Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное

образовательное учреждение высшего образования

«Сыктывкарский государственный университет имени Питирима Сорокина»

(ФГБОУ ВО «СГУ им. Питирима Сорокина»)

Институт точных наук и информационных технологий

Кафедра информационных систем

Курсовая работа по дисциплине «Информационные системы»

**ПРОЕКТИРОВАНИЕ И РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ «БИБЛИОТЕКА» НА ПЛАТФОРМЕ ANDROID С ИСПОЛЬЗОВАНИЕМ ТЕХНОЛОГИИ «КЛИЕНТ-СЕРВЕР»**

Направление подготовки

09.03.03 Прикладная информатика

Направленность (профиль) программы

Прикладная информатика в экономике

Исполнитель:

Пестерев Владислав Олегович \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Личная подпись

Научный руководитель:

К. ф.-м. н., доцент

Гольчевский Юрий Валентинович \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Личная подпись

Сыктывкар

2022

**Содержание**

[Введение 3](#_Toc106031230)

[Глава 1. Анализ предметной области 4](#_Toc106031231)

[1.1 Объект исследования 4](#_Toc106031232)

[1.2 Анализ бизнес-процесса 7](#_Toc106031233)

[1.3 Концептуальное моделирование данных предметной области 11](#_Toc106031234)

[Глава 2. Проектирование базы данных 13](#_Toc106031235)

[2.1 Логическое проектирование базы данных 13](#_Toc106031236)

[2.2 Физическое проектирование базы данных 13](#_Toc106031237)

[Глава 3. Разработка программного обеспечения 20](#_Toc106031238)

[1.1 Инструменты, используемые при создании программного обеспечения 20](#_Toc106031239)

[1.2 Пользовательский интерфейс и тестирование приложения 21](#_Toc106031240)

[1.3 Программная реализация 32](#_Toc106031241)

[Заключение 42](#_Toc106031242)

[Список литературы 43](#_Toc106031243)

# Введение

Темп современной жизни человека возрастает с каждым днём. Следовательно, люди стараются выполнить максимальное количество задач с максимальной эффективностью. В таких целях создаются программы, автоматизирующие существующие бизнес-процессы организаций.

Одной из таких программ является мобильное приложение-клиент на платформе Android – «Библиотека», автоматизирующий соответствующий бизнес-процесс.

Мобильное приложение автоматизирует работу библиотеки и сводит к минимуму взаимодействия между посетителями и персоналом библиотеки посредством автоматизации таких задач, как: просмотр книг в наличии, просмотр количества доступных книг; бронирование книг; добавление книги в список желаемого; отслеживание даты сдачи книг, взятых на руки; просмотр мероприятий, проходящих в библиотеке, запись на мероприятия.

Целью исследования является проектирование и разработка мобильного приложения для библиотеки.

В качестве достижения данной цели использовалось следующее ПО:

1. Microsoft Word – Текстовый редактор;
2. PhpMyAdmin – веб-приложение с открытым кодом, написанное на языке PHP и представляющее собой веб-интерфейс для администрирования СУБД MySQL;
3. Visual Studio Code – Текстовый редактор с подсветкой кода;
4. Android Studio – Среда разработки программного обеспечения на платформе Android;
5. Visual Paradigm Community Edition – Программное обеспечение для моделирования.

# Глава 1. Анализ предметной области

## Объект исследования

Объектом исследования является Национальная библиотека Республики Коми им. С. Я. Маршака.

Библиотека занимается хранением, составлением записей о книгах, выдачей книг, как детям, так и взрослым. Также проводятся события в образовательных, развлекательных и обучающих целях, направленные на духовное развитие детей и подростков.

Основными участниками бизнес-процессов библиотеки являются:

1. Библиотекарь – сотрудник библиотеки, выдающий и принимающий книги;
2. Взрослый посетитель (в т.ч. родитель) – посетитель библиотеки от 18 лет;
3. Ребёнок – посетитель библиотеки до 14 лет, не имеющий паспорт, однако имеющий возможность регистрации через родителя;
4. Подросток – посетитель библиотеки от 14 до 17 лет, имеющий паспорт и возможность самостоятельной регистрации;
5. Гость – не зарегистрировавшийся и неавторизованный пользователь.

Полноценным функционалом мобильного приложения может пользоваться только зарегистрировавшийся в библиотеке пользователь, имеющий свой читательский билет и в дальнейшем авторизовавшийся через мобильное приложение. Были выявлены соответствующие сущности (зарегистрировавшийся и не зарегистрировавшийся в библиотеке пользователь), которые были в дальнейшем изображены на Use-Case диаграмме (см. рис. 1, рис.2, рис.3).

Также в ходе общения с представителем библиотеки и научным руководителем было выявлено, что, необходимо создание клиентского мобильного приложения, с помощью которого посетитель библиотеки может, как просмотреть книги, доступные в библиотеке, так и забронировать имеющиеся в наличии в библиотеке книги, а также продлить книгу одним нажатием кнопки в приложении.

