

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего
образования

«ПЕНЗЕНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Факультет

Кафедра

вычислительной техники

«Вычислительная техника»

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

ДИПЛОМНЫЙ ПРОЕКТ

на тему

Приложение для экспресс-оценки знаний сотрудников
организации

Студент Кокорев Владислав Алексеевич
(подпись, дата) (ФИО полностью)
Руководитель Егоров В.Ю.
(подпись, дата) (фамилия, инициалы)
Нормоконтролёр Бычкова А.С.
(подпись, дата) (фамилия, инициалы)
Рецензент Савастьев И.Р.
(должность, место работы) (подпись, дата) (фамилия, инициалы)

Работа допущена к защите (протокол заседания кафедры от 28.05.2025 №16)

Заведующий кафедрой Митрохин М.А.
(подпись) (фамилия, инициалы)
Работа защищена с отметкой удовлетворительно (протокол заседания ГЭК от 19.06.2025 № 9)

Секретарь ГЭК

Гурин Е.И.
(подпись) (фамилия, инициалы)

Пенза 2025

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

Факультет
Вычислительной техники

Кафедра
Вычислительная техника

«Утверждаю»
Заведующий кафедрой
Митрохин М.А.
6 мая 2025 г.

**ЗАДАНИЕ
НА ДИПЛОМНЫЙ ПРОЕКТ**

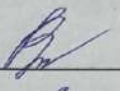
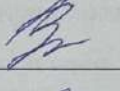
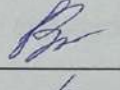
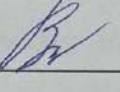
1. Студент Кокорев Владислав Алексеевич гр. 20ВОЭ1 факультета вычислительной техники.
2. Тема работы: «Приложение для экспресс-оценки знаний сотрудников организации». Тема утверждена приказом ПГУ №601/о от 06.05.2025 г.
3. Руководитель работы к.т.н., доцент каф. ВТ «ПГУ» Егоров В.Ю.
4. Задание на дипломную работу (назначение разработки, исходные данные и т.п.).

Разработать программное обеспечение для проведения массового экспресс тестирования сотрудников учреждения. Программное обеспечение должно иметь возможность регистрации и авторизации пользователей, формирования новых и коррекции имеющихся тестов. Пользователи приложения должны иметь возможность ознакомления и прохождения тестирования, с возможностью формирования личного рейтинга Программное обеспечение должна разрабатываться с использованием IDE «AndroidStudio», языка Kotlin, СУБД SQLite.

5. Перечень подлежащих разработке вопросов.

Требуется произвести анализ предметной области, определить функциональный состав программы, выбрать стек технологий для реализации программы, разработать UML диаграммы, спроектировать модель СУБД, разработать структуру программы, с последующим написанием программного кода.

6. Календарный график выполнения работы

№ п/п	Наименование этапов работы	Объем работы	Срок выполнения	Подпись руководителя
1	Анализ предметной области	10	06.05.25 – 08.05.25	
2	Проектирование и отладка программного обеспечения	10	08.05.25 – 09.05.25	
3	Разработка системы и базы данных	20	09.05.25 – 11.05.25	
4	Отладка и тестирование	10	11.05.25 – 17.05.25	
5	Разработка экономической и БЖД частей	10	17.05.25 – 19.05.25	
6	Оформление графической части	10	19.05.25 – 25.05.25	
7	Оформление выпускной работы	30	25.05.25 – 30.05.25	

Дата выдачи задания «06» мая 2025г.

Руководитель дипломной работы _____

 06.05.25 Егоров В.Ю.
(подпись, дата)

Задание к исполнению принял студент _____

 06.05.25 Кокорев В. А.
(подпись, дата)

Работу к защите допустить

Декан ФВТ
(подпись, дата)

 19.06.2025 Л.Р. Фионова

Реферат

Пояснительная записка содержит 100 листа, 10 рисунков, 16 таблиц, 9 источников и 4 приложения.

ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ, ТЕСТИРОВАНИЕ, СОТРУДНИКИ ПРЕДПРИЯТИЯ, ОТДЕЛ КАДРОВ, МОБИЛЬНОЕ ПРИЛОЖЕНИЕ, БАЗА ДАННЫХ

Объектом исследования является программное обеспечение для работников учреждения и сотрудников отдела кадров и профильных структурных подразделений, отвечающих за повышение квалификации персонала.

Цель работы – разработать мобильное программное приложение для тестирования работников учреждения без отрыва от выполнения своих обязанностей. Требуется разработать архитектуру приложения, спроектировать модель базы данных.

Технологии разработки –мобильное приложение реализовано на языке Kotlinc использованием IDEAndroidStudio. Хранение данных организовано с помощью СУБД Sqlite.

В результате ВКР разработано программное средство для помощиотделов кадров и повышения квалификации, хранящей тесты, с помощью которых можно вести экспресс тестирование сотрудников.

ПГУ 1.090501.06.001 ПЗ

Изм.	Лист	№ докум.	Подпис	Дата			
Разраб.		Кокорев В.А.		16.06.25			
Пров.		Егоров В.Ю.		16.06.25			
Н. Контр.		Бычков А.С.		16.06.25			
Утв.		Митрохин М.А.		16.06.25			
					Лит.	Лист	Листов
						2	100
					ФВТ, каф. ВТ		
					гр. 20ВОЭ1		

Приложение для экспресс-оценки знаний сотрудников организации
Пояснительная записка

Оглавление

Перечень сокращений и обозначений.....	5
Введение	6
1 Анализ предметной области	8
1.1 Описание предметной области предприятия	8
1.2 Анализ текущего порядка ведения учета	9
1.3 Цели и задачи разрабатываемого программного средства	10
1.4 Анализ целевой аудитории	11
2 Техническое задание	13
2.1 Назначение разработки	13
2.2 Требования к программе	13
2.2.1 Требования к функциональным характеристикам	13
2.2.2 Требования к надежности	14
2.2.3 Требования к составу и параметрам технических средств	14
2.2.4 Требования к информационной и программной совместимости	15
2.3 Требования к программной документации	15
2.4 Порядок контроля и приемки	16
3 Обоснование выбранных средств разработки	18
3.1 Существующие аналоги	20
4 Проектирование системы.....	24
5 Проектирование системы.....	35
6 Программная реализация мобильного приложения	44
7 Оценка затрат на разработку и внедрение программного средства	49
8 Безопасность жизнедеятельности.....	53
Заключение	57

Список использованных источников	59
Приложение А–UML-диаграммы.....	60
Приложение А.1–Диаграмма последовательностей.....	61
Приложение А.2 –Диаграмма развертывания	62
Приложение А.3 –Диаграмма последовательности	63
Приложение А.4 –Диаграмма последовательности	64
Приложение А.5 –Диаграмма данных.....	65
Приложение Б–Функционального тестирования	66
Приложение В– Листинг программы.....	75
Приложение Г– Презентация.....	92

Перечень сокращений и обозначений

В настоящей работе применяются следующие сокращения и обозначения

БД – база данных

НФ – нормальная форма

ПК – персональный компьютер

ПО – программное обеспечение

ПФР – пенсионный фонд России

СУБД – система управления базами данных

ЭВМ – электронная вычислительная машина

ID (Identifier) – идентификатор сущности

PK (PrimaryKey) – первичный ключ

SQL (StructuredQueryLanguage) – язык структурированных запросов

UML (UnifiedModelingLanguage) – унифицированный язык моделирования

					<i>ПГУ 1.090501.10.001 ПЗ</i>	Лист
Изм.	Лист	№ докум.	Подпись	Дата		5

Введение

В условиях цифровой трансформации процессов управление знаниями сотрудников становится ключевым фактором конкурентоспособности организаций. Традиционные методы оценки компетенций, такие как очные экзамены или ручная проверка заданий, зачастую требуют значительных временных и ресурсных затрат, что затрудняет оперативное принятие кадровых решений. В этой связи возникает необходимость во внедрении инновационных инструментов, способных автоматизировать процессы тестирования, обеспечить объективность оценок и повысить эффективность обучения персонала.

Данная ВКР посвящена разработке приложения для экспресс-оценки знаний сотрудников организации, которое предлагает современное решение для быстрого и стандартизированного тестирования.

Актуальность проекта обусловлена растущим спросом на цифровые платформы, способные интегрироваться в корпоративные системы управления человеческими ресурсами (HR) и оптимизировать процессы адаптации, аттестации и профессионального развития сотрудников.

Основная цель ВКР — создание удобного, безопасного и функционального приложения, которое позволит:

- автоматизировать процесс проведения тестирований;
- обеспечить дифференцированный доступ для пользователей с разными ролями (сотрудник/преподаватель);
- упростить анализ результатов и мониторинг прогресса обучающихся.

Задачи, решаемые в работе, включают:

- реализацию системы регистрации и авторизации пользователей с разделением на роли;
- разработку модуля для создания тестов с поддержкой различных типов вопросов (открытые, закрытые, множественный выбор);

- внедрение механизма приглашений для подключения сотрудников к платформе;
- реализацию системы рейтинговой оценки, отражающей прогресс пользователей.

Значимость проекта заключается в его практической направленности. Приложение ускоряет процесс оценки знаний и способствует повышению мотивации сотрудников за счет прозрачной системы рейтингов.

Для организаций внедрение приложения означает сокращение издержек на проведение аттестаций, минимизацию человеческого фактора в оценках и возможность оперативно корректировать учебные программы на основе аналитики.

Приложение может быть адаптировано для использования в корпоративном секторе, образовательных учреждениях или центрах повышения квалификации, что подчеркивает её универсальность и потенциал для масштабирования. Внедрение подобных решений соответствует трендам цифровизации образования и управления персоналом, делая проект актуальным и востребованным на рынке IT-технологий.

ВКР состоит из следующих частей:

- раздел анализа предметной области;
- раздел выбора стека технологий;
- раздел проектирования архитектуры приложения;
- раздел разработки приложения;
- раздел оценки экономической эффективности приложения;
- раздел БЖД.

1 Анализ предметной области

1.1 Описание предметной области предприятия

Предметная область разрабатываемого приложения охватывает процессы организации по проведению тестирования знаний сотрудников в рамках корпоративного обучения или аттестации.

Современные организации всё чаще сталкиваются с необходимостью оперативно оценивать компетенции персонала, адаптировать учебные программы и мотивировать сотрудников к профессиональному развитию. Однако традиционные методы оценки (устные экзамены, бумажные тесты, ручная проверка) требуют значительных временных затрат, подвержены человеческим ошибкам и не предоставляют аналитику в реальном времени.

Реализуемое в ВКР программное обеспечение призвано автоматизировать ключевые этапы оценки знаний: создание тестов, распределение доступа, прохождение тестирования, сбор и анализ результатов.

Внедрение такой системы позволяет стандартизировать процесс, обеспечить прозрачность оценок и оперативно корректировать учебные программы на основе объективных данных.

В организациях, для которых предназначено приложение, процесс оценки знаний сотрудников включает:

- формирование тестовых заданий преподавателями или HR-специалистами;
- распределение тестов среди целевых групп сотрудников;
- прохождение тестирования с фиксацией времени, результатов и ошибок;
- анализ персональных и агрегированных результатов для выявления слабых зон в подготовке.

Критически важными аспектами предметной области являются:

- безопасность данных (защита персональной информации и результатов тестов);
- гибкость настройки тестов (разные типы вопросов, тайминг, система оценивания);
- удобство интерфейса как для преподавателей (создание тестов), так и для студентов (прохождение);
- интеграция с корпоративными системами (например, HR-платформами для автоматического обновления рейтингов).

1.2 Анализ текущего порядка ведения учета

На большинстве предприятий оценка знаний сотрудников осуществляется с помощью устаревших методов или ограниченных цифровых решений, таких как GoogleForms, Kahoot и Quizizz. Несмотря на их популярность, эти инструменты не полностью удовлетворяют потребностям корпоративной среды. В таблице 1 приведены проблемы перечисленных решений.

Таблица 1 – Популярные решения для экспресс тестирования сотрудников

Аспект	Решение	Проблема
Создание тестов	GoogleForms	Ограниченные типы вопросов, отсутствие шаблонов для корпоративных стандартов.
Прохождение тестов	Kahoot	Игровой формат не подходит для формальных аттестаций, нет режима анонимности.
Управление доступом Проверка корректности	Quizizz	Невозможность ограничить доступ по ролям, отсутствие интеграции с корпоративными аккаунтами.

Аспект	Решение	Проблема
Аналитика	Все три платформы	Поверхностная статистика, невозможность генерации индивидуальных отчетов.

Приложения Kahoot! и Quizizz, несмотря на их интерактивность и геймификацию, ориентированы на образовательный сектор и не учитывают специфику корпоративного тестирования:

- отсутствие ролевой модели преподаватели и студенты используют одинаковые интерфейсы, что повышает риск ошибок (например, случайное удаление теста студентом);
- нет системы рейтингов результаты не агрегируются в единый рейтинг сотрудников, что затрудняет оценку прогресса команды;
- ограниченная безопасность данные хранятся в публичных облаках без шифрования, что не соответствует корпоративным стандартам.

GoogleForms, хотя и позволяет создавать тесты с автоматической проверкой, обладает рядом недостатков:

- не поддерживает таймеры для ограничения времени прохождения;
- нет возможности привязать тест к группам или структурным подразделениям компании;
- результаты приходится экспортировать вручную для анализа, что увеличивает нагрузку на HR-отдел.

1.3 Цели и задачи разрабатываемого программного средства

Цель ВКР — создание мобильного приложения для автоматизации процессов оценки знаний сотрудников, обеспечивающего:

- эффективное взаимодействие между преподавателями и студентами;
- минимизацию ручного труда при проведении тестирований;
- прозрачность и объективность оценок;

— хранение тестов на устройстве пользователя для возможного прохождения тестирования в автономном режиме.

Задачи приложения:

- реализация системы регистрации и авторизации с разделением прав;
- студентпрохождение тестов, доступ к личному рейтингу и истории попыток;
- преподаватель создание тестов, управление доступом студентов, анализ результатов;
- разработка модуля для создания тестов;
- внедрение механизма приглашений студентов;
- автоматическая генерация рейтингов на основе результатов тестов;
- обеспечение безопасности данных.

1.4 Анализ целевой аудитории

Целевая аудитория приложения «Приложение для экспресс-оценки знаний сотрудников организации» включает группы пользователей с разными потребностями:

- преподаватели (HR-специалисты);
- создают тесты, настраивают критерии оценивания, управляют доступом студентов;
- анализируют результаты (средние баллы, частота ошибок, прогресс групп);
- сотрудники (студенты);
- проходят тесты, отслеживают свой рейтинг относительно коллег;
- получают обратную связь по ошибкам;
- администраторы системы;
- настраивают интеграции с корпоративными сервисами;
- обеспечивают техническую поддержку;
- руководство компании;

— использует аналитику для принятия кадровых решений (повышение, обучение).

Из анализа целевой аудитории видно что в процесс тестирования и повышения квалификации сотрудников вовлечены все уровни компании.

					ПГУ 1.090501.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		12

2 Техническое задание

2.1 Назначение разработки

Разработка предназначена для автоматизации процесса быстрой оценки знаний сотрудников с целью повышения эффективности корпоративного обучения, объективности итоговой аттестации, а также для прозрачного формирования рейтинга компетенций персонала.

Цель проекта — создание мобильного приложения на платформе Android с использованием языка Kotlin и базы данных SQLite, позволяющего организовать процесс тестирования с гибкой системой ролей, удобной регистрацией и авторизацией пользователей, а также аналитикой результатов.

2.2 Требования к программе

2.2.1 Требования к функциональным характеристикам

Мобильное приложение должно выполнять следующие функции:

- регистрация и авторизация новых пользователей (с разграничением ролей: студент/преподаватель);
- преподаватель;
- создание и редактирование тестов (с различными типами вопросов);
- приглашение студентов на тестирование;
- управление доступом студентов к тестированию (подтверждение или удаление подачи на тест);
- просмотр статистики и результатов тестирования;
- студент;
- прохождение назначенных тестов;
- просмотр своего личного рейтинга и истории попыток;
- работа с локальной базой данных SQLite для хранения пользователей, тестов, результатов и рейтингов;
- обеспечение валидации вводимых данных на каждом этапе взаимодействия;

- защита личных данных пользователей;
- вывод информации о приложении, его создателях и инструкции по использованию.

Входными данными приложения выступает информация о пользователях, тестах, результатах тестирования.

Выходные данные — индивидуальные и групповые результаты тестов, а также рейтинги и аналитика успеваемости.

2.2.2 Требования к надежности

Приложение должно быть устойчиво к ошибкам и сбоям. В мобильном приложении реализована обработка возможных исключений при работе с пользовательским вводом, базой данных или внутренними сервисами.

Все поля ввода должны проходить проверку на корректность (например, отсутствие пустых полей, невозможность дублирования логинов).

В случае некорректных или нештатных действий пользователя должно выводиться информативное сообщение, предотвращающее сбой работы приложения.

В базе данных SQLite должны быть настроены ограничения уникальности и ссылочной целостности для предотвращения потери, некорректного ввода или дублирования данных.

Вся логика работы приложения строится на событийной архитектуре, что повышает контроль за корректностью операций.

Приложение должно сохранять устойчивость к потере соединения (например, при сбое питания мобильного устройства или аварийном завершении работы).

В приложении должно быть реализовано периодическое локальное резервное копирование базы данных.

2.2.3 Требования к составу и параметрам технических средств

Приложение разрабатывается для мобильных устройств под управлением операционной системы Android версии 8.0 (Oreo) и выше:

					ПГУ 1.090501.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		14

- минимальные требования к устройству;
- процессор с тактовой частотой не менее 1,4 ГГц;
- оперативная память — не менее 2 Гбайт;
- доступное место в хранилище — не менее 200 Мбайт;
- дисплей с минимальным разрешением 720×1280.

Для корректной работы требуется наличие стандартных сервисов Android (Storage, Network).

Для работы с базой данных используется встроенная СУБД SQLite, не требующая отдельной установки.

Разработка и отладка приложения должны проводиться на компьютере с установленной IDE AndroidStudio последней стабильной версии.

2.2.4 Требования к информационной и программной совместимости

Приложение должно корректно функционировать на всех популярных Android-устройствах (смартфоны, планшеты).

Не требует установки сторонних серверных или облачных решений — все данные хранятся локально на устройстве пользователя (с возможностью интеграции с корпоративными системами по API в будущих версиях).

