

МИНОБРНАУКИ РОССИИ  
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ  
ВЫСШЕГО ОБРАЗОВАНИЯ  
«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»  
(ФГБОУ ВО «ВГУ»)

Факультет компьютерных наук  
Кафедра программирования и информационных технологий

Курсовой проект по курсу «Технологии программирования»  
«Приложение для учёта доходов и расходов»

Зав. кафедрой \_\_\_\_\_ *Махортов С.Д., д. ф.-м. н., профессор*

Обучающиеся \_\_\_\_\_ *Филонов П.О., 3 курс, 7 группа*

\_\_\_\_\_ *Толчеев Д.В., 3 курс, 7 группа*

\_\_\_\_\_ *Мергенов Д.А., 3 курс, 7 группа*

Руководитель \_\_\_\_\_ *Тарасов В.С., ст. преподаватель*

# Содержание

|  |    |
|--|----|
| Содержание .....   | 2  |
| Введение .....   | 4  |
| 1. Анализ предметной области .....   | 5  |
| 1.1. Глоссарий .....   | 5  |
| 1.2. Существующие решения .....  | 6  |
| 2. Постановка задачи .....   | 7  |
| 3. Анализ задачи .....   | 8  |
| 3.1. Анализ использования приложения .....   | 8  |
| 3.2. Взаимодействие системы компонентов .....  | 10 |
| 3.3. Варианты состояния системы .....  | 11 |
| 3.4. Развёртывание приложения .....  | 12 |
| 4. Требования к программному продукту .....  | 13 |
| 5. Планирование работ .....  | 14 |
| 5.1. Средства реализации .....   | 14 |
| 5.2. Основные виды работ, которые необходимо выполнить в процессе разработки программного средства ..... | 15 |
| 5.3. Состав команды, распределение задач по времени .....  | 16 |
| 6. Проектирование .....  | 18 |
| 6.1. Архитектура разрабатываемого программного средства .....  | 18 |
| 6.2. Схема базы данных .....   | 19 |
| 7. Сценарии воронок конверсии .....  | 21 |
| 8. Модули сервера .....  | 22 |
| 8.1. Модуль доступа к данным .....   | 22 |
| 8.2. Модуль бизнес-логики .....  | 24 |
| 8.3. Модуль web-приложения .....   | 25 |
| 9. Проект интерфейсной части программного средства .....   | 26 |
| 9.1. Авторизация .....   | 26 |
| 9.2. Регистрация .....   | 27 |
| 9.3. Кошелёк .....   | 28 |

|   |    |
|---|----|
| 9.4. Группы.....  | 29 |
| 9.5. Личный кабинет пользователя .....                                | 30 |
| 9.6. Группа.....  | 31 |
| 9.6.1. История группы .....   | 31 |
| 9.6.2. Для создателя .....  | 32 |
| 9.6.3. Для участника .....  | 33 |
| 9.6.4. Категория .....  | 34 |
| 10. Тестирование.....   | 35 |
| 10.1. Модульное тестирование.....                                     | 35 |
| 10.2. Командное тестирование .....                                    | 36 |
| 11. Оценка степени завершённости и перспективы развития проекта ..... | 38 |
| Заключение.....   | 39 |

## Введение

Финансово грамотные люди выделяются умением правильно считать деньги. Их отличает любовь к планированию, способность оптимизировать расходы и стойкость перед спонтанными покупками. Учет финансов помогает выработать дисциплину и ежедневный контроль, т.е. ежедневный отчет самому себе, о том, что, как и куда потрачено. Таким образом можно создавать разные отчеты и подводить итоги расходов раз в какой-либо промежуток времени, тем самым более детально и подробно быть в курсе движения денежных средств.

Способы учётов доходов и расходов:

— Бумажный блокнот

Это самый привычный способ ведения личного бюджета. Наверное, каждый начинал с него учитывать доходы и расходы. Покупки и прибыль записываются в блокнот и в нужный момент происходит ручной расчёт.

— Excel и Google таблицы

В этом формате можно настроить под себя статьи расходов и доходов. Цвет позволяет выделить нужные параметры. Программа по формулам сама подводит итоги, но с некоторым ожиданием времени.

— Программы и мобильные приложения

Удобство приложений заключается в том, что они всегда под рукой – в телефоне. Вносить данные можно сразу после покупки или получения прибыли. В приложении можно создать несколько категорий доходов и расходов.

Существует множество мобильных приложений для учёта финансов, однако большинство из них направлены на обслуживание личного бюджета пользователя или функция совместного учёта является платной. Данная курсовая работа посвящена разработке приложения, предоставляющего функционал не только для личного кошелька, но и для бесплатного образования групп, в которых можно вести учёт доходов и расходов сразу для нескольких пользователей, которые состоят в группе.

# 1. Анализ предметной области

## 1.1. Глоссарий

Проект – разрабатываемое приложение.

Мобильное приложение – специально разработанное приложение под конкретную мобильную платформу.

Верстка страницы – процесс формирования страницы, состоящей из программного кода, стилей оформления и подгружаемых картинок и фонов, на которые специальным образом разбивается макет, в соответствии с дизайном.

Java – строго типизированный объектно-ориентированный язык программирования общего назначения, разработанный компанией Sun Microsystems.

JDK – комплект разработчика приложений на языке Java, включающий в себя компилятор Java, стандартные библиотеки классов Java, примеры, документацию, различные утилиты и исполнительную систему Java.

Android Studio – интегрированная среда разработки для работы с платформой Android.

API – описание способов, которыми одна компьютерная программа может взаимодействовать с другой программой.

JSON – текстовый формат обмена данными.

## 1.2. Существующие решения

Аналоги:

— CoinKeeper:

— Плюсы:

- Можно импортировать все данные сразу из приложения мобильного банка
- Есть напоминания о регулярных платежах по категориям
- Статистика, которую можно кастомизировать: вы сами распределяете доходы и расходы по категориям и устанавливаете план и лимиты

— Минусы:

- Навязчивая реклама остается в приложении даже после покупки премиум-подписки
- Функция совместного учета платная

— Moneyfy:

— Плюсы:

- Можно выбрать отчетный период и составить наглядную статистику расходов с графиками
- Поддерживает несколько счетов одновременно
- Есть встроенный калькулятор

— Минусы:

- Возникают проблемы с синхронизацией с приложениями банков и другими устройствами
- Функции совместного и мультивалютного учета и планирования платежей платные

## 2. Постановка задачи

Цель курсовой работы: реализация мобильного приложения под операционную систему android для:

- Учёта доходов и расходов пользователя
- Объединения пользователей в группы для совместного ввода и отслеживания доходов и расходов

Приложение должно решать следующие задачи:

- Регистрация и авторизация пользователя
- Изменение данных пользователя
- Создание группы для совместного бюджета
- Поиск группы для совместного бюджета
- Выход из группы или её удаление
- Ввод доходов и расходов в личный бюджет
- Ввод доходов и расходов в бюджет группы
- Отслеживание истории трат и поступлений в личный бюджет
- Отслеживание истории трат и поступлений в бюджет группы
- Создание и удаление категорий доходов и расходов

Необходимо решить следующие задачи:

- Спроектировать и реализовать базу данных
- Спроектировать следующие страницы и реализовать их функционал:
  - Авторизация
  - Регистрация
  - Кошелёк
  - Личный кабинет пользователя
  - Группы
  - Группа
  - Категория
- Реализовать механизм функциональных переходов между страниц

