Минобрнауки россии

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ

ВЫСШЕГО ОБРАЗОВАНИЯ

“ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”

Факультет компьютерных наук

Кафедра программирования и информационных технологий

Ведение группового и индивидуального бюджета

Курсовой проект

*09.03.02 Информационные системы и технологии*

*Программная инженерия в информационных системах*

Зав. кафедрой\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_С. Д. Махортов, д.ф.- м.н.,доцент \_\_.\_.20\_\_

Обучающийся\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Д.А. Змаев, 3 курс, д/о

Обучающийся\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Д.В. Мамонов, 3 курс, д/о

Обучающийся\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_П.А. Смирнов, 3 курс, д/о

Руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_В.С. Тарасов, ст. преподаватель

Руководитель\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_И.В. Клейменов, ассистент

Воронеж 2023

Содержание

[Введение 3](#_Toc130575839)

[1 Постановка задачи 4](#_Toc130575840)

[1.1 Задачи проекта 4](#_Toc130575841)

[2 Анализ предметной области 6](#_Toc130575842)

[2.1 Терминология (глоссарий) предметной области 6](#_Toc130575843)

[2.2 Обзор аналогов 6](#_Toc130575844)

[2.2.1 «Финансы» 6](#_Toc130575845)

[2.2.2 «Журнал расходов» 8](#_Toc130575846)

[2.2.3 «CoinKeeper» 9](#_Toc130575847)

[3 Реализация 19](#_Toc130575848)

[3.1 Средства реализации 21](#_Toc130575849)

[3.2 Реализация Backend 21](#_Toc130575850)

[3.3 Реализация Frontend 26](#_Toc130575851)

[4 Тестирование 26](#_Toc130575852)

Введение

Существует множество вариантов ведения бюджета, как личного, так и нескольких людей (например, семейного): от обычной тетради до современного программного обеспечения.  
Тем не менее, одними из стандартных средств для ведения семейного  
бюджета является таблица Excel или ведения бюджета от руки в тетради.  
Такие способы занимают много времени, а иногда и достаточно сложны для  
человека. Затрагивая тему технического прогресса, надо отметить, что  
технологии заняли важное место в жизни человечества. Сейчас сложно  
обходиться в жизни без телефонов, планшетов, компьютеров, ноутбуков и  
т.д. Мобильное приложение для ведения семейного бюджета всегда  
возможно иметь под рукой и заполнять в любой момент времени, а также  
оно упрощает процессы подсчета и записи расходов и доходов.  
Контролировать доходы и расходы становится удобнее, проще, быстрее,  
программные средства приложения помогут наглядно увидеть отчеты о своем бюджете и выполненных финансовых операциях. К тому же, в ведение бюджета входит такая важная операция, как определенные выплаты по кредиту, которые тоже тяжело рассчитывать в тетрадке, поэтому приложение — помощник должно иметь несложный калькулятор для расчета кредитных платежей. Для достижения поставленных задач сначала надо рассмотреть аналогичные сторонние решения, а затем сформулировать полный список требований к данному приложению

1. Постановка задачи
   1. Задачи проекта

Целью курсовой работы будет создание мобильного приложения для ведения личного и группового бюджета, которое будет отвечать следующим требованиям:

К разрабатываемому приложению выдвинуты следующие требования:

* Обеспечение учета доходов и расходов
* Обеспечение группировки индивидуальных финансовых операций
* Обеспечение создания шаблонов для частых транзакций
* Обеспечение расчета кредита
* Обеспечение создания категорий
* Обеспечение создания отчета
* Обеспечение сохранения отчета в CSV/XLSL формат
* Обеспечение создания групп и приглашения в них пользователей  
  для совместного отслеживания доходов и расходов
  1. Технические требования

Программный продукт должен обеспечить:

* Авторизацию пользователя с использованием email и пароля;
* Шифрование логина и пароля при записи в БД;
* Хранение данных в БД
  1. Требования к интерфейсу

Интерфейс должен быть выполнен в едином для всего приложения стиле с выделяющимися элементами управления и легкочитаемыми надписями. Информация должна находиться в актуальных местах и оставаться понятной. Внешний вид приложения должен адаптироваться под параметра экрана пользователя.

* 1. Задачи, решаемые в процессе разработки

Перед проектом были поставлены следующие задачи:

* Анализ предметной области;
* Анализ аналогов;
* Написание технического задания;
* Проектирование веб-сервиса средствами языка UML;
* Построение use case диаграмм;
* Создание досок Trello, Miro и репозитория GitHub;
* Определение используемой платформы;
* Подключение внешнего модуля для хранения данных;
* Разработка БД;
* Разработка функциональных возможностей приложения;
* Создание макета дизайна и его реализация;
* Реализация интерфейса;
* Проведение тестирования;
* Описание процесса разработки и результата.

