МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

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

Факультет вычислительной техники

Кафедра МОиПЭВМ

Направление подготовки 09.03.02 «Информационные системы и технологии»

БАКАЛАВРСКАЯ РАБОТА

на тему

Приложение для обучения по курсу анатомия с использованием технологии дополненной реальности

Студент 20.06.20 Берсенев Роман Владимирович
(подпись, дата) (ФИО полностью)

Руководитель _____ Попова Н.А.
(подпись, дата) (фамилия, инициалы)

Нормоконтролёр ___ Такташкин Д.В.
(подпись, дата) (фамилия, инициалы)

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

Заведующий кафедрой _____ Макарычев П.П.
(подпись) (фамилия, инициалы)

Работа защищена с отметкой (протокол заседания ГЭК от №)

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

Пенза, 2020

Федеральное государственное бюджетное образовательное учреждение высшего образования

ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ПОЛИТЕХНИЧЕСКИЙ ИНСТИТУТ

Факультет	*Кафедра*
Вычислительной техники	МО и ПЭВМ

«Утверждаю»
Заведующий кафедрой
_____П.П. Макарычев
«__»_____2020г

ЗАДАНИЕ
НА ВЫПУСКНУЮ КВАЛИФИКАЦИОННУЮ РАБОТУ
БАКАЛАВРА

1. Студент_Беркутов Роман Вадимович_____гр._16ви1__факультета__ВТ___
<Фамилия, имя, отчество полностью>

2.Тема работы____Приложение для обучения по курсу анатомия с использованием технологии дополненной реальности_____

_____

Тема утверждена приказом ПГУ №_____от "_____"_____20_20__

2. Руководитель работы___к.т.н., доцент Попова Н. А._____

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

__Разработать приложение дополненной реальности и серверную часть к нему, моделирующую точную анатомическую модель человека. Приложение должно быть включено в образовательную систему Experiment, через которую происходит авторизация пользователя._____

__Приложение должно содержать интерактивные тестовые задания по курсу анатомии. И возможность просмотра модели человека в дополненной реальности._____

_Приложение должно работать на мобильных устройствах с операционной системой Android седьмой версии или выше и имеющих поддержку технологии дополненной реальности._____

_Для разработки использовать среды разработки Android Studio, NetBeans и СУБД PostgresSql для установки базы данных на серверной части приложения.

_____

_____

_____

_____

_____

_____

4. Перечень подлежащих разработке вопросов

1) Анализ требований и постановка задачи на разработку приложения и серверной части
2) Проектирование пользовательского интерфейса и базы данных приложения
3) Разработка тестов для проверки знаний студентов
4) Структура программы для отражения в AR модели человека
5) Взаимодействие с приложением Experiment
6) Локализация пользовательского приложения
7) Тестирование приложения

5. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Анализ требований и постановка задачи	15%	15.04.20-22.04.20	
2	Разработка приложения и серверной части	40%	23.04.20-13.05.20	
3	Тестирование системы	20%	14.05.20-24.05.20	
4	Оформление пояснительной записки	25%	25.05.20-10.06.20	

Дата выдачи задания "30" 12 2019

Руководитель бакалаврской работы _____
(подпись, дата)

Задание к исполнению принял студент Бердников 30.12.19
(подпись, дата)

Работу к защите допустить
Декан факультета _____
(подпись, дата)

Реферат

Пояснительная записка содержит 115 листов, 39 рисунков, 17 таблиц, 23 использованных источников и 3 приложения.

Приложение для обучения по курсу анатомия с использованием технологии дополненной реальности спроектировано и реализовано с помощью актуальных средств разработки.

Цель разработки: изучить предметную область и провести её анализ, рассмотреть конкурентные продукты, выбрать основные средства проектирования и разработки, разработать логическую и физическую модель базы данных, спроектировать пользовательский интерфейс, рассчитать экономическую выгоду, а также реализовать и протестировать серверное и клиентское приложение для информационной системы.

Результаты разработки: в процессе разработки была исследована и проанализирована предметная область образовательных приложений по анатомии, проанализированы конкурентные приложения, выбраны основные средства проектирования и разработки, разработаны логическая и физическая модель базы данных, спроектирован пользовательский интерфейс, рассчитана экономическая выгода, а также реализована и протестирована серверная и клиентская часть приложения для информационной системы.

Средства разработки: ERWIN Data Modeler, NetBeansIDE, Android Studio, PostgresSQL, Dbeaver, Gantter.

Оглавление

Введение.....7

1 Анализ предметной области.....9

1.1 Анализ предметной области.....9

1.2 Современные проблемы рынка мобильных приложений.....11

1.3 Основные технологии в разработке приложения..... 12

1.4 Анализ рынка конкурентного программного обеспечения..... 13

1.5 Анализ функциональных требований..... 15

1.6 Анализ требований к архитектуре ПО..... 17

1.7 Выбор средств разработки.....18

1.8 Результаты анализа предметной области.....19

2 Проектирование системы.....21

2.1 Базовая архитектура системы.....21

2.2 Логическая модель базы данных..... 22

2.3 Проектирование программного интерфейса..... 25

2.4 Проектирование Тестов..... 32

2.5 Проектирование программных средств..... 34

2.6 Планирование проекта и оценка бюджета.....35

3 Реализация системы.....42

3.1 Реализация серверной части приложения.....42

3.2 Реализация пользовательского приложения.....45

3.3 Реализация поддержки платформы Experiment..... 45

3.4 Реализация локализации приложения.....47

4 Тестирование системы и анализ качества кода.....49

4.1 Тестирование по принципу белого ящика.....49

4.2 Метод покрытия операторов..... 51

4.3 Метод покрытия условий.....51

4.4 Метод покрытия маршрутов.....52

4.5 Метрики кода..... 54

4.6 Конфигурационное тестирование под разные модели.....56

Заключение..... 58

Список использованных источников..... 60

Приложение А. Листинг программы AtomyAR 63

Приложение Б. Описание атрибутов таблиц физической модели БД 76

Приложение В. Реализация RESTful сервисов на языке Java..... 86

Введение

Мобильные приложения это удобный инструмент в обучении. С распространением платформы увеличилось и разнообразие коммерческих проектов. Многие из них затрагиваю сферу образования и предлагают различные способы предоставления данных: видео, тесты, задачи, обучающие игры.

В настоящее время существует достаточное количество коммерческих приложений, реализующих просмотр анатомии человеческого тела. Однако все они являются дорогостоящими, и выполняют достаточно большое число бизнес-процессов, не всегда необходимых для небольших предприятий и являющиеся самостоятельным продуктом. В связи с этим было решено разработать недорогой вариант приложения системы, связанной с целым комплексом образовательных программ. Особенностью приложения, так же будет использование передовой технологии дополненного реальности.

AR(augmented reality). Дополненная реальность – это среда, в реальном времени дополняющая физический мир, каким мы его видим, цифровыми данными с помощью каких-либо технических устройств – планшетов, смартфонов или других, и программной части. Например, Google Glass или камеры мобильного устройств.

Актуальность работы. Данная работа является актуальной потому, что в образовательных учреждениях возникает спрос на современные решения в образовательной сфере с внедрением мобильных приложений. Разрабатываемое в рамках этой квалификационной работы приложение должно облегчить процесс обучения и оценки успеваемости учеников, применяя современные технологии и решения.

Цель и задачи работы. Целью данной квалификационной работы является разработка мобильного приложения и серверной части к нему, задачей которого будет являться обучающий процесс по курсу анатомии школьников и студентов с целью изучения точного анатомического строения человеческого тела. В том числе будет проводится отслеживание результатов

изучения темы путем встроенных тестов. Это даст возможность учителям и ученикам оценивать процесс обучения.

1 Анализ предметной области

1.1 Анализ предметной области

Мобильные приложения прочно обосновались в повседневной жизни, но многие организации: учебные или творческие объединения, не полностью принимают, что может дать им использование приложений в учебе. Чаще всего учебные организации используют иные способы обучения: бумажные учебники, занятия в классах, аудиторные лекции. Но мобильные приложения помогут оказать куда больший эффект.

Если посмотреть на коммерческие организации, то можно заметить, что почти каждая пользуется различными приложениями. Они значительно упрощают отслеживание информации, поддержку продукта и способствуют быстрому обучению сотрудников.

Компании преследуют несколько целей при использовании обучающих приложений:

– стандартизация приложений поможет поддерживать скорость обучения и подготовит пользователей к решению конкретных задач на доступных примерах;

– поддержка актуальных знаний у сотрудников;

– увеличение заинтересованности в предмете обучения;

– хороший способ дистанционно повысить квалификацию пользователей, особенно с учетом, что современные средства позволяют передавать не только текстовую информацию, а также звуковую и видео;

– приложение как способ прохождения сертификации, так как его алгоритмы позволяют оценить степень изучения учебного материала, намного оперативнее, чем традиционные способы оценки;

– постоянное обновление приложений;

– возможность проводить исследования, анализ которых поможет организации скорректировать свою работу и распределить роли в зависимости от уровня овладения знаниями.

Сложно сказать об успешности той или иной компании, если у нее нет интеграции с приложениями. Это же относится и к учебным организациям.

Основным компонентом разрабатываемой программы является AR(augmented reality). Дополненная реальность - это среда, в реальном времени дополняющая физический мир, каким мы его видим, цифровыми данными с помощью каких-либо устройств - планшетов, смартфонов или других, и программной части. Например, Google Glass или шлем Железного Человека. Системы прицеливания в современных боевых самолетах - это тоже дополненная реальность.

Дополненную реальность (augmented reality, AR) надо отличать от виртуальной (virtual reality, VR) и смешанной (mixed reality, MR). В дополненной реальности виртуальные объекты проецируются на реальное окружение.

Виртуальная реальность - это созданный техническими средствами мир, передаваемый человеку через (пока что) органы чувств. Смешанная или гибридная реальность объединяет оба подхода [1].

VR и AR уже нашли своё применение в обучении. Они помогают показывать и рассказывать о вещах произошедших давно в истории, с выдающимися людьми. Ученики получают возможность взаимодействовать с объектами и персонажами. При таких сценариях обучения можно вырастить целое поколение таких профессий, которые сегодня остаются менее востребованными в обществе, но всё еще являются крайне необходимыми для сохранения многогранности современной науки. Появляется возможности большего углубления в изучаемый материал [2].

Обучающее приложение может сообщить о событиях группы, изменениях в расписании занятий, содержать обучающие статьи или отзывы от участников событий, содержать приглашения на онлайн лекции в интернете. С подобными информационными системами можно быть в состоянии отслеживать актуальны события в интересующей предметной области.

1.2 Современные проблемы рынка мобильных приложений

На данный момент рынок мобильных приложений развивается очень быстро. Есть возможность найти приложение для современных моделей телефонов под различные сферы деятельности человека. В настоящее время на рынке возможно найти продукты отечественного производства. Все современные разработчики сталкиваются со схожими проблемами:

- производительность приложения падает с ростом базы клиентов;
- перегруженность объектами баз данных;
- возможность перегрузки собственных серверов;
- излишние количество рекламы в приложении;
- безопасность приложения;
- эстетическая непривлекательность приложения.

Со всеми этими проблемами приходится сталкиваться производителям по всему миру. Уже не стоят вопросы о недостатке клиентов, а речь может быть об их избыточности. Рынок мобильных приложений – перегружен. В результате чего может быть действительно сложно создать конкурентноспособное приложение [3].

Перед разработкой приложения необходимо определить преимущества и недостатки каждого имеющегося типа конкретно.

Мобильные приложения, доступные для покупки и установки на смартфон, имеют тематический контент, наглядно структурированы по отдельным направлениям и специализированы на информационные нужды конкретных категорий пользователей.

Категории мобильных приложений по исполняемым функциям. К наиболее востребованным категориям мобильных приложений следует отнести следующие:

- развлечения (игровые приложения, мультимедиа, музыкальные библиотеки, оформление билетов на концерт, кинотеатры);
- путешествия (покупка билетов, резервирование отеля, местная карта, сервис онлайн - переводчика);

– бизнес (финансовая аналитика, фондовые биржи, торговые операции, поиск работы);

– социальные приложения (социальные сети, мессенджеры, специализированные сети);

– еда (заказ и доставка продуктов, оценка и расположение заведения питания, рецепты готовки);

– спорт (спортивные трансляции, покупка мест на спортивные мероприятия);

– новости (дайджесты, ленты, форумы).

Следует отметить, что представленный список классификаций постоянно развивается и дополняется. Более того, происходит переход мобильных приложений в новые виды сервисов, включающие 3D-изображения, дополненную реальность, датчики физического состояния человека, мобильных агентов в системах безопасности, интеллектуальных агентов в гибридных сетях и когнитивных системах и многое другое.

Также необходимо отметить, что мобильные приложения по принадлежности к разработчикам делятся:

– на нативные (создаются поставщиками операционных систем и загружаются через их магазины приложений);

– кросс-платформенные (HTML5, веб-сайт и веб-приложение, оптимизированные под мобильное устройство и его ОС);

– стандартные (гибридные) приложения (содержат в себе некоторые функции нативных и веб-приложений).

Выбор приложения в этом случае представляет интерес для разработчиков с точки зрения быстродействия, независимости от поставщика, безопасности работы, удобства разработки и ряда других обстоятельств.

1.3 Основные технологии в разработке приложения

Данная работа посвящена разработке серверной и клиентской части обучающего приложения. Актуальность исследования заключается в том, что рынок мобильных устройств продолжает активно расти и все больше

учебных заведений хотят иметь инструменты для ускорения и отслеживания прогресса своих учеников. При этом они сталкиваются с проблемой выбора обучающего приложения. Теоретическую основу для данной работы составляет технология взаимодействия компонентов REST.

В определённых случаях (многие системы, основанные на обмене данными) этот архитектурный стиль приводит к повышению производительности и упрощению архитектуры.

REST (Representational state transfer) – это стиль архитектуры программного обеспечения для распределенных систем, таких как World Wide Web, который, как правило, используется при построении вебслужб. Термин REST был введен в 2000 году Роем Филдингом, одним из авторов HTTP протокола. Системы, поддерживающие REST, называются RESTful системами.

Ключевым результатом разработки является мобильное приложение на базе андроида. По итогам работы можно сделать следующие выводы:

– рынок мобильных приложений достаточно обширен и требует создавать конкурентоспособные приложения;

– условия роста Android Studio, как платформы помогают разработчику выполнять работу в несколько раз быстрее, что позволяет сэкономить большое количество рабочего времени;

– Android Studio предоставляет средства и удобные пакеты компонентов с готовыми решениями;

– имеется обширная библиотека в сфере обучения работе на платформе.

На основе сделанных выводов будут определены основные шаги разработки.

1.4 Анализ рынка конкурентного программного обеспечения

Из-за большого количества разных групп, объединяющих людей с общими интересами, платформ для обучения в конкретной области достаточно много. В основном, они представлены отдельными решениями и

неполным курсом с платной подпиской. Был произведен анализ нескольких платформ для обучения анатомии.

1. Анатомический атлас. Классический вариант обучения студентов и школьников на данный момент. Основным недостатком является не интерактивность взаимодействия с предметом обучения.

2. Biosphera – платформа для интерактивного взаимодействия с анатомически точной моделью человека. Данная платформа имеет плохое развитие в российском сегменте. Основным недостатком является платный доступ к полной модели и ограничение одной (рисунок 1).

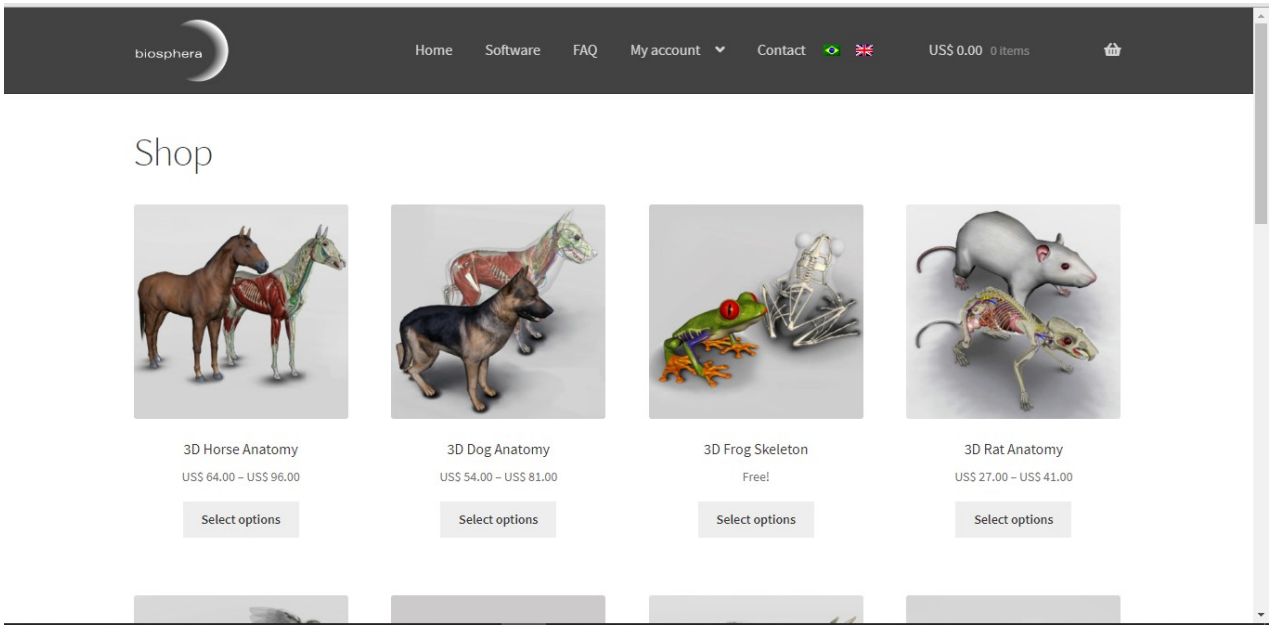

Рисунок 1 – Biosphera

3. Анатомия 3D Атлас. К подобным приложениям можно отнести множество атласов на разные тематики. Они имею хорошо описанную модель разбитую на части для удобства ознакомления. Обычно, подобные приложения очень подробны, но имеет несколько недостатков:

– пользователь для доступа ко всей модели должен получить платный аккаунт;

– ограниченность изучения модели в одной плоскости.

Подробное сравнение конкурирующих продуктов предстленно в таблице 1.

Таблица 1 – Подробное сравнение конкурирующих продуктов

№ п/п	Ф-ции	AnatomyAR	Анатомический атлас	Biosphera	Анатомия-3D Атлас
1	Прохождение тестов.	Поддерживает	Не поддерживает	Не поддерживает	Не поддерживает
2	Просмотр 3D модели	Поддерживает	Не поддерживает	Поддерживает	Поддерживает
3	Поддержка разных языков	Поддерживает	Не поддерживает	Поддерживает	Поддерживает
4	Платный контент	Не поддерживает	Не поддерживает	Поддерживает	Поддерживает
5	Поддержка образовательных учреждений	Поддерживает	Поддерживает	Не поддерживает	Не поддерживает

Рассмотренные продукты не предоставляют полный и удобный доступ пользователю. При разработке були учтены минусов представленных на рынке аналогов. В итоге разработанное приложение AnatomyAR, перенимает лучшие стороны конкурентов, избавившись от недостатков.

1.5 Анализ функциональных требований

Основные элементы приложения представлены ниже. Набор частей приложения:

- авторизация пользователя;
- пользовательское меню;
- свободный просмотр модели;
- прохождение тестовых заданий, выданных преподавателем.

Форма просмотра списка тестов. Предоставляет возможность просмотра результатов прохождения тестовых заданий. Результат выдается в процентном соотношении.

Форма просмотра модели. Позволяет просматривать 3D модель человека в дополненной реальности.

Профиль пользователя. Содержит фото пользователя, количество пройденного материала и основные данные пользователя. Возможно добавление статистики по использованию приложения.

В рамках работы требуется разработать эффективный пользовательский интерфейс (ПИ). Под эффективностью понимается совокупность свойств:

– естественность – ПИ должен отображать работу систему способом, не требующим дополнительного объяснения.

– дружественность – способность заранее избегать ситуации, приводящих к отказам в приложении или недопониманию у пользователя.

– скорость ответа – обработка действий пользователя не должна быть ощутимо долгой.

– адаптивность – система должна корректно отображаться вне зависимости платформы и характеристик оборудования пользователя в допустимых границах.

Исходя из рассмотренных требований разрабатываемое приложение будет содержать несколько вариантов использования (рисунок 2).

Рисунок 2 – Диаграмма вариантов использования приложения

Разрабатываемое приложение должно будет соответствовать актуальным требованиям рынка:

– так пользователь сможет просмотреть анатомическую модель человека в дополненной реальности;

– пройти авторизацию через общую платформу для всех приложений;

– отследить свой прогресс выполнения тестов в приложении;

– выполнить тематические задания по анатомии в виде тестов.

Качественный процесс обучения будет достигаться за счет открытости работы тестовых заданий и возможности просмотра анатомически точной модели разделенной на тематические группы.

