МИНОБРНАУКИ РОССИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
“ВОРОНЕЖСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ”

Факультет компьютерных наук

Кафедра программирования и информационных технологий

Сервис для поиска и аренды оборудования и инструментов для домашнего ремонта и строительства

Курсовой проект

09.03.04 Программная инженерия

Информационные системы и сетевые технологии

Зав. кафедрой _____С.Д. Махортов, д. ф.-м. н., профессор __._._.20__

Обучающийся _____В.Н. Ремезов, 3 курс, д/о

Обучающийся _____Д.В. Бучнев 3 курс, д/о

Обучающийся _____Е.А. Клоков, 3 курс, д/о

Руководитель _____В.С. Тарасов, ст. преподаватель

Воронеж 2024

Содержание

Введение

В современном мире, где развитие технологий и урбанизация приводят к увеличению темпов строительства и ремонта, становится все более актуальной проблема доступности специализированного оборудования и инструментов для домашних мастеров и профессионалов. Не каждый человек имеет возможность приобрести дорогостоящее оборудование для разовых или нерегулярных работ. В этом контексте особое значение приобретает создание сервисов, предоставляющих услуги по аренде оборудования и инструментов для ремонта и строительства.

В работе будет рассмотрен опыт существующих аналогичных платформ, проведен анализ рынка аренды оборудования, выявлены ключевые требования пользователей и основные технические и организационные аспекты создания сервиса. Итогом исследования станет разработка сервиса аренды, способного удовлетворить потребности как частных лиц, так и профессионалов в области строительства и ремонта.

Разработка такого сервиса отвечает современным тенденциям цифровизации и экономии ресурсов, что делает данную тему актуальной и востребованной.

1. Постановка задачи

1.1. Цели проекта

Целью курсового проекта является создание мобильного приложения для поиска и аренды оборудования и инструментов «RenTool». Данное мобильное приложение разрабатывается с целями:

— предоставить пользователям возможность арендовать инструмент;

— сокращение простоя оборудования;

— расширение клиентской базы.

1.2. Задачи приложения

Приложение решает следующие задачи:

— упрощение процесса аренды инструментов;

— обеспечение экономической выгоды аренды по сравнению с покупкой целевого инструмента для пользователя;

— устранение необходимости для пользователя в хранении крупных инструментов.

1.3. Требования к разрабатываемой системе

1.3.1. Функциональные требования

К разрабатываемой системе выдвигаются следующие функциональные требования:

Анонимный пользователь:

— поиск инструментов;

— фильтрация инструментов;

— просмотр информации о конкретном инструменте;

— регистрация в системе;

— авторизация в системе.

Пользователь, прошедший авторизацию:

— обладает всеми функциональными возможностями анонимного пользователя, за исключением авторизации и регистрации;

— запись заявки на оформление аренды;

— изменение статуса заказа;

— предварительный отказ от арендного оборудования;

— продление арендного оборудования;

— выход из системы.

Администратор системы:

— добавление нового инструмента в базу данных;

— удаление инструмента из базы данных;

— обновление информации об инструменте;

— изменение статуса инструмента.

1.3.2. Требования к оформлению мобильного приложения

Мобильное приложение должно быть оформлен в единой цветовой палитре. У экранов приложения должен быть единый стиль. В мобильном приложении должен присутствовать разработанный логотип. Основной шрифт используемый в мобильном приложении – Montseratt.

1.4. Задачи, решаемые в процессе разработки

Были поставлены следующие задачи:

— анализ предметной области;

— исследование аналогов;

— создание UML диаграмм;

— создание макетов для мобильного приложения;

— написание технического задания;

— создание дизайна для мобильного приложения;

— разработка приложения в соответствии с техническим заданием.

2. Анализ предметной области

2.1. Терминология (глоссарий) предметной области

API – набор способов и правил, по которым различные программы общаются между собой и обмениваются данными.

API Gateway – шаблон проектирования, реализующий единую точку входа в микросервисную архитектуру.

Flutter – бесплатный и открытый набор средств разработки мобильного пользовательского интерфейса, созданный компанией Google и выпущенный в мае 2017 года.

REST API — архитектурный стиль веб-служб, который использует протокол HTTP для передачи данных между клиентом и сервером.

Микросервис – подход к разработке программного обеспечения, при котором приложение строится как набор небольших, автономных, самодостаточных сервисов, каждый из которых решает отдельную бизнес-задачу или функциональность.

Фреймворк – структура, предоставляющая базовый функционал для разработки программного обеспечения. Он представляет собой набор библиотек, инструментов и рекомендаций, которые помогают разработчикам создавать приложения, не начиная с нуля.

2.2. Актуальность

Аренда инструментов для домашнего ремонта – это особенно актуальное и удобное решение в условиях современного динамичного рынка строительства и ремонта. С учетом растущей конкуренции и повышенного общественного интереса к улучшению жилищных условий, спрос на услуги по аренде инструментов непрерывно увеличивается.

Одним из ключевых преимуществ аренды инструментов для ремонта дома является доступность широкого ассортимента инструментов без необходимости значительных финансовых вложений. Вместо того чтобы покупать дорогостоящее оборудование, которое может использоваться лишь время от времени, клиенты могут арендовать необходимые инструменты на нужный период времени.

Удобство также играет важную роль в выборе аренды инструментов. Для временных ремонтных работ или проектов, требующих специализированных инструментов, аренда представляется более привлекательным вариантом по сравнению с их приобретением. Это также сокращает необходимость хранения большого количества инструментов после завершения работ, что освобождает пространство и упрощает хранение.

Более того, арендные компании часто предоставляют своим клиентам доступ к самому современному оборудованию, регулярно обновляя свой список инструментов. Это позволяет клиентам использовать самые передовые технологии и значительно повышает эффективность проводимых работ.

Аренда инструментов для ремонта является не только удобным, но и экономически выгодным решением для широкого круга клиентов, помогая им успешно осуществить различные строительные и ремонтные проекты без значительных финансовых вложений. Это открывает новые возможности для тех, кто хочет обновить свой дом, но не желает замораживать большие суммы на покупку инструментов.

2.3. Обзор аналогов

Изучение аналогичных услуг аренды инструментов представляет собой важный этап в разработке платформы. Этот анализ поможет глубже понять потребности и предпочтения целевой аудитории, что в свою очередь позволит лучше определить концепцию, цели и функциональные задачи нашего сервиса. Полученная информация поможет разработать более удобный и

эффективный в использовании сервис аренды инструментов, который полностью удовлетворит потребности пользователей.

В ходе исследования были рассмотрены 7 наиболее интересных продуктов, которые могут представлять для нашего сервиса, как прямую, так и косвенную конкуренцию. На основании этих собранных данных, были сформированы таблицы, которые позволяют оценить интересующие категории исследования и свести воедино все полученные данные для итогового анализа.

