МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
“ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”

Факультет компьютерных наук

Кафедра технологий обработки и защиты информации

Система управления домашним бюджетом с рекомендациями по сокращению расходов.

Курсовая работа по дисциплине «Технологии программирования»

09.03.02 *Информационные системы и технологии*

Обработка информации и машинное обучение

Преподаватель _____В.С. Тарасов, ст. преподаватель __.__20__

Обучающийся _____А.А. Лазуткина, 3 курс, д/о

Обучающийся _____В.И. Гараба, 3 курс, д/о

Обучающийся _____А.М. Трунова, 3 курс, д/о

Обучающийся _____Я.А. Рощупкин, 3 курс, д/о

Руководитель _____В.С. Зенин, преподаватель

Воронеж 2023

Содержание

Введение

В наше время мир стремительно меняется, и в условиях изменений люди склонны уделять особенное внимание поиску финансовой стабильности. Исследования Аналитического центра НАФИ показывают, что уровень финансовой грамотности россиян за последнее десятилетие значительно вырос. Согласно статистике, около половины россиян (51%) так или иначе планируют свой бюджет. Из них 32% ведут учет в «уме», опираясь на данные из современных банковских приложений, а оставшиеся 29% россиян записывают расходы и доходы вручную или используют специальное программное обеспечение [1]. Это свидетельствует о том, что люди ищут способы сделать процесс ведения бюджета более удобным и доступным.

Именно здесь могут прийти на помощь мобильные приложения для ведения бюджета. Множество функций позволяют контролировать свои траты, следить за достижением финансовых целей, планировать свой бюджет и отслеживать динамику доходов. Кроме того, формат мобильного приложения не требует бумаги с ручкой или доступа к компьютеру, что облегчает учет финансов.

Таким образом, мобильное приложение для ведения бюджета – это простое и удобное решение для организации финансовой сферы жизни.

1 Постановка задачи

Целью данного курсового проекта является разработка мобильного приложения для учета доходов и расходов, планирования трат и накоплений, отслеживания статистики доходов и расходов по категориям и анализа своего бюджета.

1.1 Требования к разрабатываемой системе

К разрабатываемой системе предъявляются следующие требования:

- должна обеспечиваться безопасность баз данных и защита от несанкционированного удаления данных;
- приложение должно устанавливаться и работать на мобильных устройствах под управлением операционной системы Android 10-12, имеющих доступ к сети Интернет;
- приложение должно иметь трёхуровневую архитектуру, состоящую из клиентской части, серверной части и сервера базы данных.

1.2 Задачи, решаемые в процессе разработки

В процессе разработки решаются следующие задачи:

- проектирование приложения средствами языка UML;
- разработка серверной части, которая включает в себя:
    1. реализацию ролей неавторизованного пользователя, авторизованного пользователя и администратора;
    2. реализацию функциональных возможностей ролей;
    3. разработку базы данных;
- разработка клиентской части, которая включает в себя:
    1. создание макета дизайна;
    2. реализацию макета дизайна;
- проведение тестирования проекта.

2 Анализ предметной области

2.1 Терминология (глоссарий) предметной области

Мобильное приложение – программное обеспечение, предназначенное для работы на смартфонах, планшетах и других мобильных устройствах, разработанное для конкретной платформы.

Трёхуровневая архитектура – архитектурная модель программного комплекса, предполагающая наличие в нём трёх компонентов: клиента, сервера приложений и сервера баз данных.

Клиент – интерфейсный (обычно графический) компонент, который представляет приложение для конечного пользователя.

Сервер – это компьютер или программа, которая предоставляет определенные услуги или ресурсы для других компьютеров или программ (клиентов) через сеть.

База данных – упорядоченный набор структурированной информации или данных, которые обычно хранятся в электронном виде в компьютерной системе. База данных обычно управляется системой управления базами данных (СУБД).

СУБД – система управления базами данных, комплекс программно-языковых средств, позволяющих создать базы данных и управлять данными.

Android – операционная система для мобильных устройств.

Бюджет – финансовый план, состоящий из доходов и расходов.

Доходы – деньги или материальные ценности, получаемые от предприятия, отдельного лица или какого-либо вида деятельности.

Расходы – затраты, издержки, потребление чего-либо для определенных целей.

Финансовая цель – определенная сумма денег, которая требуется для достижения некоторой материальной или нематериальной цели.

Счет – виртуальный «кошелек», с которым можно совершать различные операции: добавление или снятие суммы, установление лимита или финансовой цели.

Операция – добавление дохода к сумме на счету или вычет суммы расхода из суммы на счету.

Пользователь – лицо, которое использует действующую систему для выполнения конкретной функции.

Авторизация – предоставление определённому лицу или группе лиц прав на выполнение определенных действий, а также процесс проверки данных прав при попытке выполнения этих действий.

Регистрация – действия, направленные на создание личной учетной записи в приложении, с целью получения доступа к его полному функционалу.

Model-View-Controller (MVC) – архитектурный паттерн, который определяет разделение приложения на три компонента: модель, представление и контроллер. Модель отвечает за хранение данных, представление – за отображение этих данных пользователю, а контроллер – за управление взаимодействием между моделью и представлением.

Model-View-ViewModel (MVVM) – архитектурный паттерн, который расширяет концепцию MVC, добавляя компонент ViewModel, который отвечает за логику приложения и управление взаимодействием между моделью и представлением. ViewModel предоставляет данные и команды, которые используются представлением для отображения информации пользователю, а также обрабатывает пользовательский ввод и взаимодействие с моделью.

PostgreSQL – свободная объектно-реляционная система управления базами данных.

Java – строго типизированный объектно-ориентированный язык программирования общего назначения, который имеет множество инструментов и библиотек для обработки данных, создания графического интерфейса и управления потоком выполнения программы.

Spring Boot – популярный фреймворк для создания веб-приложений с использованием Java, который облегчает разработку, настройку и развертывание приложений. Он предоставляет множество инструментов и

библиотек для работы с базами данных, безопасности, тестирования и многого другого.

Kotlin – статически типизированный, объектно-ориентированный язык программирования, работающий поверх Java Virtual Machine и разрабатываемый компанией JetBrains.

Android SDK – универсальное средство разработки мобильных приложений для операционной системы Android. Android SDK включает в себя Android Studio – интегрированную среду разработки приложений, которая предоставляет мощный набор инструментов, в том числе средства для создания пользовательского интерфейса, отладки, тестирования и профилирования приложений.

2.2 Цели создания приложения

К целям создания приложения относятся:

— помощь пользователю в достижении своих финансовых целей;

— сбор анонимной статистики расходов пользователей по категориям.

2.3 Сфера применения

Приложение для ведения домашнего бюджета может использоваться как для ведения личного или домашнего бюджета, так и для управления финансами малого бизнеса. Оно помогает отслеживать доходы и расходы, контролировать бюджет, планировать расходы на будущее и принимать решения на основе финансовой информации.

2.4 Технический обзор

Приложение реализует следующие функции:

— учет доходов и расходов: пользователь может добавлять свои доходы и расходы в приложение, чтобы отслеживать динамику бюджета;

— категоризация расходов: приложение предоставляет пользователю возможность назначать категории своих расходов;

— анализ финансовой информации: приложение предоставляет графики со статистикой трат и пополнений, чтобы пользователь мог анализировать свои финансы.

2.5 Обзор аналогов

Существует множество приложений для ведения домашнего бюджета, и каждое из них имеет свои преимущества и недостатки. Рассмотрим несколько популярных аналогов.

2.5.1 Приложение «Wallet»

«Wallet» – это бесплатное приложение для контроля расходов и планирования личного бюджета (рис. 1).

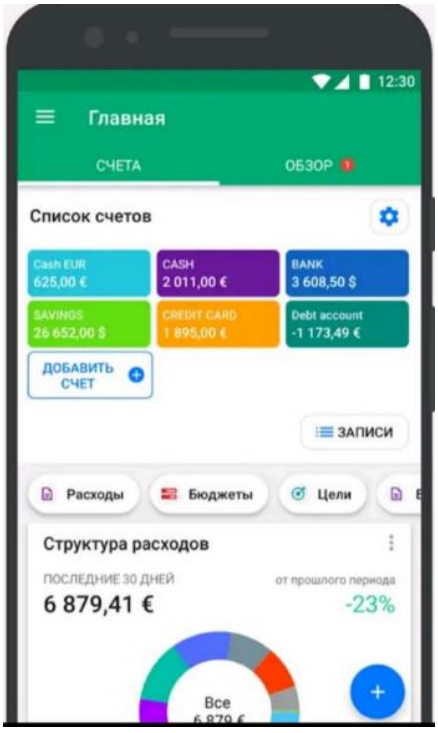

Рисунок 1 - Интерфейс приложения «Wallet»

Преимущества приложения:

— возможность подключения банковских счетов;

— количество счетов не ограничено;

— высокий уровень безопасности;

— облачная синхронизация.

Слабые стороны:

— нет возможности установить цели;

— нет суммы трат за день;

— нет вкладки с учетом долгов.

2.5.2 Приложение «PocketGuard»

«PocketGuard» – это еще одно бесплатное приложение, которое позволяет составлять бюджет, отслеживать и снижать свои расходы (рис. 2).

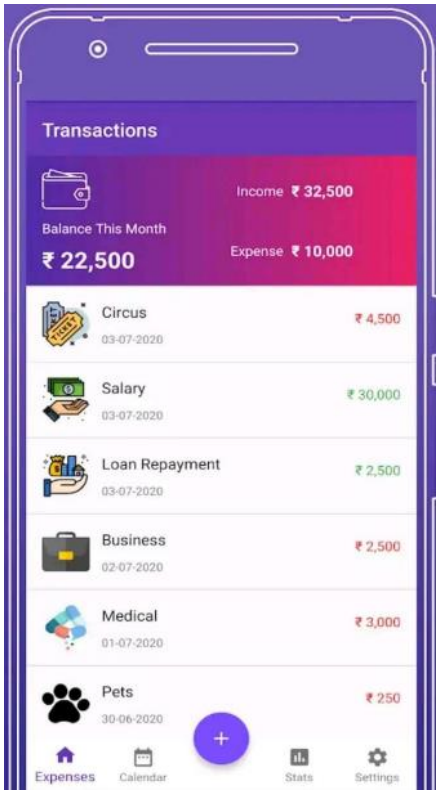

Рисунок 2 - Интерфейс приложения «PocketGuard»

Преимущества приложения «PocketGuard»:

— возможность подключения банковских счетов;

— напоминания о платежах;

— возможность устанавливать финансовые цели;

— алгоритм, рассчитывающий стратегию погашения долга;

— высокий уровень безопасности.

