**Министерство науки и высшего образования Российской Федерации**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

«Владимирский государственный университет имени Александра Григорьевича и Николая Григорьевича Столетовых» (ВлГУ)

Кафедра ВТиСУ

**КУРСОВАЯ РАБОТА**

по дисциплине «Разработка мобильных приложений»

на тему «Разработка мобильных приложений на платформе Android. Часть 9. Разработка приложения калькулятор расходов»

Студент: Куимов М. А.

Институт: ИИТЭ

Направление: 09.03.03

Группа: ПИ-121

Руководитель: Шутов А.В.

Владимир 2024 г.

**ОГЛАВЛЕНИЕ**

[**ВВЕДЕНИЕ** 3](#_Toc185916385)

[1. Теоретическая часть 4](#_Toc185916386)

[1.1. Понятие и классификация мобильных приложений 4](#_Toc185916387)

[1.2. Анализ существующих решений 6](#_Toc185916388)

[1.3. Основные этапы разработки мобильных приложений 8](#_Toc185916389)

[2. Практическая часть 10](#_Toc185916390)

[2.1. Описание поставленной задачи и предметной области 10](#_Toc185916391)

[2.2. Выбор технологий и инструментов разработки 11](#_Toc185916392)

[2.3. Общая архитектура приложения 12](#_Toc185916393)

[2.4. Обзор разработанного приложения 15](#_Toc185916394)

[**ЗАКЛЮЧЕНИЕ** 22](#_Toc185916395)

[**СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ** 22](#_Toc185916396)

## **ВВЕДЕНИЕ**

В современном обществе мобильные технологии занимают центральное место, оказывая значительное влияние на повседневную жизнь. Одной из самых популярных платформ для создания мобильных приложений является Android, что объясняется её открытой архитектурой, масштабируемостью и богатым набором инструментов для разработчиков.

Основной целью данной курсовой работы является разработка мобильного приложения для платформы Android, которое будет служить удобным инструментом для управления задачами. Пользователи смогут просматривать, редактировать, добавлять и удалять задачи, а также получать доступ к информации через простой и интуитивно понятный интерфейс с удобной навигацией.

Для достижения этой цели необходимо выполнить следующие задачи:

1. Изучить существующие инструменты и технологии для разработки приложений на Android и выбрать наиболее подходящие.

2. Провести анализ существующих решений для управления задачами на рынке мобильных приложений.

3. Спроектировать архитектуру приложения, соответствующую требуему функционалу.

1. Разработать удобный и понятный пользовательский интерфейс.

5. Выполнить тестирование приложения, выявить и устранить возможные ошибки, обеспечив его стабильную работу на различных устройствах.

Таким образом, выполнение данной курсовой работы позволит получить как теоретические знания, так и практический опыт в области создания мобильных приложений для Android, что станет важным шагом в профессиональном развитии в сфере информационных технологий.

## 1. Теоретическая часть

### 1.1. Понятие и классификация мобильных приложений

Приложение для мобильных устройств представляют собой специализированный тип программного обеспечения, который адаптирован для работы на смартфонах, планшетах и других портативных гаджетах. Их основное назначение — облегчить пользователям выполнение разнообразных задач без привязки к стационарным компьютерам. Эти программы позволяют взаимодействовать с информацией и сервисами в любой момент, что делает их незаменимыми в современном мире.

С развитием технологий мобильные приложения стали неотъемлемой частью повседневной жизни. Смартфоны используются для общения, работы, учебы и развлечений. По мере роста их популярности увеличивается и спрос на разработку программных решений, которые способны охватить разные сферы жизни — от социальных сетей до управления бизнес-процессами.

Приложения можно классифицировать по их функциям и целям. Рассмотрим основные категории:

1. Социальные сети.

Они создаются для общения, обмена информацией и построения сетевых сообществ. Такие приложения, как Facebook и Instagram, позволяют публиковать контент, обмениваться сообщениями и находить единомышленников.

Ключевые особенности:

* 1. персональные профили.
  2. лента новостей.
  3. различное взаимодействие между пользователями.
  4. группы и сообщества.

1. Игровые приложения.

В этой категории представлены как простые игры для расслабления, так и сложные проекты с развитым геймплеем. Например, Candy Crush Saga или Genshin Impact.

Ключевые особенности:

* 1. увлекательный игровой процесс.
  2. качественная графика.
  3. многопользовательские режимы.
  4. монетизация.

1. Маркетинговые приложения.

Программы этого типа помогают компаниям продвигать свои товары и услуги. Например, приложения крупных брендов предоставляют информацию о скидках, акциях и новинках.

Ключевые особенности:

* 1. брендированная оболочка.
  2. инструменты для обратной связи.
  3. программы лояльности и аналитика.

1. Медиа и контентные сервисы.