Таблица 1 - Анализ мобильных приложений конкурентов

Параметры/категории (наличие)	Стройшеринг	Арентум	YouTool	Складно
Поиск и фильтрация	Да	Да	Да	Да
Категория «Избранное»	Да	Нет	Нет	Нет
Отмена/продление аренды	Да	Да	Да	Нет
Информация о наличии инструментов	Нет	Нет	Да	Нет
Цена (аренда шуруповерта в рублях/сутки)	350	500	150	594
История аренды	Да	Нет	Нет	Да
Фото/описание инструментов	Да	Да	Да	Да

Таблица 2 - Анализ сайтов конкурентов

Параметры/категории (наличие)	ВиРент	Прокат36	Бригадир
Поиск и фильтрация	Да	Да	Да
Категория «Избранное»	Да	Нет	Нет
Отмена/продление аренды	Да	Да	Да
Информация наличии инструментов	Да	Да	Да
Цена (аренда шуруповерта в рублях/сутки)	400	280	400(+2000 залог)
История аренды	Да	Нет	Нет
Фото/описание инструментов	Да	Да	Да

2.3.1. ВиРент

ВиРент – это дочерняя компания «ВсеИнструменты.ру», которая предлагает выгодную аренду инструментов и оборудования. ВиРент предлагает широкий выбор моделей инструментов, а так же берет на себя обязательства по ремонту и доставке. Клиенты могут в короткий срок оформить заявку и в течении дня инструмент будет доставлен.

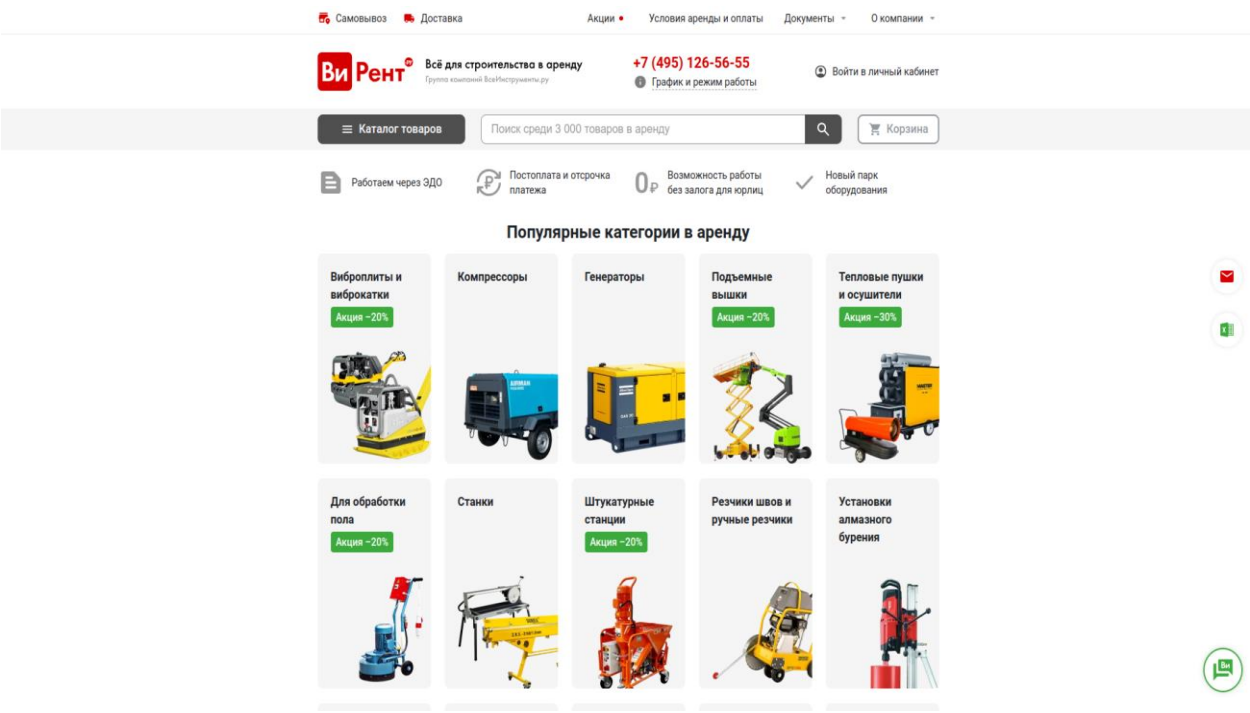

Рисунок 1 – Интерфейс ВиРент

Достоинства:

— большой выбор моделей оборудования и инструментов;

— быстрая и бесплатная доставка оборудования любых габаритов;

— выездное обслуживание;

— скидки в зависимости от срока аренды;

— частые акции и скидки;

— работает по всей России;

— современное оборудование.

Недостатки:

— более высокие цены в сравнении с другими компаниями;

— для физических лиц, помимо оплаты услуги аренды, необходимо внести залог.

2.3.2. Прокат36

Прокат36 – это сервис по аренде инструментов работающий в городе Воронеж. Компания специализируется на аренде и ремонте строительного инструмента и оборудования.

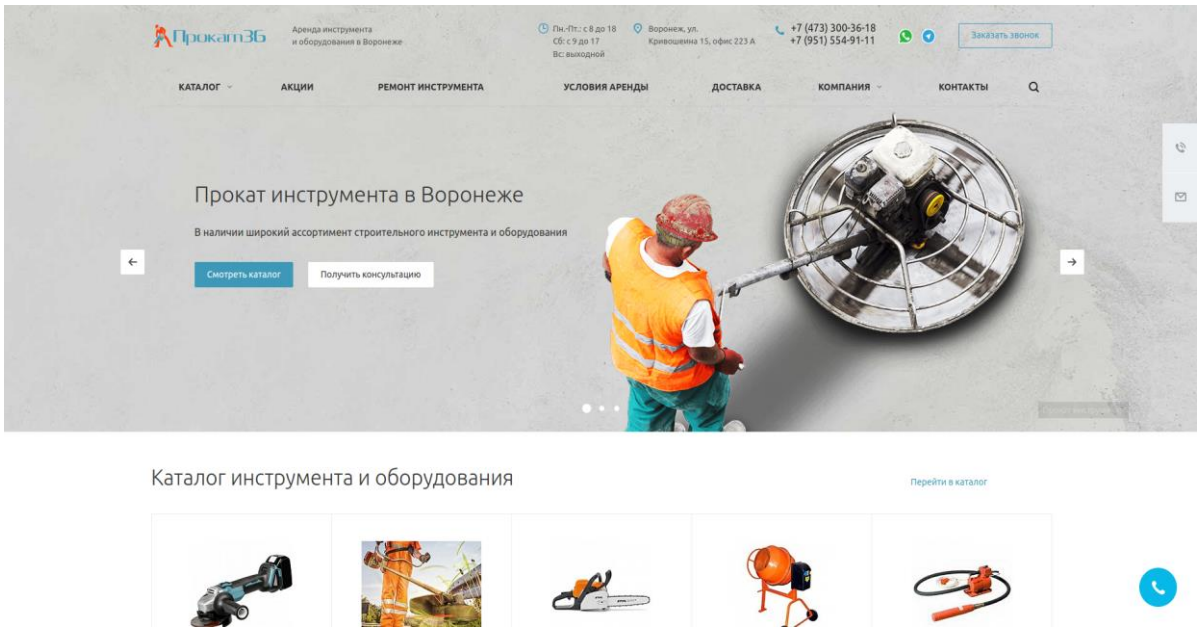

Рисунок 2 – Интерфейс Прокат36

Достоинства:

— низкие цены и специальные тарифы для постоянных клиентов;

— возможность выдачи инструмента в нерабочее время;

— помимо ремонта и обслуживания оборудования предоставляемого в аренду, занимаются ремонтом клиентского оборудования;

— постоянные скидки в выходные дни 50%.

Недостатки:

— нет конкретных сроков, когда свяжутся с клиентом после отправки заявки;

— нет возможности настроить заранее требуемый срок аренды, необходимо обсуждать это с работником компании;

— небольшой выбор инструментов.

2.3.3. Бригадир

Бригадир – это сервис по аренде инструментов работающий в городе Воронеж. Данное приложение предоставляет возможность арендовать инструмент посуточно.

