МИНОБРНАУКИ РОССИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ

«ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ» (ФГБОУ ВО «ВГУ»)

Факультет компьютерных наук

Кафедра программирования и информационных технологий

Курсовая работа

Разработка мобильного приложения «FreelanceFinder»

09.03.02 Информационные системы и технологии Программная инженерия в информационных системах

Зав. кафедрой Махортов С.Д., д.ф-м.н., профессор Обучающийся Капустин М.И., 3 курс, д/о Обучающийся Коротаев Т.А., 3 курс, д/о Обучающийся Киреев О.А., 3 курс, д/о Руководитель Тарасов В.С., ст. преподаватель

Воронеж 2024

# Содержание

[Введение 4](#_bookmark0)

1. [Постановка задачи 5](#_bookmark1)
   1. [Цели создания системы 5](#_bookmark2)
   2. [Задачи проекта 5](#_bookmark3)
   3. [Функциональные требования: 5](#_bookmark4)
      1. [Для неавторизованного пользователя: 5](#_bookmark5)
      2. [Для авторизованного пользователя (Фрилансер) 5](#_bookmark6)
      3. [Для авторизованного пользователя (Заказчик): 6](#_bookmark7)
      4. [Для администратора: 6](#_bookmark8)
   4. [Нефункциональные требования 7](#_bookmark9)
2. [Анализ предметной области 8](#_bookmark10)
   1. [Терминология 8](#_bookmark11)
   2. [Обзор аналогов 10](#_bookmark12)
      1. [Хабр Фриланс 10](#_bookmark13)
      2. [Upwork 12](#_bookmark14)
   3. [Моделирование системы 13](#_bookmark15)
      1. [Диаграмма в стиле методологии IDEF0 13](#_bookmark16)
      2. [Диаграмма прецедентов 14](#_bookmark17)
      3. [Диаграммы последовательности 18](#_bookmark18)
      4. [Диаграмма развертывания 19](#_bookmark19)
      5. [Диаграммы состояния 20](#_bookmark20)
      6. [Диаграмма объектов 23](#_bookmark21)
      7. [Диаграммы активности 24](#_bookmark22)
3. [Реализация 26](#_bookmark23)
   1. [Средства реализации 26](#_bookmark24)
   2. [Структура базы данных 27](#_bookmark25)
   3. [Информационная безопасность и защита данных 29](#_bookmark26)
      1. [SpringSecurity 29](#_bookmark27)
      2. [JWT 31](#_bookmark28)
      3. [Защита данных на серверной стороне 32](#_bookmark29)
      4. [Защита данных на клиентской стороне 33](#_bookmark30)
   4. [Примеры дизайна и его функциональное назначение 34](#_bookmark31)
      1. [Экран входа в систему 34](#_bookmark32)
      2. [Главный экран заказов 35](#_bookmark33)
      3. [Экран заказа 37](#_bookmark34)
      4. [Экран обращений 38](#_bookmark35)
4. [Аналитика 40](#_bookmark36)

[Заключение 44](#_bookmark37)

[Список использованных источников 45](#_bookmark38)

# Введение

В современном мире фриланс становится все более популярной формой заработка для многих людей. Для облегчения процесса поиска и выполнения заказов разрабатывается специализированная система - фриланс биржа, предоставляющая удобный доступ к необходимой информации и функциональности.

Фриланс биржа представляет собой программное приложение для мобильных устройств, которое помогает фрилансерам и заказчикам находить подходящие проекты, создавать собственные и получать отзывы о своей работе.

В данной курсовой работе рассматривается процесс разработки собственного мобильного приложения.

В рамках работы будут рассмотрены различные аспекты разработки, начиная с анализа предметной области, определения его концепции и основных требований. Затем будет изучено проектирование пользовательского интерфейса и пользовательского опыта, с учетом современных тенденций и личных практик в этой области. Важное внимание будет уделено выбору и интеграции соответствующих технологий для обеспечения необходимых функций.

Эта система поможет пользователям находить новые возможности для роста и развития в своей профессиональной деятельности, а также облегчит процесс поиска подходящих исполнителей для заказчиков, предоставляя им удобный доступ к базе квалифицированных специалистов. Кроме того, приложение будет способствовать улучшению взаимодействия между фрилансерами и заказчиками, обеспечивая эффективную коммуникацию и взаимопонимание.

## Постановка задачи

## Цели создания системы

Целью данной работы является создание удобной платформы для поиска и найма специалистов с функциями создания, просмотра и редактирования заказов; отклика на заказы и предложением заказа специалисту.

## Задачи проекта

* поиск подходящих проектов для фрилансеров;
* анализ навыков и опыта фрилансеров;
* организация взаимодействия между фрилансерами и заказчиками;
* обеспечение безопасности и конфиденциальности личных данных пользователей.

## Функциональные требования:

Для каждой группы пользователей предусмотрены свои функции.

## Для неавторизованного пользователя:

* + - * просмотр заказов: Возможность просматривать заказы без возможности отклика;
      * просмотр фрилансеров: Возможность просматривать фрилансеров без возможности предложения заказа;
      * регистрация и вход: Возможность регистрации нового аккаунта. Вход в существующий аккаунт для получения полного доступа к функциям приложения.

## Для авторизованного пользователя (Фрилансер):

* + - * отклик на заказ: Возможность просмотра заказов и отклика на интересующий;
      * просмотр других фрилансеров;
      * принятие заказа: Возможность принять заказ, предложенный заказчиком;
      * сбор отзывов: Возможность оставить обратную связь о заказчике;
      * личный профиль: Возможность редактировать личные данные, включая имя пользователя, почту, информацию о пользователе, контактную информацию, стоимость и пароль.

## Для авторизованного пользователя (Заказчик):

* + - * просмотр заказов: Возможность просмотра заказов без возможности отклика;
      * предложение фрилансеру заказа: Возможность просмотра фрилансеров с возможностью оставить предложение о выполнении заказа;
      * личный профиль: Возможность редактировать личные данные, включая имя пользователя, почту, информацию о пользователе, контактную информацию и пароль;
      * принятие отклика: Возможность принять отклик фрилансера;
      * сбор отзывов: Возможность оставить обратную связь о фрилансера.

## Для администратора:

* + - * управление пользователями: Возможность просматривать и удалять пользовательские аккаунты;
      * управление заказами: Возможность редактировать и удалять заказы пользователей;
      * рассмотрение жалоб: Возможность редактировать и удалять заказы и пользовательские аккаунты по поступившей жалобе.

## Нефункциональные требования:

* безопасность – приложение должно обеспечивать достаточную защиту данных пользователей;
* полезность – приложение должно помогать пользователям решать их задачи;
* удобство использования – приложение должно быть просто в освоении и использовании;
* масштабируемость – приложение должно иметь возможность легко расширяться и дополнять функциональность.

## Анализ предметной области

## Терминология

Таблица 1 - Термины и сокращения

|  |  |
| --- | --- |
| Мобильное приложение | Программное обеспечение, предназначенное для работы на смартфонах, планшетах и других мобильных устройствах,  разработанное для конкретной платформы. |
| Клиент | Это аппаратный или программный  компонент вычислительной системы, посылающий запросы серверу. |
| Сервер | Выделенный или специализированный  компьютер для выполнения сервисного программного обеспечения. |
| База данных | Упорядоченный набор структурированной информации или данных, которые обычно хранятся в электронном виде в компьютерной системе. База данных обычно управляется системой  управления базами данных. |
| HTTP | Это протокол, позволяющий получать различные ресурсы, например HTML- документы. Протокол HTTP лежит в основе  обмена данными в Интернете. |
| GitHub | Крупнейший веб-сервис для хостинга  IT-проектов и их совместной разработки. |
| Администратор | Человек, имеющий доступ к  расширенной функциональности мобильного клиента. |

|  |  |
| --- | --- |
| Пользователь | Авторизованный в мобильном клиенте  человек, пользующийся функциональностью мобильного клиента. |
| Гость | Неавторизованный в мобильном клиенте человек, пользующийся ограниченной функциональностью  мобильного клиента. |
| Аутентификация | Процедура проверки подлинности, например, проверка подлинности пользователя путем сравнения введенного им пароля с паролем, сохраненным в базе  данных. |
| Авторизация | Предоставление определенному лицу или группе лиц прав на выполнение  определенных действий. |
| Контент | Наполнение мобильного клиента. |
| Фреймворк | Программное обеспечение, облегчающее разработку и объединение разных компонентов большого  программного проекта. |
| Заказчик | Тип пользователя, который  заинтересован в выполнении задачи по разработке программного обеспечения. |
| Фрилансер | Тип пользователя, который заинтересован в получении материального заработка за выполнение задач по разработке  программного обеспечения. |

|  |  |
| --- | --- |
| Аккаунт | Персональная страница пользователя  или личный кабинет, который создается после регистрации на сайте. |
| Личный кабинет | Раздел сервиса, в котором Пользователь может получить доступ к  своим данным. |
| API | Описание взаимодействия одной компьютерной программы с другой. |
| REST API | Стиль архитектуры программного обеспечения для построения распределенных масштабируемых веб-  сервисов. |
| Back-end | Программно-аппаратная часть сервиса. |
| Front-end | Клиентская сторона пользовательского интерфейса к программно-аппаратной части  сервиса. |

## Обзор аналогов

При разработке фриланс биржи, основной задачей которой является эффективное соединение фрилансеров и заказчиков, необходимо рассматривать разработку с точки зрения актуальности и уникальности проекта. Для оценки этих качеств необходимо прибегнуть к рассмотрению аналогов существующих фриланс бирж, адекватно оценивая все положительные и негативные черты того или иного продукта.