1.6 Анализ требований к архитектуре ПО

В приложении была выбрана структура Model- View- Controller (MVC, «Модель – Представление – Контроллер», «Модель – Вид – Контроллер») – схема разделения данных приложения, пользовательского интерфейса и управляющей логики на три отдельных компонента: модель, представление и контроллер – таким образом, что модификация каждого компонента может осуществляться независимо [4]. К основным компонентам относятся:

– модель (Model) предоставляет данные и реагирует на команды контроллера, изменяя свое состояние;

– представление (View) отвечает за отображение данных модели пользователю, реагируя на изменения модели;

– контроллер (Controller) интерпретирует действия пользователя, оповещая модель о необходимости изменений.

Основная цель применения этой архитектуре состоит в отделении бизнес-логики (модели) от её визуализации (представления, вида). За счёт такого разделения повышается возможность повторного использования кода. Наиболее полезно применение данной концепции в тех случаях, когда пользователь должен видеть те же самые данные одновременно в различных

контекстах и/или с различных точек зрения. В частности, выполняются следующие задачи:

– к одной модели можно присоединить несколько видов, при этом не затрагивая реализацию модели;

– не затрагивая реализацию видов, можно изменить реакции на действия пользователя (нажатие по кнопке, ввод данных, ручное изменение) – для этого достаточно использовать другой вид контроллера.

Ряд разработчиков специализируется только в одной из областей: либо разрабатывают графический интерфейс, либо разрабатывают бизнес-логику. Поэтому возможно добиться того, что программисты, занимающиеся разработкой бизнес-логики (модели), вообще не будут осведомлены о том, какое представление будет использоваться [5].

1.7 Выбор средств разработки

Для основных этапов проектирования использовались такие средства как ERwin Data modeller и Online UML Diagram Tool [6]. Удобные инструменты в построении uml диаграмм и разработки баз данных [7].

Для разработки приложения был выбран фреймворк AR Core для разработки приложений под платформу андроид и специализирующейся на дополненной реальности , написанный на языке программирования Java. Они разработаны, чтобы сделать программирование приложений проще, так как использует ряд допущений о том, что нужно каждому разработчику для создания нового проекта. Он позволяет использовать готовые модули в процессе программирования, в сравнении с другими языками и фреймворками [8].

Для создания и редактирования базы данных были выбраны PostgreSQL – объектно-реляционная СУБД [9], DBeaver – это свободный кроссплатформенный менеджер баз данных для Linux, Windows и MacOS. Положительными чертами этих приложений является: удобство, мультифункциональность, портабельность, кроссплатформенность и принадлежность к категории свободного ПО – Open Source [10].

Основной механизм обмена запросами и обновлением состояния сущностей в базе данных реализован в NetBeans. NetBeans IDE – свободная интегрированная среда разработки приложений (IDE) на языках программирования Java, Python, PHP, JavaScript, C, C++, Ада и ряда других [11].

Основной платформой разработки приложения была выбрана Android Studio. Android studio – интегрированная среда разработки, предназначенная для создания приложений. Причем в этой среде возможно разрабатывать приложения для любых устройств на Android: смартфон, планшет, умные часы и т.д.

Для написания приложений под Android, необходима соответствующая среда разработки. Необходимые инструменты собраны используемой Android studio [12].

1.8 Результаты анализа предметной области

В результате анализа была сформулирована задача на разработку системы, актуальную на рынке приложений: разработать серверную и клиентскую часть на языке Java с использованием фреймворка AR Core, использующего MVC структуру и технологию Rest. Для создания приложения была выбрана среда разработки Android Studio. Для реализации базы данных были выбраны объектно-реляционная СУБД – PostgreSQL и свободный кроссплатформенный менеджер баз данных – DBeaver.

Все рассматриваемые методики, архитектуры и технологии соответствую актуальным требованиям рынка. Было важно учитывать комплексность задачи и возможность включения разрабатываемого приложения в большую систему обучения.

Для этого требуется выполнить следующие подзадачи:

- спроектировать и реализовать базу, используя PostgreSQL;
- сформировать классы ИС и связи между ними;
- разработать контроллеры бизнес-логики;
- создать сервисы для решения распространённых задач внутри ИС;

– реализовать слой представления в виде приложения;
– реализовать связь между приложением и базой данных.

После выполнения задач будет проведено комплексное тестирование приложение и его серверной части с базой данных. Это необходимо для последующей интеграции приложения в обучающие организации и возможного коммерческого использования.

2 Проектирование системы

2.1 Базовая архитектура системы

Очень важную роль в современной разработке мобильных приложений имеет серверная архитектура. Она определяет основные компоненты и их расположение в системе. Различные вариации уменьшат нагрузку на сервер или конечный интерфейс пользователя. Проектируемая система имеет двухуровневую архитектуру клиент-сервер. Архитектура мобильного приложения представлена на рисунке 3.

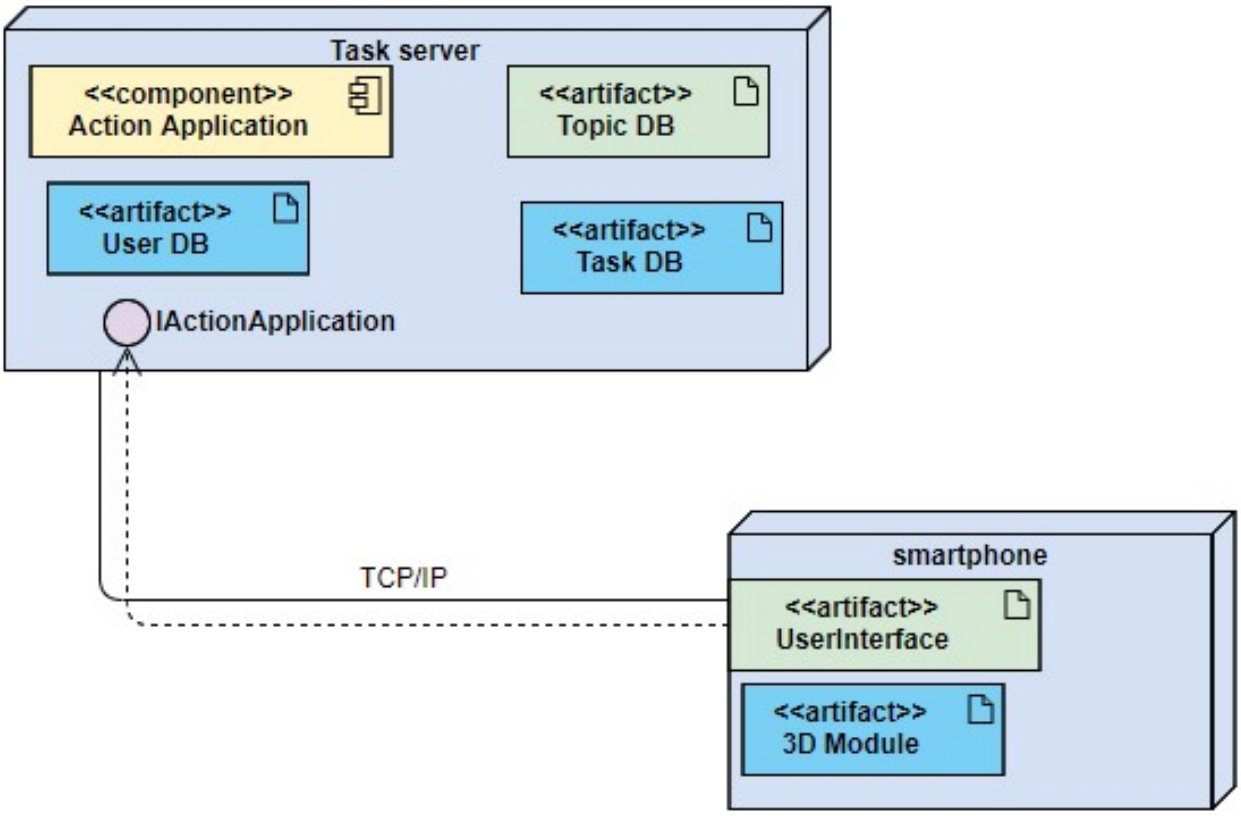

Рисунок 3 – Схема архитектуры «клиент-сервер»

Данный вид архитектуры определяется двумя основными компонентами:

– смартфон(клиент), который является интерфейсом для взаимодействия между пользователем и серверной частью информационной системы;

– сервер базы данных, который содержит всю бизнес логику и базу данных, в которой хранятся данные о заданиях, личных аккаунтах учеников и содержат информацию на всех требуемых языках.

Основной контроль учетных записей и новых заданий будет происходить на стороне сервера. Приложение будет корректно отображать новые задания и их выполнение на стороне ученика

2.2 Логическая модель базы данных

Логическая модель описывает понятия предметной области, их взаимосвязь, а также ограничения на данные, налагаемые предметной областью [14]. Кроме того, в ходе изучения предметной области были выделены категории пользователей для решения соответствующих задач, а также выделен фрагменты структуры предметной области, к которым они относятся. Подробное описание представлено в таблице 2.

Таблица 2 – Категории пользователей

№ п/п	Категория пользователей	Ф-ции	Информационные задачи
1	Ученик	Прохождение тестов, просмотр/изучение 3D модели	Прохождение тестов.
2	Учитель	Выдача заданий, просмотре выполнения заданий	Выдача тестов.
3	Сотрудник	Поддержание проекта	Контроль за состоянием базы данных.

В результате анализа предметной области, на основе выделенных сущностей, была построена логическая модель базы данных в среде CA Erwin Data Modeler [15].

Модель можно посмотреть на рисунке 4.

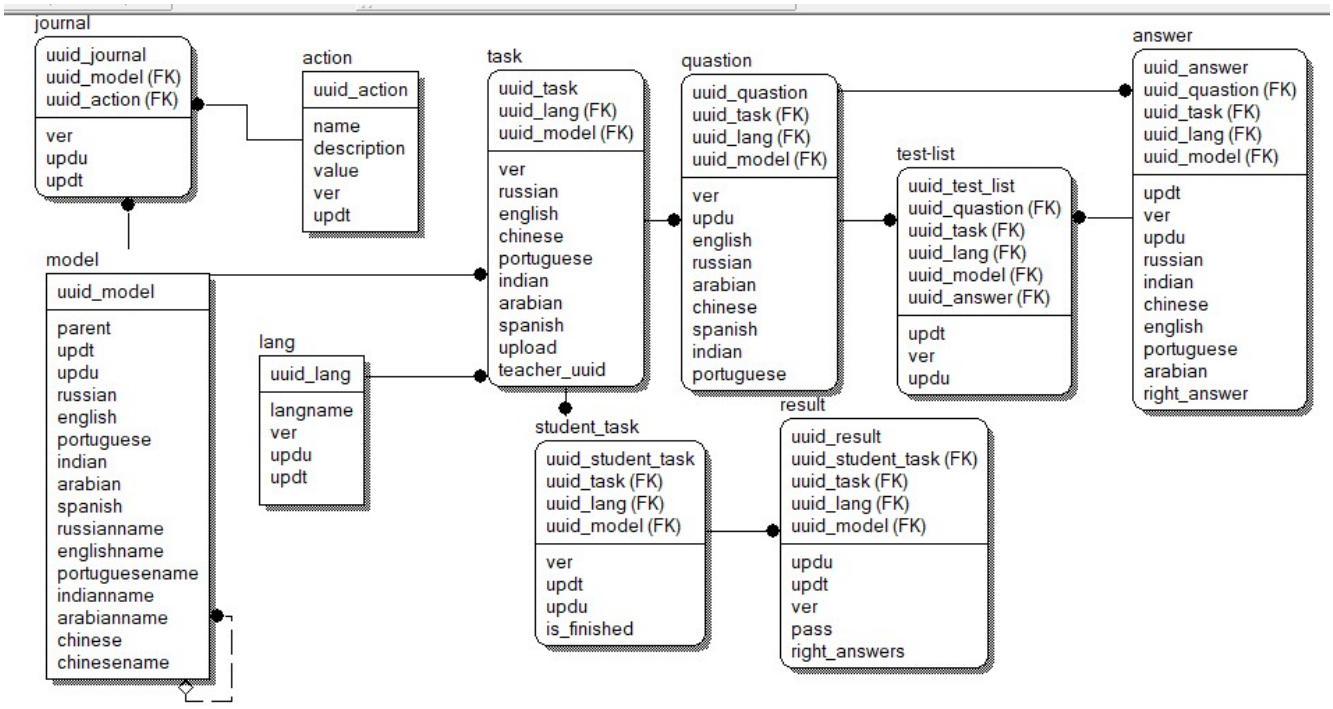

Рисунок 4 – Логическая модель базы данных приложения

В дальнейшем будет производиться переход от логической модели БД к физической [16].

Но уже на этапе логической модели стоит отметить некоторые особенность проектируемой БД:

– БД, будет поддерживать 7 языков, для пользовательского интерфейса;

– модель, используемая в дополненной реальности будет содержать иерархическую структуру. Это обусловлено предметной областью и необходимостью быстрого выполнения запросов СУБД;

– данная схема реализует возможность записи действий пользователя;

– проектируемая БД будет интегрирована в уже разработанный комплекс приложений.

Для этого добавлена возможность отслеживания внесения правок и контроль версии сущностей БД.

На основе построенной логической модели будет построена и реализована физическая. В таблице 3 приводится описание основных выделенных сущностей.

Таблица 3 – Типы сущностей и их описание

№	Наименование	Описание	Категория	Количество экземпляров	Режим обновления
1	journal	Содержит сведения о действиях пользователя	Стержневая	500	10% в год
2	model	Данные о модели человека	Стержневая	150	5% в год
3	action	Списочный перечень действий	Справочная	5	10% в год
4	lang	Списочный перечень языков	Справочная	8	0% в год
5	task	Сведения о выданных заданиях	Стержневая	1500	5% в год
6	quastion	Список вопросов	Стержневая	1000	10% в год
7	answer	Список ответов	Стержневая	4000	0% в год
8	test_list	Список вопросов с ответами	Справочна	400	10% в год
9	student_task	Задание студента	Стержневая	15000	10% в год

Продолжение таблицы 3

	Наименование	Описание	Категория	Количество экземпляров	Режим обновления
10	result	Результат выполнения задания студентом	Справочна	15000	10% в год

Данная таблица помогает лучше понять построенную логическую модель серверной части приложения.

2.3 Проектирование программного интерфейса

На основе диаграммы вариантов использования был спроектирован графический интерфейс пользователя, исходя из его функций. Все интерактивные элементы интерфейса имею анимацию, для лучшего отображения взаимодействия с пользователем.

Полное описание кода пользовательского приложения «AnatomyAR» можно посмотреть в приложении А.

Приложение начинается с формы входа. Оно дает понять основную тематику и стилистику продукта. Меню сделано специально, чтобы вызвать интерес у учеников и студентов.Оно интуитивно понятно и эргономически приспособлено к использованию. Все интерактивные элементы интерфейса имеют анимацию взаимодействия.Приложение в целом имеет настраиваемую под мобильное устройство разрешение. Им будет одинаково удобно пользоваться на устройствах разных марок и производителей.

Интерфейс формы входа приведен на рисунке 5.

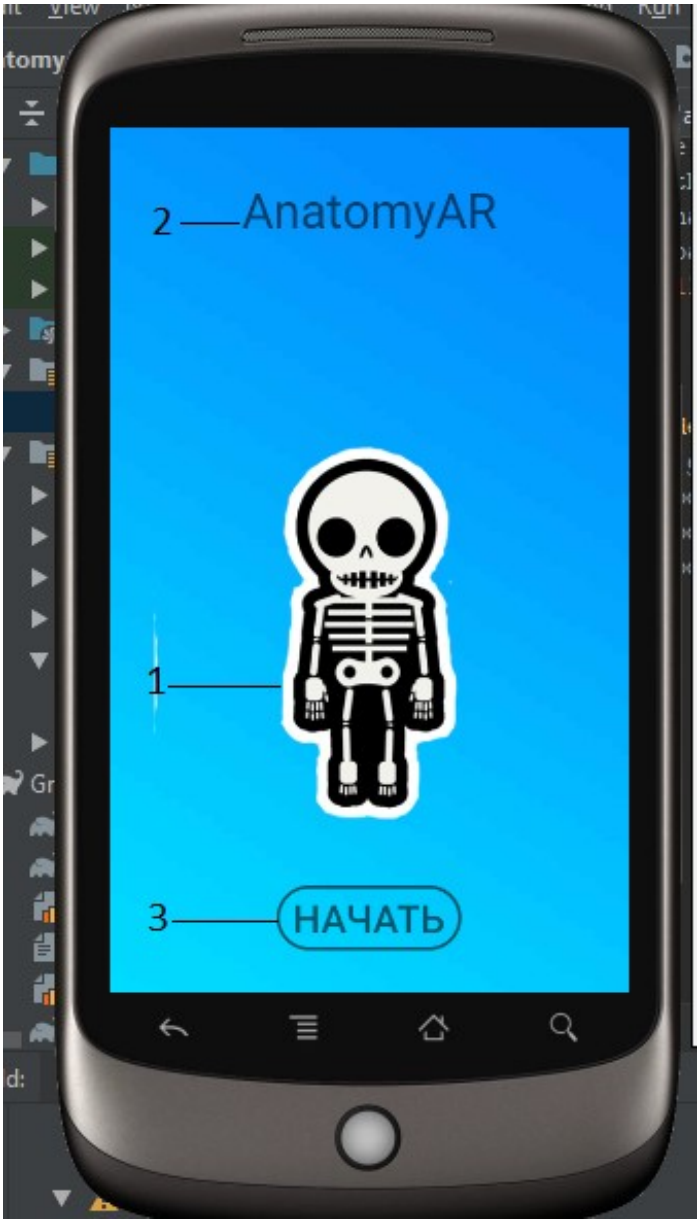

Рисунок 5 – Интерфейс формы входа

При этом на рисунке 5есть элементы:

1 – текстовая метка, компонент TextView;

2 – фоновая иконка ImageBox;

3 – кнопка входа в приложение, компонент Button.

Интерфейс меню представляет собой компоненты Button для перехода к тестам, Button2 для перехода к модели в AR, Button3 для вывода основной информации о приложении, а также TextView, который содержит название приложения. Основной интерфейс регистратора приведен на рисунке 6.

Рисунок 6 – Интерфейс формы Меню

На рисунке изображены следующие элементы:

1 – текстовая метка, компонент TextView;

2 – кнопка перехода к тестам, компонент Button;

3 –кнопка перехода к модели в дополненой реальности, компонент Button;

4 – кнопка вывода основной информации, компонент Button.

Рассмотрим меню вкладки тестов, который приведен на рисунке 7. Они нужны для более удобного доступа к тематическим тестам. Все элементы являются интерактивными и имеют анимацию при нажатии пользователем.

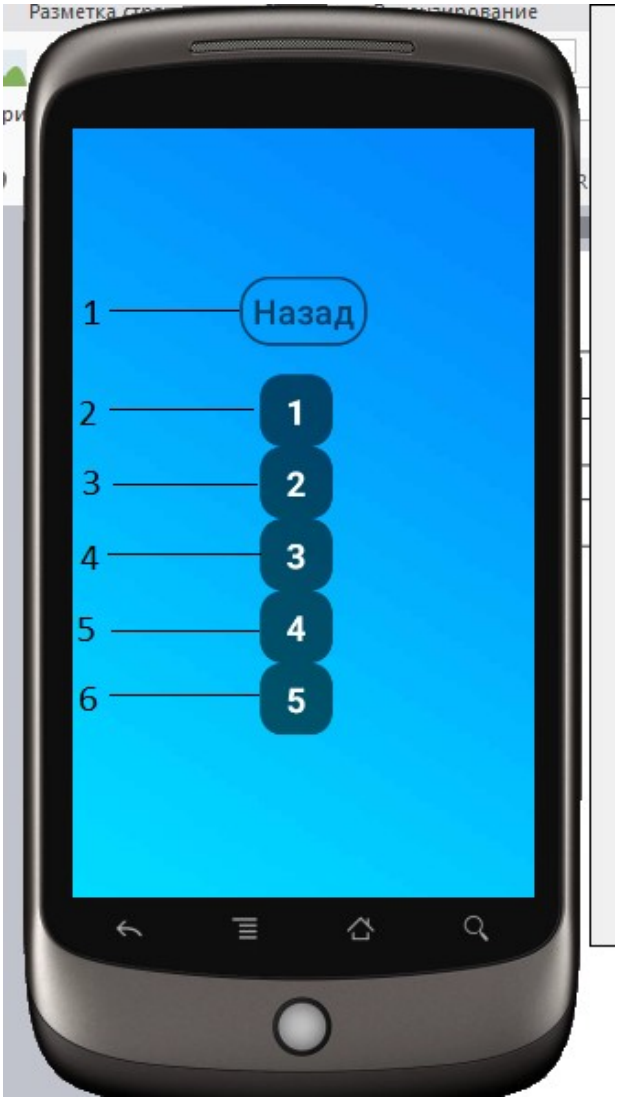

Рисунок 7 – Интерфейс пункта меню «Тест»

Рассмотрим основные элементы интерфейса, изображенные на рисунке 7:

1 – кнопки возврата к основному меню, компонент Button;

2 – кнопки перехода к первому уровню, компонент TextView;

3 – кнопки перехода ко второму уровню, компонент TextView;

4 – кнопки перехода к третьему уровню, компонент TextView;

5 – кнопки перехода к четвертому уровню, компонент TextView;

6 – кнопки перехода к пятому уровню, компонент TextView.

