|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **РОСЖЕЛДОР**  **ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ**  **ВЫСШЕГО ОБРАЗОВАНИЯ**  **СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**  **ПУТЕЙ СООБЩЕНИЯ (СГУПС)**   |  |  |  |  | | --- | --- | --- | --- | | **К защите:** |  |  | | | **Заведующий кафедрой** | **Информационные** | |  | | **технологии транспорта** | | |  | |  | д-р техн. наук, профессор | |  | |  |  | В.И. Хабаров | | | *подпись* |  | *инициалы, фамилия* | | |  |  |  | | | *дата* |  |  | |   **ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**  **(БАКАЛАВРСКАЯ РАБОТА)**   |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Тема:** | Разработка мобильного приложения для определения мошеннических транзакций в банках | | | | |  | | | | | |  | | | | | |  |  | БР.БПИ.09.2025 |  |  | |  |  | *шифр документа* |  |  |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | --- | |  |  | **Выполнил** |  |  |  | **Руководитель** | |  |  | К. В. Рязанов |  |  |  | к.т.н., доцент  В. Г. Кобылянский | | *подпись* |  | *инициалы, фамилия* |  | *подпись* |  | *инициалы, фамилия* | |  |  |  |  |  |  |  | | *дата* |  |  |  | *дата* |  |  |  |  |  |  |  |  | | --- | --- | --- | --- | --- | | **Консультанты по разделам** |  |  |  |  | | 1 |  |  |  |  | |  |  |  |  |  | | 2 |  |  |  |  | |  |  |  |  |  | | 3 |  |  |  |  | |  |  |  |  |  | | Нормоконтролер работы |  |  |  | ст. преп.  Т. А. Стенникова | |  |  | *подпись* |  |  | |  |  |  |  |  | |  |  | *дата* |  |  |   **2025 г.** |

**СИБИРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ**

**ПУТЕЙ СООБЩЕНИЯ (СГУПС)**

Факультет: Бизнес-информатики

Кафедра: Информационные технологии транспорта

Направление: 09.03.03 «Прикладная информатика»

Профиль: Интеллектуальные транспортные системы на транспорте

|  |  |
| --- | --- |
|  | ***УТВЕРЖДАЮ****: зав. кафедрой «Информационные технологии транспорта»*  д-р техн. наук, проф.  В. И. Хабаров |
|  | *«\_\_\_\_»\_\_\_\_\_\_\_\_\_\_ 2025 г.* |

**З А Д А Н И Е**

**на выполнение выпускной квалификационной работы**

|  |  |  |  |
| --- | --- | --- | --- |
| студенту | Рязанову Кириллу Вадимовичу | | |
|  |  | | |
| 1. Тема «Разработка мобильного приложения для определения мошеннических транзакций в банках» утверждена приказом № 203/c от «30» мая 2022 г. | | | |
| 2. Задание выдано «12» мая 2022 г. | | | |
| 3. Срок сдачи законченной работы на кафедру «20» июня 2022 г. | | | |
| 4. Исходные данные: данные, полученные в ходе прохождения преддипломной практики | | | |
| 5. Содержание расчетно-пояснительной записки | | | |
| Наименование разделов и вопросов | | Примерное количество страниц | График (сроки) выполнения |
| Введение | | 3 | 17.04.2025 |
| Аналитическое исследование | | 14 | 18.04.2025 |
| Проектирование информационной системы | | 12 | 22.04.2025 |
| Реализация ИС | | 11 | 24.04.2025 |
| Тестирование ИС | | 7 | 27.04.2025 |
| Заключение | | 2 | 27.04.2025 |

6. Содержание и объемы графической части

|  |  |  |
| --- | --- | --- |
| Наименование графического документа (чертежа, схемы, графика) | Количество  листов  формата А1 | График  (сроки)  выполнения |
| Презентация PowerPoint | 12 | 05.05.2025 |

7. Консультанты по разделам

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Наименование  раздела | | Фамилия, И. О.  консультанта | Подпись консультанта,  дата выдачи задания | |
|  |  | |  |  | |
| Руководитель | |  | | | В. Г. Кобылянский |
|  | | *(подпись, фамилия, И.О.)* | | |  |
| Задание к использованию принял | |  | | | К.В. Рязанов |
|  | | *(подпись студента)* | | |  |

УДК 004.41

**АННОТАЦИЯ**

В отчете о преддипломной практике 63 страница, 11 рисунков, 21 источников.

Ключевые слова: *мобильный банкинг, мошеннические транзакции, антифрод-система, машинное обучение, мобильное приложение, Android, Kotlin, уведомление пользователя, пользовательский опыт, прототип, преддипломная практика.*

Выпускная квалификационная работа посвящена разработке прототипа мобильного приложения для платформы Android (Kotlin). Проект демонстрирует улучшенный подход к информированию пользователей о рисках мошеннических транзакций. В отличие от традиционных реактивных методов, прототип проактивно и наглядно визуализирует уровень риска до завершения операции. Реализовано взаимодействие с внешним API антифрод-системы (через Retrofit, протокол KServe V2) и использование локальной базы данных SQLite. Основная цель – демонстрация концепции повышения осведомленности и осознанного контроля пользователя, улучшая его опыт взаимодействия с системами безопасности мобильного банкинга.

**ABSTRACT**

The work contains 63 pages, 11 figures, 21 sources.

Keywords: *mobile banking, fraudulent transactions, anti-fraud system, machine learning, mobile application, Android, Kotlin, User notification, user experience, prototype, pre-graduate practice.*

The final qualifying work is devoted to the development of a prototype of a mobile application for the Android platform (Kotlin). The project demonstrates an improved approach to informing users about the risks of fraudulent transactions. Unlike traditional reactive methods, the prototype proactively and visually visualizes the risk level before the operation is completed. Implemented interaction with the external API of the anti-fraud system (via Retrofit, Server V2 protocol) and the use of a local SQLite database. The main goal is to demonstrate the concept of increasing user awareness and conscious control, improving their experience of interacting with mobile banking security systems.

# ТЕРМИНЫ И ОПРЕДЕЛЕНИЯ

1. Антифрод-система (Anti-Fraud Engine) – Система, предназначенная для выявления и предотвращения мошеннических операций в режиме реального времени или постфактум, часто с использованием анализа данных, правил и алгоритмов машинного обучения.
2. Машинное обучение (ML) – Класс методов искусственного интеллекта, характерной чертой которых является не прямое решение задачи, а обучение в процессе применения решений множества сходных задач.
3. Мобильное приложение – Программное обеспечение, разработанное для работы на мобильных устройствах, таких как смартфоны и планшеты.
4. Мошенническая транзакция – Финансовая операция (перевод, платеж), совершенная злоумышленником без ведома или согласия легитимного владельца счета с целью хищения денежных средств или получения несанкционированного доступа к информации.
5. Пользовательский интерфейс (UI / GUI) – Совокупность средств (экраны, кнопки, меню), при помощи которых пользователь взаимодействует с программной системой.
6. Прототип – Ранняя, часто упрощенная, рабочая версия программного продукта, созданная для демонстрации основной концепции, проверки гипотез, сбора обратной связи или тестирования ключевого функционала.
7. Социальная инженерия – Метод получения несанкционированного доступа к информации или системам, основанный на использовании психологических манипуляций людьми, а не на технических взломах.
8. Транзакция – Любая операция, связанная с движением денежных средств по счету.
9. Фишинг – Вид интернет-мошенничества, заключающийся в создании поддельных веб-сайтов, рассылке электронных писем или сообщений от имени банков или других организаций с целью выманивания у пользователей конфиденциальной информации.
10. API (Application Programming Interface) – Программный интерфейс приложения; набор готовых классов, процедур, функций, структур и констант, предоставляемых приложением для использования во внешних программных продуктах.
11. ASP.NET Core – Кроссплатформенный фреймворк с открытым исходным кодом для создания современных веб-приложений и API, разработанный Microsoft.
12. Backend (Серверный модуль) – Программно-аппаратная часть сервиса, отвечающая за его внутреннюю логику, обработку данных, взаимодействие с базой данных и другими системами.
13. Frontend (Клиентский модуль) – Пользовательский интерфейс; часть сервиса, с которой непосредственно взаимодействует пользователь.
14. Kotlin – Статически типизированный язык программирования, работающий поверх виртуальной машины Java (JVM) и официально поддерживаемый Google для разработки приложений под ОС Android.
15. PostgreSQL – Свободная объектно-реляционная система управления базами данных (СУБД).
16. Android Studio – Официальная интегрированная среда разработки (IDE) для платформы Android.
17. UML (Unified Modeling Language) – Унифицированный язык графического описания для объектного моделирования в области разработки программного обеспечения, моделирования бизнес-процессов, системного проектирования и отображения организационных структур.

# ПЕРЕЧЕНЬ ОБОЗНАЧЕНИЙ И СОКРАЩЕНИЙ

1. ИС – информационная система.
2. IDE – интегрированная среда разработки.
3. ACID – Atomicity, Consistency, Isolation, Durability (Атомарность, Согласованность, Изолированность, Долговечность)
4. API – Application Programming Interface (Программный интерфейс приложения)
5. CPU – Central Processing Unit (Центральный процессор)
6. GUI – Graphical User Interface (Графический интерфейс пользователя)
7. ML – Machine Learning (Машинное обучение)
8. ОС – Операционная система
9. ПО – Программное обеспечение
10. SDK – Software Development Kit (Комплект для разработки программного обеспечения)
11. SQL – Structured Query Language (Язык структурированных запросов)
12. UI – User Interface (Пользовательский интерфейс)
13. UML – Unified Modeling Language (Унифицированный язык моделирования)
14. ВТБ – Банк ВТБ (используется как пример)
15. ЕСИА – Единая система идентификации и аутентификации
16. ЖКХ – Жилищно-коммунальное хозяйство
17. ИНН – Идентификационный номер налогоплательщика
18. НИУ ВШЭ – Национальный исследовательский университет «Высшая школа экономики»
19. РФ – Российская Федерация
20. СУБД – Система Управления Базами Данных
21. ФНС – Федеральная налоговая служба
22. ЦБ РФ – Центральный банк Российской Федерации

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 9](#__RefHeading___Toc2410_2887410827)

[1 Аналитическое исследование 12](#__RefHeading___Toc6444_3885207264)

[1.1 Описание предметной области 12](#__RefHeading___Toc6446_3885207264)

[1.2 Анализ аналогов 16](#__RefHeading___Toc6502_3885207264)

[1.3 Актуальность разработки 21](#__RefHeading___Toc6504_3885207264)

[2 Проектирование информационной системы 26](#__RefHeading___Toc6448_3885207264)

[2.1 Моделирование бизнес-процессов ИС 26](#__RefHeading___Toc7440_3885207264)

[2.2 Структура ИС и ее средства разработки 29](#__RefHeading___Toc7442_3885207264)

[2.3 Требования к ИС 34](#__RefHeading___Toc7444_3885207264)

[3 Реализация ИС 38](#__RefHeading___Toc3106_3748947357)

[3.1 Разработка интерфейсов мобильного приложения 38](#__RefHeading___Toc7444_3885207264_%D0%9A%D0%BE%D0%BF%D0%B8%D1%8F_1)

[3.2 Разработка взаимодействия с Антифрод-системой 46](#__RefHeading___Toc7444_3885207264_%D0%9A%D0%BE%D0%BF%D0%B8%D1%8F_2)

[4 Тестирование ИС 51](#__RefHeading___Toc32250_2731718327)

[4.1 Методология тестирования 51](#__RefHeading___Toc48834_2731718327)

[4.2 Проведение процедуры тестирования 53](#__RefHeading___Toc48836_2731718327)

[ЗАКЛЮЧЕНИЕ 59](#__RefHeading___Toc2420_2887410827)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 61](#__RefHeading___Toc2422_2887410827)

# ВВЕДЕНИЕ

В современную эпоху цифровой трансформации мобильные банковские приложения стали неотъемлемым инструментом для управления финансами, обеспечивая пользователям удобный и быстрый доступ к широкому спектру банковских услуг. Стремительное увеличение объема мобильных операций и повсеместное проникновение смартфонов в повседневную жизнь сделали их ключевым каналом взаимодействия между банками и клиентами. Однако этот прогресс сопровождается беспрецедентным ростом и усложнением мошеннических действий в данной сфере. Злоумышленники активно используют все более изощренные методы, такие как фишинг, социальная инженерия, распространение вредоносных приложений и даже применение DeepFake-технологий, превращая мобильные платформы в основную цель для атак [1-3].

Финансовые институты и регуляторные органы предпринимают значительные усилия для противодействия этим угрозам. Современные российские банки, такие как СберБанк, ВТБ и Т-Банк, активно внедряют комплексные антифрод-системы, сочетающие алгоритмы машинного обучения, поведенческий анализ и многофакторную аутентификацию [7-10]. Центральный банк РФ также усиливает контроль посредством таких инициатив, как платформа «Цифровой след» [7]. Тем не менее, несмотря на демонстрируемую эффективность этих сложных систем защиты, анализ их реализации в мобильных приложениях выявляет существенные ограничения именно в аспекте взаимодействия с конечным пользователем. Работа антифрод-систем зачастую носит реактивный характер с точки зрения пользователя (реагирование постфактум на уже инициированную операцию), а механизмы уведомления и информирования о риске внутри самого мобильного приложения не всегда оптимальны. Часто они сводятся к процедурным запросам на верификацию без ясного объяснения причин, что может вызывать раздражение и непонимание у пользователей [13]. Это подчеркивает актуальность поиска новых подходов, смещающих фокус на улучшение пользовательского опыта (UX) и проактивное вовлечение клиента в процесс обеспечения безопасности своих финансов.

