Частное учреждение образования

«Колледж бизнеса и права»

РАЗРАБОТКА МОБИЛЬНОГО ПРИЛОЖЕНИЯ

“Мои финансы”

Пояснительная записка

к курсовому проекту по предмету

«Конструирование программ и языки программирования»

КП Т.191016.401

Руководитель проекта (Е.Н.Коропа)

Обучающийся (Д.А.Короленко)

2024

Изм.

Лист

№ докум.

Подпись

Дата

Лист

У 2

Разраб.

Провер.

*Коропа Е.Н.*

Т.контр.

Н. Контр.

Утверд.

Разработка мобильного приложения «Мои Финансы»

Лит.

Листов

32

КБП

**СОДЕРЖАНИЕ**

[**Введение** 3](#_Toc167283149)

[**1 Описание задачи** 5](#_Toc167283150)

[**1.1 Анализ предметной области** 5](#_Toc167283151)

[**1.2 Постановка задачи** 5](#_Toc167283152)

[**2 Проектирование мобильного приложения** 8](#_Toc167283153)

[**2.1 Проектироваие модели** 8](#_Toc167283154)

[**2.2 Требования к мобильному приложению** 9](#_Toc167283155)

[**2.3 Структура мобильного приложения** 9](#_Toc167283156)

[**2.4 Проектирование макета мобильного приложения** 10](#_Toc167283157)

[**2.5 Защита и сохранность данных** 15](#_Toc167283158)

[**2.6 Организация и ведение информационной базы (модели)** 15](#_Toc167283159)

[**3 Проектирование мобильного приложения** 17](#_Toc167283160)

[**3.1 Программно-технические средства, необходимые для разработки приложения** 17](#_Toc167283161)

[**3.2 Описание разделов приложения** 18](#_Toc167283162)

[**3.3 Описание используемых функций и процедур** 18](#_Toc167283163)

[**3.4 Функциональное тестирование** 26](#_Toc167283164)

КП Т.191016.401 ГЧ

*Короленко Д.А.*

# **Введение**

Мобильное приложение — это специализированный софт, созданный для работы на портативных устройствах, таких как смартфоны и планшеты. Пользователи могут скачать его из приложения-магазина и использовать для доступа к различным функциям и услугам на своих устройствах. Сегодня мобильные приложения играют ключевую роль в взаимодействии с разнообразными современными ресурсами. Учитывая, что практически у каждого есть мобильный телефон, использование мобильных приложений стало неотъемлемой частью нашей жизни. Они хранят обилие полезной информации и автоматизируют множество ежедневных задач.

В наше время мобильные приложения являются важной частью современного человека. В них хранится множество полезной информации и автоматизируются многие повседневные дела.

Целью курсового проектирования является создание мобильного приложения для автоматизации отслеживания планирования движения финансов, выраженных в наличном виде.

Разработка мобильного приложения «Мои финансы» предполагает создание удобного инструмента для отслеживания движения финансов, выраженных в наличном виде.

Приложение обеспечит пользователям возможность регистрации и авторизации, что позволит не только защитить данные, но и персонализировать их. Ввод и учет наличных средств станут основой функционала, давая возможность добавлять, редактировать и удалять транзакции. Категоризация транзакций поможет пользователям лучше понимать структуру своих расходов и доходов, а отображение истории транзакций с визуализацией данных упростит поиск нужной информации

Для достижения поставленной цели были решены следующие задачи:

* изучены средства для организации системы работы финансами;
* проанализированы существующие аналоги. Мобильные средства для работы с финансами;
* разработаны алгоритмы и структуры данных для хранения информации;
* разработан пользовательский интерфейс;
* были произведены тестирование и отладка программного средства.

Пояснительная записка состоит из четырех разделов.

В первом разделе «Описание задачи» проводится анализ предметной области, изучаются задачи, которые планируется решать посредством мобильного приложения, определяется потенциальная аудитория, указывается периодичность использования приложения, определяются функции для автоматизации на основе проанализированных бизнес-процессов и бизнес-задач, изучаются существующие аналоги.

Во втором разделе «Проектирование мобильного приложения» проектируются модели, определяются требования к мобильному приложению, определяется структура мобильного приложения, проектируется макет, определяются способы защиты и сохранности данных, а также организация и ведение информационной базы.

В третьем разделе «Реализация мобильного приложения» определяются программно-технические средства, необходимые для разработки приложения, описываются разделы приложения, используемые функции и процедуры, описывается функциональное тестирование.

В четвертом разделе «Применение» описываются общие сведения и назначение мобильного приложения.

В заключении описывается поставленная задача, как она была выполнена, описываются все разработанные функции, преимущества и недостатки мобильного приложения.

# **Описание задачи**

## **Анализ предметной области**

Разрабатываемая система является мобильным приложением для упрощенного ведения личной бухгалтерии. Это приложение предоставляет широкий спектр функций и возможностей, которые помогут пользователям

эффективно планировать свой бюджет, отслеживать доходы и расходы, анализировать инвестиции и контролировать свои финансовые цели. Используя данное приложение человек, повышает свою финансовую грамотность, что важно уметь в современном мире. Первичная целевая аудитория состоит из людей 16-58 лет (женщины) и 16 – 63 лет (мужчины) – это люди, которые имеют заработок и желают правильно управлять финансами. Приложение является мобильным, в связи с тем, что каждый человек носит телефон в кармане и в любой момент может внести информацию про новый платеж.

Цель автоматизации процесса ведения личной бухгалтерии состоит в цифровизации записей о расходах и доходах

Разрабатываемое приложение актуально тем, что подходит для аудитории любого пола и возраста, так как интерфейс и функционал будут достаточно просты для восприятия любыми пользователями. На сегодняшний день очень важно быть финансового грамотным человеком и приложение «Мои финансы» будет помогать развиваться в этом направлении.

Периодичность использования приложением – каждый день. Такая периодичность обусловлена тем, что большинство людей совершают покупки каждый день и для более информативного использования приложения следует заполнять

## **Постановка задачи**

На основе изученных бизнес-моделей и бизнес-процессов разрабатываемое приложение предполагает автоматизацию следующих процессов: создание операций, кошельков/карт, категорий с возможностью их редактирования, удаление операций, кошельков/карт, категорий, настройка создаваемого объекта, конвертер с актуальным курсом валют, анализ денежного потока с использованием графиков.

Операция – это запись об 1 финансовой операции.

Кошелек/карта – это условный счет, для раздельного ведения финансов (к примеру, у пользователя несколько банковских карт или отдельный счет для откладывания)

Категория – тип дохода/расхода с указанием для детального разделения финансовых операций.

В настоящее время в магазинах приложений на смартфонах существует большое количество различных приложений, в том числе и приложений, автоматизирующих процесс слежения и управления денежными потоками, однако каждое отличается от других и имеет свой уникальный интерфейс и различный дополнительных функционал помимо общего.

Разрабатываемое приложение будет иметь как общие, так и уникальные особенности, отличающие его от других приложений. Например, мобильное приложение «Финансы», рассматриваемое в качестве аналога. На рисунке 1 представлено начальное окно, которое открывается при запуске приложения.



Рисунок 1 – начальное окно

приложения "Финансы"

На главном окне приложения «Финансы» отображаются ваши операции и диаграмма, демонстрирующая категоризацию финансовых операций

Достоинствами данного аналога является создание любого графика с возможностью выбора как стандартного периода, так и выбранного, а также наличие большего количества валют в конвертере и для установки стандартной валюты, возможность добавления фотографий в дополнение тексту при описании операции

Из недостатков выделяется отсутствие у пользователя возможности аккаунта и, следовательно, невозможно перенести на другое устройство свою финансовую



Рисунок 2 - Страница категорий

в приложении "Финансы"

На рисунке 2 представлено окно категорий в приложении «Финансы», на котором представлены категории расходов пользователя, а также при необходимости и доходов. Каждая категория имеет название, иконку и цвет отображения на диаграммах

# **Проектирование мобильного приложения**

## **Проектироваие модели**

Неотъемлемой частью проектирования модели мобильного приложения является построение диаграмм. Для разрабатываемого мобильного приложения были построены следующие диаграммы: диаграмма прецендентов (Диаграмма вариантов использования), диаграмма деятельности, диаграмма последовательности и диаграмма классов.