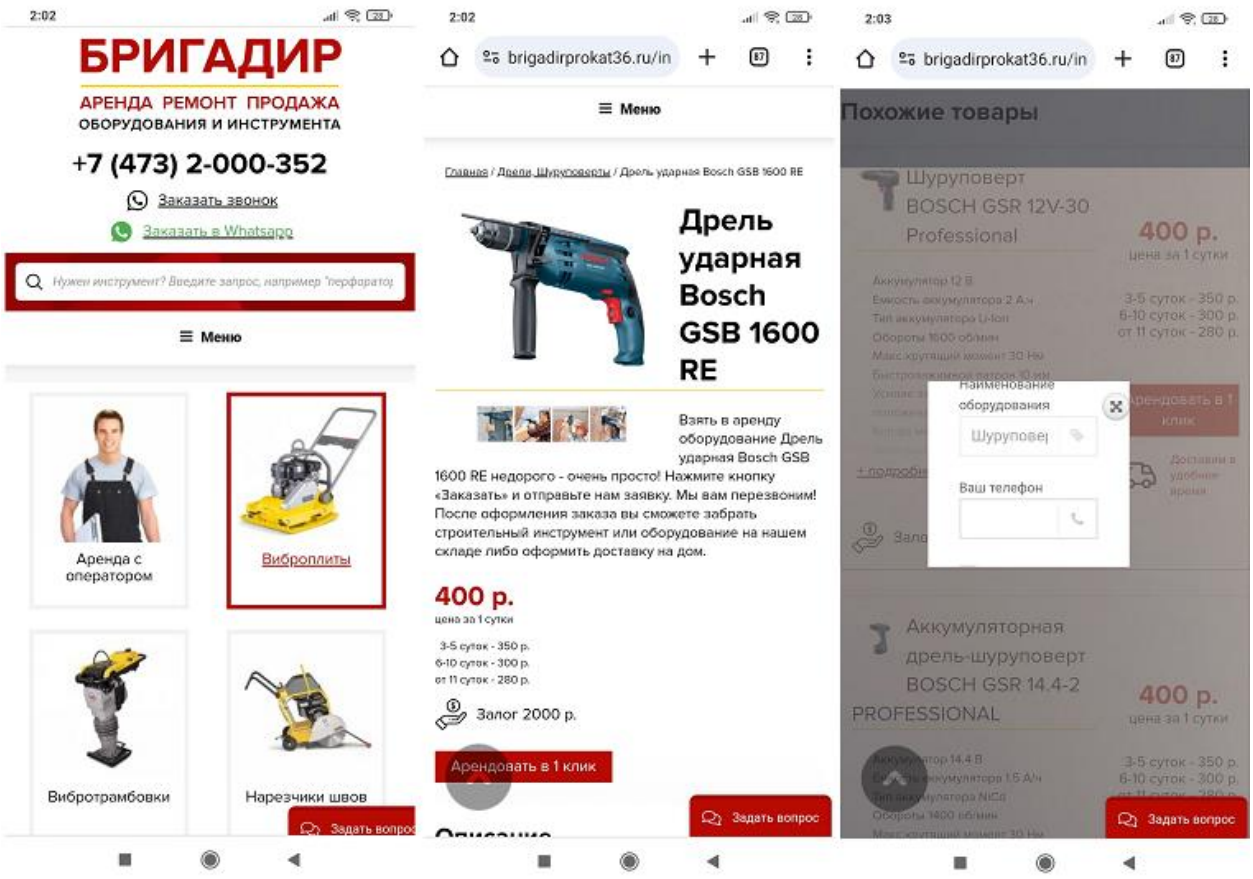

Рисунок 3 – Интерфейс Бригадир

Достоинства:

— возможность аренды с оператором;

— доставка в любое удобное клиенту время;

— диагностика и ремонт инструмента круглосуточно;

— акция на вечерний прокат инструментов и оборудования;

— чем больше суток аренды тем меньше итоговая цена.

Недостатки:

— нет конкретных сроков, когда свяжутся с клиентом после отправки заявки;

— нет возможности настроить заранее требуемый срок аренды, необходимо обсуждать это с работником компании;

— за оборудование вносится возвратно-денежный залог. Для лиц без прописки в Воронеже или близлежащих населенных пунктах, залог может быть увеличен;

— при отсутствии данных о месте проживания арендатора оборудование может быть выдано под залог, равный его полной стоимости.

2.3.4. Стройшеринг

Стройшеринг – это приложение для аренды строительного инструмента. Приложение предоставляет прокат и аренду инструмента по всей России.

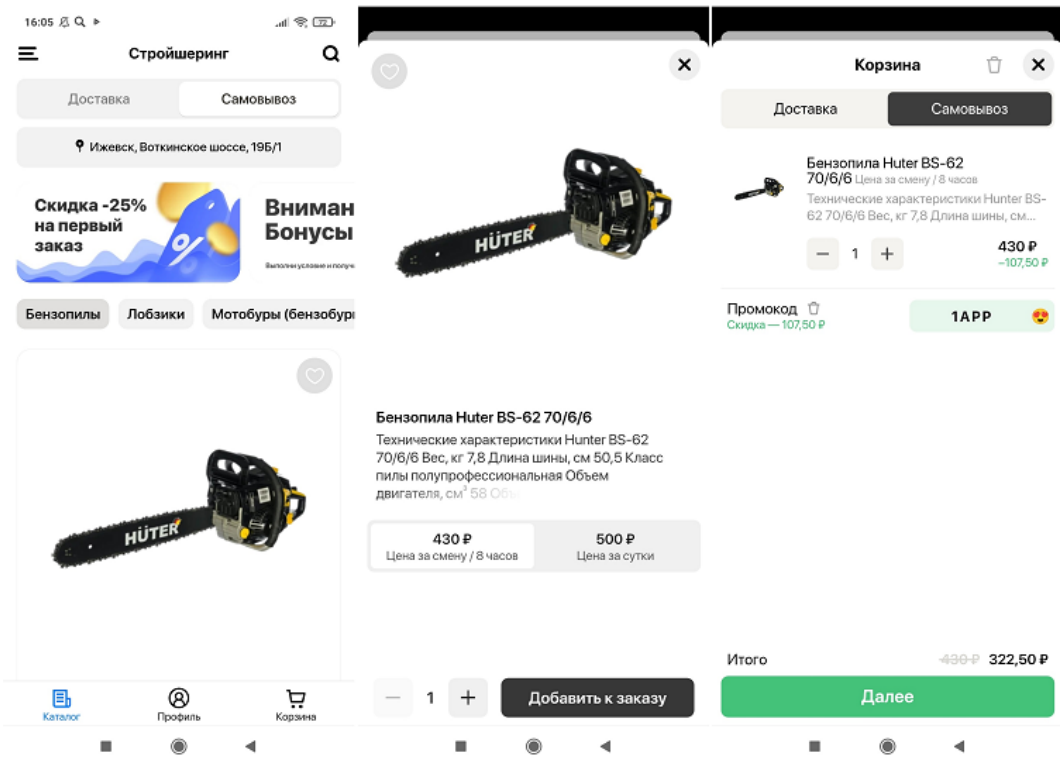

Рисунок 4 – Интерфейс Стройшеринг

Достоинства:

— система промокодов;

— четыре формы оплаты;

— возможность приоритетной доставки за дополнительную плату;

— получение подарков и кэшбеков с заказов. Оплата заказа бонусными баллами.