## Хабр Фриланс

Хабр Фриланс - это платформа для фрилансеров и заказчиков, предоставляющая простой и удобный доступ к широкому спектру услуг и

возможностей для эффективного сотрудничества. Главная цель платформы - сделать поиск и выполнение проектов более организованным, продуктивным и приятным для всех участников.

Хабр Фриланс обладает широким спектром предоставляемых услуг и с точки зрения авторского контента. На платформе представлены различные категории проектов, от разработки программного обеспечения до дизайна и копирайтинга. Кроме того, там публикуются статьи и обзоры, посвященные фрилансу и связанным с ним темам, что позволяет пользователям получать полезную и актуальную информацию.

На главном экране платформы (рисунок 1) представлены основные разделы, такие как поиск проектов, создание заданий, управление профилем и настройки. Все это позволяет пользователям легко и быстро навигироваться по платформе и находить необходимую информацию.

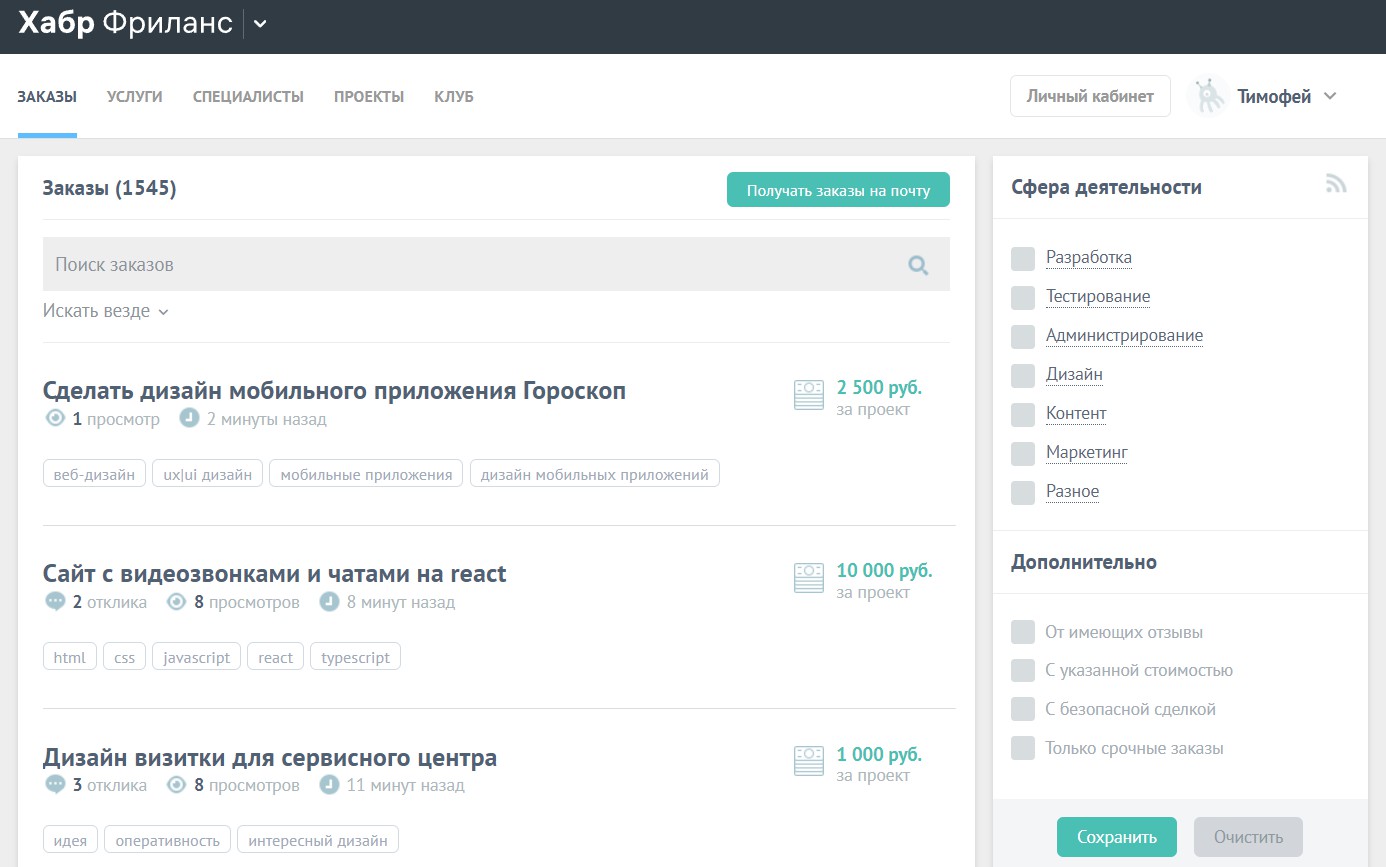


Рисунок 1 - Интерфейс страницы «Хабр Фриланс» Недостатки:

* + - * интерфейс блеклый и немного устарел;
      * отсутствие прозрачности и полноты статистики;
      * присутствуют комиссионные сборы.

## Upwork

Upwork - это международная онлайн-платформа для фрилансеров и заказчиков, которые ищут эффективные и удобные способы сотрудничества.

На этой платформе вы можете создавать профиль, указывая свои навыки и опыт, а также искать подходящие проекты и задания. Upwork предоставляет широкий спектр возможностей для фрилансеров, включая удобный поиск проектов, гибкие условия оплаты и возможность работы с клиентами со всего мира.

Кроме того, Upwork предлагает кэшбек-систему, которая позволяет фрилансерам получать вознаграждение за свои добрые дела. Например, вы можете пожертвовать часть своего заработка на благотворительность или участвовать в волонтерских проектах, чтобы получить бонусы и награды.

На рисунке 2 представлен главный экран платформы Upwork, где вы можете увидеть основные разделы и функции.

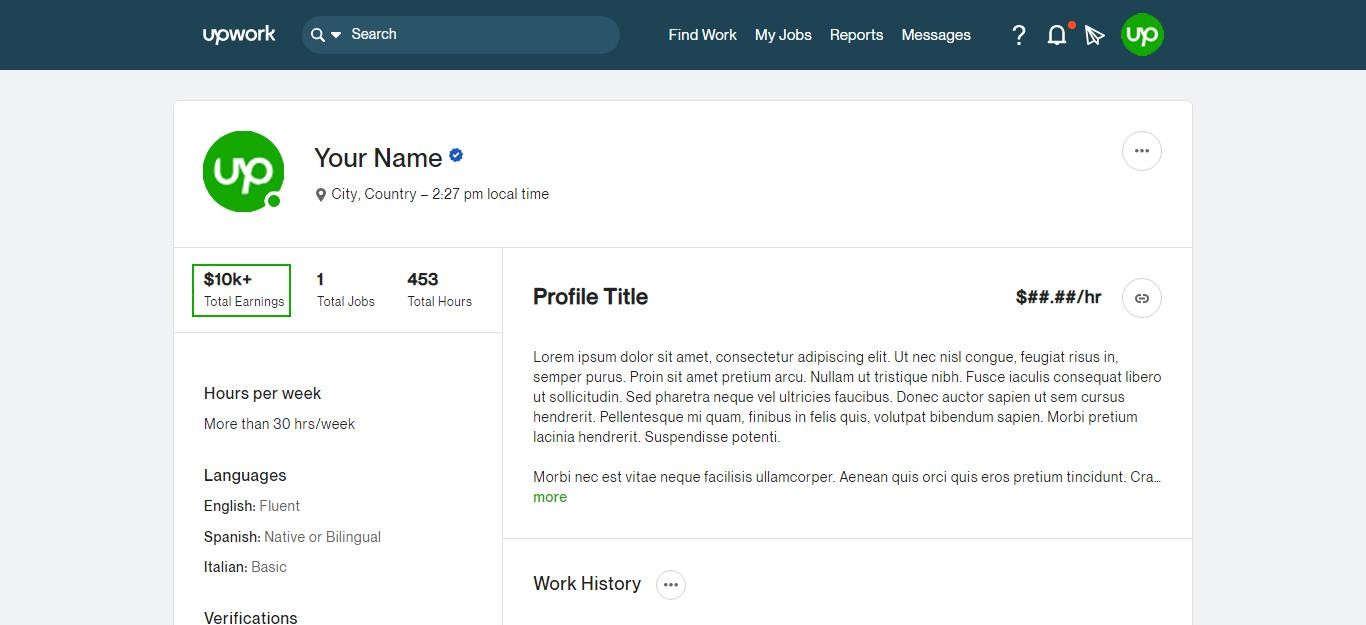


Рисунок 2 - Интерфейс страницы «Upwork»

Недостатки:

* слишком строгий и официальный интерфейс;
* отсутствие полной статистики;
* недоступен в некоторых регионах;
* присутствуют комиссионные сборы.

## Моделирование системы

## Диаграмма в стиле методологии IDEF0

IDF0 диаграмма представляет собой графическое представление бизнес- процесса в виде иерархической структуры функций. Основная цель IDF0 диаграммы состоит в том, чтобы показать, как различные функциональности взаимодействуют друг с другом и как они влияют на достижение целей организации. Она помогает улучшить понимание процессов и оптимизировать их для повышения эффективности огранизации. Данная диаграмма представлена на рисунках 3 и 4.

