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

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

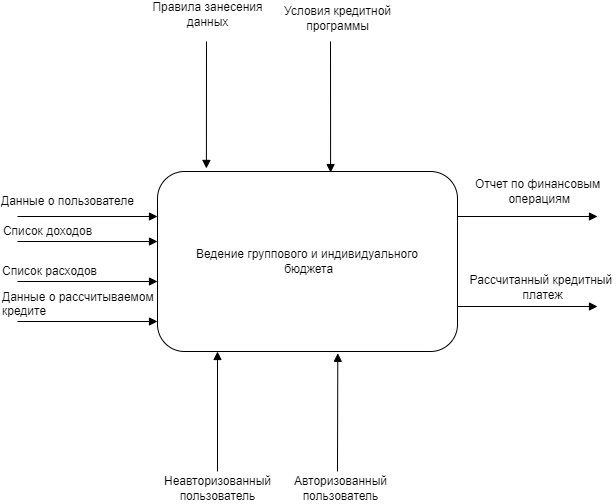
1. Постановка задачи
   1. Цели создания системы

Целью курсовой работы является создание мобильного приложения для ведения личного и группового бюджета.

* 1. Требования к разрабатываемой системе
* Обеспечение учета доходов и расходов
* Обеспечение группировки индивидуальных финансовых операций
* Использование протокола передачи данных HTTP
* Построение приложения на трехуровневой архитектуре
  1. Задачи проекта
     1. Для авторизованного пользователя
* Ведение учета доходов и расходов
* Управление шаблонами для частых операций
* Расчет ежемесячного кредитного платежа
* Управление категориями финансовых операций
* Создание и сохранение отчета в CSV формат
* Формирование групп и приглашение в них пользователей для совместного отслеживания доходов и расходов
  + 1. Для неавторизованного пользователя
* Возможность просмотра экранов приложения для ознакомления
* Расчет ежемесячного кредитного платежа
  + 1. Для создателя группы
* Возможность создания, удаления группы
* Приглашение участников в группу
* Исключение участников из группы

1. Анализ предметной области
   1. Терминология

* Мобильное приложение — программное обеспечение, предназначенное для работы на смартфонах, планшетах и других мобильных устройствах, разработанное для конкретной платформы (iOS, Android, WindowsPhone и т. д.).
* Android-приложение — программное обеспечение, предназначенное для работы на смартфонах, планшетах и других мобильных устройствах, разработанное для платформы Android.
* Клиент — это аппаратный или программный компонент вычислительной системы, посылающий запросы серверу.
* Сервер — выделенный или специализированный компьютер для выполнения сервисного программного обеспечения.
* База данных — это упорядоченный набор структурированной информации или данных, которые обычно хранятся в электронном виде в компьютерной системе. База данных обычно управляется системой управления базами данных (СУБД).
* HTTP — это протокол, позволяющий получать различные ресурсы, например HTML-документы. Протокол HTTP лежит в основе обмена данными в Интернете.
* SQL-запросы — это наборы команд для работы с реляционными базами данных.
* Дизайн-макет — это схематичное изображение финальной идеи с указанием всех деталей. В нем указываются концепция, шрифты, тексты, изображения, расположение всех элементов и общая картина продукта.
* Аутентификация — процедура проверки подлинности, например, проверка подлинности пользователя путем сравнения введенного им пароля с паролем, сохраненным в базе данных.
* Авторизация — предоставление определенному лицу или группе лиц прав на выполнение определенных действий.
* Android — это операционная система с открытым исходным кодом, созданная для мобильных устройств на основе модифицированного ядра Linux.
* Фреймворк — программное обеспечение, облегчающее разработку и объединение разных компонентов большого программного проекта.
* SQL-инъекция — внедрении в запрос произвольного SQL-кода, который может повредить данные, хранящиеся в БД или предоставить доступ к ним.
* HTTPS — расширение протокола HTTP для поддержки шифрования в целях повышения безопасности.
* Пользователь — человек, который использует приложение.
* Аккаунт или учетная запись — это персональная страница пользователя или личный кабинет, который создается после регистрации на сайте.
* Frontend — клиентская сторона пользовательского интерфейса к программно-аппаратной части сервиса.
* Backend — программно-аппаратная часть сервиса, отвечающая за функционирование его внутренней части.
* REST — архитектурный стиль взаимодействия компонентов распределённого приложения в сети.
* API — описание взаимодействия одной компьютерной программы с другой.
  1. Обзор аналогов
* фыв
  1. Моделирование системы
     1. Диаграмма в стиле методологии IDEF0



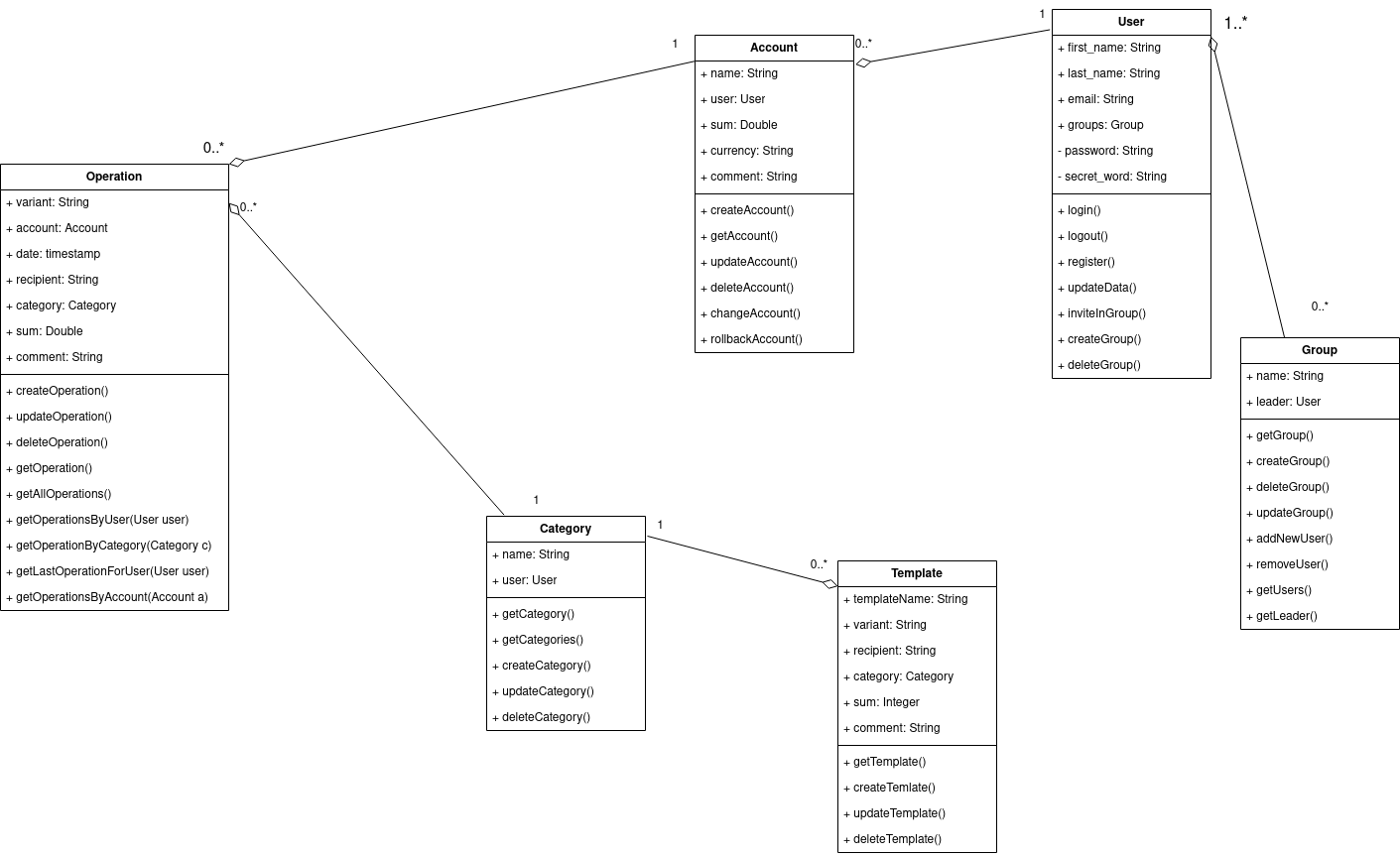
1. Диаграмма в стиле методологии IDEF0
   * 1. Диаграмма прецедентов

Рассмотрим полную диаграмму для использования приложения разными типами пользователей (неавторизованный, авторизованный, исполнитель). В данном случае необходимость составления диаграммы прецедентов продиктована прежде всего тем, что use-case диаграмма — это инструмент для моделирования системы и понимания ее функциональности и потребностей пользователей. Они помогают в определении основных действий, которые пользователь должен совершить в системе, чтобы достичь определенных целей. Они также позволяют определить возможные риски и проблемы, которые могут возникнуть в ходе использования системы.



