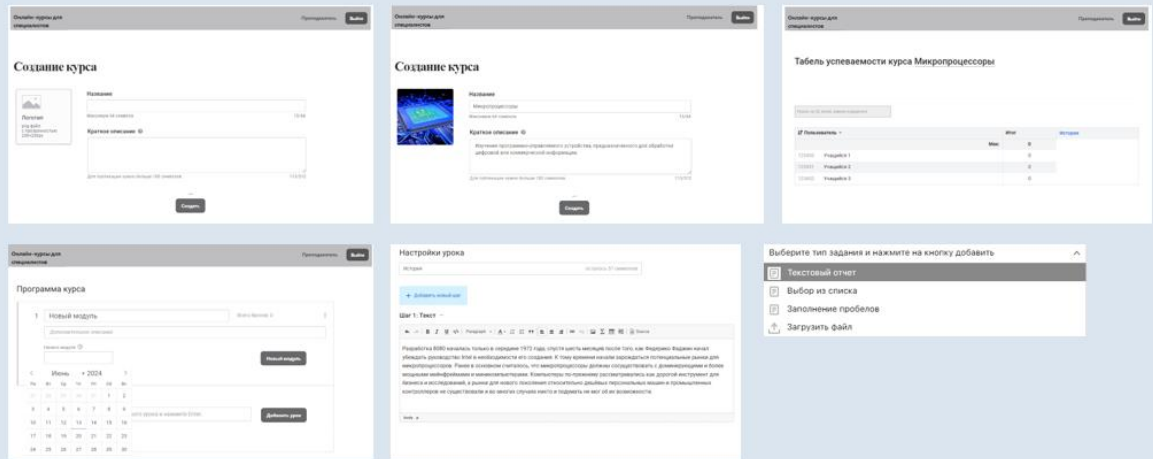
Демонстрация работы

Часть для преподавателя:



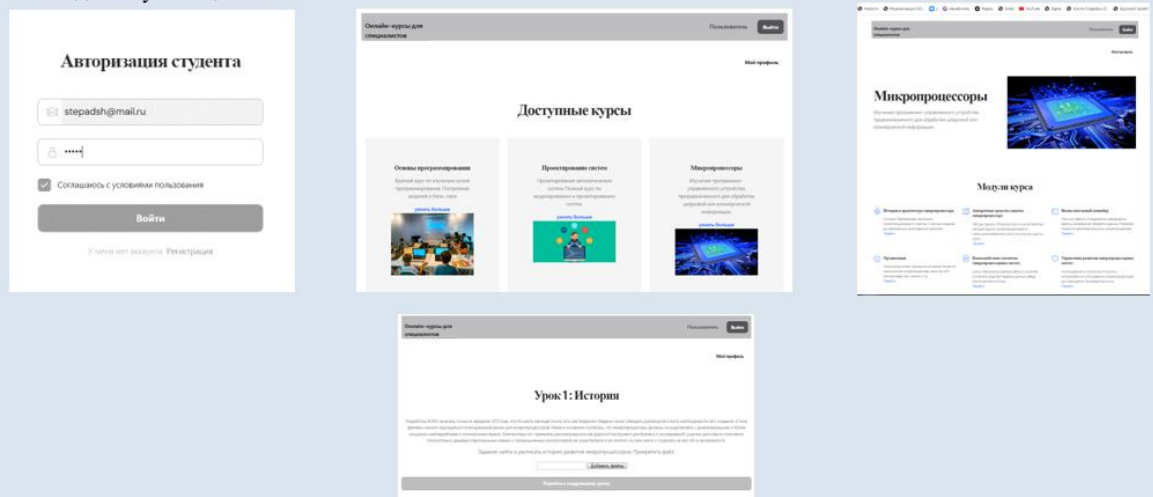
Демонстрация работы

Часть для преподавателя:



Демонстрация работы

Часть для обучающегося:





**Спасибо за
внимание!**