Совместимость гарантируется с версиями Android 8.0 и выше.

Поддерживается импорт и экспорт данных в универсальных форматах.

Приложение реализовано на языке Kotlin с использованием стандартных библиотек и компонентов Android SDK.

2.3 Требования к программной документации

Программная документация к разрабатываемому приложению должна содержать следующие разделы:

- техническое задание;
- диаграмма вариантов использования (usecasediagram);
- диаграмма развертывания приложения (deploymentdiagram);
- описание структуры и проектирования базы данных (ER-диаграмма, схема таблиц);

- описание логики работы основных модулей приложения;
- руководство пользователя с пошаговыми инструкциями по функционалу;
- текст исходного кода приложения и инструкции по его сборке и установке.

2.4 Порядок контроля и приемки

Для подтверждения корректности функционирования приложения и всех его функций разрабатывается набор тестовых сценариев.

Контроль осуществляется в следующем порядке:

- тестирование создания, редактирования и удаления тестов преподавателем;
- проверка механизма приглашения студентов к прохождению тестов и подтверждения доступа;
- тестирование прохождения тестирования студентом, сохранения и отображения результатов;
- проверка алгоритма подсчета и отображения рейтинга студентов;
- анализ корректной работы системы валидации данных при вводе пользователем;
- имитация возможных нештатных ситуаций (отключение питания, завершение приложения во время работы), оценка устойчивости приложения к сбоям и целостности сохранения данных;
- оценка работы механизмов резервного копирования и восстановления данных (при их реализации);
- проверка корректной работы приложения на различных устройствах и версиях Android, совместимость с разными аппаратными конфигурациями;
- проверка пользовательского интерфейса на соответствие требованиям эргономичность, простоты и доступности.

Приложение считается принятым, если оно исправно выполняет все заявленные функции, корректно реагирует на все тестовые сценарии, устойчиво к ошибкам ввода и сбоям, а также полностью соответствует требованиям технической и программной документации.

Приемка включает анализ отчета по тестированию, наличие всех необходимых сопровождающих документов, схем и исходного кода, а также внутренних инструкций для дальнейшей эксплуатации и поддержки мобильного приложения.

					ПГУ 1.090501.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		17

3 Обоснование выбранных средств разработки

Для реализации мобильного приложения, предназначенного для экспресс-оценки знаний сотрудников организации, были выбраны технологии Kotlin, AndroidStudio и SQLite. Такой стек подходит для создания современного, производительного и безопасного мобильного решения, соответствующего заявленным целям и функциональным требованиям проекта.

В качестве основного языка программирования выбран Kotlin. На сегодняшний день это официальный язык для разработки Android-приложений, рекомендованный компанией Google. Его выбор оправдан следующими причинами:

- безопасность и лаконичность кода. Kotlin уменьшает вероятность ошибок, связанных с null-значениями, благодаря строгой системе типов. Синтаксис языка более лаконичен по сравнению с Java, что ускоряет разработку и упрощает поддержку кода;
- совместимость с Java. Kotlin беспрепятственно интегрируется с существующими Java-библиотеками, что позволяет использовать широкий спектр готовых решений для авторизации, работы с базами данных, тестирования и UI;
- высокая популярность и поддержка сообщества. Kotlin активно развивается и имеет множество обучающих ресурсов, что снижает порог вхождения для новых разработчиков и ускоряет устранение проблем;
- производительность. Программы на Kotlin компилируются в эффективный байткод, обеспечивающий высокую скорость работы на мобильных устройствах.

В качестве основной IDE выбрана AndroidStudio — официальный инструмент для Android-разработки. Это решение обеспечивает:

- интеграцию современных инструментов (UI-дизайнер, профайлер, отладчик), что ускоряет создание, тестирование и оптимизацию интерфейса приложения;

– удобство вёрстки интерфейсов благодаря визуальным редакторам и поддержке MaterialDesign, что важно для создания интуитивно понятного приложения для студентов и преподавателей;

– гибкость работы с эмуляторами различных устройств и версий Android, что позволяет заранее выявлять и устранять возможные проблемы совместимости.

В качестве СУБД выбрана SQLite, поскольку она оптимально подходит для мобильных приложений благодаря следующим преимуществам:

– интеграция в Android. SQLite интегрирована в Android SDK, не требует отдельной установки и полностью автономна;

– высокая производительность при небольших объёмах данных, достаточных для хранения пользователей, тестов, результатов и рейтингов в рамках корпоративного приложения;

– минимальные накладные расходы — вся база хранится локально, не требуется выделенный сервер или подключение к интернету для базовых операций, что повышает безопасность и конфиденциальность данных;

– поддержка транзакций, целостности и надёжности данных: наличие ограничений и уникальных индексов позволяет реализовать логику регистрации, авторизации, учёта результатов без риска дублирования или потери информации.

Преимущества выбранного стека перед аналогами:

– в отличие от кросс-платформенных решений (ReactNative, Flutter), нативная разработка на Kotlin обеспечивает максимально высокую производительность, полную совместимость со всеми возможностями Android и быстрый доступ к новым функциям платформы;

– использование SQLite предпочтительнее по сравнению с облачными или тяжёлыми серверными СУБД для мобильных корпоративных приложений, поскольку не требует постоянного интернет-соединения, экономит ресурсы устройства и обеспечивает мгновенный отклик;

– AndroidStudio и Kotlin дают разработчику доступ к новейшим библиотекам и средствам тестирования, что невозможно при использовании устаревших сред или менее популярных языков (Java, Xamarin, PhoneGap);

– локальное хранение (SQLite) гарантирует защиту персональных данных сотрудников, что особенно важно при обработке результатов аттестаций и тестирований в рамках внутренних корпоративных стандартов.

Выбранный стек технологий: Kotlin, AndroidStudio, SQLite оптимален для реализации мобильного приложения благодаря своей безопасности, производительности, простоте архитектуры и широкой поддержке сообщества. Он покрывает все основные задачи: регистрацию и авторизацию пользователей, создание и хранение тестов, анализ результатов и построение рейтингов, а также обеспечивает надёжную работу приложения на широком спектре современных устройств.

3.1 Существующие аналоги

В настоящее время на рынке представлены различные программные решения для тестирования и оценки знаний сотрудников, ориентированные на корпоративное обучение и развитие персонала. Проведём сравнительный анализ наиболее популярных аналогов по их преимуществам и недостаткам, с акцентом на функционал, близкий к целям и задачам приложения «Приложение для экспресс-оценки знаний сотрудников организации».

Сравнительный анализ аналогов программного обеспечения приведен в таблице 2.

Таблица 2 – Сравнение аналогов приложения «Приложение для экспресс-оценки знаний сотрудников организации»

Система	Плюсы	Минусы
iSpringLearn	Мощный инструментарий для создания и прохождения тестов; поддержка SCORM и аналитика; мобильное приложение; интеграция с HR-системами	Высокая стоимость; избыточный функционал для малых компаний; платная модель
Skillspace	Простота создания тестов; широкий выбор типов вопросов; таймеры и автопроверка; мобильная версия	Нет гибкой системы ролей (ограниченные права преподавателей и студентов); ограничение функций в бесплатной версии
GoogleForms	Бесплатно; простое создание форм и тестов; массовая рассылка; экспорт результатов в таблицы	Нет полноценной авторизации и ролевой модели; отсутствует система рейтингов; минимальные возможности по аналитике
Quizizz	Интерактивный, геймифицированный интерфейс; быстрый запуск тестов; мобильные приложения; рейтинг игроков	Фокус на образовании, а не на корпоративном тестировании; ограниченное управление доступом и ролями
SunRav WEB Class	Гибкая настройка тестов; поддержка больших групп; подробная статистика	Сложный интерфейс; устаревший дизайн; не всегда интуитивен для мобильных пользователей

Система	Плюсы	Минусы
ProProfsQuizMaker	Большой выбор шаблонов; возможность автоматических приглашений; отчёты по результатам	Ограничена гибкость в адаптации под внутренние бизнес-процессы; англоязычный интерфейс; часть функций доступна только в платной версии

Большинство из представленных в таблице 2 приложений являются корпоративными платформами, требующими крупных затрат на покупку и техническое обслуживание.

ПО iSpringLearn — одна из наиболее развитых платформ для онлайн-обучения и оценки персонала, включающая инструменты создания тестов, аналитику, поддержку мобильных устройств и интеграцию с корпоративными сервисами. Однако внедрение требует значительных затрат, а для малых и средних предприятий функционал может оказаться избыточным.

ПО Skillspare — российская платформа, позволяющая быстро создавать и настраивать тесты с разными типами вопросов, использовать таймеры, раздавать доступ по API и анализировать прохождение. Сервис прост в освоении, но роль преподавателя и механизмы контроля доступа реализованы на базовом уровне, а расширенные возможности доступны только по подписке.

Платформа GoogleForms — популярный бесплатный инструмент для быстрой организации опросов и тестов. Позволяет массово рассылать задания и собирать результаты. Однако отсутствуют регистрация пользователей, полноценная система ролей, рейтингование и расширенная аналитика, что ограничивает использование в корпоративных сценариях.

ПО Quizizz — платформа с возможностью проведения интерактивных тестов, автоматическим подсчётом баллов и формированием рейтингов среди участников. Поддерживаются мобильные приложения и быстрая регистрация пользователей. Основной акцент продукта — образовательная среда:

функционал по управлению ролевым доступом и интеграции с корпоративными HR-системами ограничен.

ПО ProProfsQuizMaker — конструктор онлайн-тестов с возможностью автоматического распределения, управления расписанием тестирования, аналитики результатов и поддержкой различных типов вопросов. Недостатки включают необходимость подписки для расширенного функционала и отсутствие русскоязычного интерфейса.

ПО SunRav WEB Class — система тестирования с детальной статистикой, поддержкой кастомных отчётов, большим набором настроек для корпоративных клиентов. Однако интерфейс выглядит устаревшим, и мобильные возможности ограничен.

Существующие решения позволяют покрыть базовый функционал для тестирования сотрудников:

- создание тестов;
- проведение и анализ результатов.

Однако ни один из рассмотренных аналогов не реализует в полном объёме следующие возможности в формате отдельного мобильного приложения:

- гибкую двухуровневую ролевою модель (студент/преподаватель) с расширенным управлением доступом;
- просмотр и сравнение индивидуального рейтинга;
- удобное мобильное прохождение тестов с локальным хранением данных;
- приглашения и подтверждения на тестирование;
- низкий порог внедрения для малых и средних предприятий.

Это обосновывает выбор собственной разработки с адаптацией функционала под конкретные бизнес-процессы, ориентацией на мобильность, простоту и безопасность данных.

4 Проектирование системы

Сотрудник отдела кадров предприятия анализирует информацию о необходимости тестирования сотрудников или проведении обучения в рамках повышения квалификации.

Сотрудники по приказу регистрируются на процедуру тестирования и после одобрения участия в тестировании выполняют тест, с моментальным подтверждением результата после окончания тестирования.

Приложение предназначено для проведения тестирования и обработки сведений о тестируемых и результатах теста.

Для начала проектирования, реализуем UML диаграммы для понимания основных блоков и потоков данных программы.

Диаграмма прецедентов (рисунок 1) визуализирует реализуемую систему с точки зрения взаимодействия пользователей (актеров) с приложением. Диаграмма определяет: роли пользователей (Студент, Преподаватель), ключевые сценарии использования (usecases), взаимосвязи между прецедентами.

На диаграмме прецедентов (рисунок 1) студент – это пользователь, проходящий тестирование. Студент регистрируется в системе, ожидает подтверждения от преподавателя. Преподаватель – администратор системы, который управляет пользователями, тестами и анализирует результаты.

Прецеденты пользователя студент:

- регистрация (создание учетной записи с указанием личных данных. Включает статус «Ожидание подтверждения регистрации»);
- авторизация (вход в систему под логином пользователя после подтверждения преподавателем, для чего требуется подтверждение регистрации);
- просмотр тестов (отображение списка доступных тестов с детализацией (название, сроки, статус, прогресс));

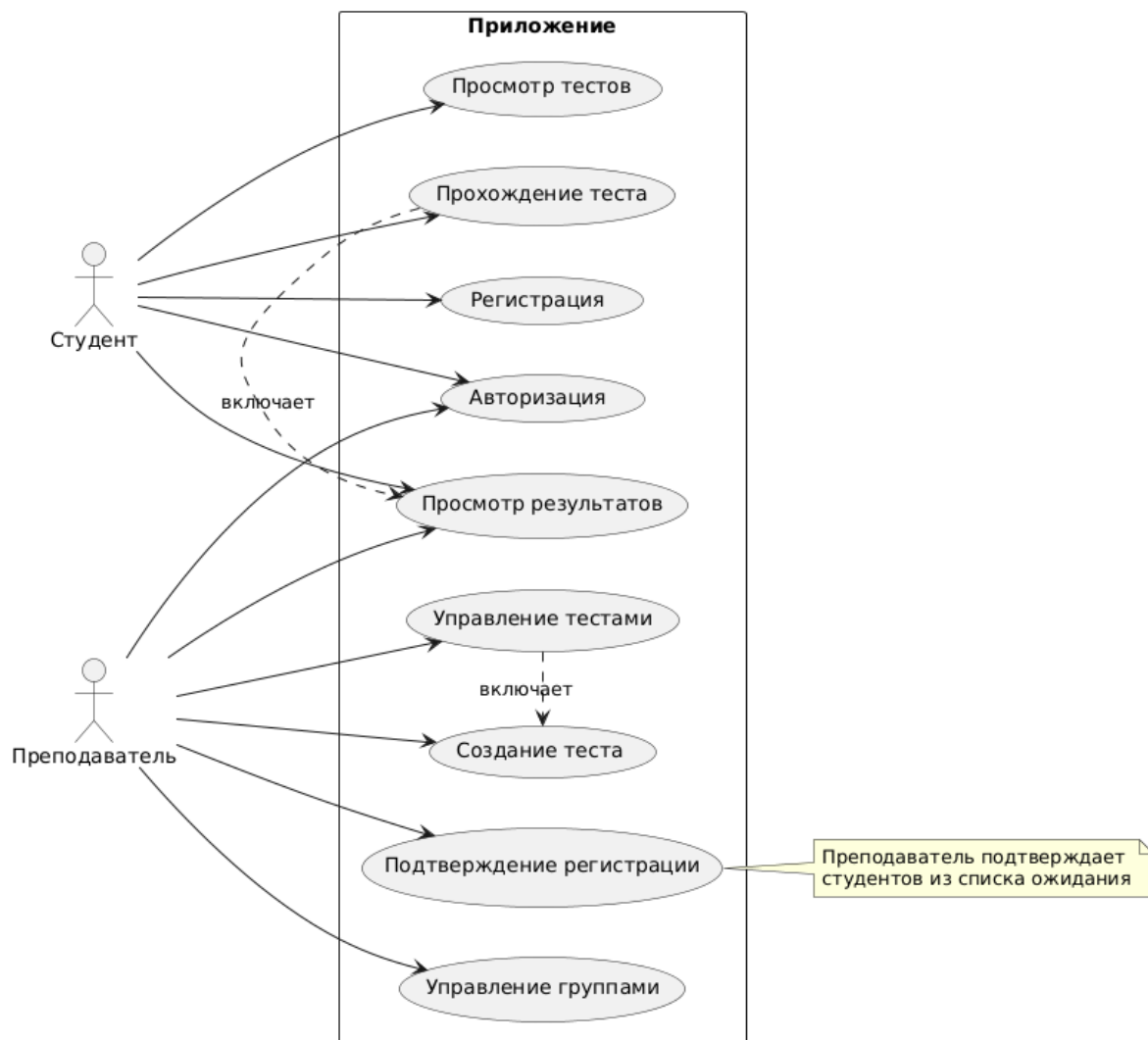


Рисунок 1 – Контекстная диаграмма

– прохождение теста (последовательное выполнение вопросов теста. Включает режим просмотр результатов);

– просмотр результатов (анализ правильных/неправильных ответов после завершения теста).

Прецеденты пользователя преподаватель:

- авторизация (вход в систему с административными правами);
- подтверждение регистрации (активация учетных записей студентов из списка ожидания);
- управление тестами (просмотр/редактирование существующих тестов);

- создание теста (разработка нового теста: добавление вопросов, настройка параметров (таймер, сроки));
- управление группами (распределение студентов по группам для назначения тестов);
- просмотр результатов (анализ статистики тестирования по студентам/группам):

Между акторами на диаграмме прецедентов (рисунок 1) существуют следующие зависимости:

- студент не может авторизоваться без предварительного подтверждения преподавателем;
- преподаватель управляет доступом к тестам и результатам;
- преподаватель вручную проверяет заявки студентов в разделе «Ожидающие регистрации». Только после активации студент получает доступ к системе;
- все операции (регистрация, создание тестов, результаты) используют WebSocket для обмена данными между устройствами студентов и преподавателя.

Описание пользователей и связей между ними позволяет сделать следующий шаг и реализовать диаграммы последовательности для каждого вида пользователей (рисунок 2).