1. Use-Case диаграмма пользования приложением
   * 1. Диаграммы классов

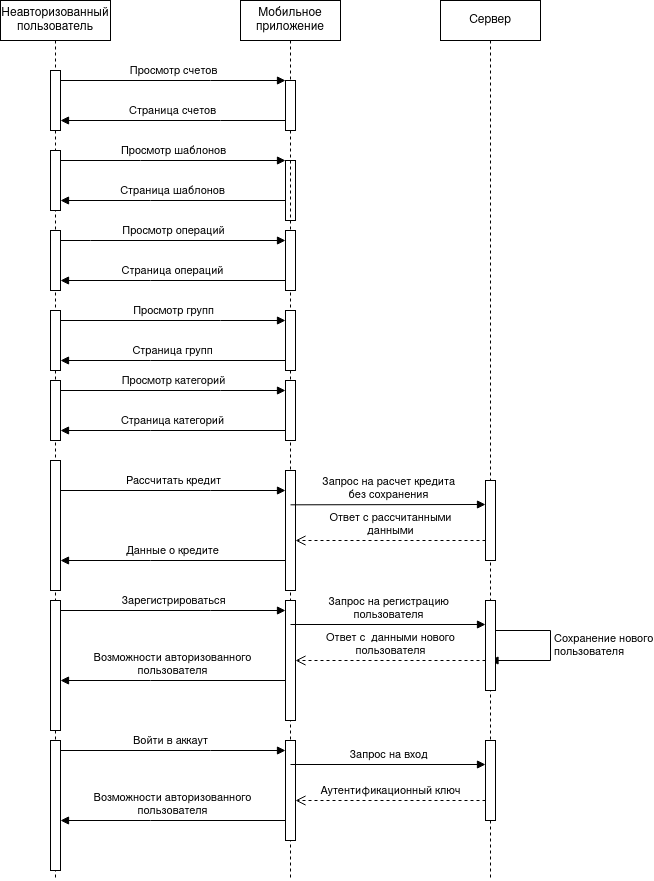
В данном пункте будут рассмотрены диаграммы классов-сущностей серверной части приложения. Диаграмма классов позволяет легко понять структуру проекта и отношения между классами. Это позволяет всем участникам проекта быть на одной волне, что позволяет сократить количество ошибок и повторных разъяснений в процессе разработки, помимо этого классы, которые созданы на диаграмме, обычно хорошо описываются, что позволяет легче отлаживать код и тестировать приложение в целом.



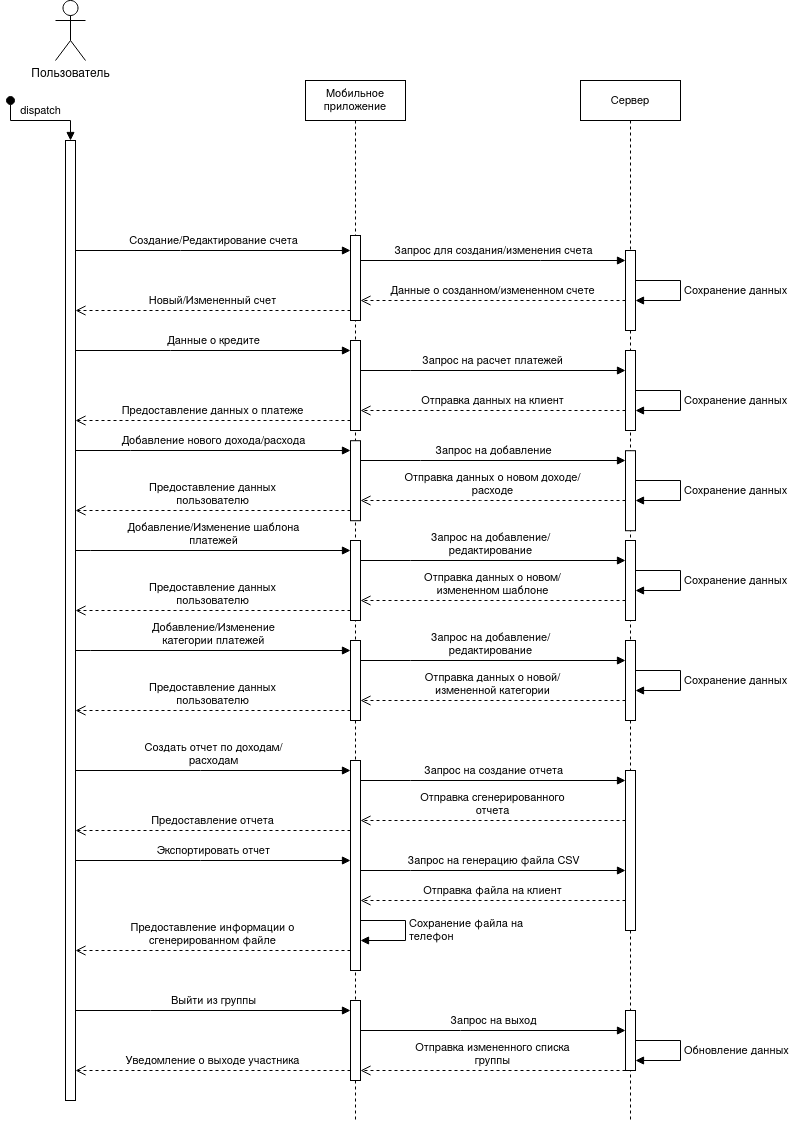
1. Диаграмма классов-сущностей серверной части приложения.
   * 1. Диаграммы последовательности

Диаграмма последовательности является важным инструментом для проекта, который помогает более глубоко понимать процесс, улучшать его эффективность и упрощать взаимодействие.

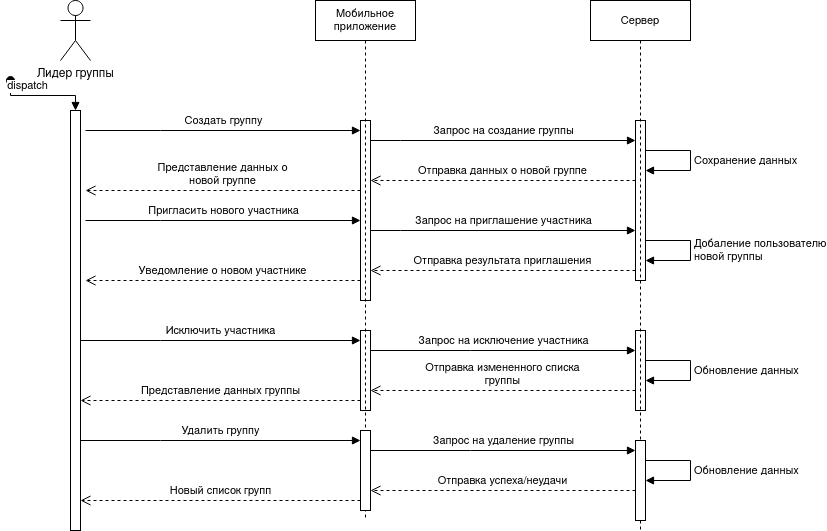
Рассмотрим диаграмму последовательности для пользователей разных типов:



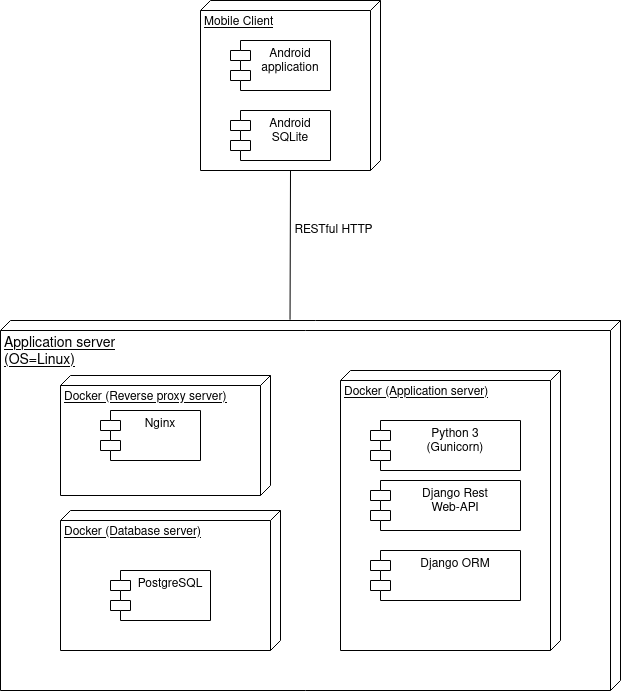
1. Диаграмма последовательности неавторизованного пользователя



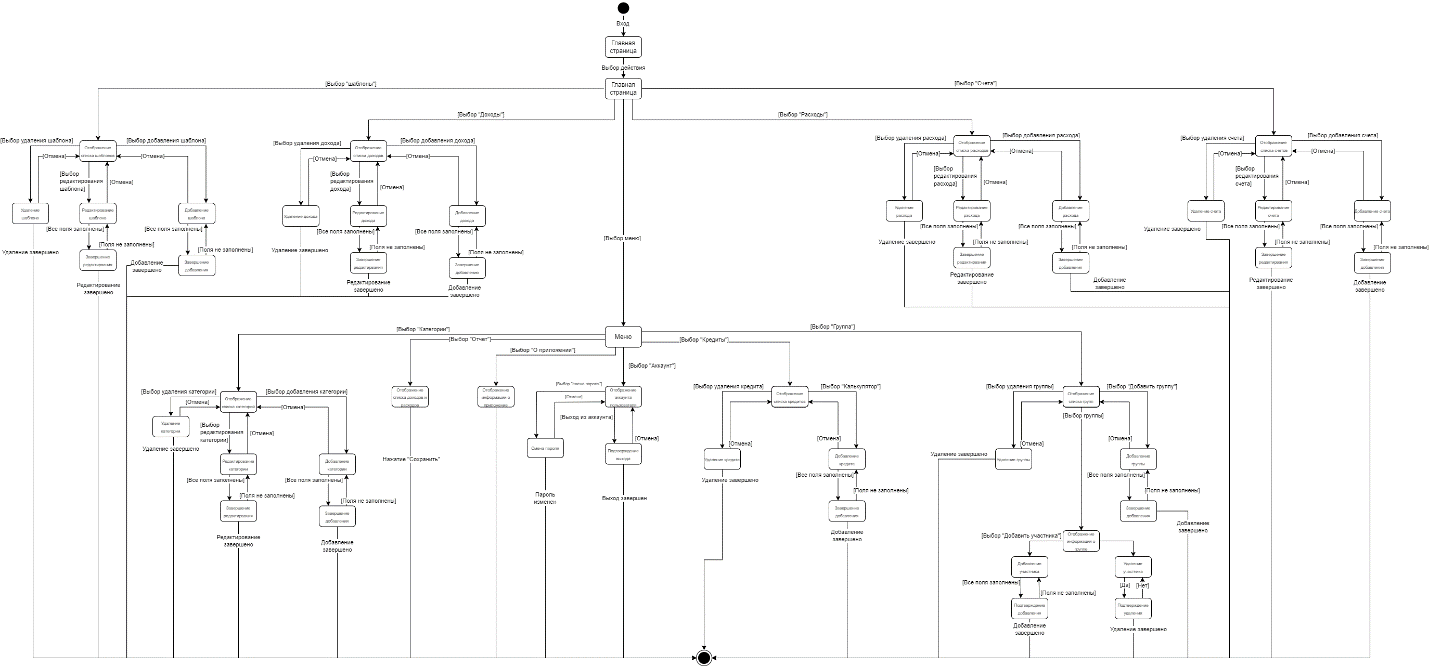
1. Диаграмма последовательности авторизированного пользователя