Слабые стороны:

— частое возникновение ошибок;

— нет годовой суммы трат.

2.5.3 Приложение «Финансы»

«Финансы» – бесплатное приложение для контроля финансов (рис. 3).

Преимущества приложения «Финансы»:

— простой и интуитивно понятный дизайн;

— возможность добавить свои категории;

— возможность добавления множества счетов.

Слабые стороны:

— частые ошибки при синхронизации;

— нет вкладки с доходами и расходами одновременно;

— присутствие рекламы;

— неудобная навигация;

— нет функции планирования расходов.

Рисунок 3 - Интерфейс приложения «Финансы»

Таким образом, каждое из этих приложений имеет свои преимущества и недостатки. Рассмотрев их, можно сделать вывод, что приложение для управления бюджетом должно быть удобным в использовании, иметь

возможность синхронизации с другими устройствами, а также предоставлять достаточное количество функций.

2.6 Требования к функциональности

Приложение должно реализовывать следующую функциональность:

— добавление доходов и расходов, разделение их на категории для удобства отслеживания динамики трат и пополнений;

— вывод графиков доходов и расходов для контроля финансов;

— установление лимитов на определённые категории;

— возможность установки финансовой цели;

— возможность добавления своих счетов;

— возможность совершить регистрацию/авторизацию в системе;

— просмотр истории операций за день или месяц.

2.7 Пользователи системы

В системе существуют такие группы пользователей, как неавторизованный пользователь, авторизованный пользователь и администратор. Для неавторизованного пользователя должны быть предоставлены следующие функции:

— добавление одного счета;

— добавление расходов и доходов на счет;

— редактирование и удаление добавленной операции;

— указание категории трат или доходов при добавлении операции;

— редактирование категорий;

— установка лимитов на категории и счет;

— указание финансовой цели на счете;

— просмотр графиков доходов и расходов;

— просмотр истории операций за день или месяц;

— просмотр прогноза доходов и расходов на следующий месяц.

Для авторизованного пользователя:

— добавление нескольких счетов;

— редактирование и удаление добавленных счетов;

— добавление расходов и доходов на выбранный счет;

— редактирование и удаление добавленной операции;

— указание категории трат или доходов при добавлении операции;

— редактирование категорий;

— установка лимитов на категории и счета;

— просмотр графиков доходов и расходов;

— просмотр истории операций за день или месяц;

— просмотр прогноза финансов на следующий месяц.

Для администратора:

— просмотр анонимной статистики количества пользователей по их ролям;

— просмотр анонимной статистики средних трат пользователей по категориям.

2.8 Требования, не касающиеся функциональной части

Требования к программному обеспечению: серверная часть приложения должна быть реализована согласно архитектурному паттерну MVC, а клиентская часть – согласно архитектурному паттерну MVVM. Также приложение должно устанавливаться и работать на любом устройстве под управлением операционной системы Android 10-12.

Требования к дизайну приложения: приложение должно быть выполнено в едином стиле. Цветовая палитра приложения должна содержать два основных цвета - #F9FAF4 для фона и #FFD166 для кнопок и навигации. Используются шрифты без засечек, всего не более 3 шрифтов. Для иконок используются распространенные обозначения.

3 Графическое описание работы системы

3.1 Диаграмма IDEF0

Рассмотрим основной бизнес-процесс на примере контекстной диаграммы, представленной на рисунке 4. Данная диаграмма представляет собой общее видение процесса работы приложения.

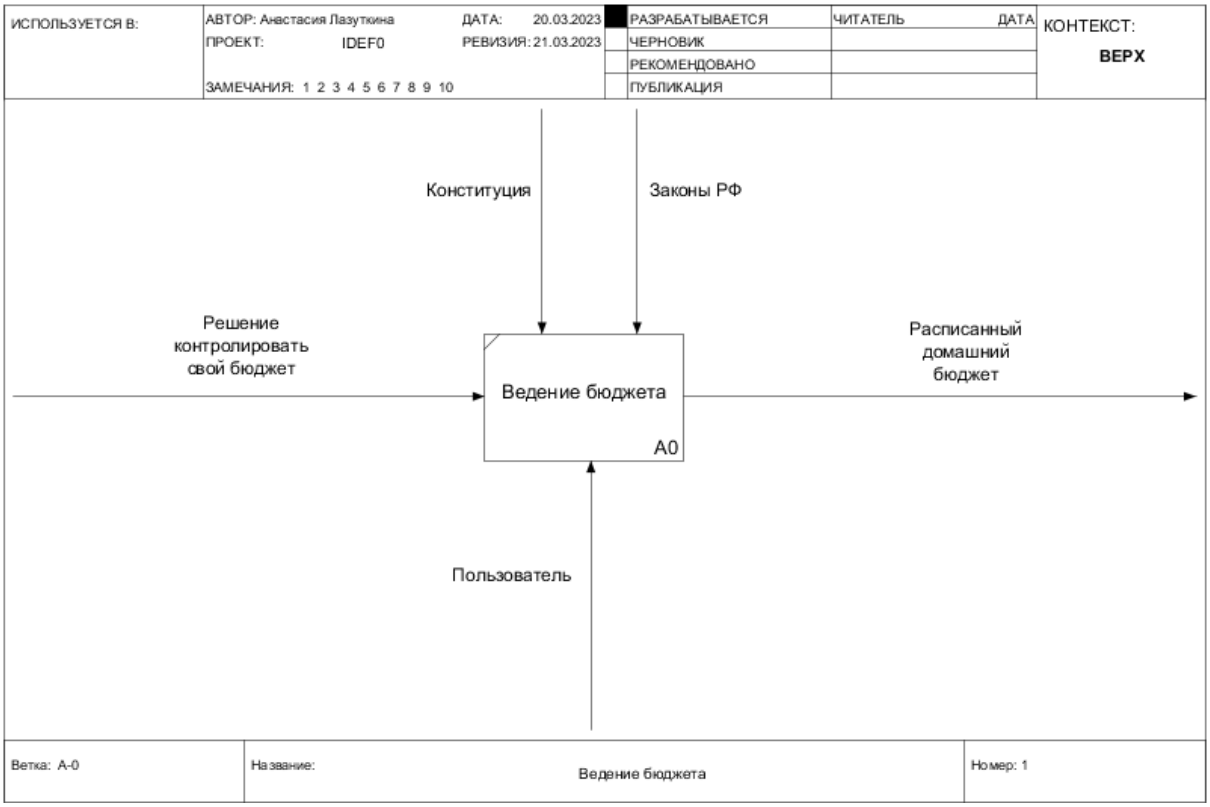

Рисунок 4 - Диаграмма IDEF0

3.2 Диаграммы прецедентов

Диаграммы прецедентов показывают действия актеров, то есть действующих лиц системы, по отношению к системе, а также их возможности.

3.2.1 Диаграмма прецедентов (авторизованный пользователь)

На данной диаграмме представлены сценарии взаимодействия авторизованного пользователя с приложением (рис. 5).

Рисунок 5 - Диаграмма прецедентов (авторизованный пользователь)

3.2.2 Диаграмма прецедентов (неавторизованный пользователь)

На данной диаграмме представлены сценарии взаимодействия неавторизованного пользователя с приложением (рис. 6).

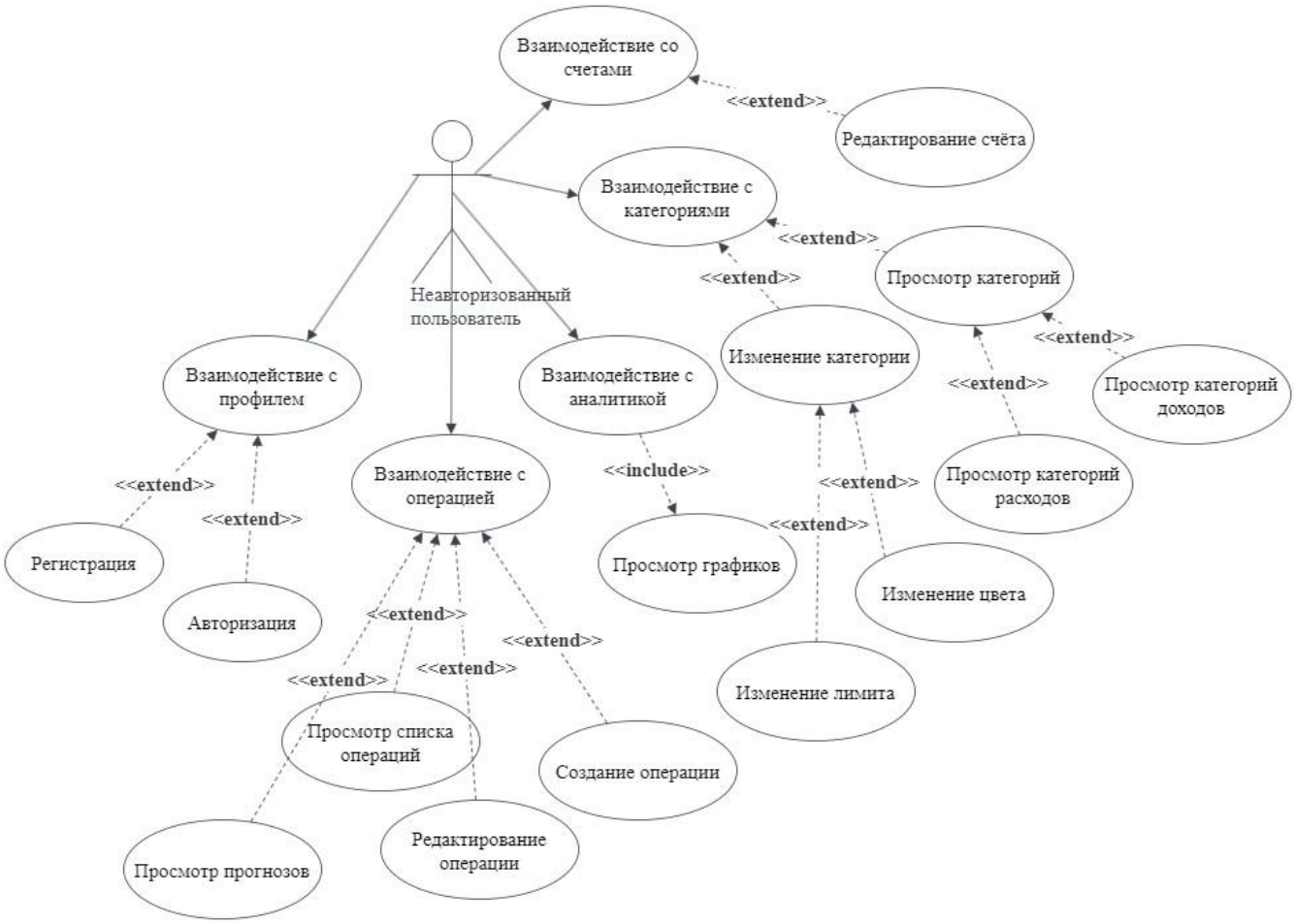

Рисунок 6 - Диаграмма прецедентов (неавторизованный пользователь)

3.2.3 Диаграмма прецедентов (администратор)

На данной диаграмме представлен сценарий взаимодействия администратора с приложением (рис. 7).