### 3. Анализ задачи

#### 3.1. Анализ использования приложения

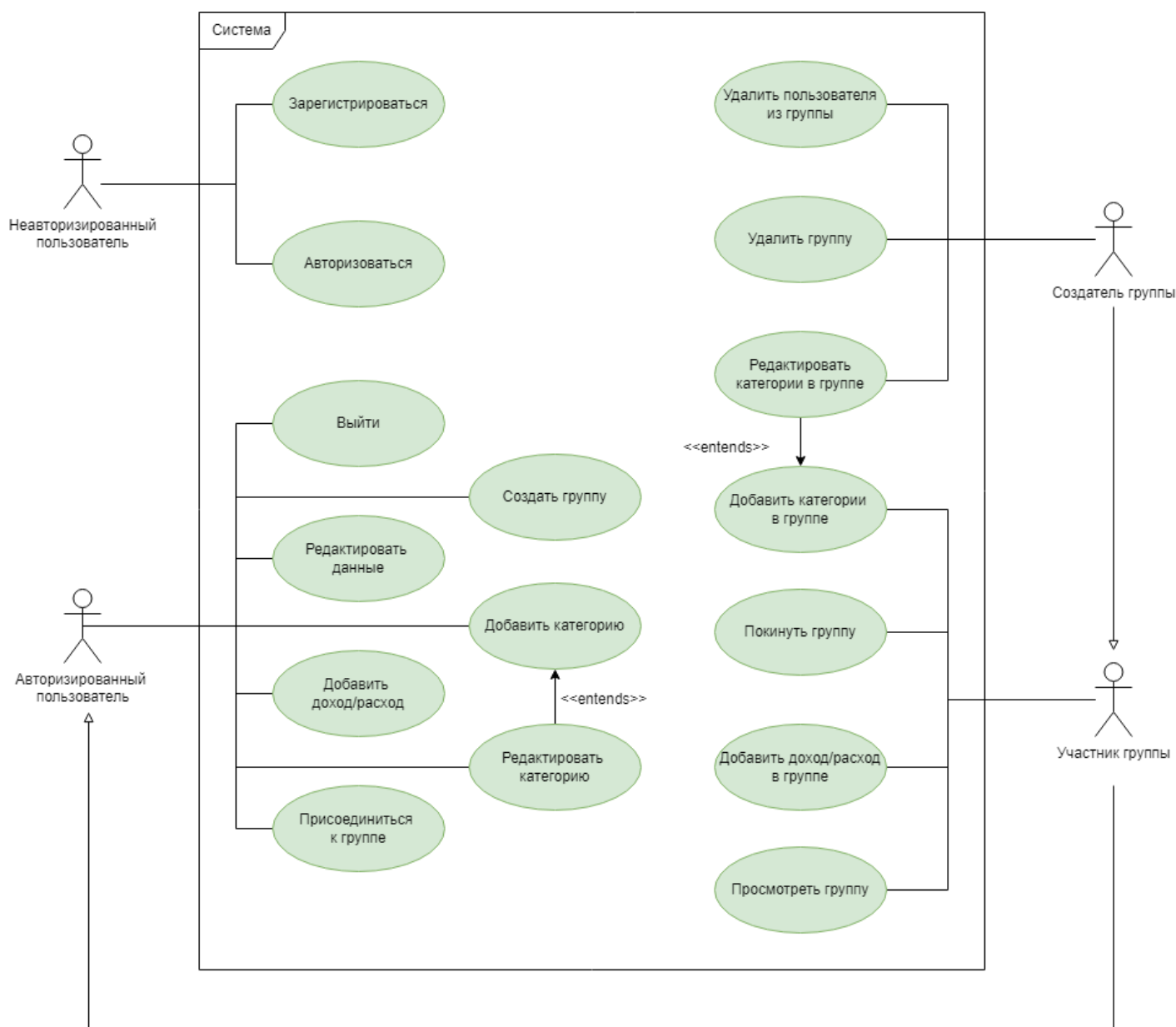


Рисунок 1 Диаграмма прецедентов

На рисунке 1 продемонстрирован функционал для пользователей.

— Неавторизованный пользователь

— Войти

— Зарегистрироваться

— Авторизованный пользователь

— Редактировать данные



- Добавить категорию
- Редактировать категорию (переименовать или удалить)
- Добавить доход
- Добавить расход
- Создать группу
- Присоединиться к группе
- Выйти
- Участник группы
  - Просмотр группы
  - Добавить доход
  - Добавить расход
  - Покинуть группу
- Создатель группы
  - Добавить категорию
  - Редактировать категорию (переименовать или удалить)
  - Удалить участника
  - Удалить группу

### 3.2. Взаимодействие системы компонентов

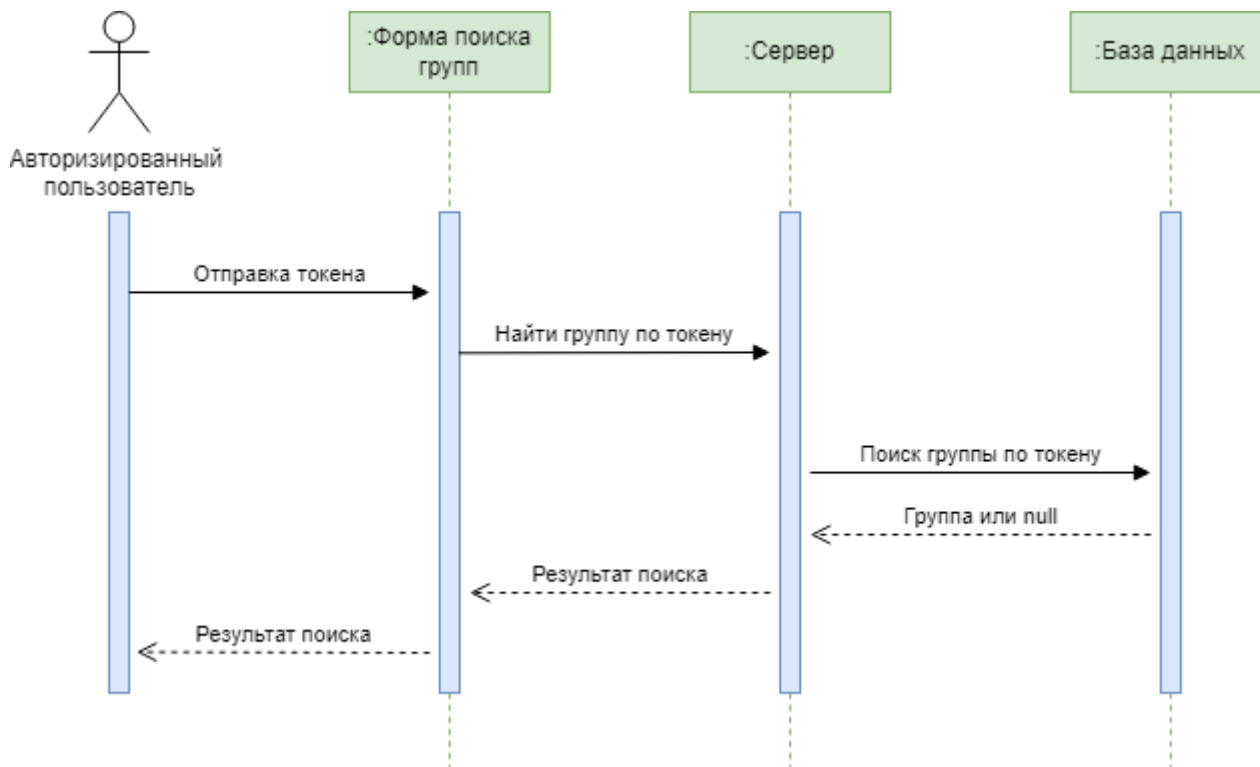


Рисунок 2 Диаграмма последовательностей

Рисунок 2 (диаграмма последовательностей) показывает упорядоченное во времени взаимодействие объектов при попытке пользователя найти группу по токену.