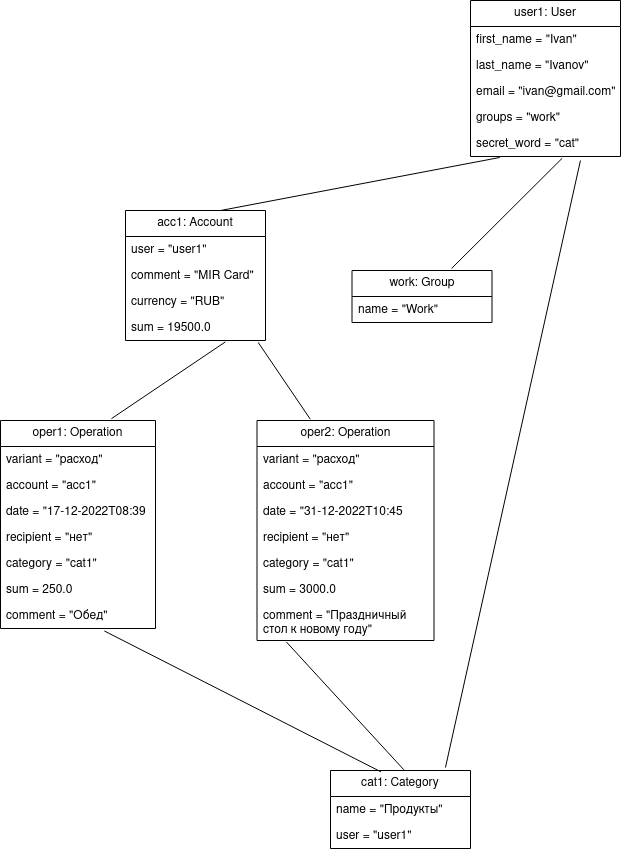
1. Диаграмма последовательности создателя группы
   * 1. Диаграмма развертывания



1. Диаграмма развертывания приложения
   * 1. Диаграммы состояния



1. Диаграмма состояния авторизованного пользователя
2. Диаграмма состояния неавторизованного пользователя
3. Диаграмма состояния авторизованного пользователя
   * 1. Диаграмма объектов



1. Диаграмма объектов
2. Реализация
   1. Средства реализации

Ниже приведен перечень используемых технологий, который в ходе разработки может расширяться.

Backend

* Python
* Django
* PostgreSQL
* Docker

Frontend:

* Android SDK
* Kotlin

Инструменты для ведения документации:

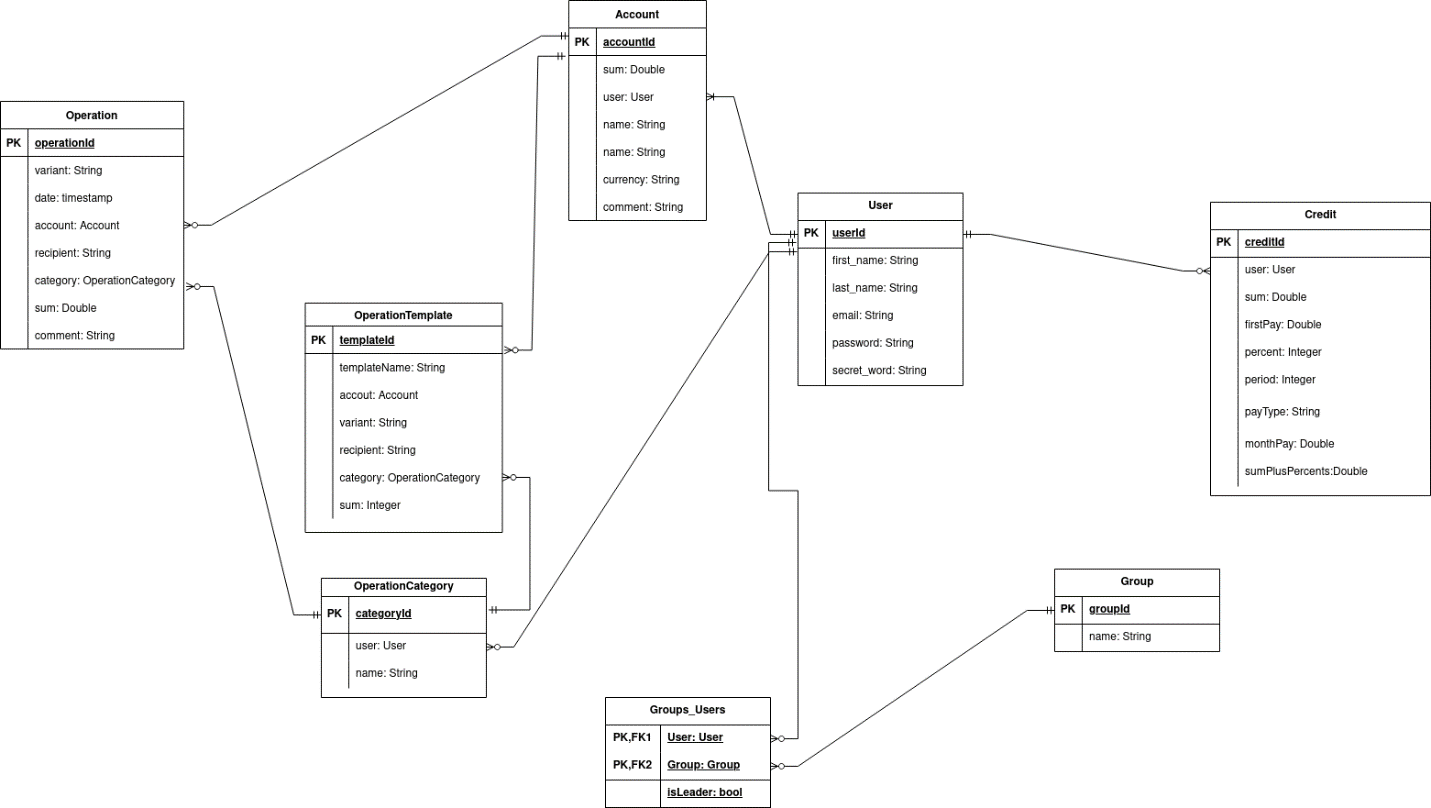
* Miro
* Swagger
* Draw.io
* Ramus
* Figma

Дополнительный инструментарий:

* Git
* GitHub
* Trello
  1. Реализация базы данных

Для хранения данных была выбрана база данных PostgreSQL. Она является продуктом с открытым исходным кодом, который поддерживается многими серверами. Присутствует поддержка различных типов данных. PostgreSQL поддерживает множественные типы данных, такие как числа разной точности, тексты с различными кодировками, изображения, звуки, видео, XML-документы, JSON-объекты и многие другие.

* + 1. ER-диаграмма



1. ER-диаграмма базы данных
   1. Реализация клиентской части

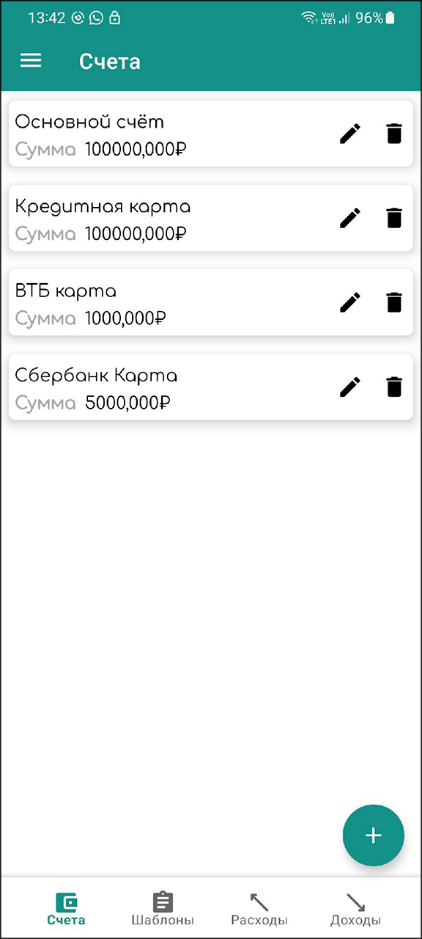
Мобильное приложение построено на архитектурном паттерне MVVM (Model-View-ViewModel), т.к. он позволяет разделить ответственность между компонентами, улучшить тестирование, управлять жизненным циклом компонентов, обеспечить масштабируемость и повысить эффективность разработки мобильного приложения.

