СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc168351207)

[1. ОБЩАЯ ЧАСТЬ 4](#_Toc168351208)

[1.1 Цель разработки 4](#_Toc168351209)

[1.2 Средства разработки 4](#_Toc168351210)

[2. СПЕЦИАЛЬНАЯ ЧАСТЬ 7](#_Toc168351211)

[2.1. Постановка задачи 7](#_Toc168351212)

[2.1.1. Входные данные 7](#_Toc168351213)

[2.1.2. Выходные данные 7](#_Toc168351214)

[2.1.3. Подробные требования к проекту 7](#_Toc168351215)

[2.2. Внешняя спецификация 8](#_Toc168351216)

[2.2.1. Описание задачи 8](#_Toc168351217)

[2.2.2. Входные и выходные данные 10](#_Toc168351218)

[2.2.3. Методы 13](#_Toc168351219)

[2.2.4. Тесты 14](#_Toc168351220)

[2.2.5. Контроль целостности данных 16](#_Toc168351221)

[2.3. Проектирование 19](#_Toc168351222)

[2.3.1. Схема архитектуры приложения 19](#_Toc168351223)

[2.3.2. Логическая схема данных 20](#_Toc168351224)

[2.3.3. Физическая схема данных 21](#_Toc168351225)

[2.3.4. Структурная схема 25](#_Toc168351226)

[2.3.5. Функциональная схема 27](#_Toc168351227)

[2.3.6. Диаграмма классов 28](#_Toc168351228)

[2.3.7. Схема тестирования 32](#_Toc168351229)

[2.3.8. Схема пользовательского интерфейса 33](#_Toc168351230)

[2.4 Результат работы программы 33](#_Toc168351231)

[3. ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ 38](#_Toc168351232)

[3.1. Инструментальные средства 38](#_Toc168351233)

[3.2 Отладка программы 39](#_Toc168351234)

[3.3. Защитное программирование 39](#_Toc168351235)

[3.4. Характеристики программы 39](#_Toc168351236)

[ЗАКЛЮЧЕНИЕ 40](#_Toc168351237)

[СПИСОК ИСПОЛЬЗОВАННЫХ МАТЕРИАЛОВ 42](#_Toc168351238)

ПРИЛОЖЕНИЕ А. Техническое задание

ПРИЛОЖЕНИЕ Б. Эскизный проект

ПРИЛОЖЕНИЕ В. Технический проект

ПРИЛОЖЕНИЕ Г. Макет приложения

ПРИЛОЖЕНИЕ Д. Сценарий тестов

ПРИЛОЖЕНИЕ Е. Скрипт БД

ПРИЛОЖЕНИЕ Е. Код программы

ПРИЛОЖЕНИЕ Ж. Результат тестовых испытаний

ПРИЛОЖЕНИЕ З. Руководствопользователя

# ВВЕДЕНИЕ

В рамках данной курсовой работы необходимо разработать приложение «LibraryStar» на базе Android при помощи инструмента Android Studio и языка программирования Java. Разработанная программа для управления личной библиотекой предназначена для хранения информации о книгах в личной библиотеке, их текстов и описаний. Основные функциональные возможности включают в себя добавление и изменение книг, их чтение, регистрация пользователей и формирования личной библиотеки пользователя на основе предоставленных книг. Эта программа помогает пользователям читать книги, управлять собственной библиотекой и повышать навык литературной грамотности.

# ОБЩАЯ ЧАСТЬ

## 1.1 Цель разработки

Основной целью разработки является создание эффективной системы учета и формирования личной библиотеки в целях экономии временных и материальных ресурсов пользователей.

## 1.2 Средства разработки

Для проектирования, разработки и тестирования мобильного приложения, реализующего свой функционал на базе платформы Android, были использованы программные средства, представленные в Таблице 1.

Таблица 1 – Программные средства разработки ПО

|  |  |  |  |
| --- | --- | --- | --- |
| № | Тип средства | Название средства | Назначение |
| 1 | 2 | 3 | 4 |
| 1 | Среда разработки | Android Studio Iguana | Среда запуска |
| 2 | Операционная система | Windows 10 Home | Организация взаимодействия  программ и пользователя |
| 3 | Текстовый редактор | Microsoft Word 2016 | Разработка документации,  формирование отчетных документов по шаблонам. |
| 4 | Графическая оболочка | PyCharm Professional | Разработка базы данных и функционала работы с ней. |
| 5 | База данных | sqlite | Хранение данных. |

В таблице 2 представлены минимальные и рекомендованные технические средства, на базе которых возможно комфортное использование реализуемого программного обеспечения.

Таблица 2 – Минимальные и рекомендованные технические средства

|  |  |  |
| --- | --- | --- |
| № | Тип оборудования | Наименование оборудования |
| 1 | 2 | 3 |
| Максимальные или рекомендуемые требования | | |
| 1 | Версия ОС: | 9.0+ |
| 2 | Процессор: | Тактовая частота процессора – 1600 МГц, 4 ядра. |
| 3 | Видеопамять [VRAM]: | 64 Mb |
| 4 | Оперативная память [RAM]: | 1024 Mb |
| Минимальные требования | | |
| 1 | Версия ОС: | 9.0. |
| 2 | Процессор: | Тактовая частота процессора – 400 МГц. |
| 3 | Видеопамять [VRAM]: | 32 Mb |
| 4 | Оперативная память [RAM]: | 256 Mb |

В качестве средств вычислительной техники при разработке ПО использовался ноутбук HP Laptop 14s-fq0111ur, для использования приложения использовалось мобильное устройство Samsung A32. Характеристики представлены в Таблице 3.

Таблица 3 – Технические характеристики ноутбука и мобильного устройства при разработке ПО

|  |  |  |
| --- | --- | --- |
| № | Тип оборудования | Наименование оборудования |
| 1 | 2 | 3 |
| Для разработки | | |
| HP Laptop 14s-fq0111ur | | |
| 1 | Процессор: | AMD Ryzen 3 3250U |
| 2 | Оперативная память: | 16 GB |
| 4 | Тип видеокарты: | Встроенная |
| 5 | Видеокарта: | AMD Radeon Graphics |
| 6 | Конфигурация накопителей: | SSD |
| 7 | Общий объем накопителей: | 256 ГБ |
| 8 | Компьютерная мышь: | Нет |
| 9 | Клавиатура: | Любая рабочая клавиатура с английской и русской раскладкой |
| 10 | Операционная система | Windows 10 Home |
| Для использования | | |
| Samsung A32 | | |
| 1 | Дисплей: | Super AMOLED, 6.4” (2400x1080) |
| 2 | Встроенная память: | 128 |
| 3 | Оперативная память: | 4 |
| 4 | Процессор | MediaTek Helio G80 |
| 5 | Количество ядер процессора | 8 |

# СПЕЦИАЛЬНАЯ ЧАСТЬ

### 2.1. Постановка задачи

Программа «LibraryStar» представляет из себя информационную систему на базе Android Studio для устройств на Android.

### 2.1.1. Входные данные

* Аутентификационные данные пользователя (Пароль, логин);
* Регистрационные данные пользователя (Пароль, пароль, логин);
* Создать книгу (Название книги, год создания книги, изображение книги, описание книги, файл книги, авторы книги, издатель книги, жанр книги)
* Создать автора (ФИО автора, год рождения автора, год смерти автора (опционально), изображение автора)

### 2.1.2. Выходные данные

Выходные данные мобильного приложения представлены в следующем виде:

* Книги (Название, авторы книги, описание книги, файл книги, изображение книги, год создания книги, жанр книги, издатель книги);
* Жанры (Название, первый цвет жанра, второй цвет жанра, изображение жанра);
* Данные пользователя (Аватарка пользователя, имя пользователя, дата регистрации пользователя);
* Авторы (ФИО, год рождения, год смерти(опционально), изображение автора);

### 2.1.3. Подробные требования к проекту

Подробные требования к разработке мобильного приложения «LibraryStar» представлены в Приложении А. Техническое задание.