В диаграмме прецендентов представляется подробное описание всего функционала приложения, вариантов его использования и связей каждого прецендента приложения между собой. В диаграмме представлено два вида связей: include(обязательная) и extend(необязательная), они служат для лучшего понимания работы приложения в целом и взаимодействия отдельных прецендентов между собой.

Последовательность шагов для реализации сценария «Добавление новой финансовой операции»

1. Пользователь авторизовывается;
2. Нажимает на кнопку «Добавить операцию»;
3. Заполняет строку для ввода суммы;
4. Выбирает нужную категорию или оставляет по умолчанию;
5. Выбирает нужный тип счета или оставляет по умолчанию;
6. По желанию добавляет описание операции, для лучшего понимания;

Последовательность шагов для реализации сценария «Осуществление ведения категории»

1. Пользователь авторизовывается;
2. Переходит на страницу категорий;
3. Нажимает кнопку «Добавить новую категорию»;
4. Вводит название новой категории;
5. Вводит

Диаграмма классов соответственно отображается разработанные классы «Category», «CategoryRepository» «ColorRepository» «Colorss» «Context» «Wallet» «WalletRepository» «ExtendedWallet» «Links» «Operation» «OperationRepository» «OperationResult» «OperationDays» «User» «UserRepository» «Validator», которые в свою очередь содержат в себе атрибуты и методы для управления данными.

Структура мобильного приложения представляет собой отображение всего функционала и возможностей данного приложения и подробно описывает его.

## **Требования к мобильному приложению**

При разработке данного приложения учитываются список требований, таких как:

Требования к операционной системе: Приложение должно быть совместимо с iOS и Android для обеспечения широкой доступности. Это позволит охватить большинство современных устройств и предоставить пользователям стабильную работу приложения.

Требования к стилистическому оформлению: Стиль должен быть минималистичным и современным, чтобы упростить навигацию и сделать интерфейс интуитивно понятным. Использование чистых линий и отсутствие излишних деталей помогут сосредоточить внимание на функционале.

Требования к графическому дизайну: Графика должна быть высокого качества, с четкими иконками и изображениями. Важно использовать унифицированные элементы дизайна для создания единой визуальной идентичности приложения.

Требования к шрифтовому оформлению: Шрифты должны быть читаемыми, с хорошим контрастом на фоне приложения. Рекомендуется использовать системные шрифты, такие как San Francisco для iOS и Roboto для Android, для обеспечения согласованности с операционной системой.

Требования к устройству: Приложение должно быть оптимизировано для работы на устройствах с различными размерами экранов, от маленьких смартфонов до больших планшетов, адаптируясь под разрешение и плотность пикселей.

Требования к контенту: Контент должен быть актуальным, точным и полезным. Тексты должны быть краткими и ясными, без сложной терминологии, чтобы пользователи могли легко понять представленную информацию и с легкостью проанализировать свое финансовое состояние.

При создании операции, кошелька и категории пользователю необходимо вводить данные текстом и для недопущения ошибок предусмотрены ограничения на поля для ввода такие как:

- максимальная длина строк

- недопущение пустых строк

- содержание подстроки “@gmail.com” в названии почты

Требования к компоновке экранов: Экраны должны быть организованы логически и эффективно, с четким разделением функций. Важно обеспечить, чтобы пользователь мог легко переходить от одной задачи к другой

## **Структура мобильного приложения**

При первом запуске приложения пользователь будет реализовывать сценарий «Регистрация», пошаговый переход между экранами, показан на wireflow диаграмме, которая представлена в приложении …

После регистрации, пользователь имеет доступ к реализации сценариев «Ведение операций», «Ведение кошельков», «Ведение категорий», «Настройка приложения» и т.д. В качестве примера на wireflow диаграмме в приложении … представлена пошаговая реализация сценария «Настройка приложения»

Физическая структура приложения включает в себя несколько каталогов. Главная папка приложения содержит 3 каталога (кроссплатформенный код и ресурсы для всех платформ, папка для ресурсов Android системы и папка для ресурсов IOS системы). Каталог с кроссплатформенными ресурсами включает в себя папку core, в которой содержаться файлы с описанием пользовательских классов, папка icons содержит в себе иллюстрационный материал. Каталог views представляет все файлы, описывающие пользовательские окна.

## **Проектирование макета мобильного приложения**

Для визуализации приложения при его проектировании создаются макеты, которые примерно отображают необходимые окна, расположения элементов на них и примерный функционал.

Макет главного экрана, представленный на рисунке 3, изображает начальный экран мобильного приложения с возможностью перехода на страницы входа или регистрации. В верхней части экрана расположен стилизованный логотип, привлекающий внимание к бренду. Простота дизайна способствует легкому восприятию и интуитивному пониманию пользователем. Экран является ключевым для взаимодействия пользователя с приложением, обеспечивая быстрый доступ к учетной записи.

Макет экрана транзакций, представлен на рисунке 4. Этот макет демонстрирует экран транзакций пользователя с отображением баланса и категорий расходов.

Читаемость: Белый фон с черным текстом и зелеными элементами делает информацию легко читаемой.

Управление финансами: Пользователи могут эффективно контролировать свои финансы, видя общий баланс и детализацию доходов и расходов.

Календарь расходов: ниже основной информации будет расположен календарь расходов, который помогает отслеживать финансовую активность.

Минимализм: Дизайн выполнен в минималистичном стиле, что способствует удобству навигации и использования приложения.

|  |  |
| --- | --- |
| Рисунок 3 – Макет загрузочного экрана | Рисунок 4 - Макет экрана со списком операций |

Макет страницы регистрации представлен на рисунке 5 - этот макет представляет собой интерфейс для создания учетной записи. В верхней части находится заголовок “Create an account”, что указывает на начало процесса регистрации. Иконка с изображением здания с колоннами может символизировать официальность или безопасност приложения. Пользователю предлагается заполнить четыре поля: “Email”, “Nickname”, “Password” и “Repeat password”, что является стандартной процедурой для регистрации. Внизу расположены две кнопки: “Cancel”, чтобы отменить регистрацию, и “Create”, чтобы завершить создание учетной записи.

На рисунке 6 отображен макет страницы авторизации, данный макет страницы авторизации включает в себя элементы, необходимые для входа пользователя в систему. В верхней части расположен заголовок “Authorisation”, подчеркивающий функцию страницы. Приветственное сообщение “Welcome back” создает дружелюбную атмосферу. Для ввода данных предусмотрены поля “Email” и “Password”, а также кнопка “Log In” для подтверждения входа. Внизу страницы предлагается возможность авторизации через социальные сети, что удобно для пользователей, предпочитающих этот метод входа.

|  |  |
| --- | --- |
| Рисунок 5 - Макет страницы регистрации | Рисунок 6 - Макет страницы авторизации |

На рисунке 7 изображен макет страницы анализа. Он включает в себя отчет о доходах и расходах, что помогает пользователям отслеживать свои финансы. Макет имеет чистый и минималистский дизайн, что делает его простым и понятным для пользователя.

В интерфейсе есть различные элементы управления, включая навигационные кнопки и интерактивные графики. Эти элементы управления обеспечивают персонализированный пользовательский опыт и делают использование приложения удобным и эффективным.