Приложение разбито по пакетам:

* api – хранит в себе еще два пакета: service и model. Service отвечает за описание интерфейсов для работы REST API, которая включает в себя методы для взаимодействия с сервером, а также включает вспомогательный объект для создания экземпляра класса. В пакете model хранятся структуры данных, которые отвечают за сериализацию и десериализацию данных из HTTP-запросов и ответов сервера;
* repository - используются для абстрагирования доступа к сетевым ресурсам, которые могут быть использованы для получения, отправки, обновления и удаления данных на сервере;
* view - отвечает за отображение данных на экране устройства и взаимодействие с пользователем;
* viewmodel - отвечает за управление данными и бизнес-логикой приложения.
  + 1. Экраны счетов

На экране «Счета» авторизованный пользователь имеет возможность просматривать уже добавленные счета, редактировать и удалять их, а также создать новый при нажатии на кнопку.

При первой регистрации пользователя автоматически создается счет «Первоначальный счет».



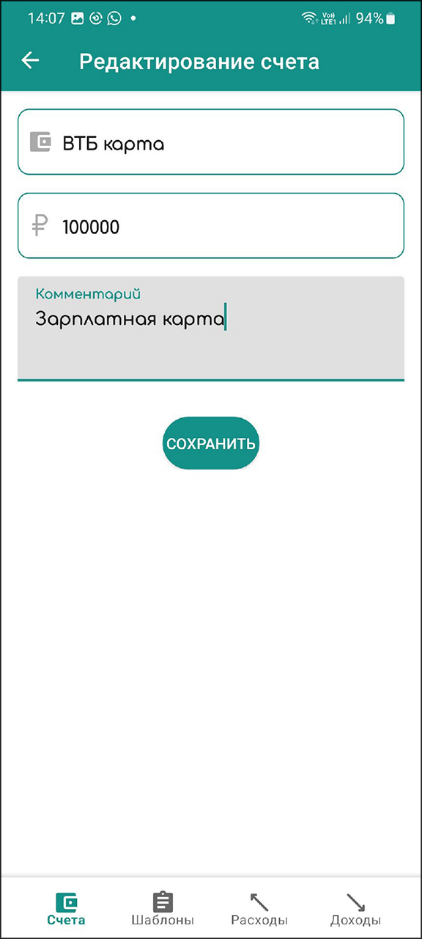
1. Экран счетов

На экране «Счет», необходимого для создания нового счета, отображены поля с название счета, изначальной суммой и комментарием, ниже расположена кнопка для сохранения.



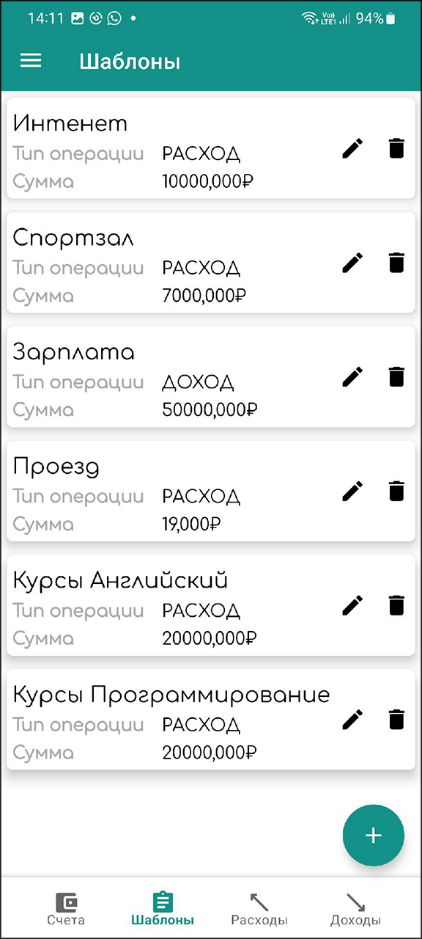
1. Экран добавления счета

На экране «Редактирование счета» отображены поля с название счета, изначальной суммой и комментарием, ниже расположена кнопка для сохранения. Данный экран отображается после нажатия кнопки с изображением ручки на экране «Счета».



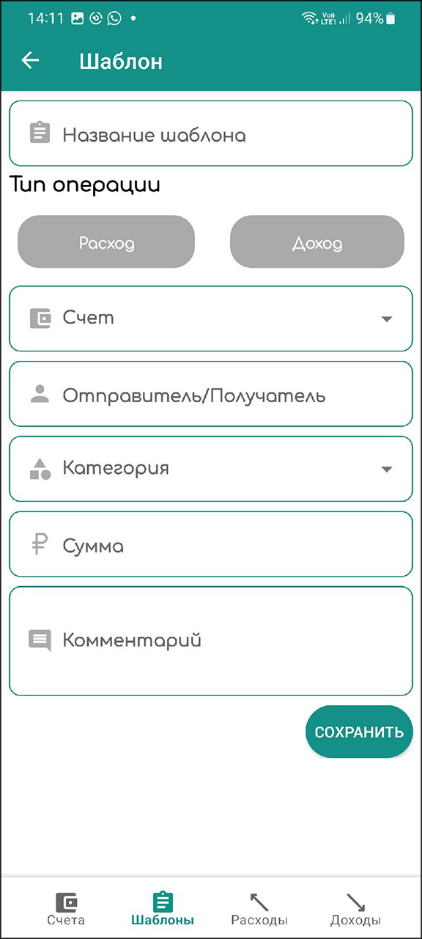
1. Экран редактирования счета
   * 1. Экраны шаблонов

На экране «Шаблоны» авторизованный пользователь имеет возможность просматривать уже добавленные шаблоны, редактировать и удалять их, а также создать новый при нажатии на кнопку.



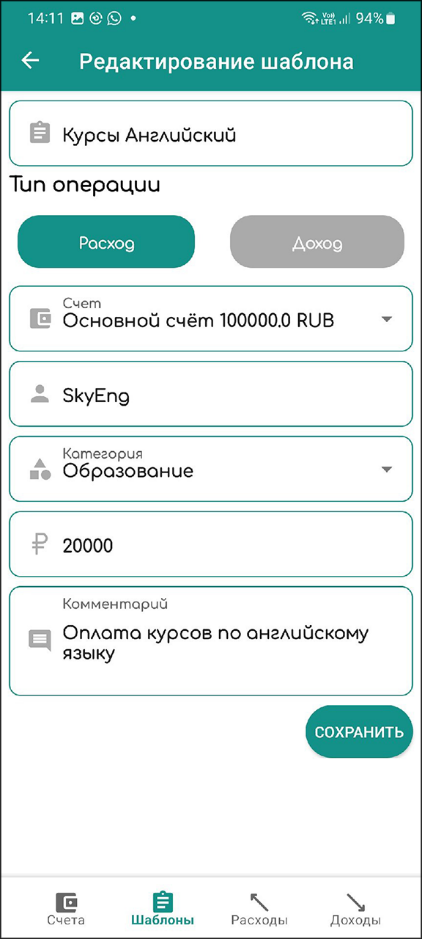
1. Экран шаблонов

Экране «Шаблон» необходим, для добавления нового шаблона. На нем отображены поля с названием шаблона, выбором счета, типом операции, отправителем или получателем, выбором категории, суммы и комментария. Ниже расположена кнопка для сохранения.



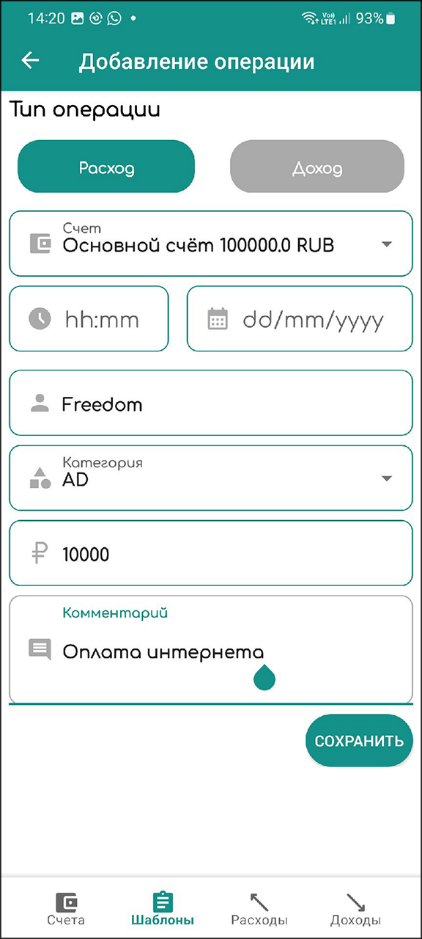
1. Экран создания шаблона

На экране «Редактирование шаблона» отображены поля с названием шаблона, выбором счета, типом операции, отправителем или получателем, выбором категории, суммы и комментария. Ниже расположена кнопка для сохранения. Данный экран отображается после нажатия кнопки с изображением ручки на экране «Шаблоны».