Цель ВКР – разработка Android-прототипа мобильного приложения для улучшенного информирования пользователя о рисках мошенничества. Приложение не просто блокирует, а проактивно и наглядно сообщает об угрозах до совершения транзакции. Используется внешнее API антифрода (Retrofit, KServe V2) и локальная SQLite БД. Акцент сделан на интуитивном UI с визуализацией риска и четкими уведомлениями, что повышает осведомленность и дает пользователю осознанный контроль над безопасностью финансов.

Для достижения поставленной цели в работе решаются следующие задачи:

* Провести анализ предметной области – выявления и предотвращения мошеннических транзакций в мобильном банкинге.
* Исследовать существующие аналоги антифрод-систем и, что особенно важно, подходы к уведомлению и информированию пользователей в мобильных приложениях ведущих российских банков.
* Обосновать актуальность разработки прототипа с фокусом на улучшение пользовательского опыта и проактивное информирование о рисках.
* Спроектировать архитектуру прототипа мобильного приложения, включая моделирование ключевых бизнес-процессов взаимодействия пользователя с интерфейсом и системой оценки риска.
* Разработать компоненты прототипа: клиентскую часть (мобильное приложение) для Android на Kotlin, реализовать взаимодействие с внешним Anti-Fraud API с использованием Retrofit, и настроить работу с локальной базой данных SQLite.
* Провести тестирование разработанного прототипа и продемонстрировать его работу по симуляции анализа транзакций и улучшенному информированию пользователя о рисках.

Объектом исследования является процесс взаимодействия пользователя с интерфейсом прототипа мобильного банковского приложения в контексте получения и интерпретации информации о потенциально мошеннических транзакциях.

Предметом исследования выступают методы и средства проектирования и разработки прототипа мобильного приложения (включая выбор архитектурных решений, технологий Kotlin, SQLite, Retrofit, Koin и подходов к дизайну UI/UX), направленного на улучшение информированности пользователя о мошеннических рисках.

Работа состоит из введения, трех глав, заключения и списка использованных источников. В первой главе проводится аналитическое исследование предметной области, аналогов (с акцентом на UX мобильных приложений) и обосновывается актуальность разработки. Во второй главе описывается процесс проектирования прототипа мобильного приложения: моделирование бизнес-процессов, определение его структуры, выбор средств разработки (Kotlin, SQLite, Retrofit, Koin) и формулируются требования к прототипу. Третья глава посвящена описанию этапов разработки самого прототипа: созданию пользовательских интерфейсов, реализации взаимодействия с внешним API антифрод-системы и локальной базой данных. В четвертой главе представлены методология и результаты тестирования разработанного прототипа. В заключении подводятся итоги проделанной работы.

## ****Аналитическое исследование****

### Описание предметной области

Предметная область данного исследования охватывает сложный и динамичный процесс выявления и предотвращения мошеннических транзакций, совершаемых посредством мобильных банковских приложений. Актуальность этой тематики обусловлена двумя взаимосвязанными тенденциями: повсеместным распространением цифровых финансовых услуг и, как следствие, стремительным ростом объема мобильных банковских операций, что, в свою очередь, привлекает повышенное внимание злоумышленников. Мобильные устройства из простого средства коммуникации превратились в полноценные финансовые терминалы, концентрирующие доступ к счетам, платежным инструментам и персональным данным, что делает их приоритетной мишенью для киберпреступников.

Современная экосистема мобильного банкинга характеризуется значительными изменениями в характере, масштабе и изощренности мошеннических схем. Если раньше основной фокус атак приходился на компрометацию данных банковских карт или взлом онлайн-банкинга через стационарные компьютеры, то сегодня вектор сместился именно на мобильные платформы. Динамичное развитие мобильных технологий и глубокая интеграция банковских сервисов в смартфоны создали благоприятную почву для новых векторов атак. Необходимо подробно рассмотреть как сами характерные черты современных угроз, так и многоуровневые стратегии их преодоления, применяемые финансовыми институтами.

Одной из ключевых характеристик текущего этапа является активизация кибермошенников, использующих широкий арсенал приемов для несанкционированного доступа к средствам и данным клиентов банков. Наиболее часто встречающимися и действенными методами остаются фишинг и социальная инженерия. Статистика подтверждает, что человеческий фактор остается одним из самых уязвимых звеньев в цепочке безопасности: практика показывает, что около 45% успешных мошеннических атак реализуются именно с использованием этих техник [1]. Суть фишинга заключается в создании поддельных веб-сайтов, мобильных приложений или рассылке электронных писем и SMS-сообщений (smishing), которые максимально точно имитируют легитимные ресурсы и коммуникации банка. Цель – обманным путем заставить пользователя ввести свои аутентификационные данные (логин, пароль, одноразовые коды), реквизиты карты или другую конфиденциальную информацию. Социальная инженерия идет дальше, используя психологические манипуляции – создание ощущения срочности, страха, авторитета или доверия – для того, чтобы жертва сама совершила необходимые злоумышленнику действия (например, перевела деньги на "безопасный счет" или сообщила код из SMS).

Дополнительную и возрастающую опасность представляют вредоносные приложения (malware), специально разработанные для атак на пользователей мобильного банкинга. Они могут распространяться через неофициальные магазины приложений, фишинговые ссылки или даже маскироваться под обновления легитимных программ. Такие приложения, часто выглядящие как копии оригинальных банковских продуктов, после установки на устройство могут выполнять широкий спектр вредоносных действий: перехватывать SMS-сообщения с кодами подтверждения операций, записывать нажатия клавиш (keylogging) для кражи паролей, отображать фальшивые экраны входа поверх настоящего банковского приложения (overlay attacks) или получать несанкционированный доступ к данным, хранящимся на устройстве [2]. Борьба с мобильным вредоносным ПО осложняется фрагментацией экосистемы Android и возможностью установки приложений из непроверенных источников.

В последние годы арсенал мошенников пополнился техниками, основанными на глубоких нейронных сетях (DeepFake-технологии). Эти методы искусственного интеллекта позволяют с высокой степенью реалистичности генерировать или изменять аудио- и видеоконтент, например, подделывать голос сотрудника банка или даже создавать видеозвонок от имени знакомого человека, просящего о финансовой помощи. Использование DeepFake в сочетании с социальной инженерией (например, через голосовой фишинг – vishing) представляет собой серьезную угрозу, так как подрывает доверие к традиционным методам верификации личности и делает обман еще более убедительным [3].

Противодействие такому многообразию мошеннических действий требует комплексного подхода, сочетающего мероприятия организационного и технологического характера. Организационные меры направлены прежде всего на повышение уровня цифровой грамотности и информированности клиентов о существующих угрозах и методах защиты. Многие крупные российские банки регулярно проводят информационные кампании, публикуют обучающие материалы, рассылают предупреждения о новых видах мошенничества, стремясь сформировать у пользователей навыки безопасного поведения в цифровой среде (например, не переходить по подозрительным ссылкам, не сообщать коды из SMS, использовать сложные пароли).

Однако одних организационных мер недостаточно. Ключевую роль в защите играют технологические решения, интегрированные непосредственно в банковские системы и мобильные приложения. Сюда относятся как базовые средства защиты (например, использование шифрования, двухфакторной аутентификации), так и более продвинутые системы. Важнейшую роль играют специализированные антивирусные программы (хотя их эффективность на мобильных устройствах может быть ограничена) и, что более значимо, системы анализа транзакций и поведенческого анализа. Последние, часто работающие на основе алгоритмов машинного обучения (ML), способны в режиме реального времени анализировать огромные массивы данных и выявлять аномальные или нетипичные паттерны поведения пользователя, которые могут свидетельствовать о мошенничестве. Анализируются десятки и сотни параметров: характеристики самой транзакции (сумма, получатель, время, геолокация), история операций клиента, параметры используемого устройства (модель, ОС, IP-адрес, наличие root-доступа), особенности взаимодействия пользователя с приложением (скорость навигации, паттерны ввода данных) и многое другое. Системы ML обучаются на исторических данных, постоянно адаптируясь к новым схемам мошенничества, что позволяет выявлять даже ранее неизвестные атаки [4]. Они особенно эффективны в обнаружении таких сценариев, как несанкционированный доступ к аккаунту, переводы на счета, связанные с мошеннической деятельностью, или массовые фишинговые атаки.

Регуляторные органы, в частности Центральный банк России, также играют важную роль, устанавливая требования и стандарты безопасности для финансовых организаций в сфере удаленного банковского обслуживания. ЦБ РФ выпустил серию рекомендаций и нормативных актов, предписывающих банкам внедрять современные антифрод-системы, проводить их регулярный аудит, использовать усиленную аутентификацию (включая биометрические методы для операций с высоким уровнем риска), а также соблюдать требования федерального закона №115-ФЗ "О противодействии легализации (отмыванию) доходов, полученных преступным путем, и финансированию терроризма", который также затрагивает аспекты идентификации клиентов и мониторинга подозрительных операций [5, 6].

Несмотря на значительные усилия, предпринимаемые банками и регуляторами, угроза мобильного мошенничества не только сохраняется, но и продолжает эволюционировать. Высокий уровень технической оснащенности злоумышленников, их способность быстро адаптироваться к новым мерам защиты, глобальный характер киберпреступности и постоянное развитие цифровых технологий способствуют сохранению высокого уровня риска. Важно понимать, что не существует единого универсального решения; эффективная защита – это непрерывный процесс адаптации и совершенствования. Тем не менее, своевременное обнаружение подозрительной активности и принятие превентивных мер (например, временная блокировка операции с запросом дополнительного подтверждения у клиента) позволяют предотвратить значительную часть финансовых потерь и защитить клиентов.

Таким образом, глубокое понимание многогранной природы мобильного мошенничества, включая векторы атак, используемые технологии и психологические аспекты, а также знание существующих методов и инструментов противодействия, является критически важным для разработки эффективных систем защиты. Дальнейшее изучение этой сложной предметной области, включая анализ ограничений существующих подходов и поиск новых решений (например, в области улучшения взаимодействия с пользователем при обнаружении риска, что является целью данной ВКР), позволит предложить практические рекомендации по совершенствованию механизмов защиты клиентов банков и повышению общей безопасности и доверия к мобильным финансовым сервисам.

### Анализ аналогов

Современные российские банки, осознавая масштабы угрозы мобильного мошенничества, активно инвестируют в разработку и внедрение комплексных систем обнаружения и предотвращения фрода. Эти системы, как правило, представляют собой многоуровневые решения, сочетающие алгоритмы машинного обучения (ML) для выявления аномалий, поведенческий анализ для понимания типичных действий клиента и многофакторную аутентификацию (MFA) для подтверждения легитимности операций. В рамках данного исследования был проведен углубленный анализ того, как эти сложные бэкенд-системы проявляются в функционале и пользовательском интерфейсе мобильных приложений ведущих банков России – СберБанка (СберБанк Онлайн), ВТБ (ВТБ Онлайн) и Т-Банка (Тинькофф). Также были учтены общие регуляторные инициативы, формирующие ландшафт безопасности.

На макроуровне важную роль играет инициатива Центрального банка Российской Федерации – платформа мониторинга транзакций «Цифровой след», запущенная в 2024 году. Она агрегирует обезличенные данные от 320 кредитных организаций, используя графовые алгоритмы для выявления сложных мошеннических схем, таких как использование подставных лиц для массовой регистрации карт (дропперство). Обнаружение 8,4 тыс. таких случаев с оборотом 3,2 млрд рублей [7] демонстрирует потенциал централизованного подхода. Банки получают от платформы сигналы, которые они могут использовать для блокировки подозрительных операций или для обогащения данных собственных антифрод-систем. Однако эта система работает на межбанковском уровне и напрямую не влияет на интерфейс взаимодействия пользователя с конкретным банковским приложением, хотя и повышает общую эффективность борьбы с фродом.

СберБанк Онлайн является одним из самых популярных банковских приложений в России, оно предлагает широкий спектр услуг – от переводов и платежей до инвестиций и страхования. В части антифрода, в приложении реализован интерфейс для взаимодействия с бэкенд-модулем AI Fraud Detection. Эта система анализирует более 150 параметров транзакции в реальном времени, включая геолокацию устройства, скорость набора данных, историю операций и характеристики получателя [8]. С точки зрения пользователя, при обнаружении высокого риска (например, перевод крупной суммы новому получателю из нетипичного местоположения) происходит следующее:

* Приостановка операции: Транзакция не выполняется мгновенно, а ставится на паузу.
* Push-уведомление: Пользователь получает уведомление о необходимости дополнительного подтверждения.
* Верификация: В качестве метода подтверждения может использоваться голосовая биометрия или другие методы MFA.

Недостатки с точки зрения UX: Хотя система эффективна в блокировке, само уведомление часто носит процедурный характер ("Подтвердите операцию голосом"), не всегда предоставляя пользователю ясную информацию о причине подозрения или уровне риска. Пользователь может не до конца понимать, почему требуется дополнительное действие, что может вызывать замешательство или восприниматься как неудобство. Визуализация самого риска в интерфейсе истории или при совершении операции минимальна или отсутствует.

Мобильное приложение ВТБ также предоставляет полный набор банковских сервисов. Его антифрод-система VTB Anti-Fraud активно использует данные из глобальной базы паттернов мошенничества FICO Falcon для оценки риска [9]. Взаимодействие с пользователем при высоком риске (например, при попытке смены привязанного номера телефона или при крупном переводе на незнакомый счет) часто реализуется через автоматизированный звонок от робота. Этот робот, имитируя голос сотрудника службы безопасности, запрашивает подтверждение операции.

* Преимущества: Прямой звонок может быть эффективен для привлечения внимания.
* Недостатки с точки зрения UX: Звонки с незнакомых номеров часто игнорируются пользователями или воспринимаются как спам (или даже как попытка мошенничества со стороны "лже-сотрудников"). Роботизированный голос не всегда вызывает доверие. Важно, что этот механизм выносит взаимодействие за пределы привычного интерфейса приложения, что может быть менее удобно и интуитивно. Опять же, визуализация риска и проактивное информирование внутри самого приложения до момента звонка выражены слабо.