## 2.2. Внешняя спецификация

### 2.2.1. Описание задачи

* Программа представляет из себя информационную систему, работающую с API, у которой есть БД с некоторыми таблицами:
* Авторы: Таблица авторов книг;
* Авторы книг: Таблица связей между авторами и книгами;
* Книги: Таблица книг с информацией, такой как название, количество страниц и издатель;
* Жанры: Таблица жанров книг;
* Книги пользователей: Таблица связей между пользователями и книгами;
* Пользователи: Таблица пользователей с информацией, такой как логин, пароль и роль;
* Роли: Таблица ролей пользователей.
* Связи БД: Авторы-Книги (Многие ко многим через промежуточную таблицу) Жанры-Книги (Один ко многим) Пользователи-Книги (Многие ко многим через промежуточную таблицу).
* Предусмотрено 3 роли (пользователь, издатель, администратор). Пользователь может читать книги, просматривать жанры, авторов, добавлять книги к себе в библиотеку. Права и возможности ролей отличаются тем, что все роли имеют функционал пользователя, и лишь добавляют дополнительный функционал. Например, издатель, в отличии от пользователя, может указываться как издатель определенной книги. А администратор же в свою очередь может добавлять новые книги и авторов в базу данных.
* Основная схема связей БД должна выглядеть следующим образом: есть книги, у которых есть жанр и авторы. Есть пользователи, у которых есть любимые книги и соответствующая роль.
* Система должна предоставлять возможность добавлять новые книги и добавлять их к себе в библиотеку.
* Система должна предоставлять возможность читать книги, управляя жестом перелистывания.
* Для системы предусмотрена авторизация и регистрация новых пользователей.
* Вход в приложение будет начинаться с ввода логина и пароля от аккаунта. Требования для логина: не короче 6 символов. Требования для пароля: не короче 8 символов, хотя бы один спецсимвол, хотя бы одна цифра, без пробелов, только латинские буквы.
* Многопользовательская поддержка: данные о пользователях и их ролях хранятся в БД.
* Книги должны быть реализованы в виде файла форматом pdf.
* Для интерфейса программы должен быть использован стиль библиотеки Material Design Android.

### 2.2.2. Входные и выходные данные

В таблице 4 представлены входные данные.

Таблица 4 – Входные данные

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Имя | Ограничения | Обязательное | Структура, формат | Форма ввода |
| 1 | 2 | 3 | 4 | 5 |
| Входные данные | | | | |
| Логин | Максимальная длина строки 255 символов, не менее 6 символов | Да | Строка | Текстовое поле |
| Пароль | Максимальная длина строки 255 символов, не менее 8 символов, только латинские буквы, по крайней мере одна цифра, по крайней мере один спецсимвол | Да | Строка | Текстовое поле |
| Фотография книги | Формат медиа-файла | Да | Файл | Проводник |
| Название книги | Максимальная длина строки 100 символов | Да | Строка | Текстовое поле |
| Издатель книги | Выбор значения из выпадающего списка | Да | Строка | Выпадающий список |
| Жанр книги | Выбор значения из выпадающего списка | Да | Строка | Выпадающий список |
| Автор книги | Выбор значения из выпадающего списка | Нет | Строка | Выпадающий список |
| Год создания книги |  | Да | Целое число | Текстовое поле |
| Файл книги | Формат pdf-файла | Да | Файл | Проводник |
| ФИО автора | Выбор значения из выпадающего списка | Нет | Строка | Выпадающий список |
| Год рождения автора |  | Да | Целое число | Текстовое поле |
| Фотография автора | Формат медиа-файла | Да | Файл | Проводник |
| Год смерти автора | Формат pdf-файла | Нет | Файл | Проводник |

В таблице 5 представлены выходные данные.

Таблица 5 – Выходные данные

|  |  |  |  |
| --- | --- | --- | --- |
| Название | Пустое поле | Формат | Формат вывода |
| Фотография книги | Нет | Строка-ссылка | Изображение |
| Файл книги | Нет | Байты | Набор изображений |
| Год создания книги | Нет | Целое число | Строка |
| ФИО автора | Нет | Строка | Строка |
| Название жанра | Нет | Строка | Строка |
| Год рождения автора | Нет | Строка | Строка |
| Год смерти автора | Нет | Строка | Строка |
| Первый цвет жанра | Нет | Строка | Цвет градиента карточки |
| Второй цвет жанра | Нет | Строка | Цвет градиента карточки |
| Логин пользователя | Нет | Строка | Строка |
| Фотография пользователя | Нет | Строка-ссылка | Изображение |
| Фотография жанра | Нет | Строка-ссылка | Изображение |
| Фотография автора | Нет | Строка-ссылка | Изображение |

### 2.2.3. Методы

Необходимо провести нормализацию базы данных до третьей формы и придерживаться принципа размещения всех функций по манипулированию данными внутри самой базы данных.

Также следует предоставить готовые методы для сложных вычислений через программный интерфейс, чтобы клиенты не занимались такими задачами. База данных должна быть расположена удаленно от клиентской части.

Необходимо реализовать программный интерфейс для взаимодействия между клиентскими частями и сервером базы данных.

Клиентские части должны быть написаны в соответствии с методологией объектно-ориентированного программирования и включать элементы реактивного программирования. Компоненты клиентских частей должны быть четко структурированы и расширяемы для возможности будущего дополнения.

Необходимо использовать инкапсуляцию, которая позволяет защитить данные от нежелательного доступа и изменений, а также облегчить поддержку кода и его повторное использование с помощью скрытии определённых данных программы от других данных. В приложении требуется использовать оптимизацию.

Оптимизация, с другой стороны, направлена на улучшение производительности приложения, т.е. на его скорость работы и эффективность использования ресурсов устройства. Оптимизация может включать изменение кода приложения, чтобы он работал быстрее, или использование оптимизированных библиотек и инструментов.

### 2.2.4. Тесты

1. По формальности тестирования.

Тестирование по тестам направлено на проверку функций мобильного приложение на соответствие написанным тест-кейсам.

Используется для того, чтобы с помощью некоторых шагов, убедиться в правильной работе программы.

1. По исполнению кода.

Динамическое тестирование направлено на проверку программного кода во время его исполнения. Используется в созданном мобильном приложении для того, чтобы выявить ошибки, которые могут быть пропущены при статическом тестировании.

1. По уровню тестирования.

Системное тестирование направлено на проверку работы всей системы на соответствие заявленным требованиям к программному продукту. Позволяет в разработанном приложении выявить ошибки, связанные с взаимодействием различных компонентов системы.

1. По целям.

Функциональное тестирование направлено на проверку реализации функций программного продукта и их точности, что позволяет убедиться в правильной работе разработанного мобильного приложения в соответствии с ожиданиями пользователя.

1. По степени автоматизации.

Ручное тестирование направлено на проверку программы без использования дополнительных программных средств, которая позволяет оценить работу разработанного мобильного приложения с точки зрения пользователя.

1. По позитивности сценария.

Позитивное тестирование направлено на проверку соответствия функций мобильного приложения ожидаемому поведению.

Негативное тестирование направлено на проверку работы программы в случае, когда поведение пользователя отличается от ожидаемого, что позволяет в разработанном мобильном приложении выявить ошибки, которые могут возникнуть в реальном использовании программы.

1. По знанию системы.

Тестирование «белого ящика» направлено на тестирование мобильного приложения для глубокого понимания, как работает программа и выявление ошибок, связанные с кодом.

Подробные сценарий и результаты тестовых испытаний мобильного приложения в Приложении B. Сценарий и результаты тестовых испытаний.

1. По разработке тестовых испытаний.

Тестирование на основе требований направлено на тестирование разработанного мобильного приложения для того, чтобы убедиться в том, что программа работает в соответствии с требованиями.

### 2.2.5. Контроль целостности данных

В процессе разработки были реализованы методы валидации входных данных в информационной системе, которые обеспечивают целостность этих входных данных.

Контроль целостности, описывающих ситуации и реакции приложения на выполнения функций представлен в таблице 6.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Ситуация | Аномалия | Реакция | Примечание |
| 1 | 2 | 3 | 4 | 5 |
| 1 | Регистрация | Введёно два различных пароля и/или введены некорректные, согласно ограничениям, данные в  соответствующие поля | Приложение обработает ошибку и выведет пользователю диалоговое окно с содержанием ошибки. | Регистрация возможна только с совпадающими паролями,  логином, паролем, соответствующим ограничениям полей |
| 2 | Авторизация | Введён существующий логин и/или пароль и/или введены некорректные, согласно ограничениям, данные в  соответствующие поля | Приложение обработает ошибку и выведет пользователю диалоговое окно с  содержанием ошибки. | Авторизация возможна только с существующим логином и паролем одного пользователя |
| 3 | Создание книги | Введёны данные, не соответсвующие ограничениям соответствующих полей или не заполнено одно из полей (кроме авторов) | Приложение обработает ошибку и выведет пользователю диалоговое окно с  содержанием ошибки. | Создание книги возможно только с валидными полями, соответствующим ограничениям полей |
| 4 | Создание автора | Введёны данные, не соответсвующие ограничениям соответствующих полей или не заполнено одно из полей (кроме года смерти автора) | Приложение обработает ошибку и выведет пользователю диалоговое окно с  содержанием ошибки. | Создание автора возможно только с валидными полями, соответствующим ограничениям полей |

Для тестирования работы API была использована сторонняя программа под названием Postman. Postman является мощным инструментом для разработчиков, специально предназначенным для тестирования API. С помощью Postman была создана возможность отправки и анализа запросов к API. Программа обладает интуитивно понятным интерфейсом, что делает настройку и выполнение тестовых сценариев простыми и удобными. Postman также предлагает функционал для автоматизации тестирования, создания коллекций запросов и обеспечения повторяемости и согласованности тестовых сценариев. Это позволяет обнаружить и исправить возможные проблемы в работе API, гарантируя его надежную функциональность. На рисунках 1-3 показаны примеры тестирования API.