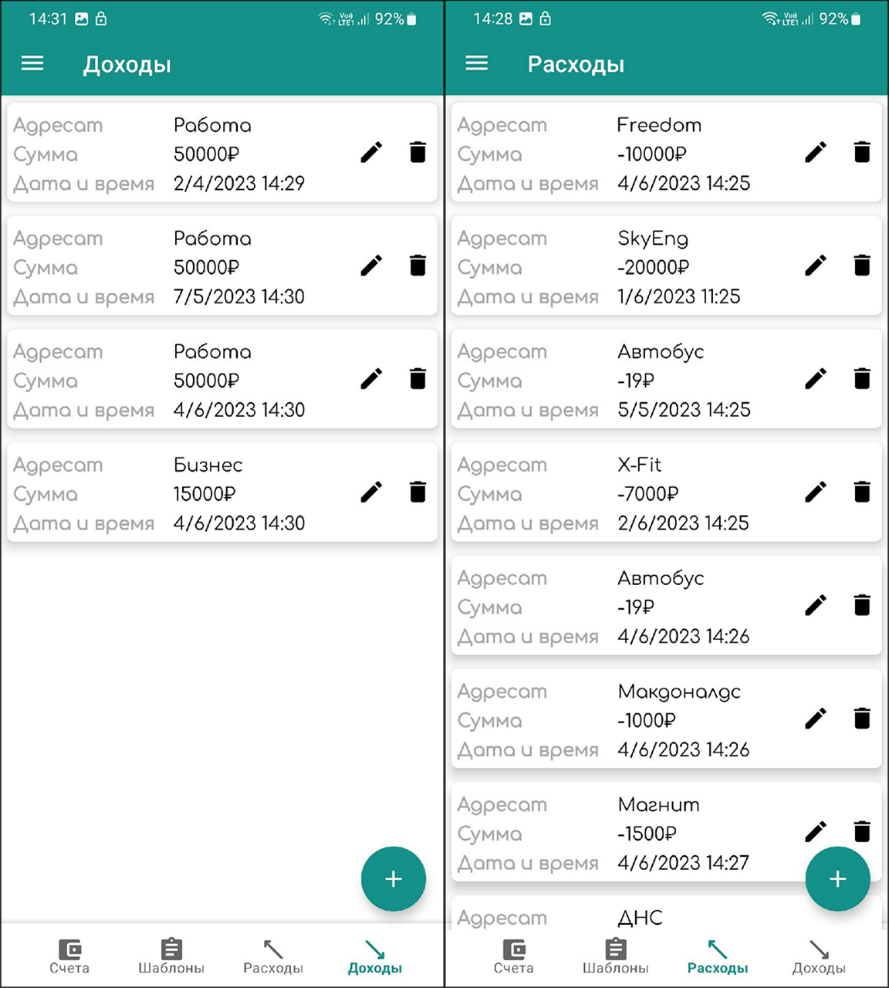
1. Экран редактирования шаблона

При нажатии на шаблон, отображается экран «Добавление операции», в котором следующие поля предзаполнены: тип операции, счет, отправитель или получатель, категория, сумма, комментарий.



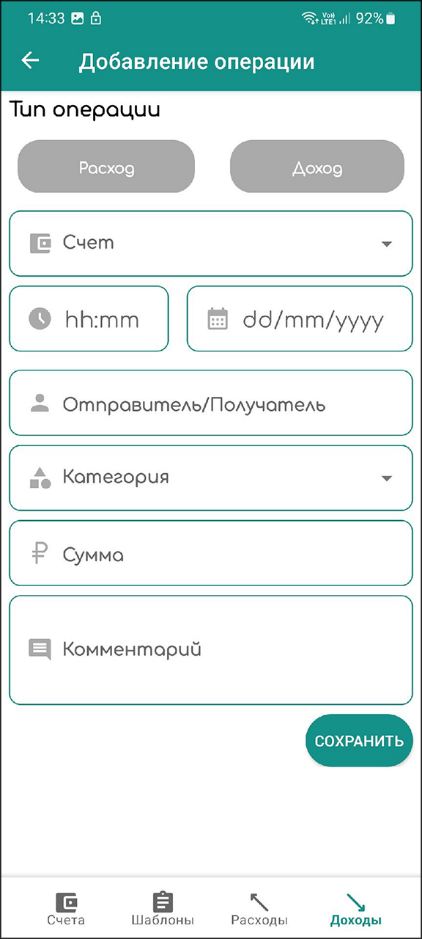
1. Экран добавления операции при нажатии на шаблон
   * 1. Экраны финансовых операций

На экране «Доходы» и «Расходы» авторизованный пользователь имеет возможность просматривать уже добавленные финансовые операции, редактировать, удалять их, а также создать новый при нажатии на кнопку.



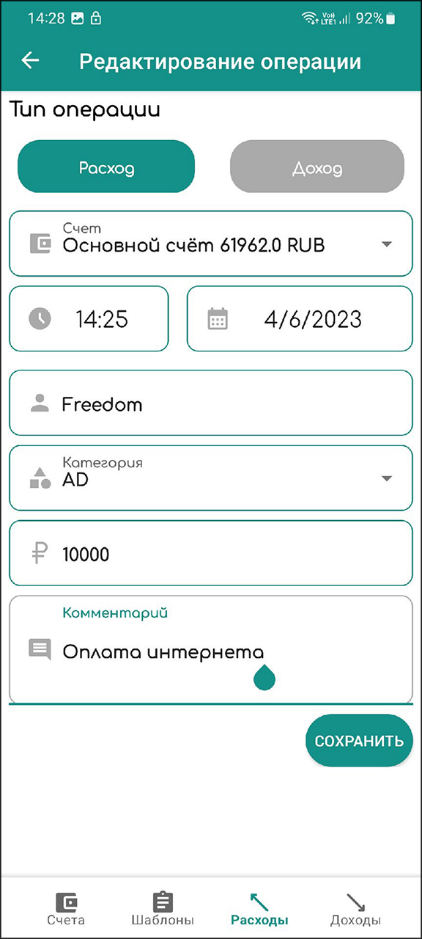
1. Экраны доходов и расходов

На экране «Добавление операции» отображены поля с типом операции, выбором счета, времени и даты, отправителем или получателем, выбором категории, суммы и комментария. Ниже расположена кнопка для сохранения.



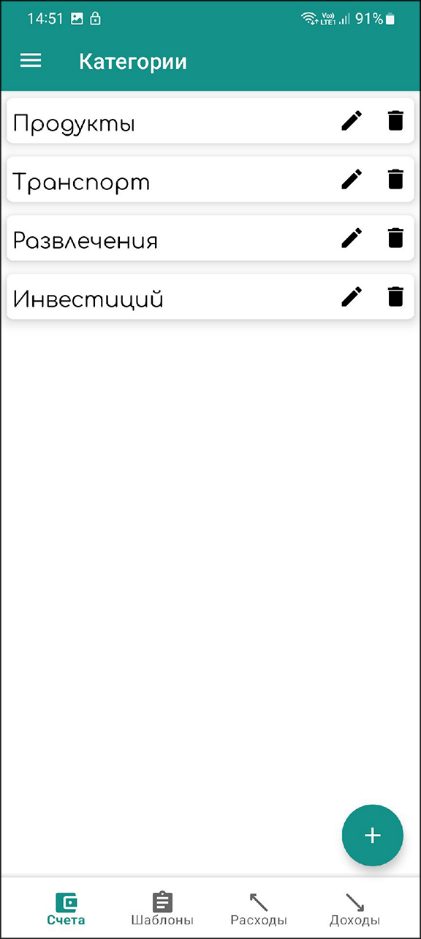
1. Экран добавления операции

На экране «Редактирование операции» отображены поля с типом операции, выбором счета, времени и даты, отправителем или получателем, выбором категории, суммы и комментария. Ниже расположена кнопка для сохранения.



1. Экран редактирование операции
   * 1. Экран категорий

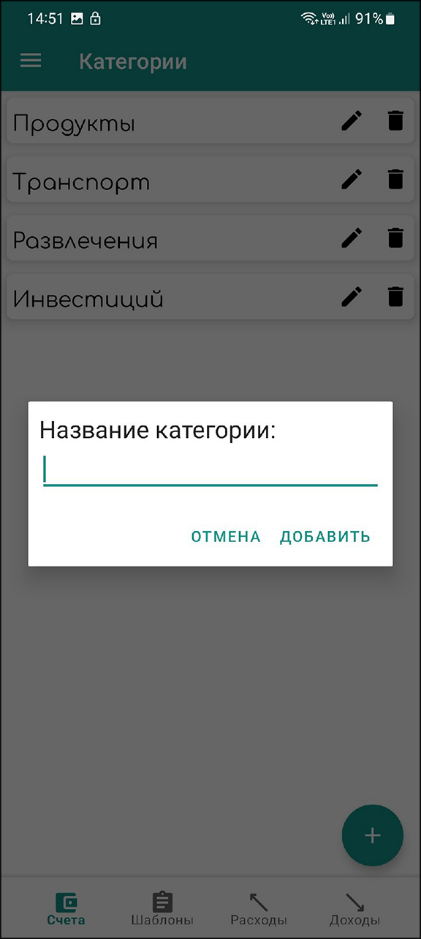
На экране «Категории» авторизованный пользователь имеет возможность просматривать уже добавленные категории, редактировать и удалять их, а также создать новый при нажатии на кнопку. При первой регистрации пользователя автоматический создается три категории: «Продукты», «Транспорт» и «Развлечения».



1. Экран категорий

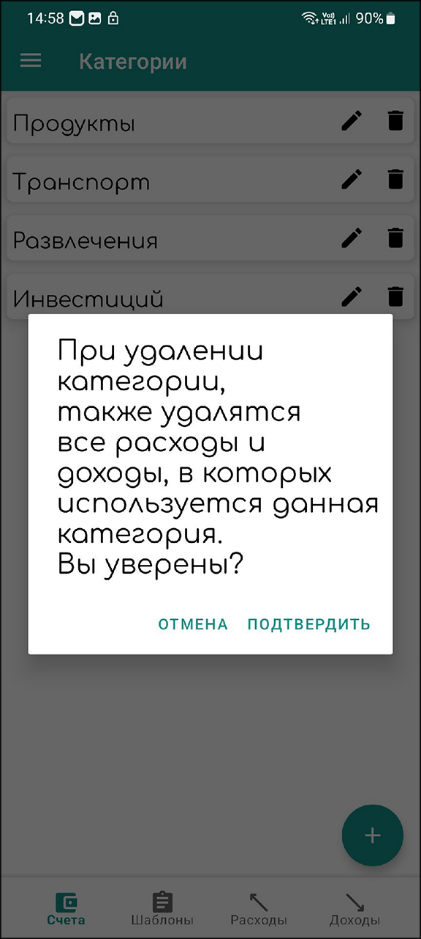
При попытке добавления новой категории открывается диалоговое окно с полем для названия категории и двумя кнопками: «Добавить» и «Отмена».