Эти приложения предоставляют доступ к музыке, видео, текстам и прочим видам контента. Spotify и YouTube — яркие примеры. Они часто работают на основе подписок.

Ключевые особенности:

* 1. персонализация контента.
  2. удобный интерфейс.
  3. монетизация.
  4. поддержка потокового воспроизведения.
  5. возможности взаимодействия между пользователями.

С точки зрения технологий разработки приложения делятся на:

1. Нативные.

Такие программы создаются с использованием инструментов, специфичных для операционных систем, например Kotlin для Android или Swift для iOS. Они обеспечивают максимальную производительность и интеграцию с функциями устройства.

1. Кроссплатформенные.

Эти приложения, такие как TikTok или Instagram, разрабатываются на общих фреймворках (например, Flutter или React-native) и работают на нескольких платформах. Они сочетают доступ к аппаратным возможностям устройства с гибкостью использования веб-технологий.

1. Веб-приложения.

Они функционируют через браузер, а не как установленные программы. Twitter Lite и Spotify Web Player являются примерами такого подхода. Эти решения обеспечивают доступность и универсальность.

Подобная классификация позволяет понять, что мобильные приложения охватывают широкий спектр потребностей пользователей, обеспечивая удобство и доступность в любых условиях.

### 1.2. Анализ существующих решений

На рынке мобильных приложений для контроля расходов уже существуют популярные инструменты, такие как Mint, YNAB (You Need A Budget), PocketGuard и другие. Эти приложения помогают пользователям следить за своими финансами, анализировать расходы и планировать бюджет. У каждого из них есть как преимущества, так и недостатки, что делает их подходящими для разных категорий пользователей.

* Mint

Приложение с богатым функционалом для отслеживания доходов и расходов. Оно автоматически синхронизируется с банковскими счетами, создает категории расходов и предлагает графики для анализа бюджета. Среди преимуществ — бесплатный доступ и простота использования. Однако пользователи часто сталкиваются с ограничениями при интеграции с некоторыми банками и редкими ошибками в автоматической классификации транзакций.

* YNAB (You Need A Budget)

Это мощный инструмент для детального планирования бюджета. Преимущества включают поддержку совместного использования с членами семьи и подробные отчеты. Однако сервис требует подписки, что может быть недостатком для пользователей, ищущих бесплатные решения.

* PocketGuard

Простое в использовании приложение для контроля расходов, которое фокусируется на том, чтобы показать, сколько средств осталось на повседневные траты после учета регулярных платежей. Преимущество — автоматизация расчетов и удобный интерфейс. К недостаткам можно отнести ограниченные возможности для глубокого анализа бюджета.

Анализ существующих решений позволяет выделить ключевые функции, необходимые для успешного приложения:

1. Понятный и удобный интерфейс

Пользователь должен легко добавлять расходы, видеть общую картину бюджета и использовать приложение без необходимости изучать сложные инструкции.

1. Анализ расходов

Графики и отчеты должны быть наглядными, чтобы пользователь мог легко видеть, куда уходят деньги, и корректировать свои финансовые привычки.

1. Синхронизация и доступ с различных устройств

Важно, чтобы приложение поддерживало работу как на смартфонах, так и на планшетах, с возможностью синхронизации данных. Это позволит пользователю отслеживать расходы из любого места.

Таким образом, эффективное приложение для учета расходов должно сочетать простоту, универсальность и функциональность. Оно может стать полезным инструментом как для личного контроля финансов, так и для совместного использования в рамках семьи или небольшой команды.

### 1.3. Основные этапы разработки мобильных приложений

Создание мобильного приложения представляет собой сложный и многогранный процесс, состоящий из этапов анализа, проектирования, разработки, тестирования, развёртывания и дальнейшей поддержки.

Каждый из этих этапов играет ключевую роль в создании качественного, надёжного и соответствующего ожиданиям пользователя продукта.

Для разработки мобильного приложения на платформе Android используются такие технологии, как язык программирования **JavaScript**, фреймворк **React Native**, серверная платформа **Node.js**, а также базы данных, такие как **SQLite**.

Основные этапы разработки включают:

1. Анализ требований

* Определение цели приложения: что оно должно делать и какие задачи решать.
* Изучение потребностей целевой аудитории, выявление их предпочтений и проблем.
* Постановка задач и разработка списка ключевых функций, которые будут реализованы в приложении.

1. Проектирование

* Создание структуры приложения, определение основных модулей и связей между ними.
* Разработка прототипов экранов: черновые макеты интерфейса, отражающие пользовательский опыт (UX).
* Дизайн интерфейса (UI): создание привлекательного визуального стиля, удобного для взаимодействия.

1. Разработка
2. Тестирование

* Функциональное тестирование.
* Нагрузочное тестирование.
* Тестирование совместимости.