При нажатии на компонент TextView, произойдет переход к тематическому тесту. Перед открытием самого теста появится диалоговое окно с правилами для данного тематического теста и возможностью возврата

к меню тестов. Интерфейс диалога, предшествующему прохождению теста, приведен на рисунке 8.

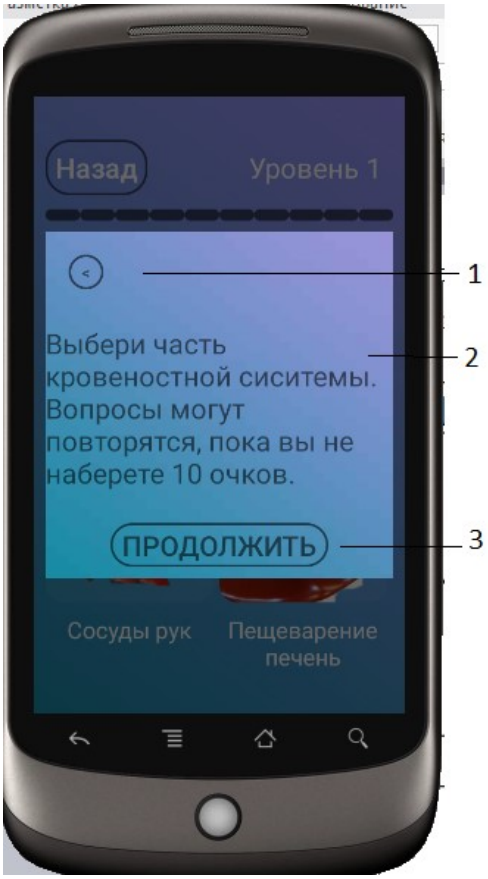

Рисунок 8 – Интерфейс диалога, предшествующему прохождению теста

Рассмотрим основные элементы интерфейса, изображенные на рисунке 6:

1 – компонент Button, даля возврата ко вкладке «Тест», выделен большим шрифтом;

2 – компонент TextView, с описанием задания теста и правилами его прохождения;

3 – компонент Button, для начала прохождения теста;

Интерфейс окна теста приведен на рисунке 9.

Рисунок 9 – Интерфейс теста

Рассмотрим основные элементы интерфейса, изображенные на рисунке 7:

1 – компонент Button, для возврата к меню тестов;

2 – компонент TextView, для отображения текущего уровня;

3 – компонент TextView, для отображения прогресса прохождения уровня;

4 – компоненты ImageView, для выбора ответа;

5 – компоненты TextView, для отображения описания к компонентам ImageView.

Далее рассмотрим вкладку «Модель», графический интерфейс которой приведен на рисунке 10.

Рисунок 10 – Интерфейс пункта меню «Модель»

3D модель приложения AnatomyAR, отображаемая в дополненной реальности. приведена на рисунке 11. Модели были подготовлены согласно курсу анатомии [17].

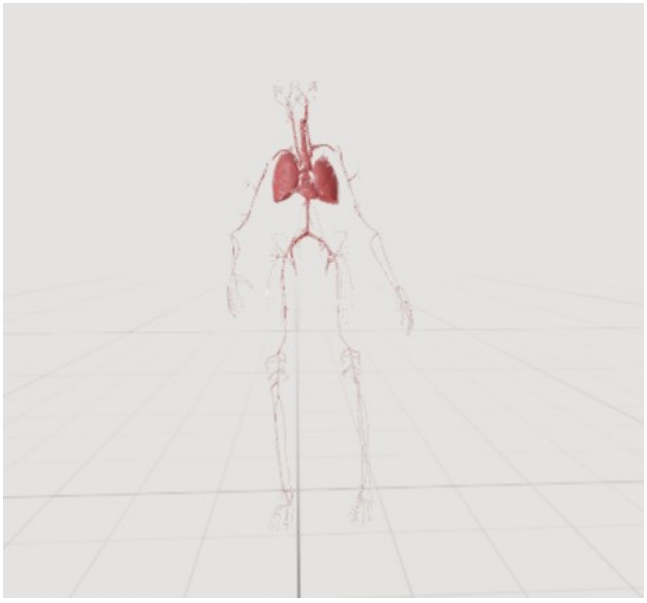

Рисунок 11 – Интерфейс пункта меню «Модель»

Основным элементом -является компонент ARfragment, для размещения 3D моли человеческого тела с помощью камеры устройства в дополненной реальности.

Теперь рассмотрим диалоговое окно, которое появляется по нажатию на компонент Button в основном меню, диалог приведен на рисунке 12.

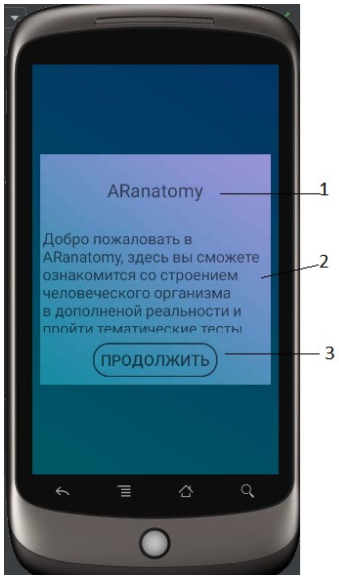

Рисунок 12 – Диалоговое окно с информацией о приложении.

Рассмотрим основные элементы интерфейса, изображенные на рисунке 2.9:

1 – Название приложения, компонент TextView;

2 – Описание приложения, компонент TextView;

3 – Кнопка возврата в меню, компонент Button.

Элементы интерфейса разработаны в общей стилистике. Пользователю будет удобно ориентироваться по приложению. Особенности разрабатываемого приложения гарантируют удобную настройку элементов интерфейса под различные экраны современных мобильных устройств. Они не привязаны к одному разрешению и могут менять конфигурацию в зависимости от диагонали устройства пользователя.

2.4 Проектирование Тестов

Тесты в приложении разделены по темам и служат для лучшего понимания учениками материала.

Прогресс по прохождению теста приведен на рисунке 13.

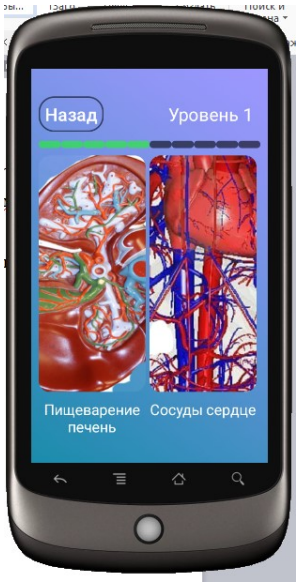

Рисунок 13 – Прогресс по прохождению теста

Прогресс по прохождению теста отображается в виде заполнения шкалы в верху экрана. Правильный ответ заполняет зеленым одно деление, неправильный - уменьшает прогресс шкалы на два деления.

Результат прохождения теста приведен на рисунке 14

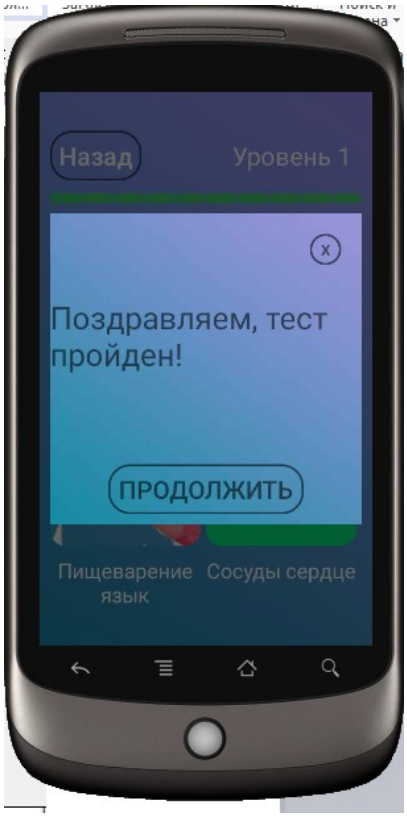

Рисунок 14 – Результат прохождения теста

Данное диалоговое окно позволяет перейти к следующему тематическому тесту или вернуться к общему меню тестов.

2.5 Проектирование программных средств

На основе анализа предметной области был разработан набор классов, приведенный на рисунке 15.

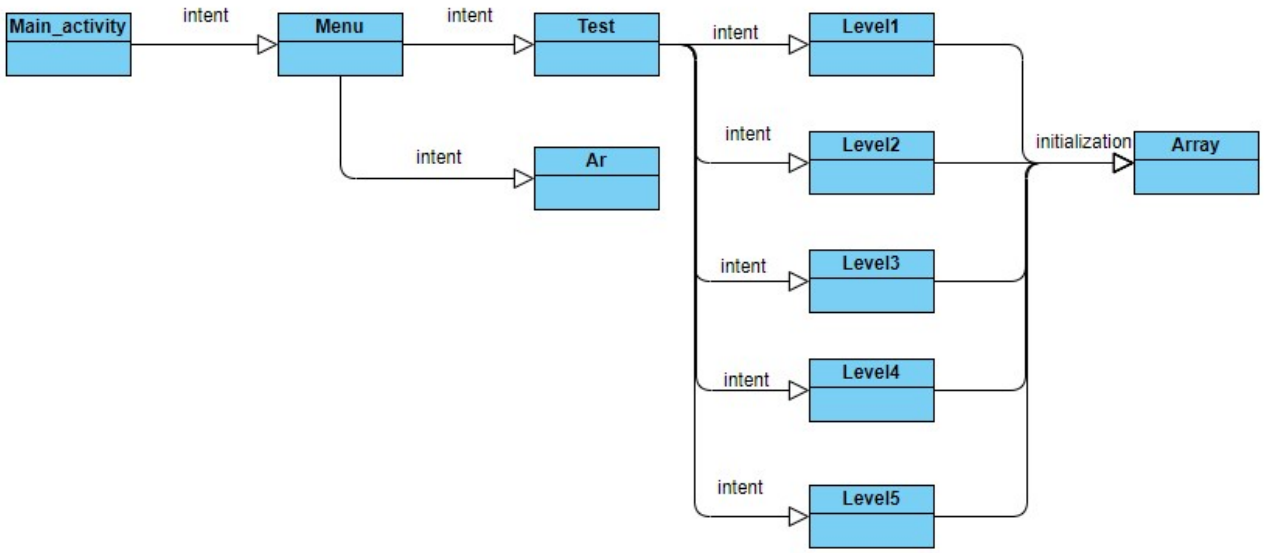

Рисунок 15 – Диаграмма классов приложения

На данной диаграмме приведены следующие классы:

– Main_activity представляет окно перехода в меню;

– Meny предназначен для выбора основной деятельности пользователя;

– Test представляет основное окно выбора задания;

– Ar представляет окно с дополненной реальностью;

– Level1 представляет собой первый тест;

– Level2 представляет собой второй тест;

– Level3 представляет собой третий тест;

– Level4 представляет собой четвертей тест;

– Level5 представляет собой пятый тест;

– Array представляет структуру хранения задания для тестов.

Таким образом, исходя из анализа предметной области, были выделены основные классы, а также связи между ними. Следует сказать, что классы Level отличаются по тематике.

Для поддержания отдельных элементов интерфейса не требуется собственный класс. Достаточно добавить описание вызываемых диалоговых окон в уже существующих классах, указа ресурсы характерные для этого класса. Так можно вторично использовать уже инициированные элементы.

2.6 Планирование проекта и оценка бюджета

Для планирования проекта были продуманы и добавлены в Gantter [18] основные задачи, приведенные на рисунке 16.

	Реализация проекта	1день	01/04/2020	01/04/2020		Программист,Аналитик,Тестировщик,I
	Начало проекта	1день	01/04/2020	01/04/2020		
	Анализ предметной области и требований	4дней	02/04/2020	07/04/2020	1,2	Аналитик
	Анализ программ аналогов	1день	02/04/2020	02/04/2020	2	
	Анализ бизнес-процессов предметной области	1день	03/04/2020	03/04/2020	4	
	Выбор инструментальных средств	1день	06/04/2020	06/04/2020	5	
	Анализ предметной области и требований завершен	1день	07/04/2020	07/04/2020	6	
	Проектирование приложения	9дней	08/04/2020	20/04/2020	7	Программист
	Проектирование базы данных	2дней	08/04/2020	09/04/2020		Программист
	Проектирование пользовательского интерфейса	2дней	10/04/2020	13/04/2020	9	Программист,Аналитик
	Проектирование тестов	2дней	14/04/2020	15/04/2020	10	Программист,Аналитик
	Проектирование программных средств	2дней	16/04/2020	17/04/2020	11	Программист,Аналитик
	Проектирование приложения завершено	1день	20/04/2020	20/04/2020	12	
	Реализация системы	20дней	21/04/2020	18/05/2020		Программист,Аналитик,Бумага[1]
	Реализация физической базы данных	6дней	21/04/2020	28/04/2020	8	Программист
	Реализация серверной части приложения	6дней	29/04/2020	06/05/2020	15	Программист
	Реализация программных средств	7дней	07/05/2020	15/05/2020	16	
	Реализация приложения завершена	1день	18/05/2020	18/05/2020	17	
	Тестирование системы и анализ качества кода	12дней	19/05/2020	03/06/2020		Программист
	Функциональное тестирование	3дней	19/05/2020	21/05/2020	14	Тестировщик
	Структурное тестирование	3дней	22/05/2020	26/05/2020	20	Тестировщик
	Модульное тестирование	3дней	27/05/2020	29/05/2020	21	Тестировщик
	Анализ качества кода	2дней	01/06/2020	02/06/2020	22	Тестировщик,Программист
	Тестирование системы и анализ качества кода завершены	1день	03/06/2020	03/06/2020	23	
	Завершение проекта	8дней	04/06/2020	15/06/2020		Бумага[1],Тестировщик
	Составление документации	3дней	04/06/2020	08/06/2020	19	Тестировщик
	Внедрение проекта	4дней	09/06/2020	12/06/2020	26	Программист
	Проект завершен	1день	15/06/2020	15/06/2020	27	
	Конец проекта	1день	16/06/2020	16/06/2020	25	

Рисунок 16 – Задачи для реализации проекта

Во время разработки проекта нужно внимательно следить за расходами. Многие ресурсы могут быть израсходованы не по назначению и привести к экономическим потерям компании. Основные ресурсы, необходимые для реализации проекта, приведены на рисунке 17.

	Название ресурса	Тип	Едини измер матер	Крат назв	Группа	Стандартная ставка	Ставка сверхурочн	Затраты на использ.	Начисление	Базовый календарь
1	Программист	Трудовой		П		27 000,00р./месяц	200,00р./час	0,00р.	Пропорциональ	Стандартный
2	Аналитик	Трудовой		А		20 000,00р./месяц	150,00р./час	0,00р.	Пропорциональ	Стандартный
3	Тестировщик	Трудовой		Т		15 000,00р./месяц	150,00р./час	0,00р.	Пропорциональ	Стандартный
4	Бумага	Материальный		Б		250,00р.		0,00р.	Пропорциональ	

Рисунок 17 – Ресурсы, необходимые для реализации проекта

Для каждой задачи были распределены материальные и человеческие ресурсы.

Для наиболее наглядного отображения плана и графика работ по проекту, была построена диаграмма Ганта, которая приведена на рисунке 18.

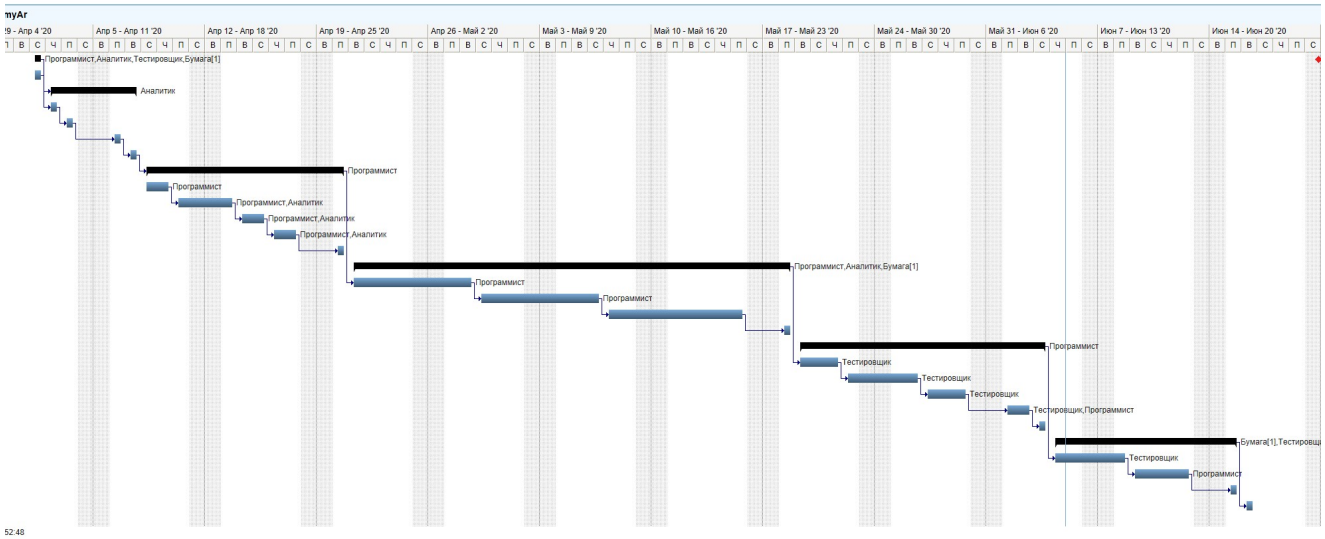

Рисунок 18 – Диаграмма Ганта

По диаграмма Ганта можно проследить нагрузку на персонал и расход материалов, для разработки. Можно заметить, что была небольшая переработка с проектированием проекта. Статистика выполнения заданий приведена на рисунке 19.

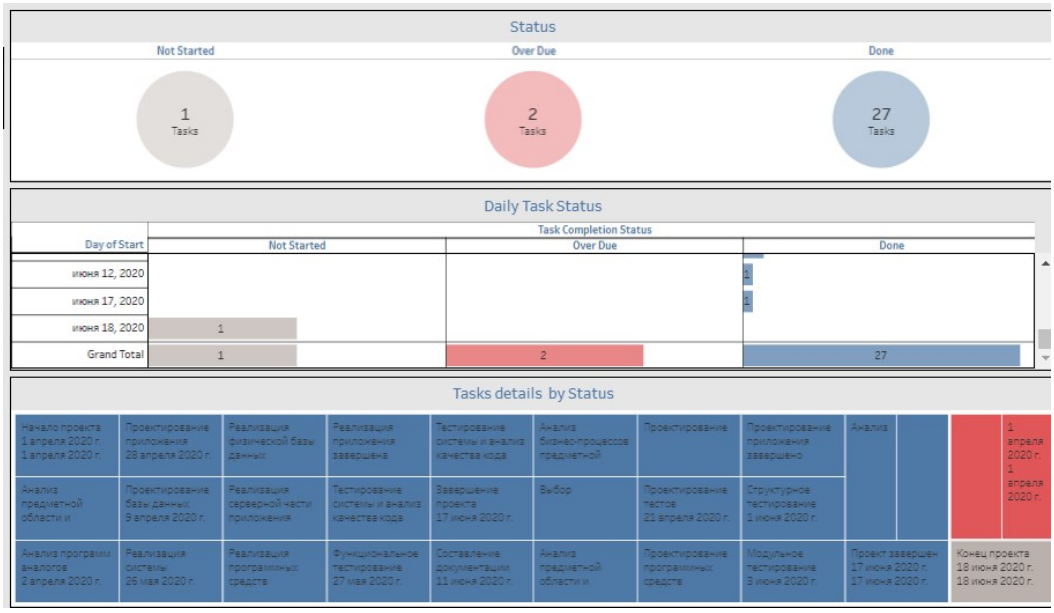

Рисунок 19 – Статистика выполнения заданий

Статистика помогает понять, что не было перерывов в разработке и время сотрудники были заняты на протяжении всего проекта. Временная шкала разработки приведена на рисунке 20.