В целом, этот макет является примером того, как можно эффективно использовать пространство на экране мобильного устройства для предоставления полезной и актуальной информации пользователю, сохраняя при этом простоту и удобство использования.

На рисунке 8 изображен макет страницы категорий. Этот макет представляет собой интерфейс пользователя для мобильного приложения или веб-сайта, который помогает отслеживать и управлять финансами. В верхней части изображены две основные категории: “Increase” и “Expenses”, каждая из которых имеет свои подкатегории, обозначенные как “Category name”.

Подкатегории можно редактировать, что показано значком карандаша рядом с каждым названием. Также есть возможность добавить новую подкатегорию, что обозначено плюсом внизу списка. В нижней части изображены иконки для навигации по приложению.

|  |  |
| --- | --- |
| Рисунок 7 - Макет страницы анализа | Рисунок 8 - Макет страницы категорий |

Макет страницы кошельков и карт представлен на рисунке 9. На верхней части экрана отображается общий баланс. Ниже представлены детали индивидуальных карт, включая названия карт, типы и информацию о приросте и расходах. В нижней части интерфейса расположены пять иконок, представляющих различные вкладки и разделы в приложении. Данный макет прост, и визуально предоставляет всю необходимую и важную информацию для пользователя.

На рисунке 10 изображен макет конвертера. Этот макет представляет собой дизайн страницы конвертера валют. Основными валютами конвертации являются белорусский рубль, евро, доллар США и российский рубль. Это интересно для пользователей стран СНГ, которым нужно быстро перевести значения между этими валютами. Этот дизайн может быть полезен для людей, которые часто путешествуют или занимаются международным бизнесом, и им требуется удобный инструмент для быстрого и точного перевода валют. Он также может быть полезен для тех, кто следит за курсами валют и хочет иметь возможность быстро получить актуальную информацию. Этот макет является примером того, как можно сделать сложный процесс простым и понятным для пользователя.

|  |  |
| --- | --- |
| Рисунок 9 - Макет страницы кошельков/карт | Рисунок 10 - Макет страницы конвертера |

Макет страницы для добавления новой операции представлен на рисунке 11. Этот макет представляет собой интерфейс для добавления новой операции. В верхней части изображения находится заголовок “Newoperation”.). В нижней части изображения, как и на других страницах расположено меню переходов по страницам

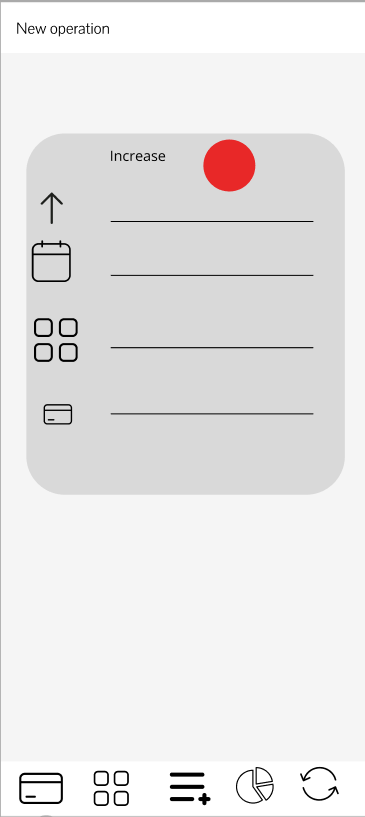


Рисунок 11 - Макет страницы

добавления новой операции

## **Защита и сохранность данных**

Для предотвращения несанкционированного доступа к данным пользователей третьих лиц пароли хешируются. При авторизации сравниваются получившиеся хеши. Для авторизации пользователю необходимо ввести почту и пароль, заданный пользователем при регистрации

Данные хранятся на удаленном сервере, который находится под управлением и наблюдением разработчиков ПО «Финансы»

Доступ к базе данных осуществляется по паролю

## **Организация и ведение информационной базы (модели)**

Организация и ведение информационной базы в приложении осуществляется при помощи клиент-серверной архитектуры. Пользователь (клиент) общается с базой данных (сервер) путем отправки запросов. Данные хранятся с использованием СУБД SQLite в таблицах:

- Categories;

- Colors;

- Operations;

- Users;

- Wallets.

На рисунках 12 – 16 представлены схемы таблиц базы данных,

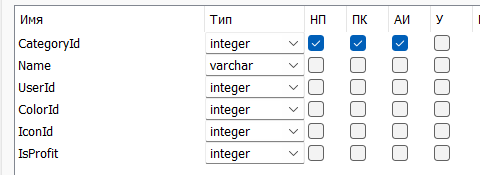


Рисунок 12 - Cхема таблицы "Categories"

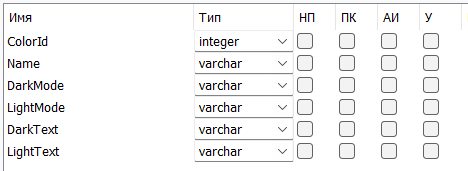


Рисунок 13 - Схема таблицы "Colors"

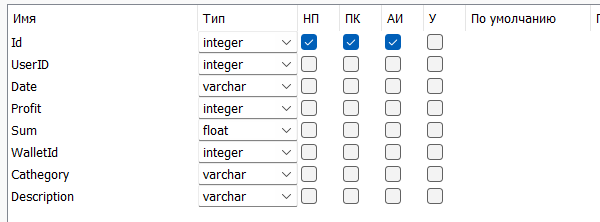


Рисунок 14 - Схема таблицы "Operations"

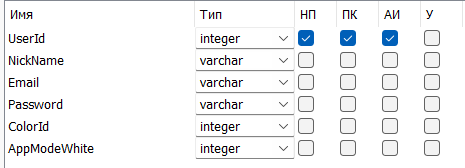


Рисунок 15 - Схема таблицы "Users"

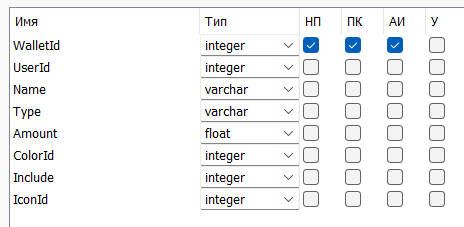


Рисунок 16 - Схема таблицы "Wallets"

Информация перед сохранением в базу данных валидируются на стороне клиента для предотвращения сохранения неверных данных. На имя почты накладывается ограничение наличия подстроки «@gmail.com» и длина 50 символов, на названия категорий, кошельков ограничение длины символов, сумма операции не может быть более 10000 денежных единиц.

# **Проектирование мобильного приложения**

## **Программно-технические средства, необходимые для разработки приложения**

Для разработки мобильного приложения финансы был выбран фрэймворк Xamarin — это фреймворк для кроссплатформенной разработки приложений, в том числе мобильных, с использованием языка программирования C#. Xamarin – предоставляет простой способ создания как интерфейса так и логической структуры приложения

Ключевые особенности Xamarin:

1. Кроссплатформенность: Xamarin позволяет разработчикам использовать один и тот же код на C# для создания приложений для различных платформ;
2. Интеграция с .NET: Xamarin тесно интегрирован с .NET, что позволяет использовать все преимущества этой платформы;
3. Совместимость с Visual Studio: Xamarin полностью совместим с Visual Studio, что обеспечивает удобную среду разработки, в том числе написание и отладку кода.