Через форму поиска группы отправляется токен на сервер. Сервер обращается на слой ниже и происходит поиск в базе данных. Обратно возвращается результат: группа, если найдена по токену; null, если группа не найдена.

### 3.3. Варианты состояния системы

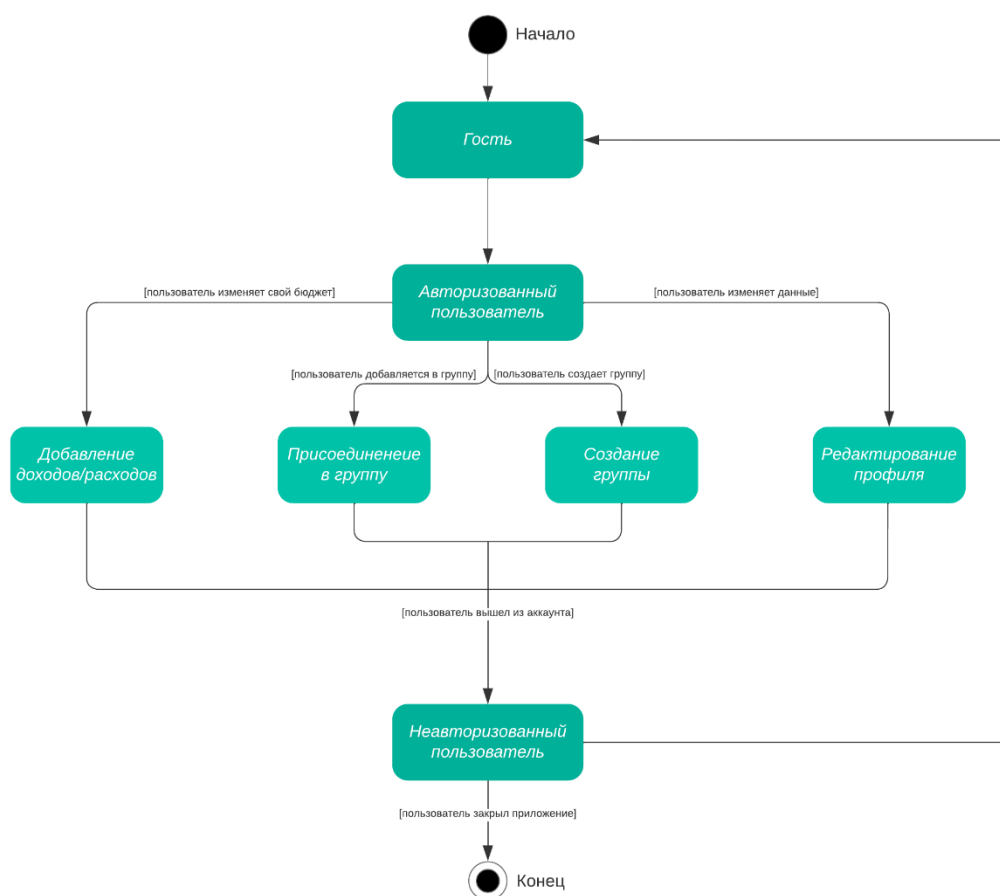


Рисунок 3 Диаграмма состояний

Диаграмма состояний, изображённая на рисунке 3, показывает состояние авторизованного пользователя, его возможные действия, пока тот не выходит из системы.

### 3.4. Развёртывание приложения

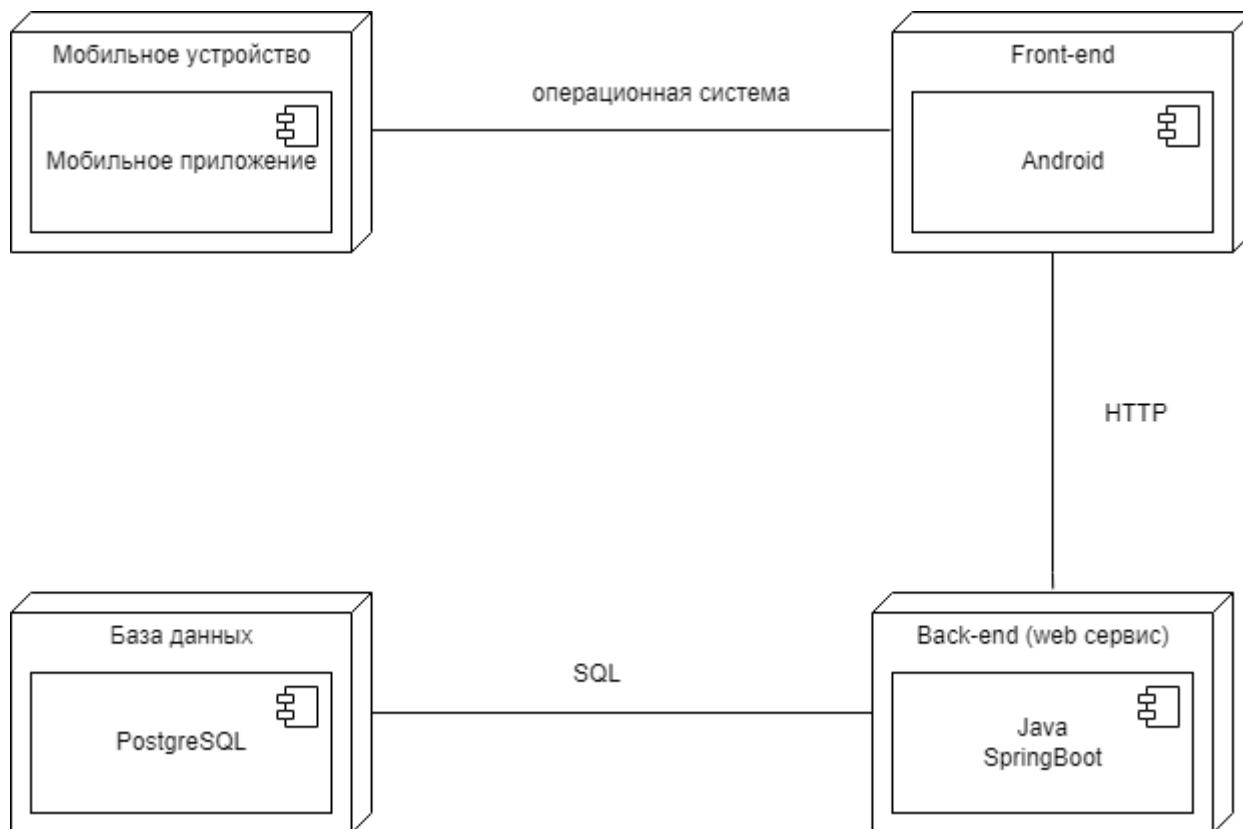


Рисунок 4 Диаграмма развёртывания

На рисунке 4 продемонстрирована диаграмма развёртывания. На ней указаны узлы сети и их взаимодействия между собой. В качестве клиента выступает мобильное устройство пользователя с операционной системой android. На серверной части развёрнуто приложение на SpringBoot и база данных PostgreSQL.