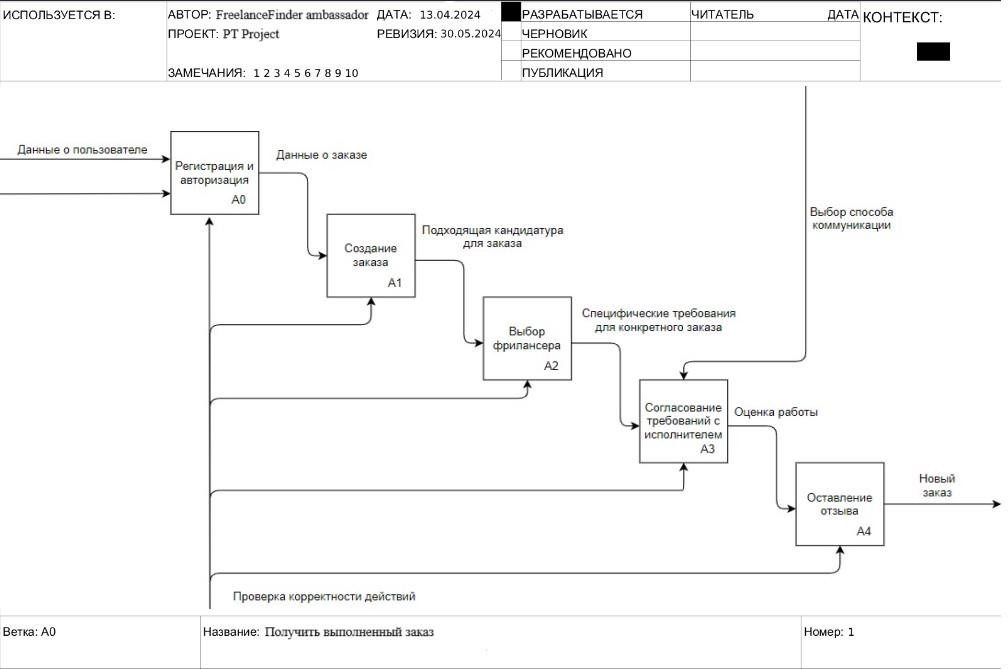


Рисунок 3 - Диаграмма в стиле методологии IDEF0-1

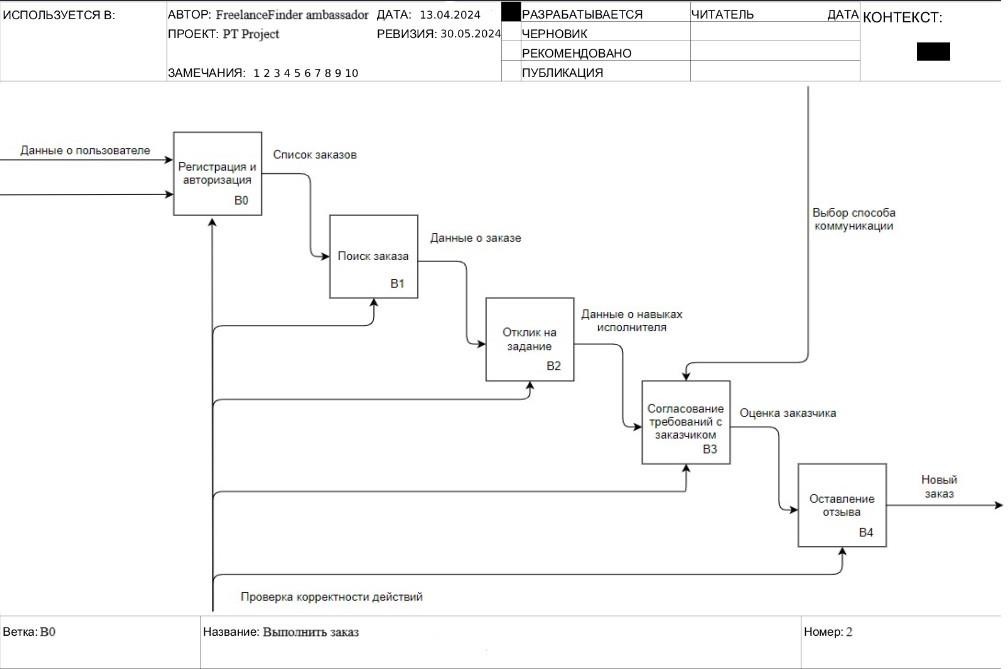


Рисунок 4 - Диаграмма в стиле методологии IDEF0-2

## Диаграмма прецедентов

Рассмотрим полную диаграмму для использования приложения разными типами пользователей. В данном случае необходимость составления диаграммы прецедентов продиктована прежде всего тем, что use-case диаграмма — это инструмент для моделирования системы и понимания ее функциональности и потребностей пользователей. Они помогают в определении основных действий, которые пользователь должен совершить в системе, чтобы достичь определенных целей. Они также позволяют определить возможные риски и проблемы, которые могут возникнуть в ходе использования системы. Данная диаграмма представлена на рисунках 5, 6, 7, 8.