1. Развёртывание

* Подготовка приложения к публикации: финальная проверка, создание необходимых сертификатов и метаданных.
* Публикация в магазинах приложений, таких как **Google Play** и **App Store**.

1. Поддержка и обновление

* Мониторинг работы приложения и устранение возможных ошибок после выпуска.
* Добавление новых функций, основанных на обратной связи пользователей.
* Регулярное обновление для соответствия современным требованиям безопасности и производительности.

Эти этапы обеспечивают полный жизненный цикл приложения и позволяют достичь оптимального результата.

## 2. Практическая часть

### 2.1. Описание поставленной задачи и предметной области

Предметной областью данной курсовой работы является создание мобильного приложения для учета и анализа расходов пользователей. Основная цель — предоставить пользователям удобный инструмент для отслеживания финансовых операций и управления бюджетом.

Приложение предназначено для людей, которые хотят оптимизировать свои финансы, вести учет трат и принимать осознанные решения о расходах. Основными пользователями являются частные лица, стремящиеся к финансовой дисциплине и планированию бюджета.

З**адача проекта** — создать мобильное приложение для платформы Android, которое обеспечивает функциональные возможности для отслеживания финансовых операций.

Функциональные возможности далее будут описаны в Таблице 1.

|  |  |
| --- | --- |
| Функциональные возможности | |
| 1. | Регистрация и авторизация пользователей для доступа к данным. |
| 2. | Возможность редактирования профиля пользователя. |
| 3. | Возможность создания, редактирования и удаления финансовых счетов. |
| 4. | Отображение перечня финансовых счетов. |
| 5. | Возможность создания, финансовых операций для конкретного счета в виде расходов и доходов, включающих в себя название, описание и сумму. |
| 6. | Отображение истории операций для конкретного счета в виде расходов и доходов, включающих в себя название, описание и сумму. |
| 7. | Отображение суммы накоплений для конкретного счета в виде автоматического подсчета всей истории операций по данному счету. |

Таблица 1 – Функциональные возможности приложения

### 2.2. Выбор технологий и инструментов разработки

Для реализации данного проекта было принято решение использовать проверенные и современные технологии, которые обеспечивают стабильную работу приложения и соответствуют актуальным стандартам разработки. Эти технологии были выбраны благодаря их популярности, функциональности и поддержке сообщества, что позволяет создать качественное мобильное приложение для управления расходами.

1. React Native

React Native является основным инструментом разработки мобильного приложения, обеспечивая его кроссплатформенность и высокую производительность. Это позволяет приложению работать как на iOS, так и на Android, что существенно расширяет его аудиторию. React Native предоставляет гибкий подход к созданию пользовательских интерфейсов и имеет широкую библиотеку компонентов.

1. Axios

Для выполнения HTTP-запросов к серверу используется библиотека Axios. Она позволяет эффективно работать с REST API, управлять асинхронными запросами и обрабатывать данные, полученные с сервера. Axios также поддерживает обработку ошибок и кэширование запросов, что улучшает производительность приложения.

1. Node.js и CMS Strapi

Серверная часть приложения разработана на платформе Node.js с использованием CMS Strapi, которая обеспечивает управление контентом и автоматическую генерацию REST API.

* Node.js гарантирует высокую производительность и позволяет обрабатывать множество запросов одновременно.
* CMS Strapi упрощает настройку структуры данных и администрирование, предоставляя удобный интерфейс для управления записями о расходах и категориях.

1. SQLite

В качестве базы данных используется SQLite, которая хранит данные о расходах, категориях и пользователях. SQLite отлично подходит для мобильных приложений благодаря своей лёгкости, автономности и быстродействию. Она легко интегрируется с серверной частью через Strapi, что упрощает управление данными.

1. Postman

Для тестирования и отладки REST API используется инструмент Postman. Он позволяет проверить корректность работы серверной части, убедиться в правильной обработке запросов и минимизировать ошибки во взаимодействии между клиентом и сервером.

Эти технологии были выбраны из-за их совместимости и возможности интеграции. Использование React Native позволяет оптимизировать разработку под две платформы сразу, а связка Node.js и CMS Strapi обеспечивает удобное управление серверной частью. SQLite, благодаря своей лёгкости, делает приложение быстрым и эффективным даже на устройствах с ограниченными ресурсами.

### 2.3. Общая архитектура приложения

Разработка архитектуры приложения для калькулятора расходов — важнейший этап, влияющий на стабильность, масштабируемость и удобство работы с системой. Для реализации была выбрана технология React Native, обеспечивающая кроссплатформенность, а взаимодействие с серверной частью осуществляется через REST API. Графический пример общей архитектуры на рис. 1.

Основные компоненты архитектуры:

1. Клиентская часть

Приложение предоставляет удобный интерфейс, позволяющий пользователю легко вводить данные о расходах и доходах на разных счетах, повышая финансовую грамотность пользователя.