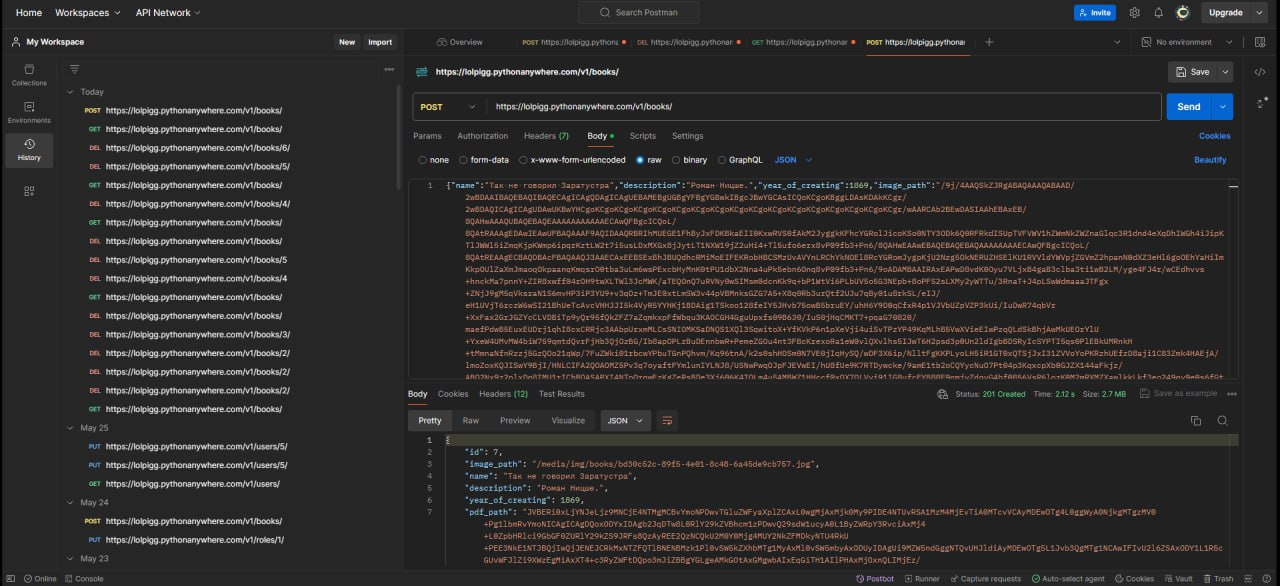


Рисунок 1 – Тестирование запроса POST

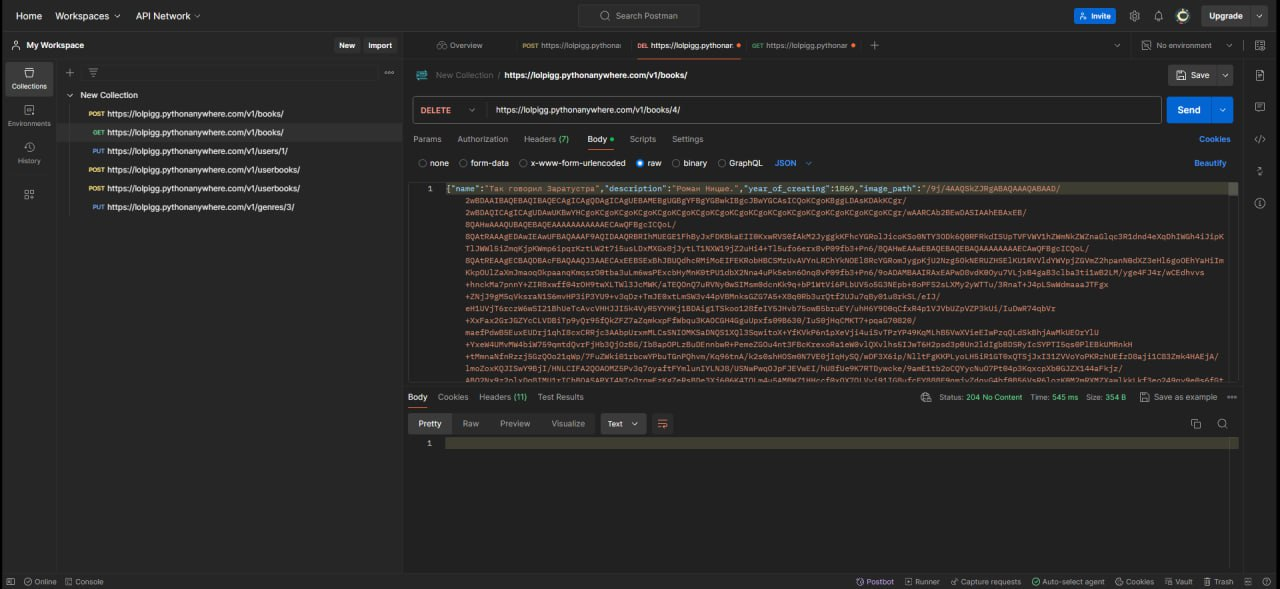


Рисунок 2 – Тестирование запроса DELETE

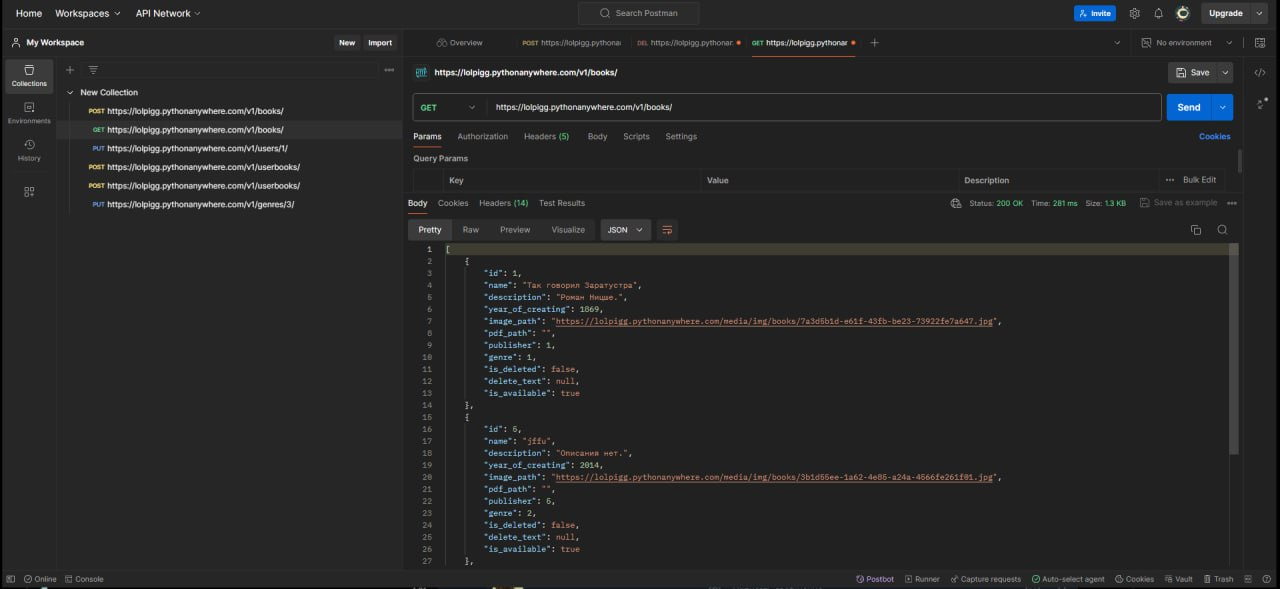


Рисунок 3 – Тестирование запроса GET

## 2.3. Проектирование

### 2.3.1. Схема архитектуры приложения

Компонент пользовательский интерфейс виден пользователю и взаимодействует с ним. В зависимости от действия пользователя осуществляются запросы к API, которая возвращает ответы на запрос в виде данных.

API используется для взаимодействия с базой данных sqlite, на котором хранятся данные, и для получения и отправки данных.