Для разработки сервера был использован

Ngrok — это платформа, которая с помощью установленной утилиты, позволяет организовать удалённый доступ на веб-сервер или какой-то другой сервис, запущенный на вашем компьютере. Доступ организуется через созданный при запуске ngrok безопасный туннель. Ngrok предоставляет такие функции как:

1. Удаленный доступ к локальному серверу: Ngrok позволяет сделать ваш локальный сервер доступным из интернета. Это особенно полезно для тестирования и демонстраций приложений;
2. Простота настройки: Ngrok устанавливается и запускается очень быстро;
3. Поддержка HTTPS: Ngrok автоматически создает безопасное подключение к вашему локальному серверу с помощью протокола HTTPS;
4. Поддержка различных протоколов: Ngrok поддерживает протоколы HTTP и HTTPS, а также позволяет настраивать авторизацию, фильтрацию запросов и регистров фиксации;
5. Интуитивно понятный интерфейс: Ngrok имеет интуитивно понятный интерфейс, что облегчает его использование.

ASP.Net - это веб-фреймворк, разработанный Microsoft, для создания современных веб-приложений и веб-сервисов. Он является частью платформы .NET и предлагает множество инструментов и библиотек для разработки веб-приложений

Актуальность информации о курсах валют поддерживается благодаря запросам сервера на API «Беларусбанка» каждый час.

## **Описание разделов приложения**

Разработанное приложение имеет в себе несколько разделов, каждый из которых находится на отдельном экране и несет свою смысловую и функциональную нагрузку.

- экран для выбора;

- экран регистрации;

- экран авторизации;

- главный экран;

- экран кошельков;

- экран нового кошелька;

- экран категорий;

- экран новой категории;

- экран графиков;

- экран новой операции;

- экран выбора периода;

- экран конвертера.

Главный экран является тот который открывается при запуске приложения. Макет главного экрана представлен на рисунке 3, на нем изображен темный фон с 2 кругами-кнопками, верхний с надписью «Sign in», что переводітся как авторизоваться, нижний с надписью «Sign up» - зарегистрироваться. При нажатии на каждый из кругов открывается соответствующая страница приложения. В нижней части макета и экрана расположены 2 треугольника разного цвета под разными углами в качестве декоративных элементов.

Навигация осуществляется по средством нажатия на соответствующую кнопку в нижнем навигационном меню, а также нажатием на кнопку «+» в правом нижнем углу для перехода на страницы добавления нового объекта соответствующего странице класса.

## **Описание используемых функций и процедур**

Просмотр движения финансов за выбранный период;

Выбор периода для просмотра (анализа) данных;

Функция «Добавление новой записи о финансовой операции (в выбранном типе операции – доход, расход) с указанием даты, категории, суммы и описания» осуществлена в коде клиента, что позволяет пользователю создавать новые операции. В нижеописанном коде представлен класс «Operation».

using SQLite;

namespace Server.Operations

{

[Table("Operations")]

public class Operation

{

[PrimaryKey, AutoIncrement]

public int Id { get; set; }

public int UserID { get; set; }

public string Date { get; set; }

public bool Profit { get; set; }

public decimal Sum { get; set; }

public int WalletId { get; set; }

public string Cathegory { get; set; }

public string Description { get; set; }

public Operation() { }

public Operation(int operationId ,int userID, string date, bool profit, decimal sum, int wallet, string cathegory, string description)

{

Id = operationId;

UserID = userID;

Date = date;

Profit = profit;

Sum = sum;

WalletId = wallet;

Cathegory = cathegory;

Description = description;

}

public override string ToString() => $"{Id} {UserID} {Date} {Profit} {Sum} {WalletId} {Cathegory} {Description}";

}

}

Ниже представлен код, который реализует функции добавления и проверки новой операции

namespace FinanceApplication.views

{

[XamlCompilation(XamlCompilationOptions.Compile)]

public partial class NewOperationPage : ContentPage

{

Context context = new Context();

bool income = true;

decimal sum = 0;

public NewOperationPage(Context context)

{

InitializeComponent();

NavigationPage.SetHasNavigationBar(this, false);

this.context = context;

IncomePage.IsVisible = false;

ConsumptionPage.IsVisible = false;

TransferPage.IsVisible = false;

sumImage.Source = ImageSource.FromResource(Icons.Iconspath[11]);

walletImage.Source = ImageSource.FromResource(Icons.Iconspath[2]);

cathegoryImage.Source = ImageSource.FromResource(Icons.Iconspath[3]);

descriptionImage.Source = ImageSource.FromResource(Icons.Iconspath[5]);

dateImage.Source = ImageSource.FromResource(Icons.Iconspath[11]);

sumImageС.Source = ImageSource.FromResource(Icons.Iconspath[11]);

walletImageС.Source = ImageSource.FromResource(Icons.Iconspath[2]);

cathegoryImageС.Source = ImageSource.FromResource(Icons.Iconspath[3]);

descriptionImageС.Source = ImageSource.FromResource(Icons.Iconspath[5]);

dateImageС.Source = ImageSource.FromResource(Icons.Iconspath[11]);

xmarkConsume0.Source = xmarkConsume1.Source = xmarkConsume2.Source = xmarkConsume3.Source = ImageSource.FromResource(Icons.Iconspath[16]);

xmarkConsume0.IsVisible = xmarkConsume1.IsVisible = xmarkConsume2.IsVisible = xmarkConsume3.IsVisible = false;

xmarkIncome0.Source = xmarkIncome1.Source = xmarkIncome2.Source = xmarkIncome3.Source = ImageSource.FromResource(Icons.Iconspath[16]);

xmarkIncome0.IsVisible = xmarkIncome1.IsVisible = xmarkIncome2.IsVisible = xmarkIncome3.IsVisible = false;

List<string> walletsNames = new List<string>();

foreach (Wallet wallet in context.Wallets)

walletsNames.Add(wallet.Name);

WalletPicker.ItemsSource = walletsNames;

WalletPickerС.ItemsSource = walletsNames;

WalletPicker.SelectedItem = walletsNames[0];

WalletPickerС.SelectedItem = walletsNames[0];

List<string> CategoriesNames = new List<string>();

foreach (Category category in context.Categories)

CategoriesNames.Add(category.Name);

CathegoryPicker.ItemsSource = CategoriesNames;

CathegoryPickerС.ItemsSource = CategoriesNames;

CathegoryPicker.SelectedItem = CategoriesNames[0];

CathegoryPickerС.SelectedItem = CategoriesNames[0];

}

private void buttonTochangePage(object sender, EventArgs e)

{

SelectionPage.IsVisible = false;

TransferPage.IsVisible = true;

}

private void buttonToConsumePage(object sender, EventArgs e)

{

SelectionPage.IsVisible = false;

ConsumptionPage.IsVisible = true;

income = true;

}

private void buttonToIncomePage(object sender, EventArgs e)

{

SelectionPage.IsVisible = false;

IncomePage.IsVisible = true;

EntrySum.Focus();

income = false;

}

private async void ToCardPage(object sender, EventArgs e) =>

await Navigation.PushAsync(new CardPage(context));

private async void ToListPage(object sender, EventArgs e) =>

await Navigation.PushAsync(new ListPage(DateTime.Now, context));

private async void Cancel\_Clicked(object sender, EventArgs e) => await Navigation.PopAsync();

private void Create\_Clicked(object sender, EventArgs e)

{

if (WalletPicker.SelectedItem == null || CathegoryPicker.SelectedItem == null)

return;

if (!decimal.TryParse(EntrySum.Text, out sum)) return;

CreateOperation(true, sum);

}

private void Create\_ClickedС(object sender, EventArgs e)

{

if (WalletPicker.SelectedItem == null || CathegoryPicker.SelectedItem == null)

return;

Console.WriteLine("-----1");

if (!decimal.TryParse(EntrySum.Text, out sum)) return;

CreateOperation(false, sum);

}

private async void CreateOperation(bool include, decimal sum)

{

Device.StartTimer(TimeSpan.FromSeconds(1), () =>

{

return false;

});

DateTime date = Datepicker.Date;

Console.WriteLine("-----2");