Основные компоненты:

1. Пользовательский интерфейс (UI):

* Включает основные экраны: добавление расходов и доходов, регистрации и авторизации пользователя, редактирование аккаунта, добавление различных счетов для ведения раздельного контроля финансов для каждого счета пользователя.
* Поддерживает адаптивный дизайн для корректного отображения на различных устройствах.
* Удобная навигация, обеспечивающая доступ к ключевым функциям.

1. Сетевой слой:

* Используется библиотека Axios для выполнения асинхронных HTTP-запросов к серверу.
* Обрабатываются запросы на добавление, удаление, изменение и получение данных о расходах
* Поддерживает CRUD операции для корректной работы с запросами данных.

1. Модели данных:

* Хранят информацию о расходах, категориях и пользовательских предпочтениях.
* Позволяют эффективно управлять состоянием приложения.

1. Логика взаимодействия:

* Реализуется через функциональные компоненты и хуки React.
* Управляет обработкой пользовательских действий, таких как ввод расходов или выбор временного периода для отчета.

2. Серверная часть

Серверная архитектура поддерживает высокую производительность и лёгкость администрирования.

Основные компоненты:

a) CMS Strapi:

* Генерирует REST API для взаимодействия с клиентской частью.

b) REST API:

Обеспечивает стандартные операции с данными (CRUD):

* **GET:** Получение различных счетов, расходов и доходов по каждому счету.
* **POST:** Добавление новых данных.
* **PUT:** Изменение существующей информации.
* **DELETE:** Удаление данных.

c) SQLite

* Хранит информацию о расходах и пользовательских данных.
* Интеграция с Strapi позволяет автоматически управлять данными.

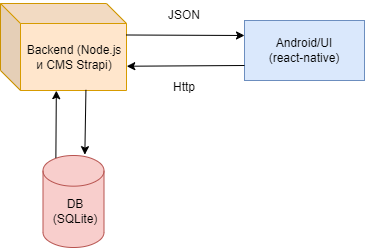


Рисунок 1 – Общая архитектура приложения

Эта архитектура гарантирует надёжность, масштабируемость и удобство использования. Приложение для учета расходов будет функциональным, интуитивно понятным и устойчивым, отвечая всем современным требованиям и предоставляя пользователям эффективный инструмент для управления своими финансами

## 2.4. Обзор разработанного приложения

При реализации разработанного приложения первичным экраном для любого пользователя стал экран регистрации. Регистрации пользователь должен заполнить соответствующие поля: почты, логина, пароля и подтверждения пароля. Экран регистрации представлен на рис 2.

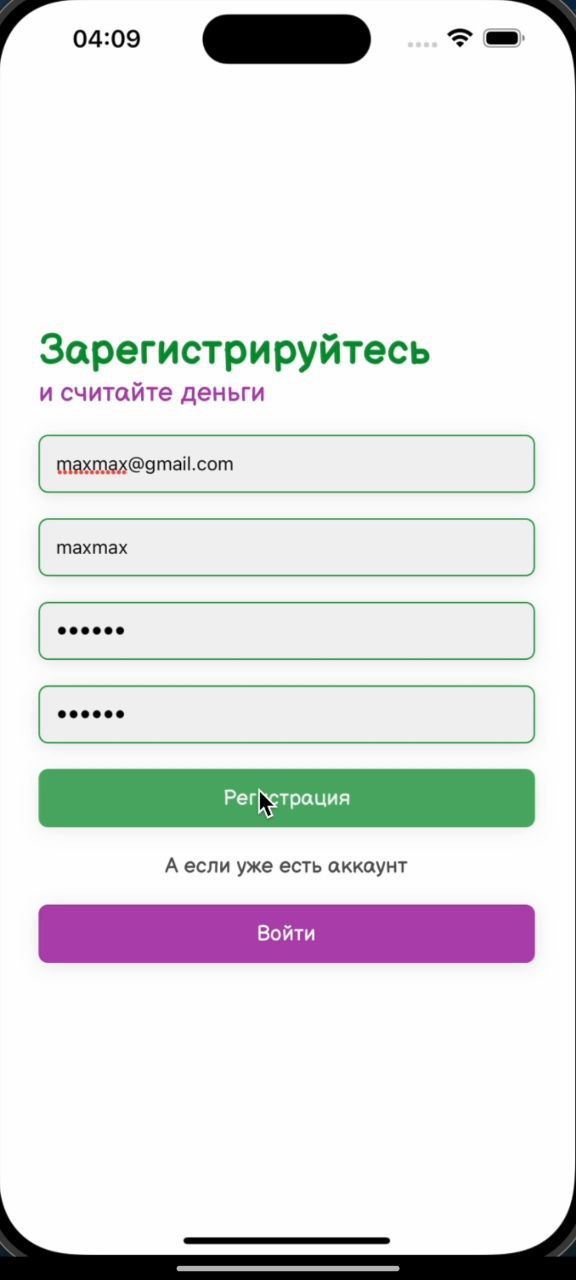


Рис. 2 – Экран регистрации пользователя

В случае, если у пользователя уже есть аккаунт, внизу экрана регистрации имеется навигационная кнопка, переходя на экран авторизации. Экран авторизации изображен на рис. 3. На этом экране есть два интуитивно понятных поля – логин и пароль, а также кнопка возврата к экрану регистрации в случае ошибки пользователя. Это позволяет пользователю легко войти в приложение и продолжить дальнейший опыт использования.