Приложение Т-Банка известно своим фокусом на цифровые сервисы и удобный интерфейс, интегрирующий банковские и лайфстайл-услуги. Антифрод-система Т-Банка использует адаптивную аутентификацию, гибко подстраивая требования к верификации в зависимости от расчетного уровня риска транзакции. Для рутинных операций с низким риском (например, перевод небольшой суммы до 10 тыс. рублей знакомому получателю) может быть достаточно SMS-кода или даже его отсутствия. Для операций с повышенным риском (крупные суммы, новые получатели, нетипичные действия) система может запросить подтверждение через Face ID, Touch ID или пин-код [10]. Т-Банк также применяет «динамическую биометрию», запрашивая подтверждение отпечатком пальца только при отклонении параметров транзакции от типового профиля клиента [12]. Это позволяет снизить количество ложных срабатываний (на 27% в 2023 году [10]) и сократить среднее время подтверждения операции (до 4,7 секунды [12]).

* Преимущества: Высокий уровень удобства для пользователя при сохранении безопасности за счет гибкости.
* Недостатки с точки зрения UX: Несмотря на продвинутую систему аутентификации, коммуникация самого факта риска пользователю остается неявной. Пользователь может не понимать, почему в одном случае потребовалось дополнительное подтверждение, а в другом нет. Система эффективно реагирует на риск изменением процедуры, но не всегда проактивно информирует пользователя о сути этого риска. Визуальное представление уровня опасности транзакции в интерфейсе приложения не является стандартной практикой.

Анализ показывает, что ведущие банки активно работают над балансом между безопасностью и удобством использования (usability) [12]. Они внедряют сложные бэкенд-системы и адаптируют методы аутентификации. Учитываются и регуляторные требования, например, связанные с директивой PSD3 или необходимостью проверки ИНН контрагента через API ФНС для повышения прозрачности [11].

Однако, несмотря на технологические достижения, сохраняется проблема пользовательского восприятия антифрод-мер. Опрос НИУ ВШЭ подтверждает, что 68% пользователей испытывают раздражение от частых или не всегда понятных запросов на верификацию [13]. Это указывает на ключевой недостаток существующих подходов: недостаточный фокус на качестве коммуникации риска с пользователем внутри мобильного приложения.

Текущие решения отлично справляются с задачей блокировки или дополнительной верификации подозрительных операций, но они не всегда:

* Проактивно и понятно информируют пользователя о причинах и уровне риска до окончательного подтверждения операции.
* Предоставляют наглядную визуализацию риска непосредственно в интерфейсе совершения транзакции или в истории операций.
* Способствуют повышению осведомленности пользователя о потенциальных угрозах через интерактивные элементы в контексте его реальных действий.

Именно эти аспекты – улучшение пользовательского опыта (UX) через проактивное, визуально понятное и контекстное информирование о рисках мошенничества – являются основной целью и новизной решения, разрабатываемого в рамках данной выпускной квалификационной работы. Предлагаемый прототип нацелен не на замену существующих антифрод-движков, а на демонстрацию более эффективного интерфейса взаимодействия пользователя с результатами их работы, что может снизить фрустрацию, повысить бдительность и дать пользователю больше контроля и понимания в вопросах безопасности своих финансов.

### Актуальность разработки

Несмотря на то, что существующие антифрод-системы, внедренные ведущими российскими банками, демонстрируют определенную эффективность, снижая, по оценкам Центрального банка, уровень успешного мошенничества на 18–34% [7], их функционирование сопряжено с рядом существенных ограничений. Как было показано в анализе аналогов, многие системы все еще в значительной степени зависят от реактивных методов – то есть реагируют на уже инициированную подозрительную операцию блокировкой или запросом дополнительной верификации, не всегда предоставляя пользователю упреждающую информацию о самом риске. Зачастую наблюдается недостаточная персонализация проверок, приводящая к ложным срабатываниям и раздражению клиентов [13], а также слабая интеграция с внешними источниками данных, например, с Единой биометрической системой (ЕБС) или государственными сервисами, что могло бы повысить точность оценки риска. Эти факторы в совокупности подчеркивают острую необходимость в разработке решений нового поколения, способных не просто блокировать фрод, а действовать предиктивно и проактивно, вовлекая клиента в процесс обеспечения безопасности через понятное и своевременное информирование.

Актуальность разработки прототипа мобильного приложения, фокусирующегося на улучшении пользовательского опыта взаимодействия с рисками мошеннических транзакций, определяется не только технологическими ограничениями существующих систем, но и целым комплексом социально-экономических и поведенческих факторов, сложившихся в России в последние годы. Именно сейчас создается уникальное окно возможностей и потребностей для внедрения подобных решений.

Во-первых, масштабы проблемы мобильного мошенничества достигли критического уровня. Статистика Центрального Банка РФ неумолима: в 2023 году объем несанкционированных операций, совершенных с использованием электронных средств платежа, вырос на 11.5% по сравнению с 2022 годом, достигнув 15,8 млрд рублей [18]. При этом подавляющее большинство хищений (около 877 тысяч операций на сумму 14,2 млрд рублей) было совершено с использованием методов социальной инженерии, когда граждане под психологическим давлением сами переводят деньги мошенникам или раскрывают свои данные [18]. Количество фишинговых атак, нацеленных на пользователей мобильных устройств, также продолжает расти, как и число случаев заражения вредоносным ПО. Эти цифры наглядно демонстрируют, что технологических барьеров самих по себе недостаточно – необходимо активно работать с "человеческим фактором", повышая осведомленность и бдительность пользователей именно в момент совершения операции.

Во-вторых, изменились ожидания и поведение самих пользователей. Мобильный банкинг стал для россиян повседневным инструментом: по данным НАФИ, более 80% экономически активного населения регулярно пользуются банковскими приложениями для управления своими финансами [19]. Это формирует высокие требования не только к функциональности, но и к удобству (usability) и прозрачности предоставляемых сервисов. Пользователи ожидают интуитивно понятного интерфейса и мгновенной обратной связи. В то же время, как уже упоминалось [13] и подтверждается другими опросами, существующие механизмы антифрода часто вызывают негативную реакцию. Исследование ВЦИОМ показывает, что каждый третий россиянин (34%) сталкивался с блокировкой операций банком, и хотя большинство (72%) считают такие меры оправданными, значительная часть испытывает неудобства, особенно если причины блокировки или запроса верификации остаются неясными [20]. Это создает разрыв между необходимостью обеспечения безопасности и стремлением к бесшовному пользовательскому опыту. Предлагаемый в данной работе подход направлен на преодоление этого разрыва путем интеграции информации о риске непосредственно в пользовательский сценарий, делая безопасность более понятной и менее навязчивой.

В-третьих, наблюдаются важные технологические тренды, которые делают реализацию предлагаемого решения своевременной и перспективной. Развитие алгоритмов машинного обучения (ML) позволяет не только с высокой точностью выявлять подозрительные паттерны, но и оценивать степень риска количественно (например, в виде вероятности мошенничества). Эти данные, которые сейчас используются в основном внутри антифрод-систем для принятия решения о блокировке, могут быть транслированы пользователю в доступной и визуально понятной форме. Современные возможности мобильных платформ (быстрые push-уведомления, гибкие интерфейсы, графика) позволяют реализовать это эффективно. Кроме того, растет понимание важности объяснимого ИИ (Explainable AI, XAI) [21], то есть способности систем не просто выдавать результат, но и объяснять его. Применительно к антифроду, это означает потенциальную возможность не просто показать "высокий риск", но и дать краткое пояснение (например, "нетипичная геолокация" или "перевод новому получателю"), что еще больше повысит доверие и понимание со стороны пользователя.

Именно поэтому целью данной работы была выбрана разработка прототипа мобильного приложения, демонстрирующего, как текущие банковские приложения могли бы кардинально улучшить взаимодействие пользователей с рисками мошеннических транзакций. Вместо того чтобы просто реагировать на угрозу, предлагаемое решение проактивно информирует пользователя о потенциальной опасности до завершения операции, используя понятную визуализацию и четкие сообщения. Приложение, симулирующее использование ML-модели для анализа транзакций, направлено на устранение ключевых недостатков существующих систем в части коммуникации с клиентом. Основной задачей является создание интуитивно понятного интерфейса, который не перегружает пользователя техническими деталями, но дает ему достаточно информации для осознанного принятия решения и формирует более ответственное отношение к финансовой безопасности.

Важность разработки такого мобильного приложения заключается не только в потенциальном повышении уровня фактической безопасности (пользователь, предупрежденный о риске, с большей вероятностью откажется от сомнительной операции), но и в существенном улучшении пользовательского опыта (UX). Современные клиенты, особенно молодое поколение, ожидают от банков не только надежности, но и прозрачности, удобства и персонализации услуг. Мобильное приложение, которое предоставляет пользователям возможность визуально контролировать риски своих транзакций, получать мгновенные, но ненавязчивые и понятные уведомления о подозрительных действиях, а также включает элементы интерактивного обучения безопасному поведению в контексте реальных операций, способно значительно повысить доверие к финансовому учреждению и лояльность клиентов. Более того, успешная реализация и демонстрация данного подхода может стать основой для дальнейших исследований и практического внедрения подобных интерфейсных решений в реальные банковские продукты, способствуя развитию финансовых технологий (FinTech) в направлении большей клиентоориентированности и безопасности.

Таким образом, разработка мобильного приложения, фокусирующегося на улучшенном информировании пользователя о мошеннических транзакциях, представляет собой своевременный и крайне актуальный шаг. Это решение напрямую отвечает на острейшие вызовы, стоящие перед финансовой отраслью России – беспрецедентный рост кибермошенничества и необходимость повышения цифровой грамотности населения. Оно учитывает текущие запросы пользователей на прозрачность и удобство, устраняя выявленные недостатки существующих систем в плане коммуникации риска. Наконец, оно использует современные технологические возможности для создания более эффективного и интуитивного интерфейса безопасности. Реализация данного проекта не только решает конкретную прикладную задачу в рамках ВКР, но и вносит вклад в поиск путей повышения общей защищенности граждан в цифровой финансовой среде, открывая новые горизонты для дальнейших исследований и разработок в области борьбы с мошенничеством.

## ****Проектирование информационной системы****

### Моделирование бизнес-процессов ИС

Для построения диаграммы вариантов использования, описывающей процесс взаимодействия пользователя с мобильным приложением, на диаграмму был помещен актор — пользователь [14]. В результате, на диаграмме были добавлены варианты использования, такие как «Совершение транзакции», «Создание карты» и «Блокировка карты». Эти варианты использования были соединены с актором связью «Association», что позволяет отразить взаимодействие пользователя с системой, рисунок 2.1.

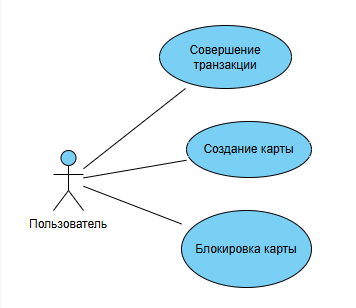


Рисунок 2.1 — Диаграмма вариантов использования

Далее были добавлены сценарии, относящиеся к мобильному приложению, включая «Отправить данные о транзакции», «Получить данные о проверке транзакции», «Отобразить пользователю результаты проверки транзакции», «Отобразить пользователю push-уведомление» и «Послать запрос на блокировку карты». Эти сценарии также были соединены с мобильным приложением связью «Association», что демонстрирует их функциональную связь, рисунок 2.2 [15].

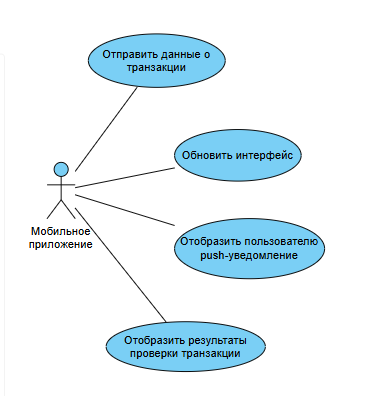


Рисунок 2.2 — Добавление сценариев для мобильного приложения

Также были добавлены сценарии, относящиеся к Антифрод-системе, включая «Получить данные транзакции», «Проверить транзакцию», «Отобразить пользователю результаты проверки транзакции» и «Отослать данные о проверки». Эти сценарии также были соединены с актором связью, рисунок 2.3.

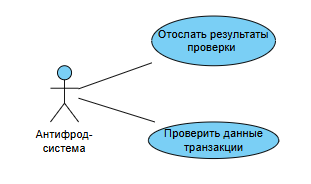


Рисунок 2.3 — Добавление сценариев для Антифрод-системы

Для построения диаграммы последовательности, описывающей процессы взаимодействия пользователя с системой при обработке транзакций и других операциях, на диаграмму был помещен актор — «Пользователь». Также были добавлены линии жизни объектов, представляющих ключевые компоненты системы: «Мобильное приложение», «Серверное приложение» и «Антифрод-система», рисунок 2.4.

Далее были добавлены последовательные сообщения между актором и объектами для иллюстрации основных сценариев взаимодействия, представленных на диаграмме:

* Сценарий создания карты: Пользователь инициирует создание карты (сообщение 1), мобильное приложение обновляет данные пользователя(1.1), интерфейс пользователя обновляется (1.2).
* Сценарий совершения транзакции и ее проверки: Пользователь инициирует транзакцию (2), мобильное приложение запрашивает проверку у антифрод-системы (2.1), антифрод-система проверяет транзакцию (2.2), антифрод-система возвращает результат проверки (2.3), мобильное приложение отображает результаты проверки и push-уведомление пользователю (2.4).
* Сценарий блокировки карты: Пользователь инициирует блокировку (3), мобильное приложение обновляет данные пользователя(3.1), интерфейс пользователя обновляется (3.2).