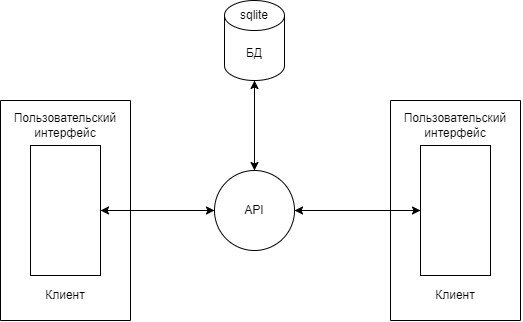


Рисунок 4 – Архитектурная схема мобильного приложения.

### 2.3.2. Логическая схема данных

Разработанная логическая модель базы данных, представленная на Рисунке 8, послужила основой для реализации логики манипуляции данными в проектируемой базе данных.



Рисунок 5 - Логическая схема базы данных

### 2.3.3. Физическая схема данных

Разработанная физическая модель базы данных, изображенная на Рисунке 9, послужила основой для реализации логики хранения и защиты данных в проектируемой базе данных.

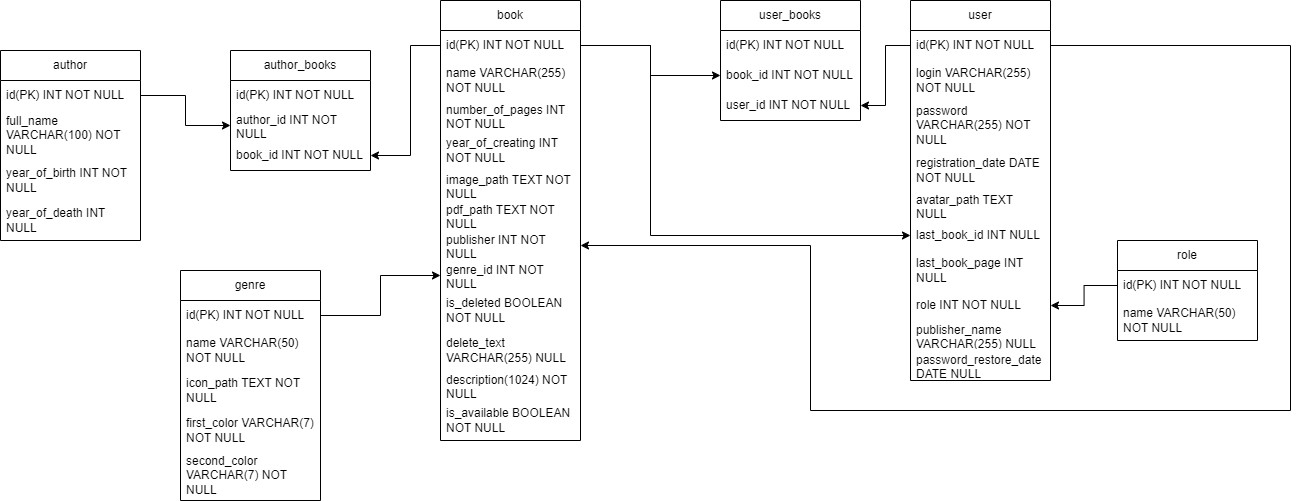


Рисунок 6 – Физическая модель базы данных

В проектируемой базе данных было выделено 7 сущностей (таблиц), каждая из которых содержит набор атрибутов (полей) для записи информации. Уникальное поле-идентификатор, присутствующее в каждой таблице, используется для формирования связей между ними.

В таблице 6 представлен словарь данных реализуемой базы данных для данного мобильного приложения.

Таблица 6 – Словарь базы данных

|  |  |  |  |
| --- | --- | --- | --- |
| Наименование поля | Ключ | Тип данных | Описание |
| 1 | 2 | 3 | 4 |
| Сущность «Role» | | | |
| id | PK | int | Идентификатор роли |
| name |  | varchar(50) | Название роли |
| Сущность «User» | | | |
| id | PK | int | Идентификатор пользователя |
| login |  | varchar(255) | Логин |
| password |  | varchar(255) | Пароль |
| role | FK | Int | Идентификатор роли |
| registration\_date |  | Date | Дата регистрации |
| password\_restore\_date |  | Date | Дата запроса о восстановлении пароля |
| avatar\_path |  | Text | Путь к изображению аватара |
| last\_book\_id | FK | Int | Id последней книги |
| last\_book\_page |  | Int | Номер последней страницы |
| publisher\_name |  | Text | Название издательства |
| Сущность «Book» | | | |
| id | PK | int | Идентификатор книги |
| name |  | varchar(255) | Название книги |
| description |  | varchar(1024) | Описание книги |
| number\_of\_pages |  | int | Количество страниц |
| year\_of\_creating |  | int | Год написания |
| image\_path |  | text | Путь к изображению книги |
| pdf\_path |  | text | Путь к файлу книги |
| genre\_id | FK | int | Идентификатор жанра |
| is\_deleted |  | boolean | Логическое удаление |
| delete\_text |  | varchar(255) | Причина удаления |
| is\_available |  | boolean | Выпущена ли книга |
| publisher | FK | int | Выпущена ли книга |
| Сущность «User\_books» | | | |
| id | PK | int | Идентификатор таблицы |
| book\_id |  | int | Идентификатор книги |
| user\_id |  | int | Идентификатор пользователя |
| Сущность «Author\_books» |
| id | PK | int | Идентификатор таблицы |
| book\_id | FK | int | Идентификатор книги |
| author\_id | FK | int | Идентификатор автора |
| Сущность «Author» | | | |
| id | PK | int | Идентификатор таблицы |
| full\_name |  | Varchar(100) | Название направления |
| year\_of\_birth |  | int | Год рождения автора |
| year\_of\_death |  | int | Год смерти автора |
| direction\_id | FK | int | Идентификатор направления |
| Сущность «Genre» | | | |
| id | PK | int | Идентификатор таблицы |
| name |  | Varchar(50) | Название жанра |
| icon\_path |  | text | Путь к иконке жанра |
| first\_color |  | Varchar(7) | Первый цвет градиента жанра |
| second\_color | FK | Varchar(7) | Второй цвет градиента жанра |

### 2.3.4. Структурная схема

На Рисунке 10 представлен общий план структурной схемы мобильного приложения по вкладкам, в которой графически показано взаимодействие всех модулей и их взаимодействие друг с другом.

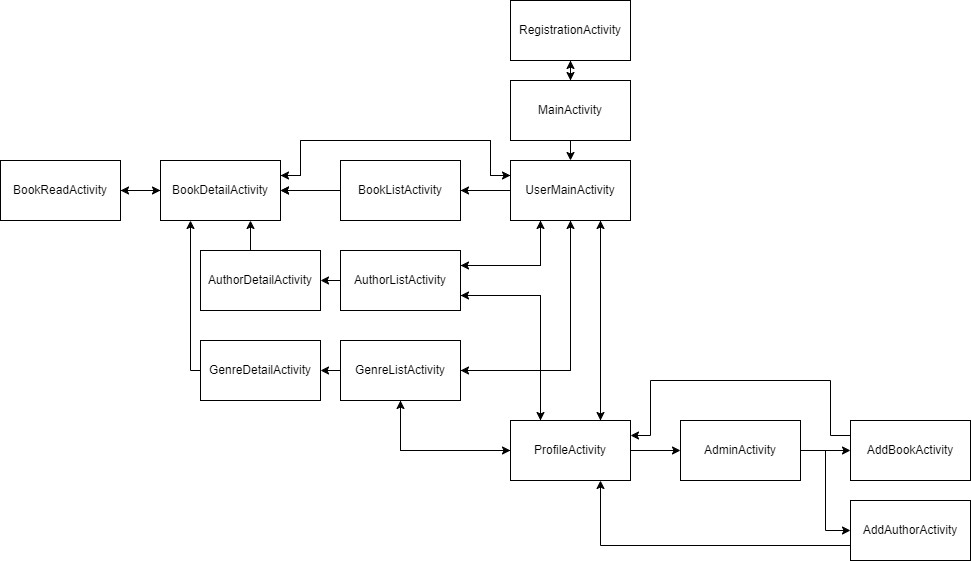


Рисунок 7 – Структурная схема

1. MainActivity – Окно авторизации
2. RegistrationActivity – Окно регистрации
3. UserMainActivity – Окно главного меню
4. BookListActivity – Окно списка книг
5. BookReadActivity – Окно чтения книги
6. AuthorListActivity – Окно списка авторов
7. GenreListActivity – Окно списка жанров
8. ProfileActivity – Окно профиля пользователя
9. AdminActivity – Окно меню администратора
10. AddBookActivity – Окно добавления книги
11. AddAuthorActivity – Окно добавления автора
12. BookDetailActivity – Окно детальной информации книги
13. AuthorDetailActivity – Окно детальной информации автора
14. GenreDetailActivity – Окно книг по жанру

### 2.3.5. Функциональная схема

На Рисунке 11 представлена функциональная схема мобильного приложения с вкладками, на котором графически изображены возможности пользователя в процессе эксплуатации.