Рисунок 7 - Диаграмма прецедентов (администратор)

3.3 Диаграмма развёртывания

Диаграмма развертывания показывает, какие аппаратные компоненты существуют, какие программные компоненты работают на каждом узле, и как различные части этого комплекса соединяются друг с другом (рис. 8).

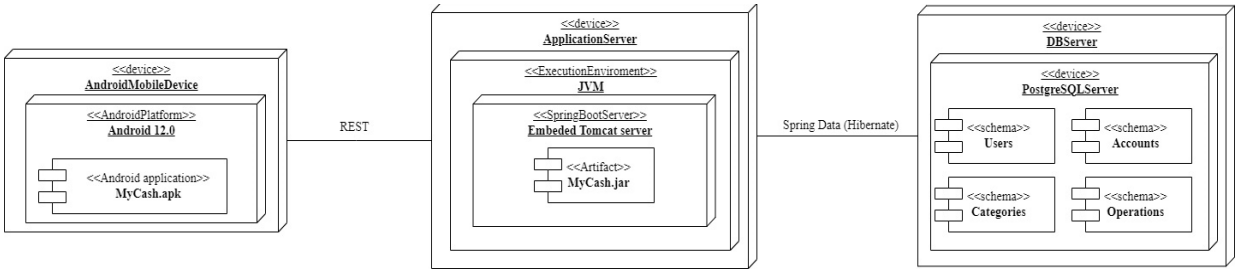

Рисунок 8 - Диаграмма развёртывания

3.4 Диаграмма состояний(пользователь)

Диаграмма состояний показывает, как объект переходит из одного состояния в другое. На рисунке 9 показано изменение состояний пользователя от первого входа в приложение до выхода. Пользователь может находится в следующих состояниях: незарегистрированный, зарегистрированный, неавторизованный, авторизованный, удалённый.

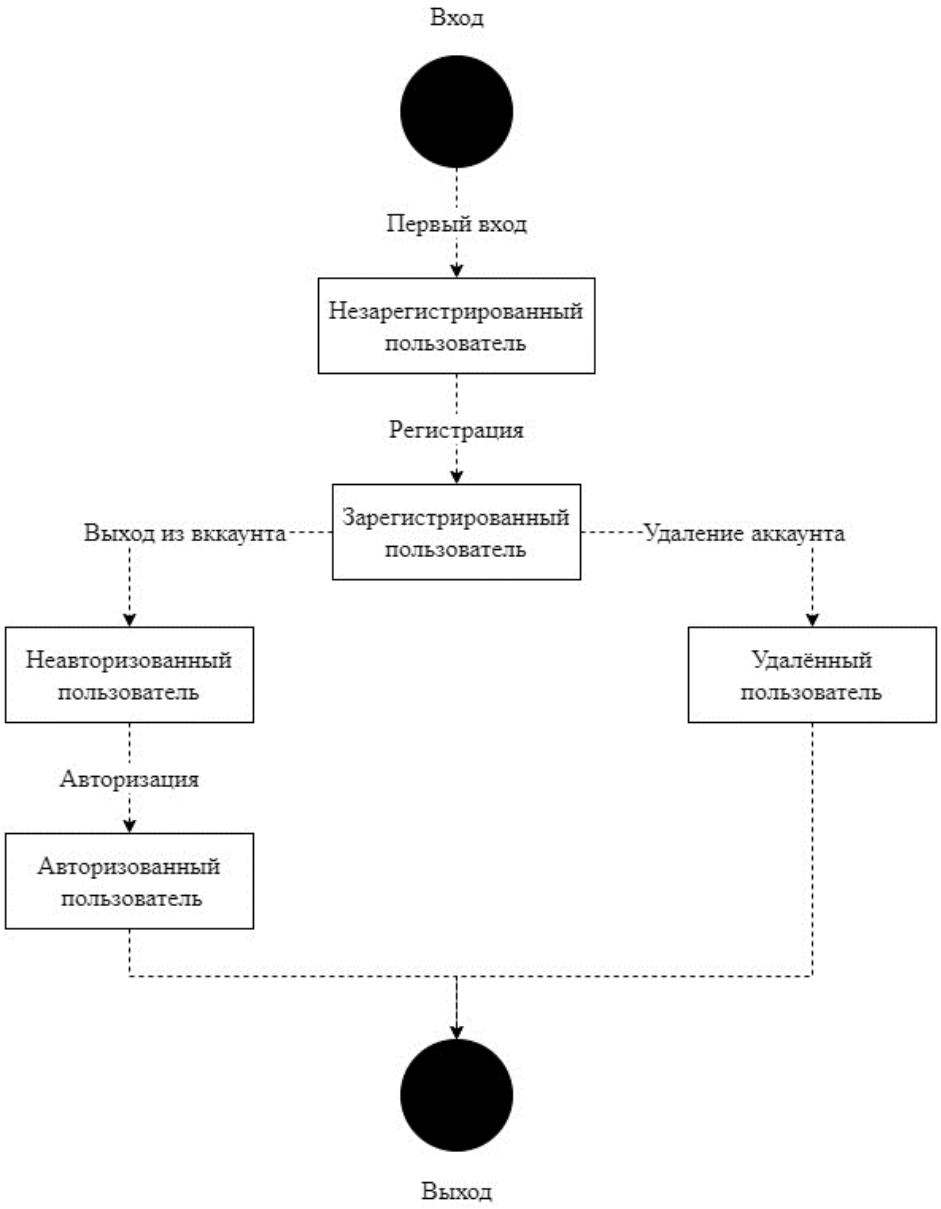

Рисунок 9 - Диаграмма состояний пользователя

3.5 Диаграмма сотрудничества

Диаграмма сотрудничества показывает связи и взаимодействия неавторизованного пользователя с системой и базой данных (рис. 10).

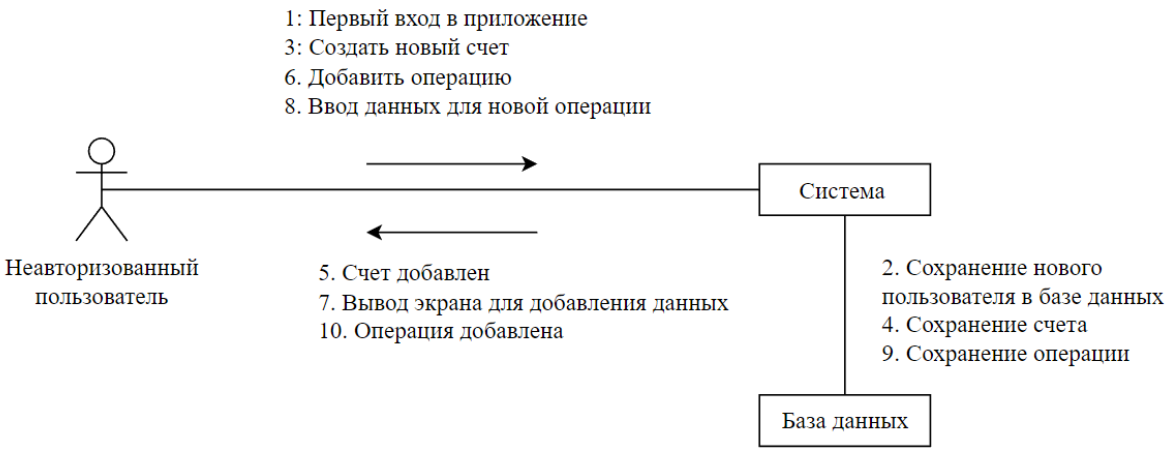

Рисунок 10 - Диаграмма сотрудничества

3.6 Диаграмма последовательности

Диаграмма последовательности отображает детальное описание логики сценариев использования и уточняет диаграмму прецедентов (рис. 11, 12).