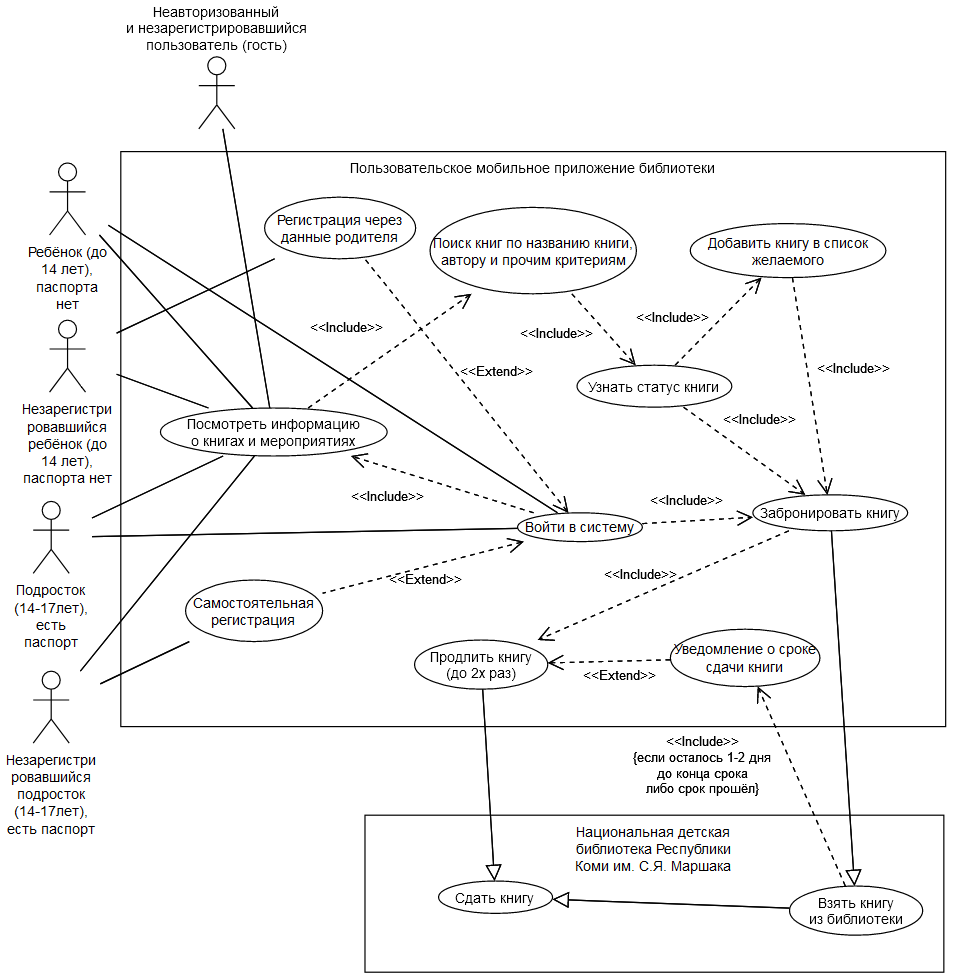


Рис. 1. Use-Case диаграмма. Часть 1

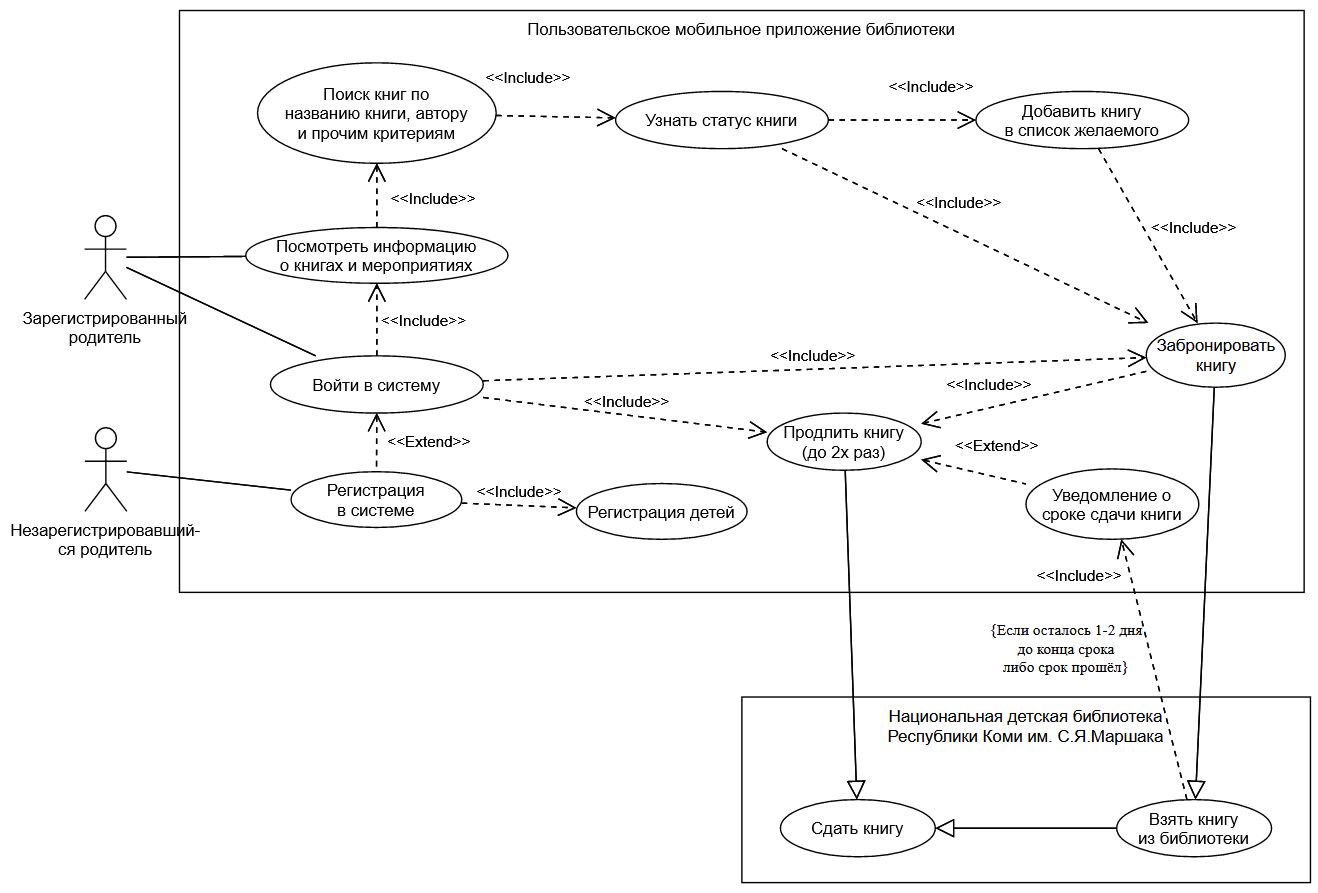


Рис. 2. Use-Case диаграмма. Часть 2

Вне данной курсовой работы в рамках сценария Use-Case предположим, что существует некоторое приложение для ПК, которым пользуется библиотекарь, чтобы взаимодействовать с записями (книги, пользователи, мероприятия, списки книг) базы данных, и впоследствии выдаёт книги посетителям и принимает книги с рук посетителей (см. рис. 3).

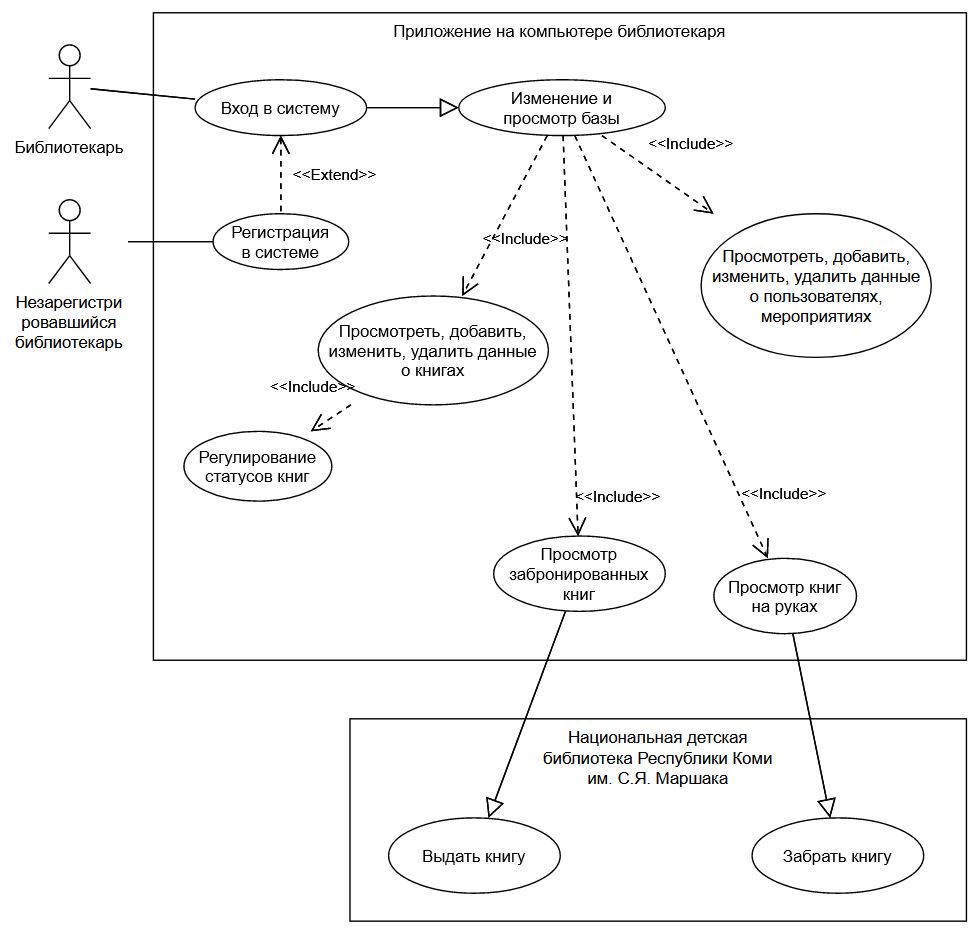


Рис. 3. Use-Case диаграмма. Часть 3

## Анализ бизнес-процесса

Основным бизнес-процессом исследования является «Автоматизация процессов функционирования библиотеки».

В качестве наглядной демонстрации бизнес-процесса использовалась диаграмма SADT (Structured Analysis and Design Technique). SADT-модель предоставляет полное и наглядное представление бизнес-процесса, для его упрощенного понимания [1].

Стандартом функционального моделирования является методология SADT – Structured Analyze and Design Technique (метод структурного анализа и проектирования) [5]. Методика SADT – это совокупность правил и процедур, предназначенных для построения функциональных моделей управляющих информационных систем, работающих в какой-либо предметной области. SADT-модель дает полное, точное и адекватное описание целенаправленной управляющей информационной системы, т.е. системы, имеющей конкретное целевое назначение [6].

SADT-диаграмма позволяет представить моделируемый процесс в виде следующих элементов:

* входные действия;
* выходные действия;
* управление;
* механизмы.

Диаграмма SADT нулевого уровня данного бизнес-процесса представлена на рис. 4.



Рис. 4. SADT нулевого уровня «Автоматизация процессов функционирования библиотеки»

На входе данного процесса: события и мероприятия, происходящие в библиотеке; книги; требования посетителей библиотеки.

В управлении бизнес-процесса находятся: график работы и адрес библиотеки, статусы, определяющие доступность выдачи пользователю тех или иных книг, определенные условия проведения мероприятий – место проведения, возрастной ограничение, количество посетителей, время проведения.

В механизмах участвуют само клиентское мобильное приложение, через которое пользователь получает информацию о книгах и может забронировать книгу, добавить ее в список желаемого или записаться на мероприятие, и совершить прочие действия; база данных, а также сам библиотекарь, который контролирует поступающие заявки на бронь книг и в течение дня собирает книги для выдачи их посетителю.

На выходе мы получаем книги, которые получает на руки посетитель библиотеки, и которые он в будущем будет обязан вернуть библиотеке; а также выходом будет результат того, что посетители будут посещать мероприятия, происходящие в библиотеке.

Далее на рис. 5 происходит декомпозиция бизнес-процесса.