При нажатии на кнопку с изображением карандаша, откроется диалоговое окно, схожее с диалоговым окном, открывающимся при добавлении новой категории.



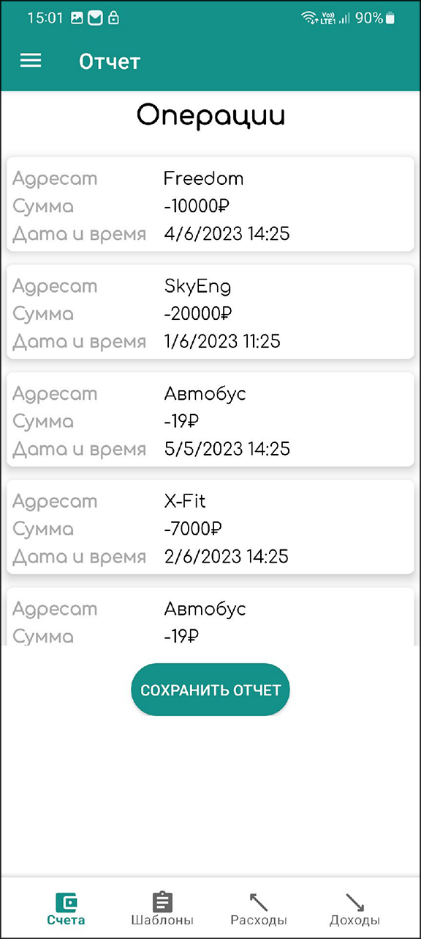
1. Экран создания и редактирования категории

При попытке удаления категории, путем нажатия на кнопку с изображением мусорной корзины, появится диалоговое окно с подтверждением удаления категории.



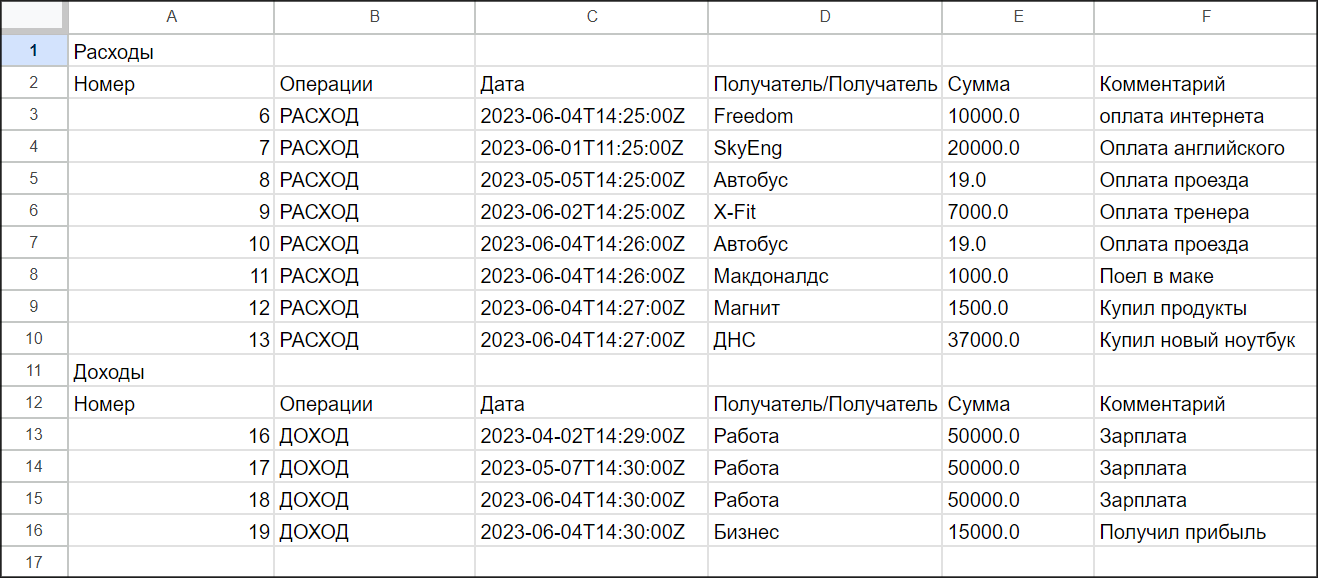
1. Диалоговое окно удаления категории
   * 1. Экран отчета

На данном экране отображены все финансовые операции, добавленные авторизованным пользователем. Под списком операций имеется кнопка для сохранения отчета, при нажатии на нее составляется таблица в формате «CSV».



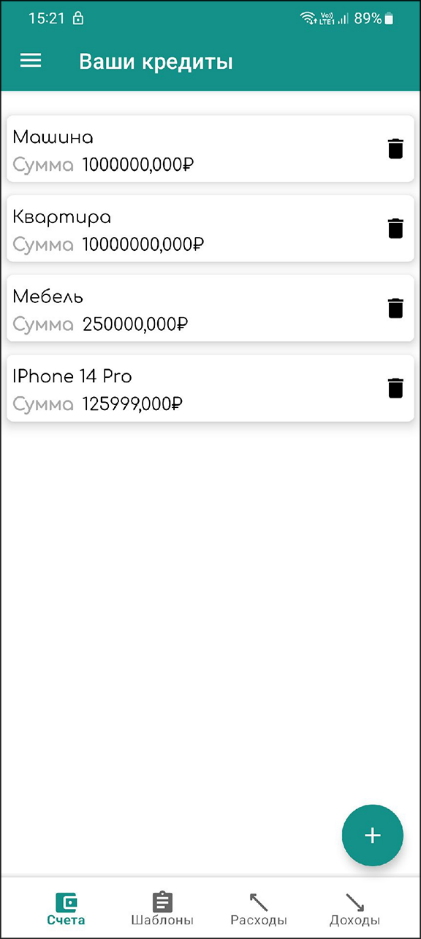
1. Экран отчета

Таблица сохраняется в локальное хранилище устройства в директории «Документы».

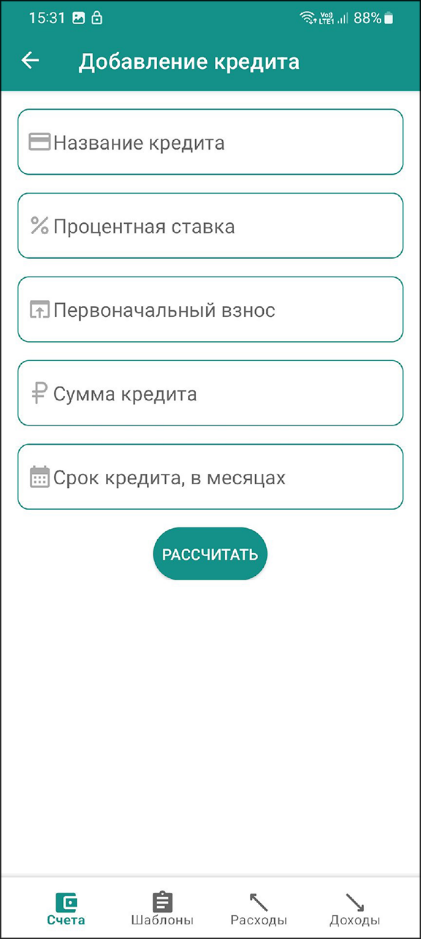


1. Результат сохранения отчета
   * 1. Экраны кредитов

На экране «Ваши кредиты» авторизованный пользователь имеет возможность просмотреть ранее добавленные кредиты, а также добавить новый при нажатии на кнопку, расположенной в правом нижнем углу экрана.

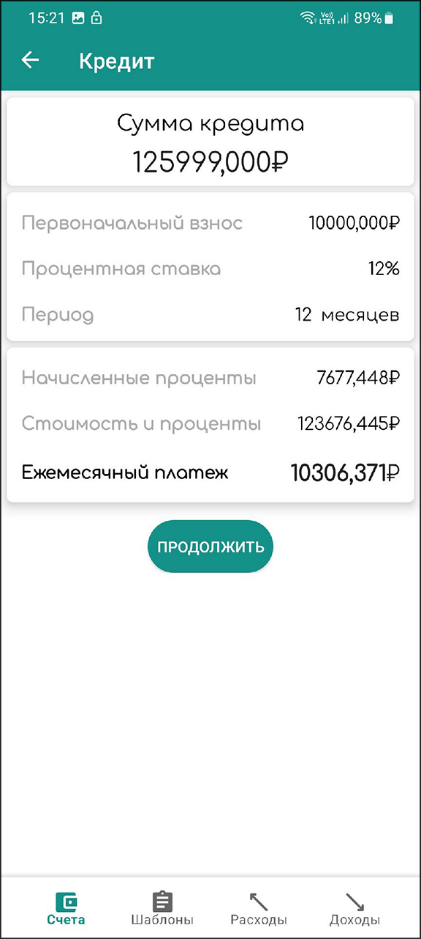


1. Экран кредитов



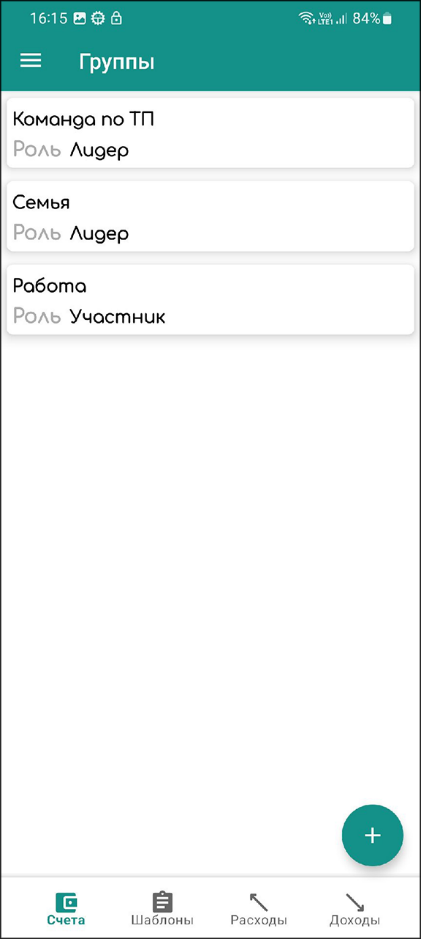
1. Экран добавления кредита

После корректного ввода всех полей и нажатие на кнопку «Рассчитать», а также при нажатии на кредит из экрана «Ваши кредиты», отобразится экран «Кредит», в котором показана информация о добавленном кредите: сумма кредита, первоначальный взнос, процентная ставка, период, начисленные проценты, стоимость и проценты и ежемесячный платеж. Также под информацией о кредите расположена кнопка «Продолжить», при нажатии на которую, отобразится экран «Ваши кредиты».