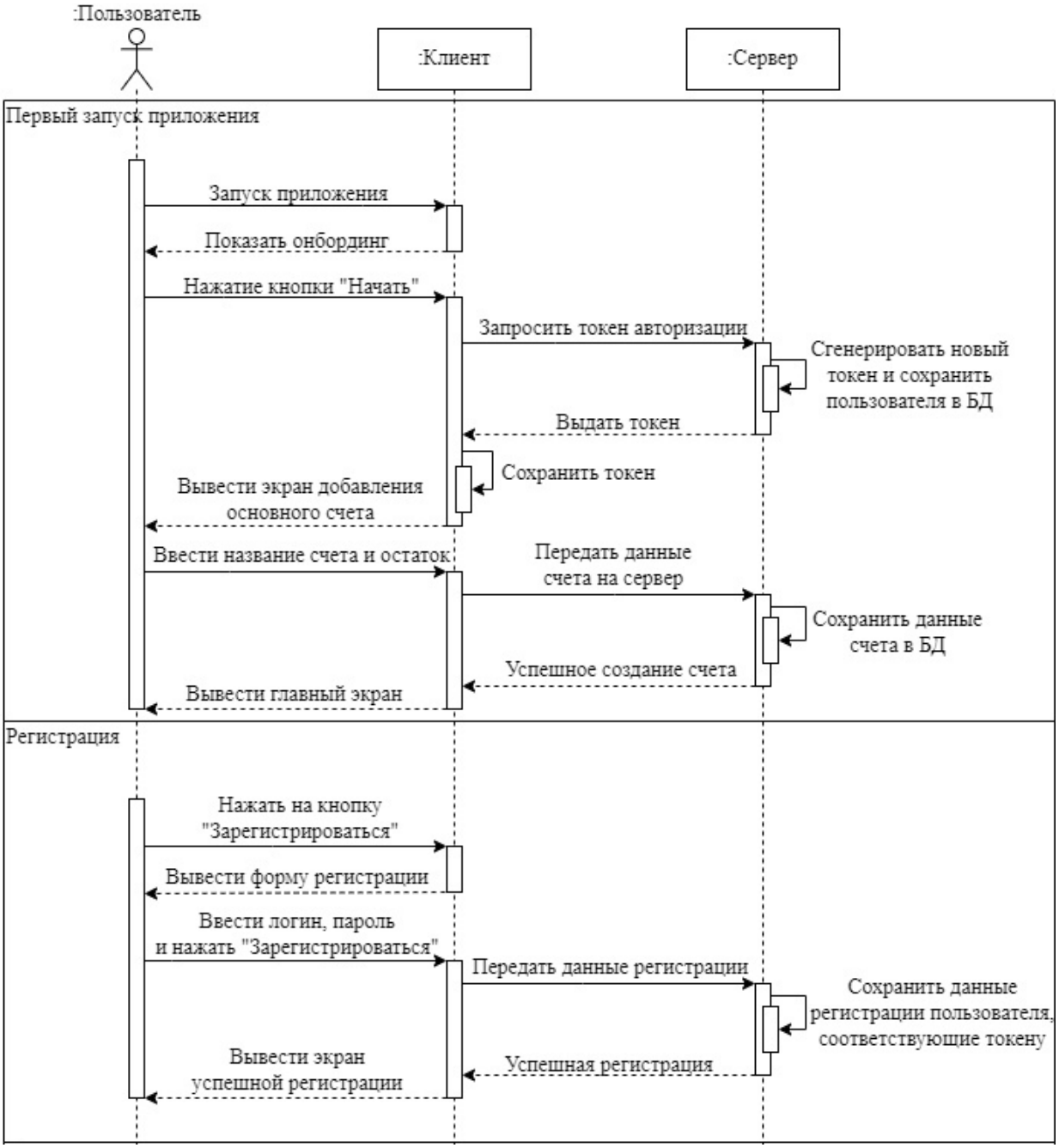

Рисунок 11 - Диаграмма последовательности (1 часть)

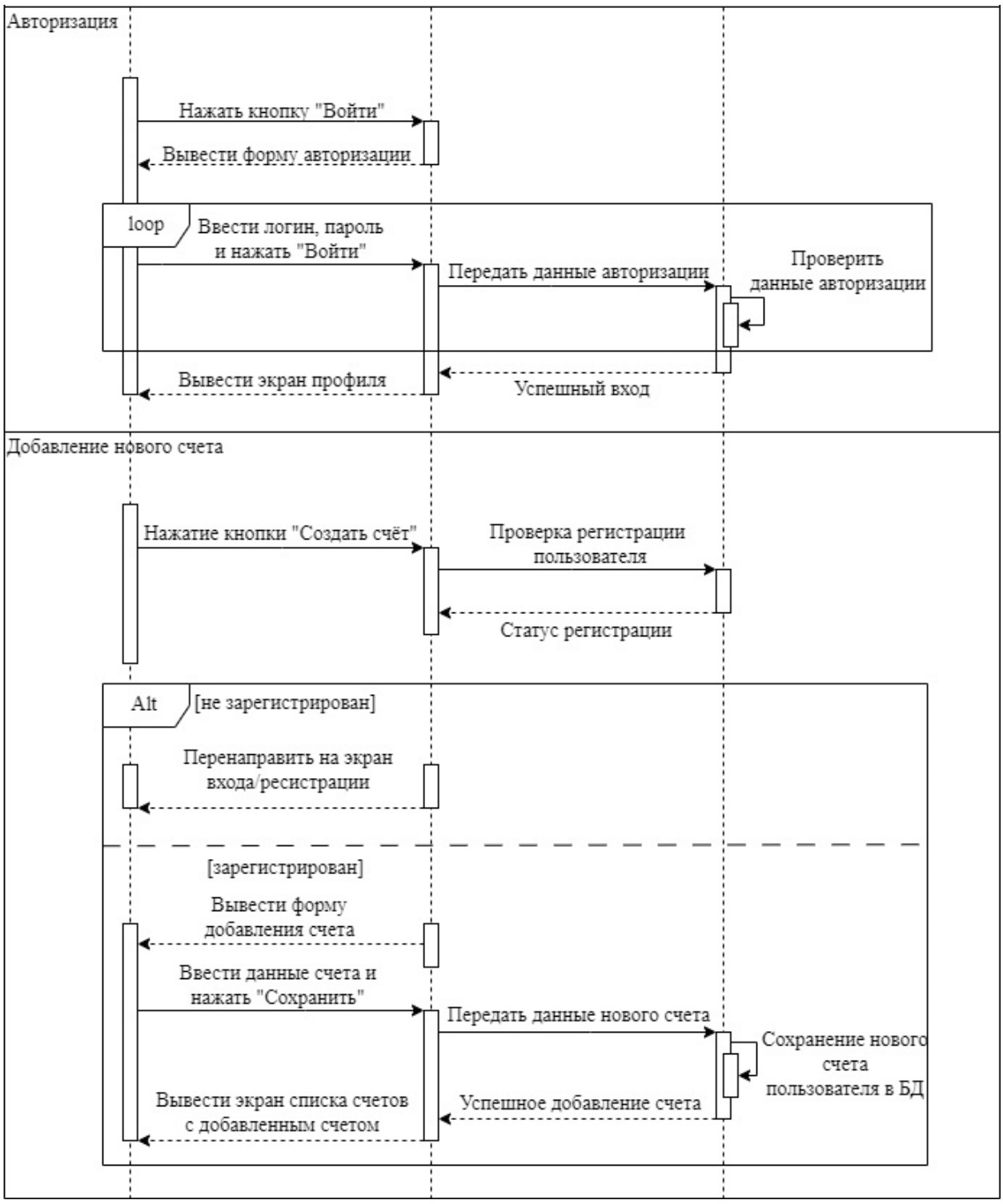

Рисунок 12 - Диаграмма последовательности (2 часть)

3.7 Диаграмма классов

Диаграмма классов определяет типы классов системы и связи, которые существуют между ними. На диаграммах классов изображаются также атрибуты классов, операции классов и ограничения, которые накладываются на связи между классами (рис. 13).

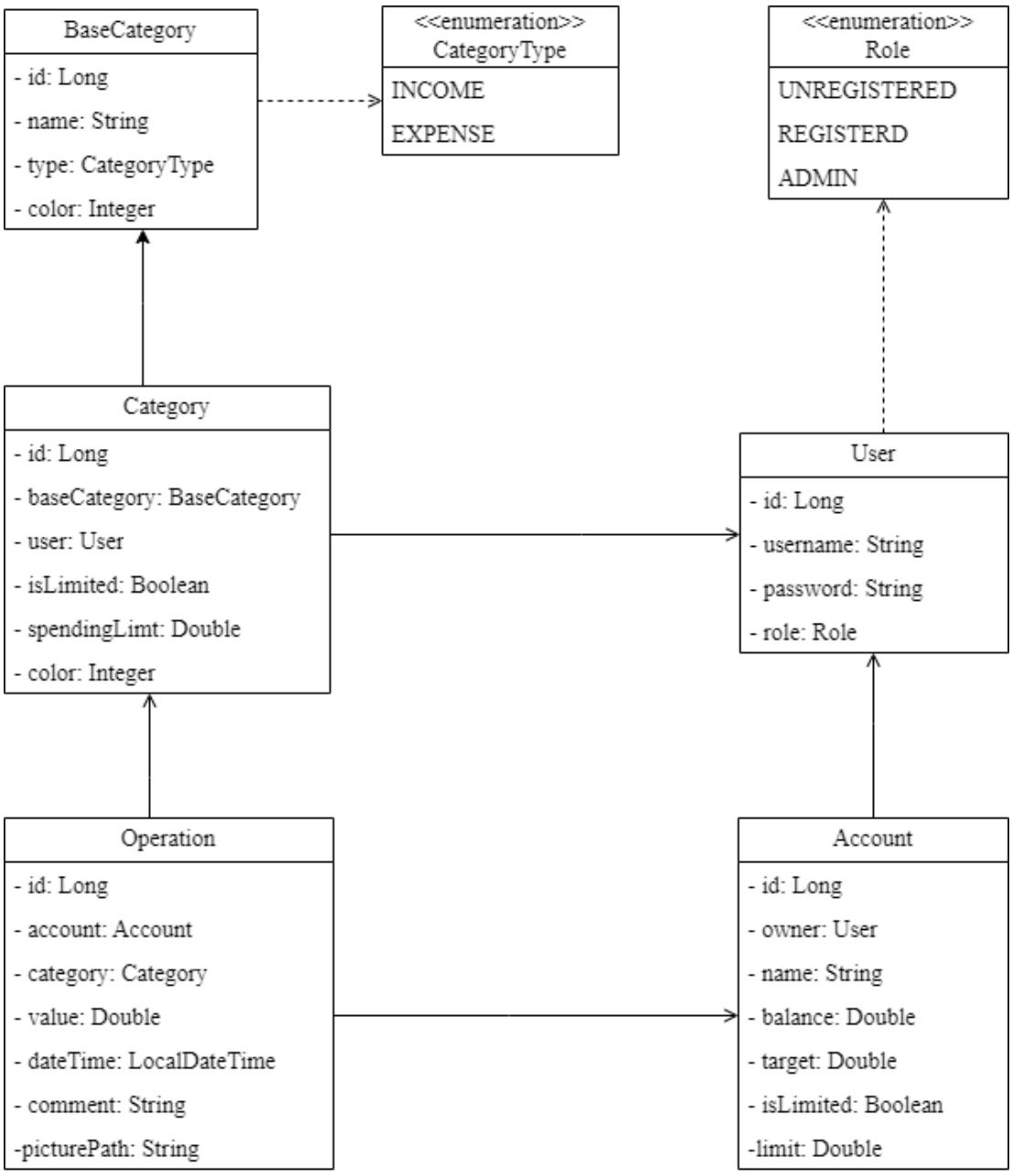

Рисунок 13 - Диаграмма классов

4 Реализация

4.1 Анализ средств реализации

Для реализации серверной части выбраны технологии:

— язык программирования Java;

— фреймворк Spring Boot;

— СУБД PostgreSQL.