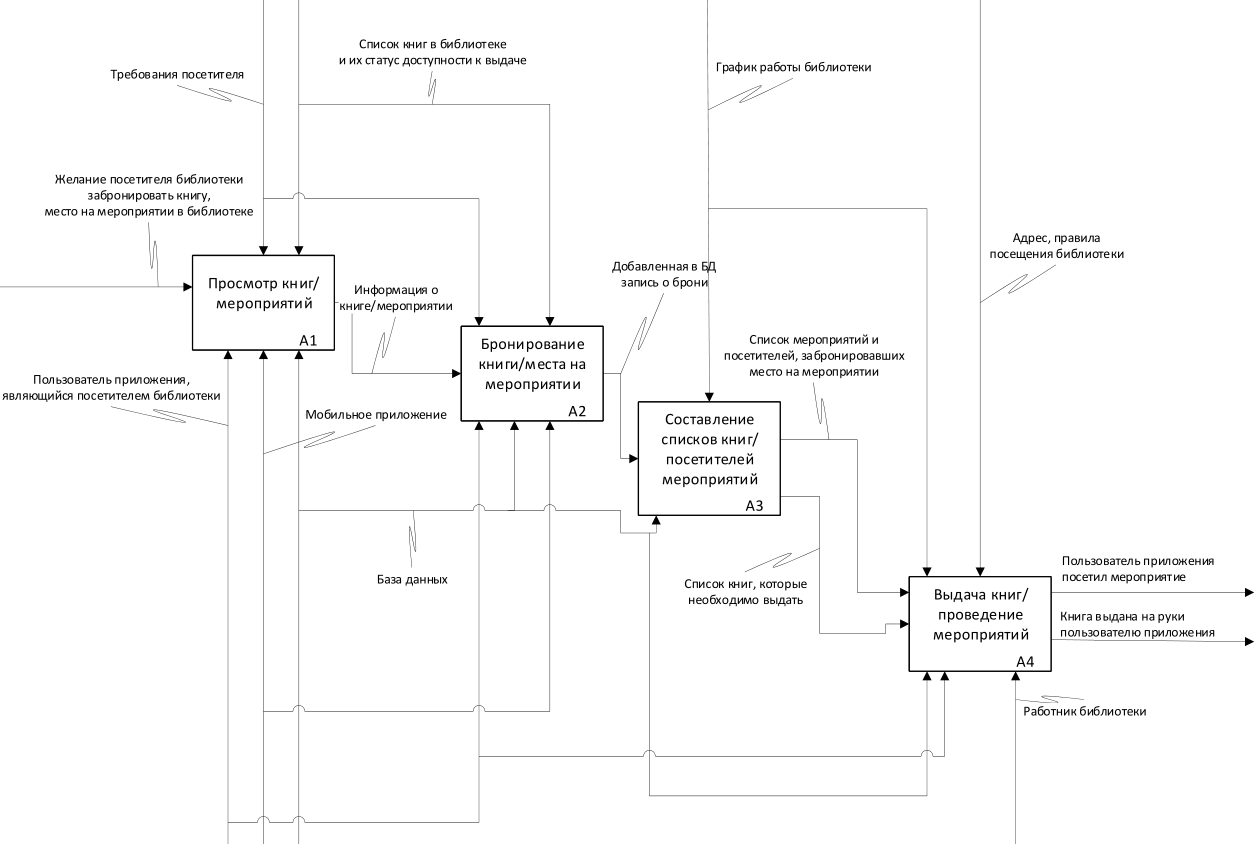


Рис. 5. Диаграмма декомпозиции первого уровня процесса «автоматизация процессов функционирования библиотеки»

Итак, выполнив анализ предметной области и главного бизнес-процесса можно выделить требования к результату исследования – мобильному приложению:

1. Отображение списка книг, доступных в библиотеке;
2. Отображение статуса книги: недоступен, доступен к выдаче или уже выдан;
3. Возможность добавления книги в список желаемого;
4. Бронирование книги в библиотеке с целью будущего получения на руки;
5. Просмотр списка книг на руках, забронированных книг, книг в списке желаемого;
6. Поиск книги по атрибутам «автор», «название», «тема»;
7. Регистрация нового посетителя библиотеки, авторизация в системе.

## Концептуальное моделирование данных предметной области

Концептуальная модель предметной области является основной для построения логические и физической модели данных приложения.

Для описания концептуальной модели используется диаграмма «сущность-связь» - ERD. ERD-модель обычно используется при разработке реляционной базы данных.

Проанализировав основной бизнес-процесс исследования «Автоматизация процессов функционирования библиотеки», можно выделить сущности:

1. «Книга» [2] – один из видов печатной продукции: непериодическое издание, состоящее из сброшюрованных или отдельных бумажных листов (страниц) или тетрадей, на которых нанесена типографским или рукописным способом текстовая и графическая (иллюстрации) информация, имеющее, как правило, твёрдый переплёт.

Атрибуты сущности «книга»:

* 1. Название;
  2. Автор;
  3. Обложка;
  4. Тема;
  5. Описание.

1. «Событие» – некоторое мероприятие, происходящее в библиотеке с участием посетителей библиотеки.

Атрибуты сущности «Событие»:

* 1. Название;
  2. Количество посетителей;
  3. Место проведения;
  4. Дата проведения;
  5. Возрастное ограничение;
  6. Описание.

1. «Посетитель» - зарегистрировавшийся пользователь библиотеки, который может посещать библиотеку и пользоваться мобильным приложением.

Атрибуты сущности «Посетитель»:

* 1. ФИО;
  2. Пол;
  3. Номер телефона;
  4. Дата рождения;
  5. Email;
  6. Логин;
  7. Пароль.

Как итог, на рис. 6 представлена концептуальная модель предметной области.

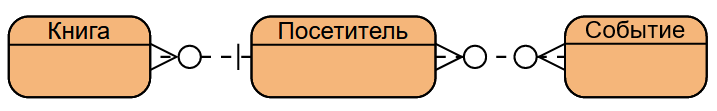


Рис. 6. Концептуальная модель предметной области

# Глава 2. Проектирование базы данных

## 2.1 Логическое проектирование базы данных

При построении логической модели данных учитывается организация данных. В исследовании используется реляционная модель организации данных. Основным понятием в реляционной модели является «отношение», которое обычно визуально представляется в виде таблицы.

Переходя от концептуальной модели к логической, у нас появились конкретные атрибуты и типы данных (см. рис. 7).

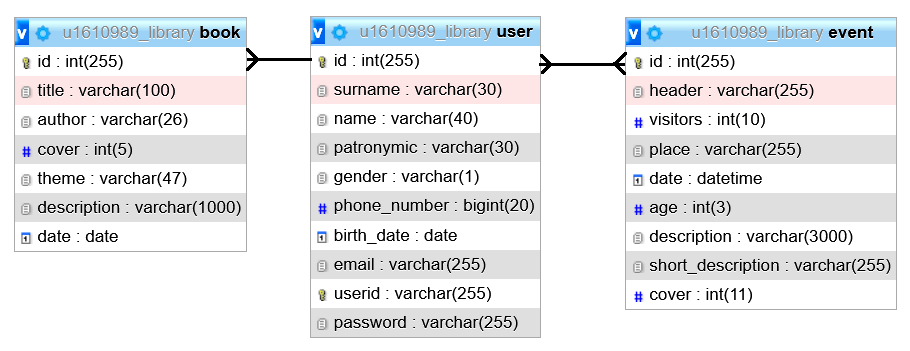


Рис. 7. Логическая модель базы данных

Итак, у нас имеются отношения:

1. «Книга» - «Пользователь» («Book» и «User» по рис. 7) - 1:∞;
2. «Пользователь – «Событие» («User» и «Event» по рис. 7) - ∞:∞.

Чтобы обозначить данные отношения и избежать избыточности, были созданы дополнительные сущности: «Пользователь-Событие», «Книги на руках», «Список желаемого», «Забронированные книги».

## 2.2 Физическое проектирование базы данных

Для разработки базы данных используется СУБД phpMyAdmin. В связи с этим физическая модель базы данных принимает вид, представленный на рис. 8.

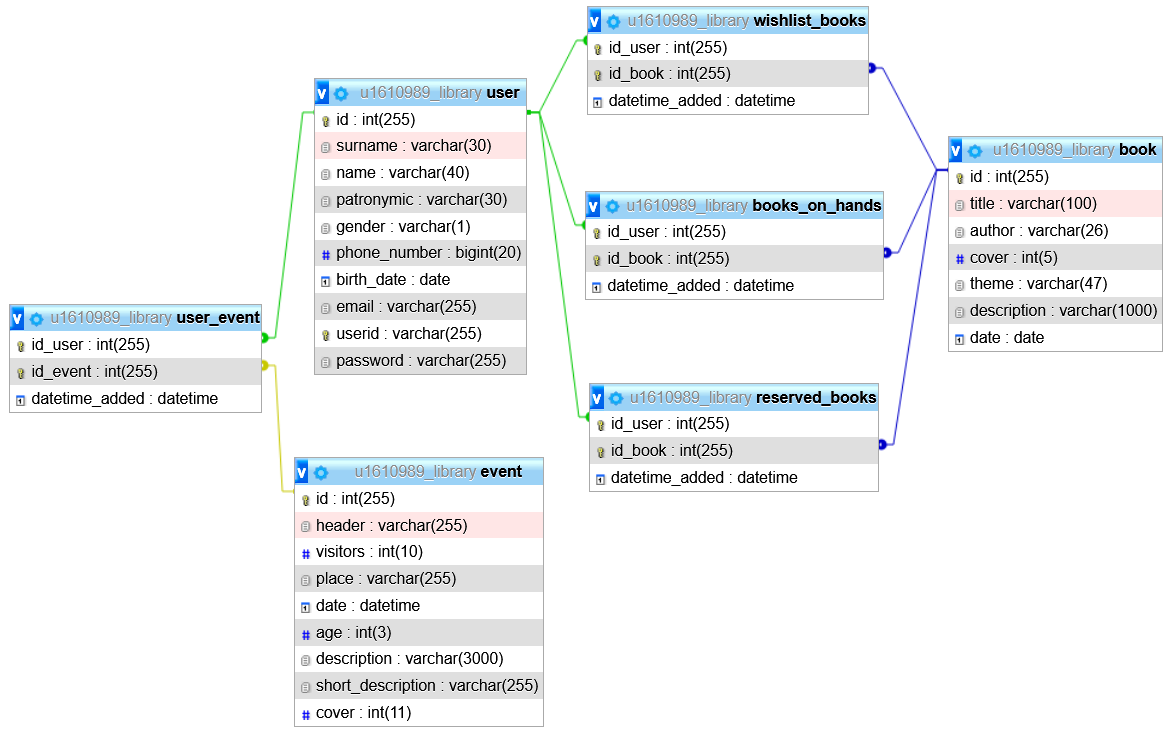


Рис. 8. Физическая модель базы данных

Как было описано в Главе 2 п.1. появились новые отношения: «Пользователь-Событие», «Книги на руках», «Список желаемого», «Забронированные книги» (Соответственно рис. 8 – «user\_event», «books\_on\_hands», «wishlist\_books», «reserved\_books»).