Недостатки:

— срок аренды можно выставить только на одни сутки или на восемь часов;

— нельзя указать время доставки инструмента.

2.3.5. Арентум

Арентум – это сервис, который предоставляет возможность аренды инструмента и проката оборудования в городе Самара. Сервис сдает в аренду все распространенные виды строительно-монтажной техники.

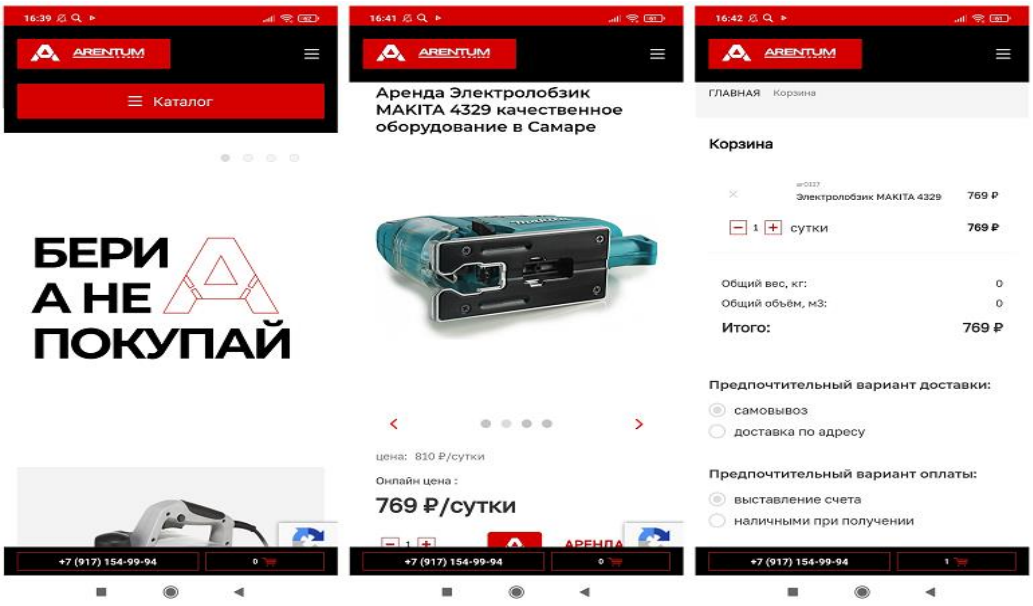

Рисунок 5 – Интерфейс Арентум

Достоинства:

— широкий выбор инструмента и оборудования.

Недостатки:

— функция для выбора нескольких единиц конкретного инструмента работает некорректно;

— нельзя указать время доставки инструмента;

— множество багов при попытке сделать заказ.

2.3.6. YouTool

YouTool – это сервис по прокату инструментов и строительной техники, работающий в 10 крупных городах России. Платформа позволяет арендовать или сдавать в аренду, отремонтировать и продать инструмент с прозрачной историей владения. Платформа дорабатывается (локализация) для экспансии Северной Америки. Дата выхода в Канаду – 2022 год.

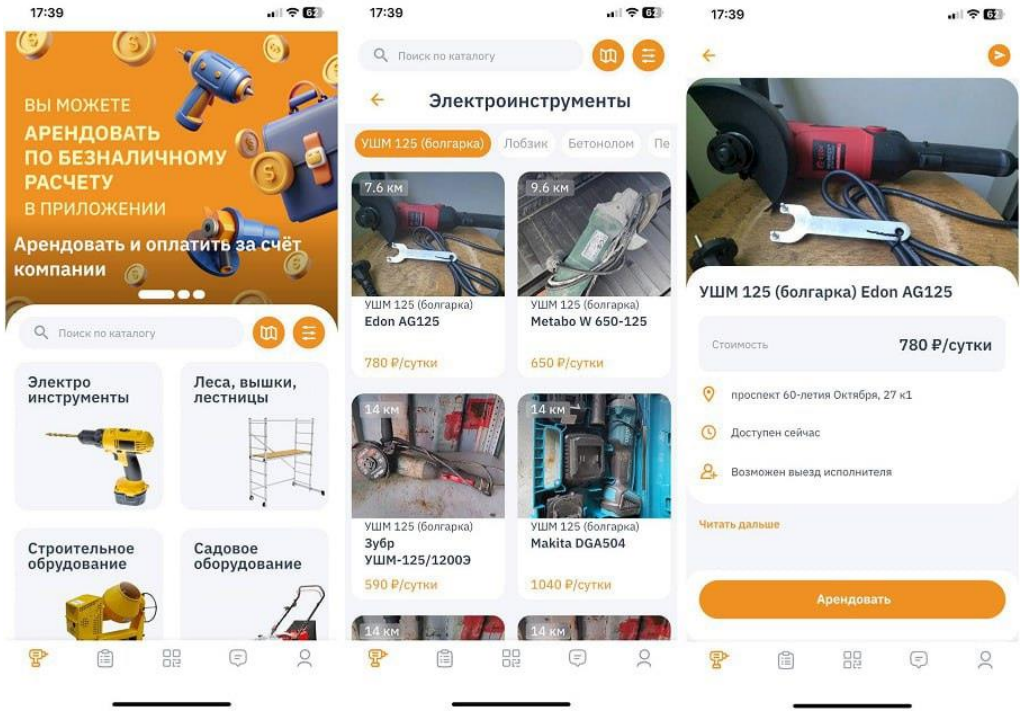

Рисунок 6 – Интерфейс YouTool

Достоинства:

— большой выбор инструментов для аренды;

— возможность аренды инструмента в рассрочку;

— аренда по безналичному расчету в приложении;

— сдача личного инструмента пользователя под аренду;

— блокировка пользователей с низким рейтингом;

— низкая стоимость аренды;

— ремонт инструментов пользователя;

— выездное обслуживание арендованного оборудования;

— система авторизации пользователя с помощью документа, удостоверяющего личность.

Недостатки:

— нет уведомления пользователя о том, что вывод денежных средств со сдачи личного инструмента пользователя в аренду только через ИП или самозанятость, что может запутать арендодателя (пользователя);

— плохая служба поддержки;

— долгая система верификации, сильно отличающаяся от заявленной;

— не интуитивный выбор даты аренды при заключении договора.

2.3.7. Складно

Складно — это сервис для удобной и безопасной аренды инструмента и кладовок у дома. Принцип работы заключается в открывании ячейки постамата через мобильное приложение.