Для реализации клиентской части выбраны технологии:

— язык программирования Kotlin;

— Android SDK.

Сочетание всех этих средств позволяет разработчикам создавать высокопроизводительные и надежные приложения с удобным интерфейсом и хорошей масштабируемостью. При разработке серверной части Java предоставляет широкий спектр инструментов и библиотек, Spring Boot облегчает разработку и внедрение приложений, а PostgreSQL обеспечивает надежное хранение и обработку данных. В процессе разработки клиентской части Kotlin предоставляет удобный и безопасный язык программирования для разработки, а Android SDK обеспечивает все необходимые средства для создания, тестирования и развертывания приложений.

4.2 Разработка серверной части

4.2.1 MVC

Серверная часть приложения реализована согласно архитектурному паттерну Model-View-Controller (MVC).

Архитектура MVC описывает простой способ построения структуры приложения, целью которого является отделение бизнес-логики от пользовательского интерфейса. В результате, приложение легче масштабируется, тестируется и сопровождается [4].

На рисунке 14 представлена концептуальная схема шаблона MVC.

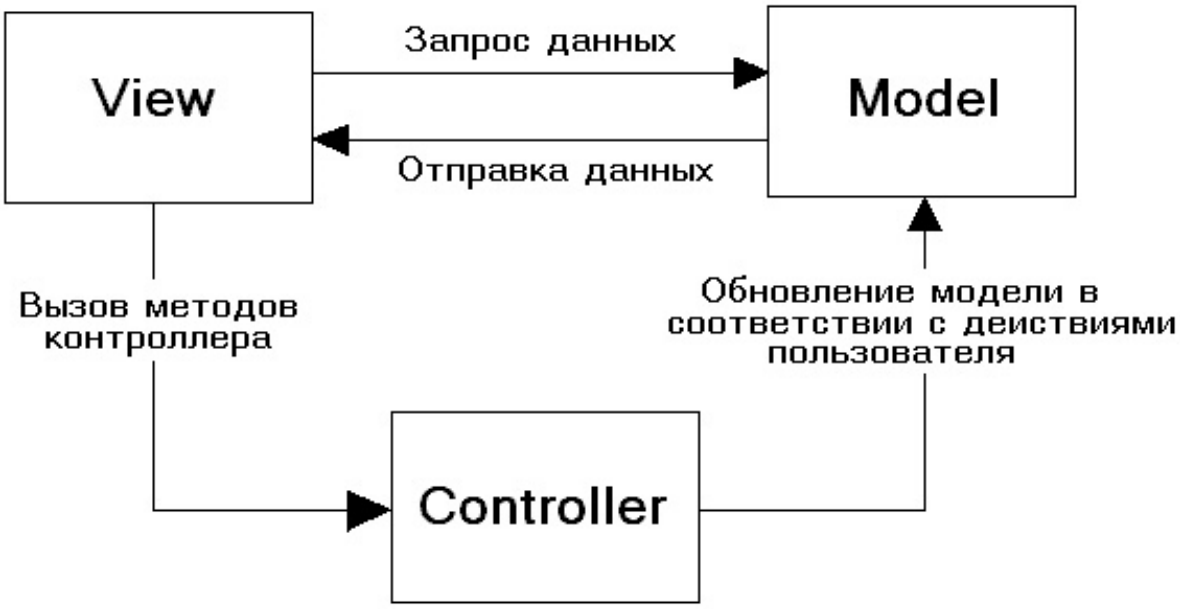

Рисунок 14 - Схема шаблона MVC

4.2.2 Spring Boot

При реализации серверной части был использован фреймворк Spring Boot. Выбор данного фреймворка обусловлен тем, что Spring Boot:

- автоматически конфигурирует проекты на основе одного из стартовых пакетов для них;
- облегчает создание и развертывание приложений на Spring;
- быстро и легко управляет зависимостями и подгружает необходимые модули;
- поддерживает встроенный сервер для запуска приложений;
- может автоматически создать и настроить базу данных для приложения.

Spring Boot следует многоуровневой архитектуре, в которой каждый уровень взаимодействует с уровнем непосредственно ниже или выше него [5]. Схема взаимодействия уровней представлена на рисунке 15.

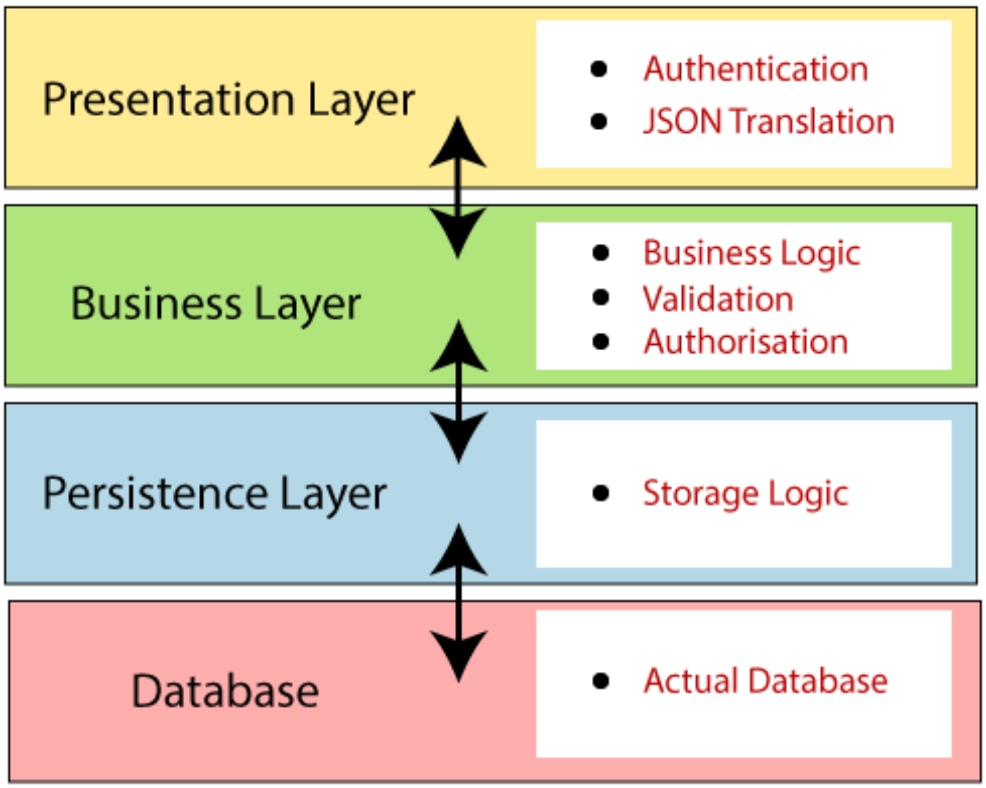

Рисунок 15 - Уровни Spring Boot

В данной архитектуре используются следующие уровни:

— уровень представления Presentation Layer. Данный уровень обрабатывает запросы, преобразует параметр JSON в объект, аутентифицирует запрос и передает его на бизнес-уровень;

— бизнес-уровень Business Layer. Данный уровень обрабатывает всю бизнес-логику. Он получает объекты из уровня доступа к данным и передает их на уровень представления, либо, наоборот, получает данные с уровня представления и передает их на уровень данных;

— уровень доступа к данным Persistence Layer. Spring data позволяет сохранять классы в базу данных. Persistence Layer содержит всю логику хранения и транслирует бизнес-объекты из строк базы данных и в них;

— уровень базы данных Database Layer. На уровне базы данных выполняются операции CRUD (создание, извлечение, обновление, удаление).

4.3 Разработка клиентской части

4.3.1 Kotlin

Для клиентской части приложения был выбран язык программирования Kotlin. Он был объявлен приоритетным языком программирования для Android-разработки компанией Google в 2019 году [7]. Использование Kotlin даёт следующие преимущества:

— полная совместимость с Java;

— лаконичность;

— защита от NullPointerException;

— легкое изучение;

— большое сообщество.

4.3.2 Model-View-ViewModel

Приложение имеет архитектуру, соответствующую шаблону Model-View-ViewModel (MVVM). Схема взаимодействия основных частей шаблона представлена на рисунке 16.

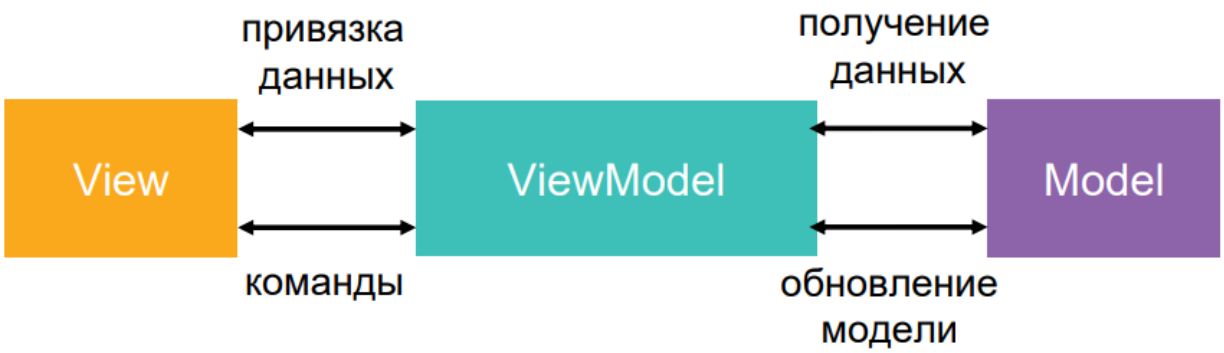

Рисунок 16 - Схема Model-View-ViewModel

В настоящее время архитектурный паттерн MVVM является одним из наиболее оптимальных для разработки мобильных приложений из-за высокой скорости разработки и гибкости [10]. Данный паттерн имеет следующую логику распределения ответственности между модулями:

— модуль Model отвечает за создание моделей данных и может содержать в себе бизнес-логику;

— модуль View в MVVM охватывает интерфейс, логику отображения и обработку пользовательских событий;
— модуль ViewModel предоставляет данные и команды, которые используются представлением для отображения информации пользователю, а также обрабатывает пользовательский ввод и взаимодействие с моделью.