Структура базы данных:

База данных имеет 7 отношений: User, Event, Book, User\_Event, Wishlist\_Books, Books\_On\_Hands, Reserved\_Books. Между ними обеспечена целостность данных, а также каскадное обновление. При удалении данных в связанных таблицах соответствующие поля заполняются значениями NULL.

Таблица «User» содержит (рис. 9):

1. Id (Первичный ключ, Индекс, Автоинкремент), тип – Integer, размер – 255;
2. Surname, тип – Varchar, размер – 30;
3. Name, тип – Varchar, размер – 40;
4. Patronymic, тип – Varchar, размер – 30;
5. Gender, тип – Varchar, размер – 1;
6. Phone\_number, тип – Integer, размер – 255;
7. Birth\_date, тип – Date;
8. Email, тип – Varchar, размер – 255;
9. UserID, тип – Varchar, размер – 255;
10. Password, тип – Varchar, размер – 255.

В данном контексте UserID является полем логина пользователя.

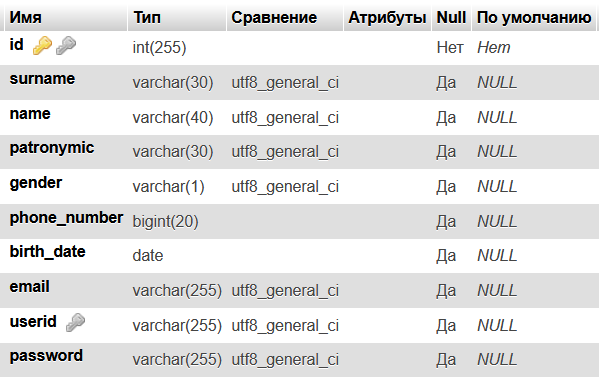


Рис. 9. Таблица «User»

Таблица «Event» содержит (рис. 10):

1. Id (Первичный ключ, Индекс. Автоинкремент), тип – Integer, размер – 255;
2. Header, тип – Varchar, размер – 255;
3. Visitors, тип – Integer, размер – 10;
4. Place, тип – Varchar, размер – 255;
5. Date, тип – Datetime;
6. Age, тип – Integer, размер – 3;
7. Description, тип – Varchar, размер – 3000;
8. Short\_description, тип – Varchar, размер – 255;
9. Cover, тип – Integer, размер – 255.

В данном случае Cover представлен типом Integer ввиду соответствующего названия в папке, в которой находятся изображения с одинаковым графическим типом файлов.



Рис. 10. Таблица «Event»

Таблица «Book» содержит (рис. 11):

1. Id (Первичный ключ, Индекс, Автоинкремент), тип – Integer, размер – 255;
2. Title, тип – Varchar, размер – 100;
3. Author, тип – Varchar, размер – 26;
4. Cover, тип – Integer, размер – 5;
5. Theme, тип – Varchar, размер – 47;
6. Description, тип – Varchar, размер – 1000;
7. Date, тип – Date.



Рис. 11. Таблица «Book»

Таблица «User\_Event» содержит (рис. 12):

1. Id\_User (Первичный ключ, Индекс), тип – Integer, размер – 255;
2. Id\_Event (Первичный ключ, Индекс), тип – Integer, размер – 255;
3. Datetime\_Added, тип – Datetime.

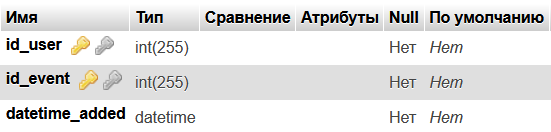


Рис. 12. Таблица «User\_Event»

Таблица «Wishlist\_Books» содержит (рис. 13):

1. Id\_User (Первичный ключ, Индекс), тип – Integer, размер – 255;
2. Id\_Book (Первичный ключ, Индекс), тип – Integer, размер – 255;
3. Datetime\_Added, тип – Datetime.

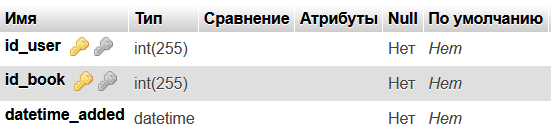


Рис. 13. Таблица «Wishlist\_Books»

Таблица «Books\_On\_Hands» содержит (рис. 14):

1. Id\_User (Первичный ключ, Индекс), тип – Integer, размер – 255;
2. Id\_Book (Первичный ключ, Индекс), тип – Integer, размер – 255;
3. Datetime\_Added, тип – Datetime.

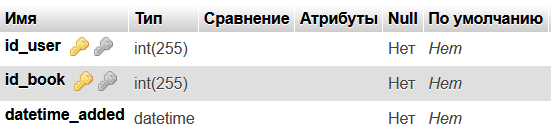


Рис. 14. Таблица «Books\_On\_Hands»

Таблица «Reserved\_Books» содержит (рис. 15):

1. Id\_User (Первичный ключ, Индекс), тип – Integer, размер – 255;
2. Id\_Book (Первичный ключ, Индекс), тип – Integer, размер – 255;
3. Datetime\_Added, тип – Datetime.

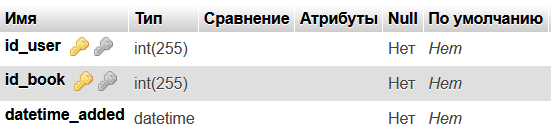


Рис. 15. Таблица «Reserved\_Books»

Cущность User имеет связь «1:∞» с сущностями: User\_Event, Wishlist\_Books, Books\_On\_Hands, Reserved\_Books по внешнему ключу Id\_User в соответствующих таблицах.

Сущность Book имеет связь «1:∞» с сущностями: Wishlist\_Books, Books\_On\_Hands, Reserved\_Books по внешнему ключу Id\_Book в соответствующих таблицах.

Сущность Event имеет связь «1:∞» с сущностью User\_Event, по внешнему ключу Id\_Event в соответствующей таблице.

Схема данных представлена на рис. 16.

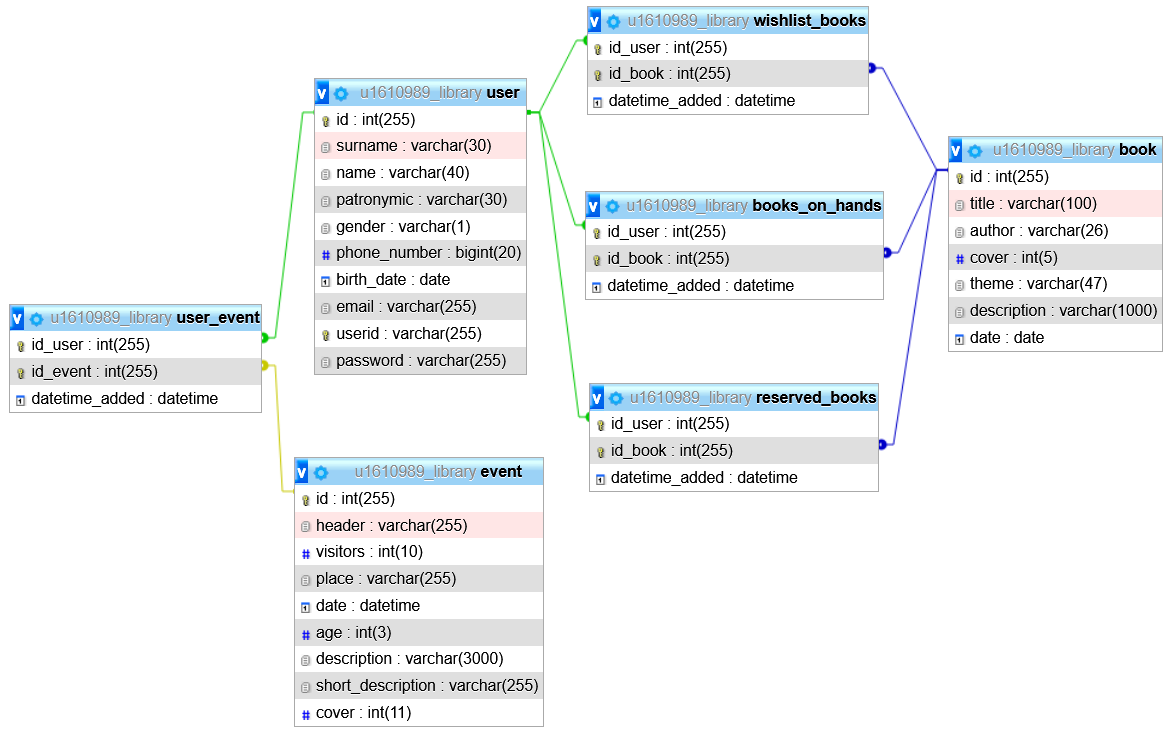


Рис. 16. Схема данных

# Глава 3. Разработка программного обеспечения

## Инструменты, используемые при создании программного обеспечения

Программное обеспечение приложения «Библиотека» можно разделить на две части: клиентская и серверная.