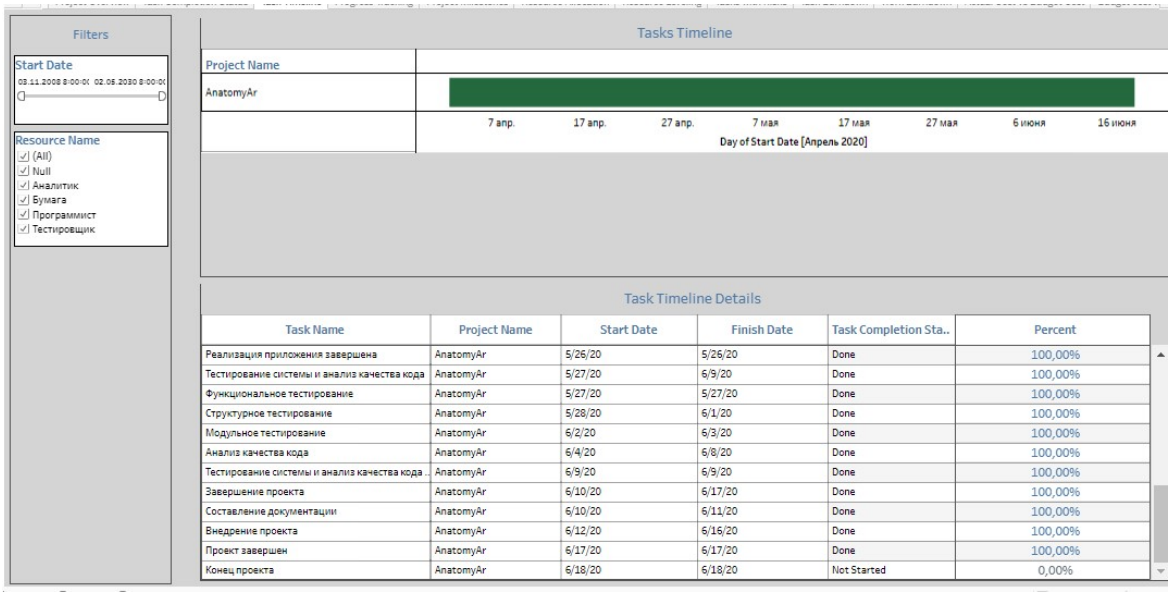

Рисунок 20 – Временная шкала разработки

Можно заметить, что наибольшая нагрузка приходилась на программиста во время работы. Он участвовал в проекте 280 часов из общих 660. Это можно отследить на схеме прогресса выполнения заданий, изображенной на рисунке 21.

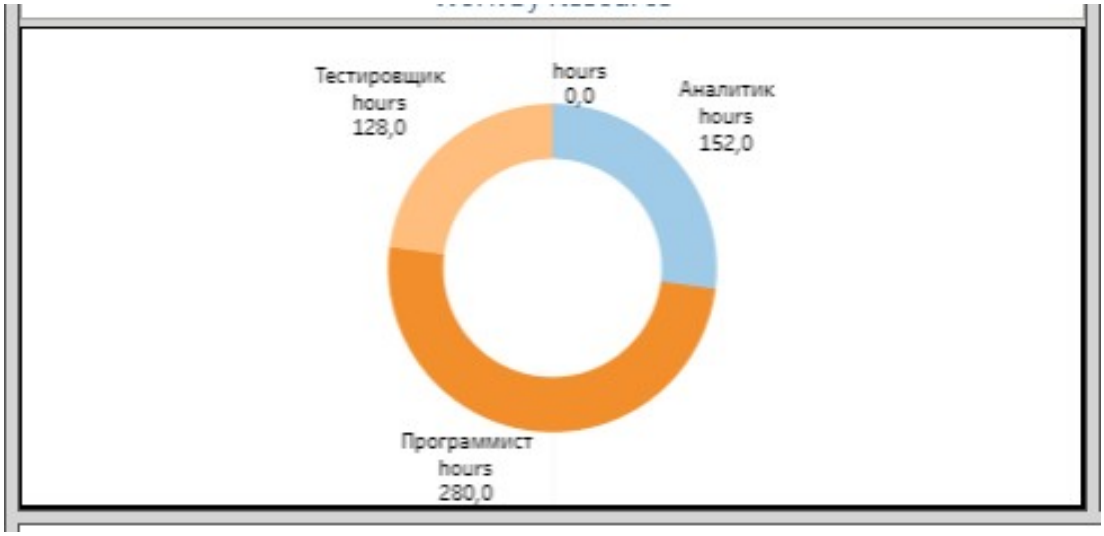

Рисунок 21 – Схеме прогресса выполнения заданий

Важно отметить, что наибольшие затраты по времени были в начале проекта и приходились на его проектирование. По схеме занятости ресурсов, на рисунке 22, можно увидеть основное распределение временных ресурсов сотрудников.

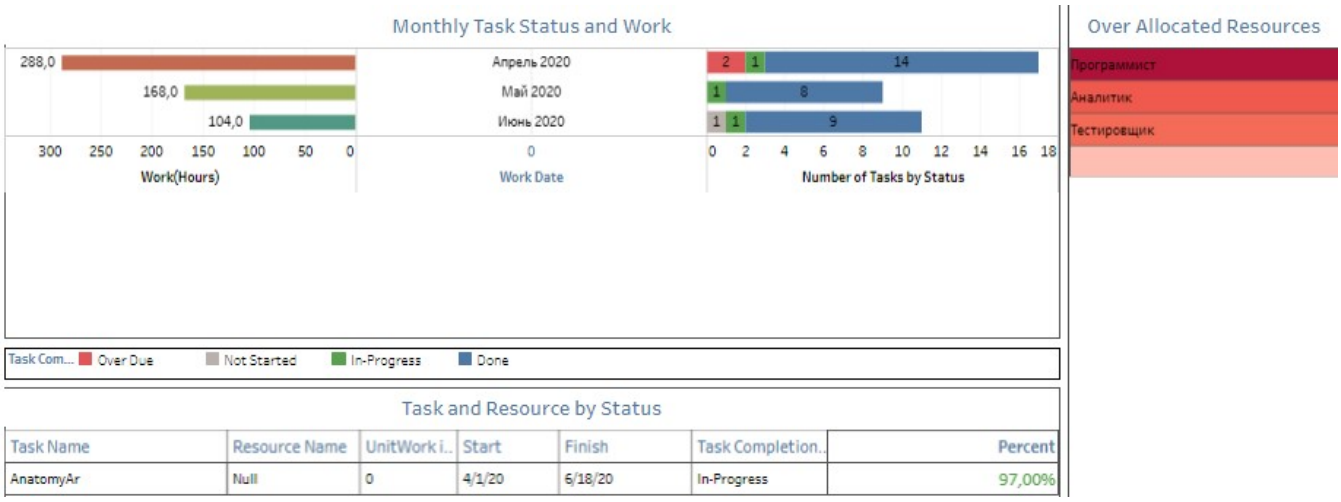

Рисунок 22 – Схеме занятости ресурсов

В разработке приложения были изменения в скорости решения поставленных задач. Они связаны с особенностями поставленных задач. Отследить изменения можно на схеме выполнения оставшихся задач, приведенной на рисунке 23.

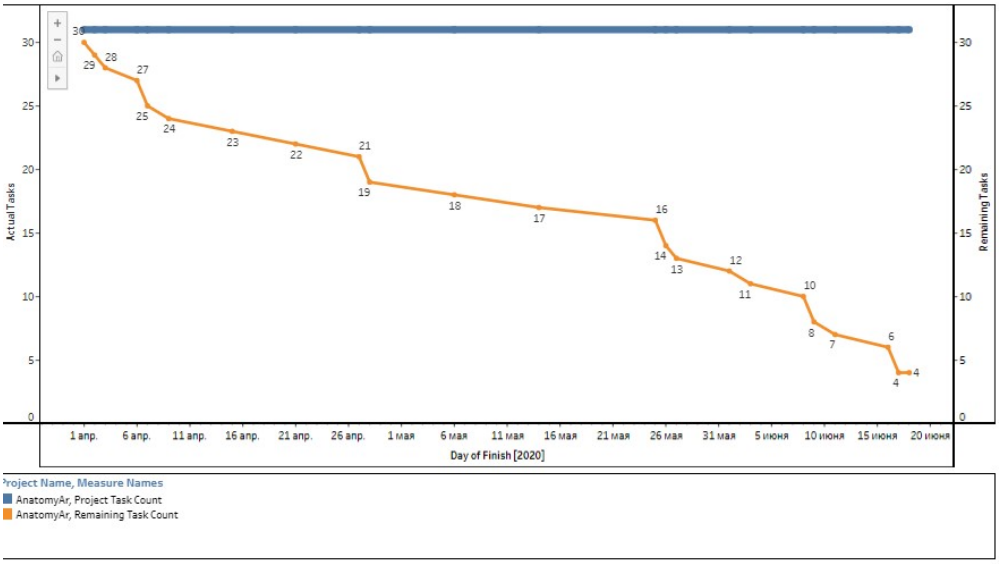

Рисунок 23 – Схеме выполнения оставшихся задач

Изменения в скорости решения поставленных задач можно проследить и в скорости выполнения задач сотрудниками. Они связаны с загруженностью работников в этот период времени и оставшиеся время разработки от общего. Отследить изменения можно на схеме выполнения работы, приведенной на рисунке 24.

Рисунок 24 – Схеме выполнения оставшейся работы

С экономической точки зрения нужно отметить самые затратные по ресурсам этапы разработки приложения, для дальнейшей их оптимизации. Отследить этапы разработки в денежном эквиваленте можно на схеме затраты по сравнению с базовыми затратами приведенной на рисунке 25.

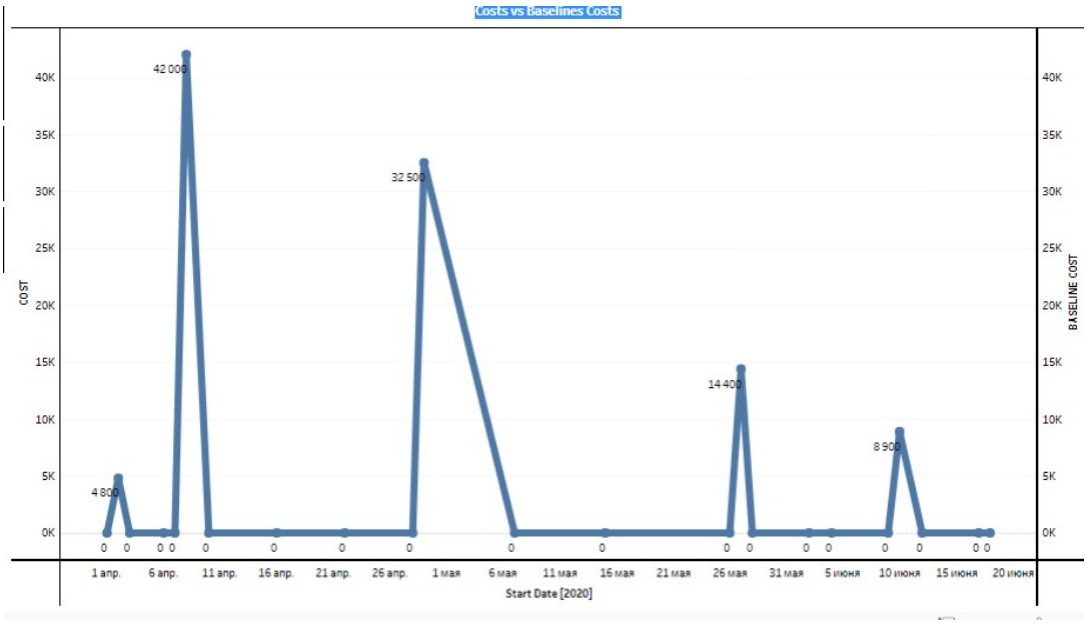

Рисунок 25 – Затраты по сравнению с базовыми затратами

По схеме затраты по сравнению с базовыми затратами, можно определить, что основное повышение стоимости разработки приходилось на время переработки сотрудников и выполнение ими двух задач: анализ предметной области и проектирование приложения. В дальнейшем стоит учитывать подобные ситуации, для их избежания. Очень важно сохранять здоровье сотрудников и экономическую выгоду от проекта.

В целом затраты по временным, человеческим и материальным ресурсам не превысили прогнозируемых показателей.

Важно увидеть итог по разрабатываемому проекту. Так можно оценить проделанную работу. Общая статистика по проекту приведена на рисунке 26.

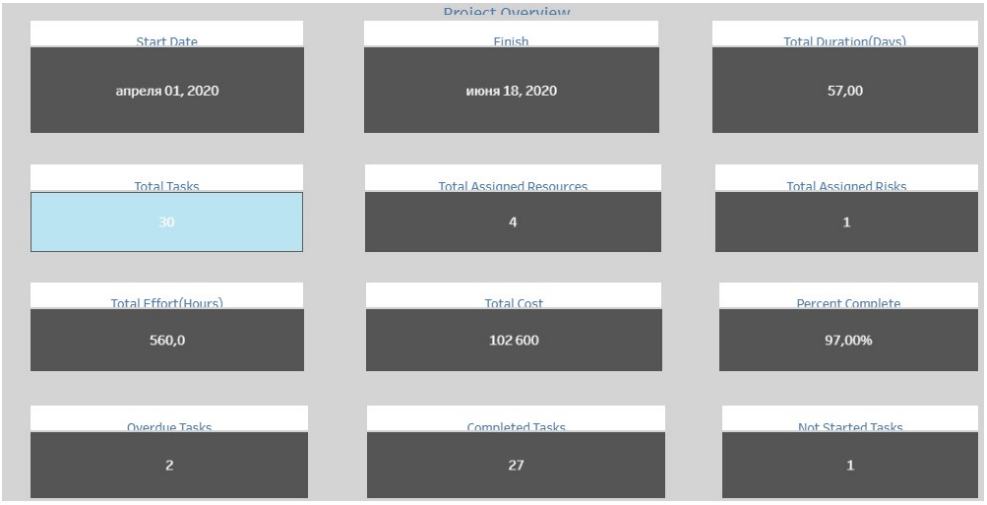

Рисунок 26 – Статистика по проекту

Таким образом, из полученной статистики можно сделать вывод, что длительность реализации проекта составляет 57 дней, все поставленные задачи решены полностью, и проект завершен на 100%. Затраты составляют 102600 руб., что говорит об экономической выгодности покупки разработанного продукта относительно существующих на рынке наиболее популярных аналогов.

3 Реализация системы

3.1 Реализация серверной части приложения

Этап реализации заключается в увязке логической структуры БД и физической среды хранения с целью наиболее эффективного размещения данных, т.е. отображении логической структуры БД в структуру хранения[14]. Перечень основных таблиц, необходимых для реализации приложения обучения курсу анатомии с дополненной реальностью приведен в таблице 4.

Таблица 4 – Описание основных таблиц

№	Наименование таблицы	Описание
1	journal	Содержит сведения о действиях пользователя
2	action	Списочный состав возможных действий
3	model	Содержит иерархию элементов модели
4	lang	Списочный перечень доступных языков
5	task	Содержит сведения о задании
6	student_task	Задание закрепленное за студентом
7	result	Содержит сведения о результатах прохождения задания
8	quastion	Вопросы для задания
9	answer	Ответы для задания
10	test_list	Список заданий и ответов к ним

Описание связей между таблицами представлено в таблице 5.

Таблица 5 – Описание связей между основными таблицами

№	Наименование таблицы 1	Наименование таблицы 2	Описание	Мощность
1	action	journal	Действия записываются в журнал	1:M
2	model	journal	Модель записывается в журнал	1:M
3	model	model	Иерархия	M:1
4	model	task	Задание с моделью	1:M
5	lang	task	Язык у задания	1:M
6	task	student_task	Задание для нескольких студентов	1:M
7	student_task	result	Результат прохождения задания	1:M
8	task	quastion	Вопросы в задании	1:M
9	quastion	answer	Ответы к вопросам	1:M
10	quastion	test_list	Вопросы списки ответов	1:M
11	answer	test_list	Ответы и списки ответов	1:M

Полное описание атрибутов таблиц приведено в приложении Б. Схема физической модели, построенная в CA Data Modeler, приведена на рисунке 27.

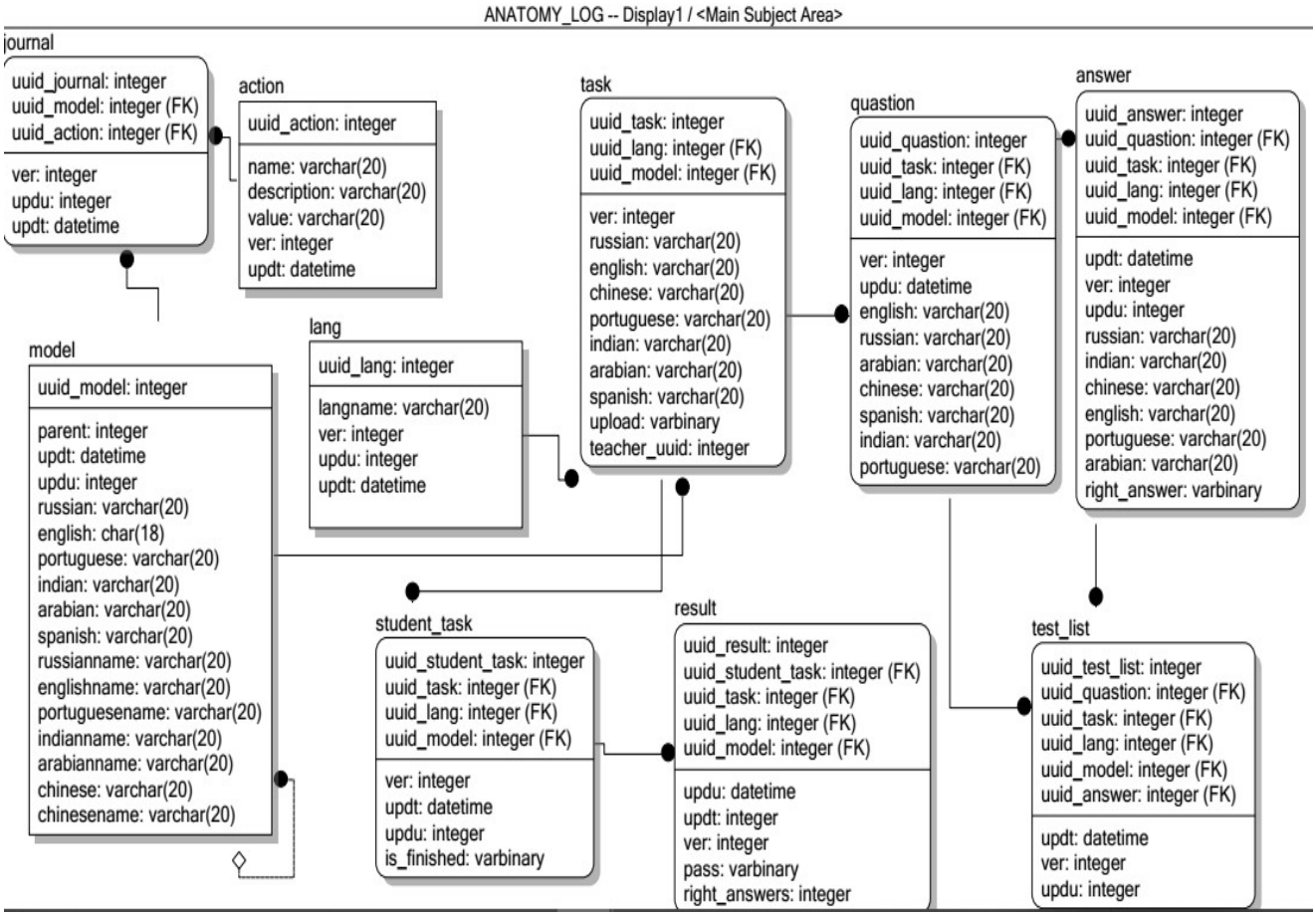

Рисунок 27 – Схема физической модели базы данных

Для реализации обмена запросами между базой данных созданной в PostgresSQL и развернутой на сервере была использована мотодология RESTful. Она позволила обмениваться сообщениями и передает состояния представлений со стороны сервера на мобильное устройство пользователя. Разработанная структура для осуществления технологии представлена в приложении В.

Ее реализация связана с обменом между клиентом и сервером пакетов, содержащих состояние сущностей БД, расположенной на стороне сервера. Это обеспечивает быстрое взаимодействие и малую нагрузку на сеть. Подобная технология широко распространена на рынке мобильных приложений и онлайн сайтов. Это говорит об актуальности технологии.

Запросы разворачивают на стороне сервера с помощью. Пример реализации технологии RESTful представлен на рисунке 28.