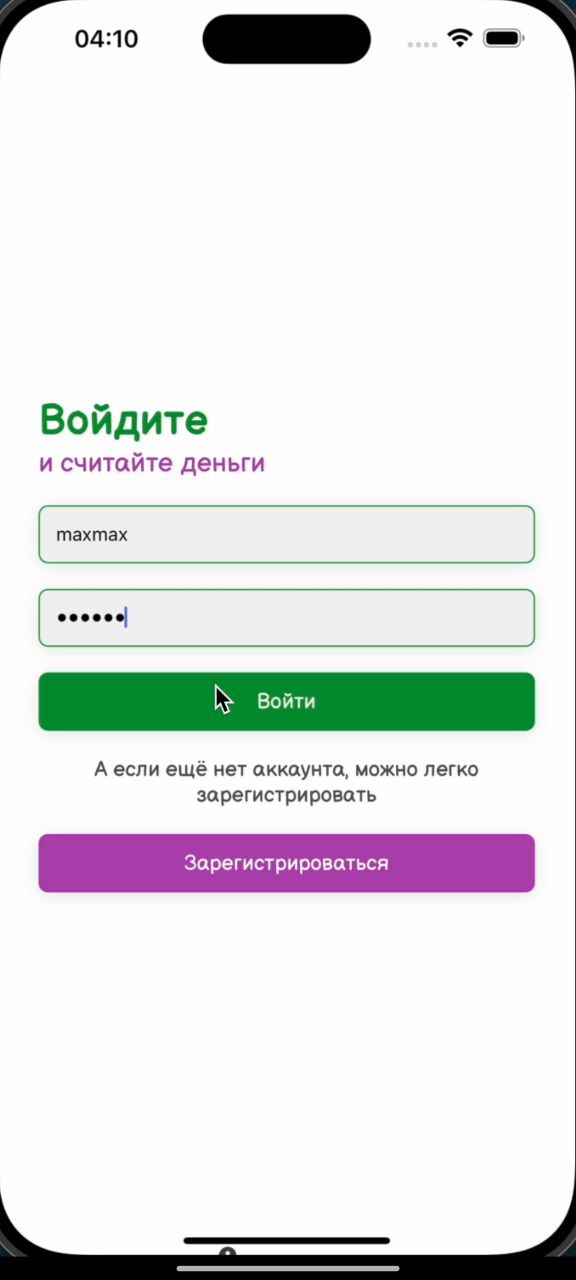


Рис. 3 – Окно входа

После успешной идентификации, пользователь переходит в свой профиль, где может наблюдать свой логин и аватар. Окно профиля изображено на рис. 4.