1. Анализ предметной области
   1. Терминология (глоссарий) предметной области

* Python - высокоуровневый язык программирования общего  
  назначения с динамической строгой типизацией и автоматическим  
  управлением памятью, ориентированный на повышение  
  производительности разработчика, читаемости кода и его качества,  
  а также на обеспечение переносимости написанных на нём  
  программ.
* Django - свободный фреймворк для веб-приложений на языке  
  Python, использующий шаблон проектирования MVC.  
  PostgreSQL - свободная объектно-реляционная система управления  
  базами данных.
* Docker - программное обеспечение для автоматизации  
  развёртывания и управления приложениями в средах с поддержкой  
  контейнеризации, контейнеризатор приложений.
* 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
* SQLite - компактная встраиваемая СУБД
  1. Обзор аналогов
     1. «Финансы»

Приложение Финансы поможет взять под контроль свои финансы, бюджет и деньги, при этом не отнимет много времени. Вам больше не потребуется заглядывать в кошелек или заходить в свой банковский счет, чтобы узнать Ваше текущее финансовое состояние. Пользуясь приложением Финансы, Вы сможете спокойно совершать траты, и при этом копить и делать

1. Пример интерфейса приложения «Финансы»

Достоинства:

* Учет доходов и расходов
* Выбор валюты
* Категории
* Напоминания
* Графики

Недостатки:

* Отсутствие экспорта в excel
* Отсутствие калькулятора ипотеки/кредита
* Отсутствие сканирования чеков

Приложения «Финансы» имеет оценку в Play Market: 4.9/5 и количество скачиваний равное пяти миллионам.

* + 1. «Журнал расходов»

Журнал расходов - приложение, предназначенное для учета расходов и доходов.  
Приложение позволяет просто и удобно добавить информацию о расходах в момент покупки или получения дохода.  
Удобный и информативный интерфейс поможет контролировать ваши расходы и планировать бюджет.

1. Пример интерфейса приложения «Журнал расходов»

Достоинства:

* Учет доходов и расходов
* Выбор валюты
* Резервная копия данных
* Экспорт в excel
* Пароль

Недостатки:

* Отсутствие калькулятора ипотеки/кредита Отсутствие калькулятора ипотеки/кредита
* Отсутствие категорий
* Отсутствие напоминаний

Приложения «Финансы» имеет оценку в Play Market: 4.6/5 и количество скачиваний равное одному миллиону.

* + 1. «CoinKeeper»

CoinKeeper — самый удобный способ контроля и планирования бюджета, созданный специально для телефонов и планшетов  
Наверняка, вы пробовали другие финансовые менеджеры и бросили их через неделю? Всё ещё пользуетесь сложными таблицами, Excel или тетрадью?

1. Пример интерфейса приложения “CoinKeeper”

Преимущества:

* Учет доходов и расходов
* Выбор валюты
* Категории
* Пароль
* Экспорт в excel (только премиум)
* Парсинг смс
* Напоминания
* Графики

Недостатки:

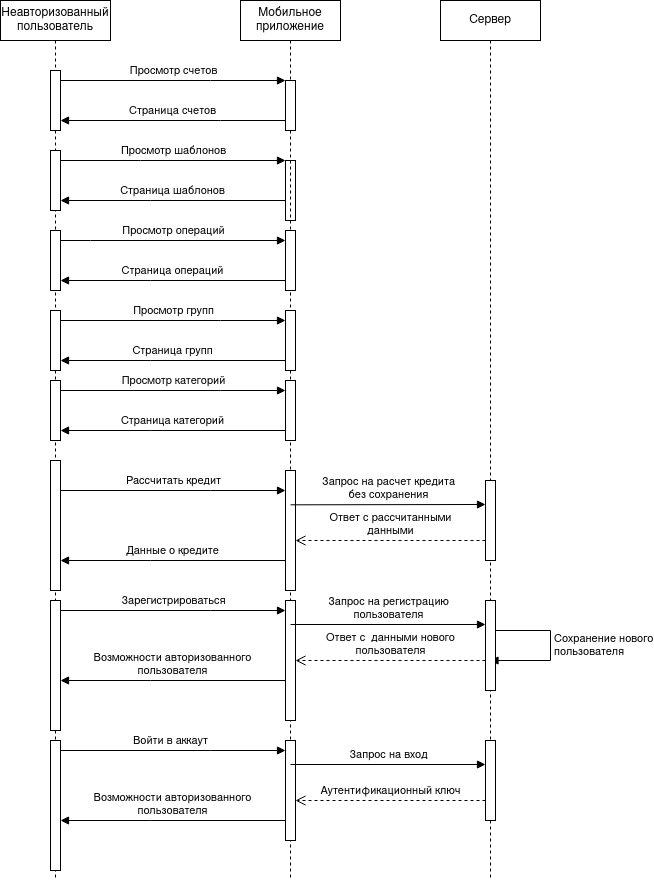
* Реклама подписки при каждом действии
* Неудобный интерфейс
* Отсутствие калькулятора ипотеки/кредита
* Отсутствие сканирования чеков
* Отсутствие напоминаний