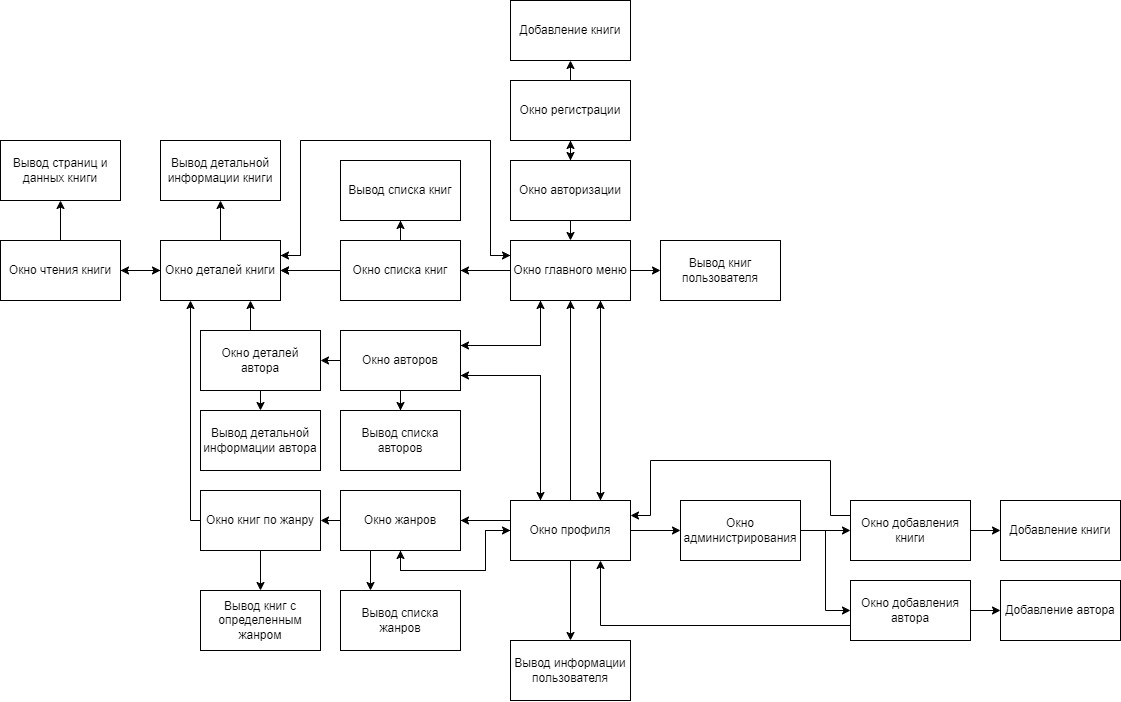


Рисунок 8 – Функциональная схема

### 2.3.6. Диаграмма классов

На рисунках 9 - 13 представлена диаграмма класса мобильного приложения. На ней графически изображены классы, которые состоят из полей, методов и свойств.

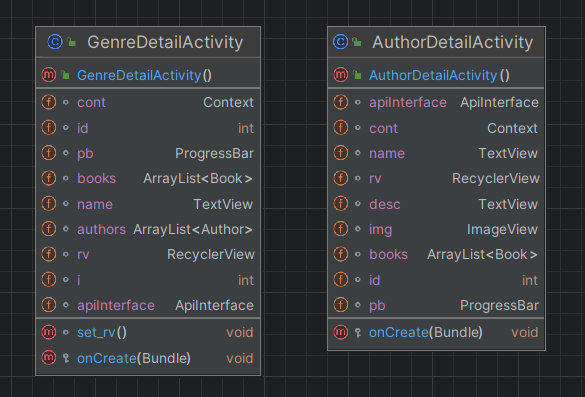


Рисунок 9 – Диаграмма классов (1)

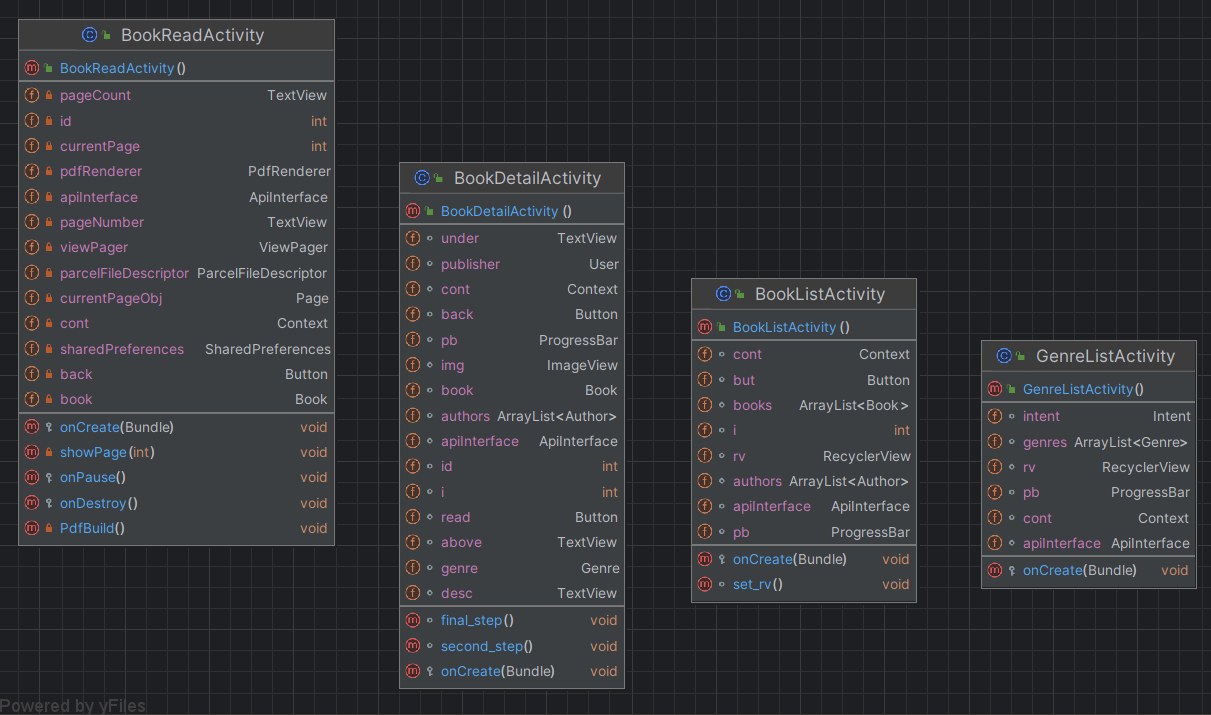


Рисунок 10 - Диаграмма классов (2)

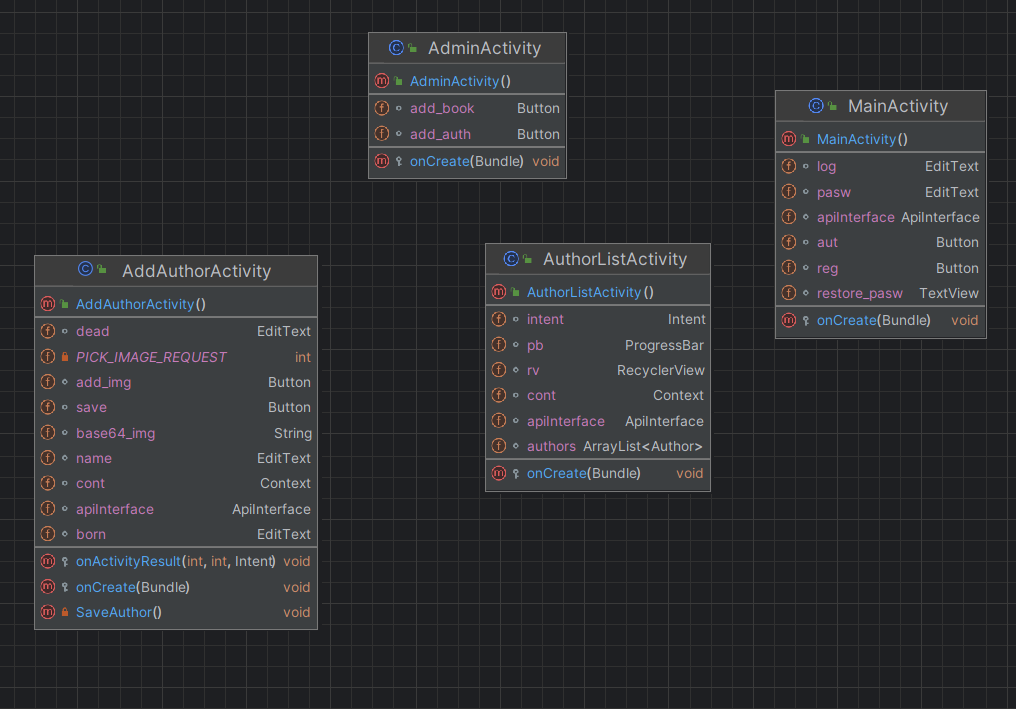


Рисунок 11 - Диаграмма классов (3)