4.3.3 Data Binding

С увеличением количества кода появляются проблемы, связанные со смешиванием логики и представления, что может приводить к сбоям в работе приложения. Для решения этой проблемы была использована библиотека Data Binding, которая открывает возможности для разделения данных, логики и представления.

Эта библиотека помогает ускорить и упростить процесс разработки, уменьшив при этом количество ненужного кода. Это позволяет сосредоточиться над созданием логики, а не поведением представления в различных ситуациях [2].

4.3.4 Navigation Component

Для разработки навигации в приложении был выбрана библиотека Navigation Component. Это комплексное решение проблем для любого типа навигации в приложениях Android. Компонент Navigation обеспечивает новый тип навигации в разработке Android, включающей навигационный граф, который позволяет видеть все экраны и пути навигации между ними [3].

Возможности библиотеки Navigation Component:

— упрощенная настройка стандартных шаблонов навигации;
— обработка транзакций фрагментов;
— безопасность типов при передаче информации в процессе навигации;
— обработка анимации переходов;
— централизация и визуализация навигации;
— корректная обработка действий «назад» и «вперёд»;

— реализация и обработка глубоких ссылок.

4.4 Интерфейс приложения

4.4.1 Экран «Приветствие»

Данный экран появляется при первом входе в приложение, в нем содержится приветственный текст и кнопка для начала работы. По центру экрана находится логотип приложения (рис. 17).

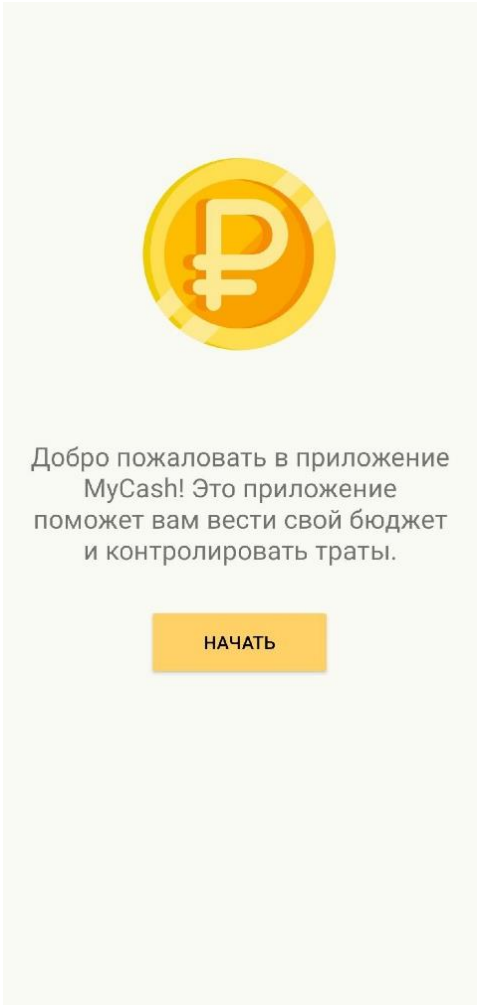

Рисунок 17 - Экран «Приветствие»

4.4.2 Экран «Начало работы»

На данном экране предлагается ввести название счета и сумму остатка на данном счету. Внизу находится кнопка, которая ведет на главный экран (рис. 18).

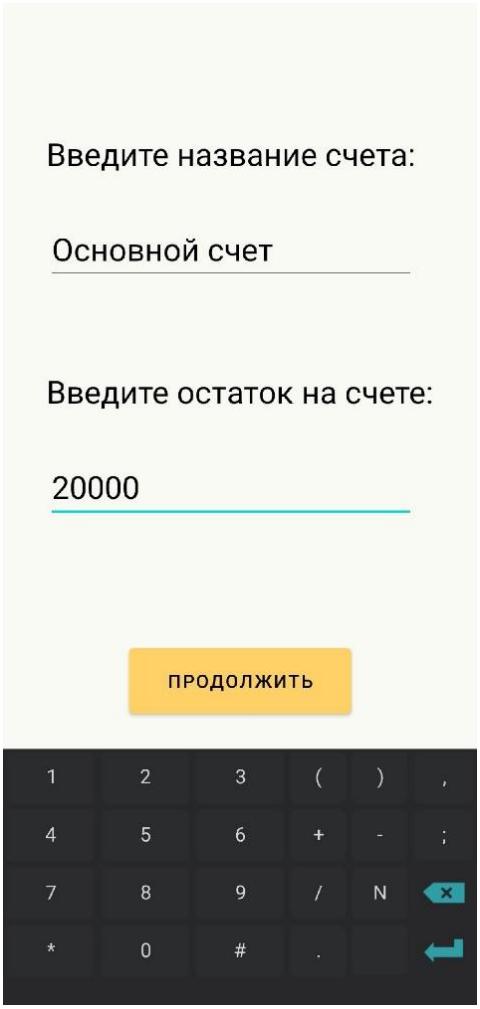

Рисунок 18 - Экран «Начало работы»

4.4.3 Экран «Главная»

На данном экране выводится сумма на счету и история операций, которую можно посмотреть, как за определенный день (рис. 19), так и за определенный месяц (рис. 20). При выборе следующего месяца выводится прогноз расходов и доходов с рекомендациями по сокращению расходов (рис. 21). В правом нижнем углу находится кнопка «+», ведущая на экран «Добавление операции». В нижней панели находится меню, состоящее из вкладок «Графики», «Категории», «Главная», «Счета» и «Профиль».

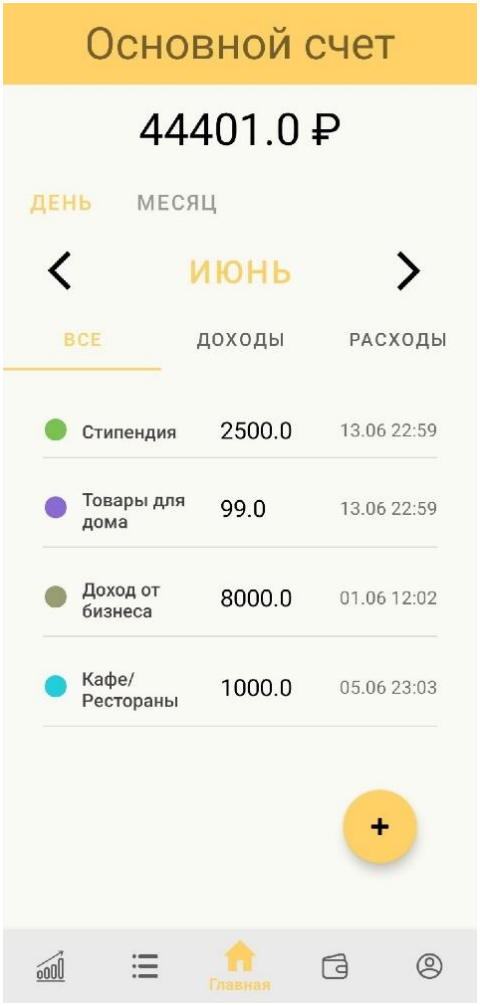

Рисунок 19 - История всех операций за определённый день

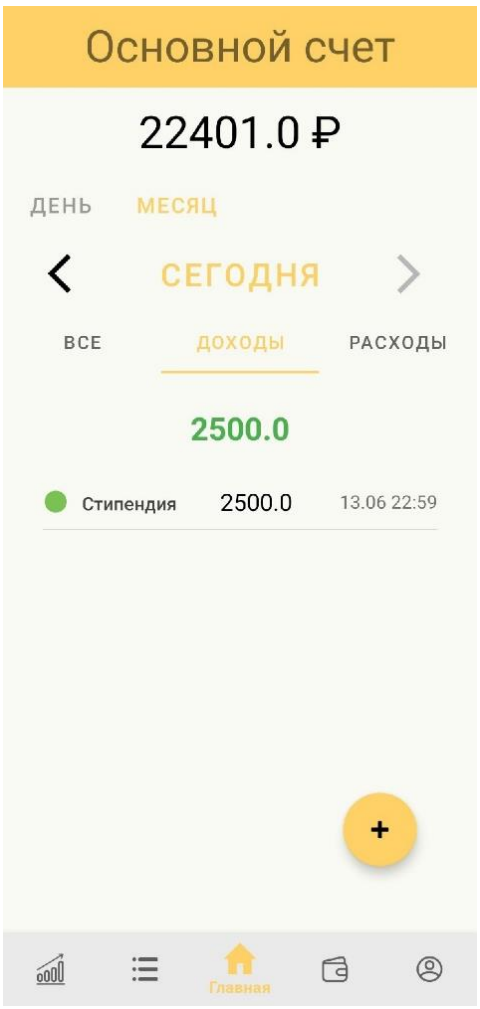

Рисунок 20 - История доходов за определённый день

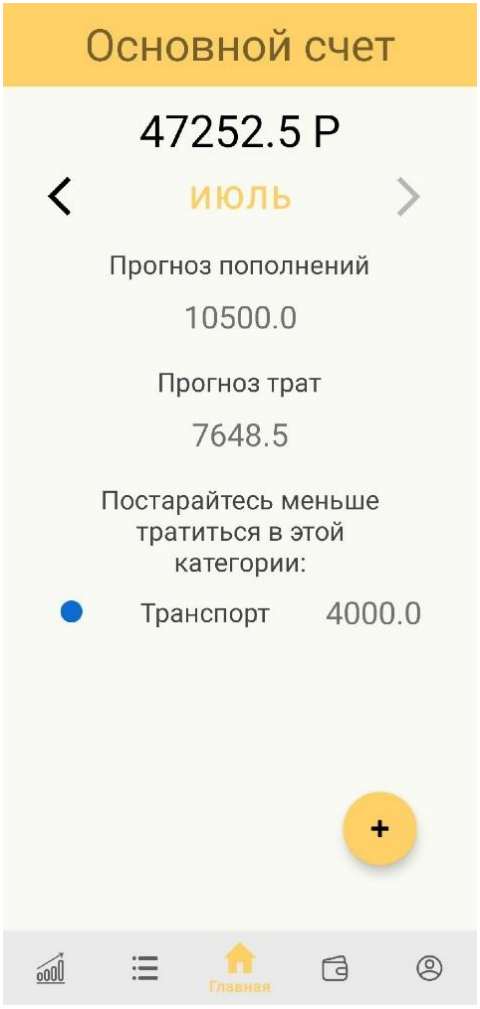

Рисунок 21 - Прогнозы на следующий месяц

4.4.4 Экран «Добавление операции»

Данный экран предназначен для совершения операций над счетом – добавления расходов или доходов (рис. 22). В верхней части экрана можно выбрать тип операции, «Доходы» или «Расходы». Далее идет поле, в которое требуется ввести сумму операции, поле выбора счета, выбора категории и даты. К операции можно добавить комментарий и фотографию.