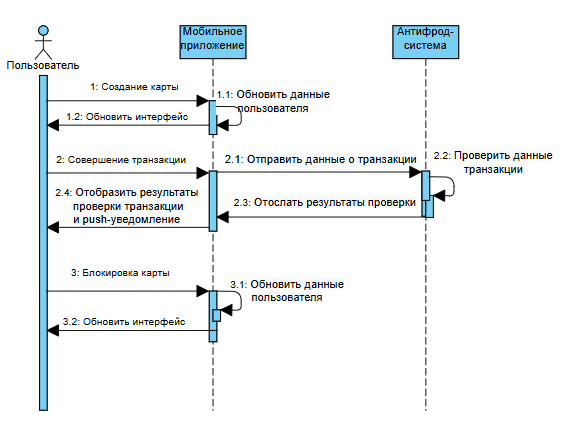


Рисунок 2.5 — Добавление последовательных сообщений

### Структура ИС и ее средства разработки

Для достижения поставленных в рамках данной выпускной квалификационной работы целей, связанных с демонстрацией улучшенного пользовательского опыта информирования о потенциально мошеннических транзакциях, была спроектирована упрощенная архитектура системы. Основной акцент сделан на разработке функционального прототипа мобильного приложения, способного наглядно представить предлагаемый механизм взаимодействия с пользователем, без необходимости развертывания сложной, масштабируемой серверной инфраструктуры.

В рамках выбранного подхода, система состоит из мобильного приложения, локальной базы данных и внешнего сервиса моделирующего антифрод систему.

Клиентский модуль (Мобильное приложение): Разрабатываемое нативное мобильное приложение для платформы Android. Оно является центральным элементом прототипа и предоставляет пользователю графический интерфейс для:

* Инициирования банковских транзакций (в рамках прототипа – симуляция ввода данных).
* Просмотра истории операций, хранящейся локально на устройстве.
* Взаимодействия с внешним API сервиса антифрода для анализа транзакций.
* Получения и отображения результатов проверки транзакций непосредственно от API антифрода, с четкой визуализацией уровня риска.
* Локального хранения необходимых данных (история операций, пользовательские настройки).

Модуль анализа и определения мошенничества (Anti-Fraud Engine API): Предполагается наличие внешнего, стороннего или предоставленного для целей демонстрации, специализированного сервиса (API), реализующего логику выявления мошеннических транзакций. Мобильное приложение будет напрямую обращаться к этому API для получения оценки риска по каждой симулируемой транзакции.

Вместо централизованной серверной СУБД, для хранения данных непосредственно на устройстве пользователя будет использоваться встраиваемая база данных SQLite. Это решение обусловлено целями создания именно демонстрационного прототипа:

* Простота реализации: SQLite не требует настройки и администрирования отдельного сервера баз данных, что значительно упрощает разработку и тестирование прототипа.
* Достаточность для демонстрации: Для хранения истории операций одного пользователя и пользовательских настроек в рамках прототипа возможности SQLite являются более чем достаточными.
* Фокус на UX: Данный подход позволяет сосредоточить усилия на разработке интерфейса и логики взаимодействия с пользователем в мобильном приложении, что является ключевой задачей данной ВКР, а не на построении отказоустойчивой системы хранения данных.
* Автономность (частичная): Возможность просмотра истории операций без активного сетевого подключения улучшает пользовательский опыт в рамках демонстрации.

В базе данных SQLite будут храниться:

* Условный профиль пользователя (минимальные данные для персонализации интерфейса в прототипе).
* История симулированных транзакций с результатами проверки риска, полученными от Anti-Fraud API.

Для эффективной реализации описанных компонентов мобильного приложения был подобран технологический стек, ориентированный на быструю разработку прототипа для платформы Android и демонстрацию ключевой функциональности.

Основой для разработки мобильного приложения (клиентского модуля) является Kotlin SDK в связке с Android SDK. Kotlin – современный, официально поддерживаемый Google язык для Android-разработки, выбранный за его выразительность, безопасность (null safety) и отличную поддержку асинхронного программирования с помощью корутин, что важно для взаимодействия с сетью и базой данных без блокировки основного потока. Разработка ведется с ориентацией на актуальные версии Android API (например, API Level 34), обеспечивая совместимость с большинством современных устройств (минимальный уровень поддержки, например, API Level 24). Сборка проекта осуществляется с помощью системы Gradle.

Для взаимодействия мобильного приложения с внешним API антифрод-системы будет использоваться библиотека Retrofit. Это стандарт де-факто для сетевого взаимодействия в Android-разработке. Retrofit позволяет декларативно описывать REST API с помощью аннотаций в интерфейсах Kotlin, что делает код для сетевых запросов чистым и легко читаемым. Он легко интегрируется с библиотеками для преобразования JSON (или других форматов) в объекты Kotlin (например, Gson, Moshi), автоматизируя процесс сериализации/десериализации данных.

Для организации структуры кода мобильного приложения, повышения его тестируемости и упрощения управления зависимостями между компонентами (например, предоставление экземпляра Retrofit-клиента, репозитория для работы с базой данных) будет использоваться библиотека Koin. Koin – это легковесный и прагматичный фреймворк для внедрения зависимостей, написанный на Kotlin и для Kotlin. Его API интуитивно понятен и требует меньше шаблонного кода по сравнению с некоторыми альтернативами (например, Dagger/Hilt), что идеально подходит для быстрой разработки прототипа. Он позволяет следовать принципам SOLID, разделяя ответственность между классами и упрощая их замену или тестирование. Koin предоставляет удобные механизмы для управления областью видимости зависимостей, в том числе привязанные к жизненному циклу компонентов Android (Activity, Fragment, ViewModel).

Как уже упоминалось, для хранения данных будет использоваться SQLite, интегрированная в Android SDK. Доступ к базе данных будет осуществляться через стандартные механизмы Android Room Persistence Library или напрямую через SQLiteOpenHelper для максимальной простоты в рамках прототипа. Room предпочтительнее, так как предоставляет абстракцию над SQLite, упрощая работу с БД и проверку запросов на этапе компиляции.

Разработка клиентского Android-приложения на Kotlin будет вестись в Android Studio (актуальная версия, например, Hedgehog | 2023.1.1 или новее). Эта IDE предоставляет все необходимые инструменты: редактор кода с поддержкой Kotlin, визуальный редактор макетов, эмулятор Android, инструменты отладки и профилирования, интеграцию с Gradle, а также инструменты для работы с локальной базой данных SQLite (Database Inspector).

Важно отметить, что в данной архитектуре отсутствует промежуточный серверный компонент (Backend), который традиционно обрабатывал бы запросы от мобильного приложения, взаимодействовал с базой данных и внешними сервисами. Данный слой был исключен исходя из следующих факторов:

* Фокус на демонстрации UX: Основная цель ВКР – продемонстрировать улучшенный пользовательский опыт в мобильном приложении при информировании о фроде. Прямое взаимодействие приложения с API антифрода и локальной БД достаточно для этой демонстрации.
* Упрощение разработки: Разработка, развертывание и поддержка полноценного backend-сервера (например, на ASP.NET Core, как рассматривалось в альтернативных, более сложных архитектурах) значительно увеличивает трудоемкость проекта, что нецелесообразно для создания прототипа.
* Снижение инфраструктурных требований: Исключение backend устраняет необходимость в серверных ресурсах, хостинге и настройке сетевой инфраструктуры для демонстрации.

Выбранная архитектура с прямым взаимодействием мобильного приложения с локальной базой данных SQLite и внешним Anti-Fraud API, а также использование Kotlin, Retrofit и Koin, позволяет сфокусироваться на ключевой задаче данной ВКР – разработке и демонстрации прототипа мобильного приложения с улучшенным информированием пользователя о мошеннических транзакциях. Этот подход значительно упрощает разработку по сравнению с созданием полномасштабной, отказоустойчивой системы с выделенным backend, что полностью соответствует целям и ограничениям выпускной квалификационной работы, ориентированной на демонстрацию конкретного аспекта пользовательского опыта. Использование Koin и Retrofit обеспечивает современный подход к разработке Android-приложений, делая код структурированным, тестируемым и поддерживаемым в рамках прототипа.

### Требования к ИС

На основе анализа предметной области и существующих аналогов сформулированы следующие требования к разрабатываемой информационной системе (ИС) для определения мошеннических транзакций и уведомления о них пользователя.

Система должна предоставлять пользователю возможность симулировать инициирование банковских транзакций через интерфейс мобильного приложения, рисунок 1.

Передача данных о транзакции: Мобильное приложение должно обеспечивать безопасную передачу данных о симулируемой транзакции (сумма, получатель, тип операции и т.д.) на серверный модуль для дальнейшей обработки, рисунки 2 и 4.

Анализ транзакции на мошенничество: Серверный модуль должен взаимодействовать с модулем анализа и определения мошенничества (Anti-Fraud Engine), использующим ML-модель, для оценки риска симулируемой транзакции, рисунок 4.

* Получение результатов проверки: Мобильное приложение должно получать от серверного модуля результат проверки транзакции, включая оценку уровня риска, рисунок 2.
* Проактивное уведомление пользователя: В случае выявления высокого риска мошенничества, система должна до завершения симулируемой транзакции отправлять пользователю push-уведомление о подозрительной операции.
* Визуализация риска: Мобильное приложение должно наглядно отображать пользователю информацию о результатах проверки транзакции, включая уровень риска.
* Просмотр истории транзакций: Система должна предоставлять пользователю доступ к истории симулированных транзакций с указанием их статуса и результатов проверки на мошенничество.

К нефункциональным требованиям системы относится:

* Производительность: Время отклика системы на действия пользователя в мобильном приложении (например, навигация, инициирование транзакции) не должно превышать 1-2 секунд. Время от инициирования симулируемой транзакции до получения пользователем push-уведомления (в случае высокого риска) не должно превышать 5-7 секунд, чтобы обеспечить возможность проактивного вмешательства.
* Надежность: Система должна обеспечивать стабильную работу мобильного приложения и серверной части. Данные о транзакциях и пользователях (в рамках прототипа – анонимизированные/симулированные) должны надежно храниться в базе данных SQLite, обеспечивая целостность и доступность
* Безопасность: Взаимодействие между мобильным приложением и серверным модулем должно осуществляться по защищенному каналу. Должны быть предусмотрены меры по защите API от несанкционированного доступа.
* Удобство использования: Интерфейс мобильного приложения должен быть интуитивно понятным, простым в освоении и использовании для целевой аудитории. Уведомления о риске должны быть сформулированы ясно и однозначно, не вызывая паники у пользователя, но подчеркивая серьезность ситуации. Процесс подтверждения или отклонения подозрительной транзакции должен быть максимально простым и быстрым.

Требования к интерфейсу мобильного приложения включают в себя:

* Графический интерфейс пользователя (GUI): Разработка нативного GUI для платформы Android с использованием Kotlin SDK и современных практик дизайна.
* Отображение транзакций: Четкое и структурированное отображение информации о симулируемых транзакциях (сумма, дата, получатель, статус).
* Визуализация риска: Использование понятных визуальных элементов (цветовая шкала, иконки, прогресс-бары) для индикации уровня мошеннического риска транзакции.
* Элементы управления: Наличие ясно обозначенных кнопок и интерактивных элементов для выполнения действий: инициирования транзакции, подтверждения/отклонения операции, навигации по приложению.
* Уведомления: Использование стандартных механизмов push-уведомлений ОС Android для оперативного информирования пользователя. Текст уведомлений должен быть кратким и информативным.
* Навигация: Логичная и простая структура навигации в приложении, обеспечивающая быстрый доступ ко всем основным функциям (транзакции, история, обучение, настройки).

Учитывая архитектуру, сфокусированную на мобильном клиенте с локальным хранилищем и взаимодействием с внешним API. Требования к программному обеспечению, необходимому для функционирования разработанного прототипа мобильного приложения, а также для его разработки и сборки.