Диаграммы последовательности (рисунок 2) визуализируют пошаговое взаимодействие пользователей с системой в хронологическом порядке. Они демонстрируют:

- последовательность операций для ключевых сценариев;
- взаимодействие между компонентами системы (UI, бизнес-логика, БД);

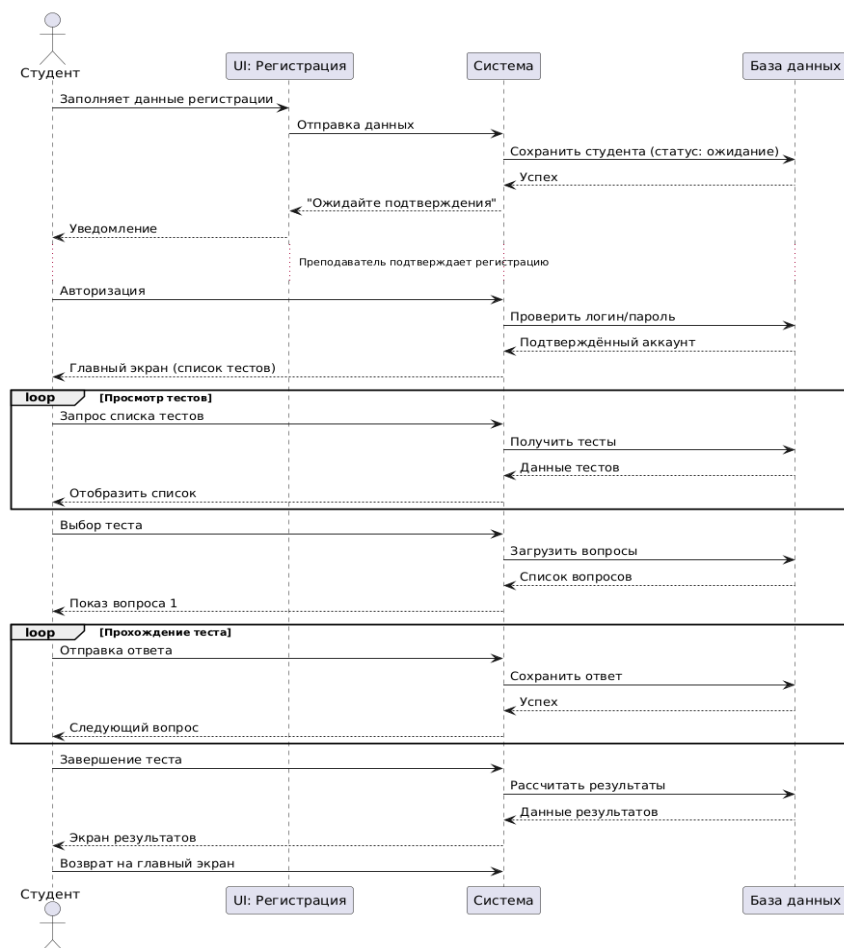


Рисунок 2 – Диаграмма последовательности. Студент

– потоки данных и синхронизацию через WebSocket.

Сценарий включает полный цикл взаимодействия студента с системой: регистрацию, авторизацию, выбор теста, прохождение и просмотр результатов.

Детализация этапов выполнения программы согласно диаграмме последовательности:

– этап 1.Регистрация:

- 1) заполнение данных (Студент–Интерфейс). Студент вводит регистрационные данные: ФИО, логин, пароль;
- 2) отправка данных (Интерфейс–Система). Интерфейс передает данные в системный модуль;
- 3) сохранение в БД (Система–База данных). Система сохраняет запись в таблице User со статусом ожидание;

- 4) уведомление (Система –Студент). Студент получает сообщение: «Ожидайте подтверждения преподавателем»;
- этап 2.Авторизация (после подтверждения):
 - 1) аутентификация (Студент–Система). Студент вводит подтвержденные учетные данные;
 - 2) верификация (Система – База данных). Проверка статуса аккаунта (active) в таблице User;
 - 3) отображение главного экрана (Система – Студент). Показ списка доступных тестов с параметрами: название теста, срок сдачи, прогресс (0/6), статус выполнения;
- этап 3.Работа со списком тестов (loop):
 - 1) Запрос данных (Студент–Система);
 - 2) система формирует SQL-запрос;
 - 3) обработка данных (База данных–Система). Фильтрация по сроку действия и группе студента;
- 4) визуализация (Система–Студент). Отображение карточек тестов с индикаторами статуса;
- этап 4.Прохождение теста:
 - 1) старт теста (Студент–Система). Система создает запись в таблице Session со статусом in_progress;
 - 2) загрузка вопросов (Система–База данных). Выполняется запрос:
 - 3) интерактивное тестирование (loop). Для каждого вопроса формируются показ диалогового окна с вариантами ответов, фиксация ответа в таблице Result (JSON-массив выбранных ID), обновление прогресса в реальном времени;
- этап 5.Завершение теста:
 - 1) финализация (Студент–Система) и расчет результатов;
 - 2) формирование отчета (Система–Студент). Визуализация правильных/неправильных ответов;

3) возврат на главный экран и автоматическое обновление статуса теста.

Ключевыми особенностями реализации являются:

- синхронизация через WebSocket: При сохранении ответа система отправляет пуш-уведомление преподавателю;
- офлайн-работа: Данные кэшируются в SQLite при потере соединения;
- таймер теста: Обратный отсчет на клиентской стороне с сохранением состояния.

Диаграмма последовательности преподавателя (рисунок 3) имеет следующие этапы:

Авторизация и управление доступом;

- вход в систему (Преподаватель – Интерфейс). Преподаватель вводит логин/пароль.

- проверка прав на авторизацию (Система – База данных).

Верификация роли преподаватель в БД.

- работа с ожидающими регистрации (loop):
- преподаватель–система. Запрос списка неподтвержденных студентов.

- система–База данных. Извлечение данных – отображение списка.

- для каждого подтверждения (цикл):

- преподаватель–Система: Выбор кнопки «Подтвердить».

- система–База данных. Обновление статуса на активен.

- система–Преподаватель. Обновление интерфейса в реальном времени.

- создание теста:

- инициация (Преподаватель–Система). Преподаватель выбирает кнопку «Создать тест».

- сохранение параметров (Преподаватель–Система). Ввод названия, сроков, настроек таймера.

- система–База данных: Сохранение теста – получение ID теста.
- добавление вопросов (loop). Для каждого вопроса:
- преподаватель–Система: Ввод вопроса/вариантов ответов;
- система–База данных: Сохранение – синхронизация через WebSocket.

Механизм WebSocket в данной системе позволяет мгновенно делать рассылку созданных преподавателем тестов на устройства студентов.

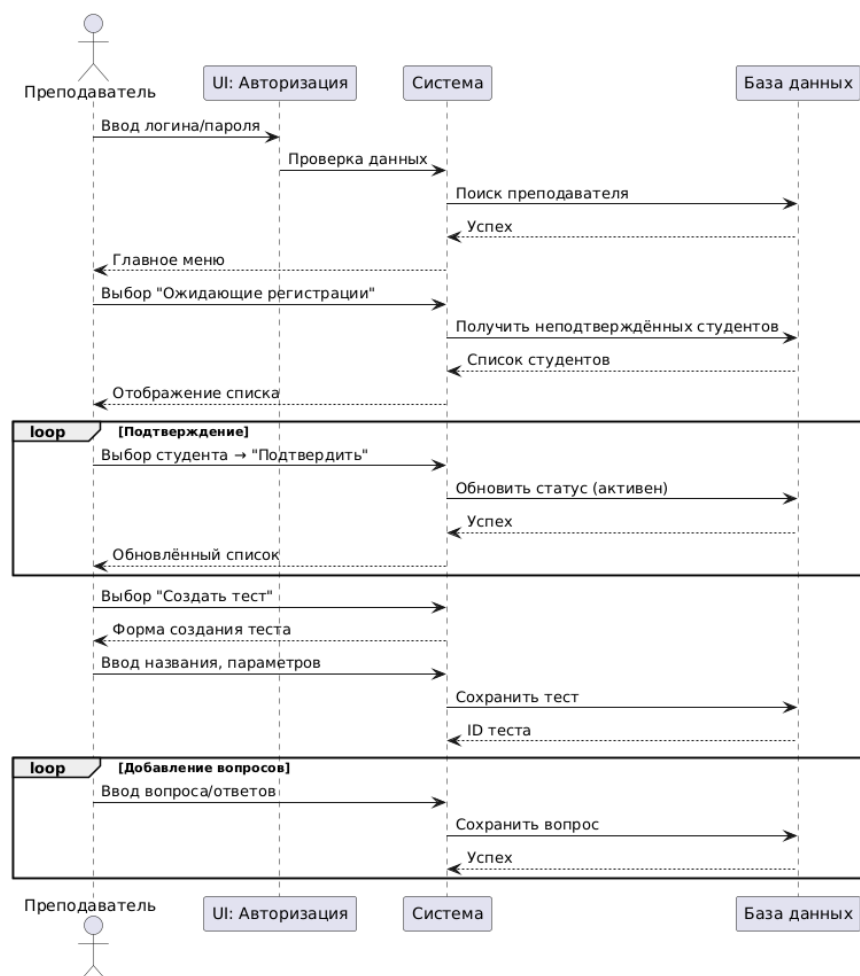


Рисунок 3 – Диаграмма последовательности. Преподаватель

Используя последовательность представлений (рисунок 3), создается диаграмма компонентов (рисунок 4).

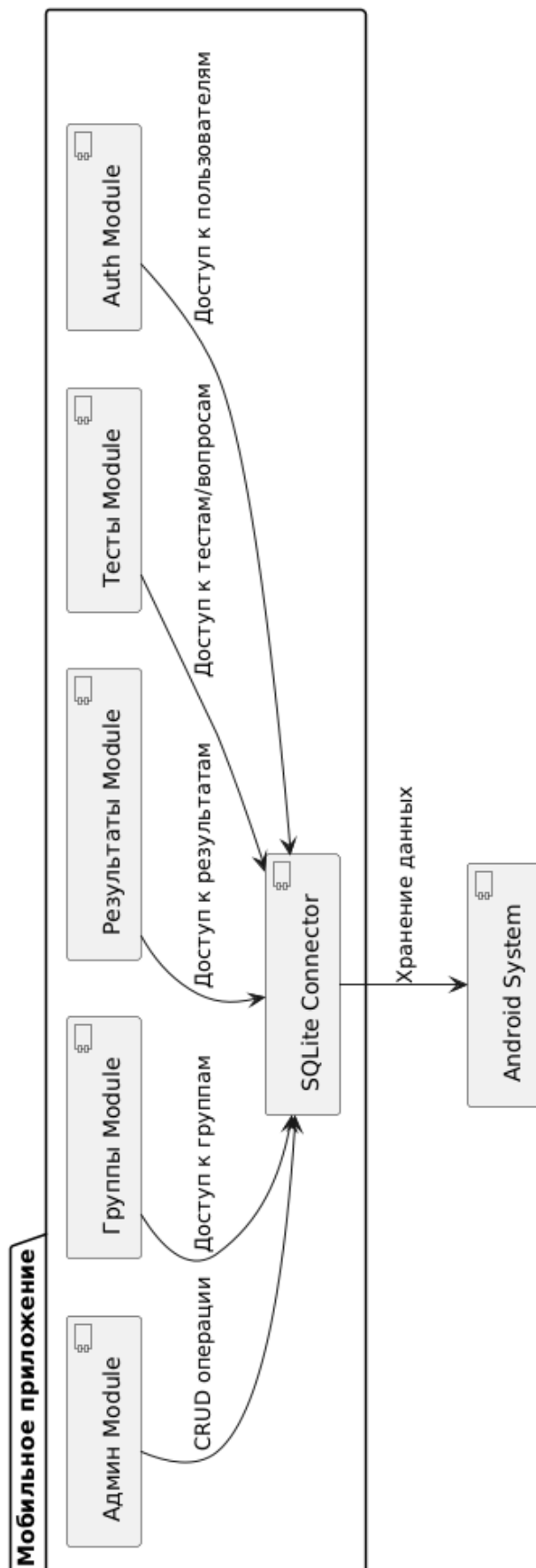


Рисунок 4 – Контекстная диаграмма

Диаграмма компонентов отображает логическую структуру системы: ключевые программные модули, зависимости между компонентами, интеграцию с платформой Android.

Таблица 1–Описание компонентов и связей диаграммы развертывания

Компонент	Функционал	Зависимости
AuthModule	Регистрация/авторизация, проверка прав.	SQLiteConnector
Тесты Module	Управление тестами: создание, загрузка вопросов, таймеры, прогресс.	SQLiteConnector, WebSocket
Результаты Module	Расчет баллов, формирование отчетов, статистика.	SQLiteConnector
Группы Module	Распределение студентов по группам, назначение тестов.	SQLiteConnector, WebSocket
Админ Module	Функции преподавателя: подтверждение регистраций, CRUD-операции.	Все модули
SQLiteConnector	Единый шлюз для работы с БД: ORM-маппинг, транзакции, миграции.	Android System (File I/O)

Логическая декомпозиция системы на высокосвязные модули позволяют отобразить физическую архитектуру системы используя диаграмму развертывания (рисунок 5) для:

- распределение компонентов по аппаратным узлам;
- способы взаимодействия между устройствами;
- точки хранения данных и сетевой инфраструктуры.

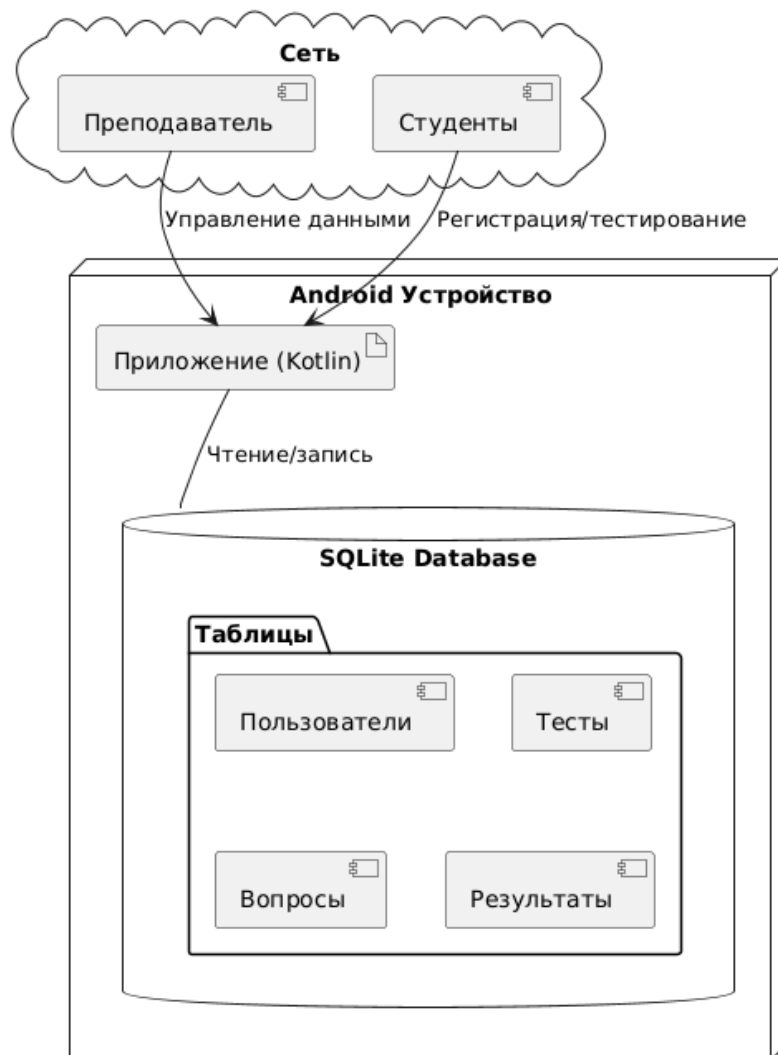


Рисунок 5 – Диаграмма развертывания

Описание компонентов и связей диаграммы развертывания представлено в таблице 2.

Таблица 2–Описание компонентов и связей диаграммы развертывания

Узел/Компонент	Описание
Android Устройство	Физическое устройство (смартфон/планшет) пользователя.
Приложение (Kotlin)	Исполняемый артефакт: клиентская часть на Kotlin.

Узел/Компонент	Описание
SQLiteDatabase	Локальное хранилище на устройстве. Структура: - пользователи: логины, пароли, роли, статусы подтверждения; - тесты: названия, сроки сдачи, параметры таймеров; - вопросы: формулировки, варианты ответов, правильные решения; - результаты: ответы студентов, баллы, прогресс.
Сеть (WebSocket)	Облако коммуникации: обеспечивает синхронизацию в реальном времени.
Преподаватель	Устройство преподавателя: управляет данными через WebSocket-соединение.
Студенты	Устройства студентов: регистрация и тестирование через WebSocket.

Особенности описанной архитектуры программного приложения заключается в том, что используются:

- децентрализованная модель. Каждое устройство имеет автономную БД и работу без интернета. WebSocket обеспечивает синхронизацию при появлении сети;
- оптимизация трафика. Инкрементальные обновления (только измененные данные). Сжатие JSON-пакетов;
- механизм разрешения коллизий. Приоритет данных преподавателя. Временные метки последнего изменения.

В следующем разделе требуется реализовать базу данных на основании проекта программного обеспечения, реализованного выше.

5 Проектирование системы

На основе анализа предметной области разработана концептуальная модель данных мобильного приложения .

Концептуальная модель данных состоит из сущностей, связей и правил. Далее идет описание объектов модели, на основании которой реализуются логическая и физическая модели базы данных.

Сущности и их назначение:

- User(Хранит информацию о пользователях (студентах и преподавателях));
- Group(Представляет учебные группы студентов);
- Test(Содержит данные о тестах (название, сроки выполнения));
- Question(Вопросы в составе теста с указанием типа);
- Answer (Варианты ответов на вопросы);
- Session(Информация о сессии прохождения теста);
- Result(Результаты ответов на конкретные вопросы);

ключевые связи;

- User–Group (Многие-ко-многим);
 - 1) преподаватель управляет несколькими группами;
 - 2) студент может состоять в нескольких группах;
- Test–Group (Многие-ко-многим);
 - 1) тест назначается нескольким группам;
 - 2) группа может иметь несколько тестов;
- Test–Question (Один-ко-многим);
 - 1) тест содержит несколько вопросов;
 - 2) каждый вопрос принадлежит одному тесту;
- Question–Answer (Один-ко-многим);
 - 1) вопрос имеет несколько вариантов ответов;
 - 2) ответ принадлежит одному вопросу;
- Session–Result (Один-ко-многим);

- 1) в сессии фиксируются результаты ответов;
- 2) каждый результат относится к одной сессии;
 - бизнес-правила;
- 1) преподаватель подтверждает регистрацию студентов;
- 2) студент видит только тесты, назначенные его группам;
- 3) каждая сессия тестирования привязана к конкретному пользователю и тесту;
- 4) результаты хранят выбранные ответы на каждый вопрос;
 - особенности модели;
- 1) разделение ролей (студент/преподаватель) определяет доступные действия;
- 2) статус сессии (не начата/в процессе/завершена) управляет процессом тестирования;
- 3) групповая принадлежность определяет доступ к тестам.

На основе концептуальной модели данных была разработана логическая модель БД (таблица 3).