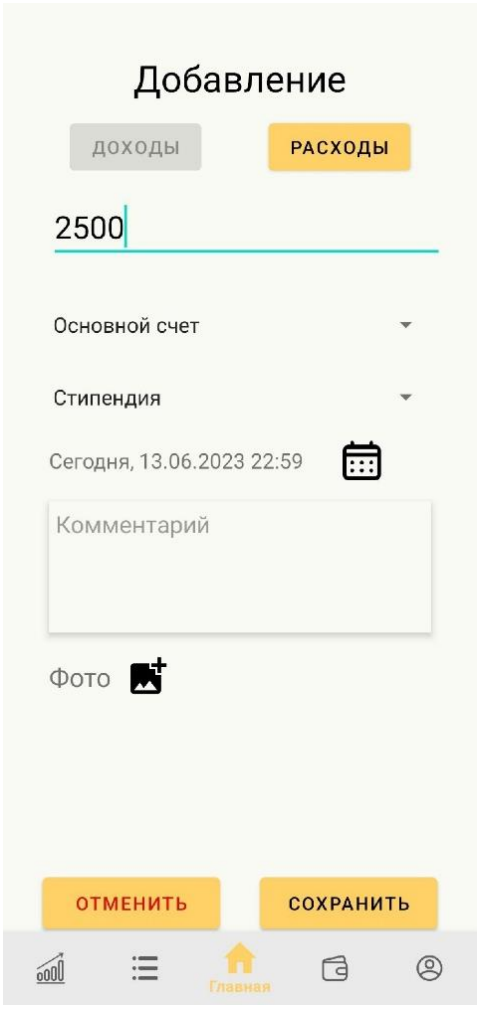

Рисунок 22 - Добавление дохода

4.4.5 Экран «Профиль»

При переходе на этот экран неавторизованный пользователь увидит предложение зарегистрироваться и две кнопки: «Зарегистрироваться» и «Войти» (рис. 23). При нажатии на кнопку «Зарегистрироваться» пользователь переходит на экран «Регистрация», при нажатии на кнопку «Войти» пользователь переходит на экран «Авторизация».

Авторизованный пользователь при переходе на этот экран увидит настройки профиля, кнопки «Удалить профиль» и «Выйти» (рис. 24).

Рисунок 23 - Экран «Профиль» неавторизованного пользователя

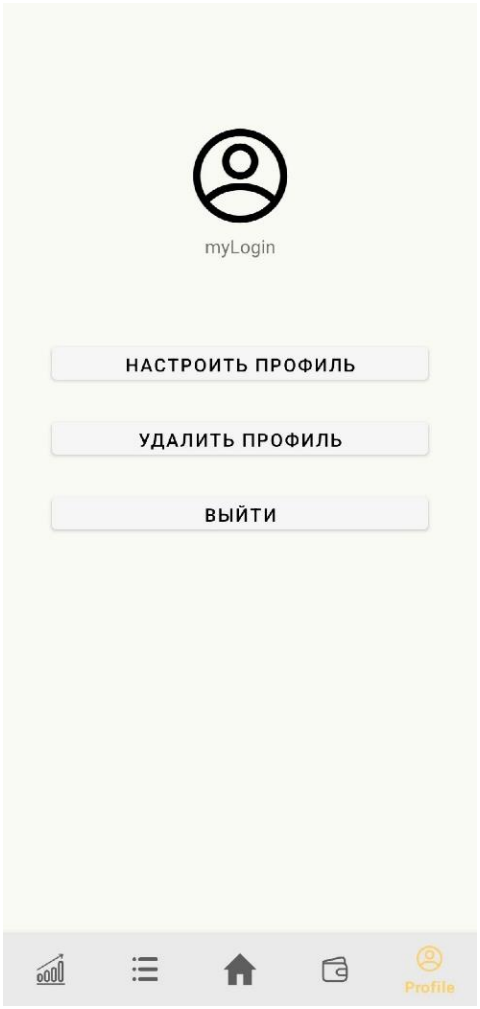

Рисунок 24 - Экран «Профиль» авторизованного пользователя

4.4.6 Экран «Регистрация»

Данный экран доступен для неавторизованных пользователей и позволяет зарегистрироваться в системе (рис. 25).

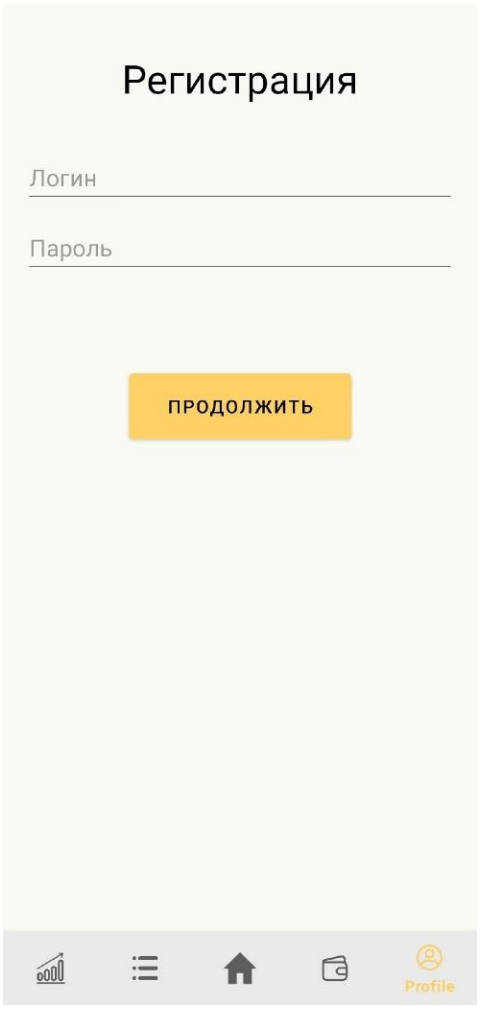

Рисунок 25 - Экран «Регистрация»

4.4.7 Экран «Авторизация»

Данный экран доступен для неавторизованных пользователей и позволяет осуществить вход в ранее зарегистрированный аккаунт (рис. 26).

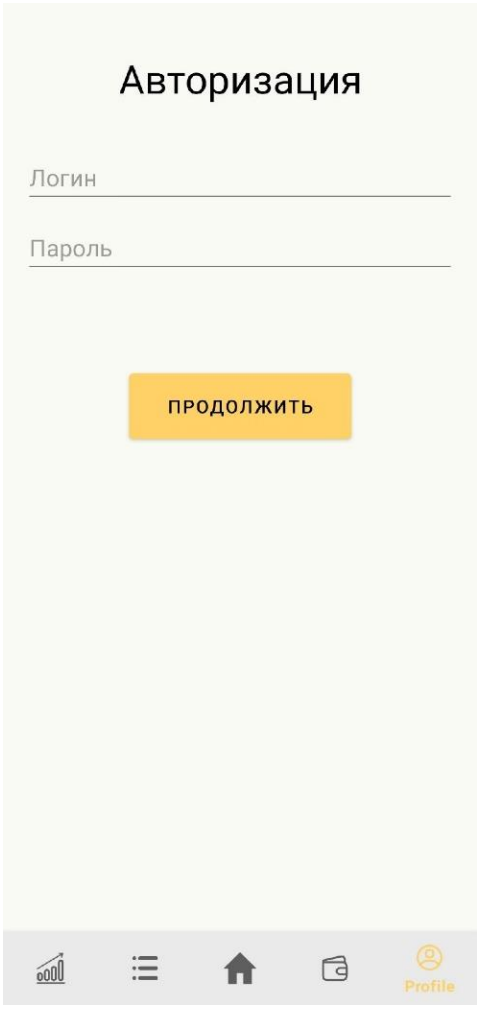

Рисунок 26 - Экран «Авторизация»

4.4.8 Экран «Аналитика»

На данном экране авторизованный пользователь может выбрать счет, аналитику которого желает увидеть, тип графика (общий, только по доходам или только по расходам), и период (по дням или месяцам). Ниже выводится выбранная диаграмма (рис. 27).

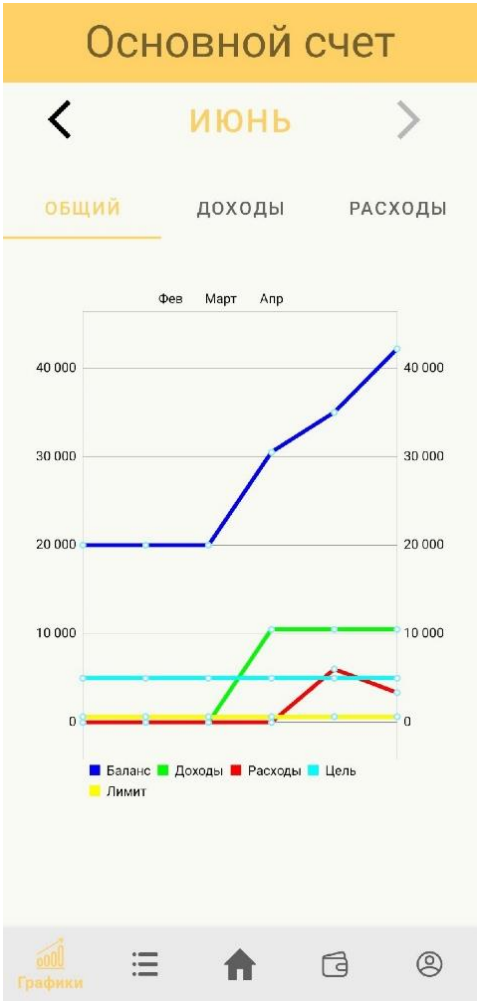

Рисунок 27 - Экран аналитики расходов

4.4.9 Экран «Категории»

При переходе на этот экран пользователь видит список категорий. В верхней панели можно выбрать, какие категории показывать - для доходов или расходов (рис. 28). Ниже выводится список всех категорий, для каждой можно выбрать цвет (рис. 29), настроить лимит и уведомления.