## **4. Требования к программному продукту**

Требования к приложению:

- Решение поставленных целей приложения
- Запуск на установленной версии android (от 8.0)
- Адаптивность к android устройствам установленной версии
- Кодирование конфиденциальной информации (паролей пользователей)
- Удобный интерфейс и выполнение поставленных правил юзабилити:
  - Удобная навигация
  - Упрощённая форма регистрации
  - Упрощённая структура

## **5. Планирование работ**

### **5.1. Средства реализации**

Клиентская часть:

- Java 8
- Android JDK 24.0.2
- Android Studio 4.1.3
- XML 1.0
- Retrofit 2.1.0

Серверная часть:

- Java 11
- SpringBoot 2.6.7
- Java fluent validator 1.10.0
- Spring security
- JWT 0.8.0
- Springfox 3.0.0
- PostgreSQL 14.2

## **5.2. Основные виды работ, которые необходимо выполнить в процессе разработки программного средства**

- Анализ предметной области
- Распределение задач
- Выбор средств реализации
- Проектирование UML-диаграмм
- Реализация серверной части:
  - Проектирование базы данных
  - Реализация базы данных
  - Реализация модулей приложения
- Реализация клиентской части:
  - Проектирование страниц приложения
  - Реализация страниц приложения
  - Взаимодействие с сервером
- Оформление git flow

### 5.3. Состав команды, распределение задач по времени

Этап 1 (16.02.2022 - 16.03.2022):

- Филонов Павел
  - Создание и оформление github репозитория с документацией
  - Разработка курсового проекта для 1-го этапа
  - Разработка технического задания
- Толчеев Данила
  - Анализ предметной области
  - Дизайн страниц приложения на miro

Этап 2 (20.03.2022 - 30.04.2022):

- Филонов Павел
  - Front-end разработка
    - Вёрстка страниц
  - Back-end разработка
    - Проектирование и реализация базы данных
    - Реализация модулей приложения
  - Разработка курсового проекта для 2-го этапа
- Толчеев Данила
  - Проектирование и реализация UML-диаграмм
    - Use case
    - Диаграмма последовательностей
    - Диаграмма состояний
    - Диаграмма развёртывания
    - Диаграмма классов
  - Визуальное оформление git flow

Этап 3 (01.05.2022 – 06.06.2022):

- Филонов Павел
  - Front-end



- Взаимодействие с сервером
- Взаимодействие страниц приложения между собой
- Доработка back-end
- Толчеев Данила
- Тестирование
- Джейхун Мергенов
- Тестирование

## 6. Проектирование

### 6.1. Архитектура разрабатываемого программного средства

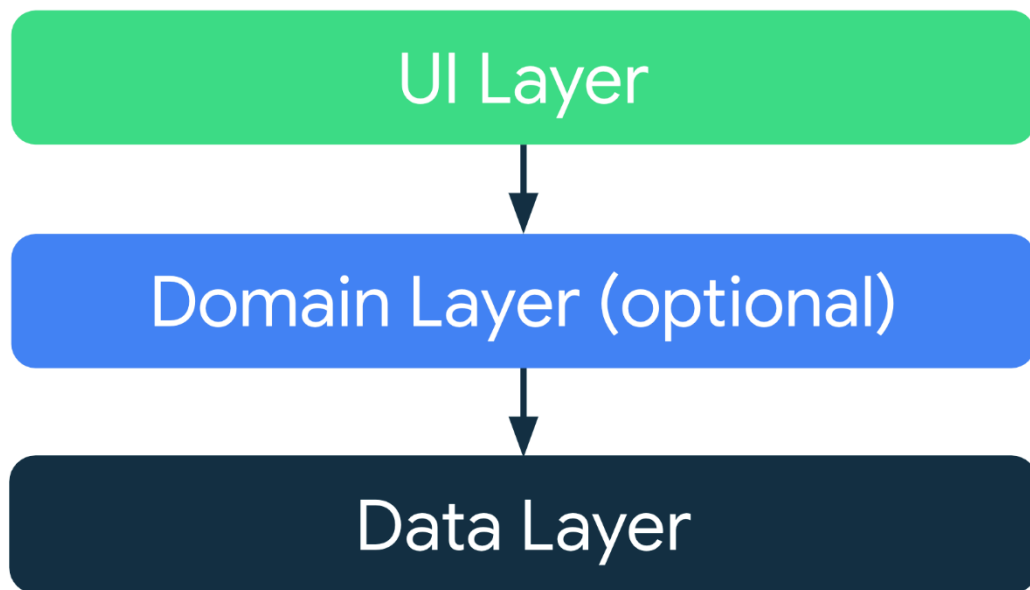


Рисунок 5 Структура

Приложение представляет из себя структуру из следующих слоёв:

- UI Layer – слой пользовательского интерфейса, служащий для взаимодействия пользователя с приложением
- Domain Layer – слой бизнес логики приложения. Является посредником между пользовательским интерфейсом и данными.
- Data Layer – слой уровня доступа к данным. Используется для создания, хранения и изменения данных

## 6.2. Схема базы данных

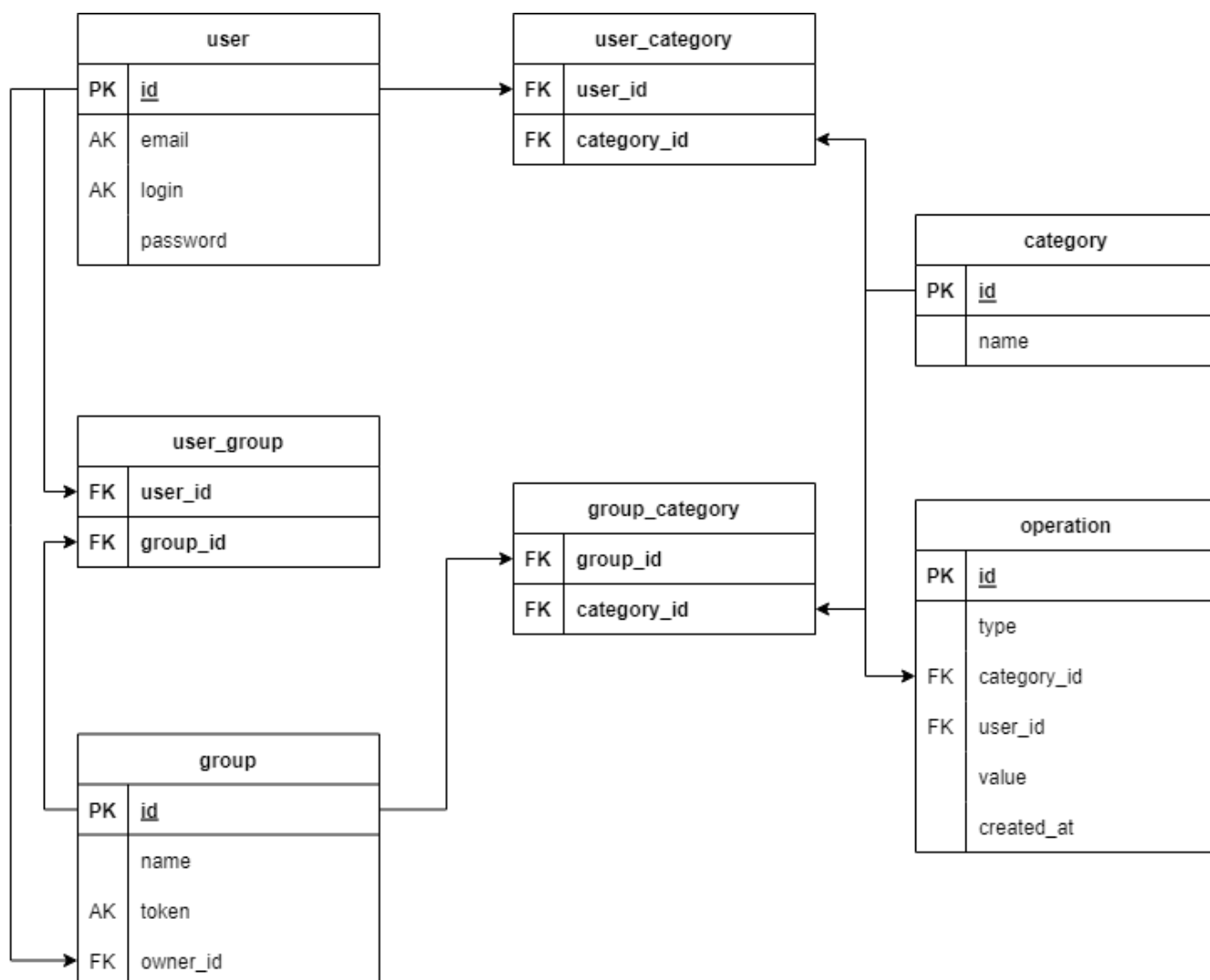


Рисунок 6 - Структура базы данных

Сущности:

— user:

- id – уникальный идентификатор
- email – email пользователя
- login – уникальный логин пользователя
- password – пароль пользователя

— group:

- id - уникальный идентификатор
- name – название группы

- token – уникальный токен группы
- owner\_id – уникальный идентификатор создателя группы
- category
  - id - уникальный идентификатор
  - name – название категории
- operation:
  - id - уникальный идентификатор
  - type – тип денежной операции (доход/расход)
  - category\_id – уникальный идентификатор категории операции
  - user\_id – уникальный идентификатор пользователя, совершившего операцию
  - value – значение операции (в рублях)
  - created\_at – дата проведения операции

Из-за связей «многие-ко-многим» возникают сущности:

- user\_group:
  - user\_id – уникальный идентификатор пользователя
  - group\_id – уникальный идентификатор группы
- user\_category:
  - user\_id – уникальный идентификатор пользователя
  - category\_id – уникальный идентификатор категории
- group\_category:
  - group\_id – уникальный идентификатор группы
  - category\_id – уникальный идентификатор категории

## 7. Сценарии воронок конверсии

- Посетил страницу авторизации - авторизовался - перешёл на страницу групп - ввёл токен - нажал кнопку поиска - присоединился к группе - посмотрел пользователей группы
- Посетил страницу авторизации - авторизовался - ввёл название новой категории - добавил категорию
- Посетил страницу авторизации - авторизовался - перешёл в личный кабинет - ввёл новый пароль - обновил пароль

## 8. Модули сервера

### 8.1. Модуль доступа к данным

Модуль доступа к данным представляет из себя слой, содержащий логику работу с базой данных и обращений к ней. База данных строится на основе сущностей, с помощью JPA, для обращения к которой в модуле содержатся репозитории.

JPA – это спецификация, которая позволяет работать с базой данных посредством сущностей в объектно-ориентированном виде.

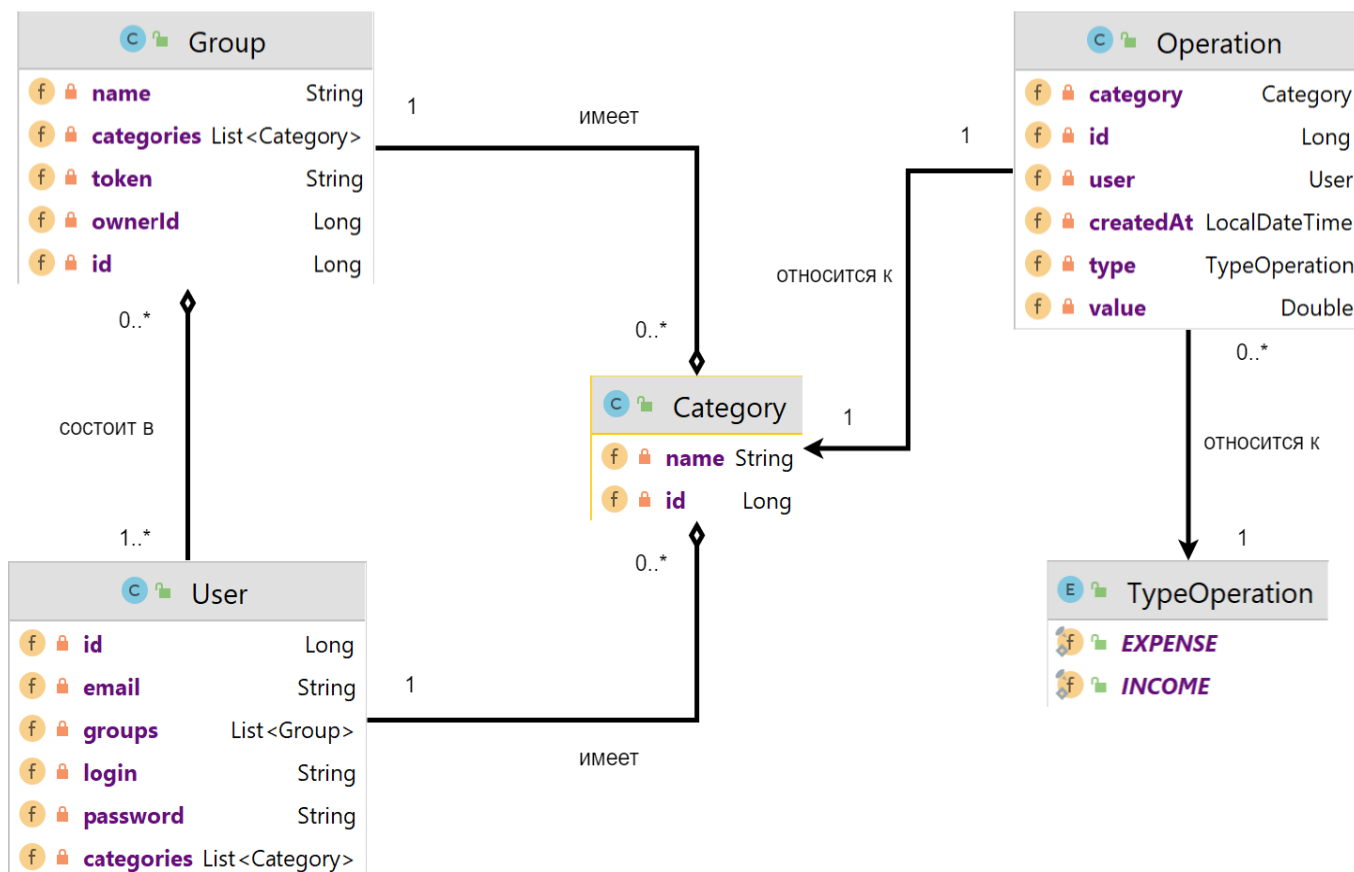


Рисунок 7 - Диаграмма классов

На рисунке 9 продемонстрирована диаграмма классов сущностей, которые представляют из себя объекты бизнес логики приложения. На основе их содержимого (названия, полей) определяется структура базы данных. На диаграмме, кроме таблиц сущностей, также отмечены отношения между ними. На отношениях содержится информация, как можно интерпретировать связь в точки зрения бизнес требования.