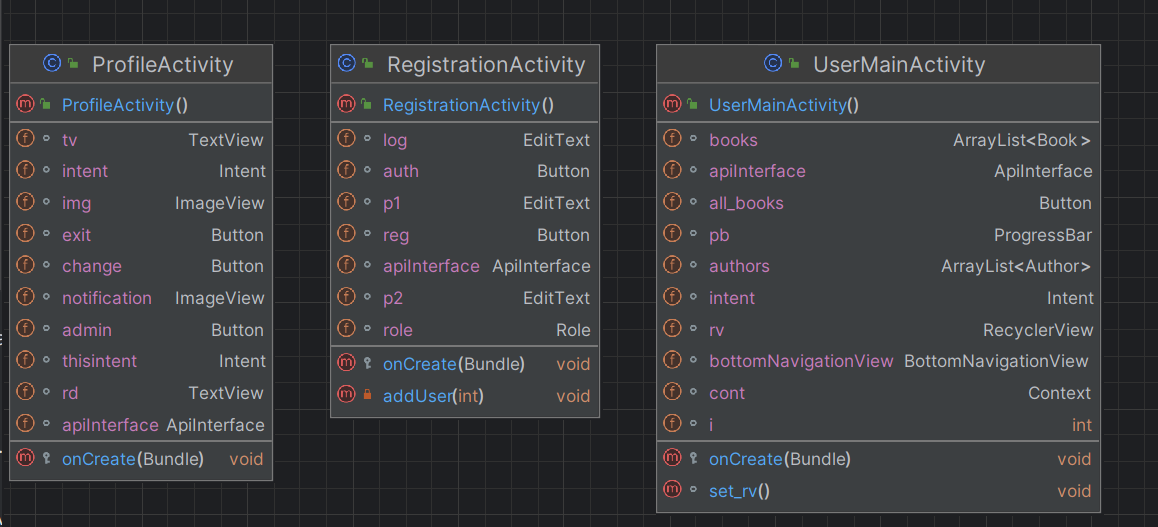


Рисунок 12 - Диаграмма классов (4)

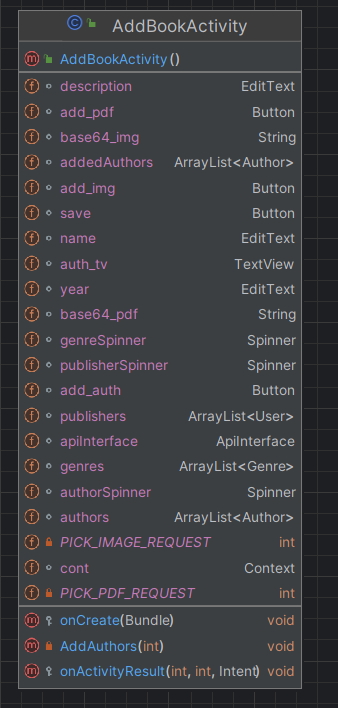


Рисунок 13 - Диаграмма классов (5)

Таблица 7 – Описание классов

|  |  |
| --- | --- |
| Название класса | Описание класса |
| MainActivity | Окно авторизации |
| RegistrationActivity | Окно регистрации |
| UserMainActivity | Окно главного меню |
| BookListActivity | Окно списка книг |
| BookReadActivity | Окно чтения книги |
| AuthorListActivity | Окно списка авторов |
| GenreListActivity | Окно списка жанров |
| ProfileActivity | Окно профиля пользователя |
| AdminActivity | Окно меню администратора |
| AddBookActivity | Окно добавления книги |
| AddAuthorActivity | Окно добавления автора |
| BookDetailActivity | Окно детальной информации книги |
| AuthorDetailActivity | Окно детальной информации автора |
| GenreDetailActivity | Окно книг по жанру |

### 2.3.7. Схема тестирования

На рисунке 14 представлена схема тестирования, по которой проводилось тестирование мобильного приложения.

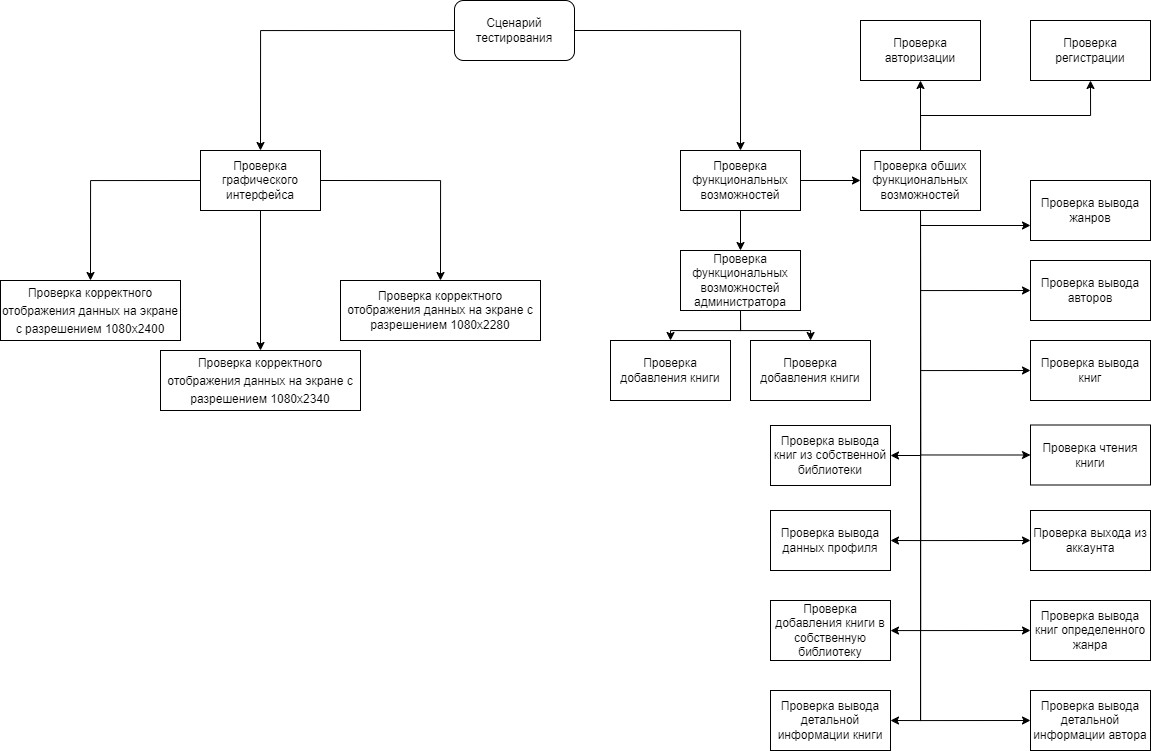


Рисунок 14 – Схема тестирования

### 2.3.8. Схема пользовательского интерфейса

Согласно схеме на рисунке 15 неавторизированный пользователь, авторизованный пользователь, администратор, менеджер. Могут пользоваться приложением и своими функциями в зависимости от роли. Неавторизованный пользователь ограничен в функционале, но может использовать базовый функционал без авторизации. Внизу экрана находятся вкладки по которым можно перемещаться.