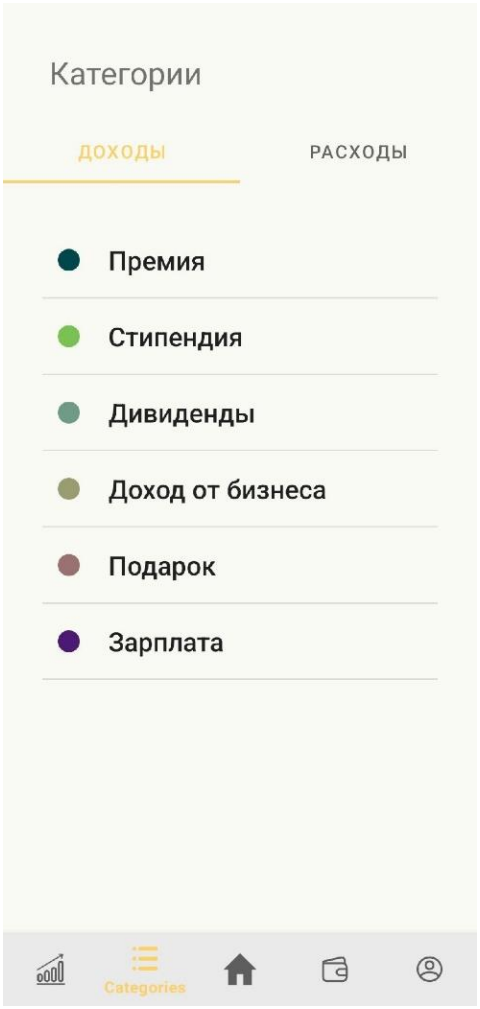

Рисунок 28 - Экран «Категории»

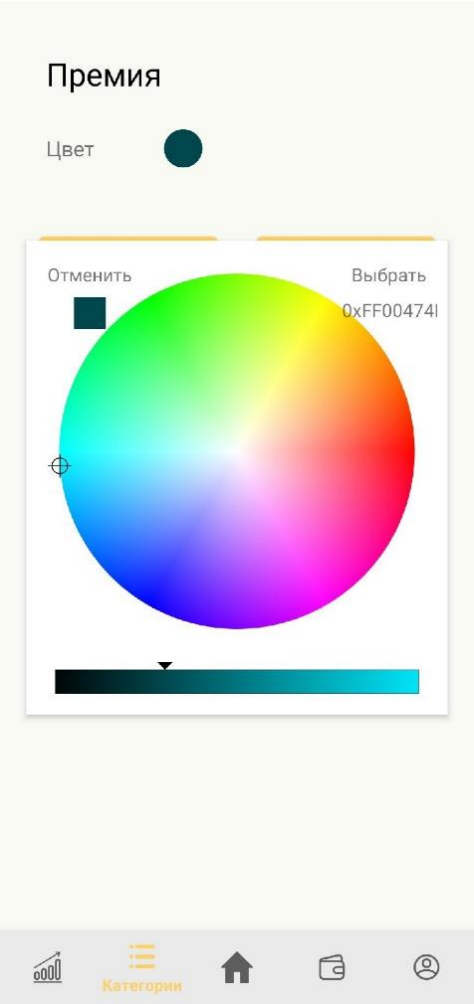

Рисунок 29 - Экран выбора нового цвета категории

4.4.10 Экран «Счета»

На данном экране находится список всех счетов пользователя. Ниже располагается кнопка «+», позволяющая добавить новую категорию (рис. 30). Если пользователь не авторизован, то при нажатии кнопки «+» он переходит на экран профиля с предложением зарегистрироваться или войти. Авторизованный пользователь может добавить любое количество новых счетов и настраивать их. Для счетов доступны настройка лимита, настройка финансовой цели, уведомлений и удаление счета (рис. 31).

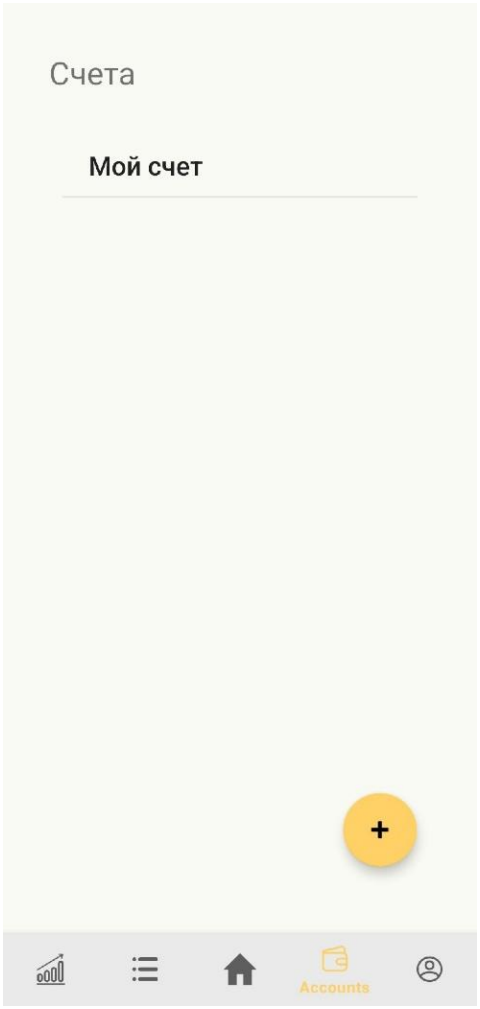

Рисунок 30 - Экран «Счета» (для авторизованного пользователя)

Рисунок 31 - Экран «Добавление счета»

4.4.11 Навигация по приложению

Для навигации в этом приложении используется меню в нижней панели. Меню состоит из пяти пунктов: «Графики», «Категории», «Главная», «Счета» и «Профиль».

5 Тестирование

Было проведено тестирование веб-приложения MyCash при помощи следующих технологий:

- TestContainers – это библиотека, которая поддерживает тесты JUnit и предоставляет временные экземпляры основных баз данных или веб-браузеров для запуска в Docker-контейнере. TestContainers предоставляет API для автоматизации настройки окружения;
- JUnit – это фреймворк для языка программирования Java, предназначенный для автоматического тестирования программ;
- Mockito – фреймворк для тестирования приложений. Этот инструмент упрощает разработку юнит-тестов для классов с внешними зависимостями. Фиктивная реализация интерфейса, статического метода или класса, которую производит Mockito, позволяет определить вывод конкретных вызовов методов: фиктивные классы записывают взаимодействие с системой, а тесты могут его проверить и подтвердить.

Заключение

В данной курсовой работе была рассмотрена разработка системы управления домашним бюджетом с рекомендациями по сокращению расходов. Была проведена аналитическая работа, определены требования к системе, произведен выбор технологий и инструментов для реализации проекта.

В результате работы была создана система, позволяющая пользователю управлять своими доходами и расходами, а также планировать свой бюджет с помощью прогнозов и графиков.

Серверная часть приложения и база данных были размещены в контейнере Docker Compose на хостинге. Разработанное приложение удовлетворяет поставленным требованиям. В процессе разработки были выполнены все поставленные задачи.

Для дальнейшего развития проекта планируется добавление функции отложенных операций, интеграция банковских сервисов для автоматического добавления транзакций, а также добавление новых тем для приложения и поддержка английского языка.

Список используемой литературы

1. Половина россиян ведут семейный бюджет / Аналитический центр НАФИ [Электронный ресурс]. – Режим доступа: https://nafi.ru/analytics/polovina-rossiyan-vedut-semeynyy-byudzhet/. – Заглавие с экрана. – (Дата обращения: 22.05.2023)
2. Реализация паттерна MVVM на Android через Data Binding / Владимир Иванов (IT-копирайтер) [Электронный ресурс]. – Режим доступа: https://www.azoft.ru/blog/mvvm-android-data-binding/. – Заглавие с экрана. – (Дата обращения: 22.05.2023).
3. Навигация для Android с использованием Navigation Component / KotlinStudio [Электронный ресурс]. – Режим доступа: https://learn.microsoft.com/ru-ru/dual-screen/android/api-reference/dualscreen-library/navigation-component/?tabs=kotlin – (Дата обращения: 22.05.2023).
4. Что такое архитектура MVC: Model-View-Controller / Merion Networks. – 2022. [Электронный ресурс]. – Режим доступа: https://wiki.merionet.ru/servernye-resheniya/100/chto-takoe-arhitektura-mvc-model-view-controller/ – (Дата обращения: 25.05.2023)
5. Spring Boot 101: Введение в создание веб-приложений / Machine learning [Электронный ресурс]. – Режим доступа: https://vc.ru/u/1389654-machine-learning/586955-spring-boot-101-vvedenie-v-sozdanie-veb-prilozheniy – (Дата обращения: 25.05.2023).
6. Spring Boot Architecture [Электронный ресурс]. – Режим доступа: https://www.javatpoint.com/spring-boot-architecture. – (Дата обращения: 25.05.2023).

7. Kotlin is now Google’s preferred language for Android app development / Frederic Lardinois – May 7, 2019. [Электронный ресурс]. – Режим доступа: https://techcrunch.com/2019/05/07/kotlin-is-now-googles-preferred-language-for-android-app-development/ – (Дата обращения: 20.05.2023).
8. Android SDK / skillfactory media. [Электронный ресурс]. – Режим доступа: https://blog.skillfactory.ru/glossary/android-sdk/ – (Дата обращения: 23.05.2023).
9. Приложения для учета финансов / Екатерина Надежкина – 19.08.2021. [Электронный ресурс]. – Режим доступа: https://daily.afisha.ru/money/20691-9-luchshih-prilozheniy-dlya-ucheta-finansov/ – (Дата обращения: 24.05.2023).
10. Процесс выбора архитектуры для мобильного приложения / Курганова А.Г / student – 2021. [Электронный ресурс]. – Режим доступа: https://cyberleninka.ru/article/n/protsess-vybora-arhitektury-dlya-mobilnogo-prilozheniya – (дата обращения: 25.03.2023).