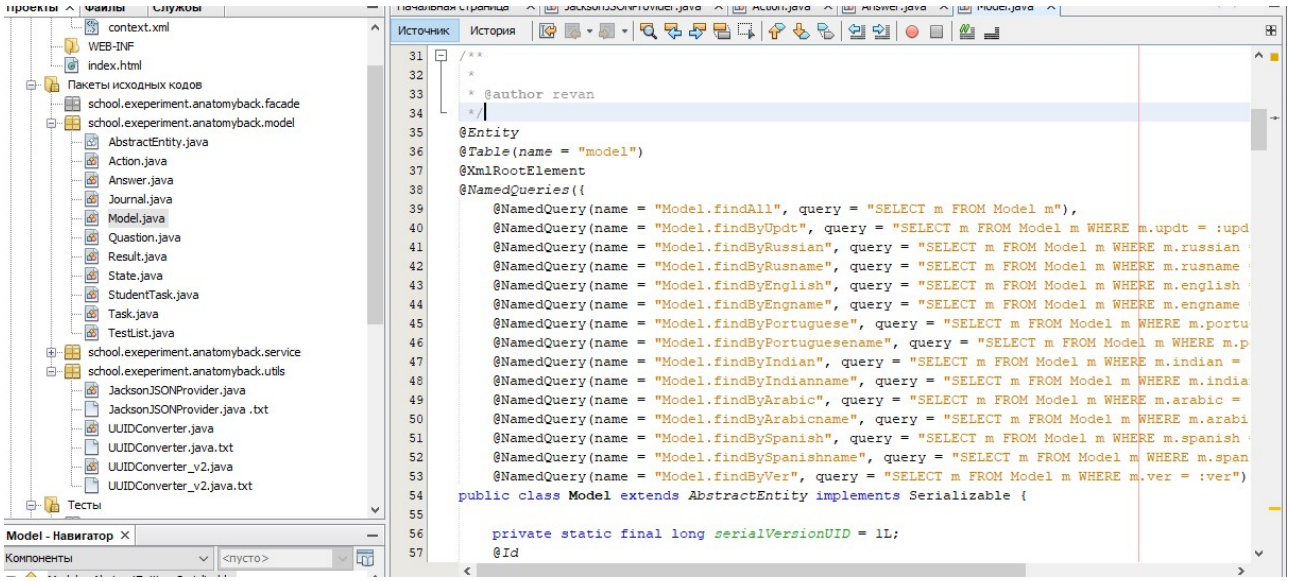

Рисунок 28 – Технология RESTful

Все сущности разработанной базы данных были описаны на языке программирования Java в среде разработки NetBeans IDE 8.2.

3.2 Реализация пользовательского приложения

Разработанное приложение обладает несколькими особенностями, выделяющих его на фоне конкурентов:

– поддержка единой платформы Experiment;

– поддержка технологии дополненной реальности от Google - ARCore;

– интерактивные тестовые задания;

– поддержка широкого круга мобильных устройств;

– поддержка нескольких пользовательских языков;

– обучающая направленность;

– поддержка технологии RESTful.

3.3 Реализация поддержки платформы Experiment

Для использования всех преимуществ обучающей платформы Experiment пользовательское приложение проведет проверку устройства на наличие Experiment Starter, изображенный на рисунке 29. Он отвечает за единый аккаунт пользователя для всех приложений.

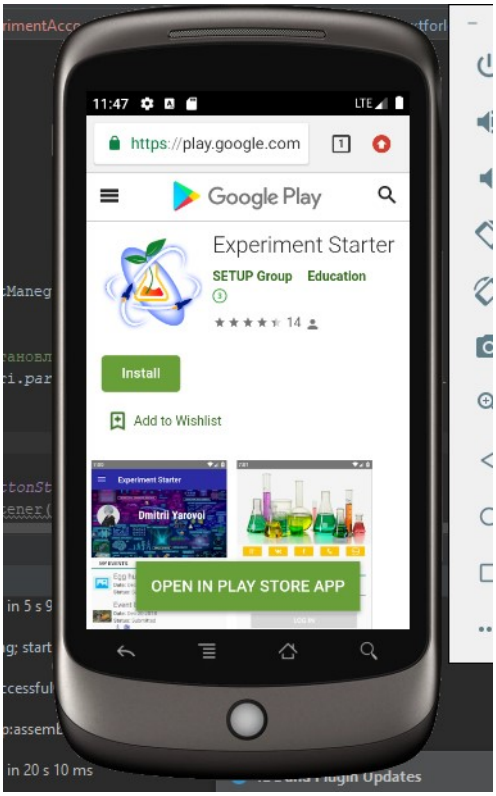

Рисунок 29 – Переход к Experiment Starter, из приложения

Работа перехода в Google play для установки Experiment Starter описана ниже. Платформа Experiment затрагивает не только рынок мобильных приложений для обучения, но и служит удобным местом взаимодействия учеников и учителей.

На рисунке 30, изображен сайт платформы Experiment. Она предназначен для пользователей со всего мира.

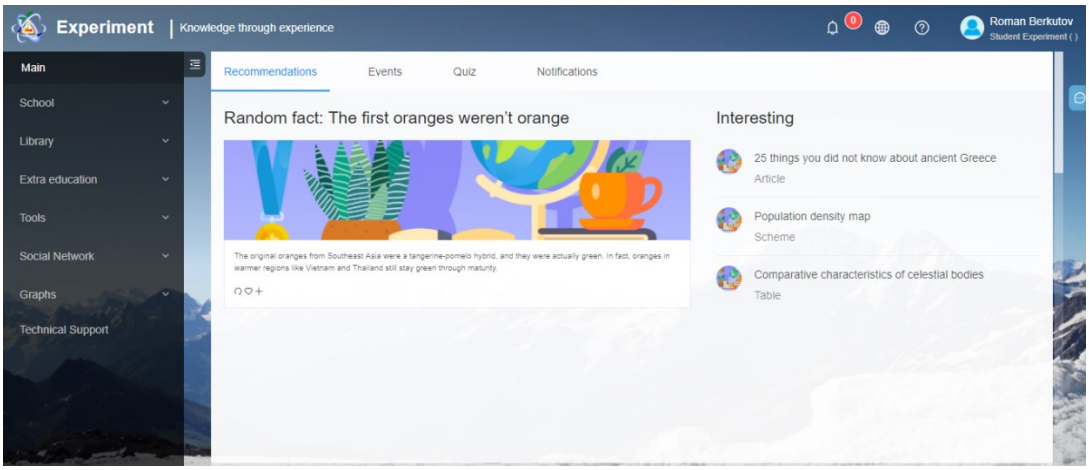

Рисунок 30 – Сайт Experiment

Платформа поддерживает единый аккаунт для сайта и приложений. Если вы авторизованы в Experiment Starter, то вам не прийдется проходить повторную авторизацию в каждом отдельном приложении платформы.

3.4 Реализация локализации приложения

Для конечного пользователя важен не только удобный интерфейс, единство платформы, но и поддержка его родного языка. С этой целью приложение было локализовано на разные языки. Приложение с локализаций русского языка приведена на рисунке 31.

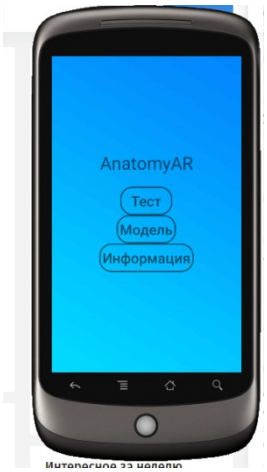

Рисунок 31 – Интерфейс на русском языке

На рисунке 32 изображен интерфейс с поддержкой английского языка. Приложение может взять информацию о вашей локализации из данных аккаунта или с вашего мобильного устройства, на котором установлено приложение.

Рисунок 32 – Интерфейс на английском языке

Не только элементы главного миню были переведены, но и описание частей человеческого организма, как показано на рисунке 33.

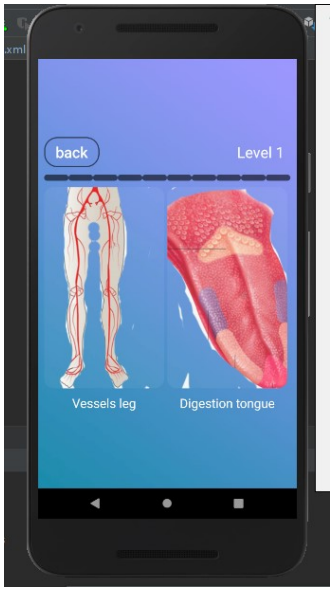

Рисунок 33 – Интерфейс теста на английском языке

Android Studio обладает удобным инструментарием для облегчения локализации мобильных приложений.

4 Тестирование системы и анализ качества кода

4.1 Тестирование по принципу белого ящика

Тестирование – важный этап в разработке проекта. Необходимо проверить не только качество кода продукта, но и его работоспособность в различных условиях с различными переменными. Для проверки программного обеспечения существует несколько основных методов. Одним из таких является тестирование по принципу белого ящика [19].

Тестирование по принципу белого ящика характеризуется тем, как тесты покрывают логику (исходный текст программы). Тестирование по принципу белого ящика (или структурное тестирование) включает в себя метод покрытия операторов, метод покрытия условий и метод покрытия маршрутов.

Тестирование может быть проведено на примере прецедента диаграммы вариантов использования «Посмотреть анатомию человека» (рисунок 34 – 36).

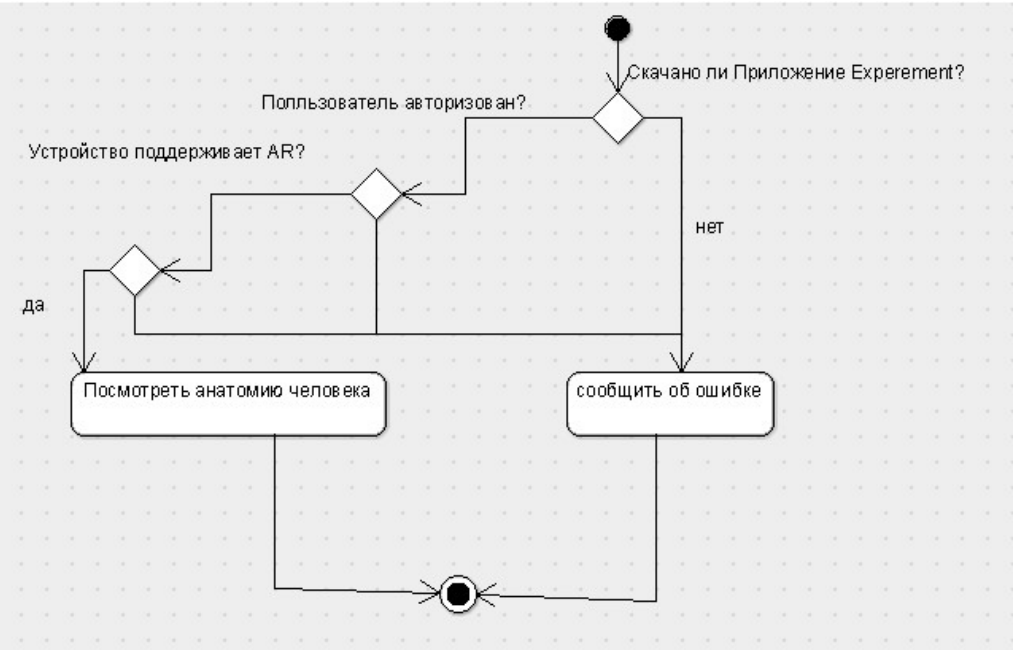

Рисунок 34 – Блок-схема прецедента «Посмотреть анатомию человека»

Данный вариант позволяет проследить ход работы ARCore на мобильном устройстве приложения. На первом этапе проходит проверка

наличия общего для образовательной платформы. Тестирование будет проводится с помощью функционального графа (рисунок 35) и управляющего графа (рисунок 36).

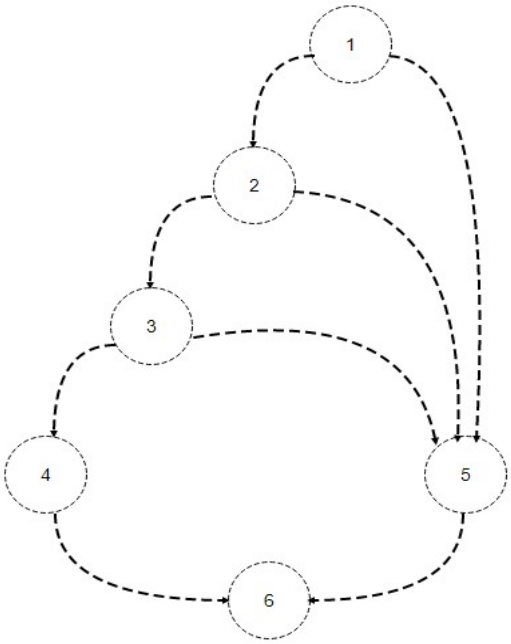

Рисунок 35 – Функциональный граф прецедента «Посмотреть анатомию человека»

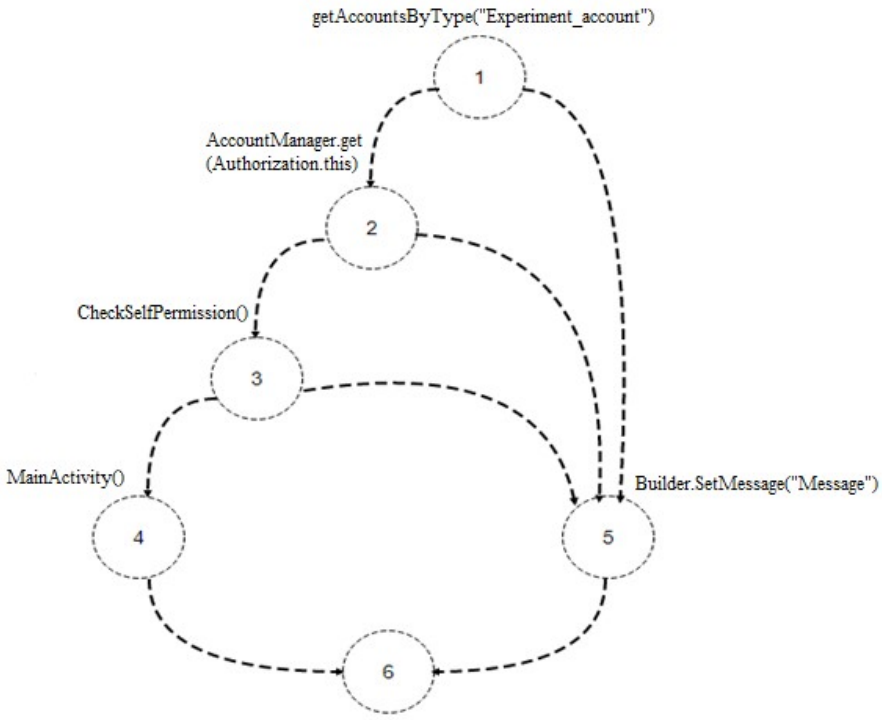

Рисунок 36 – Управляющий граф прецедента «Посмотреть анатомию человека»

Графы позволяют наглядно рассмотреть сценарий работы приложения и помогают разработчику предусмотреть различные ситуации.

4.2 Метод покрытия операторов

Критерии покрытия операторов подразумевает выполнение каждого оператора программы, по крайней мере, один раз [20].

Для выполнения каждого оператора было составлено два теста. Результаты приведены в таблице 6.

Таблица 6 – Результат тестирования методом покрытия операторов

Операторы	Тест	Фактический результат	Результат тестирования
1, 2, 3, 4	Приложение скачано, Пользователь авторизован, телефон поддерживает режим	Орган успешно выбран.	Успешно.
1, 5, 6	Приложение не скачано.	Орган не выбран.	Успешно.

Можно понять, что программно предусмотренно не сколько путей решения возможных ситуаций.

4.3 Метод покрытия условий

Суть метода покрытия условий заключается в том, чтобы составить число тестов, достаточное для того, чтобы все возможные результаты каждого условия в решении выполнялись, по крайней мере, один раз. Также должно соблюдаться условие выполнения каждого оператора, по крайней мере, один раз.

Для выполнения каждого условия было составлено четыре теста. Они должны затронуть основные возможные варианты условий выполнения задачи. Результаты приведены в таблице 7.

Таблица 7 – Результат тестирования методом покрытия условий

Условия	Тест	Фактический результат	Результат тестирования
1 – нет	Приложение не скачано.	Вывод ошибки приложения.	Успешно.
1 – да, 2 – нет	Приложение скачано, Пользователь не авторизован.	Вывод ошибки авторизации.	Успешно.
1 – да, 2 – да, 3 – нет	Приложение скачано, Пользователь авторизован, Устройство не поддерживает режим.	Вывод ошибки в получении прав.	Успешно.
1 – да, 2 –да, 3 – да	Приложение скачано, Пользователь авторизован, Устройство поддерживает режим.	Орган успешно выбран.	Успешно.

Можно понять, что программно предусмотренно не сколько путей решения возможных ситуаций.

4.4 Метод покрытия маршрутов

Для выполнения всех маршрутов было составлено четыре теста. Они предполагают охват всех возможных вариантов выполнения задачи. Для

нескольких возможных «маршрутов», через которые может пройти система, будут рассмотрены различные варианты исхода теста. В результатах тестирования будет описан фактический результат прохождения выбранного маршрута. По итогу фактического результата выставляется критерий [21]. Результаты приведены в таблице 8.

Таблица 8 – Результат тестирования методом покрытия условий

Маршрут	Тест	Фактический результат	Результат тестирования
1, 5, 6	Приложение не скачано.	Вывод ошибки приложения.	Успешно.
1, 2, 5, 6	Приложение скачано, Пользователь не авторизован.	Вывод ошибки авторизации.	Успешно.
1, 2, 3, 5, 6	Приложение скачано, Пользователь авторизован, Устройство не поддерживает режим.	Вывод ошибки прав.	Успешно.
1, 2, 3, 4, 6	Приложение скачано, Пользователь авторизован, Устройство поддерживает режим.	Орган успешно выбран.	Успешно.

Метод покрытия условий позволяет избежать негативных условий выполнения задачи.

4.5 Метрики кода

Ме́трика програ́ммного обеспе́чения (англ. software metric) – мера, позволяющая получить численное значение некоторого свойства программного обеспечения или его спецификаций [22].

Количество строк исходного кода (Lines of Code – LOC, Source Lines of Code – SLOC) является наиболее простым и распространенным способом оценки объема работ по проекту (Листинг приведен в конце отчета).

Получаем:

- количество строк исходного кода=1200;
- число пустых строк=120;
- число строк, содержащих комментарии=30;
- процент комментариев=0,025;
- среднее число строк для функций=12;
- среднее число строк, содержащих исходный код для функций=15;
- среднее число строк для модулей.=20.

Метрики цикломатической сложности по Мак-Кейбу – является одним из наиболее распространенных показателей оценки сложности программных проектов. Данный показатель был разработан ученым Мак-Кейбом в 1976 г., относится к группе показателей оценки сложности потока управления программой и вычисляется на основе графа управляющей логики программы (control flow graph). Данный граф строится в виде ориентированного графа, в котором вычислительные операторы или выражения представляются в виде узлов, а передача управления между узлами – в виде дуг.

Показатель цикломатической сложности позволяет не только произвести оценку трудоемкости реализации отдельных элементов программного проекта и скорректировать общие показатели оценки длительности и стоимости проекта, но и оценить связанные риски и принять необходимые управленческие решения.

Упрощенная формула вычисления цикломатической сложности представляется следующим образом:

$$C = e - n + 2,$$

где e – число ребер;

n – число узлов на графе управляющей логики (рисунок 37).

Как правило, при вычислении цикломатической сложности логические операторы не учитываются. Ее можно проследить с помощью графа управляющего логикой (рисунок 37).

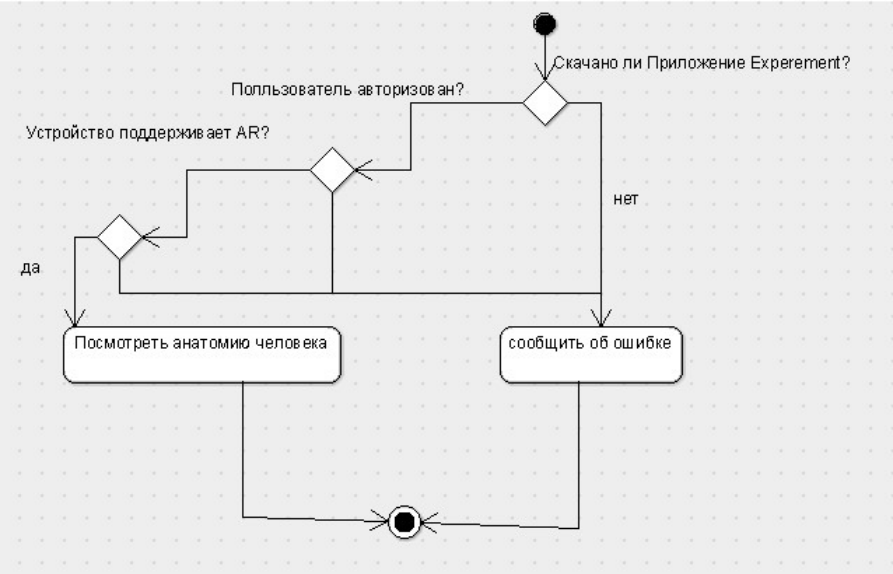

Рисунок 37 – граф управляющий логикой

В итоге мы получили что цикломатическая сложность равна 2.

Метрика Холстеда относится к метрикам, вычисляемым на основании анализа числа строк и синтаксических элементов исходного кода программы.