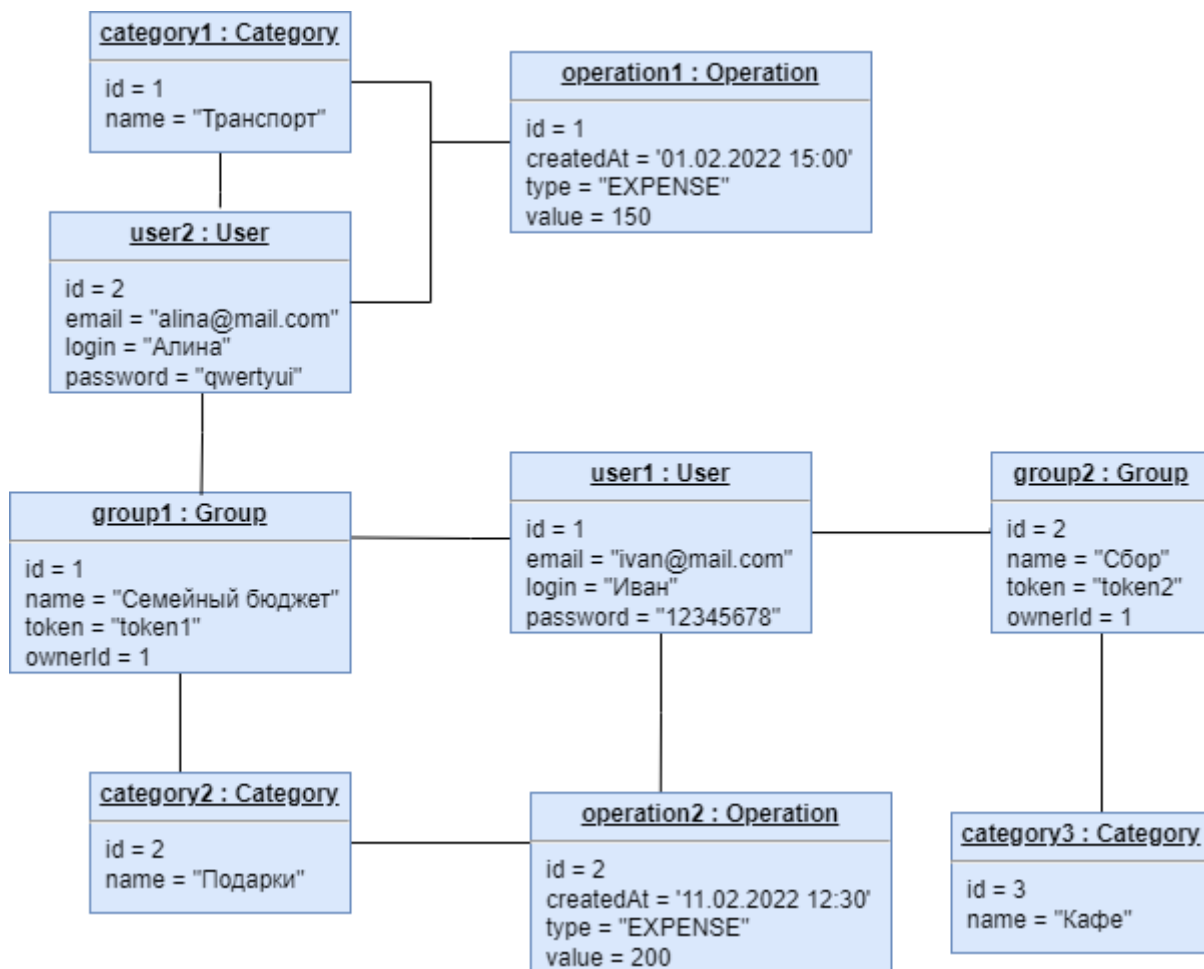


Рисунок 8 - Диаграмма объектов

На рисунке 10 представлена диаграмма объектов сущностей, которая отражает множество экземпляров классов и их отношения в некоторый момент времени.

Объекты на диаграмме:

- user1 и user2: обозначают пользователей
- group1 и group2: обозначают группы, в которых состоят пользователи
- category1, category2 и category3: обозначают категории, которые имеются в группах или личных кошельках пользователей
- operation1 и operation2: обозначают денежные операции, которые производят пользователи в личных кошельках или группах

## 8.2. Модуль бизнес-логики

Модуль бизнес-логики представляет из себя слой приложения, который отвечает за манипулирование данными, подходящих бизнес-требованиям.

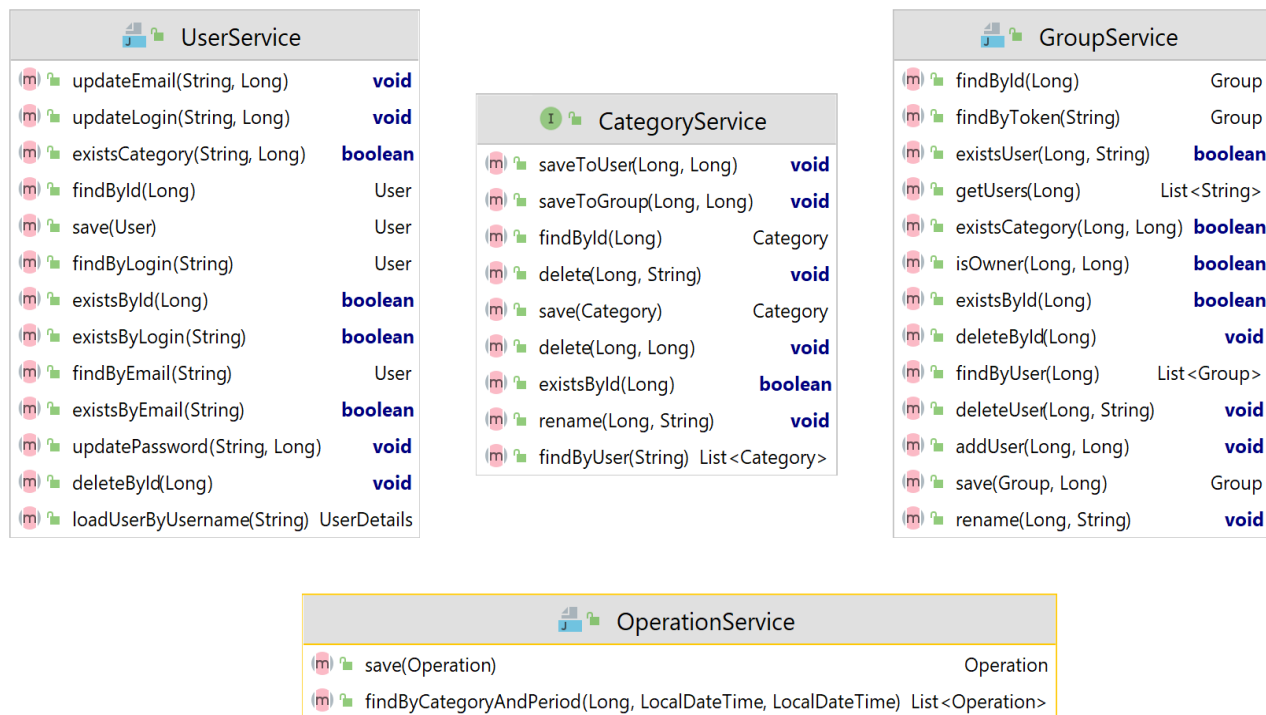


Рисунок 9 - Диаграмма классов сервисов

На рисунке 9 представлена диаграмма классов сервисов, которые находятся в модуле бизнес-логики и содержат логику, которая удовлетворяет бизнес-требованиям приложения.



### 8.3. Модуль web-приложения

Модуль web-приложения представляет из себя слой, который отвечает за доступ к приложению извне. Данный модуль хранить в себе REST-контроллеры, валидацию данных, поступающих в контроллеры и конфигурацию приложения.