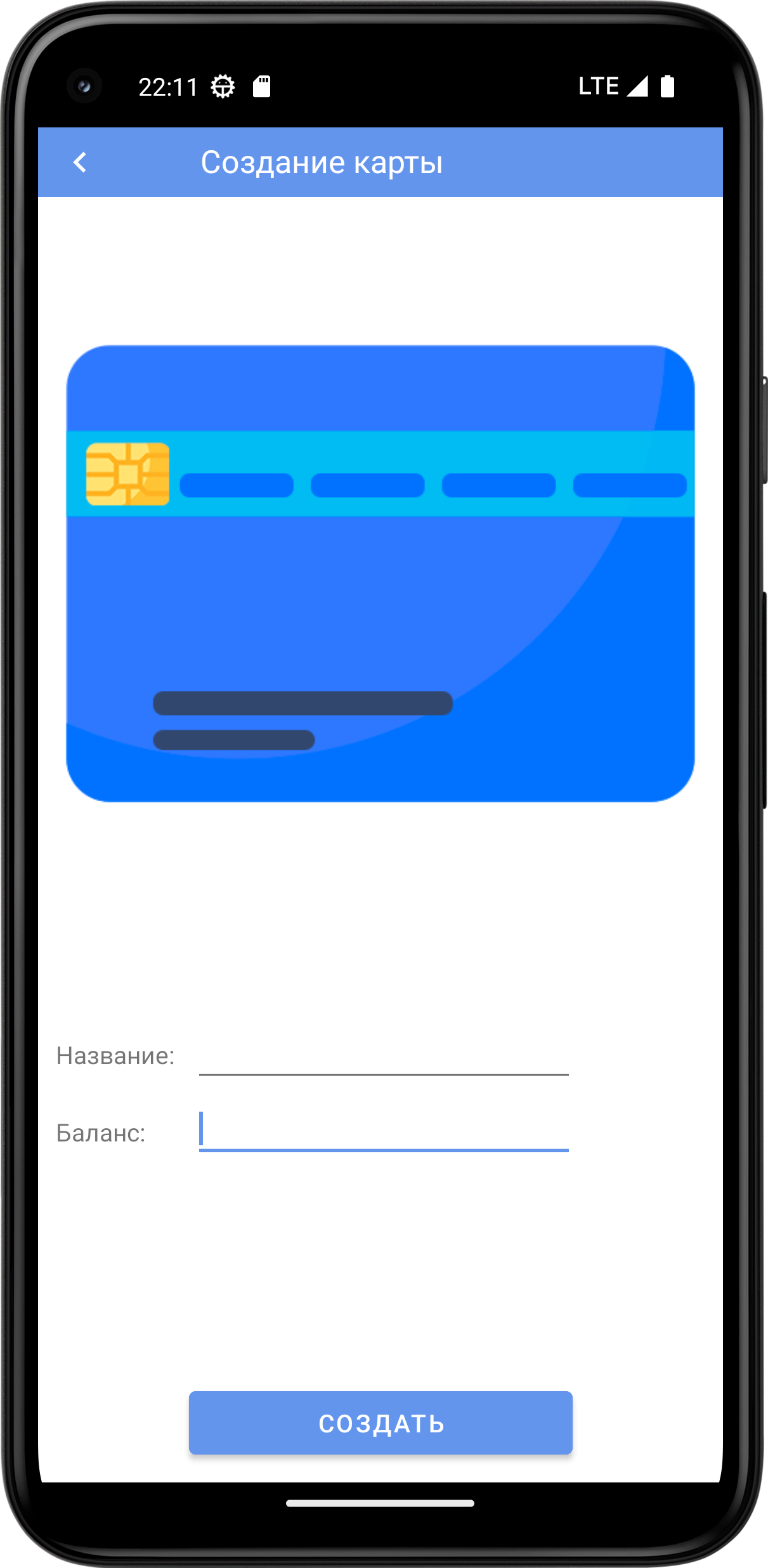
* Операционная система мобильного устройства: Android версии 7.0 (Nougat, API Level 24) или выше. Рекомендуется использовать устройства под управлением Android 11 (API Level 30) или новее для оптимальной производительности и поддержки современных функций. Целевой SDK при разработке – Android 14 (API Level 34).
* Среда исполнения: Наличие установленной среды выполнения Android (Android Runtime — ART).
* Доступ к сети Интернет: Требуется для взаимодействия с внешним API сервиса анализа и определения мошенничества (Anti-Fraud Engine API). Стабильное подключение необходимо для выполнения анализа транзакций в реальном времени в рамках демонстрации.
* Локальное хранилище: Достаточное количество свободного места во внутренней памяти устройства для хранения локальной базы данных SQLite (содержащей историю симулированных операций и пользовательские настройки прототипа). Объем требуемого места незначителен для целей демонстрационного прототипа.

## ****Реализация ИС****

### ****Разработка интерфейсов мобильного приложения****

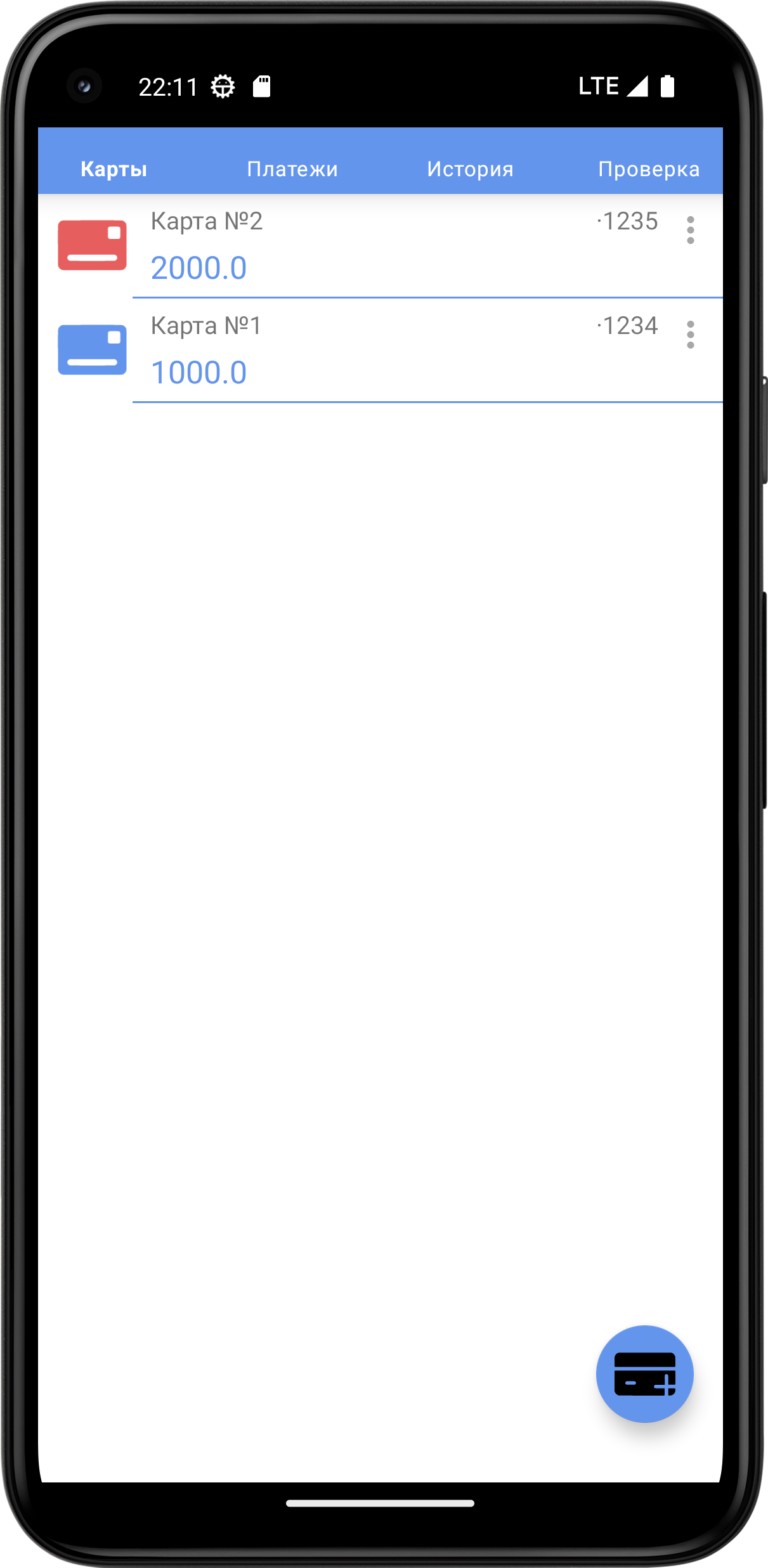
**В данном разделе представлена визуальная демонстрация результатов разработки прототипа мобильного приложения для платформы Android. Эти скриншоты иллюстрируют практическую реализацию концепций, проектных решений и требований, подробно описанных в предыдущих главах работы. Разработанное приложение является воплощением проведенного аналитического исследования и спроектированной архитектуры, наглядно показывая, как теоретические основы и сформулированные требования трансформировались в функциональный пользовательский интерфейс. Основной фокус демонстрации направлен на ключевые аспекты взаимодействия пользователя с системой в контексте предотвращения мошенничества: симуляцию основных банковских операций (добавление карт, переводы, платежи), отображение истории транзакций с визуальной оценкой риска, и, что является центральным элементом данной работы, механизм проактивного, но ненавязчивого уведомления пользователя о потенциально опасных действиях. Последующие изображения детально раскрывают функционал и интерфейс приложения, разработанного с использованием Kotlin для Android, и показывают, как оно решает поставленную задачу улучшения информированности и контроля пользователя над безопасностью своих финансов в мобильном банкинге.**

**На рисунке 3.1 показан экран «Создание карты». Этот интерфейс предназначен для симуляции добавления пользователем своих банковских карт в приложение. Пользователь может задать произвольное название карты и указать начальный баланс. Данная функция необходима для первоначальной настройки прототипа и создания объектов (карт), с которыми будут проводиться симулируемые транзакции.**



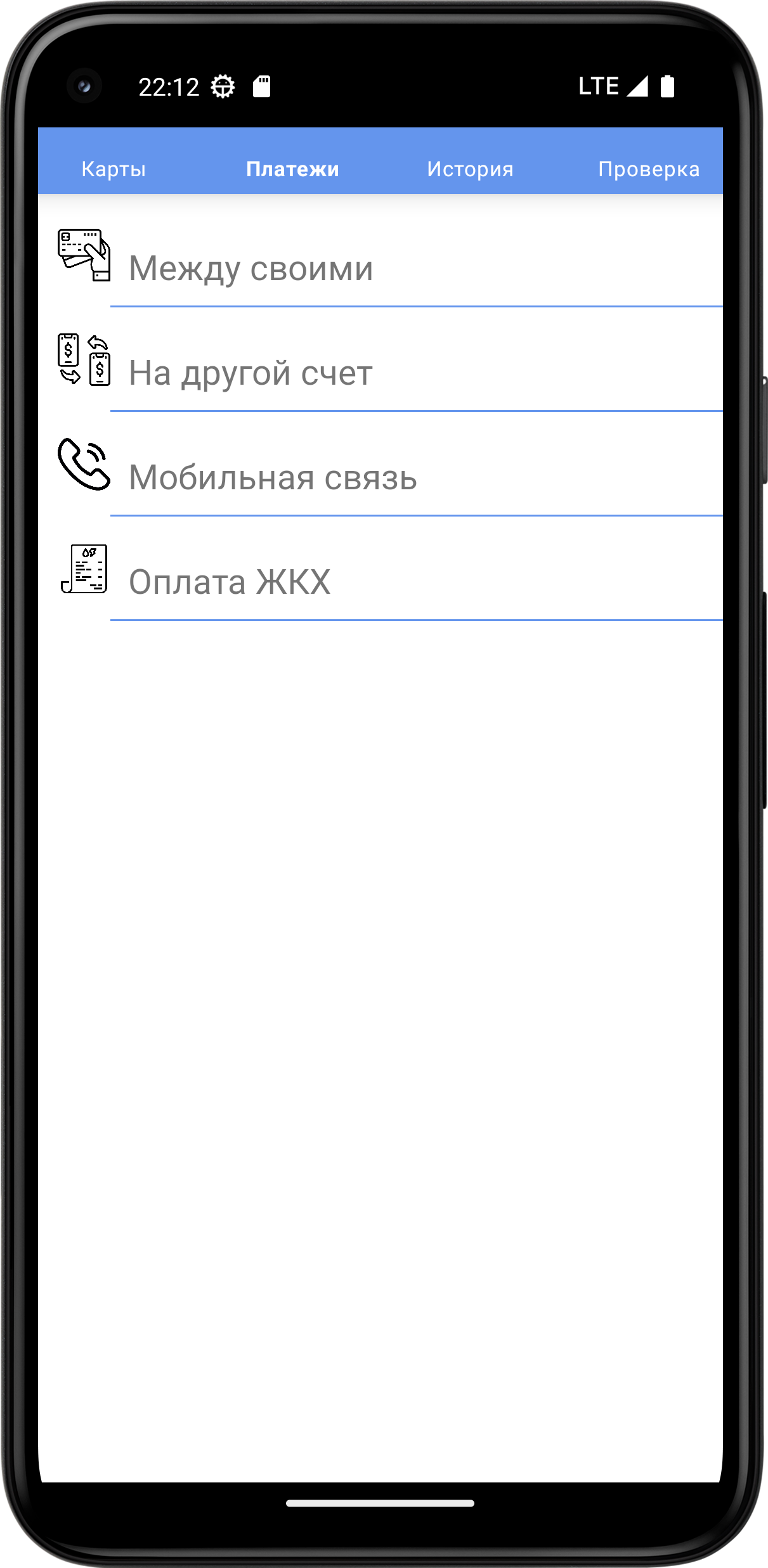
**Рисунок 3.1 — Демонстрация мобильного приложения. Создание карты**

**После добавления карт пользователь видит их список на главном экране вкладки «Карты», как представлено на рисунке 3.2. Здесь отображаются названия карт, их симулированные балансы и последние четыре цифры номера (для имитации реального интерфейса). В правом нижнем углу расположен плавающий элемент управления (Floating Action Button) для быстрого добавления новых карт, то есть для перехода к экрану на рисунке 3.1. Этот экран служит отправной точкой для большинства действий пользователя.**