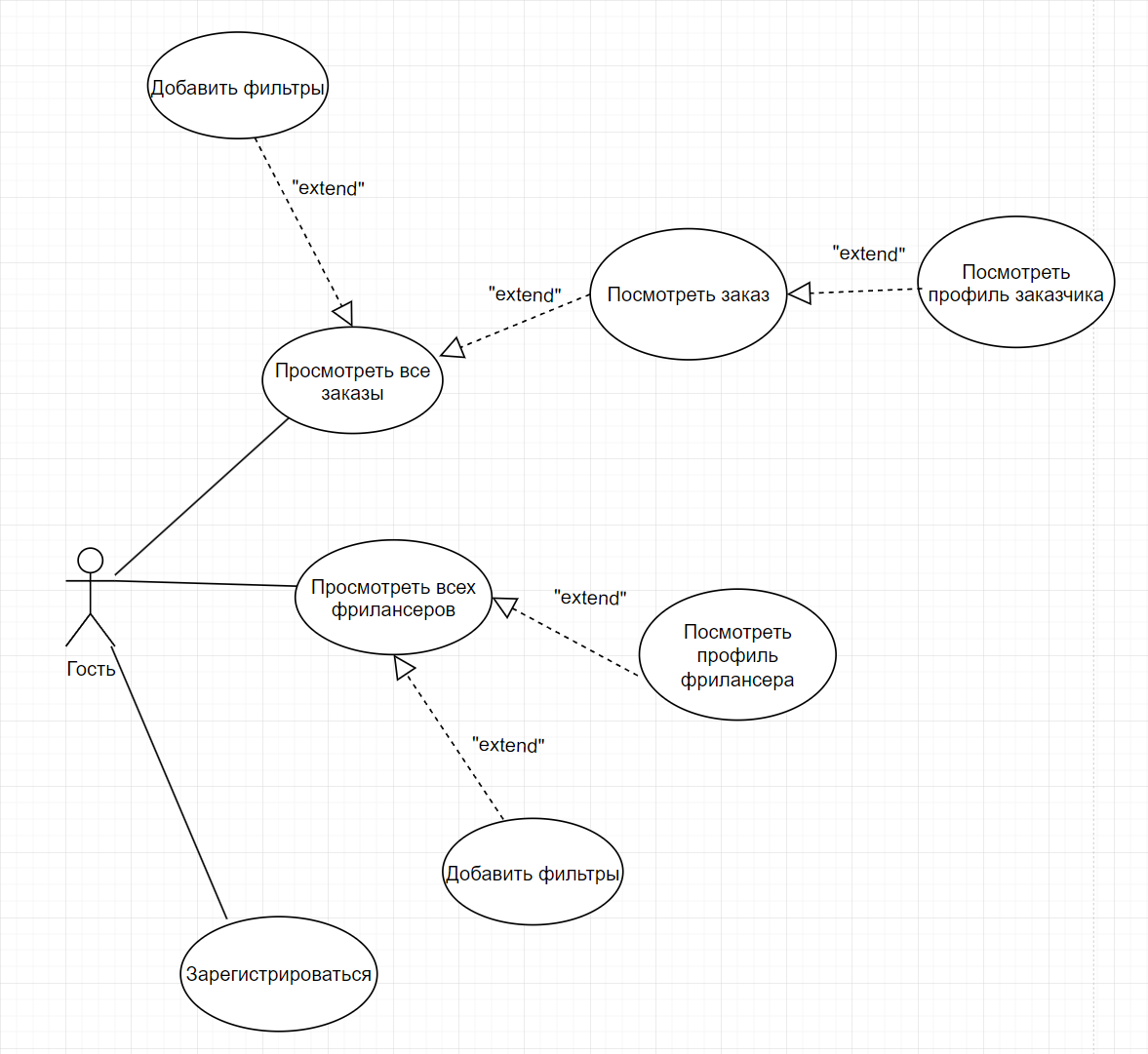


Рисунок 5 - Use-Case диаграмма пользования приложением для гостя

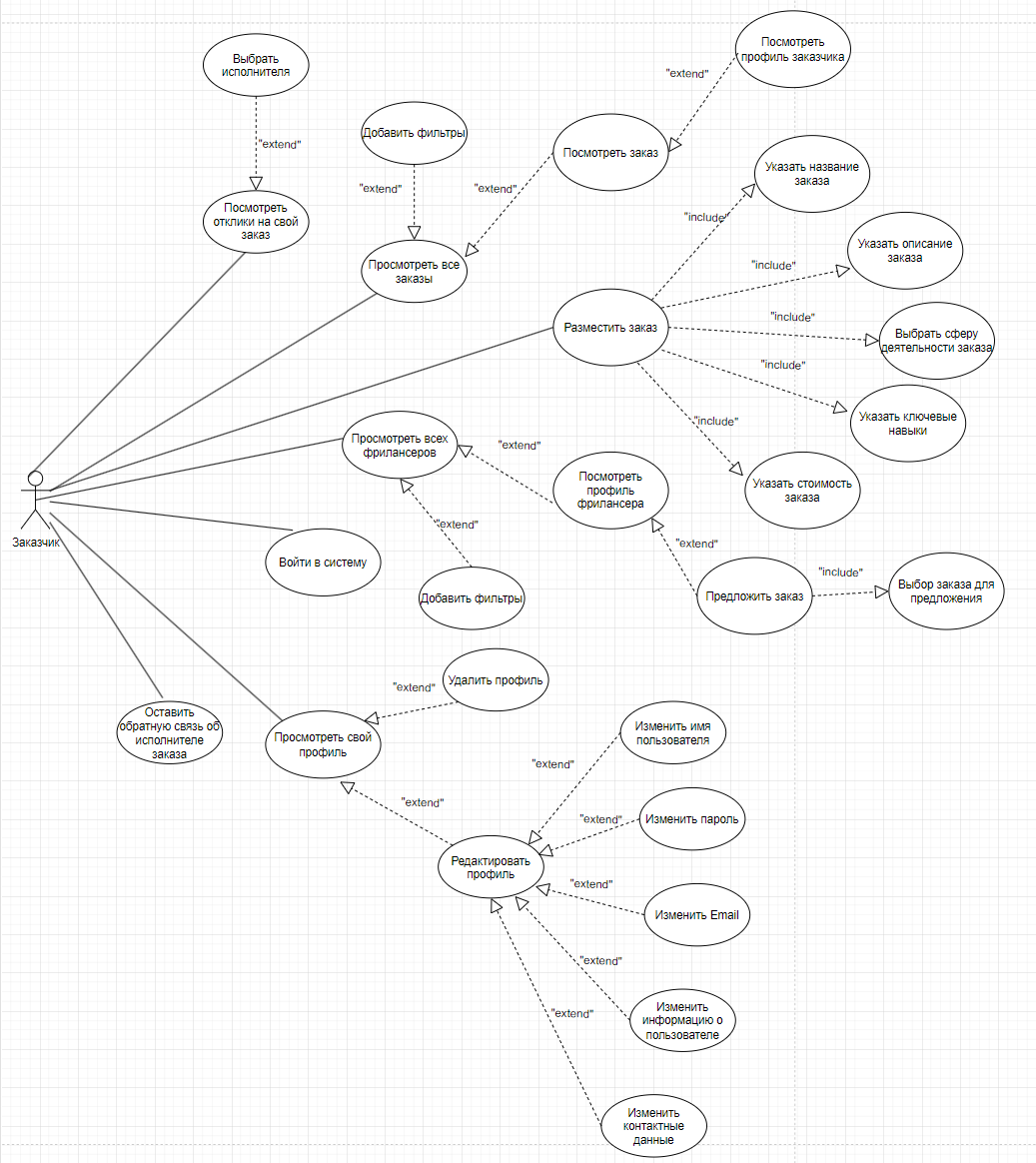


Рисунок 6 - Use-Case диаграмма пользования приложением для заказчика

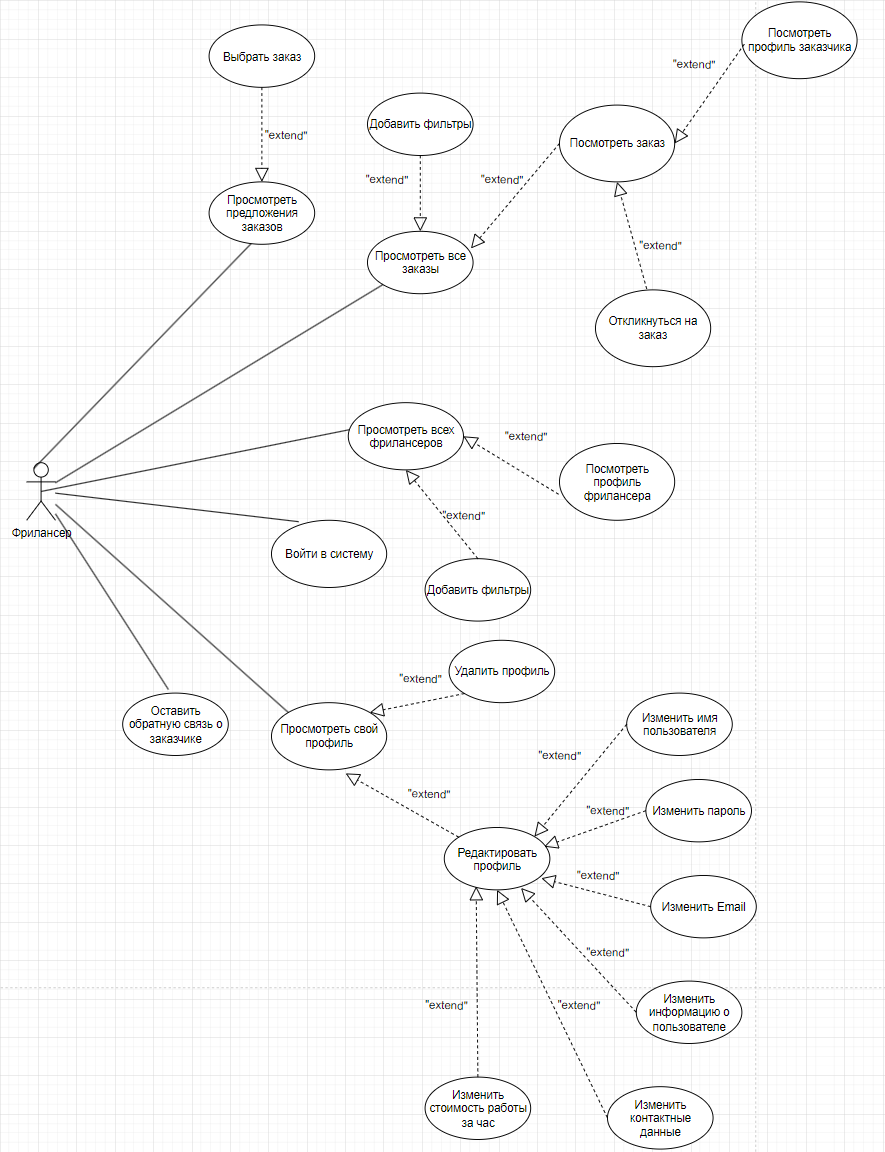


Рисунок 7 - Use-Case диаграмма пользования приложением для фрилансера

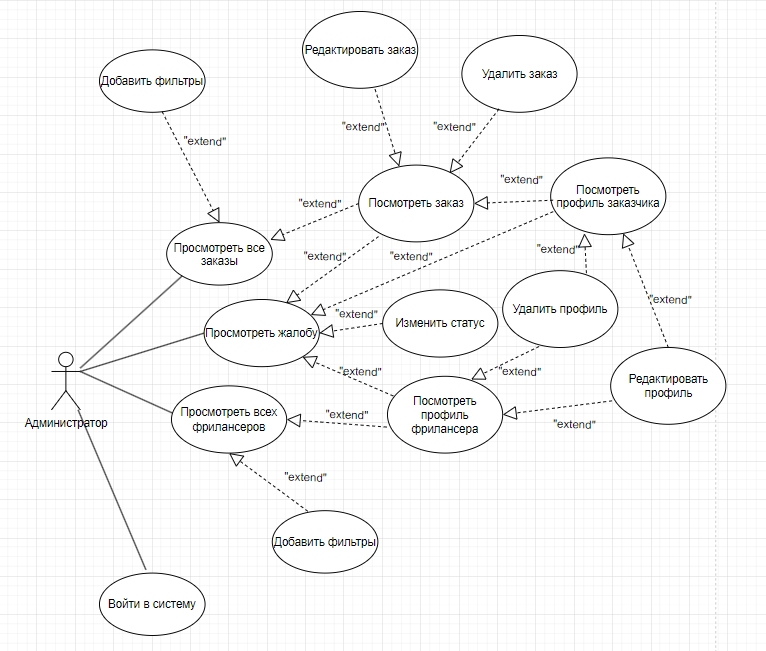


Рисунок 8 - Use-Case диаграмма пользования приложением для

администратора

## Диаграммы последовательности

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

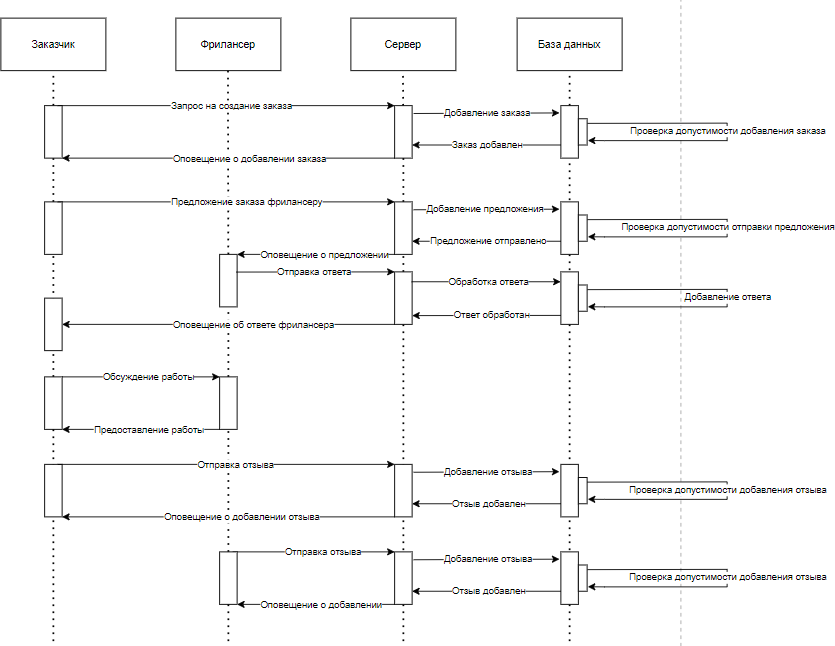


Рисунок 9 - Диаграмма последовательности

## Диаграмма развертывания

Диаграмма развертывания позволяет определить требования к аппаратному обеспечению, планировать установку и настройку компонентов системы, а также оценивать ее производительность и масштабируемость. Данная диаграмма представлена на рисунке 10.