Приложения «Финансы» имеет оценку в Play Market: 4.1/5 и количество скачиваний равное одному миллиону

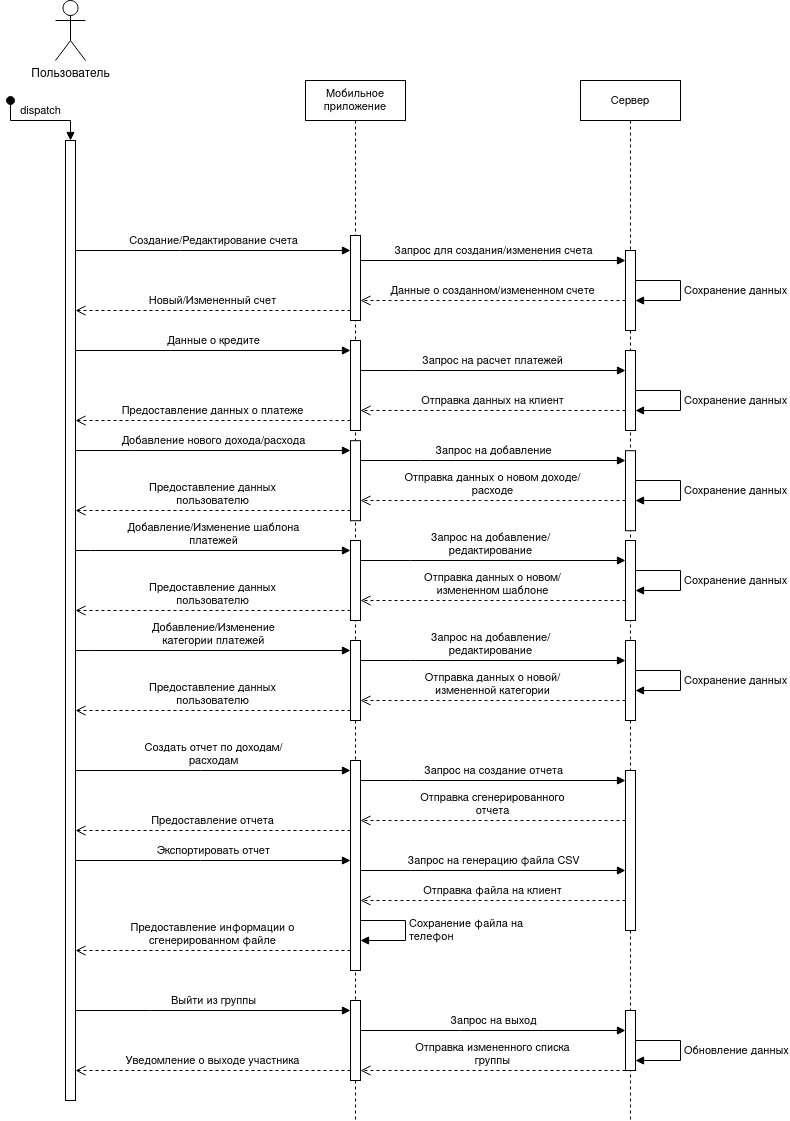
* 1. Моделирование системы
     1. Диграмма UseCase



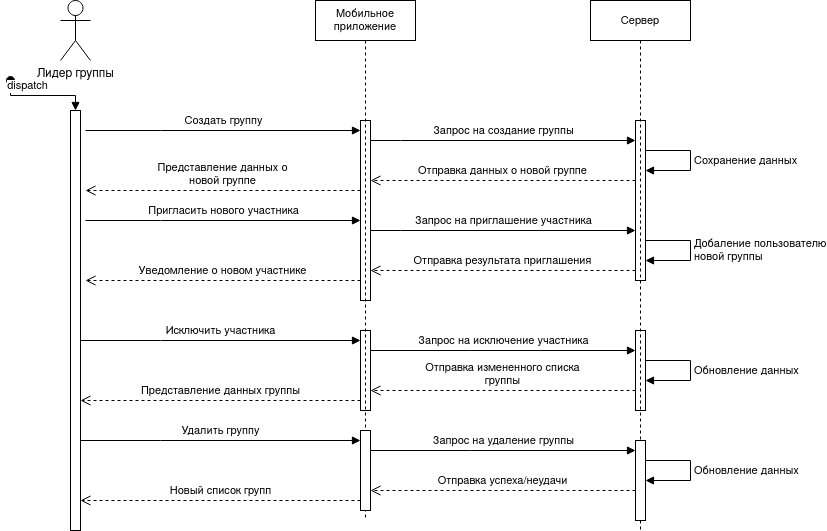
1. Диаграмма UseCase
   * 1. Диаграмма Sequence для неавторизованного пользователя