Клиентская часть состоит из приложения для сотрудников библиотеки и приложения для посетителей библиотеки.

Приложение для сотрудников библиотеки позволяет просматривать список мероприятий и их атрибуты, в том числе список посетителей, записавшихся на мероприятия, записывать вручную посетителей на мероприятия. Также приложение для сотрудников библиотеки позволяет создавать, изменять и удалять информацию о книгах, пользователях и вносить изменения в список «Книги на руках», в котором содержатся взятые посетителем книги в библиотеке. На данный момент приложение для сотрудников находится на стадии планирования.

Приложение для посетителей библиотеки позволяет зарегистрироваться и авторизоваться в библиотеке, просмотреть книги, имеющиеся в библиотечной системе, произвести поиск по атрибутам книги: автор, название, тема; добавить книгу в список желаемого, забронировать книгу в библиотеке, просмотреть списки книг: «книги на руках», «забронированные книги», «список желаемого». Есть возможность редактирования книг в списке забронированного и списке желаемого. На данный момент: 14.06.2022 приложение для посетителей разработано в концепции MVP (англ. minimum viable product - минимально жизнеспособный продукт, обладающий минимальными, но достаточными для удовлетворения первых потребителей функциями. Основная задача — получение обратной связи для формирования гипотез дальнейшего развития продукта) [2].

Клиентская часть создана с помощью среды разработки Android Studio. Это интегрированная среда разработки производства Google, с помощью которой разработчикам становятся доступны инструменты для создания приложений на платформе Android. Android Studio создавалась на базе IntelliJ IDEA. Программный код реализован на языке Java.

Серверная часть состоит из базы данных, созданной в СУБД phpMyAdmin и API (англ. Application Programming Interface — программный интерфейс приложения) [2] на языке PHP, который необходим для обмена информацией между базой данных на сервере и мобильным приложением. Работа с базой данных осуществляется на языке SQL (англ. Structured Query Language — язык структурированных запросов) [2].

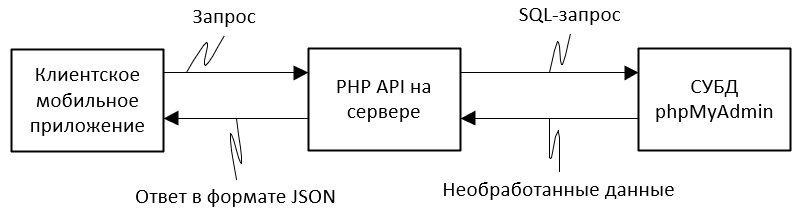


Рис. 17. Структура работы архитектуры приложения

## Пользовательский интерфейс и тестирование приложения

Пользовательский интерфейс представлен **layout** файлом (рус. Макетом), который создаётся, используя язык разметки XML. Макет определяет структуру пользовательского интерфейса экрана. Все элементы макета построены с использованием иерархии объектов **View** и **ViewGroup**. **View** – компонент, который пользователь может видеть и взаимодействовать с ним. В то время как **ViewGroup** является невидимым контейнером, который определяет структуру макета. [3]

При разработке пользовательского интерфейса использовались такие визуальные компоненты, как:

* Linear Layout – компоновка с линейным способом отображения элементов на экране. Используется для последовательного отображения элементов «один за другим». Относится к ViewGroup [4];
* Constraint Layout – нелинейная компоновка отображения элементов на экране. Используется для произвольного построения интерфейсов с возможностью наложения одного элемента на другой. Относится к ViewGroup [4];
* Text View – View-элемент, отображающий текст без возможности изменения [4];
* Recycler View – ViewGroup-элемент, который отображает набор данных в виде списка [4];
* Progress Bar – View-элемент, указывающий на статус выполнения операции [4];
* Image View – View-элемент, который отображает ресурсы изображений, например Bitmap или Drawable [4];
* Card View – ViewGroup-элемент, в котором можно отобразить любые данные в формате карточки с закругленными углами;
* Scroll View – ViewGroup-элемент, который позволяет прокручивать размещенные в ней элементы;
* Button – View-элемент, кнопка;
* Edit Text – View-элемент, представляющий собой изменяемое поле с текстом;
* Frame Layout – ViewGroup-элемент, цель которого была в данном приложении – расположение прочих ViewGroup с переключением через Bottom Navigation View;
* Bottom Navigation View – ViewGroup-элемент, в котором расположены кнопки;
* Spinner – выпадающий список, относится к View-элементам.
* Calendar View – View-элемент, виджет календаря с целью отображения и выбора элементов – дней [4];

Интерфейс Android приложения для посетителей состоит из главной активности – MainActivity, и трёх панелей, расположенных на ней:

- top\_header\_container – FrameLayout, на котором расположен верхний заголовок с датой слева и кнопкой профиля справа (см. рис. 18)



Рис. 18. Верхний заголовок клиентского приложения

- fragment\_container – FrameLayout, на котором динамически меняются визуальные данные, отображаемые приложением;

- bottom\_navigation – BottomNavigationView, на котором расположены кнопки для навигации по основным пунктам приложения (см. рис. 19).



Рис. 19. BottomNavigationView

Все три элемента создают основной вид приложения (см. рис. 18-23).

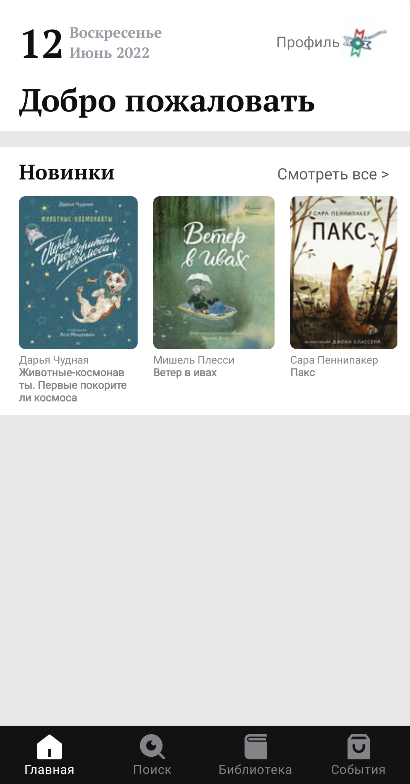


Рис. 20. Главная страница

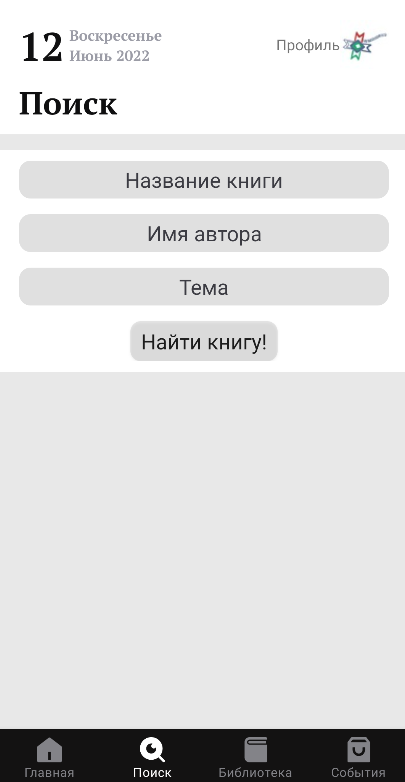


Рис. 21. Поиск книг по атрибутам

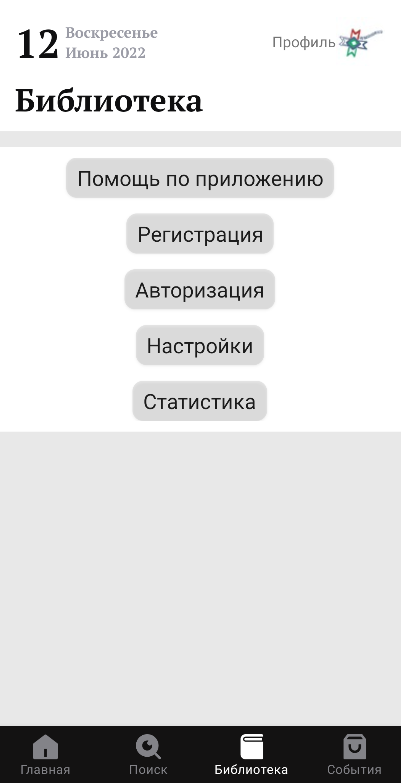


Рис. 22. Вкладка «Библиотека»