Основу метрики Холстеда составляют измеряемые характеристики программы:

– NUOprtr (Number of Unique Operators) – число уникальных операторов программы, включая символы-разделители, имена процедур и знаки операций (словарь операторов), равное 15;

– NUOprnd (Number of Unique Operands) – число уникальных операндов программы (словарь операндов), равное 10;

– Noprnd (Number of Operators) – общее число операндов в программе, равное 20.

На основании этих характеристик рассчитываются оценки:

– словарь программы (HPVoc) равный 25;

– длина программы (Halstead Program Length, HPLen) равное 90;

– объем программы (Halstead Program Volume, HPVol) равный 125,814;

– сложность программы (Halstead Difficulty, HDiff) равная 52,5.

На основе показателя HDiff предлагается оценивать усилия программиста при разработке при помощи показателя HEff (Halstead Effort): HEff равное HDiff умножить на HPVol. ИЗ чего можем сделать вывод, что расчетные значения не выходят за ожидаемые нормы.

Код приложения представлен в приложении А.

4.6 Конфигурационное тестирование под разные модели.

Конфигурационное тестирование (Configuration Testing) – специальный вид тестирования, направленный на проверку работы программного обеспечения при различных конфигурациях системы [23].

Для тестирования приложения в Android Studio были использованы виртуальные машины с различными характеристиками. Приложение поддерживает оперативную систему Android. Конфигурации виртуальных машин приведены на рисунке 38.

Type	Name	Play Store	Resolution	API	Target
	Copy_of_Nexus 5X API 28 x86		1080 × 1920: 420dpi	28	Android 9.0 (Google APIs)
	Nexus One API 28		480 × 800: hdpi	28	Android 9.0 (Google APIs)
	Pixel API 27		1080 × 1920: 420dpi	27	Android 8.1 (Google Play)

Рисунок 38 – конфигурации виртуальных машин

Android Studio позволяет проверить отображение пользовательского интерфейса для различных конфигураций экрана, приведенных на рисунке 39.

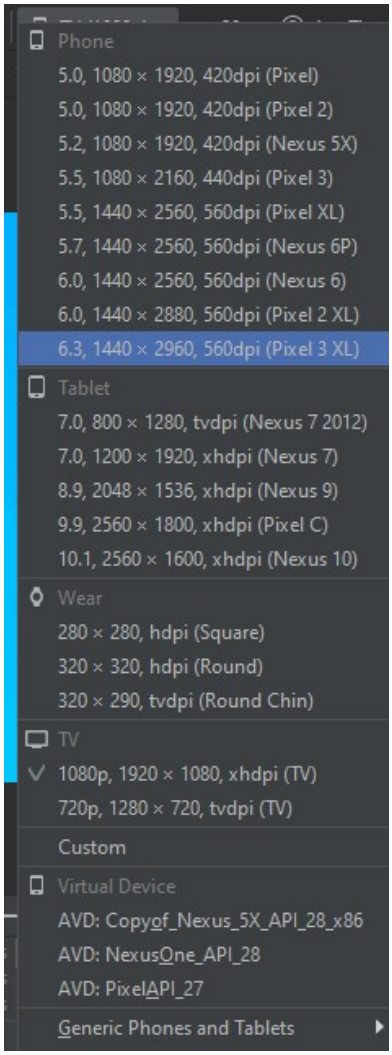

Рисунок 39 – Различные конфигурации экрана

Пользовательский интерфейс был сделан методом резиновой верстки. Это позволило обеспечить поддержку всех актуальных платформ.

Заключение

В ходе выполнения выпускной квалификационной работы была изучена предметная область и выполнена задача на разработке системы, актуальной на рынке приложений: разработана серверная и клиентская часть на языке Java с использованием фреймворка ARCore, использующего MVC структуру и технологию Rest. Для создания приложения была выбрана среда разработки Android Studio. Для реализации базы данных были выбраны объектно-реляционная СУБД – PostgreSQL и свободный кроссплатформенный менеджер баз данных – DBeaver.

Все рассматриваемые методики, архитектуры и технологии соответствуют актуальным требованиям рынка. Было важно учитывать комплексность задачи и возможность включения разрабатываемого приложения в большую платформу обучения.

Для этого были реализованны следующие подзадачи:

- спроектирована и реализована база данных, используя PostgreSQL;
- сформированы классы ИС и связи между ними;
- разработаны контроллеры бизнес - логики;
- созданы сервисы для решения распространённых задач внутри ИС;
- реализован слой представления в виде приложения;
- проведено основное проектирование приложения для обучения по курсу анатомии с использованием технологии дополненной реальности;
- реализована поддержка нескольких языков.сформировать классы;
- реализована поддержка широкого круга мобильных устройств;
- реализована поддержка основной платформы Experiment;
- получена статистика проекта, по которой можно сделать вывод, что длительность реализации проекта составляет 57 дней, все поставленные задачи решены полностью, и проект завершен на 100%.

Общие затраты составили 102600 руб;

После выполнения подзадач было проведено комплексное тестирование приложения и его серверной части с базой данных. Это

необходимо для последующей интеграции приложения в обучающие организации и возможного коммерческого использования.

Список использованных источников

1. Дополненная реальность [Электронный ресурс]. – URL: https://habr.com/ru/post/419437/– (дата обращения: 23.04.2020).

2. Технологии VR и AR в образовании [Электронный ресурс]. – URL: https://habr.com/ru/company/appodeal/blog/265695/ (дата обращения: 23.04.2020).

3. Классификация мобильных приложений [Электронный ресурс]. – URL: https:// studref.com/ 384493/ informatika/ klassifikatsiya_mobilnyh_prilozheniy (дата обращения: 20.04.2020).

4. Проблемы мобильных приложений и способы их решения [Электронный ресурс]. – URL: https://habr.com/ru/company/appodeal/blog/265695/ (дата обращения: 22.04.2020).

5. Model–View–Controller в .Net [Электронный ресурс]. – URL:https://rsdn.org/article/patterns/ModelViewPresenter.xml – (дата обращения: 20.02.2020).

6. Erwun data modeller [Электронный ресурс]. – URL: https://erwin.com/products/erwin-data-modeler/ (дата обращения: 20.04.2020).

7. UML Diagrama Tool [Электронный ресурс]. – URL:https://online.visual-paradigm.com/diagrams/features/uml-tool/ (дата обращения: 20.04.2020).

8. Кратко о типах архитектур программного обеспечения [Электронный ресурс]. – URL: https://habr.com/ru/company/1cloud/blog/424911/ (дата обращения: 20.04.2020).

9. Обзор популярных AR-фреймворков [Электронный ресурс]. – URL: https://habr.com/ru/company/lodoss/blog/358780/ (дата обращения: 20.02.2020).

10. Обзор DBeaver - свободный менеджер баз данных [Электронный ресурс]. – URL:https://ph0en1x.net/63-dbeaver-free-database-management-tool-sql.html (дата обращения: 20.02.2020).

11. IDE NetBens - среда разработки [Электронный ресурс]. URL: https://netbeans.org//
(дата обращения 15.05.2020).

12. Что такое Android studio [Электронный ресурс]. – URL: https://androfon.ru/prodvinut/chto-takoe-android-studio-gde-skachat-i-kak-ustanoit (дата обращения: 20.02.2020).

13. ИТ в образовании. [Электронный ресурс]: URL: https://studopedia.ru/9_115649_informatsionnie-tehnologii-v-obrazovanii.html (дата обращения: 15.05.2020).

14. Хомоненко, А.Д. Базы данных. Учебник для вузов / А.Д. Хомоненко, В.М. Цыганков, М.Г. Мальцев. – ДМК. : Пресс, 2013. – 232 с.

15. Аксенов, К. А. Работа с CASE-средствами BPwin, ERwin. / К. А. Аксенов, Б. И. Клебанов; Уральский гос. техн. ун-т. – Екатеринбург, 2004 – 50 с.

16. Система управления базами данных SQL Server [Электронный ресурс]. URL: https://professorweb.ru/my/sql-server/2012/level1/1_1.php/
(дата обращения 15.05.2020).

17. Анатомия человека [Электронный ресурс]. – URL: https://www.oum.ru/literature/anatomiya-cheloveka/
(дата обращения 15.05.2020).

18. Веб сервис для управления проектами - Gantter [Электронный ресурс]. URL: https://www.gantter.com/pricing/
(дата обращения 15.05.2020).

19. Тестирование методом белого ящика [Электронный ресурс]. URL: https://habr.com/ru/post/422283/
(дата обращения 15.05.2020).

20. Метод покрытия операторов [Электронный ресурс]. URL: https://pandia.ru/text/78/481/44389.php
(дата обращения 15.05.2020).

21. Покрытие решений [Электронный ресурс]. URL: https://studopedia.ru/9_40672_strukturnoe-testirovanie.html (дата обращения 15.05.2020).

22. Метрика кода [Электронный ресурс]. URL: https://habr.com/ru/company/intel/blog/106082/ (дата обращения 15.05.2020).

23. Конфигурационное тестирование [Электронный ресурс]. URL: http://www.protesting.ru/testing/types/configuration.html (дата обращения 15.05.2020).

Приложение А
Листинг программы AnatomyAR
(рекомендованное)

Файл MeinActivity

```
      package school.experiment.anatomy;
import android.content.Intent;
import android.widget.Button;

import com.elementaryschool.anatomy.R;
import school.experiment.anatomy.ExperimentAccountManeger;

public class MainActivity extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);


        /*ExperimentAccountManeger ex = new ExperimentAccountManeger(this);
        String message;
        if(!ex.isHaveApplication()){
            message = "Приложение Experiment Starter не установлено";
            startActivity(new Intent(Intent.ACTION_VIEW,
Uri.parse("https://play.google.com/store/apps/details?id=school.experiment.st
arter")));
        }*/

        Button buttonStart = (Button) findViewById(R.id.buttonStart);
        buttonStart.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                try {
                    Intent intent = new Intent( MainActivity.this, Menu.class);
                    startActivity(intent); finish();
                } catch (Exception e) {
                }
            }
        });

        Window w = getWindow();
        ((Window) w).setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
WindowManager.LayoutParams.FLAG_FULLSCREEN);
    }
}
```

Файл AR1

```
      package school.experiment.anatomy;

import android.app.Activity;
import android.support.v7.app.AppCompatActivity;
import android.util.Log;
import com.google.ar.sceneform.ux.TransformableNode;

public class AR1 extends AppCompatActivity {


    private static final String TAG = MainActivity.class.getSimpleName();
    private static final double MIN_OPENGL_VERSION = 3.0;



        ArFragment arFragment;
        ModelRenderable lampPostRenderable;

    /*@Override
    protected void onResume() {
```

```
        super.onResume();
        *//*
        if (arFragment.getArSceneView().getSession() == null) {

            Session session= null;
            try {
                session = new Session(this);
            } catch (Exception e) {
                e.printStackTrace();
            }

            Config config = new Config(session);
            config.setPlaneFindingMode(Config.PlaneFindingMode.HORIZONTAL);
            config.setCloudAnchorMode(Config.CloudAnchorMode.DISABLED);
            config.setUpdateMode(Config.UpdateMode.LATEST_CAMERA_IMAGE);
            session.configure(config);


            arFragment.getArSceneView().setupSession(session);
            try {
                arFragment.getArSceneView().resume();
            } catch (Exception e) {
                e.printStackTrace();
            }
            return;
        } else {

        }*//*
    }*/

        @Override
        @SuppressWarnings({"AndroidApiChecker", "FutureReturnValueIgnored"})
        protected void onCreate(Bundle savedInstanceState) {
            super.onCreate(savedInstanceState);
            if (!checkIsSupportedDeviceOrFinish(this)) {
                return;
            }
            setContentView(R.layout.activity_ar);
            arFragment = (ArFragment)
getSupportFragmentManager().findFragmentById(R.id.arFragment);

            ModelRenderable.builder()
                    .setSource(this, Uri.parse("cardio_vascular_model.sfb"))
                   // .setSource(this, Uri.parse("cardio_vascular_model.sfb"))
                    .build()

                    .thenAccept(renderable -> lampPostRenderable = renderable)
                    .exceptionally(throwable -> {
                        Toast toast =
                                Toast.makeText(this, "Unable to load andy
renderable", Toast.LENGTH_LONG);
                        toast.setGravity(Gravity.CENTER, 0, 0);
                        toast.show();
                        return null;
                    });

            arFragment.setOnTapArPlaneListener(
                    (HitResult hitresult, Plane plane, MotionEvent motionevent)
-> {
                        if (lampPostRenderable == null){
                            return;
                        }
                        Anchor anchor = hitresult.createAnchor();
                        AnchorNode anchorNode = new AnchorNode(anchor);

anchorNode.setParent(arFragment.getArSceneView().getScene());
```

```
                        TransformableNode lamp = new
TransformableNode(arFragment.getTransformationSystem());
                        lamp.setParent(anchorNode);
                        lamp.setRenderable(lampPostRenderable);
                        lamp.select();
                    }
            );

        }
    public static boolean checkIsSupportedDeviceOrFinish(final Activity activity)
{
        if (Build.VERSION.SDK_INT < Build.VERSION_CODES.N) {
            Log.e(TAG, "Sceneform requires Android N or later");
            Toast.makeText(activity, "Sceneform requires Android N or later",
Toast.LENGTH_LONG).show();
            activity.finish();
            return false;
        }
        String openGlVersionString =
                ((ActivityManager)
activity.getSystemService(Context.ACTIVITY_SERVICE))
                        .getDeviceConfigurationInfo()
                        .getGlEsVersion();
        if (Double.parseDouble(openGlVersionString) < MIN_OPENGL_VERSION) {
            Log.e(TAG, "Sceneform requires OpenGL ES 3.0 later");
            Toast.makeText(activity, "Sceneform requires OpenGL ES 3.0 or later",
Toast.LENGTH_LONG)
                    .show();
            activity.finish();
            return false;
        }
        return true;
    }

    }
```

Файл Array

```
      package school.experiment.anatomy;

import com.elementaryschool.anatomy.R;

public class Array {
    //array blood XD
    final int[] images1 = {
           R.drawable.blood0,
            R.drawable.blood1,
            R.drawable.blood2,
            R.drawable.blood3,
            R.drawable.blood4,
            R.drawable.blood5,
            R.drawable.blood6,
            R.drawable.blood7,
            R.drawable.blood8,
            R.drawable.blood9,
            R.drawable.degestive0,
            R.drawable.degestive1,
            R.drawable.degestive2,
            R.drawable.degestive3,
            R.drawable.degestive4,
    };


    final int[] text1 = {
         R.string.lvl1text0,
            R.string.lvl1text1,
```

```
            R.string.lvl1text2,
            R.string.lvl1text3,
            R.string.lvl1text4,
            R.string.lvl1text5,
            R.string.lvl1text6,
            R.string.lvl1text7,
            R.string.lvl1text8,
            R.string.lvl1text9,
            R.string.lvl1text10,
            R.string.lvl1text11,
            R.string.lvl1text12,
            R.string.lvl1text13,
            R.string.lvl1text14,

    };

}
```

Файл ExperimentAccountMeneger

```
      package school.experiment.anatomy;


import android.accounts.Account;
        import android.accounts.AccountManager;
        import android.content.Intent;
        import android.content.pm.PackageManager;

        import school.experiment.anatomy.MainActivity;

public class ExperimentAccountManeger {

    // bundleId для поиска приложения Experiment starter в Android
    private final String mBundleId = "school.experiment.starter";

    private String mUserEmail;
    public String getUserEmail() {
        return mUserEmail;
    }

    //Текущая активити
    private MainActivity mActivity;

    private boolean mHaveApplication;

    public boolean isHaveApplication() {
        return mHaveApplication;
    }

    private boolean mHaveAccount;
    public boolean isHaveAccount() {
        return mHaveAccount;
    }

    public ExperimentAccountManeger(MainActivity mActivity) {
        this.mActivity = mActivity;
        PackageManager manager = mActivity.getPackageManager();
        Intent intent = manager.getLaunchIntentForPackage(mBundleId);

        mHaveApplication = intent != null;

        if(mHaveApplication){
            AccountManager accountManager = AccountManager.get(mActivity);
            Account[] accounts =
```

```
accountManager.getAccountsByType("experiment_account");
            mHaveAccount = accounts.length > 0;
            if(mHaveAccount){
                Account experimentAccount = accounts[0];
                String token =
accountManager.getUserData(experimentAccount,"token");
                mUserEmail = experimentAccount.name;
            }
        }
    }
}
```

Файл Menu

```
      package school.experiment.anatomy;

import android.app.Dialog;
import android.widget.TextView;

import com.elementaryschool.anatomy.R;

public class Menu extends AppCompatActivity {

    Dialog dialog;


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.menu);

        Button button_test = (Button) findViewById(R.id.button_test);
        button_test.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                //Начало конструкции
                try {
                    Intent intent = new Intent( Menu.this, Test.class);
                    startActivity(intent); finish();
                } catch (Exception e) {
                }
                //конец конструкции
            }
        });

        Button button_model = (Button) findViewById(R.id.button_model);
        button_model.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                //Начало конструкции
                try {
                    Intent intent = new Intent( Menu.this, AR1.class);
                    startActivity(intent); finish();
                } catch (Exception e) {
                }
                //конец конструкции
            }
        });

        dialog = new Dialog(this);
        dialog.requestWindowFeature(Window.FEATURE_NO_TITLE);
        dialog.setContentView(R.layout.infodialog);
        dialog.getWindow().setBackgroundDrawable(new
ColorDrawable(Color.TRANSPARENT));
```

```
        dialog.setCancelable(false);//buttom back

        //close dialog -start
        TextView btncontinue = (TextView)dialog.findViewById(R.id.btncontinue);
        btncontinue.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // btn press
                try {
                    //back to menu
                } catch (Exception e) { //no code
                }
                dialog.dismiss();// close dialog
                //btn press end
            }
        });


        Button button_inf = (Button) findViewById(R.id.button_info);
        button_inf.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                //Начало конструкции
                try {
                    dialog.show();
                } catch (Exception e) {
                }
                //конец конструкции
            }
        });
        Window w = getWindow();
        ((Window) w).setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
WindowManager.LayoutParams.FLAG_FULLSCREEN);
    }
}
```

Файл Test

```
      package school.experiment.anatomy;

import android.content.Intent;
import android.widget.TextView;

import com.elementaryschool.anatomy.R;

public class Test extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.test);

        Button button_back = (Button) findViewById(R.id.button_back);
        button_back.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v2) {
                //Начало конструкции
                try {
                    Intent intent = new Intent( Test.this, Menu.class);
                    startActivity(intent); finish();
                } catch (Exception e) {
                }
                //конец конструкции
            }
        });
        //кнопка перехода на 1й уровень - начало
        TextView textView1 = (TextView)findViewById(R.id.textView1);
```

```
        textView1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                try {
                    Intent intent = new Intent( Test.this, Level1.class);
                    startActivity(intent); finish();
                } catch (Exception e){

                }
            }
        });

        //кнопка перехода на 1й уровень - конец
        Window w = getWindow();
        ((Window) w).setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
WindowManager.LayoutParams.FLAG_FULLSCREEN);
    }
    //Системная кнопка назад
     @Override
    public void onBackPressed(){
//Начало конструкции
         try {
             Intent intent = new Intent( Test.this, Menu.class);
             startActivity(intent); finish();
         } catch (Exception e) {
         }
         //конец конструкции
     }
    //Системная кнопка назад конец
}
```