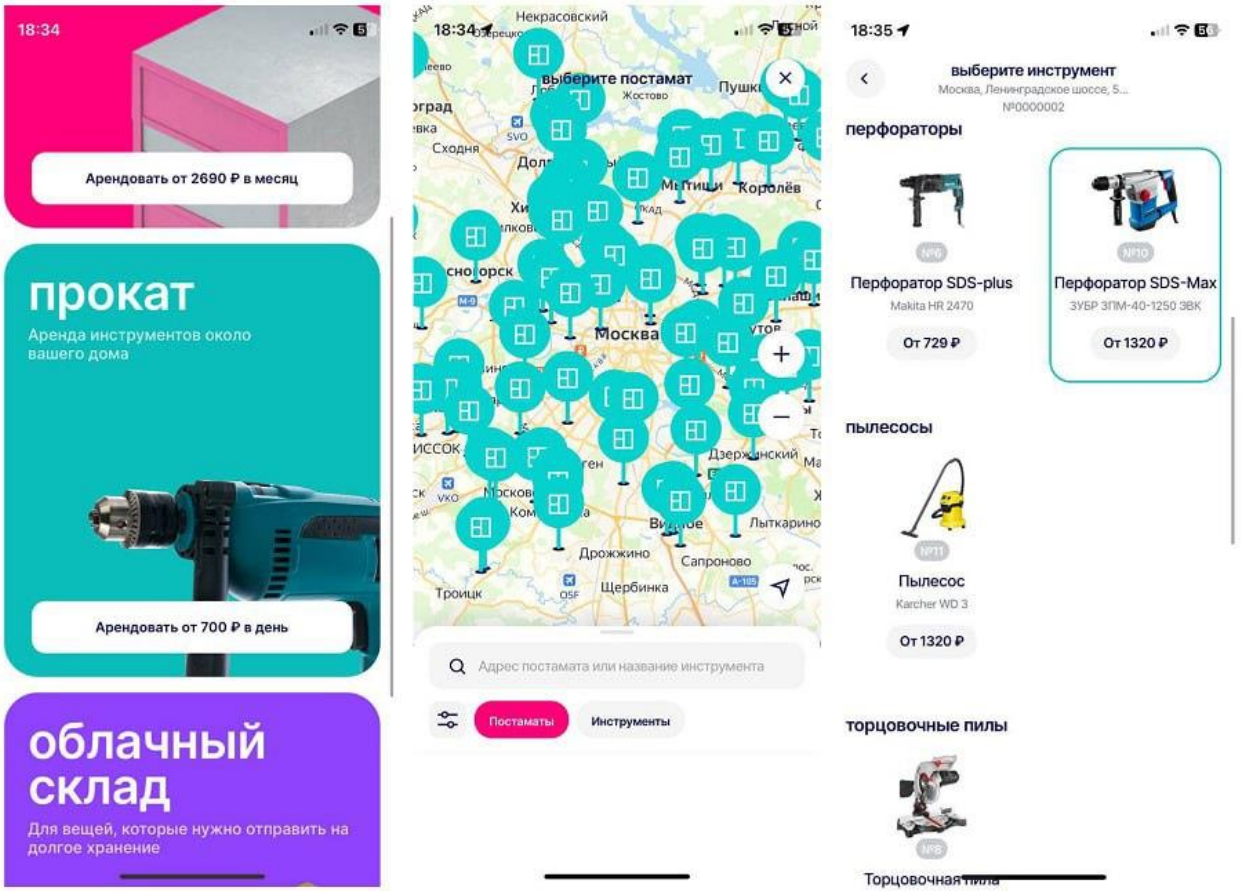

Рисунок 7 – Интерфейс Складно

Достоинства:

— легкий доступ к ячейке с арендованным инструментом;

— арендуемый инструменты известных брендов;

— возможность оплаты заказа онлайн через приложение;

— программа лояльности для оплаты заказов баллами.

Недостатки:

— плохая связь с поддержкой;

— неудобный доступ к ячейке постамата, для доступа к которой может пригодится лестница ;

— замок в постамате может не открыться из-за отсутствия электричества;

— высокая стоимость аренды в сравнении с конкурентами;

— маленький выбор инструментов для аренды.

2.4. Итог анализа

В процессе анализа предметной области было установлено, что при разработке приложения следует придерживаться следующих аспектов:

— отмена/продление аренды;

— информация о наличии инструментов;

— категория избранное;

— информация о наличии инструментов;

— история заказов;

— поиск и фильтрация инструментов;

— функция для выбора нескольких единиц конкретного инструмента;

Также было установлено, что необходимо избегать таких ошибок, как:

— не интуитивный выбор даты аренды при оформлении заказа;

— ограниченная функциональность без авторизации;

3. Реализация

3.1. Структура системы

Система разбита на ряд независимо развертываемых сервисов, которые взаимодействуют с помощью API – интерфейсов. Благодаря такому подходу каждый отдельный сервис можно развертывать и масштабировать независимо от других. В результате команды могут быстрее и чаще поставлять объемные и сложные приложения. В отличие от монолитного приложения, с микросервисной архитектурой команды могут быстрее внедрять новые возможности и вносить изменения, при этом им не приходится переписывать большие фрагменты существующего кода.

В качестве паттерна коммуникации микросервисов был выбран шаблон API шлюз (API Geteway). Данный паттерн был выбран так как он обеспечивает централизованное управление сквозной функциональностью, так же он обеспечивает независимость клиента от протоколов используемых в сервисах: REST, AMQP, gRPC и других, что в будущем позволит расширять систему, не завися от средств реализации.

На рисунке 8 представлена общая структура разрабатываемой системы.

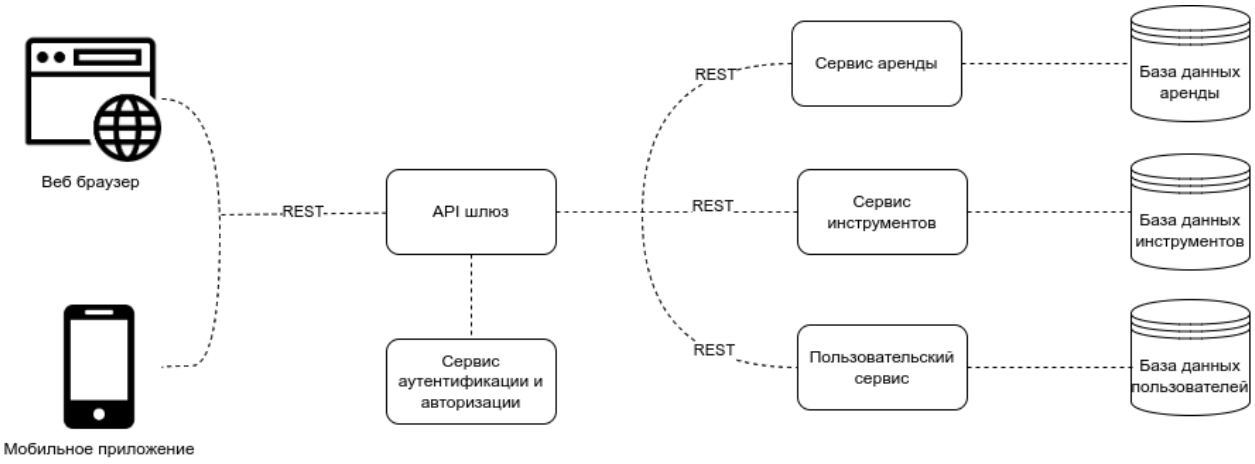

Рисунок 8 – Общая структура системы

Приложения взаимодействуют с микросервисами через API-интерфейсы REST, которые публикует каждый микросервис. Приложения обращаются к

API-интерфейсам микросервисов через API-шлюз, который также позволяет заменять одни микросервисы другими с тем же API.

Каждый микросервис состоит из сервиса и базы данных. Сервисы работают с API-интерфейсами REST, реализуют бизнес-логику и сохраняют информацию в базе данных.

3.2. Средства реализации

3.2.1. Сервер

Для реализации серверной части системы были выбраны следующие технологии:

— язык программирования Java;

— фреймворк SpringBoot;

— СУБД PostgreSQL;

— объектное хранилище MinIO;

— платформа контейнеризации Docker;

— система управления миграциями базы данных Liquibase;

— сервис авторизации, регистрации и аутентификации Keycloak;

— API шлюз Kong;

— веб сервер Nginx;

Преимуществами языка программирования Java являются его платформенная независимость, что помимо облегчения процесса разработки, так же сокращает время и ресурсы, необходимые для адаптации к различным платформам.