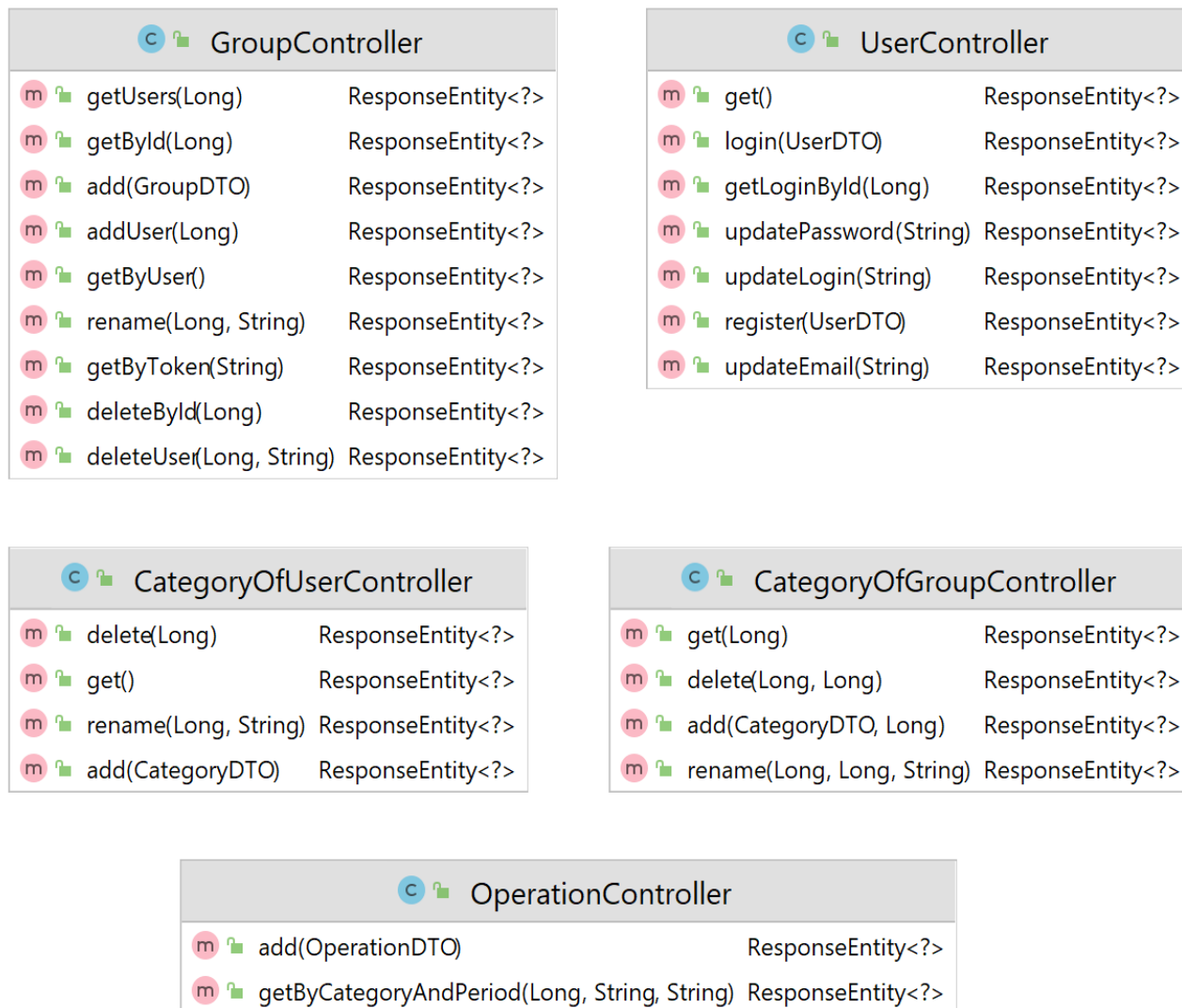
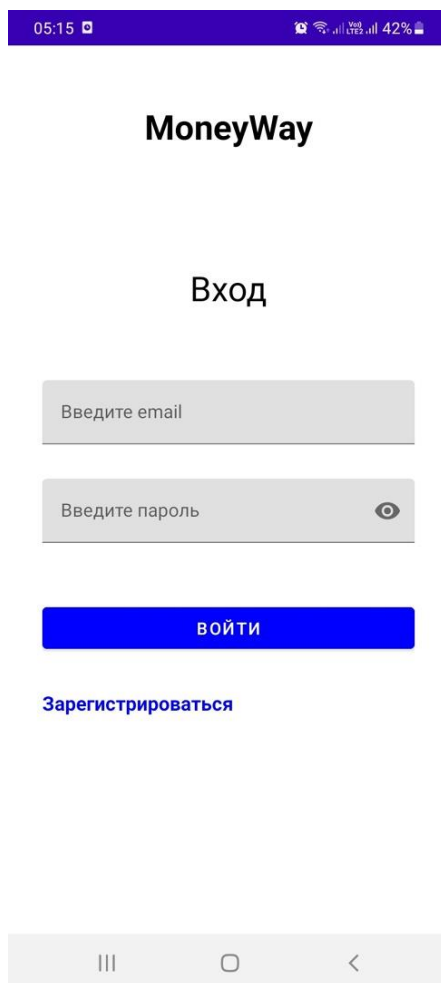


Рисунок 10 - Диаграмма классов контроллеров

Рисунок 10 отображает диаграмму классов REST-контроллеры, которые отвечают за взаимодействие с сервером извне. Контроллерам посылаются запросы в виде JSON, которые они обрабатывает, взаимодействуя с сервисами, и возвращают обратно ответы также в виде JSON.

## 9. Проект интерфейсной части программного средства

### 9.1. Авторизация



05:15

MoneyWay

Вход

Введите email

Введите пароль

ВОЙТИ

[Зарегистрироваться](#)

Рисунок 11 - Страница авторизации

Страница авторизации содержит поля, необходимые для аутентификации пользователя: email и пароля. После успешного входа, пользователь переходит на страницу кошелька. Возможен вариант перехода на страницу регистрации.

## 9.2. Регистрация

05:15

MoneyWay

Регистрация

Введите email

Введите логин

Введите пароль

Повторите пароль

ЗАРЕГИСТРИРОВАТЬСЯ

Войти

Рисунок 12 - Страница регистрации

Страница регистрации содержит поля, необходимые для инициализации пользователя: email, логина и пароля, который для предотвращения ошибки вводится два раза. После успешной регистрации, пользователь переходит на страницу кошелька. Возможен вариант перехода на страницу авторизации.

### 9.3. Кошелёк

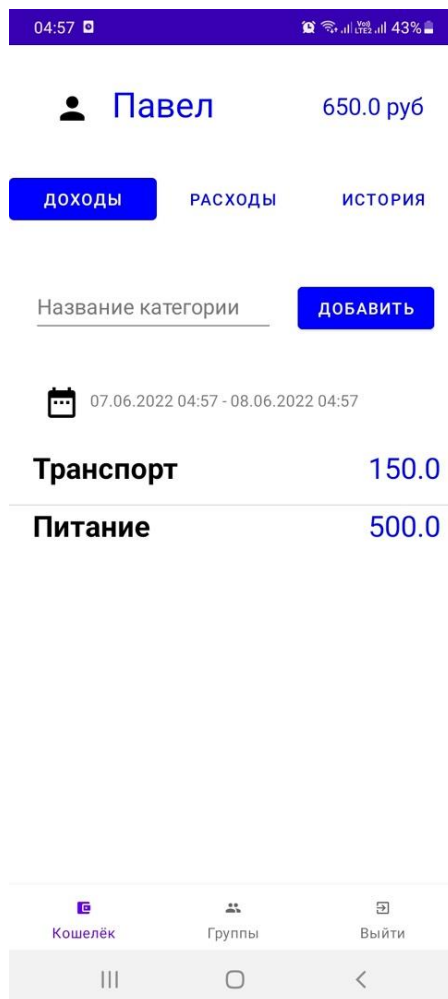


Рисунок 13 - Страница кошелька

Страница кошелька (личного бюджета) – стартовая страница авторизованного пользователя.