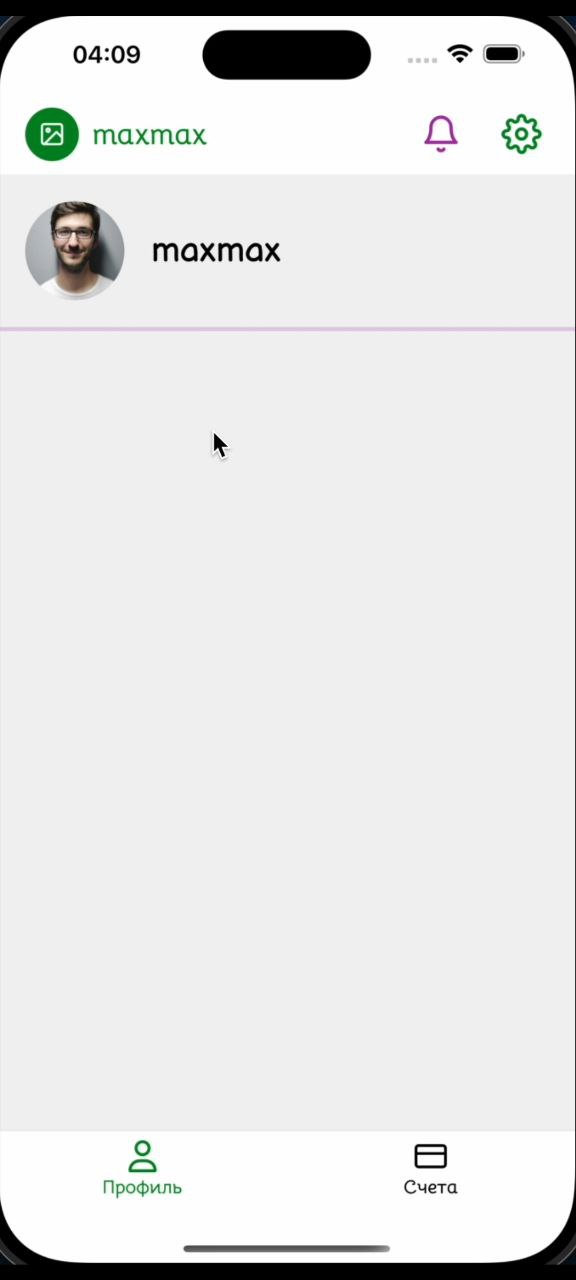


Рис. 4 – Окно профиля

При нужде, пользователь по нажатию на профиль может перейти к окну редактированию профиля, где ему доступно поменять логин и аватар с помощью изображений формата jpg, а также выйти из аккаунта. Пример окна редактирования профиля на рис. 5

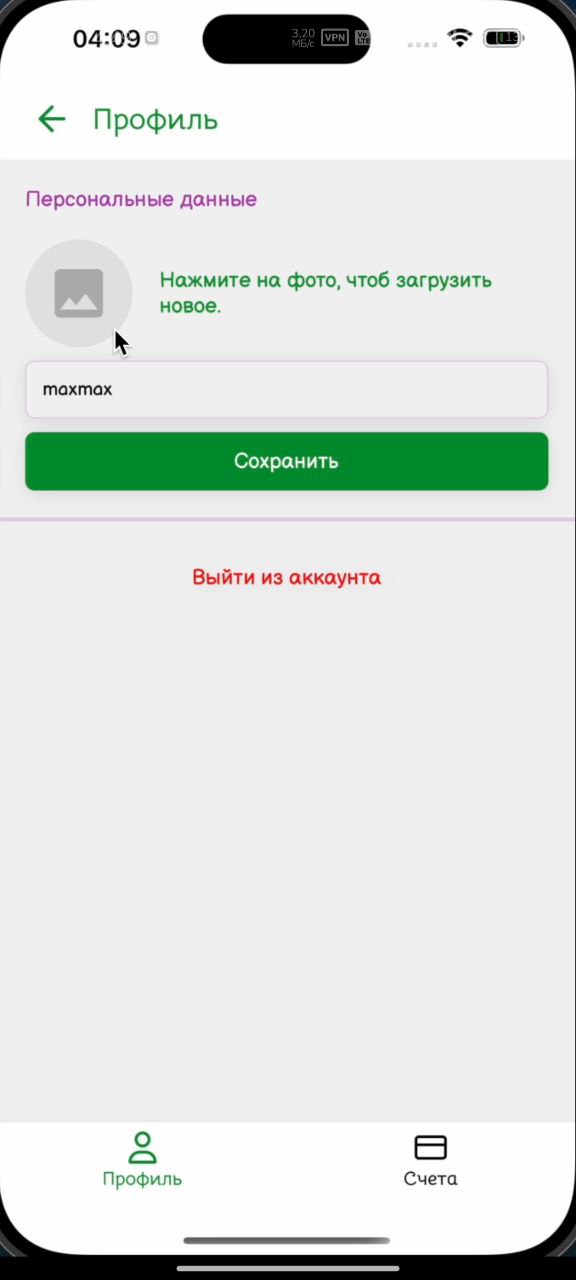


Рис. 5 – Окно редактирования профиля

С помощью кнопки «Счета» в нижнем навигационном меню пользователь может перейти к окну со списком всех активных счетов пользователя рис. 6. На экране будет представлено поле с интуитивной подсказкой, где нужно ввести название счета, и по нажатию кнопки «Сохранить» создастся новый счет с указанным названием.