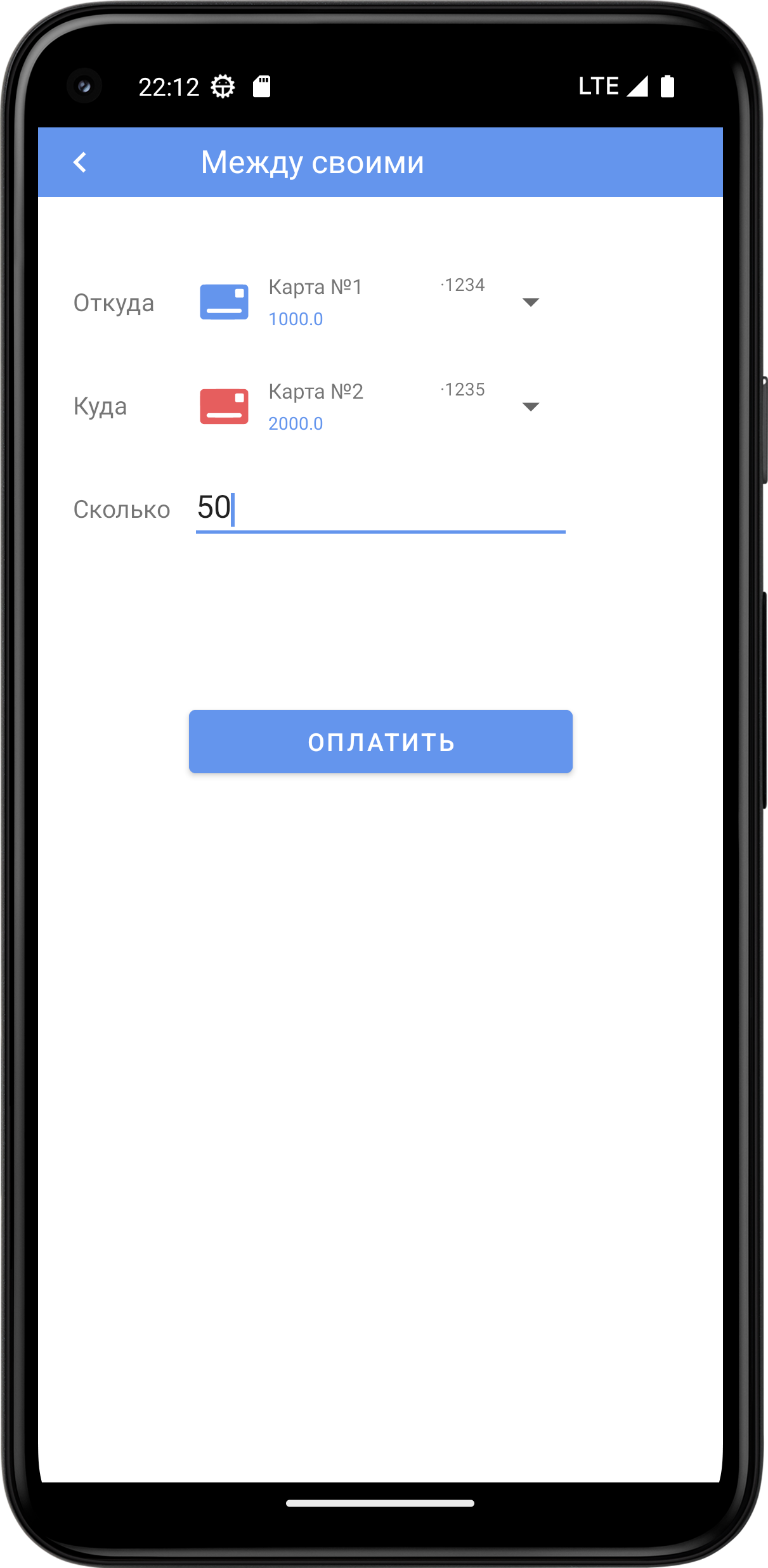
**Рисунок 3.2 — Демонстрация мобильного приложения. Вывод списка карт**

**Перейдя на вкладку «Платежи», рисунок 3.3, пользователь получает доступ к списку доступных типов симулируемых транзакций. В прототипе реализованы основные операции: перевод между своими счетами, перевод на другой счет, оплата мобильной связи и ЖКХ. Выбор одного из этих пунктов инициирует соответствующий сценарий транзакции.**



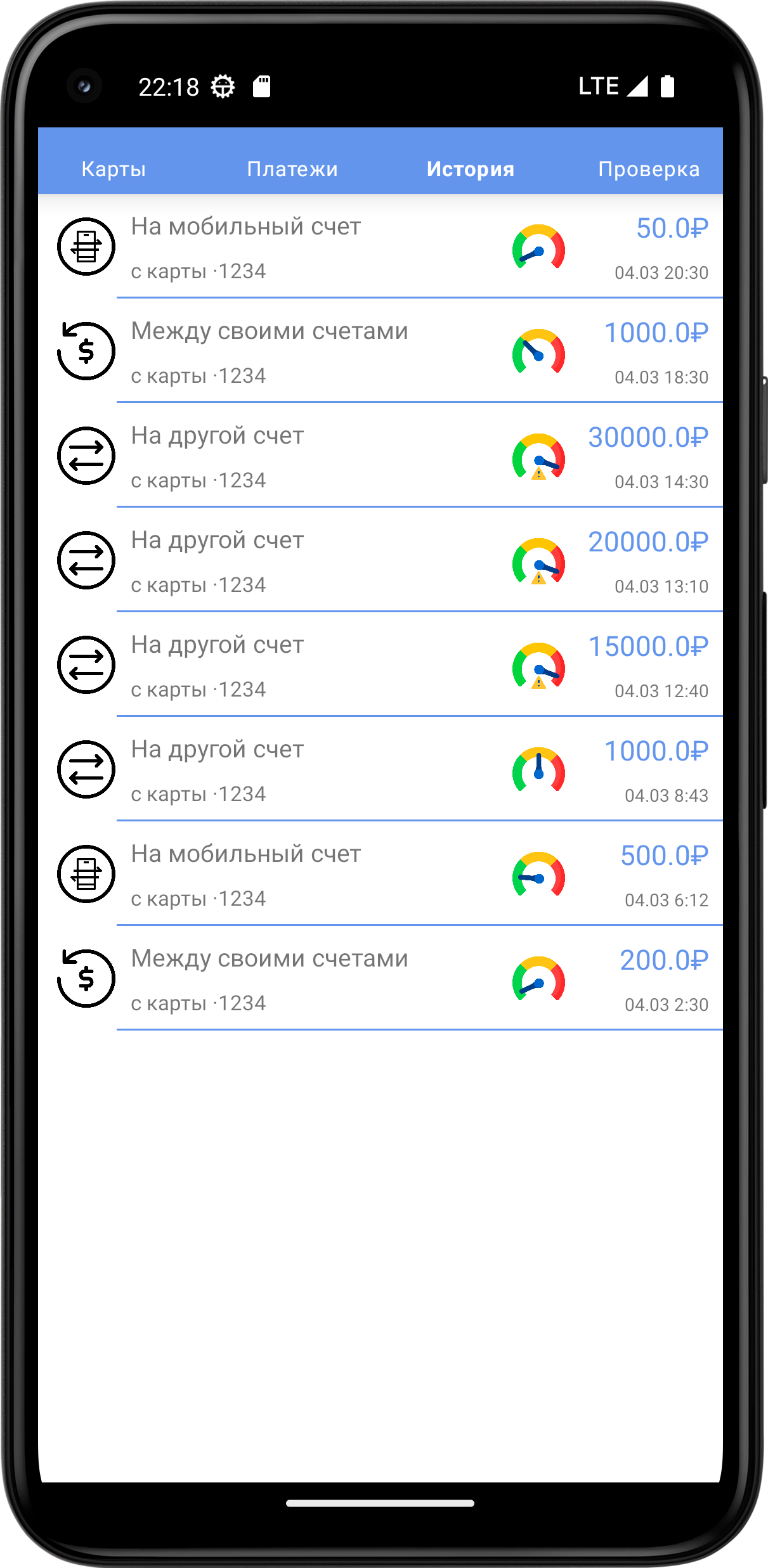
**Рисунок 3.3 — Демонстрация мобильного приложения. Вывод типов платежей**

**Рисунок 3.4 демонстрирует экран процесса совершения транзакции, в данном случае — перевода «Между своими». Пользователю предлагается выбрать карту списания («Откуда») и карту зачисления («Куда») из ранее добавленных на рисунке 3.2, а также ввести сумму перевода («Сколько»). Нажатие кнопки «Оплатить» инициирует отправку данных о транзакции на серверный модуль для анализа риска, согласно бизнес-процессу, описанному в разделе 2.1 и диаграмме на рисунке 2.4.**



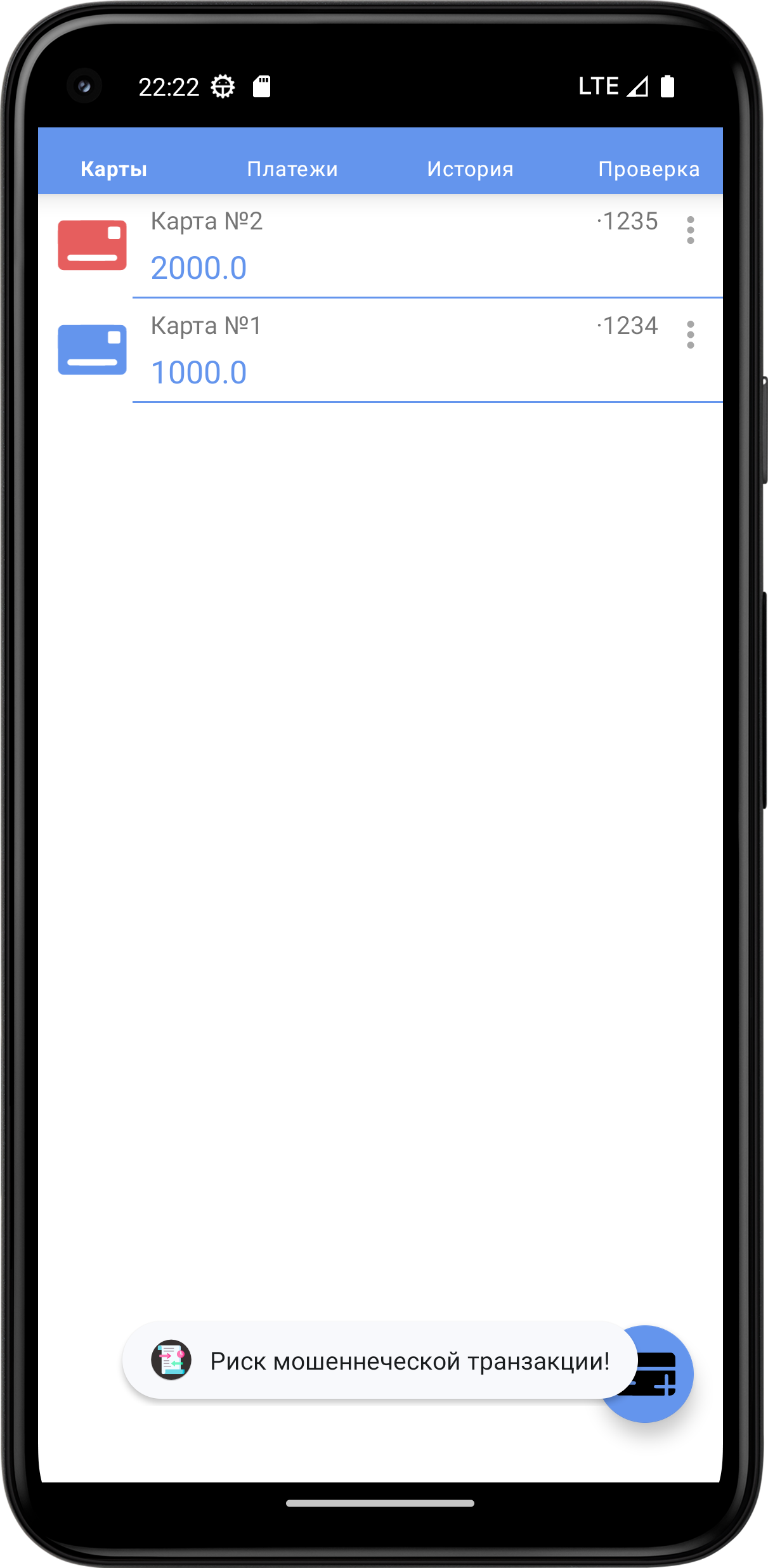
**Рисунок 3.4 — Демонстрация мобильного приложения. Процесс совершения транзакции**

**Результаты совершенных транзакций и их статус проверки на мошенничество отображаются на вкладке «История» изображены на рисунке 3.5. Каждая запись содержит иконку типа платежа, описание, сумму, дату и время. Важным элементом является иконка справа, в виде спидометра с цветовой индикацией, которая визуализирует оценку риска данной транзакции, полученную от антифрод-системы. Это позволяет пользователю ретроспективно оценить безопасность своих операций.**