Файл Level1

```
      package school.experiment.anatomy;
import com.elementaryschool.anatomy.R;

import java.util.Random;

public class Level1 extends AppCompatActivity {

    Dialog dialog;
    Dialog dialogEnd;

    public int imgLeft; //left img + txt
    public int imgRight;
    Array array = new Array(); // new aray for level
    Random random = new Random();
    public  int count =0; //wright answers

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.universal);

        //создаем переменую левелс

        TextView text_levels = findViewById(R.id.text_levels);
        text_levels.setText(R.string.level1);

        final ImageView img_left =(ImageView)findViewById(R.id.image_left);
        //corners left
        img_left.setClipToOutline(true);

        final ImageView img_right =(ImageView)findViewById(R.id.image_right);
        //corners right
        img_right.setClipToOutline(true);
```

```
        //way to left textView
        final TextView text_left = findViewById(R.id.text_left);
        //right
        final TextView text_right = findViewById(R.id.text_right);


        Button button_back = (Button) findViewById(R.id.button_back);
        button_back.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v2) {
                //Начало конструкции
                try {
                    Intent intent = new Intent( Level1.this, Test.class);
                    startActivity(intent); finish();
                } catch (Exception e) {
                }
                //конец конструкции
            }
        });
        //развернуть на весь экран - начало
        Window w = getWindow();
        ((Window) w).setFlags(WindowManager.LayoutParams.FLAG_FULLSCREEN,
WindowManager.LayoutParams.FLAG_FULLSCREEN);
        //развернуть на весь экран - конец
        //Dialog at start
        dialog = new Dialog(this);
        dialog.requestWindowFeature(Window.FEATURE_NO_TITLE);
        dialog.setContentView(R.layout.previedialog);
        dialog.getWindow().setBackgroundDrawable(new
ColorDrawable(Color.TRANSPARENT));
        dialog.setCancelable(false);//buttom back

        //close dialog -start
        TextView btnclose = (TextView)dialog.findViewById(R.id.button_close);
        btnclose.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // btn press
                try {
                    //back to level
                    Intent intent = new Intent( Level1.this, Test.class);
                    startActivity(intent); finish();
                } catch (Exception e) { //no code
                }
                dialog.dismiss();// close dialog
                //btn press end
            }
        });
        //close dialog -end
        //btn continue
        Button btncontinue = (Button)dialog.findViewById(R.id.btncontinue);
        btncontinue.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
               dialog.dismiss();
            }
        });
        //btn continue end
        dialog.show();
        //_____________________________________________________________________
        //Dialog at end game
        dialogEnd = new Dialog(this);
        dialogEnd.requestWindowFeature(Window.FEATURE_NO_TITLE);
        dialogEnd.setContentView(R.layout.enddialog);
        dialogEnd.getWindow().setBackgroundDrawable(new
```

```
ColorDrawable(Color.TRANSPARENT));
        dialogEnd.setCancelable(false);//buttom back

        //close dialog -start
        TextView btnclose2 =
(TextView)dialogEnd.findViewById(R.id.button_close);
        btnclose2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // btn press
                try {
                    //back to level
                    Intent intent = new Intent( Level1.this, Test.class);
                    startActivity(intent); finish();
                } catch (Exception e) { //no code
                }
                dialogEnd.dismiss();// close dialog
                //btn press end
            }
        });
        //close dialog -end
        //btn continue
        Button btncontinue2 = (Button)dialogEnd.findViewById(R.id.btncontinue);
        btncontinue2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                try{
                    Intent intent = new Intent( Level1.this, Level2.class);
                    startActivity(intent); finish();
                }catch (Exception e){
                    //no code
                }
                dialogEnd.dismiss();
            }
        });
        //btn continue end

        //_____________________________________________________________________
        //array progress igi
        final int[] progress ={
                R.id.point1, R.id.point2, R.id.point3, R.id.point4, R.id.point5,
                R.id.point6, R.id.point7, R.id.point8, R.id.point9, R.id.point10
        };
        //array progress = end

        //animation start
        final Animation animation = AnimationUtils.loadAnimation(Level1.this,
R.anim.alpha);
        //animation end

        imgLeft = random.nextInt(15);//randow number 0 -14
        img_left.setImageResource(array.images1[imgLeft]);
        text_left.setText(array.text1[imgLeft]);

        if (imgLeft <= 9) {
            imgRight = random.nextInt(5)+10;
        } else {
            imgRight = random.nextInt(10);
        };
        img_right.setImageResource(array.images1[imgRight]);
        text_right.setText(array.text1[imgRight]);

        //btm left click -start
        img_left.setOnTouchListener(new View.OnTouchListener() {
            @Override
            public boolean onTouch(View v, MotionEvent event) {
```

```
//условие косание картинки - начало
if (event.getAction()==MotionEvent.ACTION_DOWN){
    img_right.setEnabled(false);
    if (imgLeft<imgRight){
        img_left.setImageResource(R.drawable.img_true);
    }else {
        img_left.setImageResource(R.drawable.img_false);
    }
}else if (event.getAction()==MotionEvent.ACTION_UP){
        //start
        if (imgLeft<imgRight){
            //if left pic is wright
            if (count<10){
                count++;
            }
            //grey colorg progress
                for (int i=0; i<10; i++){
                    TextView tv = findViewById(progress[i]);

                }
            //grey colorg progress end
            //green color
            for (int i=0; i<count; i++){
                TextView tv =findViewById(progress[i]);

            }
        }else{
            //if left pick is wrong
            if (count>0){
                if (count==1){
                    count=0;
                } else {
                    count=count-2;
                }
            }
            for (int i=0; i<9; i++){
                TextView tv = findViewById(progress[i]);

            }
            //grey colorg progress end
            //green color
            for (int i=0; i<count; i++){
                TextView tv =findViewById(progress[i]);
                                            }
        }
    //end
    if (count==10){
        //END OF LEVEL
        dialogEnd.show();
    }else {
        imgLeft = random.nextInt(15);//randow number 0 -14
        img_left.setImageResource(array.images1[imgLeft]);
        img_left.startAnimation(animation);
        text_left.setText(array.text1[imgLeft]);

        if (imgLeft <= 9) {
            imgRight = random.nextInt(5)+10;
        } else {
            imgRight = random.nextInt(10);
        };
        img_right.setImageResource(array.images1[imgRight]);
        img_right.startAnimation(animation);
        text_right.setText(array.text1[imgRight]);

        img_right.setEnabled(true);
    }
```

```
                }
                //условие конец
                return true;
            }
        });
        //img left -end

        //btm right click -start
        img_right.setOnTouchListener(new View.OnTouchListener() {
            @Override
            public boolean onTouch(View v, MotionEvent event) {
                //условие косание картинки - начало
                if (event.getAction()==MotionEvent.ACTION_DOWN){
                    img_left.setEnabled(false);
                    if (imgLeft>imgRight){
                        img_right.setImageResource(R.drawable.img_true);
//block left
                    }else {
                        img_right.setImageResource(R.drawable.img_false);
                    }
                }else if (event.getAction()==MotionEvent.ACTION_UP){
                    //start
                    if (imgLeft>imgRight){
                        //if right pic is wright
                        if (count<10){
                            count++;
                        }
                        //grey colorg progress
                        for (int i=0; i<10; i++){
                            TextView tv = findViewById(progress[i]);
                            tv.setBackgroundResource(R.drawable.style_points);
                        }
                        //grey colorg progress end
                        //green color
                        for (int i=0; i<count; i++){
                            TextView tv =findViewById(progress[i]);

tv.setBackgroundResource(R.drawable.style_points_green);
                        }
                    }else{
                        //if left pick is wrong
                        if (count>0){
                            if (count==1){
                                count=0;
                            } else {
                                count=count-2;
                            }
                        }
                        for (int i=0; i<9; i++){
                            TextView tv = findViewById(progress[i]);
                            tv.setBackgroundResource(R.drawable.style_points);
                        }
                        //grey colorg progress end
                        //green color
                        for (int i=0; i<count; i++){
                            TextView tv =findViewById(progress[i]);

tv.setBackgroundResource(R.drawable.style_points_green);
                        }
                    }
                    //end
                    if (count==10){
                        //END OF LEVEL
                        dialogEnd.show();
                    }else {
                        imgLeft = random.nextInt(15);//randow number 0 -14
```

```
                        img_left.setImageResource(array.images1[imgLeft]);
                        img_left.startAnimation(animation);
                        text_left.setText(array.text1[imgLeft]);

                        if (imgLeft <= 9) {
                            imgRight = random.nextInt(5)+10;
                        } else {
                            imgRight = random.nextInt(10);
                        };
                        img_right.setImageResource(array.images1[imgRight]);
                        img_right.startAnimation(animation);
                        text_right.setText(array.text1[imgRight]);

                        img_left.setEnabled(true); //left img is worked now
                    }
                }
                //условие конец
                return true;
            }
        });
        //img right -end
    }
    //system btn back

    @Override
    public void onBackPressed(){
//Начало конструкции
        try {
            Intent intent = new Intent( Level1.this, Test.class);
            startActivity(intent); finish();
        } catch (Exception e) {
        }
        //конец конструкции

    }
}
```

Приложение Б

Описание атрибутов таблиц физической модели БД

(рекомендованное)

Таблица Б.1 – Сущность – jornal

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возможность значения Null	Роль	Пример
uuid_jornal	ключ	Целый	4	0001- 5000	Нет	PK	1736
ver	Версия	Целый	30	‘А-я’,‘0-9’	Нет		1736
updt	Кто внес иззменения	Целый	5	00001-30 000	Нет		1 500
updt	Дата мод	Дата	2	‘А-я’	Нет		22.01.20
uuid_model	ключ	Целый	4	0001- 5000	Нет	FK	20000
uuid_action	ключа	Целый	4	0001-5000	Нет	FK	1999

Таблица Б.2 – Сущность – action

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возможность значения Null	Роль	Пример
uuid_**action**	ключ	Целый	4	0001- 5000	Нет	РК	1736
ver	Версия	Целый	30	‘А-я’,‘0-9’	Нет		1736
updt	Кто внес иззменения	Целый	5	00001-30 000	Нет		1 500
updt	Дата мод	Дата	2	‘А-я’	Нет		22.01.20
name	наименование	Символьный	8	‘А-я’	Нет		выбрал
description	описание	Символьный	10	‘А-я’	Нет		Активировал переход

Таблица Б.3 – Сущность – model

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возможность значения Null	Роль	Пример
uuid_model	ключ	Целый	4	0001- 5000	Нет	PK	1736
ver	Версия	Целый	30	‘А-я’,‘0-9’	Нет		1736
updt	Кто внес иззменения	Целый	5	00001-30 000	Нет		1 500
updt	Дата мод	Дата	2	‘А-я’	Нет		22.01.20
russian	Поле языка	Символьный	30	‘А-я’	Нет		
english	Поле языка	Символьный	30	‘А-я’	Нет		
portuguese	Поле языка	Символьный	30	‘А-я’	Нет		
indian	Поле языка	Символьный	30	‘А-я’	Нет		
arabian	Поле языка	Символьный	30	‘А-я’	Нет		
spanish	Поле языка	Символьный	30	‘А-я’	Нет		
chinese	Поле языка	Символьный	30	‘А-я’	Нет		
parent	Поле родителя	Целый			нет	FK	1123

Таблица Б.4 – Сущность –lang

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возможность значения Null	Роль	Пример
uuid_model	ключ	Целый	4	0001- 5000	Нет	PK	1736
ver	Версия	Целый	30	‘А-я’,‘0-9’	Нет		1736
updt	Кто внес иззменения	Целый	5	00001-30 000	Нет		1 500
updt	Дата мод	Дата	2	‘А-я’	Нет		22.01.20
Lang_name	наименование	Символьный	8	‘А-я’	Нет		russian

Таблица Б.5 – Сущность – task

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возможность значения Null	Роль	Пример
uuid_task	ключ	Целый	4	0001- 5000	Нет	PK	1736
ver	Версия	Целый	30	‘А-я’,‘0-9’	Нет		1736
updt	Кто внес иззменения	Целый	5	00001-30 000	Нет		1 500
updt	Дата мод	Дата	2	‘А-я’	Нет		22.01.20
uuid_lang	ключ	Целый	4	0001- 5000	Нет	FK	1726
uuid_model	ключ	Целый	4	0001- 5000	Нет	FK	1436
russian	Поле языка	Символьный	30	‘А-я’	Нет		
english	Поле языка	Символьный	30	‘А-я’	Нет		
portuguese	Поле языка	Символьный	30	‘А-я’	Нет		
indian	Поле языка	Символьный	30	‘А-я’	Нет		
arabian	Поле языка	Символьный	30	‘А-я’	Нет		
spanish	Поле языка	Символьный	30	‘А-я’	Нет		
chinese	Поле языка	Символьный	30	‘А-я’	Нет		
uuid_teacher	ключ	Целый	4	0001- 5000	Нет	PK	1726

Таблица Б.6 – Сущность – quastion

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возможность значения Null	Роль	Пример
uuid_quastion	ключ	Целый	4	0001- 5000	Нет	РК	1736
ver	Версия	Целый	30	‘А-я’,‘0-9’	Нет		1736
updt	Кто внес иззменения	Целый	5	00001-30 000	Нет		1 500
updt	Дата мод	Дата	2	‘А-я’	Нет		22.01.20
uuid_lang	ключ	Целый	4	0001- 5000	Нет	РК	1726
uuid_model	ключ	Целый	4	0001- 5000	Нет	РК	1436
russian	Поле языка	Символьный	30	‘А-я’	Нет		
english	Поле языка	Символьный	30	‘А-я’	Нет		
portuguese	Поле языка	Символьный	30	‘А-я’	Нет		
indian	Поле языка	Символьный	30	‘А-я’	Нет		
arabian	Поле языка	Символьный	30	‘А-я’	Нет		
spanish	Поле языка	Символьный	30	‘А-я’	Нет		
chinese	Поле языка	Символьный	30	‘А-я’	Нет		
uuid_teacher	ключ	Целый	4	0001- 5000	Нет	РК	1726

Таблица Б.7 – Сущность – answer

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возможность значения Null	Роль	Пример
uuid_answer	ключ	Целый	4	0001- 5000	Нет	PK	1736
ver	Версия	Целый	30	‘А-я’,‘0-9’	Нет		1736
updt	Кто внес иззменения	Целый	5	00001-30 000	Нет		1 500
updt	Дата мод	Дата	2	‘А-я’	Нет		22.01.20
uuid_lang	ключ	Целый	4	0001- 5000	Нет	FK	1726
uuid_model	ключ	Целый	4	0001- 5000	Нет	FK	1436
russian	Поле языка	Символьный	30	‘А-я’	Нет		
english	Поле языка	Символьный	30	‘А-я’	Нет		
portuguese	Поле языка	Символьный	30	‘А-я’	Нет		
indian	Поле языка	Символьный	30	‘А-я’	Нет		
arabian	Поле языка	Символьный	30	‘А-я’	Нет		
spanish	Поле языка	Символьный	30	‘А-я’	Нет		
chinese	Поле языка	Символьный	30	‘А-я’	Нет		
right_answer	ключ	bool	4	0001- 5000	Нет	PK	1/0

Таблица Б.8 – Сущность – test_list

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возможность значения Null	Роль	Пример
uuid_test_list	ключ	Целый	4	0001- 5000	Нет	PK	1736
ver	Версия	Целый	30	‘А-я’,‘0-9’	Нет		1736
updt	Кто внес иззменения	Целый	5	00001-30 000	Нет		1 500
updt	Дата мод	Дата	2	‘А-я’	Нет		22.01.20
uuid_lang	ключ	Целый	4	0001- 5000	Нет	FK	1726
uuid_model	ключ	Целый	4	0001- 5000	Нет	FK	1436
uuid_answer	ключ	Целый	4	0001- 5000	Нет	FK	1736
uuid_quastion	ключ	Целый	4	0001- 5000	Нет	FK	1736

Таблица Б.9 – Сущность – student_task

Наименование	Описание	Тип данных	Размерность	Область допустимых значений	Возможность значения Null	Роль	Пример
uuid_test_list	ключ	Целый	4	0001- 5000	Нет	PK	1736
ver	Версия	Целый	30	‘А-я’,‘0-9’	Нет		1736
updt	Кто внес иззменения	Целый	5	00001-30 000	Нет		1 500
updt	Дата мод	Дата	2	‘А-я’	Нет		22.01.20
uuid_lang	ключ	Целый	4	0001- 5000	Нет	FK	1726
uuid_model	ключ	Целый	4	0001- 5000	Нет	FK	1436
uuid_task	ключ	Целый	4	0001- 5000	Нет	FK	1736
Is_finished	boolian	bool	4	0001- 5000	Нет	PK	1/0

Приложение В

Реализация RESTful сервисов на языке Java

(рекомендуемое)

Файл AbstractEntity.java

```
package school.exeperiment.anatomyback.model;
import school.exeperiment.anatomyback.utils.UUIDConverter_v2;
@MappedSuperclass
@Converter(name = "uuidConverter", converterClass = UUIDConverter.class)
public abstract class AbstractEntity {
    private static long serialVersionUID = 1L;
    @Id
    public AbstractEntity() {
        this.id = UUID.randomUUID();
    }
    public UUID getId() {
        return id;
    }
    @XmlTransient
    @JsonbTransient
    @JsonIgnore
    public String getSId() {
        return id.toString();
    }
    public void setId(UUID id) {
        this.id = id;
    }
    public UUID getUpdu() {
        return updu;
    }
    public void setUpdu(UUID updu) {
        this.updu = updu;
    }
    public Date getUpdt() {
        return updt;
    }
oid setUpdt(Date updt) {
        this.updt = updt;
    }
```

```
    public Integer getVer() {
        return ver;
    }
    public void setVer(Integer ver) {
        this.ver = ver;
    }
    public Integer getDelMark() {
        return delMark;
    }
    public void setDelMark(Integer delMark) {
        this.delMark = delMark;
    }
    public void incVer() {
        try {
            this.ver = ++this.ver;
        } catch (NullPointerException ex) {
            this.ver = 1;
        }
    }
    @Override
    public int hashCode() {
        int hash = 0;
        hash += (id != null ? id.hashCode() : 0);
        return hash;
    }
    @Override
    public boolean equals(Object object) {
        // TODO: Warning - this method won't work in the case the id fields
are not set
        if (!(object instanceof AbstractEntity)) {
            return false;
        }
        AbstractEntity other = (AbstractEntity) object;
        return !((this.id == null && other.id != null) || (this.id != null
&& !this.id.equals(other.id)));
    }
```

```
    @Override
    public String toString() {
        return "school.experiment.scenario.model." +
this.getClass().getSimpleName() + "[ id=" + id + " ]";
    }
}
```

Файл Action.java

```
package school.exeperiment.anatomyback.model;
import java.io.Serializable;
import javax.xml.bind.annotation.XmlTransient;
/**
 *
 * @author revan
 */
@Entity
@Table(name = "action")
@XmlRootElement
@NamedQueries({
    @NamedQuery(name = "Action.findAll", query = "SELECT a FROM Action a"),
   a.updt = :updt")})
public class Action extends AbstractEntity implements Serializable {
    private static final long serialVersionUID = 1L;
    @Id
    @Basic(optional = false)
    @NotNull
    @Lob
    @Column(name = "uuid")
    private Object uuid;
    @Size(max = 2147483647)
    @Column(name = "name")
    private String name;
    @Size(max = 2147483647)

    public Action() {
    }
    public Action(UUID uuid) {
```

```
        this.uuid = uuid;
    }
    public Object getUuid() {
        return uuid;
    }
    public void setUuid(UUID uuid) {
        this.uuid = uuid;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
    public String getDescription() {
        return description;
    }
    public void setDescription(String description) {
        this.description = description;
    }
    public String getValue() {
        return value;
    }
    public void setValue(String value) {
        this.value = value;
    }
    @XmlTransient
    public List<Journal> getJournalList() {
        return journalList;
    }
    public void setJournalList(List<Journal> journalList) {
        this.journalList = journalList;
    }
    @Override
    public int hashCode() {
```

```
            int hash = 0;
            hash += (uuid != null ? uuid.hashCode() : 0);
            return hash;
        }
        @Override
        public boolean equals(Object object) {
            // TODO: Warning - this method won't work in the case the id fields
are not set
            if (!(object instanceof Action)) {
                return false;
            }
            Action other = (Action) object;
            if ((this.uuid == null && other.uuid != null) || (this.uuid != null
&& !this.uuid.equals(other.uuid))) {
                return false;
            }
            return true;
        }
        @Override
        public String toString() {
            return "school.exeperiment.anatomyback.model.Action[ uuid=" + uuid
+ " ]";
        }
    }
```

Файл Answer.java

```
    package school.exeperiment.anatomyback.model;
    import java.io.Serializable;
    import javax.xml.bind.annotation.XmlTransient;
    /**
     *
     * @author revan
     */
    @Entity
    @Table(name = "answer")
    @XmlRootElement
    @NamedQueries({
        @NamedQuery(name = "Answer.findAll", query = "SELECT a FROM Answer a"),
```

```
          @NamedQuery(name = "Answer.findByVer", query = "SELECT a FROM Answer
a WHERE WHERE a.arabian = :arabian"),

          @NamedQuery(name = "Answer.findByChinese", query = "SELECT a FROM Answer
a WHERE a.chinese = :chinese"),

          @NamedQuery(name = "Answer.findByIndian", query = "SELECT a FROM Answer
a WHERE a.indian = :indian"),

          @NamedQuery(name = "Answer.findBySpanish", query = "SELECT a FROM Answer
a WHERE a.spanish = :spanish"),

          @NamedQuery(name = "Answer.findByPortuguese", query = "SELECT a FROM
Answer a WHERE a.portuguese = :portuguese"),

          @NamedQuery(name = "Answer.findByRight", query = "SELECT a FROM Answer
a WHERE a.right = :right")})

      public class Answer extends AbstractEntity implements Serializable {


          private static final long serialVersionUID = 1L;

          @Id

          @Basic(optional = false)


          @Size(max = 2147483647)

          @Column(name = "chinese")

          private String chinese;

          @Size(max = 2147483647)

          public Answer() {

          }


          public Answer(UUID uuid) {

              this.uuid = uuid;

          }

          public Answer(UUID uuid, boolean right) {

              this.uuid = uuid;

              this.right = right;

          }

          public Object getUuid() {

              return uuid;

          }

          public void setUuid(UUID uuid) {

              this.uuid = uuid;

          }
```

```
public String getDescription() {
      return description;
  }
  public void setDescription(String description) {
      this.description = description;
  }
  public String getRussian() {
      return russian;
  }
  public void setRussian(String russian) {
      this.russian = russian;
  }
  public String getArabian() {
      return arabian;
  }
  public void setArabian(String arabian) {
      this.arabian = arabian;
  }
  public String getChinese() {
      return chinese;
  }
  public void setChinese(String chinese) {
      this.chinese = chinese;
  }
  public String getIndian() {
      return indian;
  }
  public void setIndian(String indian) {
      this.indian = indian;
  }

  public String getSpanish() {
      return spanish;
  }
  public void setSpanish(String spanish) {
```

```
        this.spanish = spanish;
    }
    public String getPortuguese() {
        return portuguese;
    }
    public void setPortuguese(String portuguese) {
        this.portuguese = portuguese;
    }
    public boolean getRight() {
        return right;
    }
    public void setRight(boolean right) {
        this.right = right;
    }
    public Quastion getQuasrionUuid() {
        return quasrionUuid;
    }
    public void setQuasrionUuid(Quastion quasrionUuid) {
        this.quasrionUuid = quasrionUuid;
    }
    @XmlTransient
    public List<TestList> getTestListList() {
        return testListList;
    }
    public void setTestListList(List<TestList> testListList) {
        this.testListList = testListList;
    }

    @Override
    public int hashCode() {
        int hash = 0;
        hash += (uuid != null ? uuid.hashCode() : 0);
        return hash;
    }
    @Oerride
```

```
    public boolean equals(Object object) {
        // TODO: Warning - this method won't work in the case the id fields are not set
        if (!(object instanceof Answer)) {
            return false;
        }
        Answer other = (Answer) object;
        if ((this.uuid == null && other.uuid != null) || (this.uuid != null && !this.uuid.equals(other.uuid))) {
            return false;
        }
        return true;
    }

    @Override
    public String toString() {
        return "school.exeperiment.anatomyback.model.Answer[ uuid=" + uuid + " ]";
    }
}
```