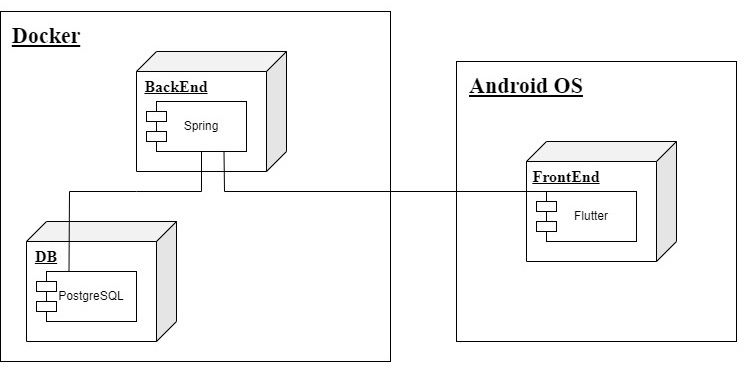


Рисунок 10 - Диаграмма развертывания приложения

## Диаграммы состояния

Диаграмма состояния позволяет определить возможные сценарии поведения системы, выделить ключевые состояния и переходы между ними, а также оценить ее надежность и устойчивость к ошибкам. Для нашего проекта были спроектированы 3 диаграммы для состояний гостя, фрилансера и заказчика. Данные диаграммы представлены на рисунках 11-13.

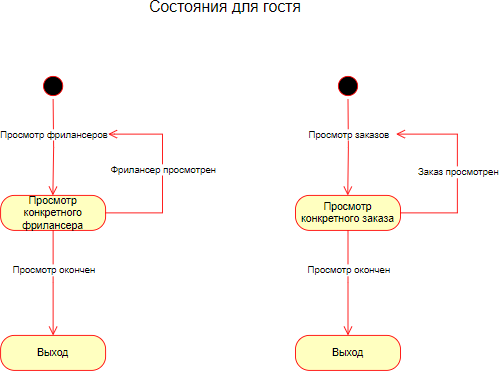


Рисунок 11 - Диаграмма состояния гостя

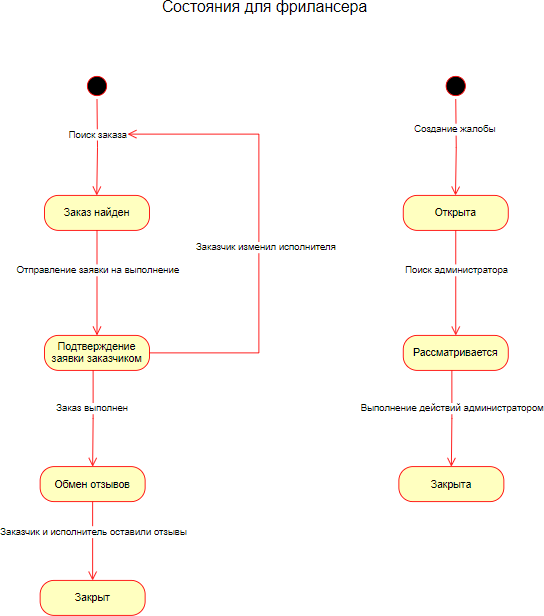


Рисунок 12 - Диаграмма состояния фрилансера

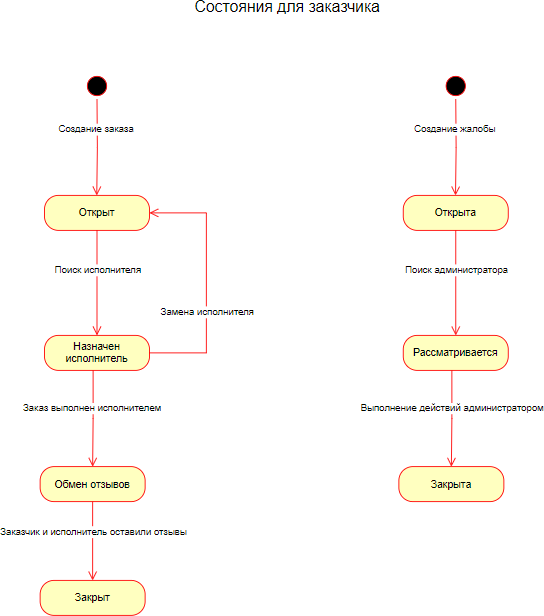


Рисунок 13 - Диаграмма состояния заказчика

## Диаграмма объектов

Диаграмма объектов позволяет определить классы объектов, их атрибуты и методы, а также взаимодействие между ними. Она помогает разработчикам лучше понимать структуру системы и проектировать ее более эффективно. Данная диаграмма представлена на рисунке 14.

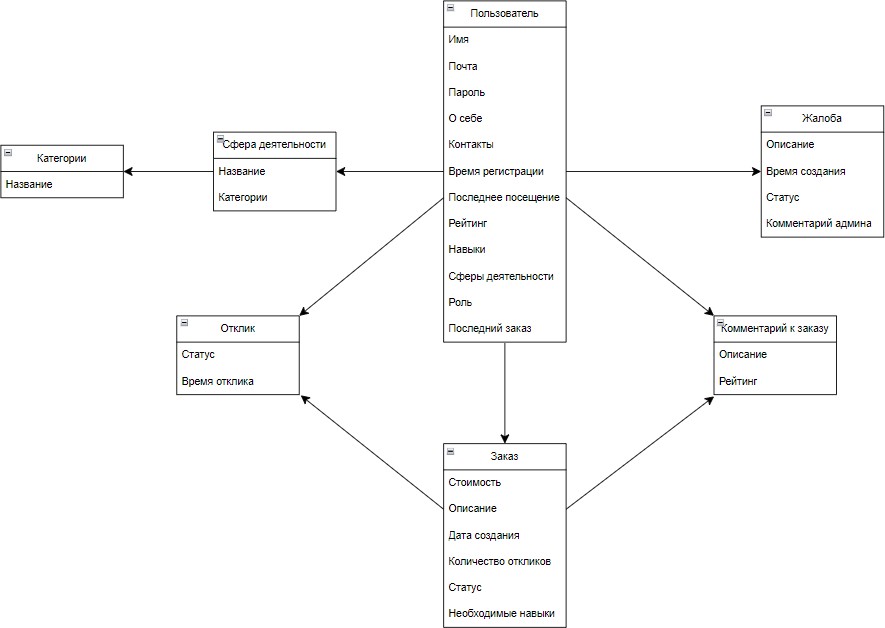


Рисунок 14 - Диаграмма объектов

## Диаграммы активности

Диаграмма активности помогает разработчикам лучше понимать процессы в системе, выявлять узкие места и оптимизировать их. Она также может использоваться для описания бизнес-процессов и управления проектами. Для данного проекта были спроектированы 3 диаграммы активности для гостя, пользователя и администратора. Данные диаграммы представлены на рисунках 15-17.