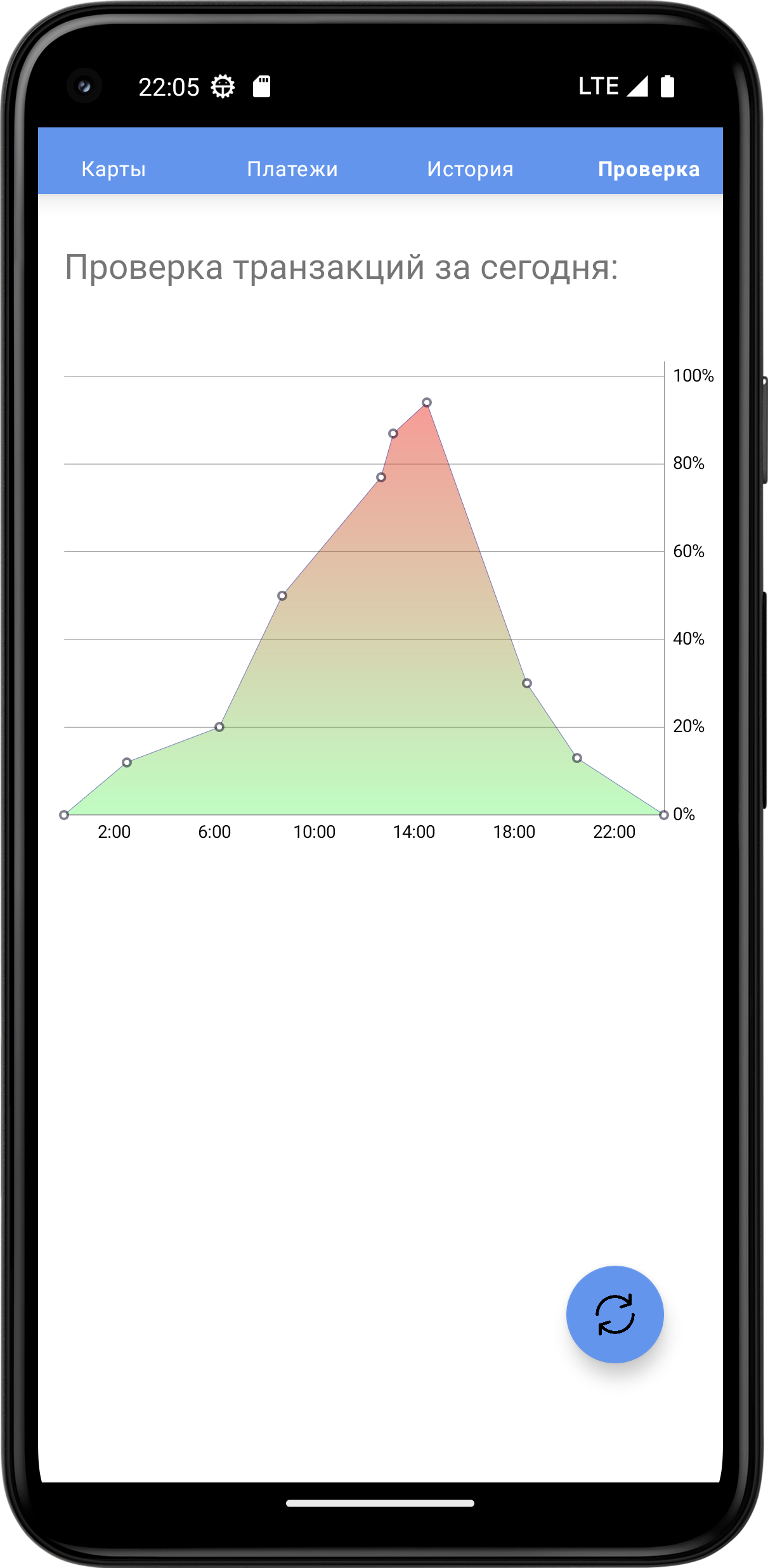
**Рисунок 3.5 — Демонстрация мобильного приложения. Вывод истории транзакций**

**Ключевой особенностью прототипа является механизм проактивного уведомления. Рисунок 3.6 показывает, как система информирует пользователя о высоком риске непосредственно после попытки совершения подозрительной транзакции. Всплывающее сообщение с текстом «Риск мошеннической транзакции!» появляется в нижней части экрана, привлекая внимание пользователя к потенциальной угрозе и позволяя ему предпринять действия до фактического завершения опасной операции.**



**Рисунок 3.6 — Демонстрация мобильного приложения. Уведомление об совершении подозрительной транзакции**

**Для предоставления пользователю обобщенной картины уровня безопасности его операций в приложении предусмотрена вкладка «Проверка», представленная на рисунке 3.7. Здесь представлена визуализация данных о рисках транзакций за определенный период в виде графика. Ось X отображает время, ось Y – процентный уровень риска. Цветовой градиент области под графиком (от зеленого к красному) интуитивно показывает периоды с низким и высоким уровнем подозрительной активности. Это дает пользователю дополнительный инструмент для анализа безопасности своих финансов.**



**Рисунок 3.7 — Демонстрация мобильного приложения. Визуализация уровня риска транзакций**

**Представленные экраны демонстрируют реализацию основных функциональных требований к прототипу, включая симуляцию банковских операций, взаимодействие с системой оценки рисков, наглядное отображение результатов проверки и, что наиболее важно, механизм проактивного уведомления пользователя о потенциально мошеннических транзакциях, а также средства для визуального анализа уровня риска. Пользовательский интерфейс спроектирован с учетом требований удобства использования.**

### ****Разработка взаимодействия с Антифрод-системой****

**Ключевым аспектом функциональности разработанного прототипа мобильного приложения является его способность надежно и эффективно взаимодействовать с внешней системой, предназначенной для оценки риска транзакций – антифрод-сервисом. Эта коммуникация является ядром механизма проактивного информирования пользователя. Взаимодействие между мобильным приложением и этим внешним сервисом анализа мошенничества осуществляется посредством сетевых запросов по протоколу HTTPS, что является стандартом де-факто и гарантирует необходимое шифрование и защиту целостности передаваемых данных (параметров транзакции и результатов анализа) во время их транзита по сети.**

**Для стандартизации и упрощения этого взаимодействия на уровне протокола обмена данными был принят подход, основанный на открытом стандарте KServe V2 Inference Protocol. Выбор этого протокола вместо разработки проприетарного API был продиктован следующими соображениями:**

* **Унификация: KServe V2 предоставляет единый, четко специфицированный формат для отправки входных данных (признаков транзакции) на анализ и получения структурированных результатов от моделей машинного обучения. Это позволяет мобильному приложению единообразно формировать запросы и парсить ответы, независимо от конкретной реализации антифрод-сервиса, если он соответствует стандарту.**
* **Стандартизация интерфейса на уровне протокола обеспечивает гибкость. В будущем антифрод-сервис может быть легко заменен или обновлен (например, переход на другую ML-модель или другого поставщика услуг) без необходимости внесения кардинальных изменений в сетевой слой мобильного приложения, пока новый сервис поддерживает KServe V2.Использование общепринятых стандартов, особенно в области MLOps и взаимодействия с ML-моделями, упрощает интеграцию, тестирование и дальнейшую поддержку системы, а также соответствует современным тенденциям в разработке.**

**Для практической реализации сетевых запросов из мобильного приложения к Anti-Fraud API была выбрана популярная и хорошо зарекомендовавшая себя библиотека Retrofit. Retrofit — это типобезопасный HTTP-клиент для Android и Java/Kotlin, разработанный Square, который значительно упрощает процесс отправки сетевых запросов и обработки ответов путем абстрагирования деталей низкоуровневой работы с HTTP. Процесс работы с Retrofit включает в себя такие процессы как:**

* **Определение API интерфейса: Первым шагом является создание Kotlin-интерфейса, который служит контрактом для взаимодействия с API. В этом интерфейсе каждая функция соответствует конкретному эндпоинту API. С помощью специальных аннотаций Retrofit (например, @GET, @POST, @PUT, @DELETE) указывается HTTP-метод и относительный путь к эндпоинту. Дополнительные аннотации (@Path, @Query, @Body и др.) используются для определения того, как параметры функции должны быть переданы в запрос (в URL, как query-параметры, в теле запроса и т.д.). Для интеграции с современным асинхронным подходом в Android используется ключевое слово suspend перед функциями интерфейса, что позволяет вызывать их из корутин без блокировки основного потока. Таким образом, интерфейс декларативно описывает, какие запросы можно выполнить и как они должны быть сформированы.**
* **Модели данных (Data Classes): Для того чтобы Retrofit мог автоматически преобразовывать данные между объектами Kotlin и форматом JSON (или другим, используемым API), необходимо определить классы данных (data classes в Kotlin). Эти классы должны точно отражать структуру JSON-объектов, ожидаемых API в теле запроса и возвращаемых в теле ответа. Имена полей в data-классах должны соответствовать ключам в JSON. Для протокола KServe V2 создаются классы, представляющие общую структуру запроса с полем inputs и ответа с полем outputs, а также вложенные классы для описания отдельных входных (InputFeature) и выходных (OutputFeature) признаков с их именами, типами данных, размерностью и самими значениями. Наличие этих четко определенных моделей данных делает работу с API типобезопасной.**
* **Конфигурирование и создание экземпляра Retrofit: В приложении необходимо настроить и создать объект Retrofit, который будет фабрикой для генерации реализаций API интерфейсов. Это делается с использованием паттерна Builder. В процессе конфигурации задается базовый URL-адрес API (основная часть URL, общая для всех эндпоинтов). Крайне важным шагом является добавление ConverterFactory – компонента, отвечающего за сериализацию (преобразование объектов Kotlin в JSON для отправки) и десериализацию (преобразование JSON-ответа сервера в объекты Kotlin). Популярными реализациями являются GsonConverterFactory или MoshiConverterFactory. При необходимости можно также сконфигурировать и передать собственный экземпляр OkHttpClient (библиотеки, которую Retrofit использует для выполнения HTTP-запросов "под капотом"), чтобы настроить таймауты, добавить перехватчики (interceptors) для логирования, модификации запросов (например, добавления заголовков аутентификации) или других продвинутых сетевых настроек.**
* **Создание реализации API сервиса: После того как экземпляр Retrofit сконфигурирован, он используется для создания конкретной реализации ранее определенного API интерфейса. Это делается вызовом метода create() у объекта Retrofit, в который передается класс интерфейса. Retrofit динамически генерирует класс, реализующий этот интерфейс, и вся логика по формированию HTTP-запросов, их отправке и обработке ответов на основе аннотаций в интерфейсе инкапсулируется в этом сгенерированном объекте.**
* **Вызов методов API: Полученный экземпляр реализации API сервиса используется в соответствующих частях приложения (например, в слое репозитория или во ViewModel) для выполнения сетевых запросов. Вызов конкретного метода API (соответствующего suspend функции в интерфейсе) должен происходить внутри корутины (например, запущенной через viewModelScope.launch). При вызове метода ему передаются необходимые параметры (например, объект data-класса, представляющий тело запроса). Крайне важно оборачивать вызов сетевого метода в блок try-catch, чтобы корректно обрабатывать возможные исключения, которые Retrofit может выбросить в случае сетевых проблем (нет соединения, таймаут), ошибок сервера (коды состояния 4xx, 5xx) или проблем с парсингом ответа (несоответствие JSON ожидаемой структуре data-класса). В блоке try обрабатывается успешный ответ (десериализованный в соответствующий data-класс), а в блоке catch обрабатываются различные типы ошибок.**

**Как упоминалось ранее, для управления зависимостями и следования принципам чистой архитектуры, настроенные экземпляры Retrofit, OkHttpClient и сгенерированной реализации AntiFraudApiService регистрируются в Koin-модулях. Это позволяет легко внедрять (inject) сервис API в те компоненты, где он необходим, без жесткой связи между ними.**

**С учетом использования Retrofit, детальный процесс анализа транзакции выглядит следующим образом: приложение собирает данные транзакции, создает соответствующий объект Kotlin data-класса (KServeRequest), затем вызывает suspend функцию из сгенерированной реализации API сервиса. Retrofit берет на себя всю работу по сериализации этого объекта в JSON, формированию корректного HTTP POST-запроса с нужными заголовками и телом, отправке его по HTTPS на указанный эндпоинт с использованием OkHttpClient. После получения ответа от сервера, Retrofit анализирует код состояния. При успехе он десериализует тело ответа (JSON) обратно в объект Kotlin data-класса (KServeResponse). Приложение затем извлекает из этого объекта необходимые данные (например, оценку риска) и обрабатывает их. В случае любой ошибки на этапах запроса или ответа, Retrofit генерирует исключение, которое перехватывается и обрабатывается в приложении, информируя пользователя о проблеме.**

**Использование HTTPS прозрачно обеспечивается OkHttpClient, на котором базируется Retrofit. При работе с Ngrok для тестирования, соответствующий HTTPS URL указывается при конфигурации Retrofit. Хотя Retrofit значительно упрощает безопасное взаимодействие, в промышленных средах может потребоваться более глубокая настройка безопасности на уровне OkHttpClient, например, через механизмы вроде Certificate Pinning. Вопросы аутентификации/авторизации (передача API-ключей, токенов и т.д.) обычно решаются через добавление OkHttp Interceptors, которые автоматически добавляют необходимые заголовки ко всем или к определенным запросам. В рамках данного прототипа эти аспекты упрощены, но Retrofit предоставляет все необходимые механизмы для их реализации в полноценной системе.**

**В итоге, библиотека Retrofit выступает ключевым инструментом в реализации сетевого слоя мобильного приложения, обеспечивая декларативный, типобезопасный и асинхронный подход к взаимодействию с внешним Anti-Fraud API по протоколу KServe V2 через HTTPS, что существенно упрощает разработку и повышает надежность коммуникации.**

## ****Тестирование ИС****

**Тестирование является неотъемлемым этапом разработки любого программного продукта, включая прототип, разработанный в рамках данной выпускной квалификационной работы. Основная цель тестирования заключалась в проверке соответствия реализованного мобильного приложения заявленным функциональным и нефункциональным требованиям, корректности его работы на целевой платформе Android, а также в оценке эффективности предложенного механизма информирования пользователя о потенциально мошеннических транзакциях.**

### Методология тестирования

В ходе разработки и оценки прототипа применялся комплексный подход к тестированию, с использованием Smoke Testing и функционального тестирования.

Smoke Testing проводился регулярно после внесения значимых изменений в кодовую базу или сборки новой версии прототипа. Цель дымового тестирования – быстрая проверка работоспособности самых критичных функций приложения: успешный запуск, навигация между основными экранами (добавление карты, платежи, история, проверка), базовая отрисовка интерфейса. Это позволяло оперативно выявить серьезные дефекты (например, падение приложения при старте), блокирующие дальнейшее, более глубокое тестирование, и убедиться в стабильности сборки перед передачей на следующие этапы проверки.