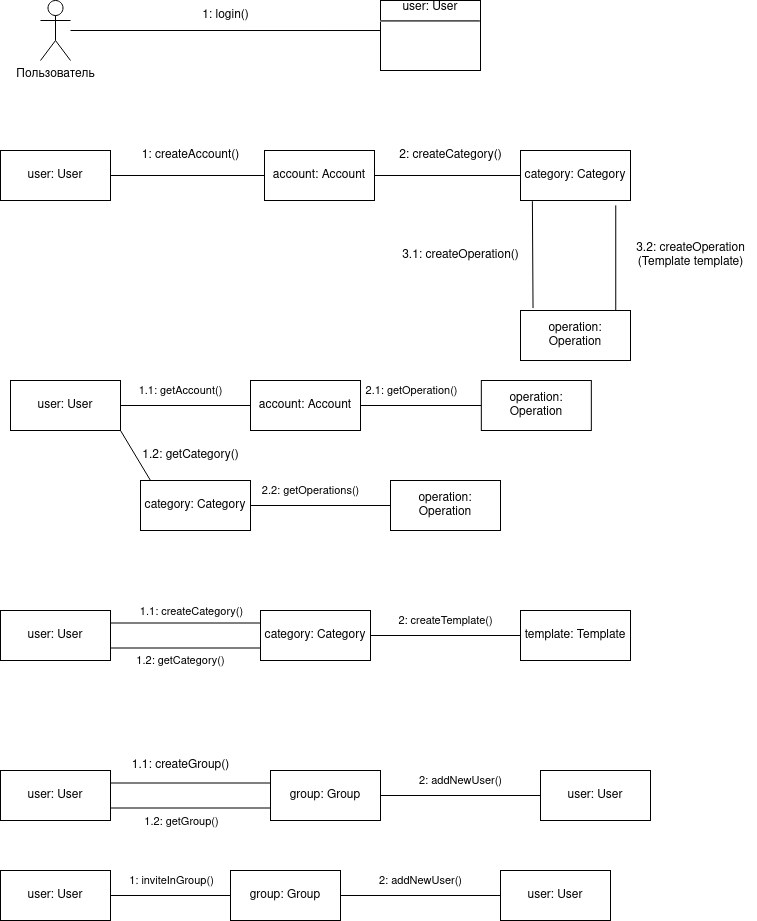
* + 1. Диаграмма Sequence для авторизованного пользователя



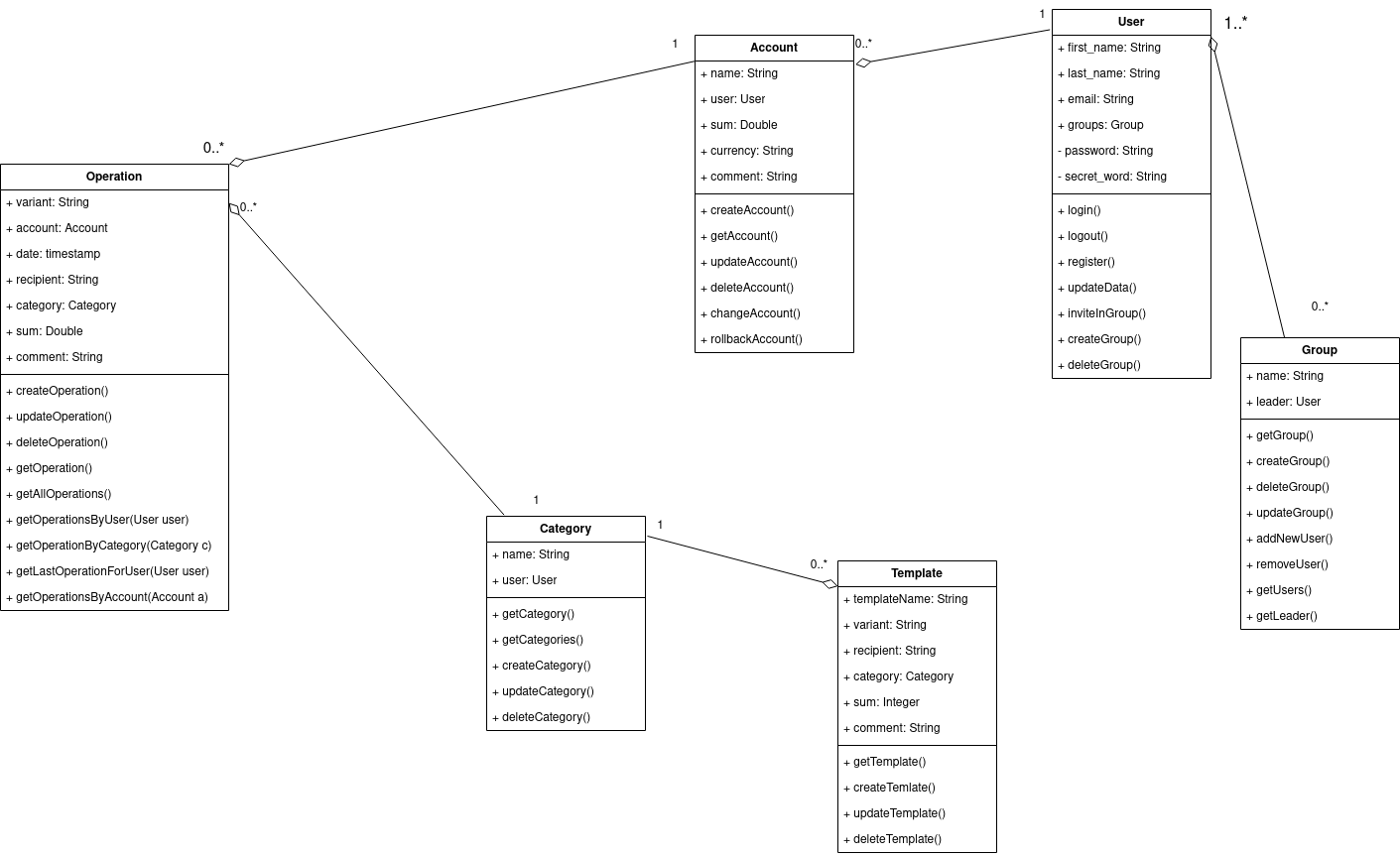
* + 1. Диаграмма Sequence для владельца группы