Operation newOperation = new Operation(

context.User.UserId, date.ToString("d"), include, sum, context.Wallets[WalletPicker.SelectedIndex].WalletId, CathegoryPicker.SelectedItem.ToString(), EntryDescription.Text);

Operation isSend = await OperationRepository.SaveOperation(newOperation);

if (isSend != null)

{

Console.WriteLine("-----3");

context.Operations.Add(isSend);

await Navigation.PushAsync(new ListPage(DateTime.Now, context));

}

}

private void EntrySumС\_Focused(object sender, FocusEventArgs e) => xmarkConsume0.IsVisible = false;

private void EntrySumС\_Unfocused(object sender, FocusEventArgs e)

{

xmarkConsume0.IsVisible = !decimal.TryParse(EntrySumС.Text, out sum);

xmarkConsume0.IsVisible = sum > 10000;

}

private void WalletPickerС\_Focused(object sender, FocusEventArgs e) => xmarkConsume1.IsVisible = false;

private void WalletPickerС\_Unfocused(object sender, FocusEventArgs e) => xmarkConsume1.IsVisible = WalletPickerС.SelectedItem == null;

private void CathegoryPickerС\_Focused(object sender, FocusEventArgs e) => xmarkConsume2.IsVisible = false;

private void CathegoryPickerС\_Unfocused(object sender, FocusEventArgs e) => xmarkConsume2.IsVisible = CathegoryPickerС.SelectedItem == null;

private void EntryDescriptionС\_Focused(object sender, FocusEventArgs e) => xmarkConsume3.IsVisible = false;

private void EntryDescriptionС\_Unfocused(object sender, FocusEventArgs e) => xmarkConsume3.IsVisible = EntryDescriptionС.Text.Length > 35;

private void EntrySum\_Focused(object sender, FocusEventArgs e) => xmarkIncome0.IsVisible = false;

private void EntrySum\_Unfocused(object sender, FocusEventArgs e)

{

xmarkIncome0.IsVisible = !decimal.TryParse(EntrySum.Text, out sum);

xmarkIncome0.IsVisible = sum > 10000;

}

private void WalletPicker\_Focused(object sender, FocusEventArgs e) => xmarkIncome1.IsVisible = false;

private void WalletPicker\_Unfocused(object sender, FocusEventArgs e) => xmarkIncome1.IsVisible = WalletPicker.SelectedItem == null;

private void CathegoryPicker\_Focused(object sender, FocusEventArgs e) => xmarkIncome2.IsVisible = false;

private void CathegoryPicker\_Unfocused(object sender, FocusEventArgs e) => xmarkIncome2.IsVisible = CathegoryPicker.SelectedItem == null;

private void EntryDescription\_Focused(object sender, FocusEventArgs e) => xmarkIncome3.IsVisible = false;

private void EntryDescription\_Unfocused(object sender, FocusEventArgs e)

{

try {

if (string.IsNullOrEmpty(EntryDescription.Text) || EntryDescription.Text.Length > 35)

xmarkIncome3.IsVisible = true;

}

catch {

}

}

}

}

Для реализации функциональности «Осуществление анализа данных с предоставлением визуализации (в виде графика) по суммам доходов и/или расходов во всех категориях» была использована сторонняя библиотека «Microcharts». Данная библиотека предоставляет функциональность для создания диаграмм разного вида и другие сопутствующие механики для декорации созданных диаграмм.

using FinanceApp.classes;

using FinanceApplication.core;

using Microcharts;

using SkiaSharp;

using SkiaSharp.Views.Forms;

using System;

using System.Collections.Generic;

using System.Linq;

using Xamarin.Forms;

using Xamarin.Forms.Xaml;

namespace FinanceApplication.views

{

[XamlCompilation(XamlCompilationOptions.Compile)]

public partial class DiagramPage : ContentPage

{

Context context = new Context();

public DiagramPage(Context context)

{

InitializeComponent();

this.context = context;

List<ExtendedCategory> extendedCategories = (from category in context.Categories

join color in context.Colors on category.ColorId equals color.ColorId

select new ExtendedCategory(category.Name, color.DarkMode, color.LightMode, category.IconId, category.CategoryId, context.User.UserId, category.ColorId, category.IsProfit)).ToList();

List<ChartEntry> entries = new List<ChartEntry>();

foreach (ExtendedCategory category in extendedCategories)

entries.Add(new ChartEntry((float)category.CategorySum) {

Color = SKColor.Parse(category.LightMode),

Label = category.Name,

}

);

foreach (ChartEntry item in entries)

{

StackLayout legendItem = new StackLayout { Orientation = StackOrientation.Horizontal };

BoxView colorBax = new BoxView { WidthRequest = 10, HeightRequest = 10, Color = item.Color.ToFormsColor() };

Label label = new Label { Text = item.Label, TextColor = Color.Black };

legendItem.Children.Add(colorBax);

legendItem.Children.Add(label);

CategoriesLegend.Children.Add(legendItem);

Console.WriteLine($"цвет {item.Color} лэбл {item.ValueLabel} лэйбл1 {item.Label} {item.Value}");

}

Chart3.Chart = new DonutChart() { Entries = entries, LabelMode = LabelMode.None};

}

}

}

Функция «Добавление нового счета» является одной из наиболее важных способностей, которая позволяет разделять свои финансы на различные кошельки, карты, крипто кошельки и т.д. Ниже описан код, который реализовывает данную функцию на стороне клиента.

using Android.Service.QuickAccessWallet;

using FinanceApp.classes;

using FinanceApp.classes.Wallets;

using FinanceApplication.core;

using FinanceApplication.core.Category;

using FinanceApplication.icons;

using System;

using System.Linq;

using Xamarin.Forms;

using Xamarin.Forms.Internals;

using Xamarin.Forms.Xaml;

namespace FinanceApplication.views

{

[XamlCompilation(XamlCompilationOptions.Compile)]

public partial class NewCardPage : ContentPage

{

Context context;

ExtendedWallet wallet;

Random random = new Random();

bool delete;

decimal sum = 0;

public NewCardPage(Context context)

{

InitializeComponent();

this.context = context;

PickerType.ItemsSource = context.WalletTypes;

wallet = new ExtendedWallet();

CodeFromConstructions();

Create.Text = "Создать";

Cancel.Text = "Отмена";

delete = false;

wallet.ColorId = random.Next(0, context.Colors.Count - 1);

int IconId = random.Next(0, Icons.WalletsIcons.Length - 1);

WalletImage.BackgroundColor = Color.FromHex(context.Colors.FirstOrDefault(color => color.ColorId == wallet.ColorId).LightMode);

PickerType.ItemsSource = context.WalletTypes;

PickerType.SelectedItem = context.WalletTypes[0];

wallet.IconId = IconId;

wallet.UserId = context.User.UserId;

Console.WriteLine("$$$$$$$$$$$$$$");

Console.WriteLine(wallet);

Console.WriteLine("$$$$$$$$$$$$$$");

walletIcon.Source = ImageSource.FromResource(Icons.WalletsIcons[IconId]);

}

public NewCardPage(Context context, ExtendedWallet wallet)

{

InitializeComponent();

PickerType.ItemsSource = context.WalletTypes;

this.context = context;

this.wallet = wallet;

CodeFromConstructions();

Create.Text = "Сохранить";

Cancel.Text = "Удалить";

EntryName.Text = wallet.Name;

int index = context.WalletTypes.IndexOf(wallet.Type);

PickerType.SelectedIndex = index;

WalletImage.BackgroundColor = Color.FromHex(wallet.LightMode);

delete = true;

}

public void CodeFromConstructions()