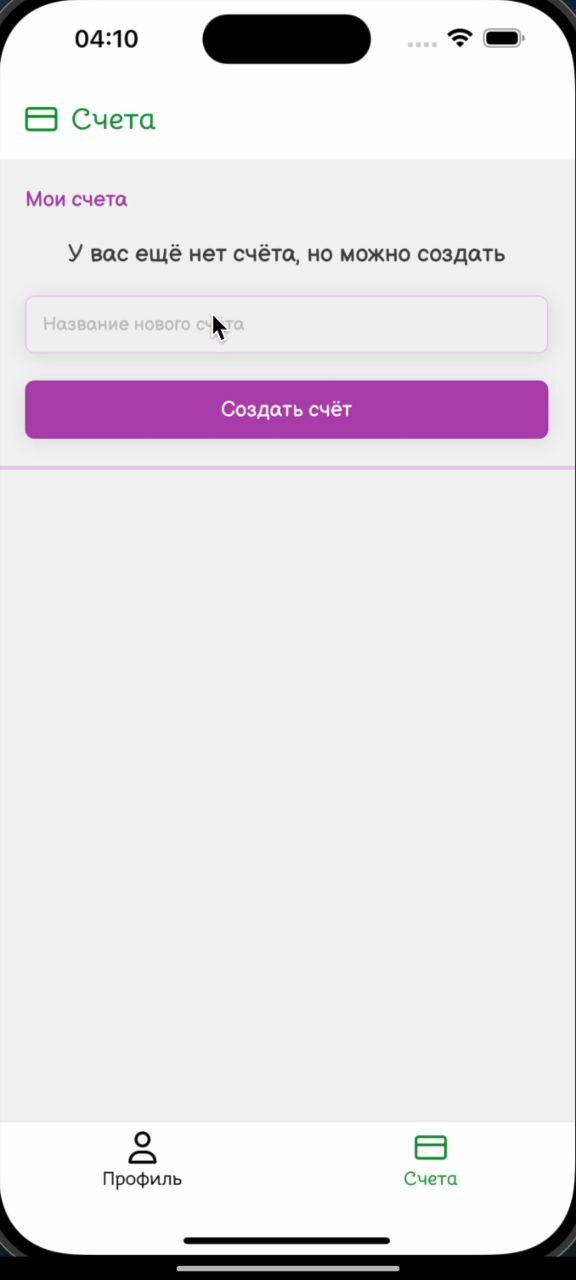


Рис. 6 – Окно списка счетов

По нажатию на один из счетов, пользователь переходит к окну счета рис. 7. В данном окне отображается название счета и общая сумма накоплений автоматически рассчитанная исходя из всех доходов и расходов внесенных пользователем. Удаление счета реализовано с помощью соответствующей кнопки. Также имеется кнопка добавления операции, после заполнения полей названия операции и суммы операции. Для реализации дохода, пользователь должен ввести положительное число, для реализации расхода – отрицательное. Внизу окна счета имеется список истории операций, который в свою очередь передает все операции доходов и расходов занесенных пользователем.

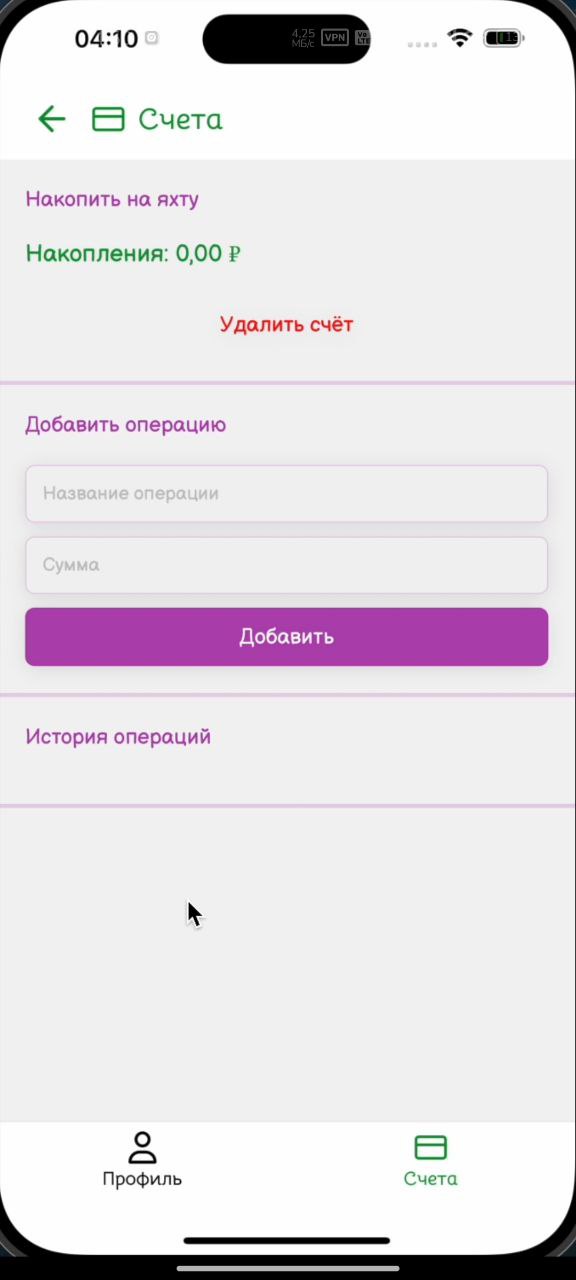


Рис. 7 – Окно счета

При удалении счета, пользователь нажимает кнопку «Удалить счет» и переходит к окну удаления рис. 8. На этом окне приложение уточняет, выполнить или отменить данную операцию.