Фреймворк SpringBoot был выбран, так как он облегчает разработку приложений на базе Spring, предоставляя такие преимущества, как автоматическая конфигурация, встроенный веб-сервер, поддержка внедрения

зависимостей и возможность создания легковесных и независимых приложений.

В качестве СУБД была выбрана PostgreSQL из-за ее гибкой системы типов, полнотекстового поиска, масштабируемости, надежности и скорости, а так же в возможности расширения функциональности с помощью пользовательских расширений.

Для организации объектного хранилища, было выбрано MinIO. Это объектное хранилище предоставляет простоту установки и настройки, поддержку S3-совместимого API, высокую доступность, отказоустойчивость и эффективное использование ресурсов.

Платформа контейнеризации Docker, предоставляет удобный способ развертывания приложений. Она обеспечивает изоляцию приложений на уровне операционной системы, что позволяет абстрагироваться от прямой зависимости конкретной инфраструктуры и обеспечивает высокую переносимость приложений.

Чтобы упростить изменения в базе данных была выбрана система управления миграциями Liquibase. Она обеспечивает автоматизацию процесса обновления схемы и данных базы, что позволяет избежать ручной работы и сократить время, необходимое для внесения изменений. Кроме того, Liquibase предоставляет такие преимущества, как поддержку различных СУБД, возможность работы в команде и отслеживание изменений, гибкость и настраиваемость процесса миграции, а также обеспечение согласованности и целостности данных.

Авторизация и аутентификация являются сложными и критическими процессами, которые требуют надежного и гибкого решения. Для этого была выбрана платформа безопасности Keycloak, которая предоставляет такие преимущества, как поддержку различных протоколов аутентификации, возможность интеграции с различными приложениями и платформами, а

также настраиваемую политику безопасности. Все это позволяет значительно упростить и улучшить процессы авторизации и аутентификации, а также обеспечить надежную защиту данных и приложений.

Так как приложение разрабатывается с использование API Geteway подхода, то в качестве готового решения был выбран Kong. Kong предоставляет поддержку различных протоколов и форматов данных, возможность настройки маршрутизации и балансировки нагрузки, интеграцию с системами аутентификации и авторизации. Так же он имеет поддержку системы плагинов, позволяющего расширять его функциональность.

Выбор веб-сервера Nginx был обусловлен его эффективностью и высокой производительностью. Он отличается низким потреблением ресурсов, что является особенно актуальным в условиях их ограничения. Кроме того, Nginx может выступать в роли обратного прокси-сервера, что позволяет значительно улучшить процессы взаимодействия между приложениями и сервисами, а также обеспечить надежную и эффективную работу системы в целом.

3.2.2. Клиент

Для реализации клиентской части системы были выбраны следующие технологии:

— фреймворк Flutter;

— поисково-информационная картографическая служба Yandex Maps;

— система геокодирования Yandex Geocoder;

— инструмент для трекинга и аналитики AppMetrica;

— локальное хранилище Realm;

Выбор фреймворка Flutter обусловлен такими преимуществами как кросс-платформенность, высокая производительность, гибкость в построении интерфейса, богатого набора виджетов и простоты разработки.

Для поддержки карт в мобильном приложении было выбрано готовое решение от Яндекс – Yandex Maps SDK. Yandex Maps предлагают лучшее и детальное покрытие для России и стран СНГ. Так же они предоставляют бесплатный доступ к API, если он не превышает заданного лимита запросов в месяц.

Yandex Geocoder предоставляет такие преимущества, как прямое и обратное преобразование между адресом и координатами, учет распространенных опечаток, выбор вида топонима, ограничение поиска указанной областью, а также поддержку различных языков и региональных особенностей карты, что позволяет значительно упростить и улучшить процессы геокодирования и поиска объектов на карте.

В качестве аналитической системы был выбран сервис AppMetrica. AppMetrica обладает широким функционалом для мобильной аналитики. Это бесплатный и удобный инструмент. Сервис просто устанавливать и настраивать. Он быстро собирает статистику и помогает оперативно принимать решения по развитию и продвижению приложений.

Для локального хранилища был выбран Realm. Realm – это локальное хранилище данных на основе объектной модели, что обеспечивает высокую производительность, гибкость и настраиваемость схемы данных, поддержку транзакций и конфликтного разрешения, а также интеграцию с другими приложениями и сервисами.

3.3. Диаграмма вариантов использования

На рисунке 9 представлена UML диаграмма вариантов использования, иллюстрирующая функционал анонимного пользователя.

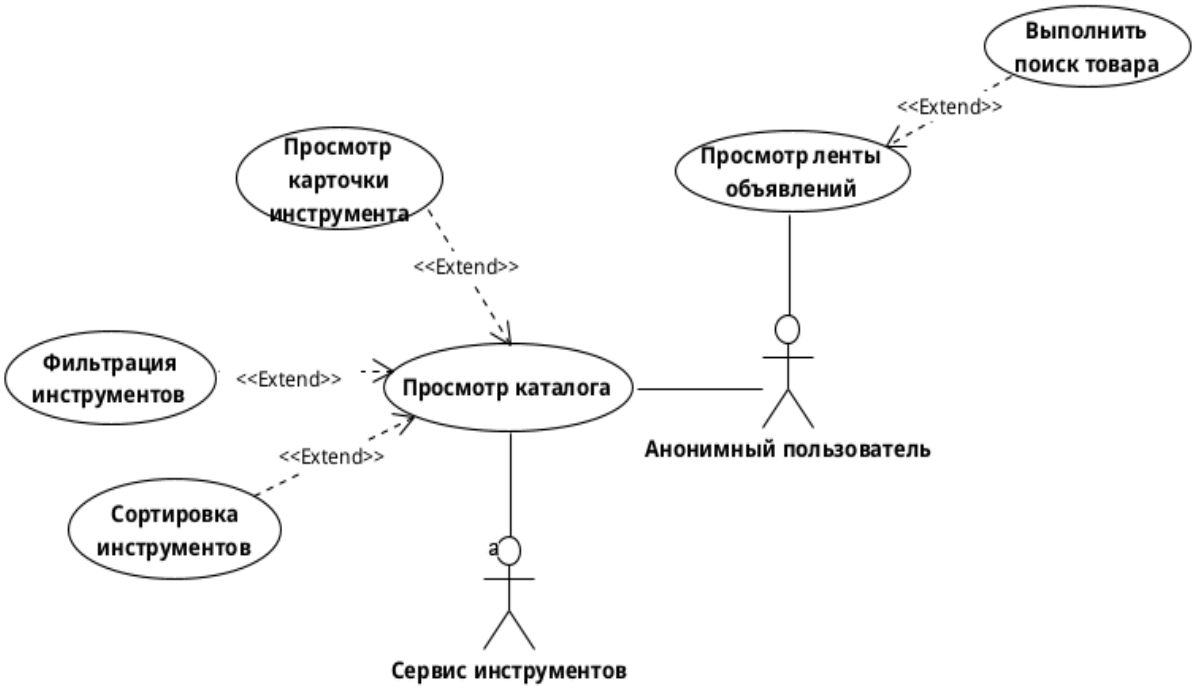

Рисунок 9 – Диаграмма анонимного пользователя

На рисунке 10 представлена UML диаграмма вариантов использования, иллюстрирующая функционал арендатора по оформлению аренды.