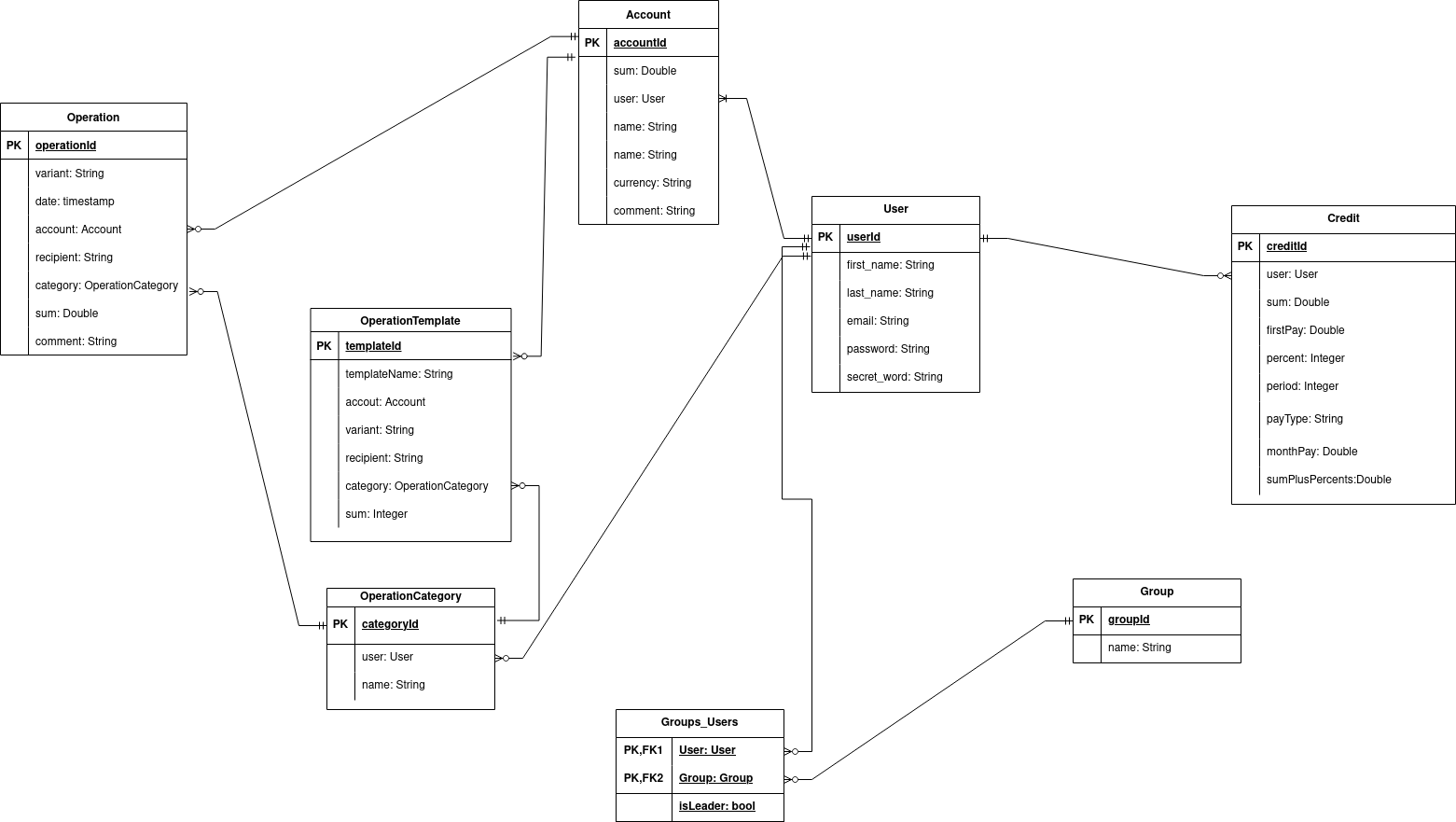
1. Диаграмма Sequence
   * 1. Диаграмма Collaboration