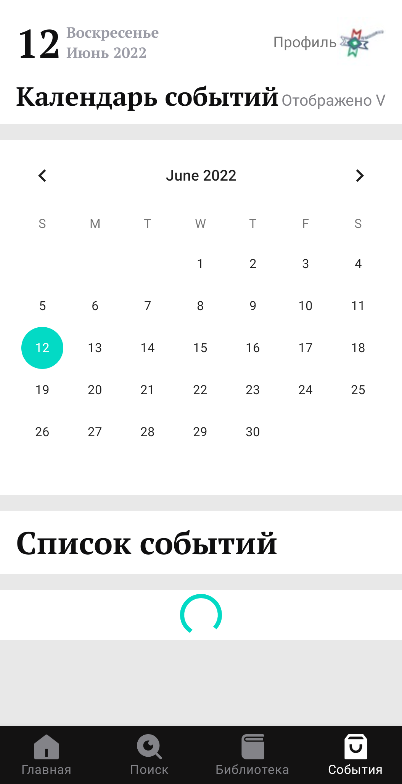


Рис. 23. События

При входе в приложение пользователь попадает на главную страницу (см. рис. 20), на которой может сразу увидеть список новых книг (RecyclerView в горизонтальной ориентации), добавленных в базу данных библиотеки (см. рис. 24). Этот список можно прокручивать слева направо, однако, он ограничен 8 элементами.

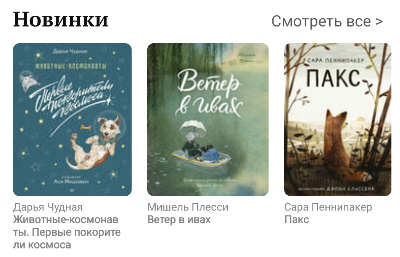
****

Рис. 24. Новые книги

При нажатии на TextView «Смотреть все» пользователь переходит на отдельную страницу с тем же списком новых книг (RecyclerView в вертикальной ориентации, см. рис. 25), на котором список реализован по логике «ленты» - записи отсортированы в хронологическом порядке от новых к старым, при достижении некоторого количества записей в RecyclerView добавляются новые данные, загруженные с сервера.

При нажатии на одну из книг в списке пользователь переходит на страницу с подробным описанием книги и двумя кнопками: «Добавить в список желаемого» и «Забронировать» (см. рис. 26). Только авторизованный в системе пользователь может воспользоваться функционалом данных кнопок.

Из вкладки «Поиск» (см. рис. 21), воспользовавшись поиском по атрибутам, можно попасть на следующую страницу с результатами поиска (запрос на тему «детск» будет искать соответственно SQL-запросу «select \* from book where theme like ‘%детск%’» - в результат попадут такие запросы, как «детская фантастика» или «детская художественная литература»), организованными аналогично в виде ленты (см. рис. 27).

Из вкладки «Библиотека» (см. рис. 22) можно по нажатию на кнопки «Регистрация» (см. рис. 28, рис. 29) или «Авторизация» (см. рис. 30) попасть в соответствующие окна, на которых можно произвести соответствующие действия по регистрации и авторизации в библиотечной системе, после которых будут доступны добавление книги в список желаемого, «забронированные»; просмотр списков книг, удаление книги из списка.

При нажатии на кнопку «подтвердить» в окне «Регистрация», если все поля заполнены верно, и пользователь с таким email ещё не был зарегистрирован, то на почту пользователя отправляются данные с временным паролем и уникальным id пользователя, по которым он может войти в систему.

На вкладке «Авторизация» пользователь по данным, пришедшим на почту (id пользователя и пароль, либо email пользователя и пароль) может авторизоваться в библиотечной системе.

На вкладке «События» пользователь может просмотреть список всех событий в хронологическом порядке начиная с текущего дня в формате ленты, а также может просмотреть список событий за определенный день, нажав на соответствующий день на CalendarView.

На вкладке «Профиль» (см. рис. 33, см. рис. 34) авторизованный пользователь может посмотреть списки книг: «Книги на руках», «Забронированные», «Список желаемого»; а также удалить книги из последних двух списков.

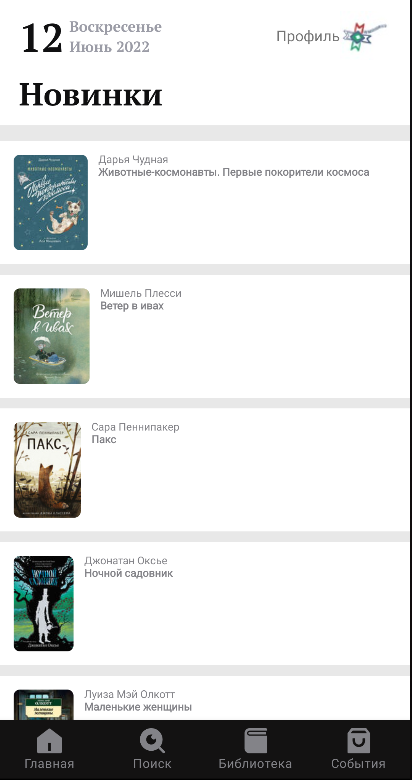


Рис. 25. Лента с записями книг

****

Рис. 26. Информация о книге

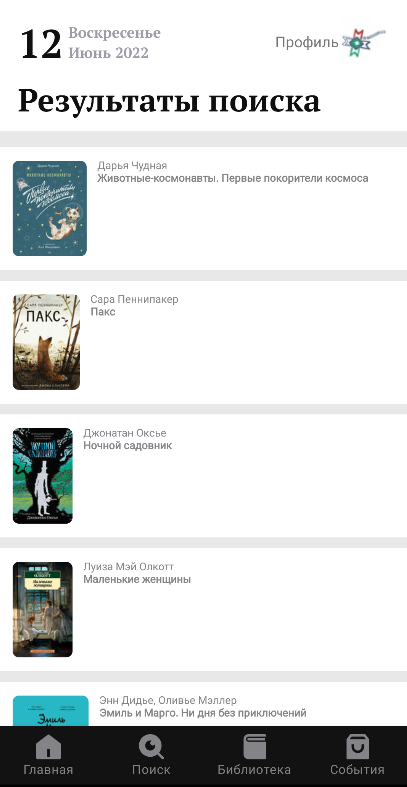


Рис. 27. Результаты поиска по запросу «Детск»