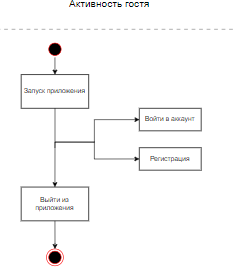


Рисунок 15 - Диаграмма активности гостя

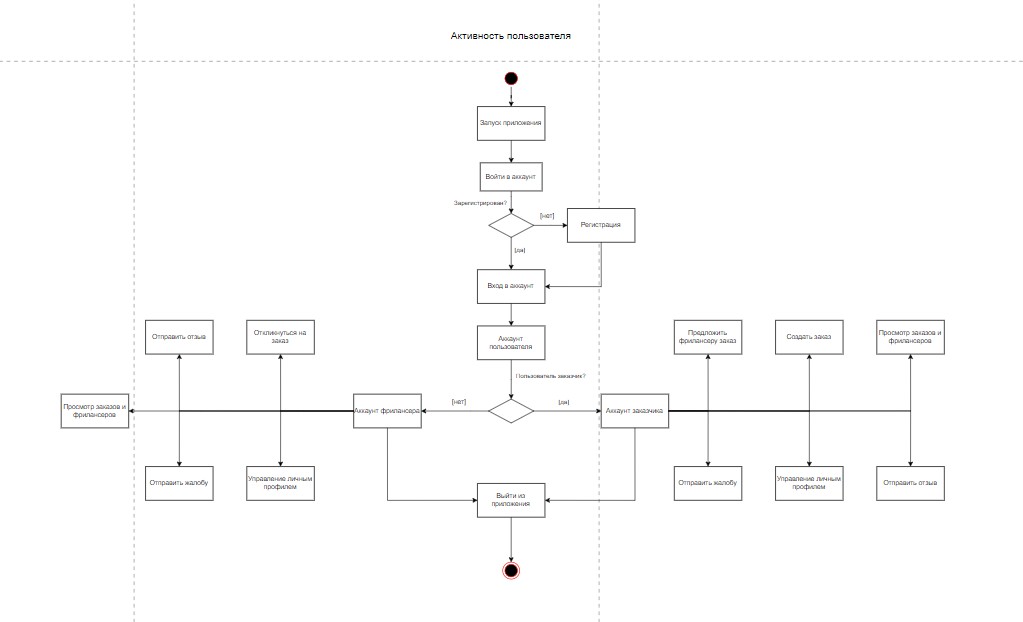


Рисунок 16 - Диаграмма активности пользователя

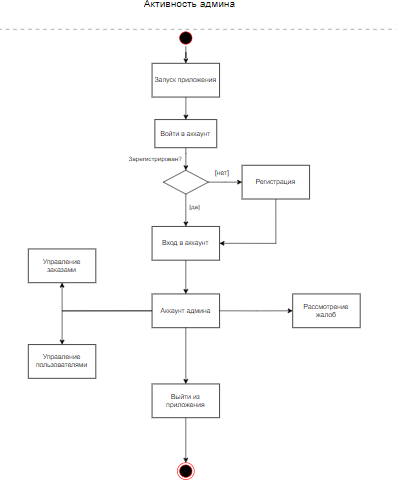


Рисунок 17 - Диаграмма активности администратора

## Реализация

## Средства реализации

Ниже приведен перечень используемых технологий. Backend

* язык программирования: Kotlin 1.6;
* фреймворк: Spring 6.1.4;
* СУБД: PostgreSQL 16.2;
* ORM: Hibernate;
* средство построения REST API: Spring Boot 3.2.3;
* средство авторизации и аутентификации: Spring Security.

Frontend:

* язык программирования: Dart 3.2.6;
* фреймворк: Flutter 3.16.9;
* средство авторизации и аутентификации: JWT. Инструменты для ведения документации:
* Miro;
* YouTrack;
* Figma.

Дополнительный инструментарий:

* GitHub.

## Структура базы данных

В этом разделе курсовой работы будет рассмотрена структура базы данных, использованная для хранения и управления данными в нашей системе. Основное внимание будет уделено объяснению сущностей, их атрибутов и взаимосвязей между ними на основе предоставленной схемы базы данных.

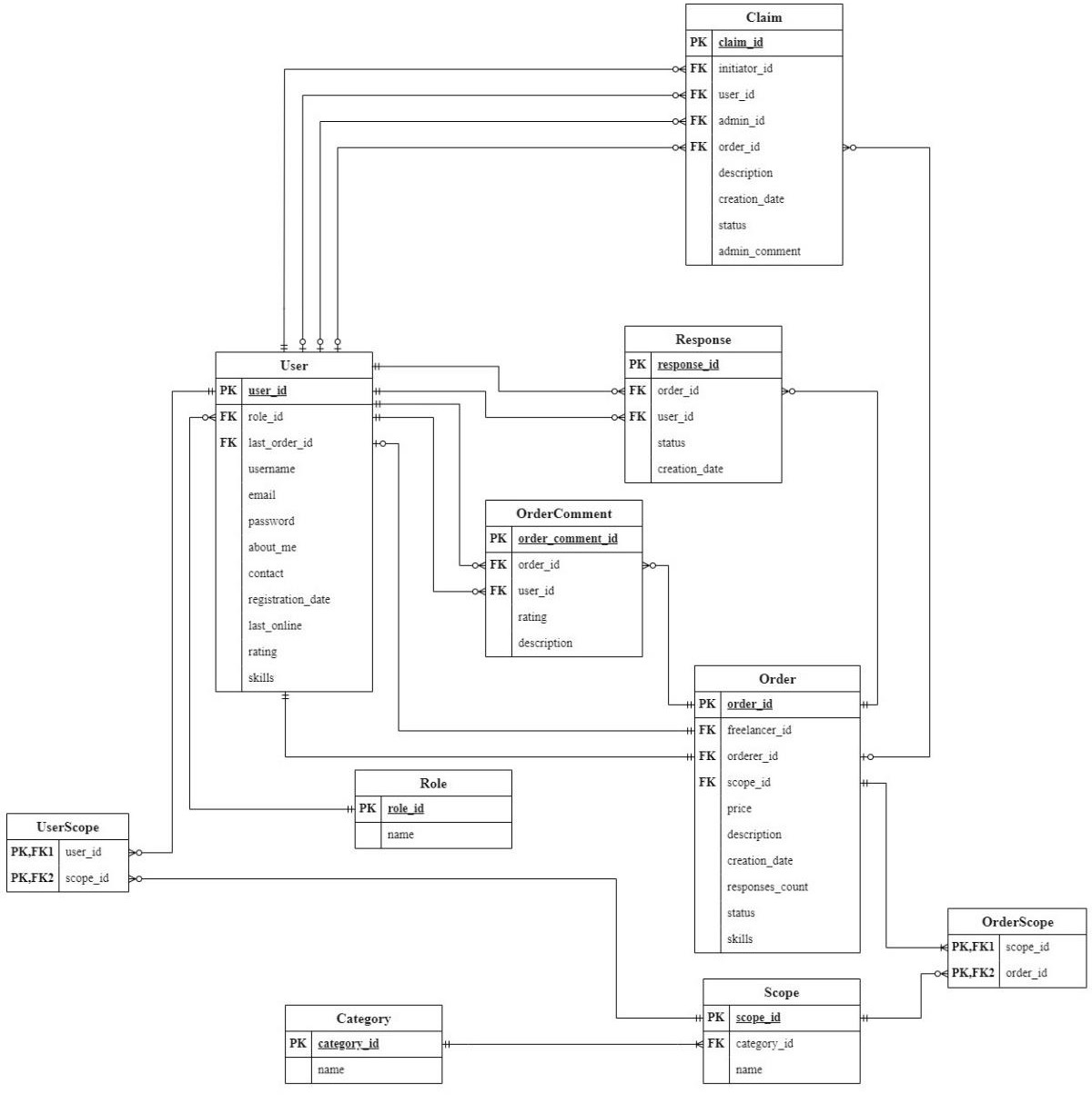


Рисунок 18 - Структура базы данных

Структура базы данных представлена в виде ER-диаграммы, которая включает в себя следующие основные таблицы: User, Claim, Response, Order, OrderComment, Role, Category, Scope, UserScope и OrderScope. Каждая из этих таблиц играет важную роль в обеспечении функциональности системы.

* User хранит информацию о пользователях системы;
* Claim содержит данные о жалобах пользователей на других пользователей или заказы;
* Response используется для подтверждения фрилансера на позицию исполнителя заказа;
* Order управляет данными о заказах;
* OrderComment содержит комментарии к заказам;
* Role хранит роли пользователей в системе;
* Category хранит категории, к которым могут относиться заказы;
* Scope описывает сферы деятельности, к которым могут относиться заказы;
* UserScope связывает пользователей с их сферами деятельности;
* OrderScope связывает заказы с их сферами деятельности.

## Информационная безопасность и защита данных

В современном мире, где киберугрозы становятся все более изощренными, обеспечение безопасности данных является критически важной задачей для любого приложения. В рамках разработки сервиса генерации табличных форм отчетности особое внимание уделено информационной безопасности и защите данных. Для реализации этих аспектов были выбраны такие технологии, как Spring Security и JWT (JSON Web Token).

## SpringSecurity

Spring Security представляет собой мощный и гибкий фреймворк для обеспечения безопасности приложений на базе Spring. Он предоставляет все необходимые инструменты для реализации аутентификации, авторизации и управления доступом. Spring Security позволяет гибко настраивать процесс входа в систему и управление доступом к различным ресурсам приложения, что делает его незаменимым для защиты серверной части сервиса.

Одним из основных преимуществ Spring Security является поддержка множества механизмов аутентификации, включая форму входа, базовую аутентификацию, OAuth2, OpenID и другие. Это позволяет адаптировать процесс аутентификации в зависимости от специфики приложения и

требований безопасности. Важным аспектом является интеграция Spring Security с другими модулями Spring, такими как Spring MVC и Spring Boot, что значительно упрощает его использование и настройку в приложении.

Кроме того, Spring Security предоставляет развитую систему конфигурации, которая позволяет настраивать и расширять функциональность безопасности в соответствии с требованиями конкретного приложения. Фреймворк включает встроенные механизмы защиты от распространенных атак, таких как CSRF (Cross-Site Request Forgery), XSS (Cross-Site Scripting), SQL-инъекции и других, что обеспечивает высокий уровень безопасности. Управление сеансами пользователей, контроль времени их жизни и предотвращение одновременных сеансов — все это помогает повысить общую безопасность системы.



Рисунок 19 - Цепочка фильтров безопасности

Таким образом, Spring Security применяется для реализации надежных механизмов аутентификации и авторизации, защиты данных от несанкционированного доступа и предотвращения распространенных видов атак на веб-приложения.