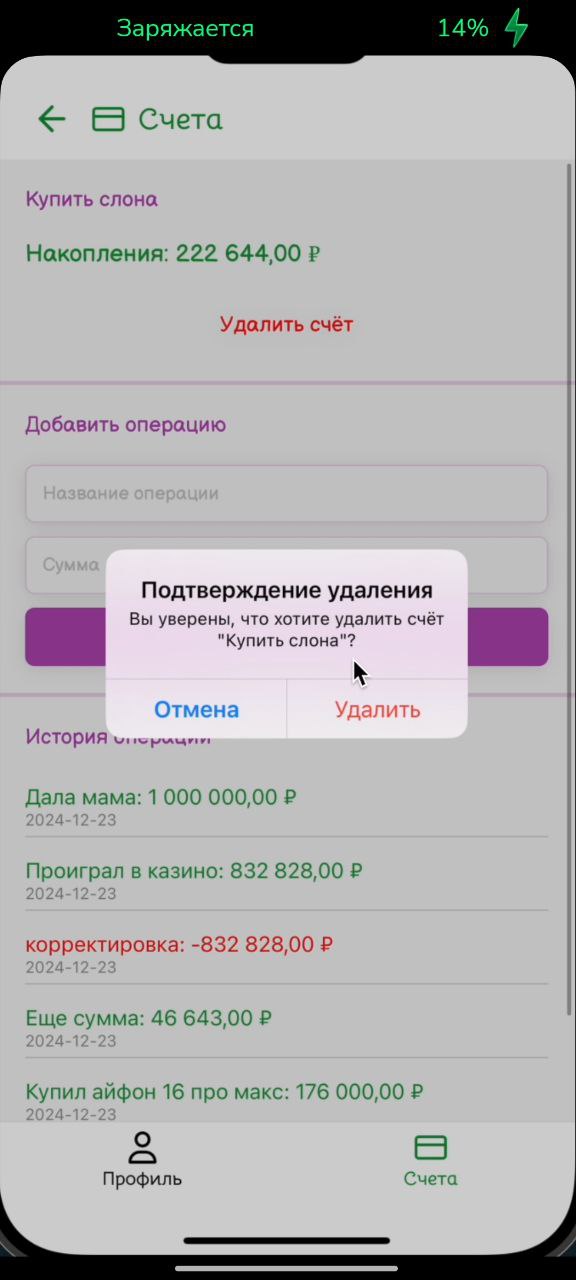


Рис. 8 – Окно счета

## **ЗАКЛЮЧЕНИЕ**

В ходе выполнения курсовой работы была достигнута основная цель как разработка мобильного приложения «Калькулятор расходов». Анализ предметной области и существующих решений позволил выделить ключевые требования к функциональности приложения, что стало основой для его проектирования и реализации.

На этапе проектирования была создана архитектура приложения, включающая клиентскую часть, разработанную с использованием React Native, и серверную часть на платформе Node.js с CMS Strapi. Взаимодействие между клиентом и сервером организовано через REST API, что обеспечило быстрое и безопасное выполнение запросов.

Разработанное приложение предоставляет основные функции для управления расходами:

1. Регистрация и авторизация пользователей.
2. Добавление, редактирование и удаление записей о расходах.
3. Обработка ошибок, таких как отсутствие соединения с сервером, с выводом соответствующих уведомлений.

В процессе тестирования приложение продемонстрировало стабильную работу, соответствие заданным требованиям и удобство использования. Благодаря использованию React Native, приложение обладает кроссплатформенностью и поддерживает как Android, так и iOS, что расширяет аудиторию пользователей.

Таким образом, задачи курсовой работы были успешно выполнены. Разработанное приложение «Калькулятор расходов» сочетает в себе функциональность, надёжность и удобство использования, предоставляя пользователям эффективный инструмент для управления финансами.

## **СПИСОК ИСПОЛЬЗОВАННОЙ ЛИТЕРАТУРЫ**

1. Лернер Э., Дитчфилд Э. React и Redux: разработка веб-приложений. — СПб.: Питер, 2019. — 352 с.
2. Соловьев А.М. Архитектура мобильных приложений. – М.: Вильямс, 2015. – 288 с.
3. Хорхе А. Разработка кроссплатформенных мобильных приложений на React Native. — М.: ДМК Пресс, 2020. — 288 с.
4. Node.js. Официальная документация по разработке серверных приложений [Электронный ресурс]. — Режим доступа: https://nodejs.org/.

5. CMS Strapi. Документация по использованию и настройке [Электронный ресурс]. — Режим доступа: https://strapi.io/documentation/

6. Романенко А.В. Разработка мобильных приложений для Android. – М.: БХВ-Петербург, 2017. – 448 с.

7. Axios. Библиотека для выполнения HTTP-запросов [Электронный ресурс]. — Режим доступа: https://axios-http.com/.