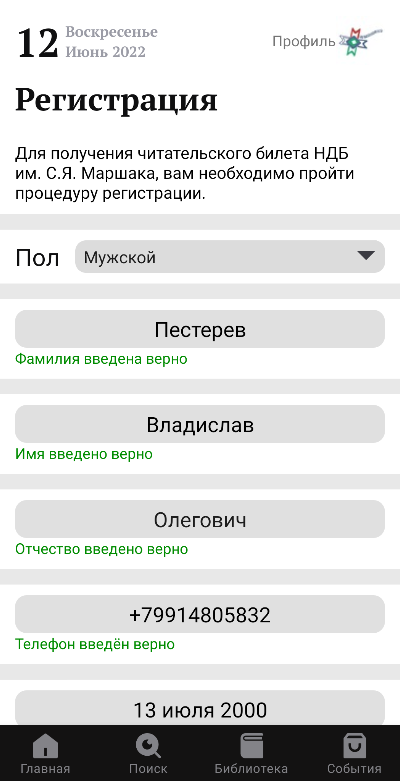


Рис. 28. Регистрация. Часть 1

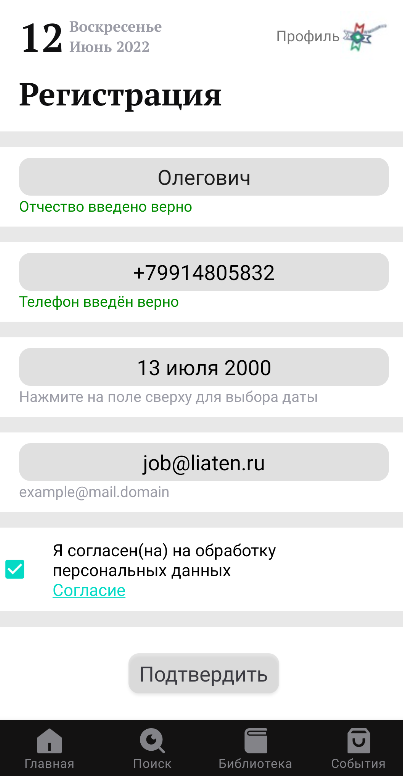


Рис. 29. Регистрация. Часть 2

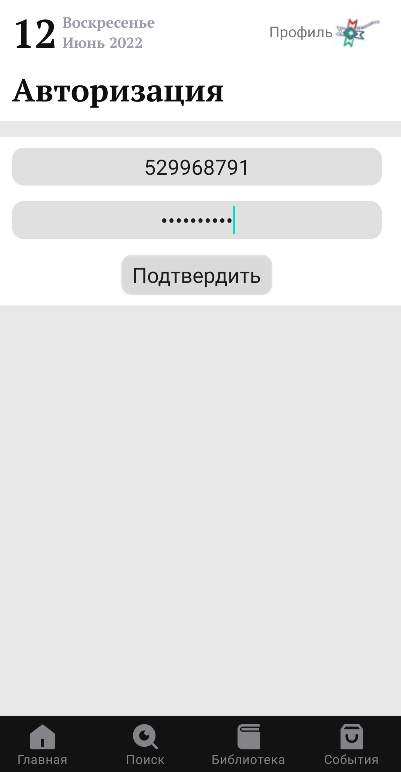


Рис. 30. Авторизация

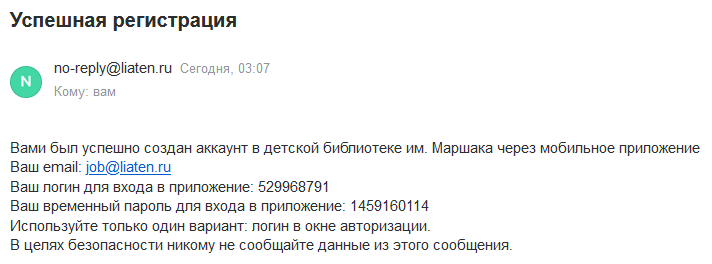


Рис. 31. Сообщение, приходящее на почту после регистрации

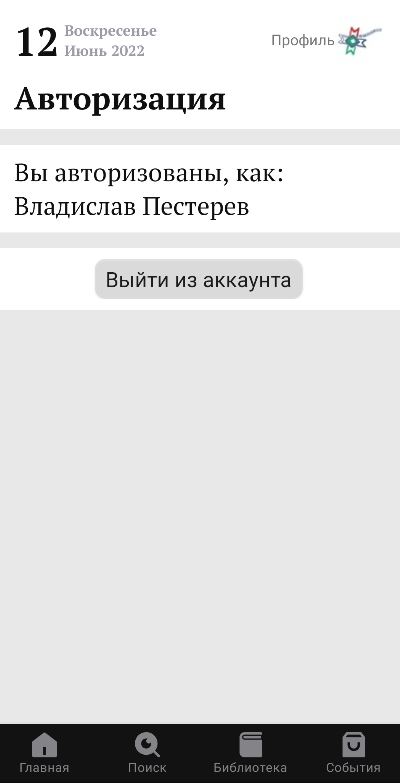


Рис. 32. Успешная авторизация в системе

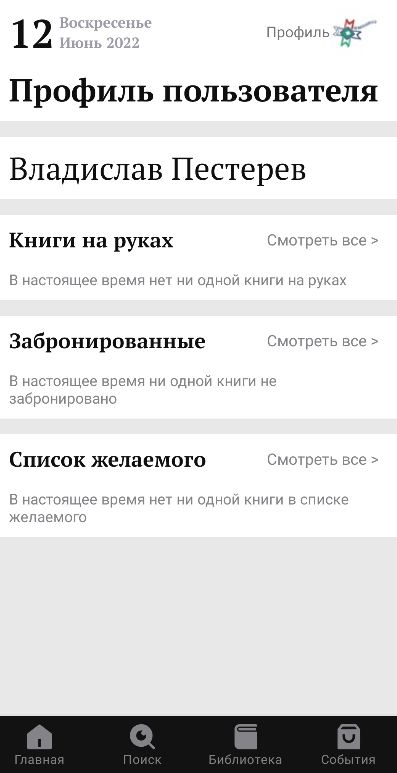


Рис. 33. Профиль c пустыми списками

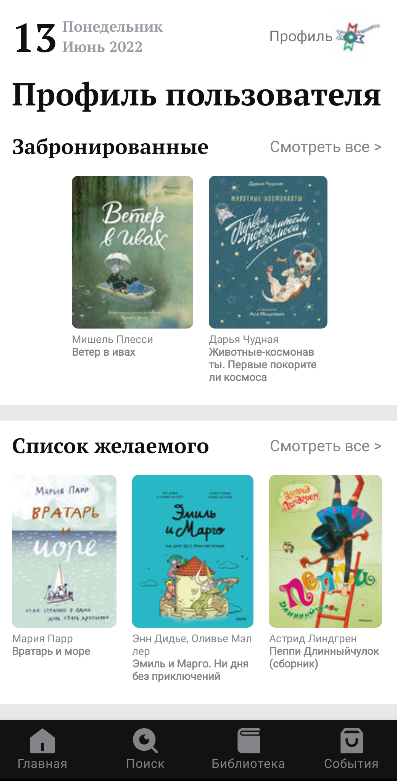


Рис. 34. Профиль с наполненными списками

## Программная реализация

Приложение состоит из базы данных на сервере и мобильного приложения на устройствах пользователей. Основная работа программы заключается в передаче данных между сервером и приложением через API на PHP. Так, мобильное приложение отправляет запрос и получает ответ в текстовом формате обмена данными JSON (JavaScript Object Notation) от сервера, который уже затем обрабатывается клиентом для вывода в приложении пользователя.

В мобильном приложении делается запрос к базе данных. Для этого происходит подключение к файлу, создаётся поток вывода из приложения, в который направляются данные.

Рассмотрим на примере загрузки данных о новых книгах по URL (см. рис. 35-46).

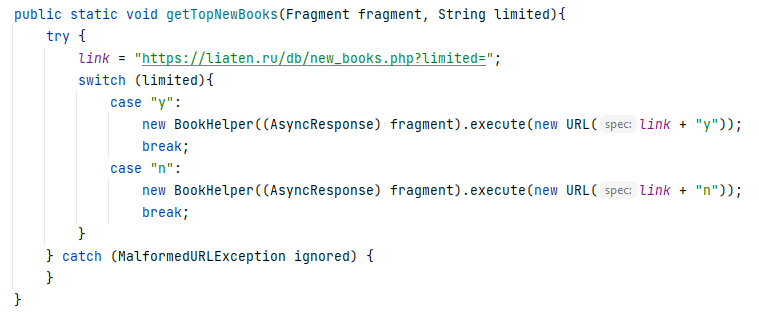


Рис. 35. Функция в классе HomeFragment, вызывающая класс BookHelper, который является AsyncTask (используется для соединения с сетью) и который возвращает список книг по URL, переданному в аргументе

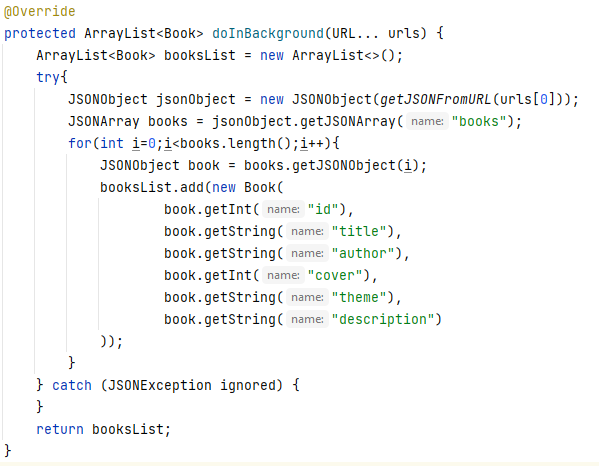


Рис. 36. Формирование списка книг из JSON-объекта, полученного методом getJSONFromURL(urls[0])



Рис. 37. Метод получения JSON по URL адресу в формате String

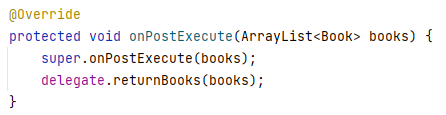


Рис. 38. Возвращение списка книг в класс HomeFragment

Рис. 39. Обработка списка книг в классе HomeFragment и вызов класса ImageDownloader (является AsyncTask) для загрузки изображения, расположенного по некоторому пути на сервере

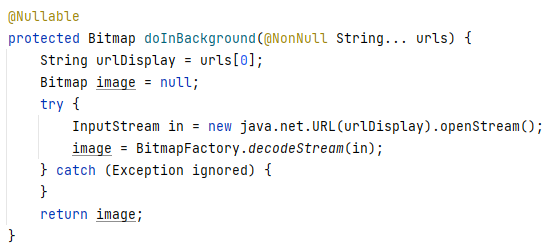


Рис. 40. Код, выполняющийся в ходе запуска класса ImageDownloader

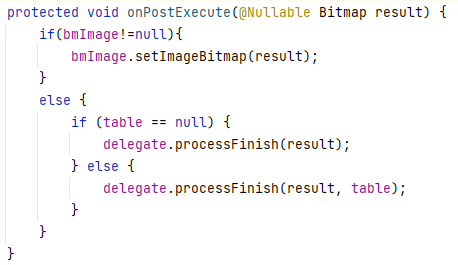


Рис. 41. После загрузки изображения возвращается изображение в формате Bitmap



Рис. 42. Обработка полученного изображения в классе HomeFragment

Ввиду того, что изображения загружаются в последнюю очередь, был создан класс ListWaiter (расширяет класс Thread), в котором при запуске идёт проверка, все ли изображения были загружены условием проверки размера списка изображений и размера возвращенного списка.