## JWT

JWT (JSON Web Token) — это стандартный формат для передачи данных между участниками через JSON. Он широко используется для аутентификации и обмена информацией между клиентом и сервером. Одним из ключевых преимуществ JWT является его безопасность. Токены подписываются с использованием секретного ключа или пары ключей (публичного и приватного), что обеспечивает целостность и подлинность данных. Это позволяет убедиться, что данные не были изменены в пути от клиента к серверу.

JWT также является отличным выбором для распределенных систем, так как не требует хранения на сервере. Токены могут быть проверены и валидированы любым сервером, что упрощает масштабирование приложения. JWT представляют собой компактные и автономные токены, которые легко передаются между клиентом и сервером через заголовки HTTP или параметры URL, что делает их использование простым и эффективным.

Гибкость JWT позволяет включать в токен произвольные данные в виде заявлений (claims). Это могут быть данные о пользователе, его правах доступа и другая информация, необходимая для аутентификации и авторизации. При входе пользователя в систему сервер генерирует JWT и отправляет его клиенту. Клиент хранит этот токен и отправляет его вместе с каждым запросом к серверу, что позволяет серверу проверять подлинность запросов и предоставлять доступ к защищенным ресурсам.

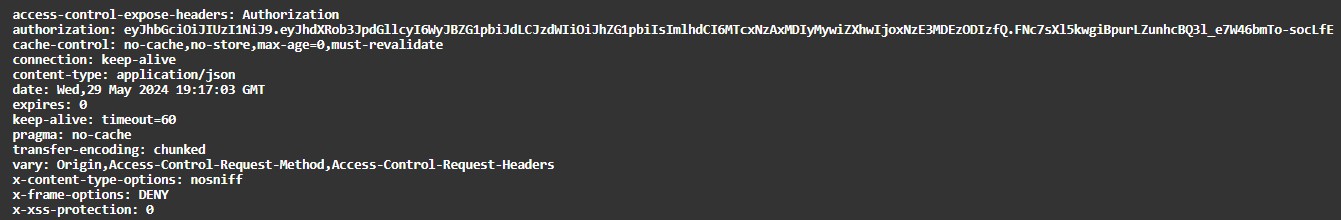


Рисунок 20 - Передача JWT-токена в заголовке

Таким образом, JWT применяется для аутентификации пользователей и передачи информации между клиентом и сервером, обеспечивая безопасный и эффективный механизм аутентификации в распределенных системах.

## Защита данных на серверной стороне

Для обеспечения защиты данных на серверной стороне в приложении FreelanceFinder были реализованы несколько ключевых аспектов безопасности. Эти аспекты включают шифрование данных, управление доступом и мониторинг безопасности.

Шифрование данных: Одним из важных методов защиты данных является их шифрование. В нашей системе используется шифрование данных как в состоянии покоя, так и в состоянии передачи. Для шифрования данных в базе данных применяются алгоритмы AES (Advanced Encryption Standard), которые обеспечивают высокий уровень безопасности. Данные, хранящиеся в базе данных, шифруются с использованием ключей шифрования, что защищает их от несанкционированного доступа в случае компрометации базы данных. Все данные, передаваемые между клиентом и сервером, шифруются с использованием протокола TLS (Transport Layer Security). Это гарантирует, что данные не могут быть перехвачены и изменены во время передачи.

Управление доступом: Spring Security играет ключевую роль в управлении доступом к ресурсам сервера. Пользователи проходят аутентификацию с использованием JWT, что позволяет серверу проверять подлинность пользователя перед предоставлением доступа к защищенным ресурсам. Реализованы роли и права доступа, которые ограничивают доступ к различным частям приложения в зависимости от роли пользователя (например, клиент, фрилансер, администратор). Это позволяет точно контролировать, какие действия может выполнять каждый пользователь.

Мониторинг безопасности: Для обеспечения непрерывного мониторинга и защиты системы используются инструменты мониторинга и

логирования. Реализованы механизмы аудита, которые фиксируют важные действия пользователей, такие как вход в систему и создание заказов. Эти данные помогают в расследовании инцидентов безопасности.

## Защита данных на клиентской стороне

На клиентской стороне приложения также реализованы различные меры безопасности для защиты данных пользователей и обеспечения безопасности взаимодействия с сервером.

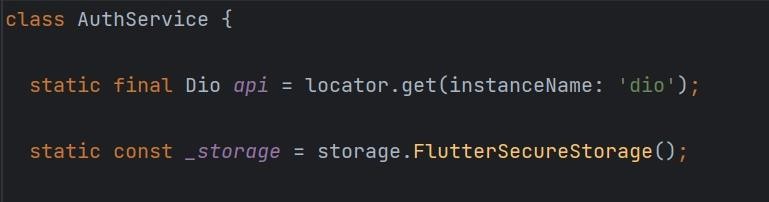


Рисунок 21 - Создание хранилища для JWT-токена

Шифрование и защита данных: Для защиты данных на клиентской стороне используются методы шифрования и безопасного хранения данных. Данные, такие как токены аутентификации и конфиденциальная информация, хранятся на устройстве в зашифрованном виде с использованием библиотеки flutter\_secure\_storage. Это обеспечивает защиту данных от несанкционированного доступа в случае, если устройство пользователя будет скомпрометировано. Токены JWT, используемые для аутентификации, хранятся в безопасном хранилище и защищены от атак, таких как XSS и CSRF. Защита от атак: Flutter-приложение защищено от различных видов атак, которые могут возникнуть на клиентской стороне. Все HTTP-запросы к серверу выполняются с использованием библиотеки http или dio, которая поддерживает безопасные методы передачи данных, такие как TLS. Для

предотвращения атак, связанных с вводом данных, таких как XSS и SQL- инъекции, все пользовательские данные проходят валидацию и очистку перед отправкой на сервер. Валидация осуществляется с использованием встроенных инструментов Flutter и дополнительных библиотек.

Управление сеансами: Для обеспечения безопасности сеансов пользователей реализованы следующие механизмы. JWT токены имеют ограниченный срок действия, после которого пользователю необходимо повторно пройти аутентификацию. Это предотвращает длительное использование токенов в случае их компрометации. В случае обнаружения подозрительной активности или при истечении времени бездействия, сеанс пользователя автоматически завершается, требуя повторного входа в систему.

## Примеры дизайна и его функциональное назначение

Ниже приведены примеры дизайна и функционала некоторых ключевых экранов в приложении FreelanceFinder. Остальные экраны приложения FreelanceFinder реализованы в соответствии с техническим заданием и функциональными требованиями, обеспечивая полный спектр возможностей для всех пользователей.

## Экран входа в систему

Экран входа в систему предназначен для аутентификации пользователей. Он обеспечивает доступ зарегистрированных пользователей к функционалу приложения, сохраняя при этом безопасность и конфиденциальность данных.

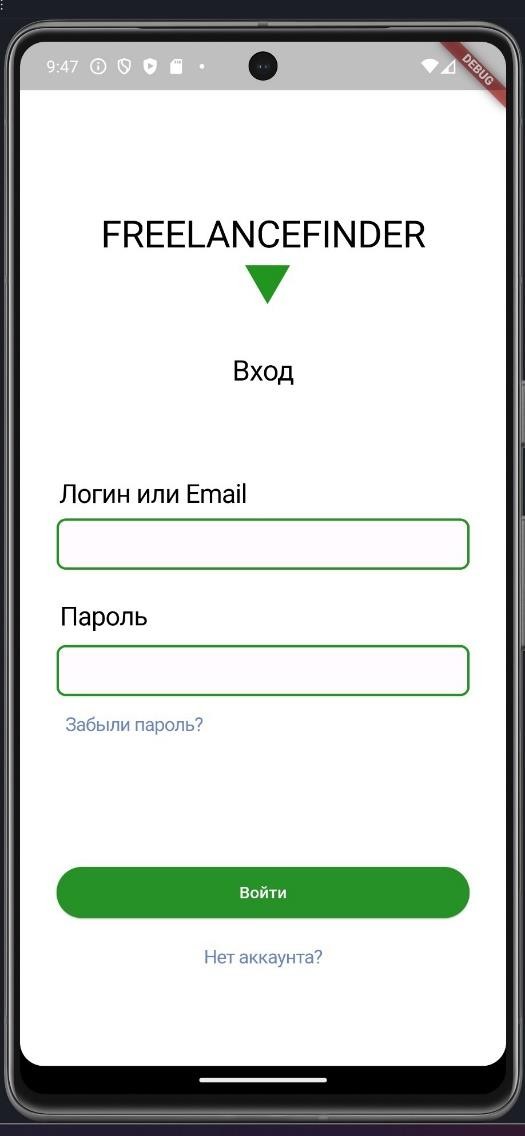


Рисунок 22 - Экран входа в систему

Форма ввода данных: Экран содержит поля для ввода адреса электронной почты (или логина) и пароля.

Кнопка «Забыли пароль?»: Опция для восстановления пароля через электронную почту.

Кнопка «Войти»: Кнопка для подтверждения ввода данных. При успешном прохождении аутентификации на приложение с сервера в ответе приходит JWT-токен.

Кнопка «Нет аккаунта?»: Ссылка для перехода на экран регистрации, если у пользователя еще нет учетной записи.

## Главный экран заказов

Главный экран заказов предназначен для отображения всех доступных заказов. Пользователи могут просматривать, фильтровать и выбирать заказы, соответствующие их навыкам и интересам.