{

NavigationPage.SetHasNavigationBar(this, false); sumImage.Source = ImageSource.FromResource(Icons.Iconspath[11]);

walletImage.Source = ImageSource.FromResource(Icons.Iconspath[2]); cathegoryImage.Source = ImageSource.FromResource(Icons.Iconspath[3]);

descriptionImage.Source = ImageSource.FromResource(Icons.Iconspath[5]); xmark1.Source = ImageSource.FromResource(Icons.Iconspath[16]);

xmark2.Source = ImageSource.FromResource(Icons.Iconspath[16]); xmark3.Source = ImageSource.FromResource(Icons.Iconspath[16]);

xmark1.IsVisible = false; xmark2.IsVisible = false;

xmark3.IsVisible = false;

}

private async void Cancel\_Clicked(object sender, EventArgs e)

{

if (!delete) await Navigation.PushAsync(new CardPage(context));

else

{

if (context.Wallets.Count == 1)

{

AlertButton\_Clicked();

return;

}

Cancel.IsEnabled = false;

Create.IsEnabled = false; int index = context.Wallets.FindIndex(wal => wal.WalletId == wallet.WalletId);

await WalletRepository.DeleteWallet(context.Wallets[index]); context.Wallets.Remove(context.Wallets[index]);

await Navigation.PushAsync(new CardPage(context));

}

Cancel.IsEnabled = true; Create.IsEnabled = true;

}

private void EntryName\_Focused(object sender, FocusEventArgs e) { }

private void PickerType\_Focused(object sender, FocusEventArgs e) { }

private void IconButton\_Clicked(object sender, EventArgs e) { }

private void ColorButton\_Clicked(object sender, EventArgs e) { }

private void EntryName\_TextChanged(object sender, TextChangedEventArgs e) { }

private void EntrySum\_TextChanged(object sender, TextChangedEventArgs e) { }

private void EntryName\_Unfocused(object sender, FocusEventArgs e) { }

private void EntrySum\_Focused(object sender, FocusEventArgs e) { }

private void EntrySum\_Unfocused(object sender, FocusEventArgs e) { }

public void Resave(Category category)

{

int index = context.Categories.IndexOf(category);

if (index != -1)

{

context.Categories[index] = category; return;

}

context.Categories.Add(category);

}

private async void Create\_Clicked(object sender, EventArgs e)

{

ValidationBeforeSaving();

Device.StartTimer(TimeSpan.FromSeconds(2), () => {

EntrySum.IsEnabled = false; ColorButton.IsEnabled = false;

IconButton.IsEnabled = false; PickerType.IsEnabled = false;

EntryName.IsEnabled = false; CheckboxOfInclude.IsEnabled = false;

return false;

});

Wallet isSend = await WalletRepository.SaveWallet(new Wallet(wallet.WalletId, context.User.UserId, EntryName.Text, context.WalletTypes[PickerType.SelectedIndex], sum, wallet.ColorId, CheckboxOfInclude.IsChecked, wallet.IconId));

Console.WriteLine("%%%%%%%%%%%%%%%%%111");

Console.WriteLine(isSend);

Console.WriteLine("%%%%%%%%%%%%%%%%%111");

Resave(isSend);

await Navigation.PushAsync(new CardPage(context));

}

private void ValidationBeforeSaving()

{

if (!Validator.ValidateString(EntryName.Text, 15)) return; if (PickerType.SelectedItem == null)

{

xmark3.IsVisible = true;

return;

}

if (!decimal.TryParse(EntrySum.Text, out sum)) return;

if (sum > 10000) return;

}

public void Resave(Wallet wallet)

{

int index = context.Wallets.IndexOf(wallet);

Console.WriteLine("%%%%%%%%%%%%%%%%%");

Console.WriteLine(wallet);

Console.WriteLine("%%%%%%%%%%%%%%%%%");

if (index != -1)

{

context.Wallets[index] = wallet; return;

}

context.Wallets.Add(wallet);

}

private async void AlertButton\_Clicked()

{

await DisplayAlert("Последний кошелек", "кошелек не может быть удален,\nтк он последний", "ОK"); await Navigation.PushAsync(new CardPage(context));

}

}

}

использование калькулятора;

использование конвертера;

выбор валюты для

перевода (для конвертера и калькулятора); выбор стандартной валюты для приложения;

выбор темы приложения; осуществление ведения категорий (добавить, редактировать, удалить), осуществление перевода «денежных средств» между счетами.

## **Функциональное тестирование**

Необходимо провести тестирование разработанного программного средства на

основании тест-кейсов. Тест-кейсы должны охватывать достаточный функционал

разработанного программного средства и демонстрировать его работу на корректных

и некорректных данных.

Проиллюстрировать результаты работы программного средства: скриншотами

экранных форм результатов выполнения тестируемых функций и выходными

формами документов.

Результаты работы программного средства могут быть представлены в

приложении Б.

Провер.

Коропа Е.Н

Т. Контр.

Разработка мобильного приложения «Мои

финансы»

Диаграмм классов

Реценз.

Утверд.

Н. Контр.

Масса

Лит.

Масштаб

КБП

Лист 1

Листов 4

У

КП Т.191016.401 ГЧ

№ докум.