Файл Journal.java

```
package school.exeperiment.anatomyback.model;

import java.io.Serializable;

import javax.xml.bind.annotation.XmlRootElement;

/**
 *
 * @author revan
 */
@Entity
@Table(name = "journal")
@XmlRootElement
@NamedQueries({
    @NamedQuery(name = "Journal.findAll", query = "SELECT j FROM Journal j"),
    @NamedQuery(name = "Journal.findByVer", query = "SELECT j FROM Journal j WHERE j.ver = :ver"),
    @NamedQuery(name = "Journal.findByUpdt", query = "SELECT j FROM Journal j WHERE j.updt = :updt")})
```

```
public class Journal extends AbstractEntity implements Serializable {
    private static final long serialVersionUID = 1L;
    @Column(name = "ver")
    private Integer ver;
    @Column(name = "updt")
    @Temporal(TemporalType.TIMESTAMP)
    private Date updt;
    @Lob
    @Column(name = "updu")
    private Object updu;
    @Id
    @Basic(optional = false)
    @NotNull
    @Lob
    @Column(name = "uuid")
    private Object uuid;
    @JoinColumn(name = "action_uuid", referencedColumnName = "uuid")
    @ManyToOne(optional = false)
    private Action actionUuid;
    @JoinColumn(name = "model_uuid", referencedColumnName = "uuid")
    @ManyToOne(optional = false)
    private Model modelUuid;
    public Journal() {
    }
    public Journal(UUID uuid) {
        this.uuid = uuid;
    }
    public void setUpdu(UUID updu) {
        this.updu = updu;
    }
    public Object getUuid() {
        return uuid;
    }
    public void setUuid(UUID uuid) {
        this.uuid = uuid;
```

```
    }
    public Action getActionUuid() {
        return actionUuid;
    }
    public void setActionUuid(Action actionUuid) {
        this.actionUuid = actionUuid;
    }
    public Model getModelUuid() {
        return modelUuid;
    }
    public void setModelUuid(Model modelUuid) {
        this.modelUuid = modelUuid;
    }
    @Override
    public int hashCode() {
        int hash = 0;
        hash += (uuid != null ? uuid.hashCode() : 0);
        return hash;
    }
    @Override
    public boolean equals(Object object) {
        // TODO: Warning - this method won't work in the case the id fields
are not set
        if (!(object instanceof Journal)) {
            return false;
        }
        Journal other = (Journal) object;
        if ((this.uuid == null && other.uuid != null) || (this.uuid != null
&& !this.uuid.equals(other.uuid))) {
            return false;
        }
        return true;
    }
    @Override
    public String toString() {
        return "school.exeperiment.anatomyback.model.Journal[ uuid=" +
uuid + " ]";
```

```
        }
    }
```

Файл Quastion.java

```
    package school.exeperiment.anatomyback.model;


    import java.io.Serializable;
    import javax.xml.bind.annotation.XmlTransient;
    /**
     *
     * @author revan
     */
    @Entity
    @Table(name = "quastion")
    @XmlRootElement
    @NamedQueries({
        @NamedQuery(name = "Quastion.findAll", query = "SELECT q FROM Quastion
q"),
        @NamedQuery(name = "Quastion.findByVer", query = "SELECT q FROM Quastion
q WHERE WHERE q.portuguese = :portuguese")})
    public class Quastion extends AbstractEntity implements Serializable {
        private static final long serialVersionUID = 1L;
        @Column(name = "ver")
        private Integer ver;
        @Id
        @Basic(optional = false)
        @NotNull
        @Lob
        @Column(name = "uuid")
        private Object uuid;
        @Column(name = "updt")
        @Temporal(TemporalType.TIMESTAMP)
        private Date updt;
        @Size(max = 2147483647)
        @Column(name = "description")
      )
        private List<TestList> testListList;
```

```
public Quastion() {
}
public Quastion(UUID uuid) {
    this.uuid = uuid;
}

public Integer getVer() {
    return ver;
}
public void setVer(Integer ver) {
    this.ver = ver;
}
public Object getUuid() {
    return uuid;
}
public void setUuid(UUID uuid) {
    this.uuid = uuid;
}
public Date getUpdt() {
    return updt;
}
public void setUpdt(Date updt) {
    this.updt = updt;
}
public String getDescription() {
    return description;
}
public void setDescription(String description) {
    this.description = description;
}
public String getRussian() {
    return russian;
}
public void setRussian(String russian) {
    this.russian = russian;
```

```
}
public String getArabian() {
    return arabian;
}

public void setArabian(String arabian) {
    this.arabian = arabian;
}
public String getChinese() {
    return chinese;
}
public void setChinese(String chinese) {
    this.chinese = chinese;
}
public String getIndian() {
    return indian;
}
public void setIndian(String indian) {
    this.indian = indian;
}
public String getSpanish() {
    return spanish;
}
public void setSpanish(String spanish) {
    this.spanish = spanish;
}
public String getPortuguese() {
    return portuguese;
}
public void setPortuguese(String portuguese) {
    this.portuguese = portuguese;
}
public void setUpdu(Object updu) {
    this.updu = updu;
}
```

```
    @XmlTransient
    public List<Answer> getAnswerList() {
        return answerList;
    }
    public void setAnswerList(List<Answer> answerList) {
        this.answerList = answerList;
    }
    public Task getTaskUuid() {
        return taskUuid;
    }
    public void setTaskUuid(Task taskUuid) {
        this.taskUuid = taskUuid;
    }
    @XmlTransient
    public List<TestList> getTestListList() {
        return testListList;
    }
    public void setTestListList(List<TestList> testListList) {
        this.testListList = testListList;
    }
    @Override
    public int hashCode() {
        int hash = 0;
        hash += (uuid != null ? uuid.hashCode() : 0);
        return hash;
    }
    @Override
    public boolean equals(Object object) {
        // TODO: Warning - this method won't work in the case the id fields
are not set
        if (!(object instanceof Quastion)) {
            return false;
        }
        Quastion other = (Quastion) object;
```

```
            if ((this.uuid == null && other.uuid != null) || (this.uuid != null
&& !this.uuid.equals(other.uuid))) {

                return false;

            }

            return true;

        }

        @Override

        public String toString() {

            return "school.exeperiment.anatomyback.model.Quastion[ uuid=" +
uuid + " ]";

        }

    }
```

Файл Result.java

```
    package school.exeperiment.anatomyback.model;

    import javax.validation.constraints.Size;

    import javax.xml.bind.annotation.XmlRootElement;

    import javax.xml.bind.annotation.XmlTransient;

    /**

     *

     * @author revan

     */

    @Entity

    @Table(name = "result")

    @XmlRootElement

    @NamedQueries({

        @NamedQuery(name = "Result.findAll", query = "SELECT r FROM Result r"),

        @NamedQuery(name = "Result.findByValue", query = "SELECT r FROM Result
r WHERE r.value = :value"),

        @NamedQuery(name = "Result.findByVer", query = "SELECT r FROM Result
r WHERE r.ver = :ver"),

        @NamedQuery(name = "Result.findByUpdt", query = "SELECT r FROM Result
r WHERE r.updt = :updt"),

        @NamedQuery(name = "Result.findByPass", query = "SELECT r FROM Result
r WHERE r.pass = :pass"),

        @NamedQuery(name = "Result.findByRightAnswers", query = "SELECT r FROM
Result r WHERE r.rightAnswers = :rightAnswers")})

    public class Result extends AbstractEntity implements Serializable {

        private static final long serialVersionUID = 1L;

        @Id
```

```
@Basic(optional = false)
@Column(name = "right_answers")
private Integer rightAnswers;
@JoinColumn(name = "student_task_uuid", referencedColumnName = "uuid")
@ManyToOne(optional = false)
private StudentTask studentTaskUuid;
@OneToMany(mappedBy = "resultUuid")
private List<TestList> testListList;
public Result() {
}
public Result(UUID uuid) {
    this.uuid = uuid;
}
public Result(UUID uuid, UUID updu) {
    this.uuid = uuid;
    this.updu = updu;
}
public UUID getUuid() {
    return (UUID) uuid;
}
public void setUuid(UUID uuid) {
    this.uuid = uuid;
}
public String getValue() {
    return value;
}
public void setValue(String value) {
    this.value = value;
}
public void setUpdu(UUID updu) {
    this.updu = updu;
}
public Boolean getPass() {
    return pass;
}
```

```
    public void setPass(Boolean pass) {
        this.pass = pass;
    }
    public Integer getRightAnswers() {
        return rightAnswers;
    }
    public void setRightAnswers(Integer rightAnswers) {
        this.rightAnswers = rightAnswers;
    }
    public StudentTask getStudentTaskUuid() {
        return studentTaskUuid;
    }
    public void setStudentTaskUuid(StudentTask studentTaskUuid) {
        this.studentTaskUuid = studentTaskUuid;
    }
    @XmlTransient
    public List<TestList> getTestListList() {
        return testListList;
    }
    public void setTestListList(List<TestList> testListList) {
        this.testListList = testListList;
    }
    @Override
    public int hashCode() {
        int hash = 0;
        hash += (uuid != null ? uuid.hashCode() : 0);
        return hash;
    }
    @Override
    public boolean equals(Object object) {
        // TODO: Warning - this method won't work in the case the id fields
are not set
        if (!(object instanceof Result)) {
            return false;
        }
        Result other = (Result) object;
```

```
            if ((this.uuid == null && other.uuid != null) || (this.uuid != null
&& !this.uuid.equals(other.uuid))) {

                return false;

            }

            return true;

        }

        @Override

        public String toString() {

            return "school.exeperiment.anatomyback.model.Result[ uuid=" + uuid
+ " ]";

        }

    }
```

Файл State.java

```
    /*

     * To change this license header, choose License Headers in Project
Properties.

     * To change this template file, choose Tools | Templates

     * and open the template in the editor.

     */

    package school.exeperiment.anatomyback.model;

    import java.io.Serializable;

    import javax.xml.bind.annotation.XmlTransient;

    @Entity

    @Table(name = "state")

    @XmlRootElement

    @NamedQueries({

        @NamedQuery(name = "State.findAll", query = "SELECT s FROM State s"),

        @NamedQuery(name = "State.findByVer", query = "SELECT s FROM State s
WHERE s.ver = :ver"),

        @NamedQuery(name = "State.findByUpdt", query = "SELECT s FROM State s
WHERE s.updt = :updt"),

        @NamedQuery(name = "State.findByState", query = "SELECT s FROM State
s WHERE s.state = :state")})

    public class State extends AbstractEntity implements Serializable {

        private static final long serialVersionUID = 1L;

        @Id

        @Basic(optional = false)
```

```
            private List<StudentTask> studentTaskList;
            public State() {
            }
            public Object getUuid() {
                return uuid;
            }
            public void setUuid(UUID uuid) {
                this.uuid = uuid;
            }
            public void setUpdu(UUID updu) {
                this.updu = updu;
            }
            public void setState(String state) {
                this.state = state;
            }
            @XmlTransient
            public List<Task> getTaskList() {
                return taskList;
            }
            @Override
            public int hashCode() {
                int hash = 0;
                hash += (uuid != null ? uuid.hashCode() : 0);
                return hash;
            }
            @Override
            public boolean equals(Object object) {
                // TODO: Warning - this method won't work in the case the id fields
are not set
                if (!(object instanceof State)) {
                    return false;
                }
                State other = (State) object;
                if ((this.uuid == null && other.uuid != null) || (this.uuid != null
&& !this.uuid.equals(other.uuid))) {
                    return false;
```

```
            }

            return true;

        }

        @Override

        public String toString() {

            return "school.exeperiment.anatomyback.model.State[ uuid=" + uuid
+ " ]";

        }

}
```

Файл StudentTask.java

```
    /*

     * To change this license header, choose License Headers in Project
Properties.

     * To change this template file, choose Tools | Templates

     * and open the template in the editor.

     */

    package school.exeperiment.anatomyback.model;

    import java.io.Serializable;

    import javax.xml.bind.annotation.XmlTransient;

    @Entity

    @Table(name = "student_task")

    @XmlRootElement

    @NamedQueries({

        @NamedQuery(name = "StudentTask.findAll", query = "SELECT s FROM
StudentTask s"),

        @NamedQuery(name = "StudentTask.findByUpdt", query = "SELECT s FROM
StudentTask s WHERE s.updt = :updt"),

        @NamedQuery(name = "StudentTask.findByVer", query = "SELECT s FROM
StudentTask s WHERE s.ver = :ver"),

        @NamedQuery(name = "StudentTask.findByIsFinished", query = "SELECT s
FROM StudentTask s WHERE s.isFinished = :isFinished")})

    public class StudentTask extends AbstractEntity implements Serializable {

        private static final long serialVersionUID = 1L;

        @Id

        @Basic(optional = false)



        private Object studenUuid;

        @OneToMany(cascade = CascadeType.ALL, mappedBy = "studentTaskUuid")
```

```
private List<Result> resultList;
@JoinColumn(name = "state_uuid", referencedColumnName = "uuid")
@ManyToOne
private State stateUuid;
@JoinColumn(name = "task_uuid", referencedColumnName = "uuid")
@ManyToOne(optional = false)
private Task taskUuid;
public StudentTask() {
}
public StudentTask(UUID uuid) {
    this.uuid = uuid;
}
public Object getUuid() {
    return uuid;
}
public void setUuid(UUID uuid) {
    this.uuid = uuid;
}
public Boolean getIsFinished() {
    return isFinished;
}
public void setIsFinished(Boolean isFinished) {
    this.isFinished = isFinished;
}
public Object getStudenUuid() {
    return studenUuid;
}
public void setStudenUuid(UUID studenUuid) {
    this.studenUuid = studenUuid;
}
public void setResultList(List<Result> resultList) {
    this.resultList = resultList;
}
public State getStateUuid() {
    return stateUuid;
```

```
        }

        public void setStateUuid(State stateUuid) {

            this.stateUuid = stateUuid;

        }

        @Override

        public boolean equals(Object object) {

            // TODO: Warning - this method won't work in the case the id fields
are not set

            if (!(object instanceof StudentTask)) {

                return false;

            }

            StudentTask other = (StudentTask) object;

            if ((this.uuid == null && other.uuid != null) || (this.uuid != null
&& !this.uuid.equals(other.uuid))) {

                return false;

            }

            return true;

        }

        @Override

        public String toString() {

            return "school.exeperiment.anatomyback.model.StudentTask[ uuid="
+ uuid + " ]";

        }

    }
```

Файл Tasл.java

```
    /*

     * To change this license header, choose License Headers in Project
Properties.

     * To change this template file, choose Tools | Templates

     * and open the template in the editor.

     */

    package school.exeperiment.anatomyback.model;

    import javax.xml.bind.annotation.XmlTransient;

    @Entity

    @Table(name = "task")

    @XmlRootElement

    @NamedQueries({
```

```
@NamedQuery(name = "Task.findAll", query = "SELECT t FROM Task t"),
 private Date updt;
@Column(name = "ver")
private Integer ver;
@Basic(optional = false)

private Model modelUuid;
@JoinColumn(name = "state_uuid", referencedColumnName = "uuid")
@ManyToOne
private State stateUuid;
@OneToMany(cascade = CascadeType.ALL, mappedBy = "taskUuid")
private List<Quastion> quastionList;
@OneToMany(cascade = CascadeType.ALL, mappedBy = "taskUuid")
private List<StudentTask> studentTaskList;
public Task() {
}
public Task(UUID uuid) {
    this.uuid = uuid;
public Object getUuid() {
    return uuid;
}
public void setUuid(UUID uuid) {
    this.uuid = uuid;
}
public String getDescription() {
    return description;
}
public void setDescription(String description) {
    this.description = description;
public void setTeacherUuid(UUID teacherUuid) {
    this.teacherUuid = teacherUuid;
}
public Boolean getUpload() {
    return upload;
public String getRussian() {
```

```
        return russian;
    }
    public void setRussian(String russian) {
        this.russian = russian;
    }
    public String getArabian() {
        return arabian;
    }
    public void setArabian(String arabian) {
        this.arabian = arabian;
    }
    public String getChinese() {
        return chinese;
    }
    public void setChinese(String chinese) {
        this.chinese = chinese;
    public void setSpanish(String spanish) {
        this.spanish = spanish;
    }
    public String getPortuguese() {
        return portuguese;
    }
    public void setModelUuid(Model modelUuid) {
        this.modelUuid = modelUuid;
    }
    public State getStateUuid() {
        return stateUuid;
    }
    public void setStateUuid(State stateUuid) {
        this.stateUuid = stateUuid;
    }
    public void setStudentTaskList(List<StudentTask> studentTaskList) {
        this.studentTaskList = studentTaskList;
    }
    @Override
```

```
    public int hashCode() {
        int hash = 0;
        hash += (uuid != null ? uuid.hashCode() : 0);
        return hash;
    }
    @Override
    public boolean equals(Object object) {
        // TODO: Warning - this method won't work in the case the id fields are not set
        if (!(object instanceof Task)) {
            return false;
        }
        Task other = (Task) object;
        if ((this.uuid == null && other.uuid != null) || (this.uuid != null && !this.uuid.equals(other.uuid))) {
            return false;
        }
        return true;
    }
    @Override
    public String toString() {
        return "school.exeperiment.anatomyback.model.Task[ uuid=" + uuid + " ]";
    }
    }
```

Файл TestList.java

```
/*
 * To change this license header, choose License Headers in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package school.exeperiment.anatomyback.model;
import java.io.Serializable;
import javax.xml.bind.annotation.XmlRootElement;
@Entity
@Table(name = "test_list")
```

```
@XmlRootElement
@NamedQueries({
    @NamedQuery(name = "TestList.findAll", query = "SELECT t FROM TestList
t"),
    @NamedQuery(name = "TestList.findByVer", query = "SELECT t FROM TestList
t WHERE t.ver = :ver"),
    @NamedQuery(name = "TestList.findByUpdt", query = "SELECT t FROM
TestList t WHERE t.updt = :updt")})
public class TestList extends AbstractEntity implements Serializable {
    private static final long serialVersionUID = 1L;
    @Id
    @Basic(optional = false)
    @NotNull
    @Temporal(TemporalType.TIME)
    @ManyToOne
    private Result resultUuid;
    public TestList() {
    }

    public void setAnswerUuid(Answer answerUuid) {
        this.answerUuid = answerUuid;
    }
    public Quastion getQuastionUuid() {
        return quastionUuid;
    }
    public void setQuastionUuid(Quastion quastionUuid) {
        this.quastionUuid = quastionUuid;
    }
    public Result getResultUuid() {
        return resultUuid;
    }
    public void setResultUuid(Result resultUuid) {
        this.resultUuid = resultUuid;

    @Override
    public boolean equals(Object object) {
```

```
            // TODO: Warning - this method won't work in the case the id fields
are not set

            if (!(object instanceof TestList)) {

                return false;

            }

            TestList other = (TestList) object;

            if ((this.uuid == null && other.uuid != null) || (this.uuid != null
&& !this.uuid.equals(other.uuid))) {

                return false;

            }

            return true;

        }

        @Override

        public String toString() {

            return "school.exeperiment.anatomyback.model.TestList[ uuid=" +
uuid + " ]";

        }
```