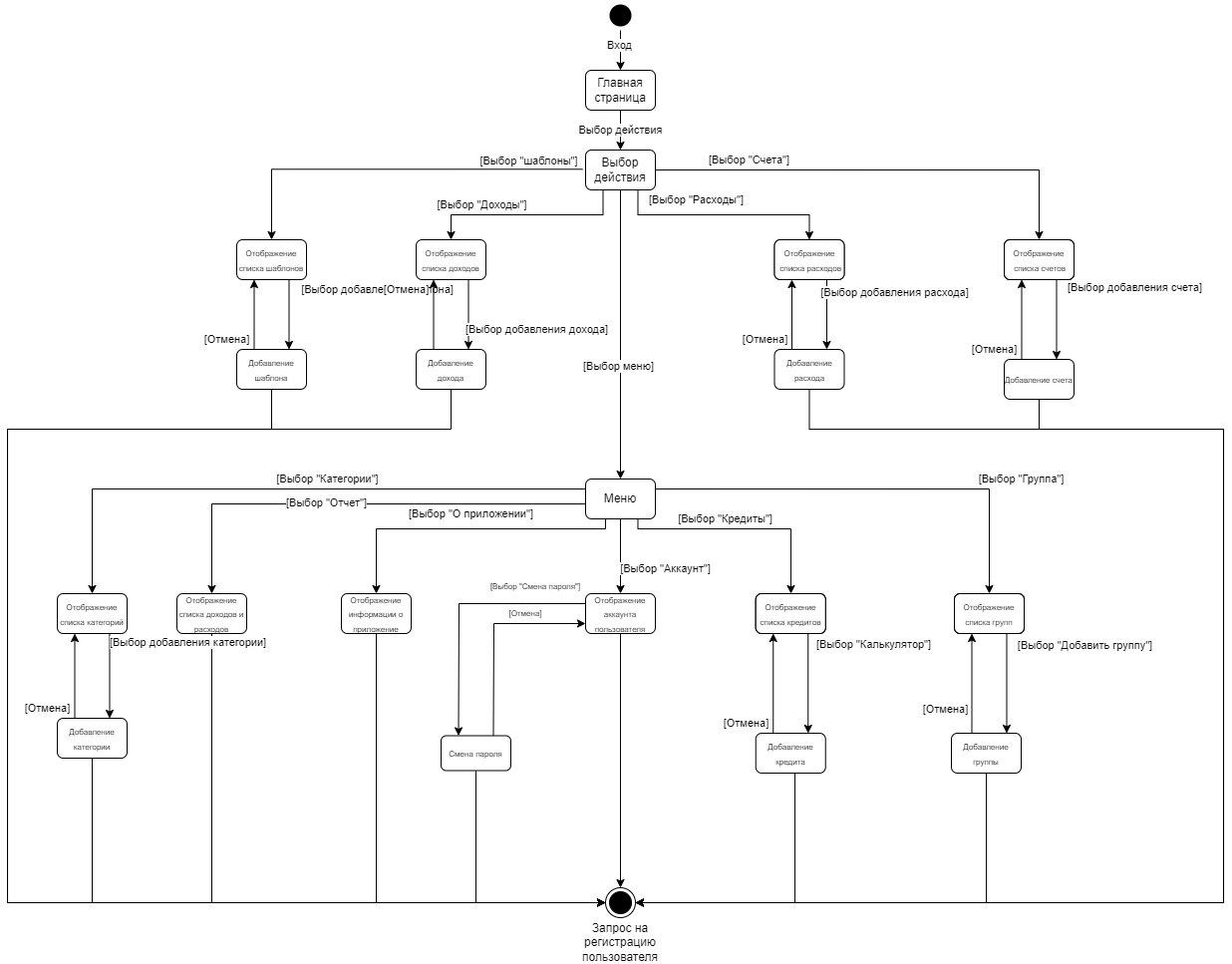
1. Диаграмма Collaboration
   * 1. Диаграмма Class