Таблица 3–Описание компонентов и связей логической модели БД

Таблица	Атрибуты (тип данных)	Описание
User	id INTEGER PK login TEXT UNIQUE password TEXT role TEXT status TEXT	Статус: pending/active
Group	id INTEGER PK name TEXT	Группы студентов
UserGroup	user_id INTEGER FK User group_id INTEGER FK – Group	Связь многие-ко-многим

Таблица	Атрибуты (тип данных)	Описание
Test	id INTEGER PK title TEXT time_limit INTEGER deadline DATETIME created_by INTEGER FK – User	time_limit=0 – без ограничений
TestGroup	test_id INTEGER FK – Test group_id INTEGER FK – Group	Назначение тестов группам
Question	id INTEGER PK text TEXT type TEXT test_id INTEGER FK – Test	Тип: single/multiple
Answer	id INTEGER PK text TEXT is_correct INTEGER question_id INTEGER FK – Question	is_correct: 0/1
Session	id INTEGER PK user_id INTEGER FK – User test_id INTEGER FK – Test status TEXT start_time DATETIME score INTEGER	Статус: not_started/in_progress/finished
Result	id INTEGER PK session_id INTEGER FK – Session question_id INTEGER FK – Question selected_answers TEXT	JSON: [answer_id1, answer_id2]

Разработанная логическая модель имеет 3 форму нормализации. Доказательством данного факта является следующая возможность проверки БД:

- 1NF: Все атрибуты атомарны (например, selected_answers хранит JSON массива);
- 2NF: Отсутствуют частичные зависимости (ключи: id для всех таблиц);
- 3NF: Нет транзитивных зависимостей (например, Test.created_by зависит только от PK).

На основе логической модели данных была разработана физическая модель БД (таблица 4 – таблица 11).

Таблица 4 – Описание атрибутов сущности User (Пользователи)

Имя атрибута	Назначение	Тип данных	Первичный ключ	Внешний ключ
id	Уникальный идентификатор пользователя	INTEGER	PK	
login	Уникальное имя для входа в систему	TEXT		
password	Зашифрованный пароль (SHA-256)	TEXT		
role	Роль пользователя: student/teacher	TEXT		
status	Статус аккаунта: pending/active	TEXT		

Таблица 5 – Описание атрибутов сущности Group (Группы)

Имя атрибута	Назначение	Тип данных	Первичный ключ	Внешний ключ
id	Уникальный идентификатор группы	INTEGER	PK	
name	Название группы (уникальное)	TEXT		

Таблица 6 – Описание атрибутов сущности UserGroup (Связь пользователей и групп)

Имя атрибута	Назначение	Тип данных	Первичный ключ	Внешний ключ
user_id	Идентификатор пользователя	INTEGER	PK(составной)	User.id (ON DELETE CASCADE)
group_id	Идентификатор группы	INTEGER	PK (составной)	Group.id (ON DELETE

Таблица 7– Описание атрибутов сущности Test (Тесты)

Имя атрибута	Назначение	Тип данных	Первичный ключ	Внешний ключ
id	Уникальный идентификатор теста	INTEGER	PK	
title	Название теста	TEXT		
time_limit	Время на выполнение (минут), 0 = без ограничений	INTEGER		

Имя атрибута	Назначение	Тип данных	Первичный ключ	Внешний ключ
deadline	Срок сдачи теста (ISO8601)	DATETIME		
created_by	Создатель теста (преподаватель)	INTEGER		

Таблица 8 – Описание атрибутов сущности TestGroup (Назначение тестов группам)

Имя атрибута	Назначение	Тип данных	Первичный ключ	Внешний ключ
test_id	Идентификатор теста	INTEGER	PK(составной)	Test.id (ON DELETE CASCADE)
group_id	Идентификатор группы	INTEGER	PK (составной)	Group.id (ON DELETE CASCADE)

Таблица 9–Описание атрибутов сущности Question (Вопросы)

Имя атрибута	Назначение	Тип данных	Первичный ключ	Внешний ключ
id	Уникальный идентификатор вопроса	INTEGER	PK	
text	Текст вопроса	TEXT		
type	Тип вопроса: single/multiple	TEXT		

Имя атрибута	Назначение	Тип данных	Первичный ключ	Внешний ключ
test_id	Идентификатор теста	INTEGER		Test.id (ON DELETE CASCADE)

Таблица 10–Описание атрибутов сущности Answer (Ответы)

Имя атрибута	Назначение	Тип данных	Первичный ключ	Внешний ключ
id	Уникальный идентификатор ответа	INTEGER	PK	
text	Текст варианта ответа	TEXT		
is_correct	Флаг правильности ответа (0/1)	INTEGER		
question_id	Идентификатор вопроса	INTEGER		Question.id (ON DELETE CASCADE)

Таблица 11–Описание атрибутов сущности Session (Сессии тестирования)

Имя атрибута	Назначение	Тип данных	Первичный ключ	Внешний ключ
id	Уникальный идентификатор сессии	INTEGER	PK	
user_id	Идентификатор пользователя	INTEGER		User.id

Имя атрибута	Назначение	Тип данных	Первичный ключ	Внешний ключ
test_id	Идентификатор теста	INTEGER		Test.id
status	Статуссессии: not_started/in_progress/finished	TEXT		
start_time	Время начала тестирования (ISO8601)	DATETIME		
score	Набранные баллы	INTEGER		

Таблица 12–Описание атрибутов сущности Result (Результаты ответов)

Имя атрибута	Назначение	Тип данных	Первичный ключ	Внешний ключ
id	Уникальный идентификатор результата	INTEGER	PK	
session_id	Идентификатор сессии	INTEGER		Session.id (ON DELETE CASCADE)
question_id	Идентификатор вопроса	INTEGER		Question.id
selected_answers	JSON-массив ID выбранных ответов (например [1,3])	TEXT		

На основании таблиц физической модели реализуем схему физическом модели базы данных (рисунок 6).

Ключевыми характеристиками физической модели являются:

- INTEGER для всех идентификаторов и числовых значений;
- TEXT для строковых данных и JSON;
- DATETIME для временных меток в формате ISO8601 (YYYY-MM-DD HH:MM:SS).

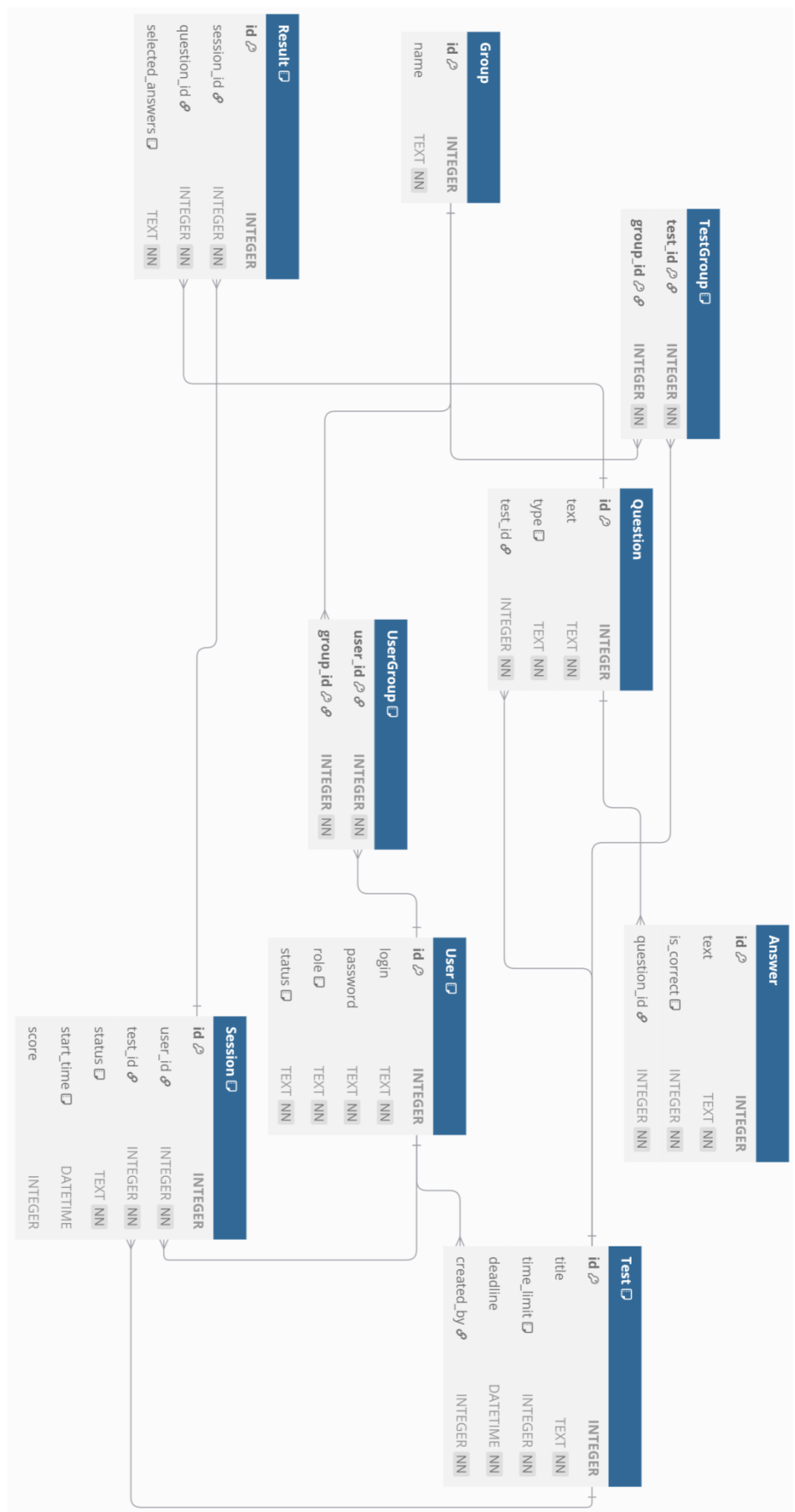


Рисунок 6 – Физическая модель базы данных

6 Программная реализация мобильного приложения

Мобильное приложение реализовано с использованием современного стека технологий Android-разработки (рисунок 7, 8):

- язык программирования: Kotlin (100% кодовой базы);
- архитектура: MVVM (Model-View-ViewModel);
- локальное хранилище: Room (SQLite);
- сетевое взаимодействие: WebSocket.

Приложение поддерживает две роли пользователей: студенты (проходят тестирование) и преподаватели (управляют учебным процессом).

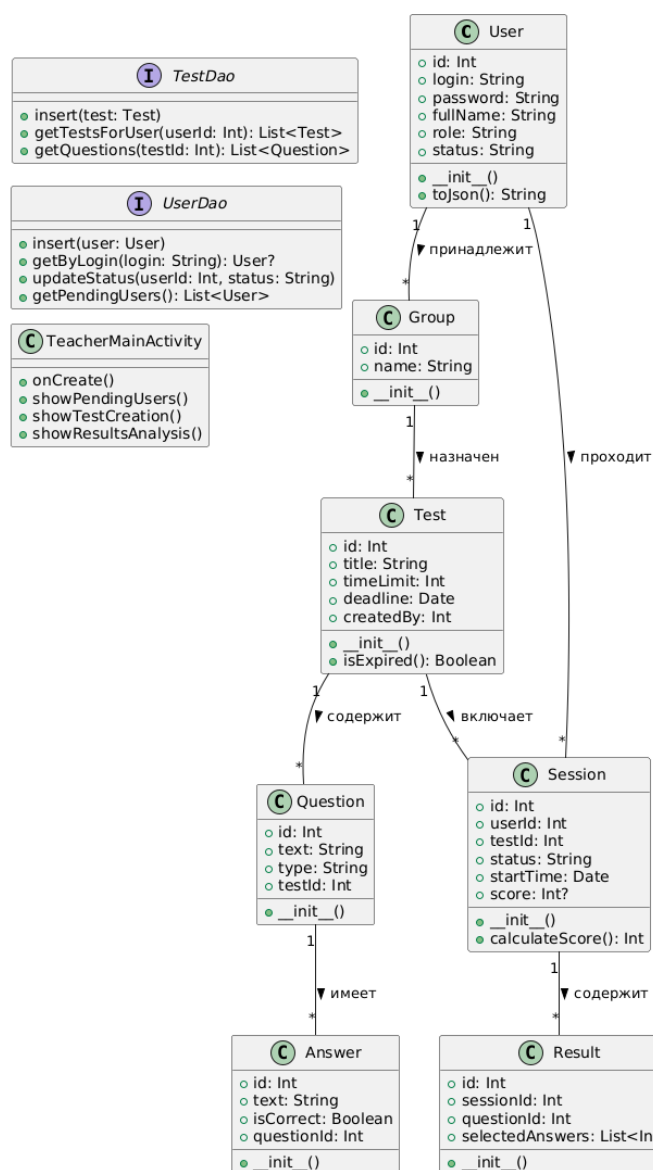


Рисунок 7 – Диаграмма классов. Часть 1

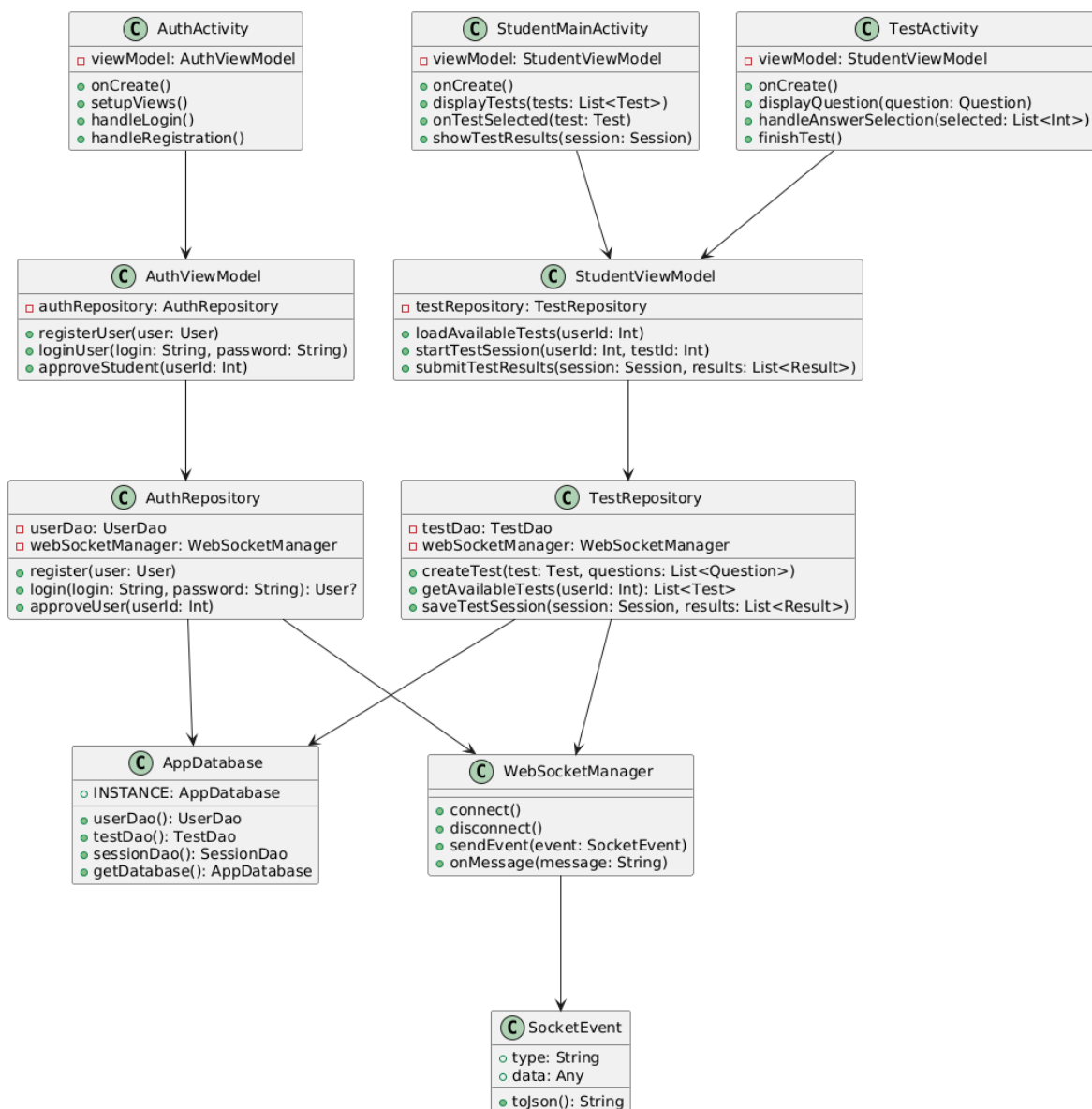


Рисунок 8 – Диаграмма классов. Часть 2

Ключевые компоненты реализованной программы:

- модель предметной области: User, Test, Question и иное;
- слой данных: Room Database + WebSocket;
- бизнес-логика: Репозитории (AuthRepository, TestRepository) ;
- управление состоянием: ViewModel;
- пользовательский интерфейс: Activities и JetpackCompose.

В таблице 11 представлено описание классов и функций реализованного приложения.

Таблица 11–Описание атрибутов сущности Session (Сессии тестирования)

Класс	Назначение	Ключевые функции/Методы
User	Представление пользователя системы	toJson(): Сериализация в JSON isTeacher(): Проверка роли преподавателя
Group	Учебная группа студентов	
Test	Тест для оценки знаний	isExpired(): Проверка просроченности теста
Question	Вопрос в тесте	
Answer	Вариант ответа на вопрос	
Session	Сессия прохождения теста	calculateScore(): Расчет баллов за тест
Result	Результат ответа на конкретный вопрос	
AppDatabase	Центральный класс базы данных (Room)	getDatabase(): Получение экземпляра БД userDao(): Доступ к пользователям
UserDao	Интерфейс доступа к данным пользователей	getByLogin(): Поиск по логину getPendingUsers(): Неподтвержденные пользователи
TestDao	Интерфейс доступа к данным тестов	getTestsForUser(): Тесты доступные пользователю
AuthRepository	Репозиторий для операций аутентификации	register(): Регистрация нового пользователя approveUser(): Подтверждение студента

Класс	Назначение	Ключевые функции/Методы
TestRepository	Репозиторий для работы с тестами	createTest(): Создание нового теста saveTestSession(): Сохранение результатов
AuthViewModel	ViewModel для экранов аутентификации	registerUser(): Обработка регистрации loginUser(): Вход в систему
StudentViewModel	ViewModel для студенческого функционала	loadAvailableTests(): Загрузка тестов submitTestResults(): Отправка результатов
AuthActivity	Экран авторизации и регистрации	handleLogin(): Обработка входа handleRegistration(): Обработка регистрации
StudentMainActivity	Главный экран студента	displayTests(): Показать список тестов onTestSelected(): Начать тестирование
TeacherMainActivity	Главный экран преподавателя	showTestCreation(): Открыть создание теста showResultsAnalysis(): Анализ результатов
TestActivity	Экран прохождения теста	displayQuestion(): Показать вопрос finishTest(): Завершить тестирование
WebSocketManager	Управление WebSocket соединением	sendEvent(): Отправка событий onMessage(): Обработка входящих сообщений

Класс	Назначение	Ключевые функции/Методы
SocketEvent	Модель события для передачи через WebSocket	toJson(): Сериализация в JSON формат

Ключевые особенности реализации мобильного приложения:

- преподаватель создает тест и сохраняется в локальной БД;
- отправляется WebSocket-событие TEST_CREATED;
- устройства студентов получают событие → сохраняют тест в свою БД;
- студенты видят новый тест в списке доступных.