Подпись

Дата

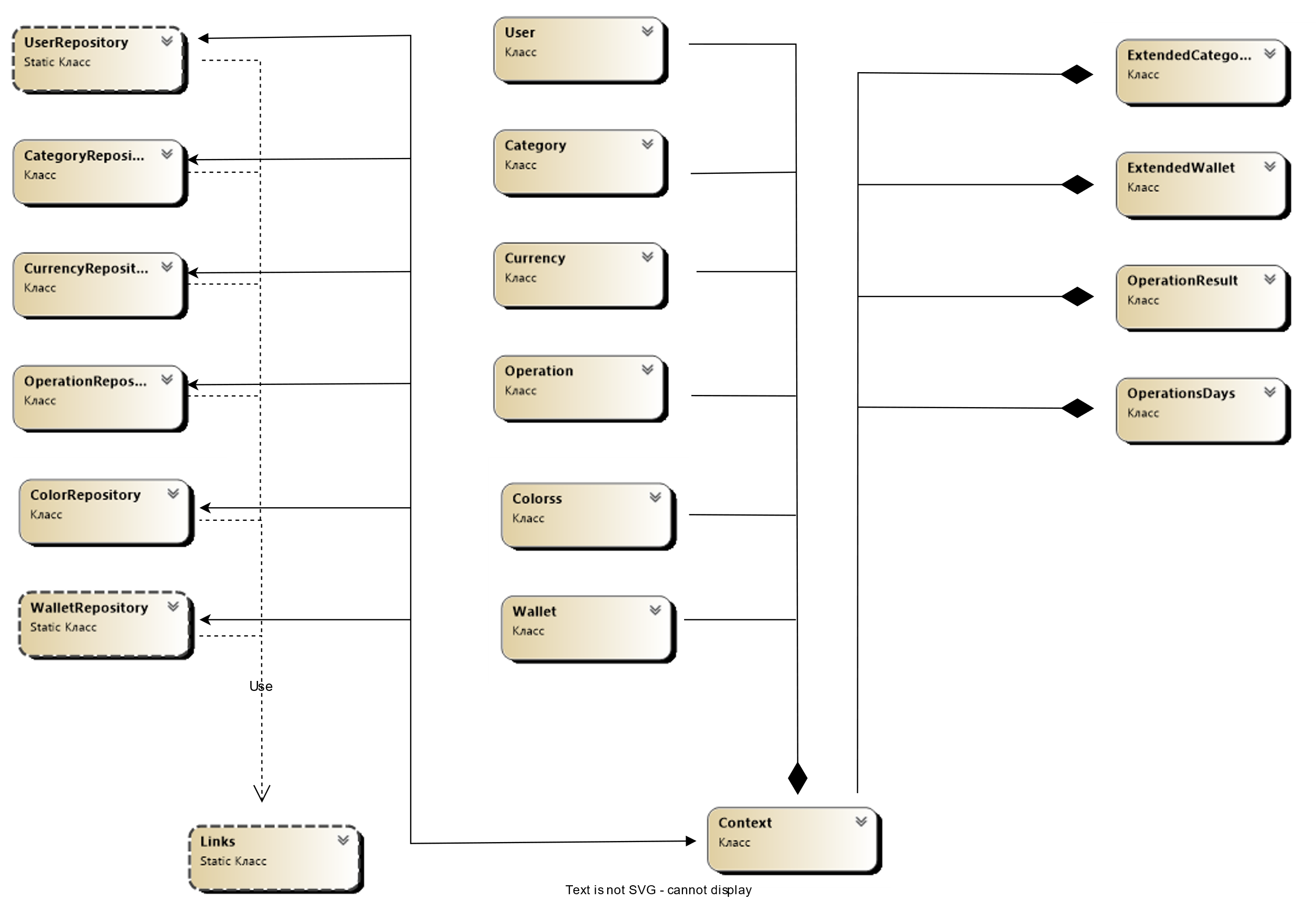
КП Т.191016.401 ГЧ

Взам.инв.№

Подп. и дата

Подп. и дата

Инв.№дубл.



Разраб.

Короленко Д.А.

Провер.

Коропа Е.Н

Т. Контр.

Реценз.

Утверд.

Н. Контр.

Масса

Лит.

Масштаб

КБП

Лист 1

Листов 4

У

КП Т.191016.401 ГЧ

№ докум.

Подпись

Дата

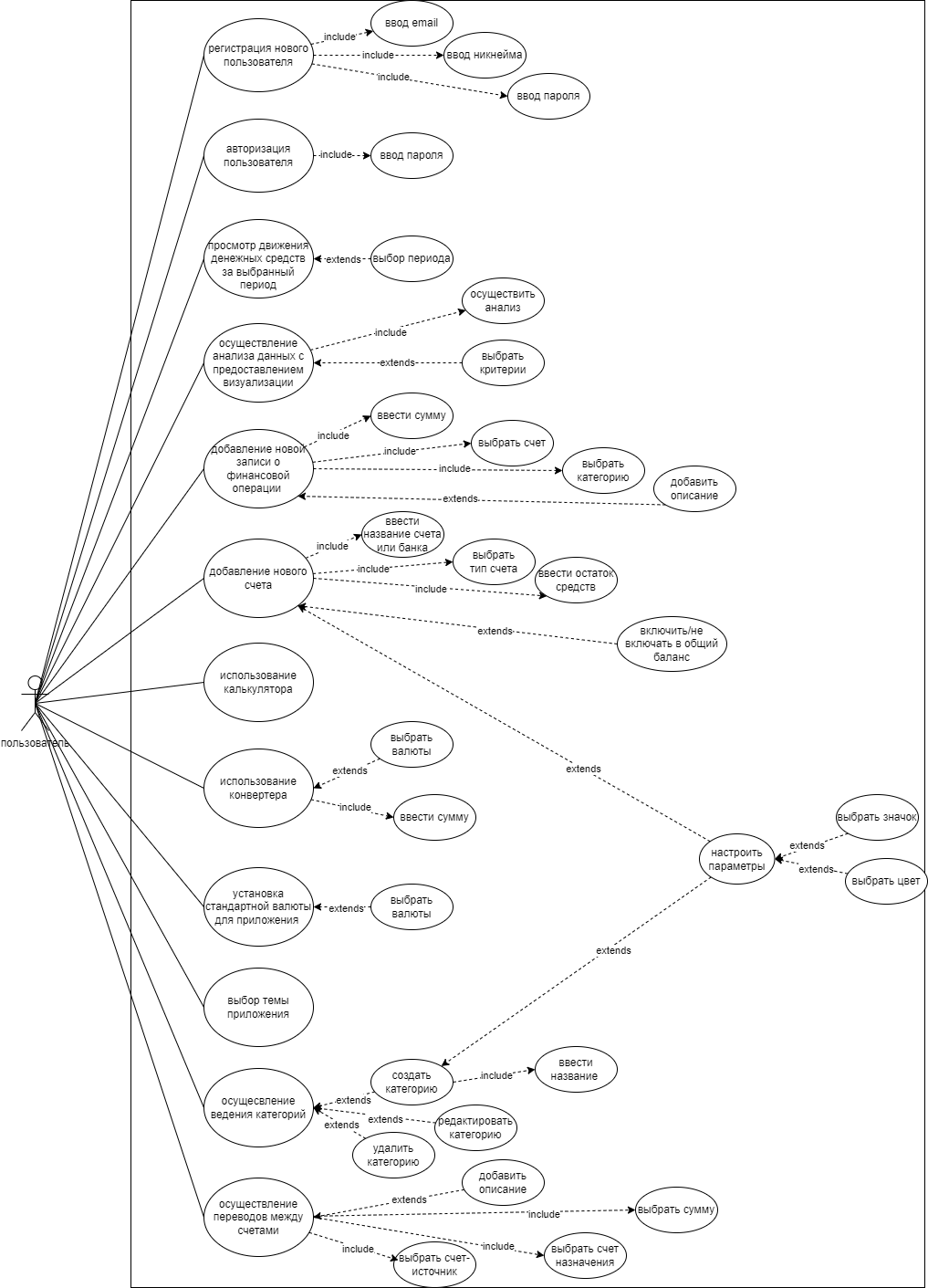
КП Т.191016.401 ГЧ

Взам.инв.№

Подп. и дата

Подп. и дата

Инв.№дубл.



Разраб.

Короленко Д.А.

Разработка мобильного приложения «Мои

финансы»

Диаграмм вариантов использования

Провер.

Коропа Е.Н

Т. Контр.

Реценз.

Утверд.

Н. Контр.

Масса

Лит.

Масштаб

КБП

Лист 1

Листов 4

У

КП Т.191016.401 ГЧ

№ докум.