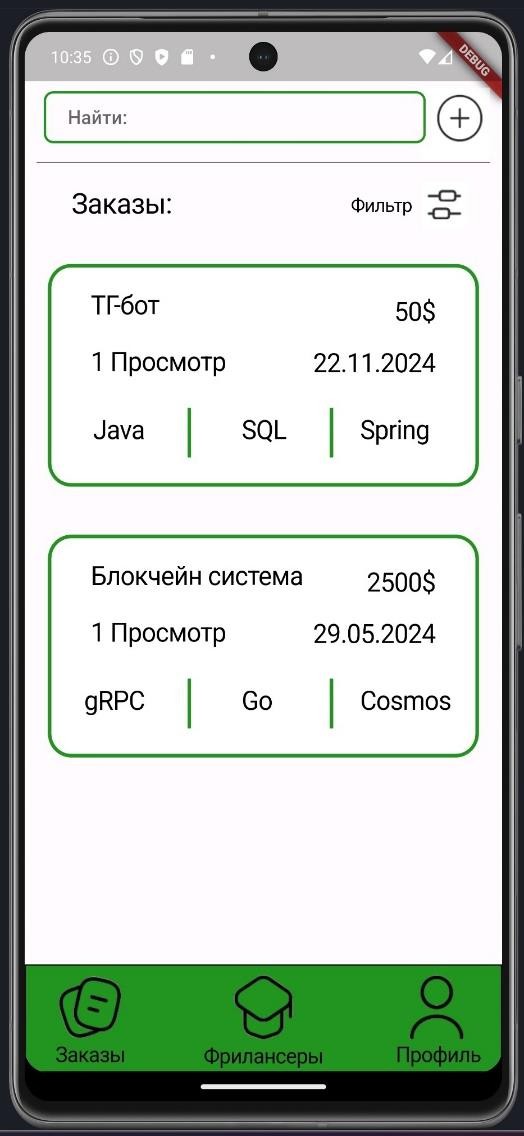


Рисунок 23 - Главный экран заказов

Поле «Найти»: Строка поиска для быстрого нахождения заказов по ключевым словам.

Кнопка создания заказа: Возможность для клиентов создать новый заказ. Фильтры: Панель с фильтрами по категориям и опции сортировки.

Список заказов: Карточки с краткой информацией о заказах (название, требуемые навыки, бюджет, дата создания заказа). При нажатии на карточку происходит переход на страницу заказа.

## Экран заказа

Экран заказа предоставляет детальную информацию о конкретном заказе. Он позволяет пользователям изучить все условия и требования, а также оставить заявку на выполнение заказа.

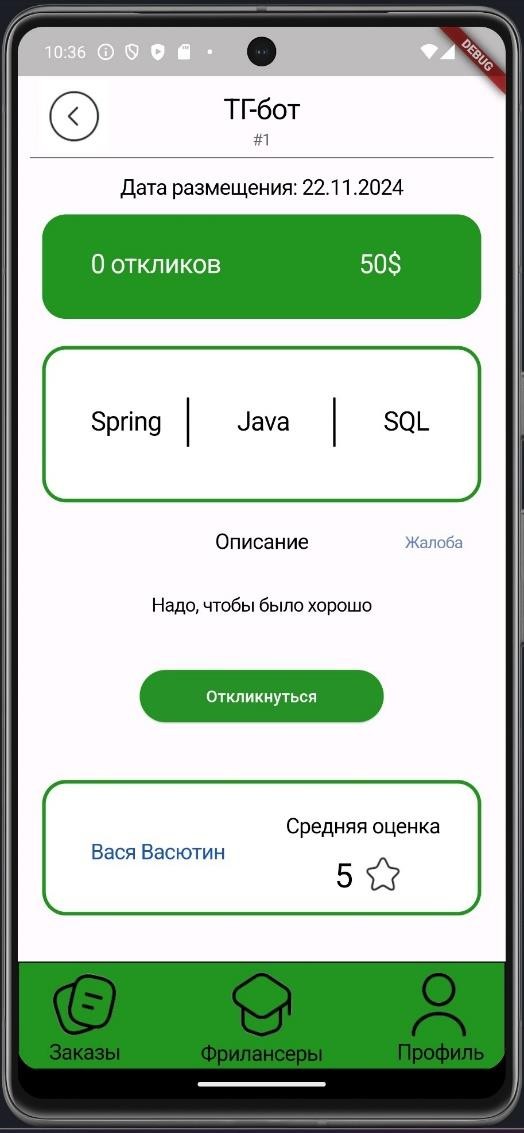


Рисунок 24 - Экран заказа

Информация о заказе: Название заказа, количество откликов и сумма оплаты за выполнение.

Поле с навыками: Область с навыками необходимыми для выполнения заказа.

Описание заказа: Полное описание заказа.

Кнопка «Жалоба»: Кнопка, позволяющая отправить жалобу на заказ или заказчика.

Кнопка «Откликнуться»: Опция для подачи заявки на выполнение заказа.

Информация о заказчике: Имя и рейтинг заказчика.

## Экран обращений

Экран обращений служит для управления всеми жалобами и просьбами пользователей к администрации. Он обеспечивает удобный доступ к истории обращений и их статусу, что позволяет администрации оперативно реагировать и помогать клиентам.



Рисунок 25 - Экран обращений

Список обращений: Перечень всех не рассмотренных обращений с указанием имени пользователя, краткого текста последнего жалобы и ее номер в списке. При нажатии на карточку перенаправляет к экрану конкретной жалобы.

## Аналитика

Для сбора метрик для нашего приложения фриланс биржи было выбрано AppMetrica от Яндекс. Это система для сбора данных об использовании приложения пользователями, которая отличается высокой скоростью и удобством настройки метрик для мобильных приложений. Кроме того, она предлагает интуитивный интерфейс и понятное руководство по использованию.

Были составлены четыре воронки конверсии:

* общая статистика;
* аккаунт;
* заказы;
* рейтинг.

Воронка «Общая статистика» необходима для просмотра информации о количестве скачиваний приложения, количества запуска приложения, а также регистрации и авторизации пользователей.

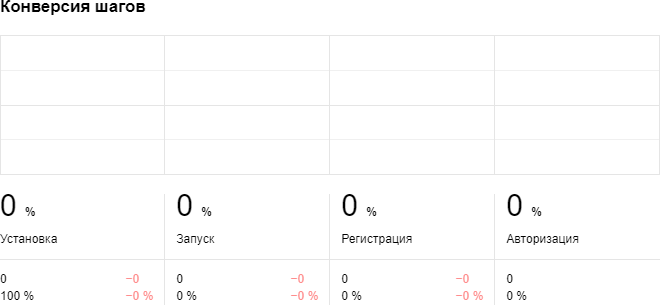


Рисунок 26 - Воронка «Общая статистика»

Воронка «Аккаунт» необходима для просмотра статистики по количеству входа в аккаунт и выхода из него, а также изменению профиля и настроек аккаунта.



Рисунок 27 - Воронка «Аккаунт»

Воронка «Заказы» необходима для отслеживания и анализа статистики, связанной с процессом создания и выполнения заказов, включая количество созданных заказов, количество заявок на заказы и количество завершенных заказов.



Рисунок 28 - «Заказы»

Воронка «Рейтинг» необходима для отслеживания статистики по рейтингу фрилансеров и заказчиков, включая количество отзывов и оценок, а также изменения рейтинга фрилансеров.

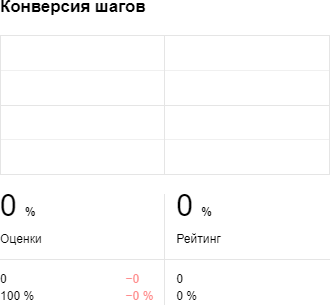


Рисунок 29 - Воронка «Рейтинг»

Яндекс Метрики предоставляет нам ценную информацию о поведении пользователей в нашем приложении, которая помогает нам оптимизировать работу приложения и улучшать пользовательский опыт.

# Заключение

В результате выполнения данного курсового проекта был проведен тщательный анализ предметной области и изучены существующие аналоги разрабатываемого приложения. На основе полученных данных были разработаны функциональные и нефункциональные требования к приложению.

Для визуализации будущего приложения были созданы макеты интерфейса, отражающие основные элементы дизайна и взаимодействия с пользователем. Была выбрана платформа для разработки приложения, которая обеспечивает наилучшую производительность и масштабируемость.

Для обеспечения эффективного управления проектом и контроля версий был создан репозиторий GitHub. Были построены UML диаграммы, отражающие структуру и взаимосвязи элементов приложения.

В ходе разработки были реализованы основные функции приложения.

В целом, данный курсовой проект позволил получить ценный опыт в разработке мобильных приложений, а также заложить основу для дальнейшего развития и улучшения разрабатываемого приложения.

# Список использованных источников

1. Котлин в действии: Д.Б. Жемеров, С.С. Исакова ; пер. с англ. А.Н. Киселева ; ред. Д.А. Мовчан. – М.: ДМК-Пресс, 2018. – 448 с.
2. The Docker Book: Containerization is the new virtualization Kindle Edition / James Turnbull. - James Turnbull, 2014. – 388 с.
3. Spring Framework Documentation [Электронный ресурс]. – Режим доступа: https://spring.io/projects/spring-boot. – Spring Boot. – (Дата обращения 14.04.2024).
4. Официальная документация Yandex Metrica [Электронный ресурс]. – Режим доступа: https://metrica.yandex.com/about? – Заглавие с экрана. – (Дата обращения: 20.04.2024).
5. Spring Android [электронный ресурс] – Режим доступа: https://spring.io/guides/gs/consuming-rest-android/ – Заглавие с экрана. – (дата обращения: 21.04.2022).