Обработка конфликтов:

- приоритет данных преподавателя;
- временные метки последнего изменения;
- автоматическое разрешение конфликтов.

Офлайн-работа:

- локальная БД как источник истины;
- отложенная синхронизация при восстановлении сети;
- кэширование критичных данных;

Безопасность:

- шифрование паролей (соль)
- HTTPS для WebSocket
- валидация входных данных

7 Оценка затрат на разработку и внедрение программного средства

Экономическое обоснование включает расчет затрат на разработку и оценку потенциального экономического эффекта от внедрения решения.

Разработка мобильного приложения осуществлялась с учетом оптимизации затрат и максимального использования существующих ресурсов. Основные затраты на разработку приведены ниже в виде затрат на помещение, оборудование, программное обеспечение трудовые и иные затраты.

Помещение и инфраструктура имеют следующие характеристики:

- использована собственная площадь: 70 м²;
- содержание помещения: 8 000 руб./мес;
- период разработки: 2 месяца;
- итого: $8\,000 \times 2 = 16\,000$ руб.

Затраты на оборудование приведены в таблице 12.

Таблица 12–Описание затрат на оборудование

Оборудование	Кол-во	Цена за единицу (руб.)	Сумма (руб.)
Компьютер (сборка)	5	60 000	300 000
МФУ Canon MF3010	1	11 500	11 500
Клавиатура и мышь Logitech	5	1 000	5 000
Монитор Samsung 24"	5	15 000	75 000
ИТОГО			391 500

Затраты на приобретение и обслуживание программного обеспечения приведены в таблице 13.

Таблица 13–Описание используемого программного обеспечения

ПО	Цена (руб.)	Срок действия
Windows 11 Pro (5 лиц.)	40 000	Бессрочно

ПО	Цена (руб.)	Срок действия
JetBrainsAllProducts	150 000	1 год
ИТОГО	190 000	

Оценка стоимости трудовых ресурсов приведена в таблице 14.

Таблица 14 – Трудовые ресурсы

Специалист	Кол-во	З/п в мес (руб.)	Период	Сумма (руб.)
Руководитель проекта	1	80 000	2 мес.	160 000
Android-разработчик	2	70 000	2 мес.	280 000
Backend-разработчик	1	60 000	2 мес.	120 000
Тестировщик	1	50 000	2 мес.	100 000
ИТОГО				660 000

Дополнительные затраты включены в стоимость программного обеспечения и представляют следующие статьи расходов:

- дополнительная з/п (15% от основной): $660\,000 \times 0.15 = 99\,000$ руб;
- страховые взносы (30%);
- 1) ОПС (22%): $759\,000 \times 0.22 = 166\,980$ руб;
- 2) ВНиМ (2.9%): $759\,000 \times 0.029 = 22\,011$ руб;
- 3) ОМС (5.1%): $759\,000 \times 0.051 = 38\,709$ руб;
- Всего взносов: 227 700 руб.

Сводная таблица затрат показывает общие расходы на реализацию мобильного приложения (таблица 15).

Таблица 15 – Сводная таблица затрат

Статья затрат	Сумма (руб.)
Оборудование	391 500
Программное обеспечение	190 000

Статья затрат	Сумма (руб.)
Основная з/п	660 000
Дополнительная з/п	99 000
Страховые взносы	227 700
Содержание помещения	16 000
ОБЩАЯ СУММА	1 584 200

Помимо затрат ожидается получение прибыли, для чего помимо затрат рассчитана экономическая эффективность проекта. Для чего первым делом были вычислены преимущества внедрения:

- сокращение времени оценки знаний на 70%;
- уменьшение затрат на проведение тестирований;
- автоматизация формирования отчетности;
- возможность массового охвата сотрудников.

В первую очередь были выявлены статьи экономии (таблица 16).

Таблица 16 – Сводная таблица экономии

Показатель	До внедрения, руб.	После внедрения, руб.	Экономия, руб.
Время на оценку 1 сотрудника (час)	3.0	0.5	2.5
Стоимость часа работы специалиста	1 500	1 500	-
Количество оценок в год	200	500	+300
Годовая экономия			1 125 000

Вторым шагом расчета экономической эффективности стал расчет окупаемости проекта:

- срок окупаемости: $1\,584\,200 / 1\,125\,000 \approx 1.4$ года
- ROI за 3 года: $(3 \times 1\,125\,000 - 1\,584\,200) / 1\,584\,200 \times 100\% = 113\%$

В стратегию получения прибыли была включена и модель распространения ПО, как важная часть рекламы и продвижения:

- бесплатная базовая версия: Для организаций до 50 сотрудников;
- премиум-подписка: 50 руб./сотрудник в месяц;
- прогнозируемый охват: 30 организаций в первый год;
- ожидаемая выручка;

1) 1 год: $30 \text{ орг.} \times 50 \text{ сотр.} \times 50 \text{ руб.} \times 12 \text{ мес.} = 900\,000 \text{ руб.}$

2) 2 год: $50 \text{ орг.} \times 100 \text{ сотр.} \times 50 \text{ руб.} \times 12 \text{ мес.} = 3\,000\,000 \text{ руб.}$

На основании расчетов выше, сделан вывод, разработка мобильного приложения экономически целесообразна:

- общие затраты на разработку: 1 584 200 руб;
- годовая экономия для заказчика: 1 125 000 руб;
- срок окупаемости: 1.4 года;
- потенциальная выручка: 900 000 руб./год (1 год), 3 000 000 руб./год(2 год).

Проект обладает значительным потенциалом масштабирования и может быть адаптирован для различных отраслей, включая образовательные учреждения и корпоративные системы обучения.

8 Безопасность жизнедеятельности

Разработка мобильного приложения осуществлялась с соблюдением строгих норм безопасности, регламентированных следующими документами:

- СанПиН 2.2.2/2.4.1340-03 «Гигиенические требования к персональным электронно-вычислительным машинам»;
- Трудовой кодекс РФ (ст. 209-231);
- ГОСТ 12.0.003-2015 «Система стандартов безопасности труда».

Ключевые требования к рабочему месту:

- эргономика рабочего пространства:
 - 1) расстояние от глаз до монитора: 50-70 см;
 - 2) угол наклона монитора: 15-20° ниже уровня глаз;
 - 3) использование мониторов с антибликовым покрытием и частотой обновления ≥ 75 Гц;
 - 4) регулируемое кресло с поддержкой поясницы и подголовником.
- Режим работы:
 - 1) обязательные перерывы каждые 45-50 минут работы;
 - 2) гимнастика для глаз (20-20-20: каждые 20 мин смотреть на объект в 20 футах ≥ 20 сек);
 - 3) производственная гимнастика 2 раза в день.
- Параметры рабочей зоны:
 - 1) минимальная площадь на рабочее место: 6 м²;
 - 2) глубина стола: ≥ 800 мм;
 - 3) пространство для ног: высота ≥ 600 мм, глубина ≥ 450 мм.

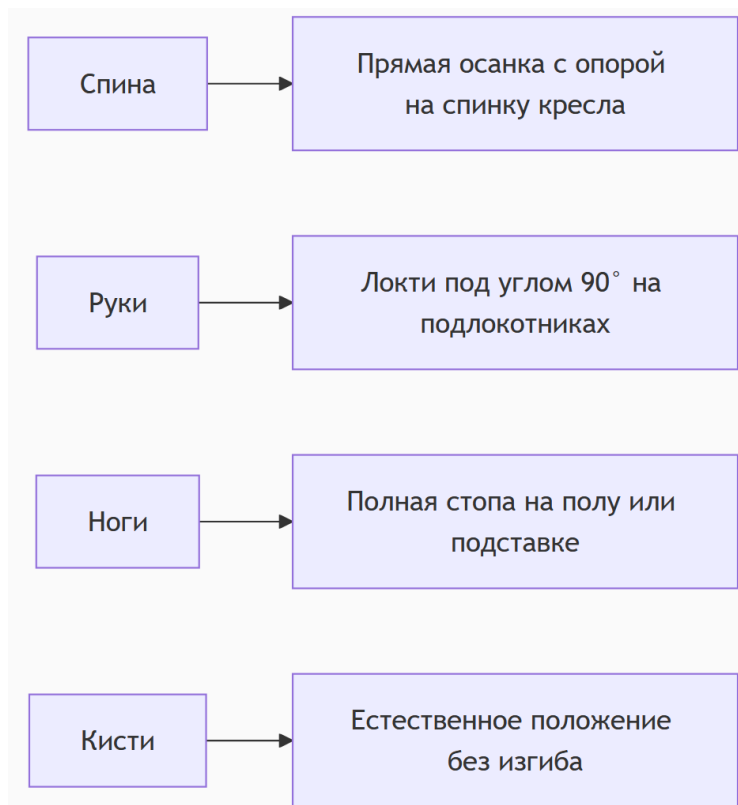


Рисунок 9 – Основные требования к осанке.

Требования к микроклимату (таблица 16) в рабочем помещении для разработчиков ПО включают требования к: освещению, вентиляции, шумоподавлению.

Таблица 16 –Оптимальные параметры среды

Параметр	Зимний период	Летний период	Допустимые пределы
Температура	22-24°C	23-25°C	19-29°C
Влажность	55-65%	50-60%	40-70%
Скорость воздуха	≤0.1 м/с	≤0.2 м/с	≤0.3 м/с
Уровень шума	≤50 дБА	≤55 дБА	≤65 дБА

Системы обеспечения комфорта:

— освещение:

- 1) комбинированное освещение (естественное + искусственное);
- 2) норма освещенности: 300-500 люкс;

- 3) коэффициент пульсации светильников $\leq 5\%$;
- 4) использование светодиодных панелей с цветовой температурой 4000К.

– вентиляция:

- 1) кратность воздухообмена: 3-4 объема/час;
- 2) установка очистителей воздуха с фильтрами;
- 3) автоматические увлажнители с поддержкой 50-60% влажности.

– шумоподавление:

- 1) звукопоглощающие панели на стенах и потолке;
- 2) акустические экраны между рабочими местами;
- 3) использование шумоподавляющих наушников при необходимости.

Помимо комфортных для труда условий, требуется соблюдение техники безопасности при работе с ПК. Для чего проведена классификация нагрузки (таблица 17) на работника, описаны меры безопасности и профилактика травм.

Таблица 16 – Классификация нагрузки разработчика

Группа	Вид деятельности	Режим работы	Перерывы
Б	Программирование	50 мин - работа	10 мин каждый час
В	Проектирование интерфейса	45 мин - работа	15 мин каждый час

Обязательные меры безопасности:

– перед началом работы:

- 1) проверка целостности кабелей и разъемов;
- 2) контроль заземления оборудования;
- 3) регулировка параметров монитора (яркость, контрастность).

– во время работы:

- 1) соблюдение «правила прямых углов» (колени, локти, спина);
- 2) использование эргономичной клавиатуры и вертикальной мыши;
- 3) применение программного обеспечения для контроля осанки (PostureReminder);
- 4) избегать и предотвращать аварийные ситуации (рисунок 10).

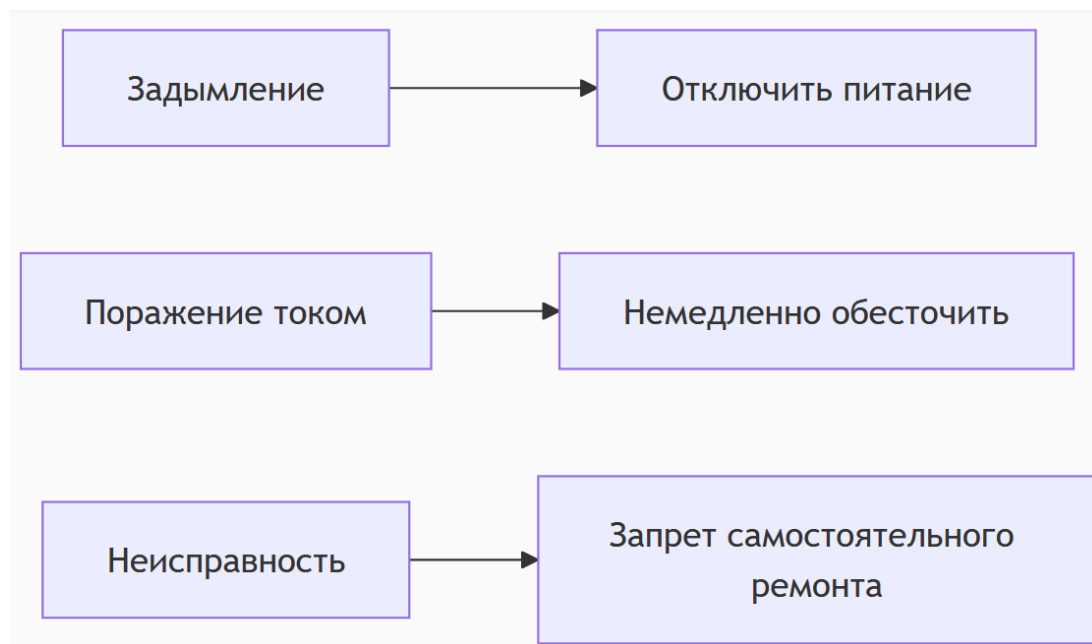


Рисунок 10 – Основные аварийные ситуации.

Профилактические мероприятия:

- ежегодный медосмотр с акцентом на зрение и опорно-двигательный аппарат;
- использование специальных очков с blue-light фильтром;
- ортопедические коврики для клавиатуры и мыши.

Пожарная безопасность:

- оборудование рабочих мест огнетушителями ОП-4;
- автоматическая система пожаротушения;
- план эвакуации с маршрутами к ближайшим выходам.

При разработке мобильного приложения были строго соблюдены все требования безопасности труда:

- рабочие места оборудованы в соответствии с ГОСТ Р 50923-96;
- микроклимат поддерживается в оптимальных параметрах;
- режим труда соответствует СанПиН 2.2.2.1332-03;
- оборудование прошло сертификацию по электробезопасности.

Заключение

В ходе выполнения выпускной квалификационной работы разработано мобильное приложение "Экспресс-оценка знаний" – инновационное решение для оперативного тестирования компетенций сотрудников организаций. Приложение реализовано как кроссплатформенное Android-решение с поддержкой офлайн-работы и синхронизацией данных в реальном времени.

Ключевые результаты работы:

- анализ предметной области. Изучены потребности организаций в инструментах оценки знаний. Определены функциональные требования для двух типов пользователей (сотрудники и администраторы). Разработаны сценарии для основных рабочих процессов;

- архитектурные решения. Реализована трехуровневая архитектура (UI - ViewModel - Repository). Внедрена локальная база данных SQLite. Обеспечена синхронизация данных через WebSocket протокол. Разработана система безопасности с ролевым доступом и шифрованием;

- технологический стек. Язык программирования: Kotlin (100% кодовой базы). Локальное хранилище: SQLite. Сетевое взаимодействие: WebSocket.

Выполнено юзабилити-тестирование с участием 10 пользователей. Подтверждена корректность работы в двух сценариях: одновременное тестирование 10 сотрудников и работа при отсутствии интернет-соединения.

Перспективы развития:

- внедрение адаптивных алгоритмов тестирования на основе ИИ;
- интеграция с корпоративными HR-системами;
- разработка модуля рекомендаций по обучению;
- добавление аналитики прогнозирования профессионального роста.

Разработанное приложение успешно решает задачу оперативной оценки знаний сотрудников, обеспечивая:

- сокращение временных затрат на проведение аттестаций;

- повышение объективности оценки компетенций;
- формирование персонифицированных траекторий развития персонала;
- упрощение процесса администрирования учебных программ.

Внедрение системы позволяет организациям перейти к модели непрерывного развития персонала, основанной на данных объективных измерений профессиональных компетенций.

					ПГУ 1.090501.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		58

Список использованных источников

1. GoogleDevelopers. AndroidJetpackGuide [Электронный ресурс] – Режим доступа:URL: <https://developer.android.com/jetpack/guide> (дата обращения: 15.05.2025).
2. RoomPersistenceLibraryDocumentation [Электронный ресурс] – Режим доступа:URL: <https://developer.android.com/training/data-storage/room> (дата обращения: 18.05.2025).
3. KotlinLanguageDocumentation [Электронный ресурс] – Режим доступа: URL: <https://kotlinlang.org/docs/home.html> (дата обращения: 20.05.2025).
4. WebSocket API: TheWebSocketProtocol [Электронный ресурс] – Режим доступа:URL: <https://datatracker.ietf.org/doc/html/rfc6455> (дата обращения: 22.05.2025).
5. Phillips, B. Android Programming: The Big Nerd Ranch Guide / B. Phillips, C. Stewart, K. Marsicano. - 5th ed. - Pearson, 2022. - 768 p.
6. Leiva, R. Android Studio 4.1 Development Essentials - Kotlin Edition / R. Leiva. - Payload Media, 2021. - 820 p.
7. SQLiteOfficialDocumentation [Электронный ресурс] – Режим доступа: URL: <https://www.sqlite.org/docs.html> (дата обращения: 28.05.2025).
8. MaterialDesign 3 Guidelines [Электронный ресурс] – Режим доступа: URL: <https://m3.material.io> (дата обращения: 01.06.2025).
9. Kaspersky, E. Secure Development Best Practices for Android Applications / E. Kaspersky // Journal of Mobile Security. - 2024. - Vol. 12, No. 3. - P. 45-62.