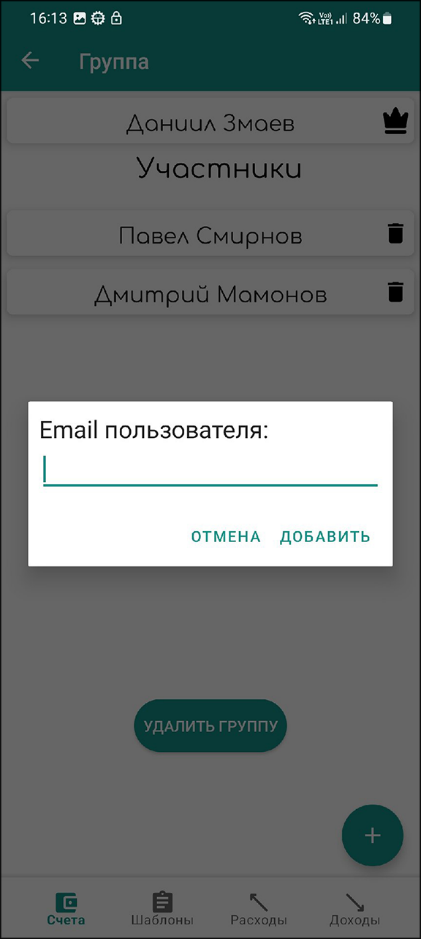
1. Экран просмотра информации о кредите
   * 1. Экраны группы

На данном экране расположены список групп, в которых состоит пользователь, а также кнопка, при нажатии на которую, всплывает диалоговое окно.



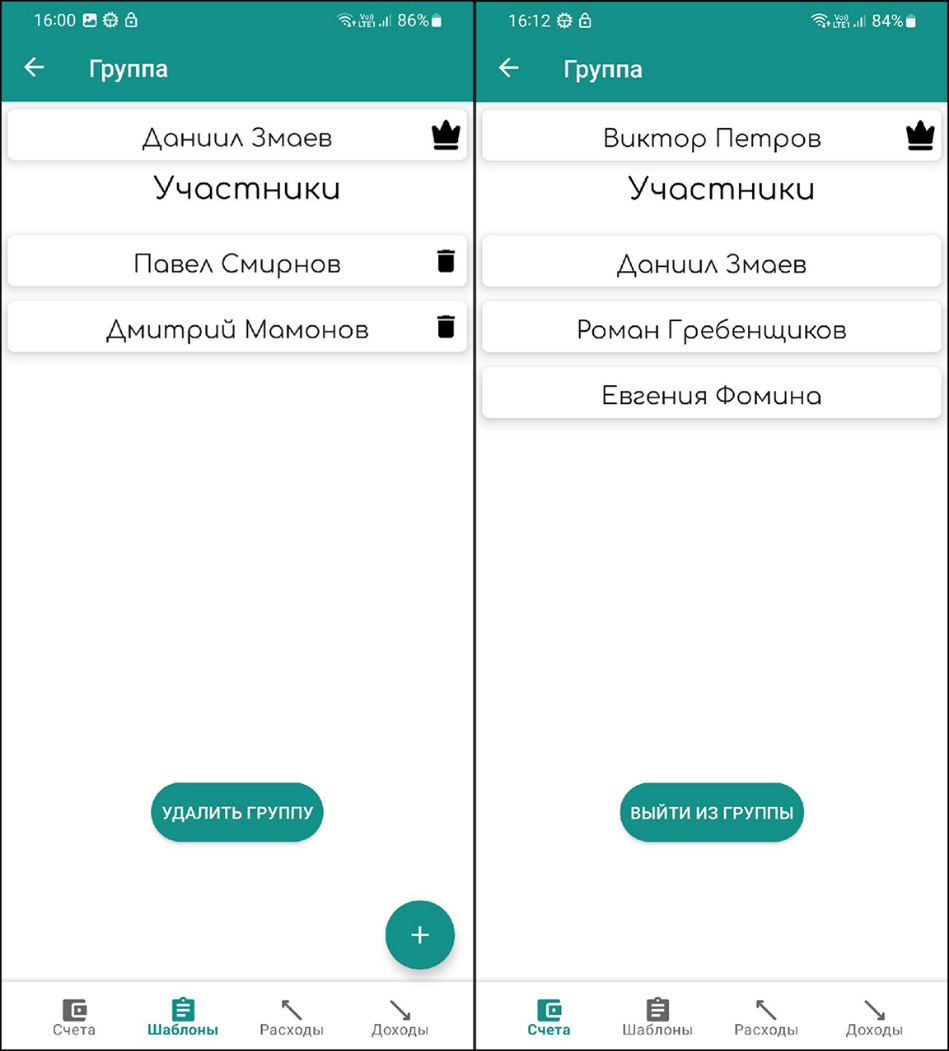
1. Экран просмотра списка групп

Диалоговое окно содержит поле для ввода почты пользователя, а также две кнопки «Отмена» и «Добавить».



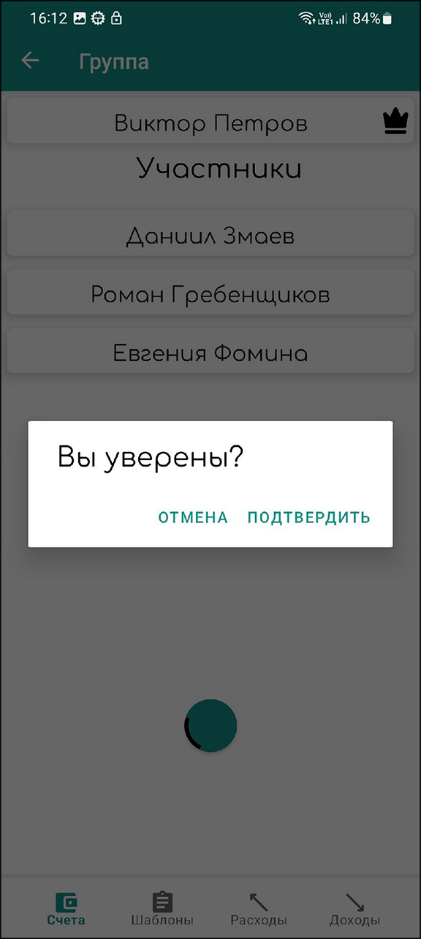
1. Диалоговое окно добавления пользователя

При нажатии на группу, открывается экран «Группа», в котором отображены все участники, состоящие в группе.



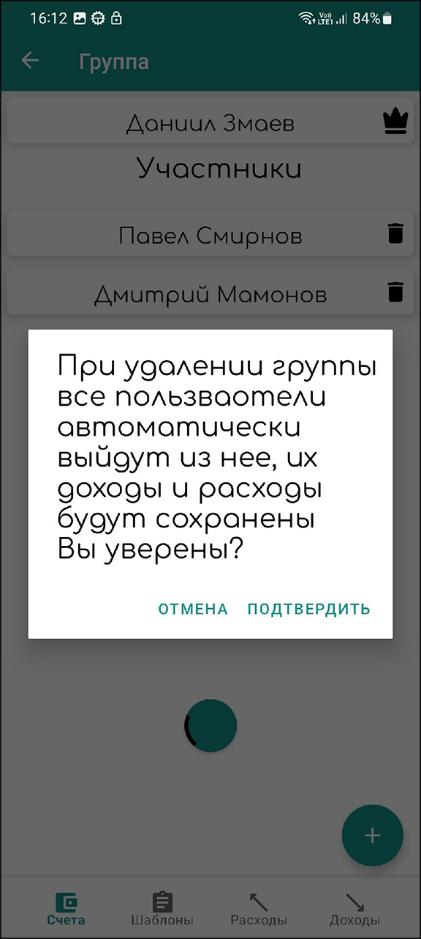
1. Экран группы для создателя и участника

При нажатии на кнопку с изображением мусорной корзины, для удаления пользователя из группы, или при нажатии на кнопку «Выйти из группы» всплывает диалоговое окно.