Впервые пользователю доступно меню, содержащее кнопки для перехода на страницу кошелька и групп, а также выхода из учётной записи на страницу авторизации.

Страница кошелька содержит кнопку для перехода в профиль, изображённой в виде силуэта человека, имя пользователя (логин) и итоговый счёт на текущий тип операции по заданному промежутку времени (по умолчанию последние сутки). Ниже есть возможность создать новую категорию, после чего идёт выбор периода времени и список доступных пользователю категорий. При нажатии на один из элементов, происходит переход на страницу выбранной категории.

## 9.4. Группы



Рисунок 14 - Страница групп

Страница групп содержит поиск группы по уникальному токену, возможность создания новой группы и список доступных пользователю групп. При нажатии на элемент списка происходит переход на страницу выбранной группы.

## 9.5. Личный кабинет пользователя

05:16 42%

← Павел

Введите email ✓

Введите логин ✓

Введите пароль ✓

Кошелёк Группы Выйти

Рисунок 15 - Страница профиля

Страница профиля содержит имя пользователя (логин) и поля с кнопками для замены данных: email, логина и пароля. Есть возможность перейти обратно в кошелёк по кнопке вблизи имени.

## 9.6. Группа

### 9.6.1. История группы

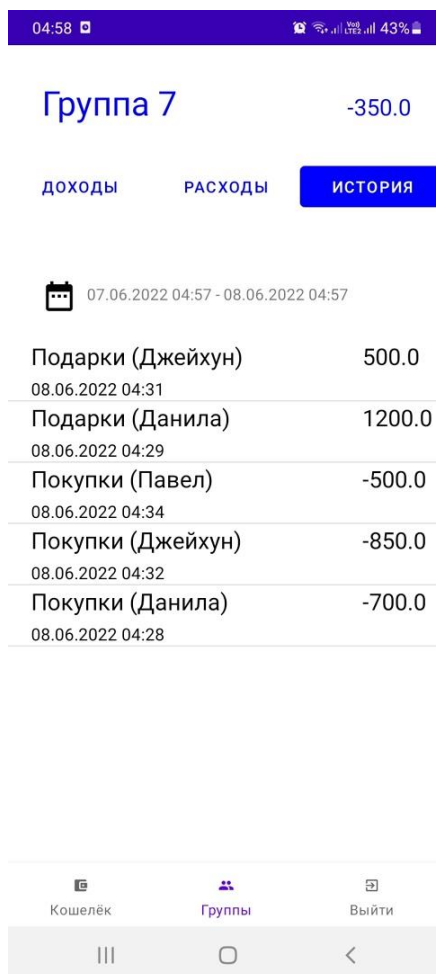


Рисунок 16 - Страница истории группы

Страница истории группы содержит подробную информацию о всех операциях, произведённых в группе, а именно: название категории, имя (логин) пользователя, который записал операцию, дата операции и значение.

## 9.6.2. Для создателя



Рисунок 17 - Страница подробностей группы для создателя

Страница подробностей группы для создателя содержит конфиденциальную информацию о группе, а именно, уникальный токен, который можно скопировать по кнопке, находящейся рядом с надписью. Группу можно переименовать и не только посмотреть участников, но и удалить их. Есть возможность удаления группы.



### 9.6.3. Для участника



Рисунок 18 - Страница подробностей группы для участника

Страница подробностей группы для участника содержит неполную информацию о группе, так как у участника нет прав на конфиденциальные данные группы. Пользователь может посмотреть список участников, а также выйти из группы, после чего перейдёт на страницу групп.

## 9.6.4. Категория



Рисунок 19 - Страница категории

Страница категории содержит кнопку для удаления данной категории. Существует возможность переименовать категорию и добавить для неё операцию.

## 10. Тестирование

### 10.1. Модульное тестирование

Модульное тестирование реализуется с помощью библиотек JUnit и Mockito. Mockito позволяет заглушать поведение классов, от которых зависит тестируемый класс. Для этого перед исполнением основного метода устанавливается поведение функции второстепенного класса, который используется основным.

```
@Test
void findByToken() {
    // setup
    when(groupRepository.findByToken("Right token")).thenReturn(group);

    // test execution
    Group groupByToken = groupService.findByToken("Right token");

    // test check
    assertEquals(group, groupByToken);
    verify(groupRepository).findByToken("Right token");
}
```

Рисунок 20 - Тестирование метода поиска группы по токену

На рисунке 13 продемонстрировано тестирование метода получения свободных номеров в заданном промежутке времени.

Тестирование разделяется на этапы:

- **setup**: установка тестовых переменных и установка предопределённого поведения методов внутренних классов, которые встретятся во время тестирования основной функции. При вызове метода `when` передаётся вызов метода, поведение которого необходимо заглушить. Результат вызова задаётся последующим методом `thenReturn`, которому передаётся список, который необходимо вернуть при вызове ранее обозначенного метода.
- **test execution**: выполнение теста и, как в данном случае, получение результата тестирования (списка номеров)
- **test check**: проверка результата тестирования. Происходит сравнение списков на соответствие. Метод `verify` принимает второстепенный класс, который встретился в реализации и его вызванный метод с переданными параметрами.

## 10.2. Командное тестирование

Для анализа выполнения основных правил юзабилити приложением воспользовались пять команд.

Таблица 1 - Отзывы

| Функционал                          | Команда №1 | Команда №2 | Команда №3 | Команда №4 | Команда №5 |
|-------------------------------------|------------|------------|------------|------------|------------|
| Регистрация                         | +          | +          | +          | +          | +          |
| Авторизация                         | +          | +          | +          | +          | +          |
| Просмотр кошелька                   | +          | +          | +          | +          | +          |
| Добавление категорий                | +          | +          | +          | +          | +          |
| Добавление операций                 | +          | +          | +          | +          | +          |
| Удаление категорий                  | +          | +          | +          | +          | +          |
| Просмотр операций                   | +          | +          | +          | +          | +          |
| Просмотр групп                      | +          | +          | +          | +          | +          |
| Создание группы                     | +          | +          | +          | +          | +          |
| Поиск группы по токену и вступление | +          | +          | +          | +          | +          |

|   |   |   |   |   |   |
|---|---|---|---|---|---|
| Просмотр<br>содержимого<br>группы                 | + | + | + | + | + |
| Добавление<br>и удаление<br>категорий в<br>группе | + | + | + | + | + |
| Добавление<br>операций в<br>группе                | + | + | + | + | + |
| Выход из<br>группы                                | + | + | + | + | + |
| Удаление<br>группы                                | + | + | + | + | + |
| Выход из<br>аккаунта                              | + | + | + | + | + |

## **11. Оценка степени завершённости и перспективы развития проекта**

Приложение отвечает заявленным требованиям и задачам, однако, для поддержания и совершенствования проекта, необходимо проделать работу в направлении оптимизации для более быстрого взаимодействия клиентской части с серверной, а также обработки данных на стороне сервера.

## Заключение

Результатом работы является клиент-серверное мобильное приложение, поддерживаемое операционной системой android и выполняющая функции учёта доходов и расходов, как личных, так и нескольких людей, которые объединяются в группы и имеют отдельный общий кошелёк.

Выполнены следующие задачи:

- Спроектирована и разработана база данных. Развёрнута на удалённом сервере.
- Разработана многомодульная серверная часть. Развёрнут на удалённом сервере.
- Разработана клиентская часть приложения для мобильных устройств под операционной системой android: сверстаны страницы и налажена связь с сервером.

Разработанное приложение отвечает заявленным требованиям. Тестирование проекта окончено удачно.