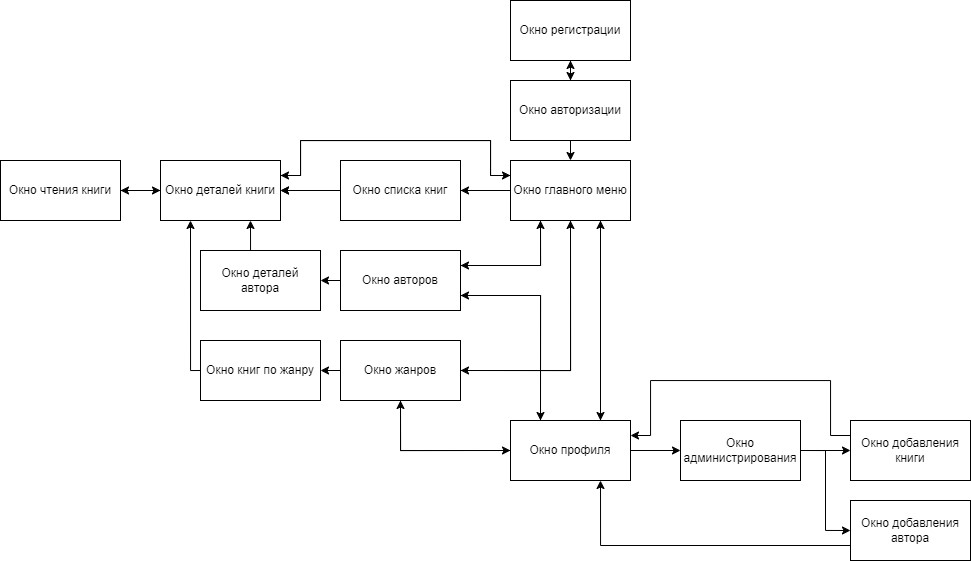


Рисунок 15 – Схема пользовательского интерфейса

## Результат работы программы

В результате поставленной задачи было разработано требуемое программное решение в виде мобильного приложения.

На Рисунках 16-19 представлен некоторый результат работы программы. Более развёрнуто результат работы программы описан в Приложении Г «Руководство пользователя».