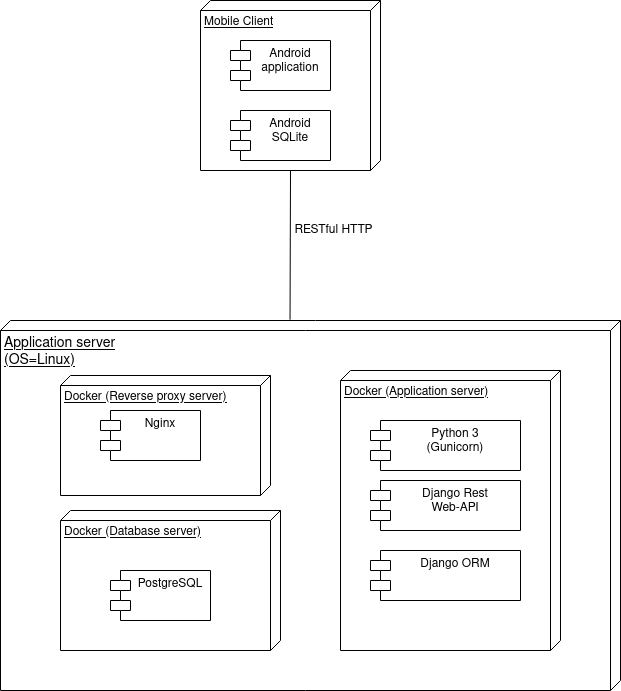
1. Диаграмма Class
   * 1. Диаграмма ER



1. Диаграмма ER
   * 1. Диаграмма Statechart



1. Диаграмма statechart
   * 1. Диаграмма Deployment



1. Диаграмма Deployment
   1. Границы Проекта

Основными границами проекта являются:

* Осуществление учета доходов и расходов;
* Осуществление расчета кредита;
* Осуществление возможности создания шаблонов финансовых операций;
* Осуществление формирования отчета об осуществленных финансовых операциях;
* Осуществление создания, редактирования и удаления категорий;
* Исключение возможности регистрации иначе, кроме как с использованием email;
* Ограниченные сроки разработки и запуска проекта.

На фоне основных сценариев и функциональных требований, можно сформулировать то, что точно выходит за основные границы проекта:

* Обзор банков и их предложений по кредиту, ипотеке, рассрочке или иным формам финансовых отношений;
* Оформление кредита, ипотеки, рассрочки или иных форм финансовых отношений;
* Мониторинг курса валют
* Работа с криптовалютами

1. Реализация
   1. Средства реализации

Для разработки приложения будет использоваться:

* Python - высокоуровневый язык программирования общего  
  назначения с динамической строгой типизацией и автоматическим  
  управлением памятью, ориентированный на повышение  
  производительности разработчика, читаемости кода и его качества,  
  а также на обеспечение переносимости написанных на нём  
  программ.
* PostgreSQL — объектно-реляционная система управления базами данных. Является продуктом с открытым исходным кодом, который поддерживается многими серверами, в связи с чем и был выбран
* Swagger — это фреймворк для спецификации REST API

Инструментарий для ведения документации:

* Miro — платформа для совместной работы распределенных команд
* Draw.io — это сервис, предназначенный для формирования диаграмм и схем

Дополнительный инструментарий:

* Git — распределённая система управления версиями.
* GitHub — крупнейший веб-сервис для хостинга IT-проектов и их совместной разработки. Веб-сервис основан на системе контроля версий Git
* Trello — визуальный инструмент, обеспечивающий эффективность командной работы на любом проекте
  1. Реализация Backend

Серверное приложение построено на архитектуре MVC (Model-View-Controller), так как она является одной из самых распространенных для построения приложений и данную архитектуру веб-фреймворк Django предлагает по умолчанию.

Вся логика приложения разбита на отдельные пакеты — модули, каждый из которых представляет свое отдельное приложение MVC:

* users — приложение, отвечающее за функциональность для пользователей;
* operations — пакет, отвечающий за функциональность, связанную с финансовыми операциями, счетами, категориями операций, шаблонами и кредитами;
* groups — приложение, отвечающее за функциональность для работы с группами пользователей
  + 1. Модуль — приложение users

В данном приложении имеется следующая структура:

* models.py: в этом файле определяются классы модели профиля пользователя приложения и модели, отвечающей за сопоставление пользователей группам;
* serializers.py: данный файл предоставляет классы, на основе которых происходит сериализация/десериализация моделей в JSON формат;
* views.py: в этом файле имеется класс UserProfileViewSet, который является контроллером для данного приложения и содержит в себе переопределенные базовые методы создания, редактирования, а также дополнительные функции-представления для работы с пользователями;
* apps.py: данный файл хранит в себе конфигурацию для моделей текущего модуля.

«Подумать над диаграммами классов»

В профиле пользователя имеется внешний ключ на модель User, которую предоставляет веб-фреймворк Django как модель для аутентификации в приложение. Она содержит следующие используемые поля:

* last\_name: фамилия пользователя;
* first\_name: имя пользователя;
* username: логин, под которым происходит аутентификация в систему;
* password: пароль, хранящийся в базе данных в зашифрованном виде.

Дополнительные функции-представления в модуле добавляют следующие возможности:

* Получить информацию о зарегистрированном в текущий момент в системе пользователе (функция me, тип запроса GET).
* Получить список счетов пользователя по его первичному ключу в базе данных (функция accounts, тип запроса GET).
* Получить список всех финансовых операций пользователя по его первичному ключу, также если пользователь состоит в группе, то можно получить список операций, начиная с той, которую он добавил первой после вступления в группу ( функция operations, тип запроса GET).
* Получить список кредитов зарегистрированного в текущий момент пользователя (функция credits, тип запроса GET).
* Получить список групп, в которых состоит зарегистрированный в текущий момент пользователь (функция groups, тип запроса GET).
* Сменить свой пароль для входа в приложение (функция reset\_password, тип запроса POST).
  + 1. Модуль — приложение operations