Функциональное тестирование - это основной вид тестирования, направленный на детальную проверку соответствия функциональности прототипа требованиям, описанным в разделе 2.3. Функциональное тестирование охватывало все ключевые сценарии использования приложения, имитируя действия конечного пользователя. Проверялись:

* Корректность симуляции добавления банковских карт и сохранения их данных в локальной базе данных SQLite.
* Правильность ввода данных при симуляции различных типов транзакций (переводы, платежи).
* Успешность и корректность взаимодействия с внешним Anti-Fraud API (отправка запроса в формате KServe V2 через Retrofit, обработка успешного ответа с разными уровнями риска, обработка ошибок сети и API).
* Отображение истории симулированных транзакций, включая корректное считывание данных из SQLite и правильную визуализацию уровня риска (цветовая индикация).
* Срабатывание и корректное отображение проактивного уведомления о высоком риске транзакции.
* Функционирование экрана визуализации общей статистики рисков (график).
* Работа базовой навигации и элементов управления интерфейса.
* Функциональное тестирование проводилось на основе заранее подготовленных тестовых сценариев (см. таблицу 4.1).
* Тестирование пользовательского интерфейса и удобства использования (UI/UX Testing):

Поскольку одной из ключевых целей работы является демонстрация улучшенного пользовательского опыта, особое внимание уделялось тестированию интерфейса. Проверялись:

* Интуитивность навигации и понятность элементов управления.
* Корректность отображения интерфейса на экранах с разным разрешением и плотностью пикселей.
* Ясность и информативность визуализации уровня риска (цветовые индикаторы, график).
* Понятность и ненавязчивость механизма уведомления о высоком риске.

В рамках этого тестирования собиралась обратная связь (аналогично подходу User Acceptance Testing - UAT, но в упрощенном формате для прототипа) от коллег или научных руководителей для оценки восприятия предложенного решения и внесения необходимых корректировок в дизайн и формулировки.

### Проведение процедуры тестирования

Тестирование разработанного прототипа мобильного приложения проводилось на протяжении всего цикла его создания. Для обеспечения проверки совместимости и корректного отображения интерфейса использовались как виртуальные устройства (эмуляторы Android Studio), так и физические устройства.

Тестовая среда:

* Виртуальные устройства: Google Pixel 5 (Android 12, разрешение 1080x2340), Nexus 5X (Android 8.1, разрешение 1080x1920).
* Физическое устройство: Xiaomi Redmi Note 9 Pro (Android 11, разрешение 1080x2400).

Использование различных устройств и версий ОС Android позволило проверить адаптивность интерфейса и выявить потенциальные проблемы совместимости.

Регулярно применялось дымовое тестирование после интеграции новых функций или исправления ошибок. Основной объем проверок пришелся на функциональное тестирование согласно сценариям, приведенным в таблице 4.1. Особое внимание уделялось проверке сквозного сценария: симуляция транзакции, отправка запроса на Anti-Fraud API, получение ответа, отображение результата в истории и (при необходимости) показ уведомления.

Тестирование взаимодействия с Anti-Fraud API проводилось как с использованием тестового endpoint, так и с имитацией различных ответов (низкий риск, высокий риск, ошибка) для проверки всех ветвей логики приложения.

На этапе тестирования UI/UX была получена обратная связь, на основании которой внесены корректировки в цветовую схему индикаторов риска для лучшей читаемости и изменены формулировки в уведомлениях для большей ясности. В ходе функционального тестирования были выявлены и устранены ошибки, связанные с обработкой сетевых ошибок при обращении к API и корректным обновлением данных в локальной базе SQLite после совершения транзакции.

Основные сценарии тестирования прототипа мобильного приложения приведены в таблице 4.1.

Таблица 4.1 – Основные сценарии тестирования прототипа мобильного приложения

| Тест | Описание теста | Ожидаемый результат | Результат |
| --- | --- | --- | --- |
| **Добавление карты** | Ввод названия и баланса карты на экране "Создание карты". Нажатие кнопки сохранения. | Карта успешно сохраняется в локальной базе данных SQLite. Новая карта появляется в списке на главном экране вкладки "Карты" с указанными данными. | Совпал с ожидаемым |
| **Отображение списка карт** | Переход на вкладку "Карты". | Корректно отображается список всех ранее добавленных карт с их названиями, балансами и маскированными номерами. Отображается кнопка для добавления новой карты. | Совпал с ожидаемым |
| **Инициация транзакции (Перевод)** | Переход на вкладку "Платежи", выбор "Перевод между своими". Выбор карт списания/зачисления, ввод суммы. Нажатие "Оплатить". | Приложение формирует данные о транзакции. Инициируется отправка запроса на Anti-Fraud API. Пользователь видит индикатор процесса (если реализован). | Совпал с ожидаемым |

Продолжение таблицы 4.1

| Тест | Описание теста | Ожидаемый результат | Результат |
| --- | --- | --- | --- |
| **Взаимодействие с API (Низкий риск)** | Симуляция транзакции. Anti-Fraud API возвращает ответ с низким уровнем риска. | Приложение успешно обрабатывает ответ API. Транзакция сохраняется в локальной истории SQLite с пометкой о низком риске (зеленый индикатор). *Проактивное уведомление не отображается*. | Совпал с ожидаемым |
| **Взаимодействие с API (Высокий риск)** | Симуляция транзакции. Anti-Fraud API возвращает ответ с высоким уровнем риска. | Приложение успешно обрабатывает ответ API.  *уведомление* (например, Toast) с предупреждением о риске. Транзакция сохраняется в истории SQLite с пометкой о высоком риске (красный индикатор). | Совпал с ожидаемым |
| **Взаимодействие с API (Ошибка сети)** | Симуляция транзакции при отсутствии интернет-соединения или недоступности эндпоинта API. | Приложение корректно обрабатывает ошибку сети (например, показывает сообщение "Ошибка сети, не удалось проверить транзакцию"). Транзакция может не сохраняться в историю или сохраняться с неопределенным статусом. | Совпал с ожидаемым |

Продолжение таблицы 4.1

| Тест | Описание теста | Ожидаемый результат | Результат |
| --- | --- | --- | --- |
| **Взаимодействие с API (Ошибка сервера)** | Симуляция транзакции. Anti-Fraud API возвращает ошибку сервера (например, статус 500). | Приложение корректно обрабатывает ошибку сервера (например, показывает сообщение "Ошибка сервера при проверке транзакции"). Статус транзакции в истории соответствующий. | Совпал с ожидаемым |
| **Отображение истории транзакций** | Переход на вкладку "История". Прокрутка списка. | Отображается полный список симулированных транзакций из локальной базы SQLite в хронологическом порядке. Для каждой транзакции указаны тип, сумма, дата и корректный индикатор риска. | Совпал с ожидаемым |
| **Отображение графика риска** | Переход на вкладку "Проверка". | Корректно отображается график, визуализирующий динамику уровня риска транзакций за выбранный период (данные берутся из истории в SQLite). | Совпал с ожидаемым |

Продолжение таблицы 4.1

| Тест | Описание теста | Ожидаемый результат | Результат |
| --- | --- | --- | --- |
| **Навигация между экранами** | Последовательные переходы между всеми вкладками и экранами приложения (Карты, Платежи, История, Проверка, Создание карты, Процесс транзакции). | Переходы осуществляются плавно, без сбоев. Состояние экранов сохраняется корректно (где это предусмотрено). Активная вкладка подсвечивается. | Совпал с ожидаемым |
| **Адаптивность интерфейса** | Запуск и использование приложения на устройствах/эмуляторах с разным разрешением экрана (указанных в тестовой среде). | Все элементы интерфейса (текст, кнопки, иконки, график) отображаются корректно, без наложений, обрезаний или выхода за границы экрана. | Совпал с ожидаемым |

# ЗАКЛЮЧЕНИЕ

В ходе прохождения практики были выполнены следующие ключевые этапы, заложившие основу для дальнейшей разработки:

* Проведен глубокий анализ предметной области: Изучены современные угрозы мобильного банковского мошенничества (фишинг, социальная инженерия, вредоносные ПО, DeepFake-технологии), а также существующие методы противодействия.
* Исследованы аналоги: Проанализированы антифрод-решения и подходы к уведомлению пользователей в ведущих российских банках (СберБанк, ВТБ, Т-Банк) и регуляторные инициативы (ЦБ РФ), выявлены их сильные стороны и ограничения, что позволило обосновать актуальность предлагаемого подхода.
* Выполнено проектирование информационной системы: Разработаны модели бизнес-процессов взаимодействия пользователя с системой с использованием нотации UML (диаграммы вариантов использования, последовательности, деятельности), наглядно демонстрирующие предлагаемый сценарий проактивного информирования.
* Определена сервис-ориентированная архитектура системы, включающая клиентский модуль (мобильное приложение), серверный модуль (Backend), модуль анализа и базу данных.
* Сформулированы детальные функциональные, нефункциональные требования, а также требования к интерфейсу и программно-аппаратному обеспечению.
* Разработан и продемонстрирован протопит мобильного приложения

Основным результатом выпускной квалификационной работы является комплексное проектирование информационной системы и создание теоретической и технологической базы для разработки прототипа. Ключевое отличие проектируемого решения заключается в акценте на проактивное уведомление пользователя о высоком риске до завершения транзакции, предоставлении наглядной визуализации риска и интуитивно понятных инструментов для подтверждения или отклонения операции, а также интерактивном обучении, что направлено на повышение безопасности и улучшение пользовательского опыта по сравнению с существующими подходами.

Задачи, поставленные перед работой, были успешно выполнены. Приобретены и закреплены навыки анализа предметной области, системного проектирования, моделирования бизнес-процессов, выбора современных технологий и инструментов разработки, создан прототип мобильного банковского приложения, который позволяет продемонстрировать новый подход к уведомлению пользователей о рисках совершения мошеннических транзакций.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Группа IB. Тренды мошенничества в 2024–2025 гг. URL: https://www.group-ib.ru (дата обращения: 02.04.2025)
2. Положительные технологии. Уязвимости мобильных банкингов – 2025. URL: https://www.ptsecurity.com (дата обращения: 02.04.2025)
3. Отчет Национальной службы кибербезопасности. Годовой отчет – 2024. URL: https://nbki.ru (дата обращения: 02.04.2025)
4. Центральный банк Российской Федерации. Указание №5434-У «О стандартах биометрической аутентификации» (2025 г.). URL: https://cbr.ru (дата обращения: 02.04.2025)
5. Лаборатория Касперского. Социальная инженерия в РФ – 2025. URL: https://www.kaspersky.ru (дата обращения: 02.04.2025)
6. Минцифра России. Постановление Правительства РФ №235-Пр «О мерах защиты пользователей электронных платёжных систем». URL: https://digital.gov.ru (дата обращения: 02.04.2025)
7. Банк России. Отчет «О состоянии рынка платежных услуг за 2023 год». URL: https://cbr.ru (дата обращения: 19.04.2025)
8. СберБанк. Техническая документация API «Sberbank AI Solutions». URL: https://developer.sber.ru (дата обращения: 19.04.2025)
9. ВТБ. Пресс-релиз «Запуск системы VTB Anti-Fraud». URL: https://vtb.ru/press (дата обращения: 19.04.2025)
10. Петров А.И. Интервью с CTO Т-Банка // Журнал «Банковские технологии». – 2024. – №3. – С. 45–49.
11. Европейский Союз. Директива 2024/0078 «О платежных услугах (PSD3)». URL: https://eur-lex.europa.eu (дата обращения: 19.04.2025).
12. Тинькофф Банк. Отчет «Итоги внедрения динамической биометрии». URL: https://www.tinkoff.ru (дата обращения: 21.04.2025).
13. НИУ ВШЭ. Исследование «Поведенческие паттерны пользователей мобильного банкинга». – М.: Изд-во ВШЭ, 2024. – 134 с.
14. Нотация и семантика языка UML: Информация Автор: Александр Леоненков | Школа IT-менеджмента АНХ при Правительстве РФ http://www.intuit.ru/studies/courses/32/32/info (дата обращения: 22.04.2025)
15. Язык UML 2 в анализе и проектировании программных систем и бизнес-процессов: Информация Автор: Александр Леоненков  
    http://www.intuit.ru/studies/courses/480/336/info (дата обращения: 22.04.2025)
16. Хомоненко А.Д., Басыров А.Г., Бубнов В.П., Забродин А.В., Краснов С.А., Лохвицкий В.А.,Тырва А.В. Модели и методы исследования информационных систем: монография Издательство "Лань" 204.с. - 2019г https://e.lanbook.com/reader/book/119640#94 (дата обращения: 22.04.2025)
17. Моделирование бизнес-процессов: учебное пособие Кравченко А. В.,Драгунова Е. В.,Кириллов Ю. В. Издательство Лань Новосибирский государственный технический университет , 136 стр., 2020г.  
    https://e.lanbook.com/book/152364 (дата обращения: 22.04.2025)
18. Обзор операций, совершенных без согласия клиентов финансовых организаций за 2023 год // Центральный банк Российской Федерации (Банк России). – 2024. – [Электронный ресурс]. URL: https://cbr.ru/analytics/ib/operations\_review/ (дата обращения: 27.04.2025).
19. Мобильный банкинг в России: исследование НАФИ // Аналитический центр НАФИ. – 2023. – [Электронный ресурс]. URL: https://nafi.ru/analytics/mobilnyy-banking-v-rossii-issledovanie-nafi-2023/ (дата обращения: 27.04.2025).
20. Безопасность банковских карт и счетов: опрос ВЦИОМ // Всероссийский центр изучения общественного мнения (ВЦИОМ). – 2023. – № 4715. – [Электронный ресурс]. URL: https://wciom.ru/analytical-reviews/analiticheskii-obzor/bezopasnost-bankovskikh-kart-i-schetov (дата обращения: 27.04.2025).
21. Arrieta A.B., Díaz-Rodríguez N., Del Ser J., et al. Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI // Information Fusion. – 2020. – Vol. 58. – P. 82-115. – DOI: 10.1016/j.inffus.2019.12.012. URL: https://www.sciencedirect.com/science/article/pii/S156625351930408X (дата обращения: 27.04.2025).