Рис. 43. Исполняемый код класса ListWaiter

Если размер возвращенного списка не больше списка обложек книг, тогда вызывается класс RecyclerInitializer, в котором происходит вызов RecyclerViewAdapter для заполнения RecyclerView полученными данными с сервера.

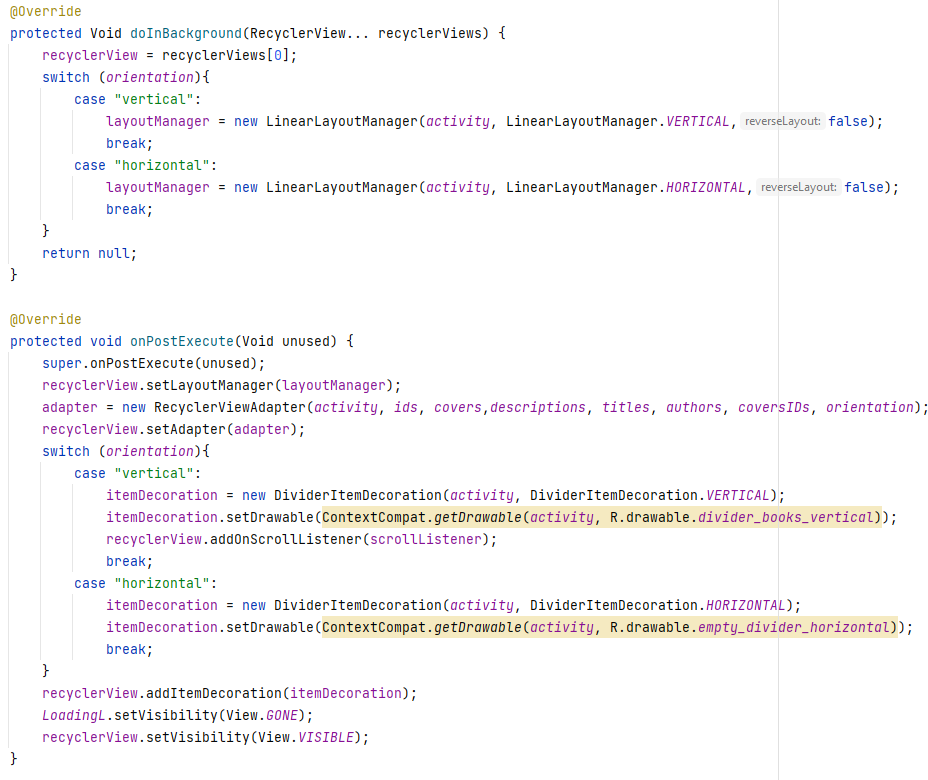


Рис. 44. Исполняемый код RecyclerInitializer

Пример работы класса RecyclerViewAdapter представлен ниже:

@Override  
public void onBindViewHolder(@NonNull ViewHolder holder, int position) {  
 int coverWidth = 0;  
 int coverHeight = 0;  
 try {  
 coverWidth = covers.get(position).getIntrinsicWidth();  
 coverHeight = covers.get(position).getIntrinsicHeight();  
 }  
 catch (IndexOutOfBoundsException e){  
 e.printStackTrace();  
 }  
 double pixelsWidth = coverWidth \* *scale* + 0.5f;  
 double pixelsHeight = coverHeight \* *scale* + 0.5f;  
 double coefficient = 0;  
 switch (*orientation*){  
 case "horizontal":  
 coefficient = pixelsHeight / 400;  
 break;  
 case "vertical":  
 coefficient = pixelsHeight / 250;  
 break;  
 }  
 pixelsWidth /= coefficient;  
 pixelsHeight /= coefficient;  
 FrameLayout.LayoutParams lp = new FrameLayout.LayoutParams((int) pixelsWidth, (int) pixelsHeight);  
 holder.cover.setLayoutParams(lp);  
 String author = setHyphenation(authors.get(position));  
 String title = setHyphenation(titles.get(position));  
 Spanned sp = Html.*fromHtml*(author + "<br><b>" + title + "</b>");  
 holder.title.setText(sp);  
 try {  
 holder.cover.setImageDrawable(covers.get(position));  
 }  
 catch (IndexOutOfBoundsException e){  
 e.printStackTrace();  
 }  
 holder.layout.setOnClickListener(view -> {  
 new FragmentHelper((MainActivity) context,  
 false, true).execute(new BookInfo(  
 ids.get(position),  
 titles.get(position),  
 authors.get(position),  
 descriptions.get(position),  
 coversIDs.get(position)  
 ));  
 });  
}

Здесь ViewHolder является некоторый объект (см. рис. 45). Этими визуальными объектами будет заполняться RecyclerView в ходе работы класса RecyclerViewAdapter.

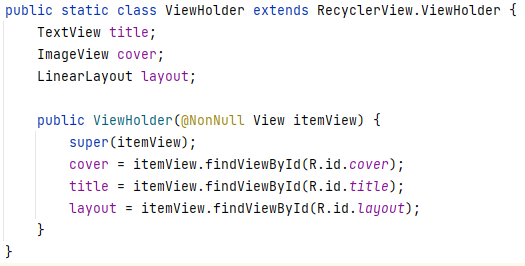


Рис. 45. Класс ViewHolder

На рис. 46 отображен исполняемый на сервере код на языке PHP, который возвращает данные из СУБД phpMyAdmin в JSON-формате.

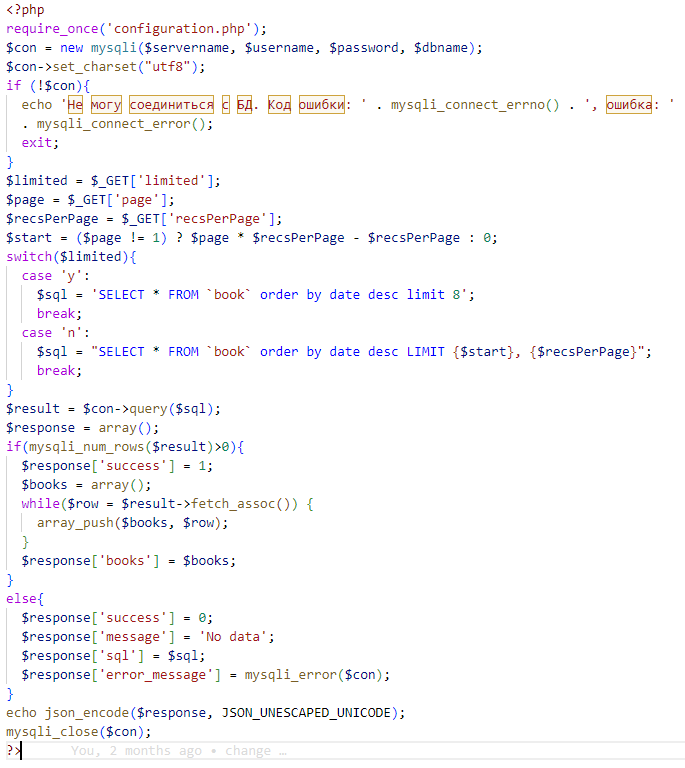


Рис. 46. Реализация API на языке PHP для получения клиентским приложением данных о новых книгах

# Заключение

В результате исследования было разработано мобильное приложение «Библиотека» для посетителей библиотеки в концепции MVP на платформе Android. Были созданы самые существенные функции приложения – просмотр книг из базы данных, регистрация и авторизация в системе, просмотр списков «На руках», «Список желаемого», «Забронированные», возможность бронирования книг и добавления книг в список желаемого.

База данных была создана в СУБД phpMyAdmin. Передача данных осуществлялась между приложением и сервером посредством API на языке PHP.

В будущем планируется дальнейшая доработка программы, повышение скорости загрузки списков книг, обложек; добавление такого функционала, как: события и все взаимодействия с событиями; возможность записи на событие; уведомления: о сроке сдачи книги, о просроченных книгах, о предстоящих событиях в библиотеке; изменение пользователем пароля; создание пользователем произвольного логина, добавление произвольного аватара пользователя.

Также планируется миграция базы данных с СУБД phpMyAdmin на СУБД Opac Global.

# Список литературы

1. Методология SADT и стандарты IDEF [Электронный ресурс] Режим доступа: <https://smekni.com/a/189592/metodologiya-sadt-i-standarty-idef/> (Дата обращения: 3.06.2021);
2. «Википедия» [Электронный ресурс] – Режим доступа: <https://ru.wikipedia.org/> (Дата обращения: 13.06.2022);
3. «Пользовательский интерфейс. View, основные атрибуты» [Электронный ресурс] - Режим доступа: <https://devcolibri.com/unit/пользовательский-интерфейс-view-основн/> (Дата обращения: 13.06.2022);
4. Android Developers [Электронный ресурс] – Режим доступа: <https://developer.android.com/guide/topics/ui/> (Дата обращения: 13.06.2022);
5. Бабенко, В.В. Практический анализ бизнес-процессов / В.В. Бабенко. — Сыктывкар: Изд-во Сыктывкарского ун-та, 2010. 290 с.;
6. Методика функционального моделирования SADT// Студопедия — поиск обучающей информации. [Электронный ресурс] — Режим доступа: <https://studopedia.info/1-75613.html> (Дата обращения: 13.06.2022).