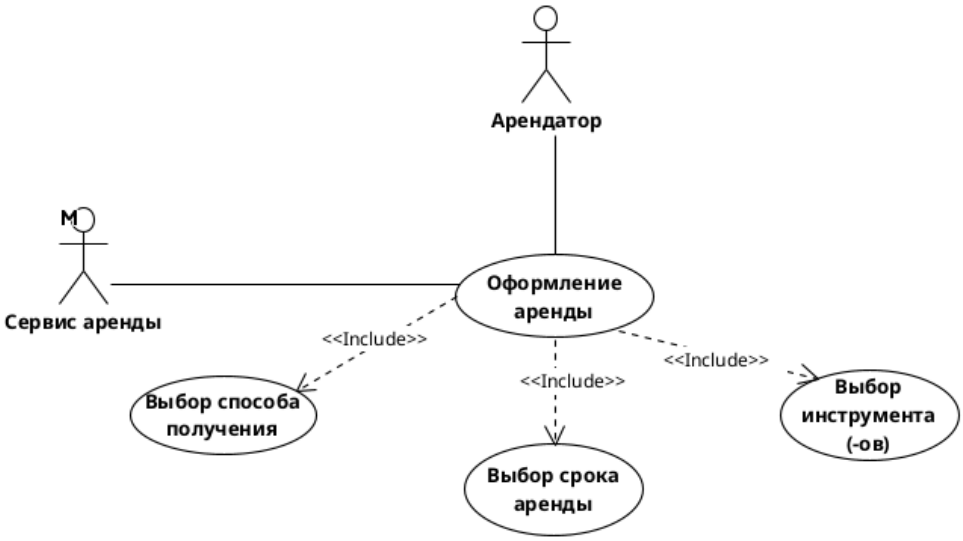

Рисунок 10 – Диаграмма арендатора (оформление аренды)

На рисунке 11 представлена UML диаграмма вариантов использования, иллюстрирующая функционал арендатора по отказу и продления аренды.

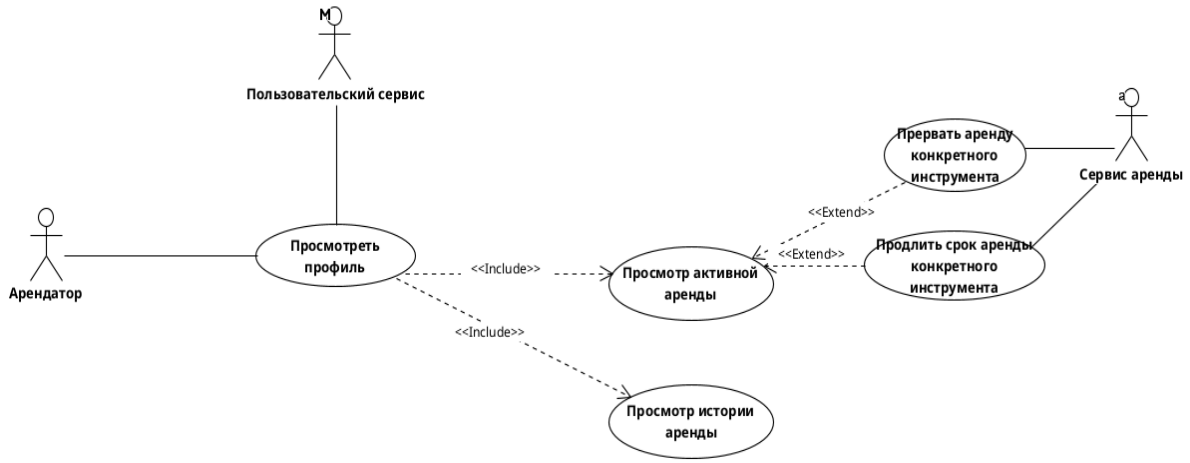

Рисунок 11 – Диаграмма арендатора (отказ или продление аренды)

На рисунке 12 представлена UML диаграмма вариантов использования, иллюстрирующая функционал сервиса инструментов.

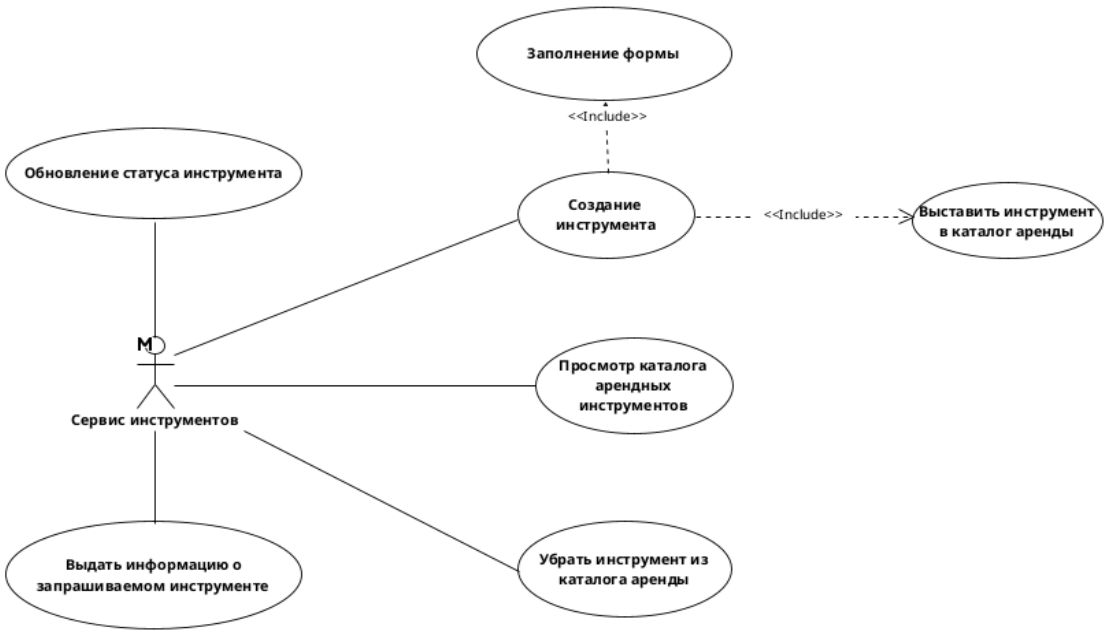

Рисунок 12 – Диаграмма сервиса инструментов

На рисунке 13 представлена UML диаграмма вариантов использования, иллюстрирующая функционал сервиса аренды.

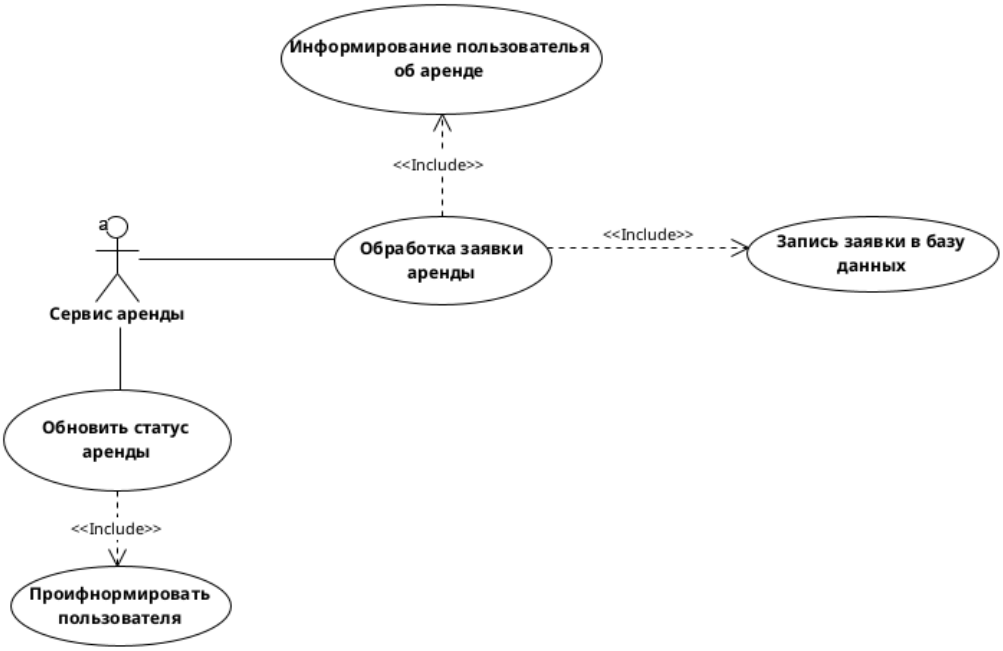

Рисунок 13 – Диаграмма сервиса аренды

На рисунке 14 представлена UML диаграмма вариантов использования, иллюстрирующая функционал пользовательского сервиса.