В данном приложении имеется следующая структура:

* models.py: в этом файле определяются классы моделей счета пользователя, категории финансовых операций, шаблона создания операции, самой финансовой операции пользователя и рассчитанного кредитного платежа;
* serializers.py: данный файл предоставляет классы, на основе которых происходит сериализация/десериализация моделей в JSON формат;
* views.py: в этом файле имеются классы, представляющие собой контроллеры для взаимодействия с моделями текущего модуля;
* services.py: этот файл содержит себе бизнес-логику, предоставляющую возможность получения операций, счетов, изменения счета при добавлении новой операции, генерации и сохранения в CSV формат и расчета кредитного платежа;
* apps.py: данный файл хранит в себе конфигурацию для моделей текущего модуля.

Каждый контроллер имеет встроенные функции-представления для получения всех записей модели из базы данных, получения записи по ее первичному ключу, редактирование и удаление записи по ее первичному ключу. При этом они реализуют дополнительные возможности:

* Получить операции зарегистрированного пользователя по первичному ключу категории операций (контроллер OperationCategoryViewSet, функция operations, тип запроса GET).
* Сгенерировать отчет по всем операциям зарегистрированного пользователя без сохранения на сервере и получить его в JSON формате (контроллер OperationViewSet, функция report, тип запроса GET).
* Сохранить сгенерированный отчет на сервере в формат CSV ( контроллер OperationViewSet, функция save\_report, тип запроса POST).
* Отправить сохраненный файл с отчетом по HTTP (контроллер OperationViewSet, функция send\_report, тип запроса GET).
* Получить операции зарегистрированного пользователя по первичному ключу счета (контроллер AccountViewSet, функция operations, тип запроса GET).
* Рассчитать кредит для неавторизованного пользователя без сохранения результата в базе данных (контроллер CreditPayViewSet, функция calc\_credit, тип запроса GET).
  + 1. Модуль — приложение groups

В данном приложении имеется следующая структура:

* models.py: в этом файле определяются класс модели группы пользователей;
* serializers.py: данный файл предоставляет классы, на основе которых происходит сериализация/десериализация модели, а также тела запроса/ответа в JSON формат;
* views.py: в этом файле имеется класс GroupViewSet, представляющий собой контроллер для взаимодействия с моделью группы;
* services.py: этот файл содержит себе бизнес-логику, предоставляющую возможности создания новой группы, добавления и исключения пользователя, выборки операций пользователей для просмотра в группе, получения списка пользователей групп и роспуска группы;
* apps.py: данный файл хранит в себе конфигурацию для модели текущего модуля.

Контроллер приложения предоставляет переопределенные функции создания и удаления группы, а также дополнительные функции — представления:

* Получить список пользователей группы по ее первичному ключу в базе данных (функция users, тип запроса GET).
* Проверить, является ли зарегистрированный пользователь лидером группы с определенным первичным ключом в базе данных (функция is\_leader, тип запроса GET).
* Добавить нового пользователя в группу по ее определенному первичному ключу, при этом в теле запроса указывается логин-email пользователя в JSON формате (функция add\_user, тип запроса POST).
* Исключить пользователя из группы по ее определенному первичному ключу, при этом в теле запроса указывается id пользователя в базе данных в JSON формате (функция remove\_user, тип запроса POST).
* Выйти зарегистрированному пользователю из группы по ее первичному ключу (функция exit\_from\_group, тип запроса POST).
  + 1. Аутентификация и авторизация

В качестве системы аутентификации в приложении используется внутренняя реализация в веб-фрейворке и дополнительный пакет Djoser, предоставляющий интерфейс для аутентификации с помощью JWT-токенов. При запросе на вход пользователя в приложение ему в теле ответа отправляется токен, который он должен указывать в заголовках к запросам на сервер. Авторизация по токену присутствует во всех контроллерах кроме контроллера взаимодействия с кредитными платежами, так как рассчитать платеж имеет возможность и неавторизованный пользователь.

* 1. Реализация Frontend

1. Тестирование

ЗАКЛЮЧЕНИЕ

В ходе выполнения курсового проекта командой было разработано мобильное приложение учета финансов с возможностью формирования отчетов и расчета кредита, соответствующее поставленным перед проектом задачам.

В начале разработки был проведен анализ предметной области, определены основные требования к разрабатываемой системе, определены основные сценарии приложения.

По результатам разработки были проведены тесты с целью выявления ошибок в работе приложения.

В процессе работы были реализованы следующие задачи проекта:

* Обеспечение учета доходов и расходов
* Обеспечение группировки индивидуальных финансовых операций
* Обеспечение создания шаблонов для частых транзакций
* Обеспечение расчета кредита
* Обеспечение создания категорий
* Обеспечение создания отчета
* Обеспечение сохранения отчета в CSV формат
* Обеспечение создания групп и приглашения в них пользователей  
  для совместного отслеживания доходов и расходов

Список используемых источников