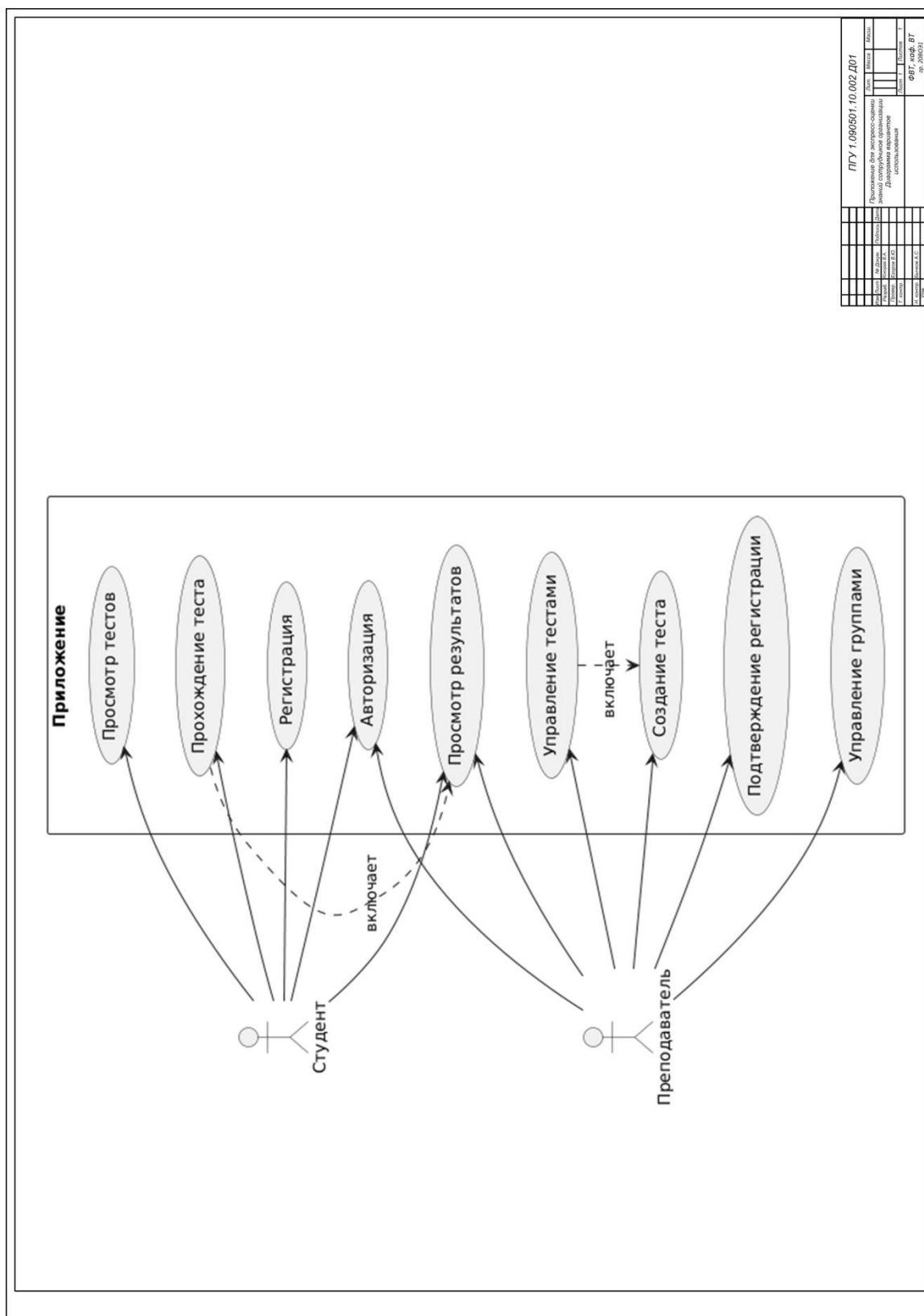
Приложение А

(обязательное)

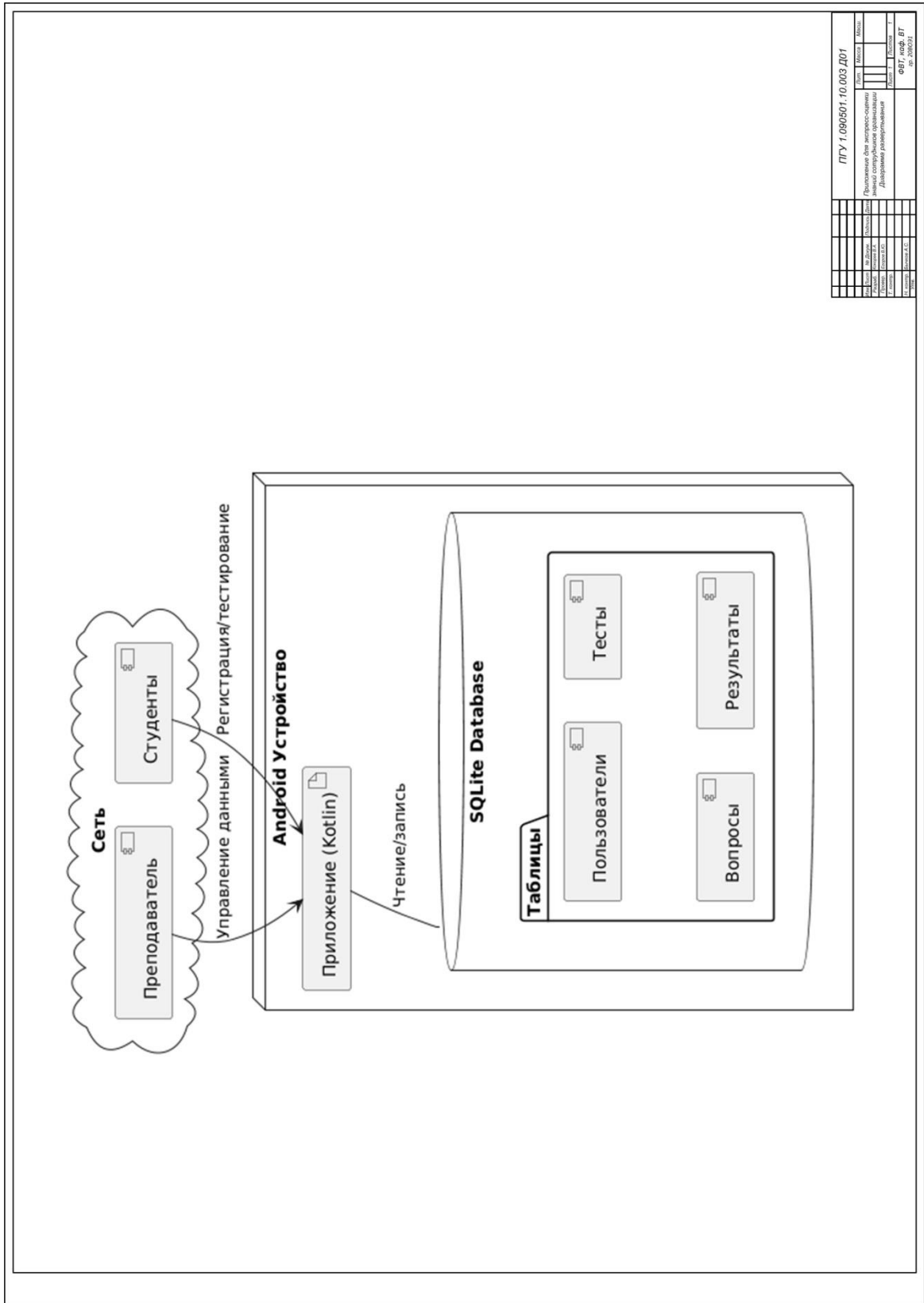
Приложение для экспресс-оценки знаний сотрудников организации
UML-диаграммы

					ПГУ 1.090501.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		60

Приложение А.1-Диаграмма последовательностей

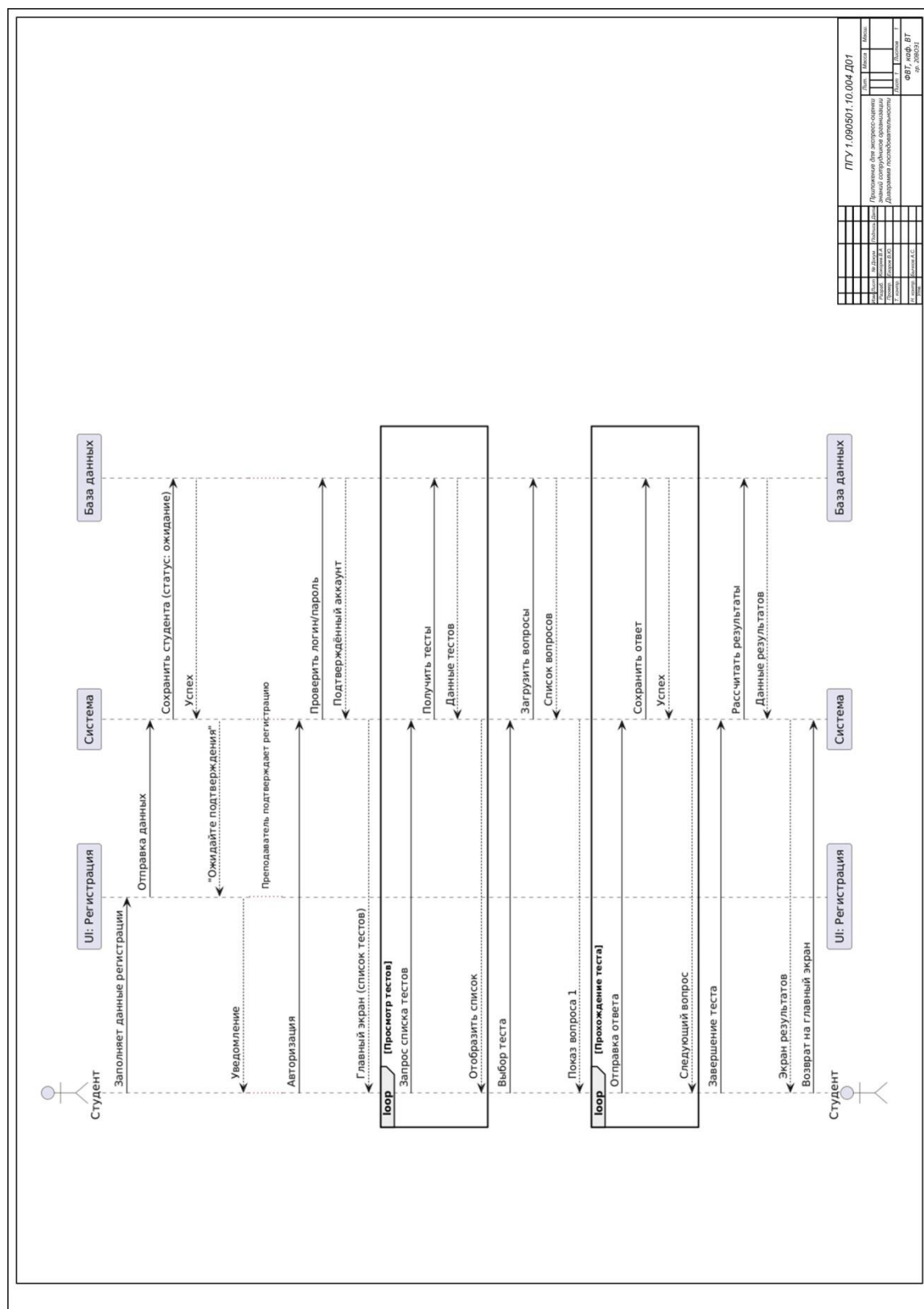


Приложение А.2-Диаграмма развертывания

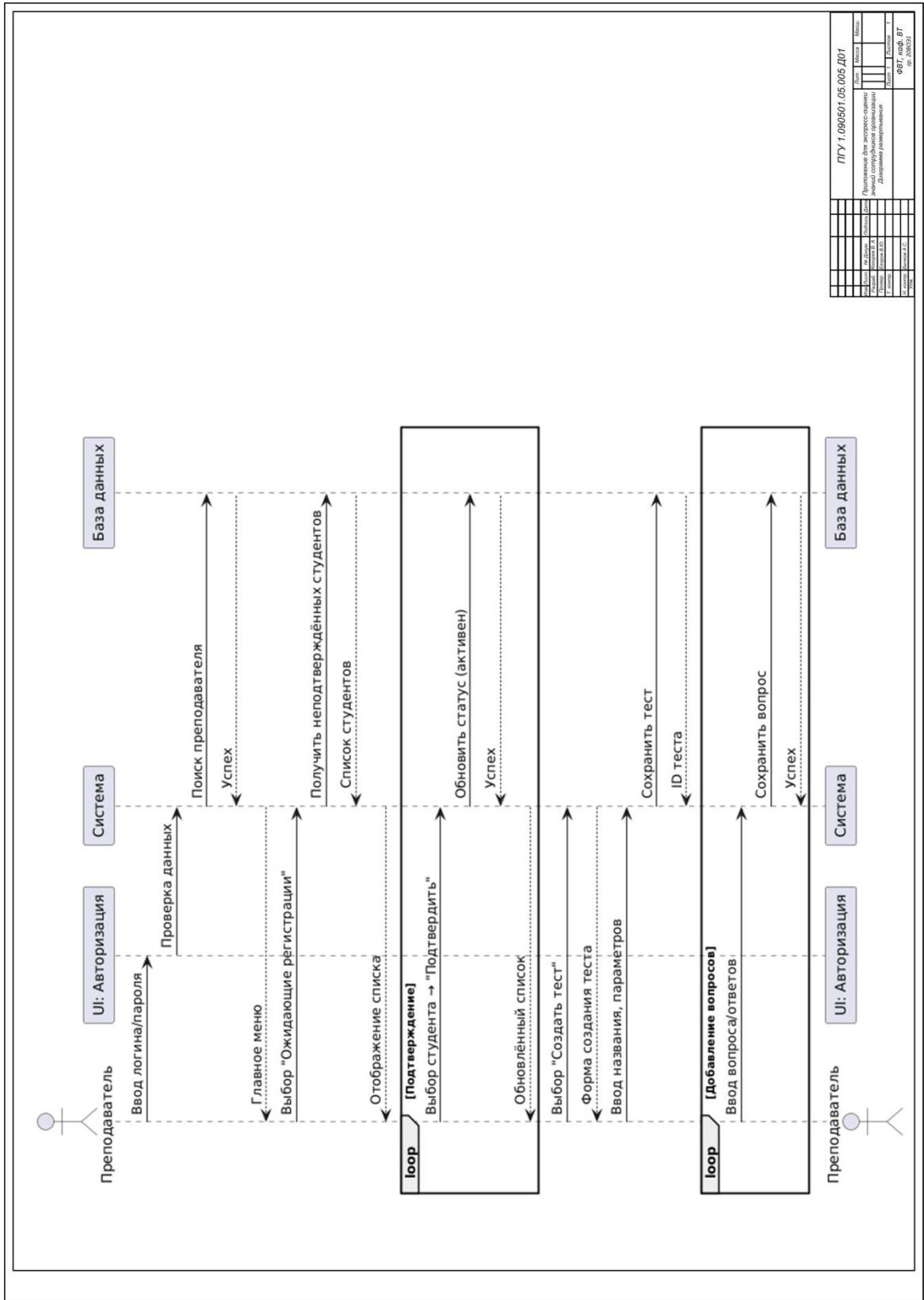


Изм.	Лист	№ докум.	Подпись	Дата

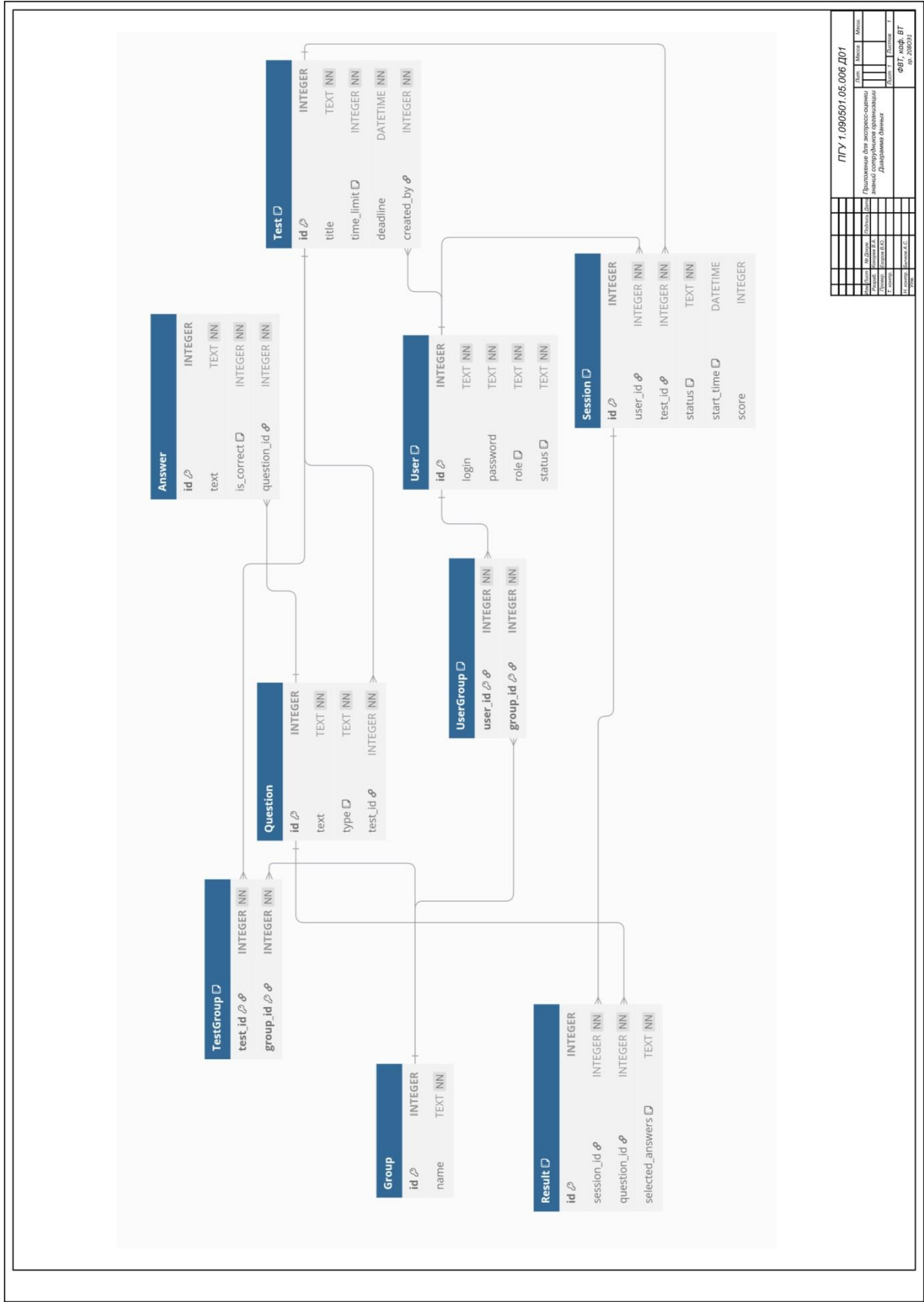
Приложение А.3-Диаграмма последовательности



Приложение А.4-Диаграмма последовательности



Приложение А.5-Диаграмма данных



ПГУ 1.090501.05.006 Д01									
Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель
Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель
Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель
Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель
Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель
Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель
Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель
Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель
Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель	Исполнитель

Приложение Б

(обязательное)

Приложение для экспресс-оценки знаний сотрудников организации

Результаты функционального тестирования

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



Рисунок Б.1 – Страница приветствия

 The registration form is displayed on a light gray background. At the top is the 'КВИЗ' logo. Below it are five input fields, each with a label and a placeholder text: 'Имя' (Введите Ваше Имя), 'Фамилия' (Введите Вашу Фамилию), 'Email' (Введите Вашу почту), 'Пароль' (Придумайте пароль), and 'Загрузить фото профиля' (with a camera icon). A blue button labeled 'Зарегистрироваться' is positioned below the fields. At the bottom, there is a link that says 'Уже есть аккаунт?'.