Подпись

Дата

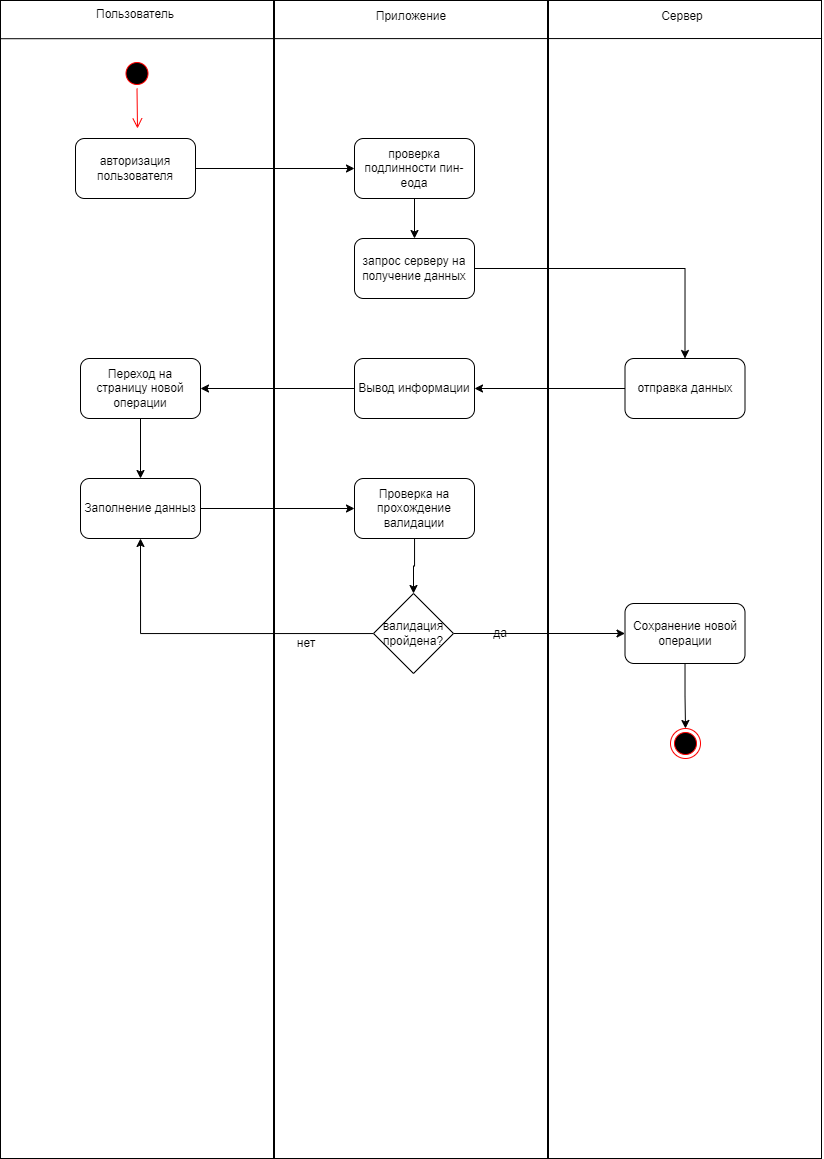
КП Т.191016.401 ГЧ

Взам.инв.№

Подп. и дата

Подп. и дата

Инв.№дубл.



Разраб.

Короленко Д.А.

Разработка мобильного приложения «Мои

финансы»

Диаграмм деятельности

Провер.

Коропа Е.Н

Т. Контр.

Реценз.

Утверд.

Н. Контр.

Масса

Лит.

Масштаб

КБП

Лист 1

Листов 4

У

КП Т.191016.401 ГЧ

№ докум.

Подпись

Дата

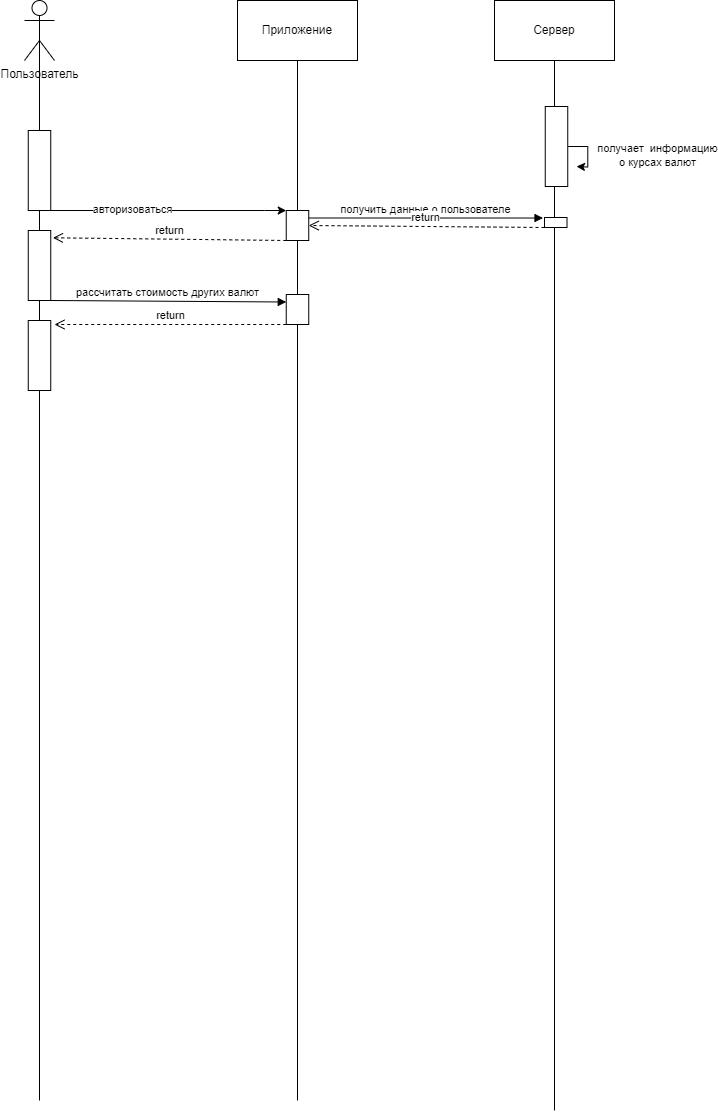
КП Т.191016.401 ГЧ

Взам.инв.№

Подп. и дата

Подп. и дата

Инв.№дубл.



Разраб.

Короленко Д.А.

Разработка мобильного приложения «Мои

финансы»

Диаграмма последовательности

Провер.

Коропа Е.Н

Т. Контр.

Реценз.

Утверд.

Н. Контр.

Масса

Лит.

Масштаб

КБП

Лист 1

Листов 4

У

КП Т.191016.401 ГЧ

№ докум.

Подпись

Дата

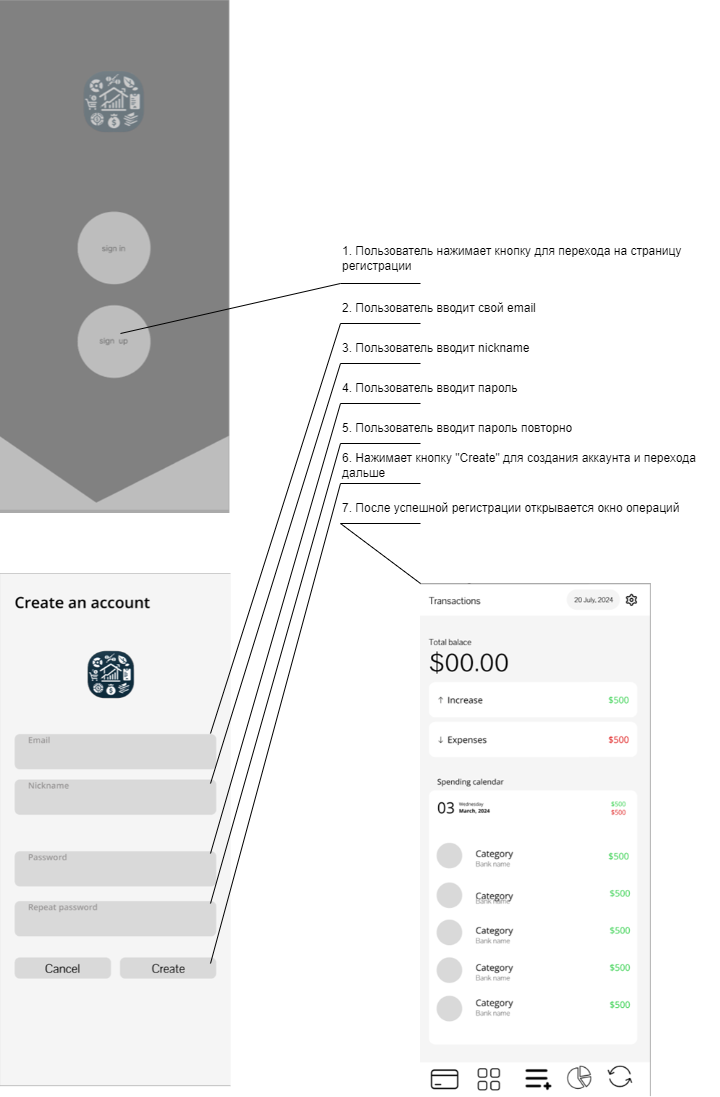
КП Т.191016.401 ГЧ

Взам.инв.№

Подп. и дата

Подп. и дата

Инв.№дубл.



Разраб.

Короленко Д.А.

Разработка мобильного приложения «Мои

финансы»

Диаграмма последовательности