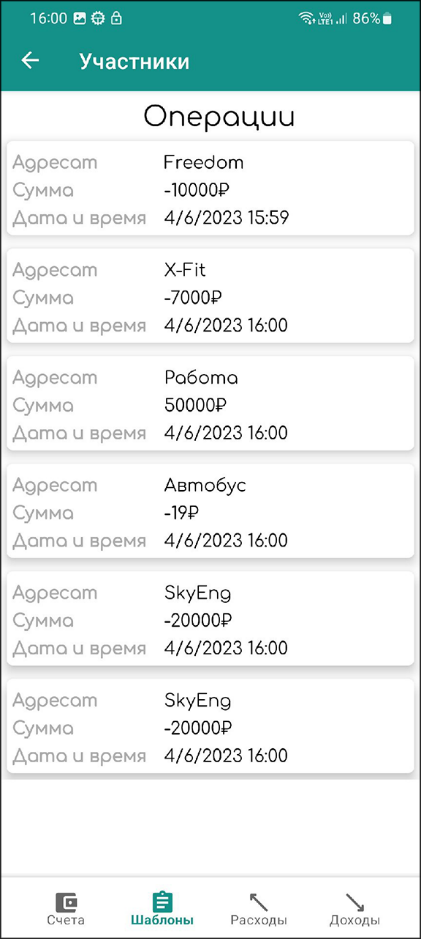
1. Диалоговое окно выхода или исключения пользователя из группы

При нажатии на кнопку «Удалить группы» создателем группы, отображается диалоговое окно с подтверждением действия.



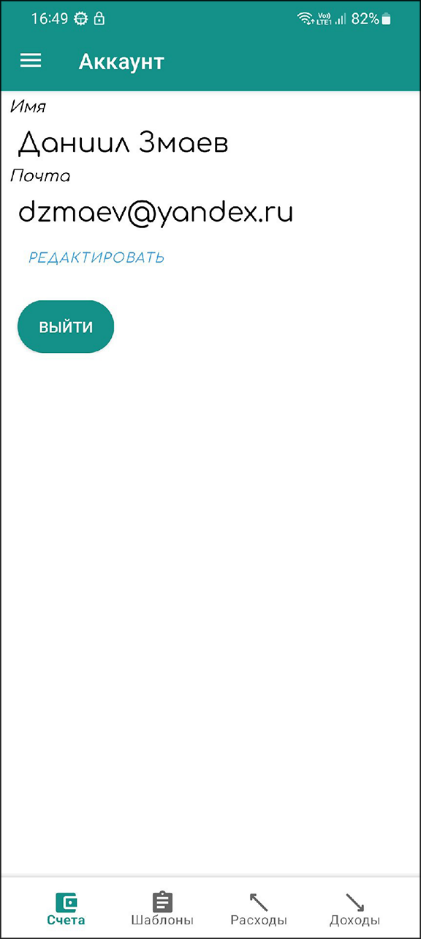
1. Диалоговое окно удаления группы

При нажатии на участника группы отображается экран финансовых операций выбранного участника.



1. Экран операций участника группы
   * 1. Экран аккаунта пользователя

На данном экране расположена информация о пользователе: имя и фамилия и почта. Также имеются две кнопку: «Редактировать» и «Выйти»



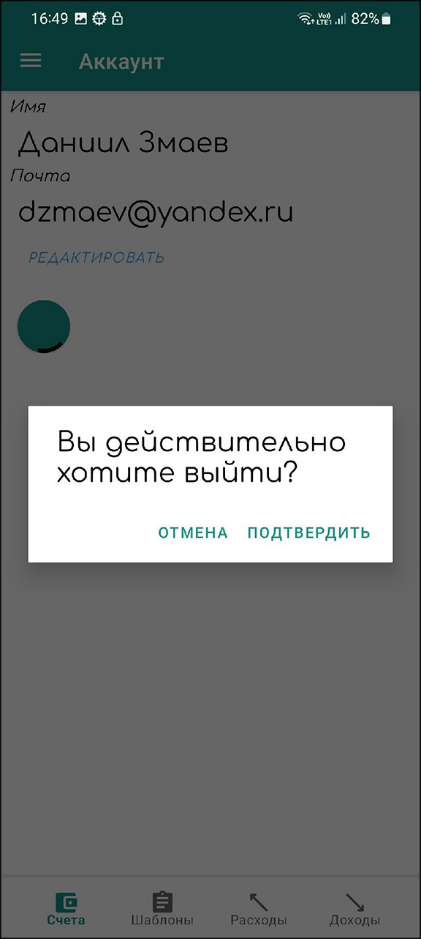
1. Экран аккаунта

При нажатии на кнопку «Редактировать», отображается экран редактирования аккаунта пользователя, в котором расположены следующие поля: имя, фамилия. Под полем фамилия имеется кнопка «Подтвердить», необходимая для сохранения отредактированных данных.



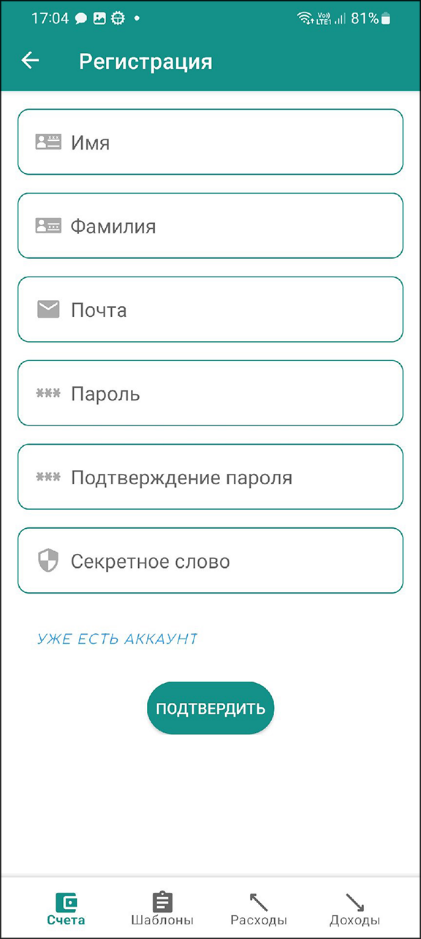
1. Экран редактирования аккаунта

При нажатии на кнопку «Выйти», появляется диалоговое окно, необходимое для подтверждения действия.



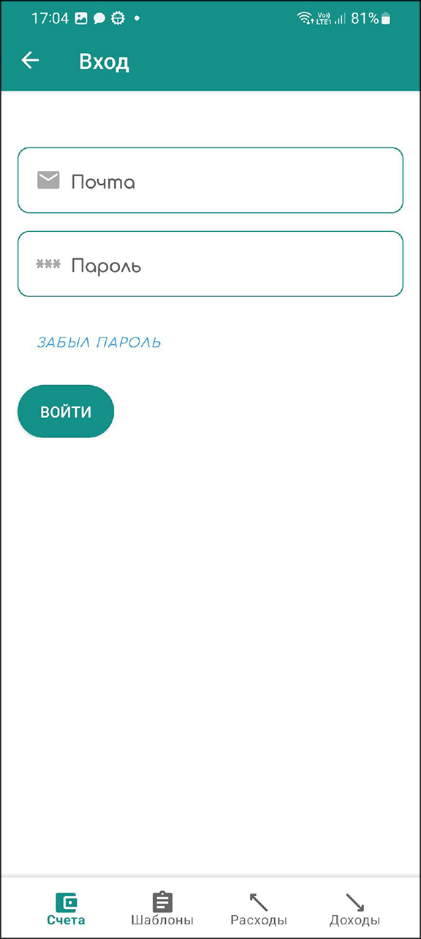
1. Диалоговое окно выхода из аккаунта
   * 1. Экран регистрации

На данной странице отображается форма для регистрации пользователя: имя, фамилия, почта, пароль, подтверждение пароля и секретное слово. Также имеются две кнопки, «Уже есть аккаунт» и «Подтвердить». Данный экран отображается при попытке пользователя добавить счет, финансовую операцию, группу.



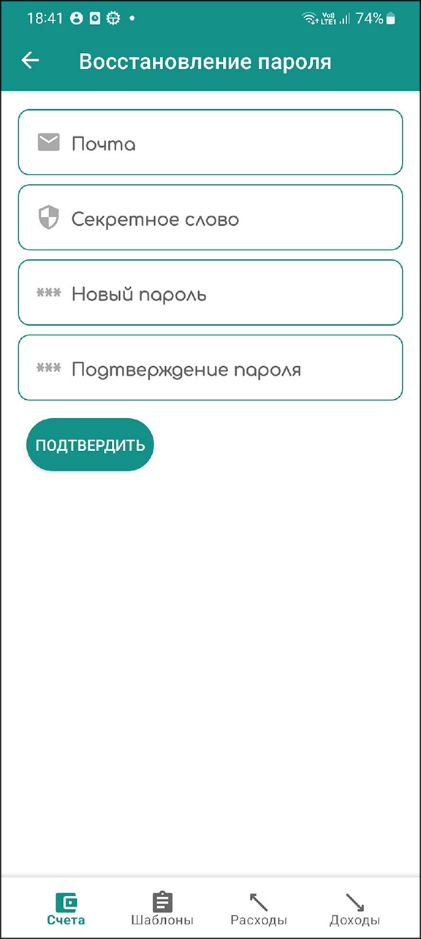
1. Экран регистрации
   * 1. Экран входа

На данном экране расположены поля для выполнения входа в аккаунт: почта и пароль, также имеются две кнопки «Забыл пароль» и «Войти». Данный экран отображается при нажатии на кнопку «Уже есть аккаунт» на экране «Регистрация».



1. Экран входа
   * 1. Экран восстановления пароля

На данном экране расположены поля для выполнения процедуры восстановления пароля: почта и секретное слово, новый пароль и его подтверждение, также имеется и «Подтвердить». Данный экран отображается при нажатии на кнопку «Забыл пароль» на экране «Вход».



1. Экрана восстановления пароля
   1. Серверная часть

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. Тестирование
2. Продуктовые воронки

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