Рисунок Б.2 – Страница регистрации пользователя



Рисунок Б.3 – Страница ожидания подтверждения регистрации



Рисунок Б.4 – Страница одобрения и возможности входа


КВИЗ
 Платформа для тестирования

Имя Пользователя

Введите имя пользователя

Пароль

Введите пароль

Роль

Выберите роль

Войти

[Забыли пароль?](#)

Рисунок Б.5 – Страница авторизации


 Иванов Иван
 Студент

Выход

🔒 📅 ⭐

☐	Название: Python Срок сдачи: 01.07.25 Статус: Не начат. Таймер: Неограничено. Вопросы: 0/6	Начать тест
☒	Название: UML Срок сдачи: 09.06.25 Статус: Завершён. Таймер: 1час 20мин. Вопросы: 45/45	Результаты
☒	Название: ОС Linux Срок сдачи: 01.07.25 Статус: В процессе. Таймер: 30мин. Вопросы: 6/15	Продолжить
☐	Название: ОС Windows Срок сдачи: 06.07.25 Статус: Не начат. Таймер: 30 мин Вопросы: 0/6	Начать тест

Рисунок Б.6 – Главная страница студента

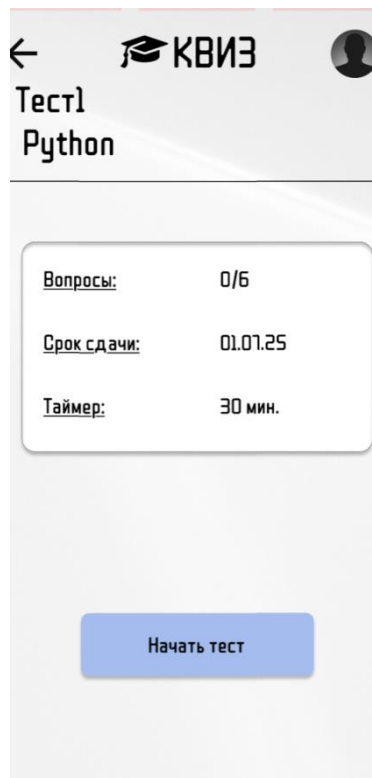


Рисунок Б.7 – Страница выбранного теста



Рисунок Б.8 – Страница вопроса

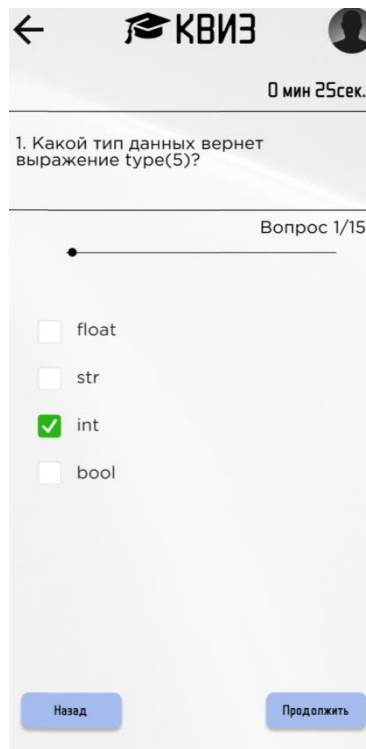


Рисунок Б.9 – Страница вопроса. Вопрос выбран.

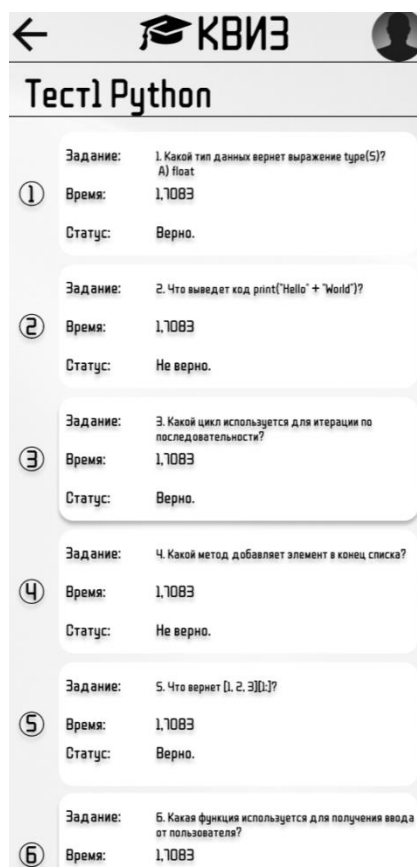


Рисунок Б.10 – Страница статистики

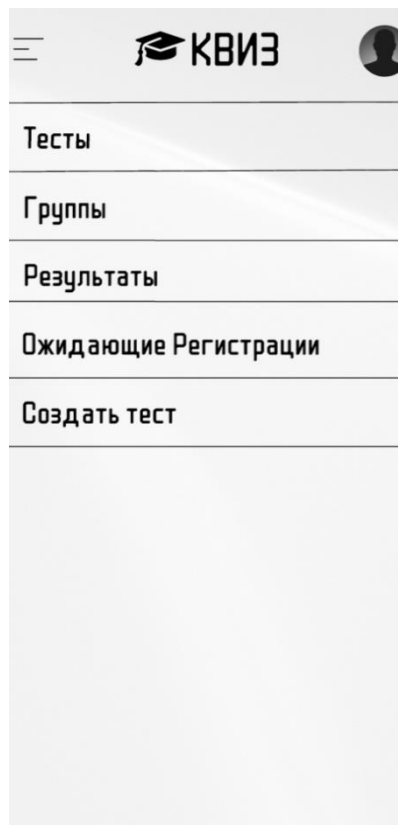


Рисунок Б.11 – Главная страница преподавателя

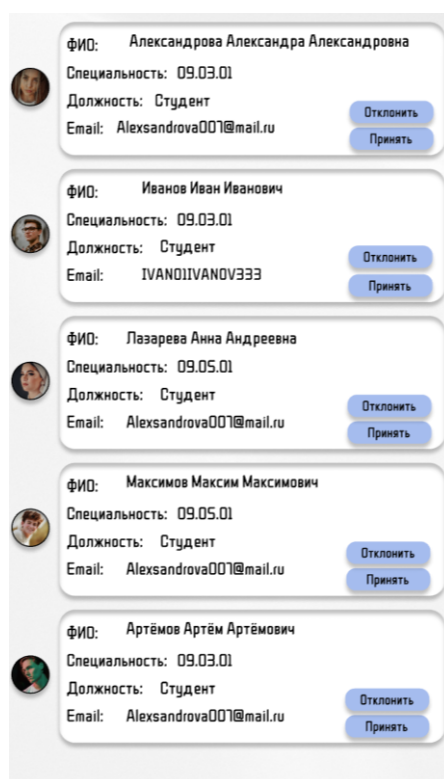


Рисунок Б.12 – Страница «Ожидание регистрации»

Рисунок Б.13 – Страница создания теста

Рисунок Б.14 – Создание вопроса теста

	Название:	Срок сдачи:	Статус:	Таймер:	Вопросы:	Редактировать	Удалить
☐	Python	01.07.25	Не начат.	Неограничено.	0/15	Редактировать	Удалить
☐	ОС Linux	01.07.25	Не начат.	Неограничено.	0/15	Редактировать	Удалить
☐	ОС Windows	01.07.25	Не начат.	Неограничено.	0/15	Редактировать	Удалить

[Создать тест](#)

Рисунок Б.15 – Управление тестами

Приложение В

(обязательное)

Приложение для экспресс-оценки знаний сотрудников организации
Листингпрограммы

					ПГУ 1.090501.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		75

Файлsign_up.kt

```
package com.example.quizoo
```

```
import android.content.Intent
```

```
import androidx.appcompat.app.AppCompatActivity
```

```
import android.os.Bundle
```

```
import android.widget.Toast
```

```
import com.google.firebase.auth.FirebaseAuth
```

```
import kotlinx.android.synthetic.main.activity_sign_up.*
```

```
class sign_up : AppCompatActivity() {
```

```
    var mAuth: FirebaseAuth? = null
```

```
    override fun onCreate(savedInstanceState: Bundle?) {
```

```
        super.onCreate(savedInstanceState)
```

```
        supportActionBar?.hide()
```

```
        setContentView(R.layout.activity_sign_up)
```

```
        mAuth = FirebaseAuth.getInstance()
```

```
        textView4.setOnClickListener(){
```

```
            val gotLogin = Intent(this, sign_in::class.java)
```

```
            startActivity(gotLogin)
```

```
        }
```

```
        button1.setOnClickListener(){
```

```
            val email = editTextEmailAddress.text.toString()
```

```
            val pswd = editTextTextPassword.text.toString()
```

```
            val confirm_pswd = editTextTextPassword2.text.toString()
```

```
            if (email.isEmpty() || pswd.isEmpty() || confirm_pswd.isEmpty()){
```



```

    }
}

```

Файл sign_in.kt

```

package com.example.quizoo

import android.content.Intent
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.widget.Toast
import com.google.firebase.auth.FirebaseAuth
import kotlinx.android.synthetic.main.activity_sign_in.button1
import kotlinx.android.synthetic.main.activity_sign_in.editTextTextPassword
import kotlinx.android.synthetic.main.activity_sign_in.editTextTextPassword2

class sign_in : AppCompatActivity() {
    var mAuth: FirebaseAuth? = null

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        supportActionBar?.hide()
        setContentView(R.layout.activity_sign_in)

        mAuth = FirebaseAuth.getInstance()

        button1.setOnClickListener {
            val email1 = editTextTextPassword.text.toString()
            val pswd1 = editTextTextPassword2.text.toString()

            if (email1.isEmpty() || pswd1.isEmpty()) {

```

					ПГУ 1.090501.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		78


```

valoption_two:String,
valoption_three:String,
valoption_four:String,
valright_answer:Int,
)

```

Файлquestion_design.kt

```

packagecom.example.quizoo

```

```

importandroid.content.Intent
importandroid.graphics.Color
importandroid.net.Uri.parse
importandroidx.appcompat.app.AppCompatActivity
importandroid.os.Bundle
importandroid.view.View
importandroid.widget.SearchView
importandroid.widget.TextView
importandroid.widget.Toast
importandroidx.core.content.ContextCompat
importkotlinx.android.synthetic.main.activity_question_design.*

```

```

classquestion_design : AppCompatActivity(), View.OnClickListener {
var submit=false

```

```

privatevarmcurrentpos=1
privatevarmquestionlist:ArrayList<questions>?=null
privatevarmselectedchoicepos:Int=0
privatevarmcorrectanswer:Int=0
override fun onCreate(savedInstanceState: Bundle?) {
super.onCreate(savedInstanceState)

```

```
setContentView(R.layout.activity_question_design)
```

```
mquestionlist = constants.getquestion()
```

```
setquesiton()
```

```
ndchoice.setOnClickListener(this)
```

```
rdchoice.setOnClickListener(this)
```

```
thchoice.setOnClickListener(this)
```

```
stchoice.setOnClickListener(this)
```

```
submitbtn.setOnClickListener(this)
```

```
nextbtn.setOnClickListener(){
```

```
    this.mcurrentpos += 1
```

```
    this.setquesiton()
```

```
    submit=false
```

```
    }
```

```
}
```

```
funsetquesiton(){
```

```
    defaultoptionview()
```

```
    if (mcurrentpos==mquestionlist!!.size){
```

```
        nextbtn.text="finish"
```

```
    }
```

```
    if (mcurrentpos>mquestionlist!!.size){
```

```
        valgotoresult=Intent(this,quizz_score::class.java)
```

```
        gotoresult.putExtra(constants.totalquestion, mquestionlist!!.size)
```

```
        gotoresult.putExtra(constants.correctanswer,mcorrectanswer)
```

```
        startActivity(gotoresult)
```

```
    }
```

```

if (submit==false){
    nextbtn.isClickable=false
    }

    valquestionn = mquestionlist!![mcurrentpos-1]
    progress_bar.progress=mcurrentpos
    tv_progess.text="$mcurrentpos"+"/".$progress_bar.max
    tv_quesiton.text=questionn.question
    imagesection.setImageResource(questionn.image)
    tv_section.text=questionn.section

    stchoice.text=questionn.option_one
    ndchoice.text=questionn.option_two
    rdchoice.text=questionn.option_three
    thchoice.text=questionn.option_four
    }

    private fun defaultoptionview(){
    var options=ArrayList<TextView>()
    options.add(0,stchoice)
    options.add(1,ndchoice)
    options.add(2,rdchoice)
    options.add(3,thchoice)

    for(option in options){

    option.background=ContextCompat.getDrawable(this,R.drawable.loginbutton)
    }
    }

    private fun selectedchoiceview(tv:TextView,choicenbr:Int){
    if (submit==false){

```

					ПГУ 1.090501.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		82

defaultoptionview()

tv.background=ContextCompat.getDrawable(this,R.drawable.clicked_button)

mselectedchoicepos=choicenbr

}

}

override fun onClick(p0: View?) {

when (p0?.id){

R.id.stchoice->{

selectedchoiceview(stchoice,1)

}

R.id.ndchoice->{

selectedchoiceview(ndchoice,2)

}

R.id.rdchoice->{

selectedchoiceview(rdchoice,3)

}

R.id.thchoice->{

selectedchoiceview(thchoice,4)

}

R.id.submitbtn->{

submit=true

varqstion= mquestionlist?.get(mcurrentpos -1)

```

if (qstion?.right_answer != mselectedchoicepos){
    answer(mselectedchoicepos,R.drawable.wrong_answer)
}else{
    mcorrectanswer++
        }
    answer(qstion!!.right_answer,R.drawable.correct_answer)

mselectedchoicepos=0

        }
    }
}

private fun answer(answer:Int,drawableview:Int){
    when(answer){
        1->{
            stchoice.background=ContextCompat.getDrawable(this,drawableview)
                }
            2->{
                ndchoice.background=ContextCompat.getDrawable(this,drawableview)
                    }
                3->{
                    rdchoice.background=ContextCompat.getDrawable(this,drawableview)
                        }
                        4->{
                            thchoice.background=ContextCompat.getDrawable(this,drawableview)
                                }
                                }
                    }
                }
            }
        }
    }

```

```
}
```

Файлquestion_design.kt

```
packagecom.example.quizoo
```

```
importandroid.content.Intent
```

```
importandroidx.appcompat.app.AppCompatActivity
```

```
importandroid.os.Bundle
```

```
import kotlinx.android.synthetic.main.activity_quizz_score.*
```

```
classquizz_score : AppCompatActivity() {
```

```
    override fun onCreate(savedInstanceState: Bundle?) {
```

```
        super.onCreate(savedInstanceState)
```

```
        setContentView(R.layout.activity_quizz_score)
```

```
        supportActionBar?.hide()
```

```
        vartotalqust=intent.getIntExtra(constants.totalquestion,10)
```

```
        varrightanswers=intent.getIntExtra(constants.correctanswer,5)
```

```
        tvscore.text="your score is $rightanswers out of $totalqust "
```

```
        btnfinish.setOnClickListener(){
```

```
            vargocateg= Intent(this,categoy::class.java)
```

```
            startActivity(gocateg)
```

```
        }
```

```
    }
```

```
}
```

ФайлQuizQuestionsActivity.kt

```
packagecom.example.quizapp
```

```
importandroid.content.Intent
```

```
importandroid.graphics.Color
```

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

```

import android.graphics.Typeface
import android.graphics.drawable.Drawable
import androidx.appcompat.app.AppCompatActivity
import android.os.Bundle
import android.util.Log
import android.view.View
import android.widget.*
import androidx.core.content.ContextCompat
import kotlin.reflect.TypeOf

```

```

class QuizQuestionsActivity : AppCompatActivity() {
    private var userName: String? = null

```

```

    private val questionsList: ArrayList<Question> = Constants.getQuestions()
    private var currentIndex = 0;
    private var selectedAlternativeIndex = -1;
    private var isAnswerChecked = false;
    private var totalScore = 0;
    private val alternativesIds = arrayOf(R.id.optionOne, R.id.optionTwo,
R.id.optionThree, R.id.optionFour)

```

```

    private var tvQuestion: TextView? = null
    private var ivImage: ImageView? = null
    private var progressBar: ProgressBar? = null
    private var tvProgress: TextView? = null
    private var btnSubmit: Button? = null
    private var tvAlternatives: ArrayList<TextView>? = null

```

```

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)