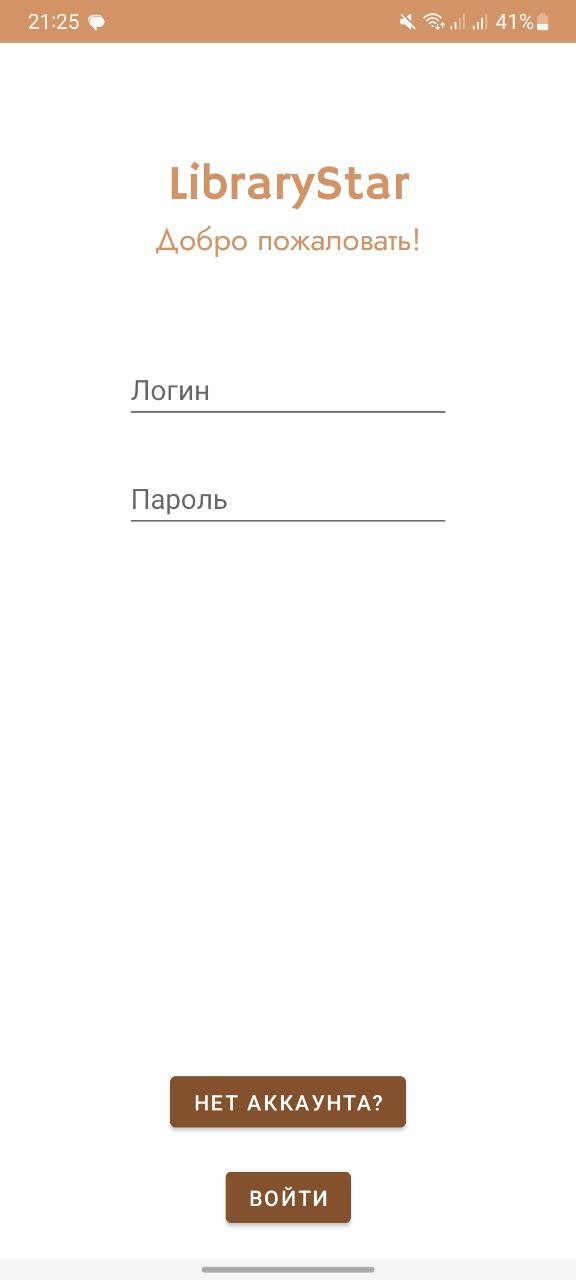


Рисунок 16 – Окно авторизации

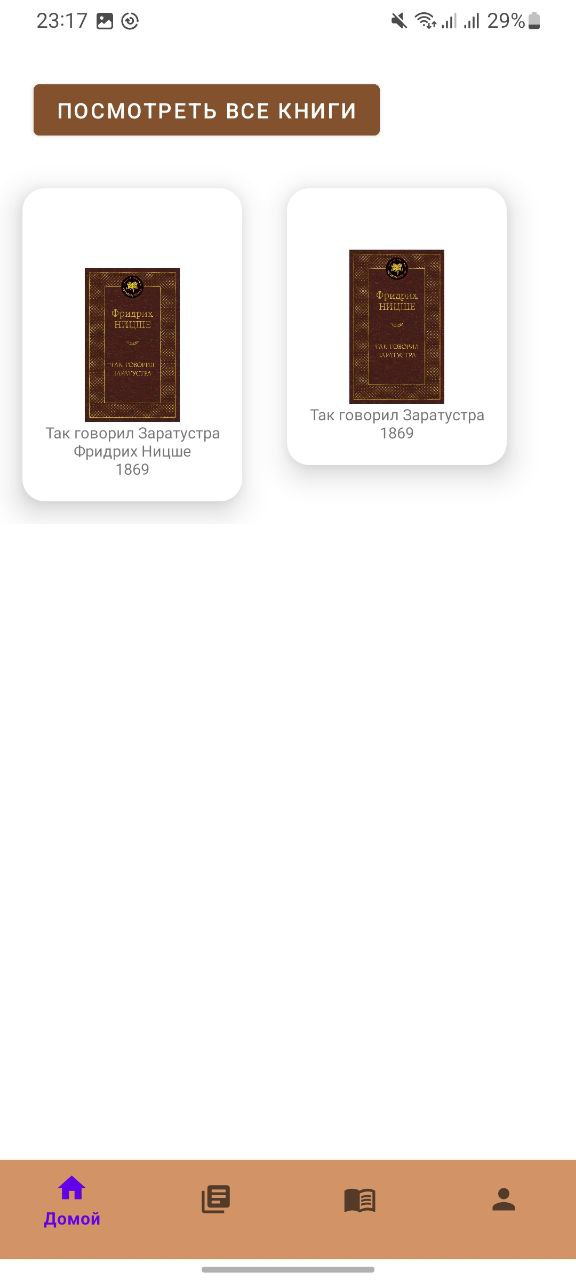


Рисунок 17 – Главное окно пользователя

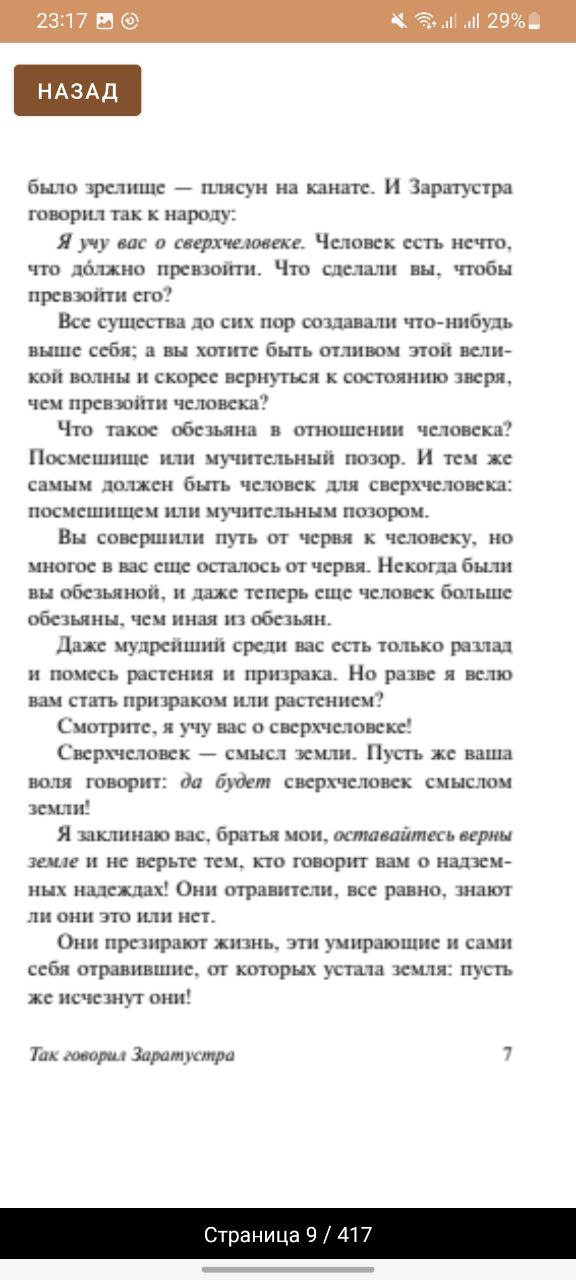


Рисунок 18 – Чтение книги

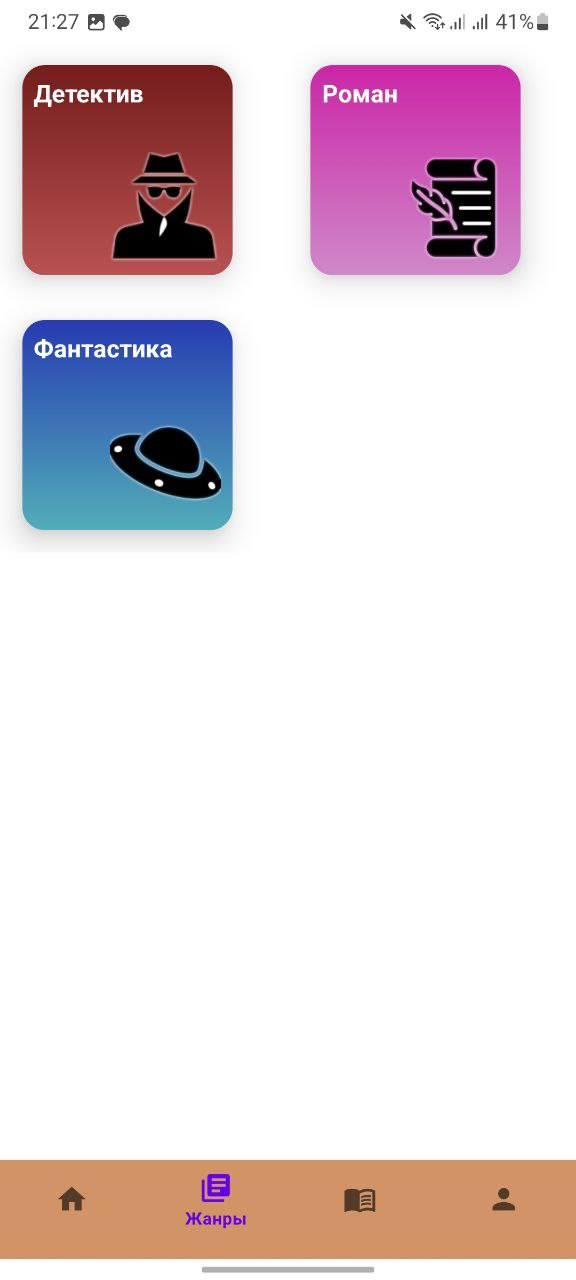


Рисунок 19 – Список жанров

# ТЕХНОЛОГИЧЕСКАЯ ЧАСТЬ

## 3.1. Инструментальные средства

Java является основным языком программирования для разработки приложений под платформу Android. Android Studio предоставляет удобную интегрированную среду разработки (IDE) с множеством инструментов для создания, отладки и тестирования приложений. Java обеспечивает высокую производительность и широкие возможности для разработки функциональных приложений, из-за чего и была выбрана основным языком для разработки.

В качестве API был использован Django REST Framework (DRF) - это мощный инструмент для создания веб-серверов и API на основе фреймворка Django. Он обеспечивает простой способ создания RESTful API, включая аутентификацию, сериализацию данных, обработку запросов и многое другое. DRF позволяет эффективно взаимодействовать с мобильным приложением, обеспечивая передачу данных между клиентом и сервером.

В качестве БД была использована SQLite - легковесная встроенная СУБД, которая идеально подходит для мобильных приложений. Она обеспечивает надежное хранение данных в локальной базе данных на устройстве пользователя. SQLite интегрируется хорошо с платформой Android и обеспечивает удобный способ хранения и управления данными в приложении.

Для разработки API была использована среда PyCharm - это популярная интегрированная среда разработки для Python. С его помощью можно эффективно создавать серверную часть приложения на Python с использованием Django, который предоставляет мощный набор инструментов для разработки веб-приложений. Django обеспечивает удобную работу с базой данных, управление URL-адресами, аутентификацию, административный интерфейс и другие функции, необходимые для создания веб-приложений.

А в качестве хостинга был выбран PythonAnywhere – это условно-бесплатный облачный хостинг, специализирующийся на размещении и запуске приложений Python и Django. Он обеспечивает удобное развертывание веб-приложений без необходимости управления инфраструктурой серверов. PythonAnywhere позволяет развернуть веб-приложение на удаленном сервере, обеспечивая доступ к нему из интернета.

Этот стек технологий был выбран из-за их популярности, надежности, удобства использования и возможности обеспечить полноценное функционирование мобильного приложения электронной библиотеки как на стороне клиента (Android), так и на стороне сервера (Django). Каждый инструмент играет свою роль в обеспечении эффективной разработки, безопасности и удобства использования приложения.

## 3.2 Отладка программы

Отладка мобильного приложения была проведена с помощью встроенного инструмента запуска проекта через AndroidStudio, который позволяет запустить эмулятор и выводит все появляющиеся ошибки, в том числе через инструмент Logcat.

## 3.3. Защитное программирование

В программном коде предусмотрены защиты пароля при помощи хеширования его через технологию sha256.

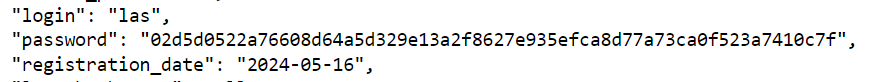


Рисунок 42 – Пример хэш-пароля в БД

## 3.4. Характеристики программы

Разработанное мобильное приложение может запускаться на любых телефонах на Android, которые имеют версию операционной системы не ниже 9.0 Pie.

# ЗАКЛЮЧЕНИЕ

В результате курсового проекта было разработано мобильное приложение по управлению личной библиотекой «LibraryStar».

В процессе разработки мобильного приложения был реализован весь функционал, который соответствует техническому заданию и выполняет заданные в нем цели.

В процессе анализа была выполнена работа по постановке и описанию задачи.

В ходе курсового проекта были выполнены необходимые работы по проектированию, которые включали в себя построение различных схем и моделей данных, а также описание входных и выходных данных. Это позволило эффективно начать работу над мобильным приложением, которое было разработано в соответствии с требованиями и удовлетворяло управления личной библиотекой.

Перед началом разработки приложения был проведен подготовительный этап, который включал установку необходимых служб для работы с хостингом и создания API. Этот этап позволил грамотно организовать работу с базой данных и обеспечить взаимодействие между мобильным приложением и API.

В процессе разработки мобильного приложения был реализован весь функционал, который позволял пользователям читать книги, формировать собственную библиотеку, просматривать авторов и прочие функции. Приложение было разработано с использованием языка программирования Java при помощи инструмента Android Studio. Также оно сочетает в себе удобный интерфейс, функциональность и простоту использования.

На этапе тестирования было проведено тщательное тестирование всех функциональных возможностей приложения, что позволило выявить и исправить недочеты, связанные с функциональностью и производительностью. После тестирования стало очевидно, что приложение работает без сбоев и вылетов благодаря использованию различных методов защитного программирования.

Выполнение курсового проекта позволило закрепить и расширить практические навыки в разработке и проектировании баз данных под управлением встроенной СУБД SQLite, создании и подключении API, его загрузку на хостинг и управление им, а также в разработке мобильных приложений. Результат выполнения курсового проекта полностью соответствовал поставленным задачам и позволил получить новые знания и опыт.

Хотя, разработанное мобильное приложение уже обладает удобным интерфейсом, функциональностью и простотой использования, всегда есть возможность улучшить его в будущем. Одним из возможных улучшений является оптимизация запросов, добавление функционала платных подписок, а также расширением возможностей администрирования и интеграции электронной почты.

В целом, создание приложение управления собственной библиотекой, значительно поможет пользователям в формировании собственной библиотеки, чтении книг, и сэкономит их время.

# СПИСОК ИСПОЛЬЗОВАННЫХ МАТЕРИАЛОВ

1. Начальное руководство для хостинга Django-проекта на PythonAnywhere URL: <https://help.pythonanywhere.com/pages/FollowingTheDjangoTutorial/> (дата обращения : 15.04.2024)
2. Metanit Выпадающий список Spinner URL: <https://metanit.com/java/android/5.4.php> (дата обращения : 04.05.2024)
3. Metanit Recycler View URL: <https://metanit.com/java/android/5.11.php> (дата обращения : 25.04.2024)
4. Использование Retrofit 2.x в качестве REST клиента — Tutorial URL: <https://habr.com/ru/articles/428736/> (дата обращения : 11.04.2024)
5. Использование Base64 с Java URL: <https://community.qlik.com/t5/Design-and-Development/Rest-API-Call-with-Base-64-Encode/td-p/2336145> (дата обращения : 01.05.2024)