```

```
setContentView(R.layout.activity_quiz_questions)
```

```
userName = intent.getStringExtra(Constants.USER_NAME)
```

```
tvQuestion = findViewById(R.id.tvQuestion)
```

```
ivImage = findViewById(R.id.ivImage)
```

```
progressBar = findViewById(R.id.progressBar)
```

```
tvProgress = findViewById(R.id.tvProgress)
```

```
btnSubmit = findViewById(R.id.btnSubmit)
```

```
tvAlternatives = arrayListOf(  
    findViewById(R.id.optionOne),  
    findViewById(R.id.optionTwo),  
    findViewById(R.id.optionThree),  
    findViewById(R.id.optionFour),  
    )
```

```
updateQuestion()
```

```
btnSubmit?.setOnClickListener {
```

```
    if (!isAnswerChecked) {
```

```
        val anyAnswerIsChecked = selectedAlternativeIndex != -1
```

```
        if (!anyAnswerIsChecked) {
```

```
            Toast.makeText(this, "Please, select an alternative",  
                Toast.LENGTH_SHORT).show()
```

```
        } else {
```

```
            val currentQuestion = questionsList[currentQuestionIndex]
```

```
            if (
```

```
                selectedAlternativeIndex == currentQuestion.correctAnswerIndex
```

```
            ) {
```

```

        answerView(tvAlternatives!![selectedAlternativeIndex],
R.drawable.correct_option_border_bg)
        totalScore++
    } else {
        answerView(tvAlternatives!![selectedAlternativeIndex],
R.drawable.wrong_option_border_bg)
        answerView(tvAlternatives!![currentQuestion.correctAnswerIndex],
R.drawable.correct_option_border_bg)
    }

    isAnswerChecked = true
    btnSubmit?.text = if (currentQuestionIndex == questionsList.size - 1)
"FINISH" else "GO TO NEXT QUESTION"
    selectedAlternativeIndex = -1
    }
    } else {
    if (currentQuestionIndex < questionsList.size - 1) {
    currentQuestionIndex++
    updateQuestion()
    } else {
    val intent = Intent(this, ResultActivity::class.java)
    intent.putExtra(Constants.USER_NAME, userName)
    intent.putExtra(Constants.TOTAL_QUESTIONS, questionsList.size)
    intent.putExtra(Constants.SCORE, totalScore)
    startActivity(intent)
    finish()
    }

    isAnswerChecked = false
    }

```

```
tvAlternatives?.let {
    for (optionIndex in it.indices) {
        it[optionIndex].let {
            it.setOnClickListener{
                if (!isAnswerChecked) {
                    selectedAlternativeView(it as TextView, optionIndex)
                }
            }
        }
    }
}
```

```
// Render Question Text
tvQuestion?.text = questionsList[currentQuestionIndex].questionText

// Render Question Image
ivImage?.setImageResource(questionsList[currentQuestionIndex].image)

// progressBar
progressBar?.progress = currentQuestionIndex + 1

// Text of progress bar
tvProgress?.text = "${currentQuestionIndex + 1}/${questionsList.size}"
```

in

```

        tvAlternatives!![alternativeIndex].text =
questionsList[currentQuestionIndex].alternatives[alternativeIndex]
    }

```

```

        btnSubmit?.text = if (currentQuestionIndex == questionsList.size - 1)
"FINISH" else "SUBMIT"
    }

```

```

private fun defaultAlternativesView() {
for (alternativeTv in tvAlternatives!!) {
alternativeTv.typeface = Typeface.DEFAULT
alternativeTv.setTextColor(Color.parseColor("#7A8089"))
alternativeTv.background = ContextCompat.getDrawable(
this@QuizQuestionsActivity,
R.drawable.default_option_border_bg
    )
}
}

```

```

private fun selectedAlternativeView(option: TextView, index: Int) {
defaultAlternativesView()
selectedAlternativeIndex = index

```

```

option.setTextColor(
Color.parseColor("#363A43")
    )
option.setTypeface(option.typeface, Typeface.BOLD)
option.background = ContextCompat.getDrawable(
this@QuizQuestionsActivity,
R.drawable.selected_option_border_bg

```

```

        )
    }

    private fun answerView(view: TextView, drawableId: Int) {
        view.background = ContextCompat.getDrawable(
            this@QuizQuestionsActivity,
            drawableId
        )
        tvAlternatives!![selectedAlternativeIndex].setTextColor(
            Color.parseColor("#FFFFFF")
        )
    }
}

```

Приложение Г

(обязательное)

Приложение для экспресс-оценки знаний сотрудников организации

Презентация

					ПГУ 1.090501.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		92

Приложение для экспресс-оценки знаний сотрудников организации

Автор работы:

студент гр. 20ВОЭ1 Кокорев Владислав Алексеевич

Научный руководитель:

к.т.н., доцент Егоров Валерий Юрьевич

Пенза, 2025

Цель работы и объекты

Цель работы – Разработать бесплатное мобильное приложение для экспресс тестирования студентов и сотрудников предприятия.

Приложение должно выполнять следующие функции:

- Формирование базы тестов.
 - Сохранение информации о тестах и оценках в локальной СУБД.
- Приложение должно иметь возможность работы мультипользовательском режиме.

Программное обеспечение реализуется с помощью среды разработки Android Studio.

В качестве СУБД используется компактная встраиваемая СУБД SQLite.

					ПГУ 1.090501.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		93

Задачи приложения

Приложение реализовано в виде мобильного клиента способного в режиме офлайн сформировать базу тестов и затем дать возможность студентам пройти тестирование с последующей беспристрастной оценкой учащихся или тестируемых.

Приложение работает в двух режимах: пользователь (студент), преподаватель.

Полученная информация хранится локально на мобильном устройстве в СУБД SQLite.

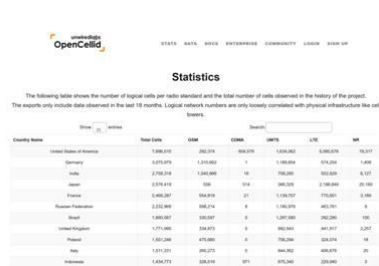
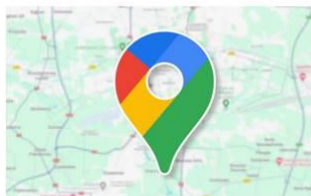
Варианты использования



Сравнение аналогов

<u>Skillspace</u>	Простота создания тестов; широкий выбор типов вопросов; таймеры и автопроверка; мобильная версия	Нет гибкой системы ролей (ограниченные права преподавателей и студентов); ограничение функций в бесплатной версии
<u>Google Forms</u>	Бесплатно; простое создание форм и тестов; массовая рассылка; экспорт результатов в таблицы	Нет полноценной авторизации и ролевой модели; отсутствует система рейтингов; минимальные возможности по аналитике
<u>Quizizz</u>	Интерактивный, геймифицированный интерфейс; быстрый запуск тестов; мобильные приложения; рейтинг игроков	Фокус на образовании, а не на корпоративном тестировании; ограниченное управление доступом и ролями
<u>SunRay WEB Class</u>	Гибкая настройка тестов; поддержка больших групп; подробная статистика	Сложный интерфейс; устаревший дизайн; не всегда интуитивен для мобильных пользователей

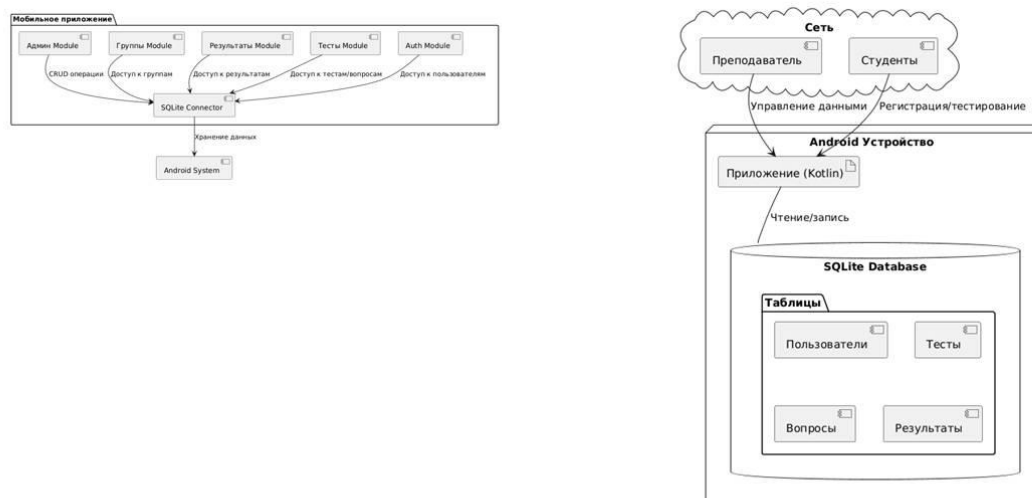
Стек технологий



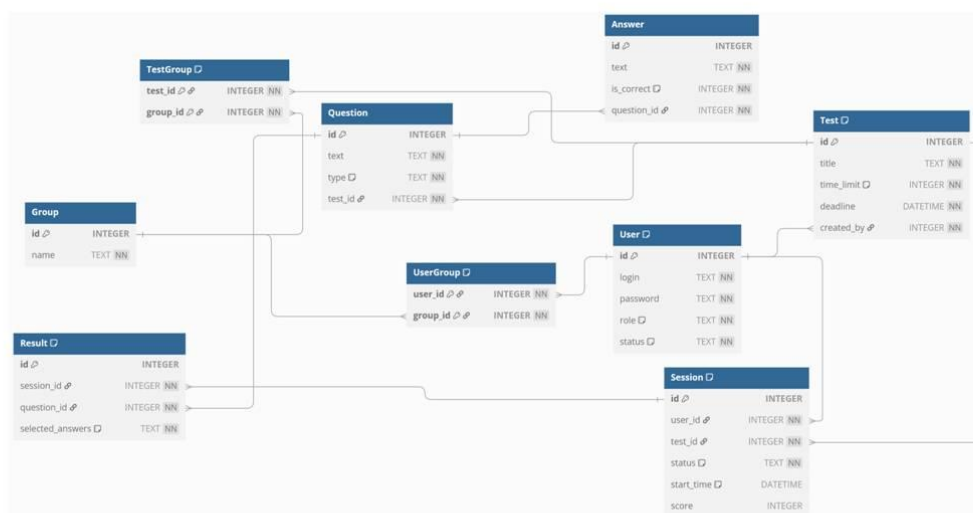
Android Monitor

Kotlin

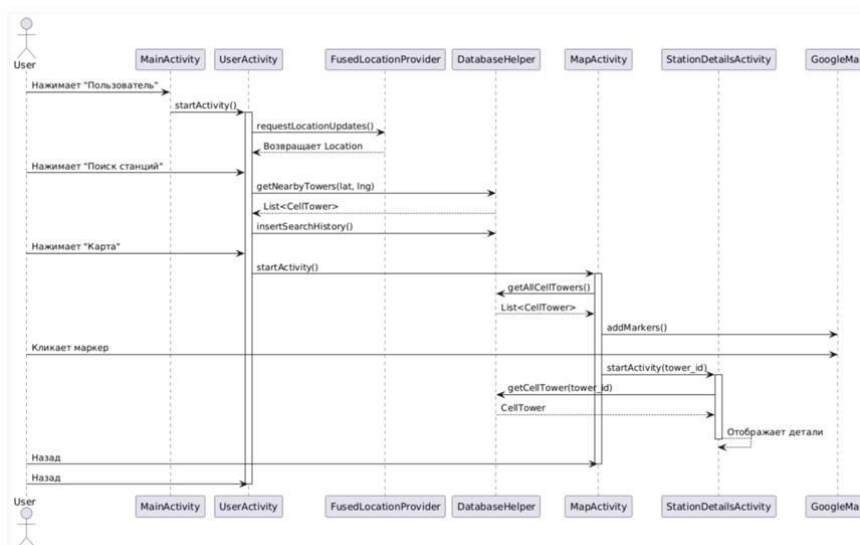
Архитектура приложения



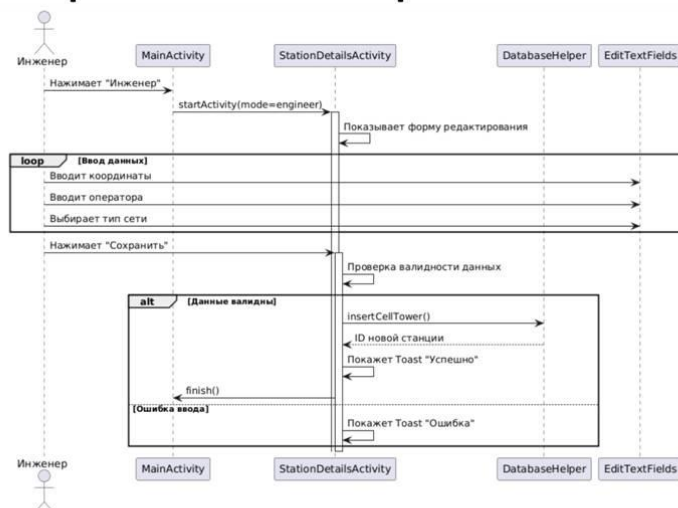
База данных приложения



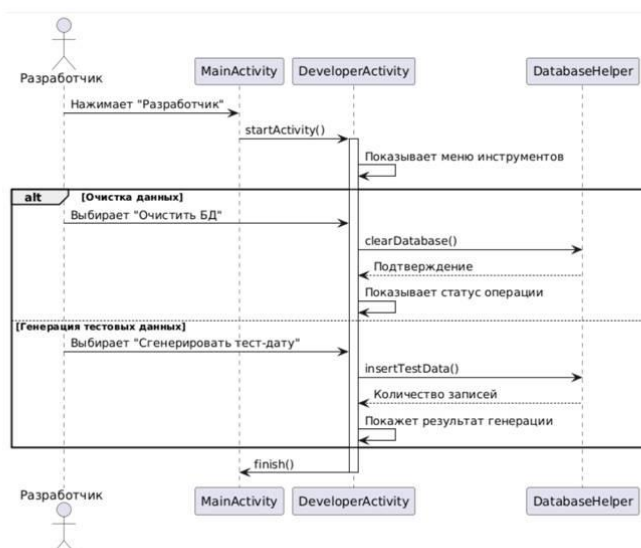
Работа приложения в режиме «Пользователь»



Работа приложения в режиме «Инженер»



Работа приложения в режиме «Разработчик»



Структура базы данных

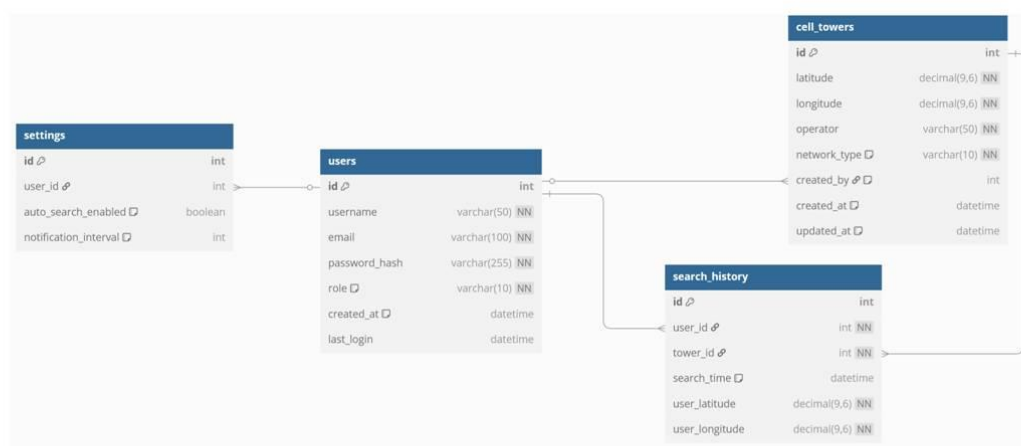
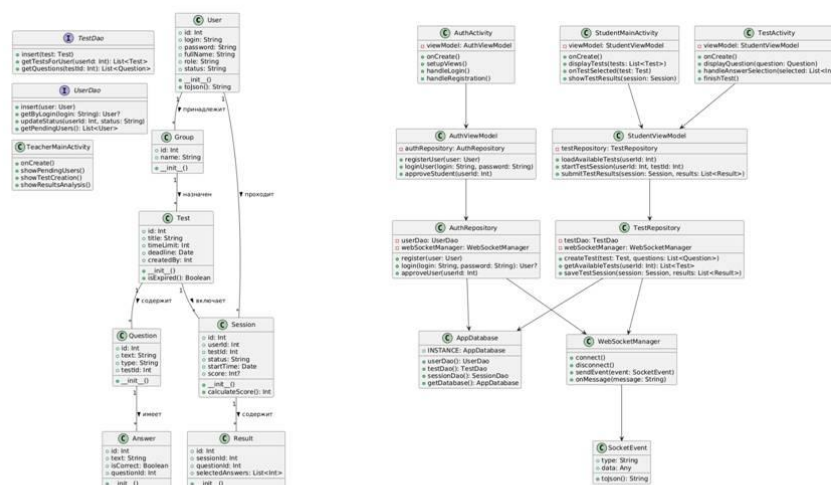
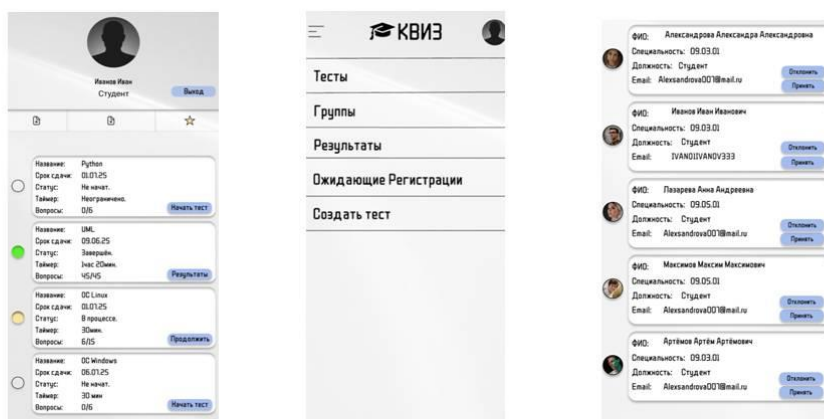


Диаграмма классов



Скриншоты и демонстрация



Результаты работы

- Разработано мобильное приложение полностью соответствующее техническому заданию;
- Проведено функциональное тестирование (успешно);
- Намечены задачи по развитию функциональных возможностей программы.

Спасибо за внимание!

					ПГУ 1.090501.10.001 ПЗ	Лист
Изм.	Лист	№ докум.	Подпись	Дата		100