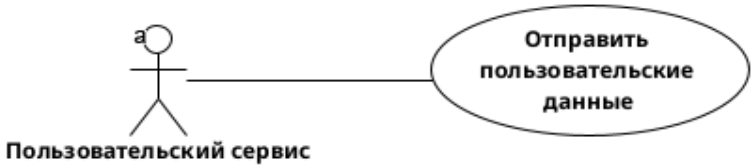

Рисунок 14– Диаграмма пользовательского сервиса

На рисунке 15 представлена UML диаграмма вариантов использования, иллюстрирующая функционал администратора системы.

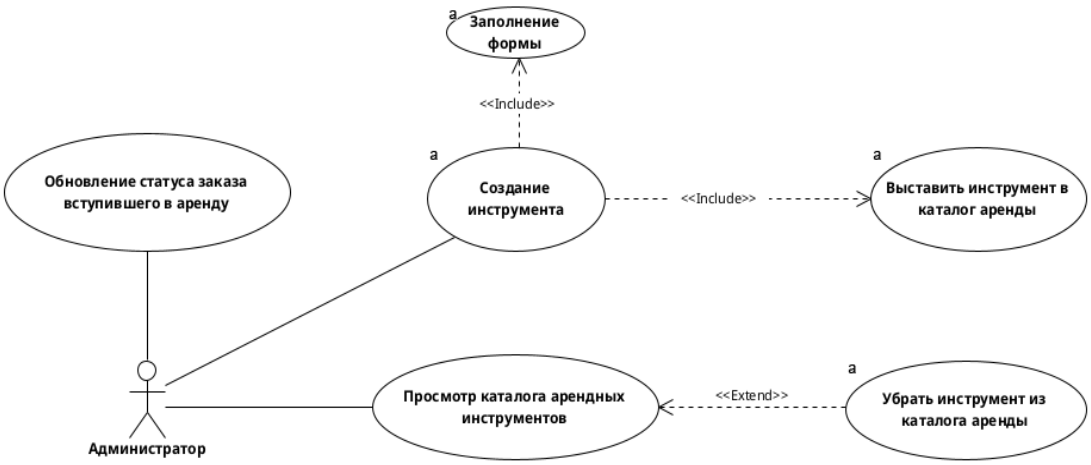

Рисунок 15 – Диаграмма администратора

3.4. Диаграмма активности

На рисунке 16 представлена UML диаграмма активности, иллюстрирующая процесс обработки заказа.

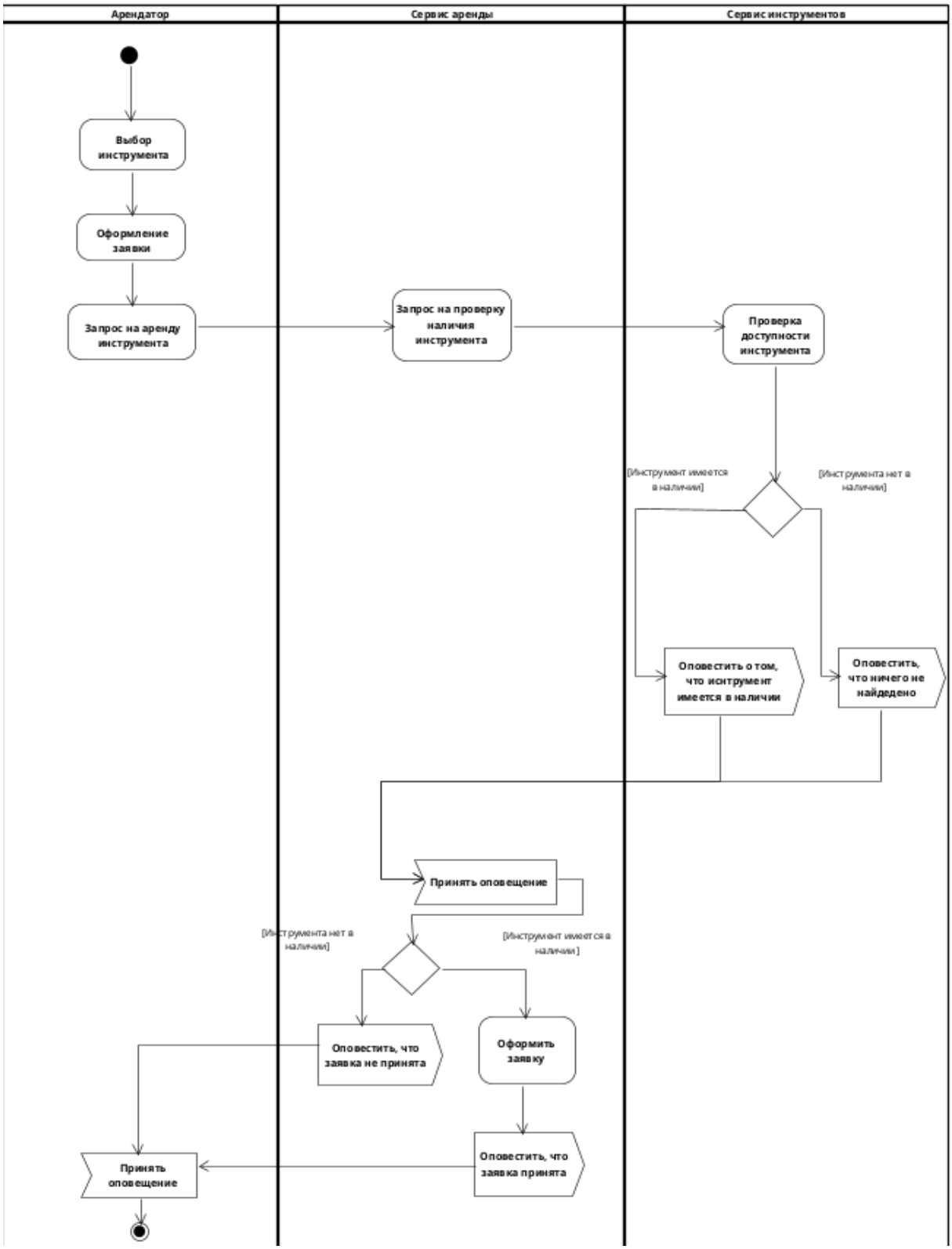

Рисунок 16 – Диаграмма активности обработки заказа

3.5. Диаграмма состояний

На рисунке 17 представлена UML диаграмма состояний инструмента.

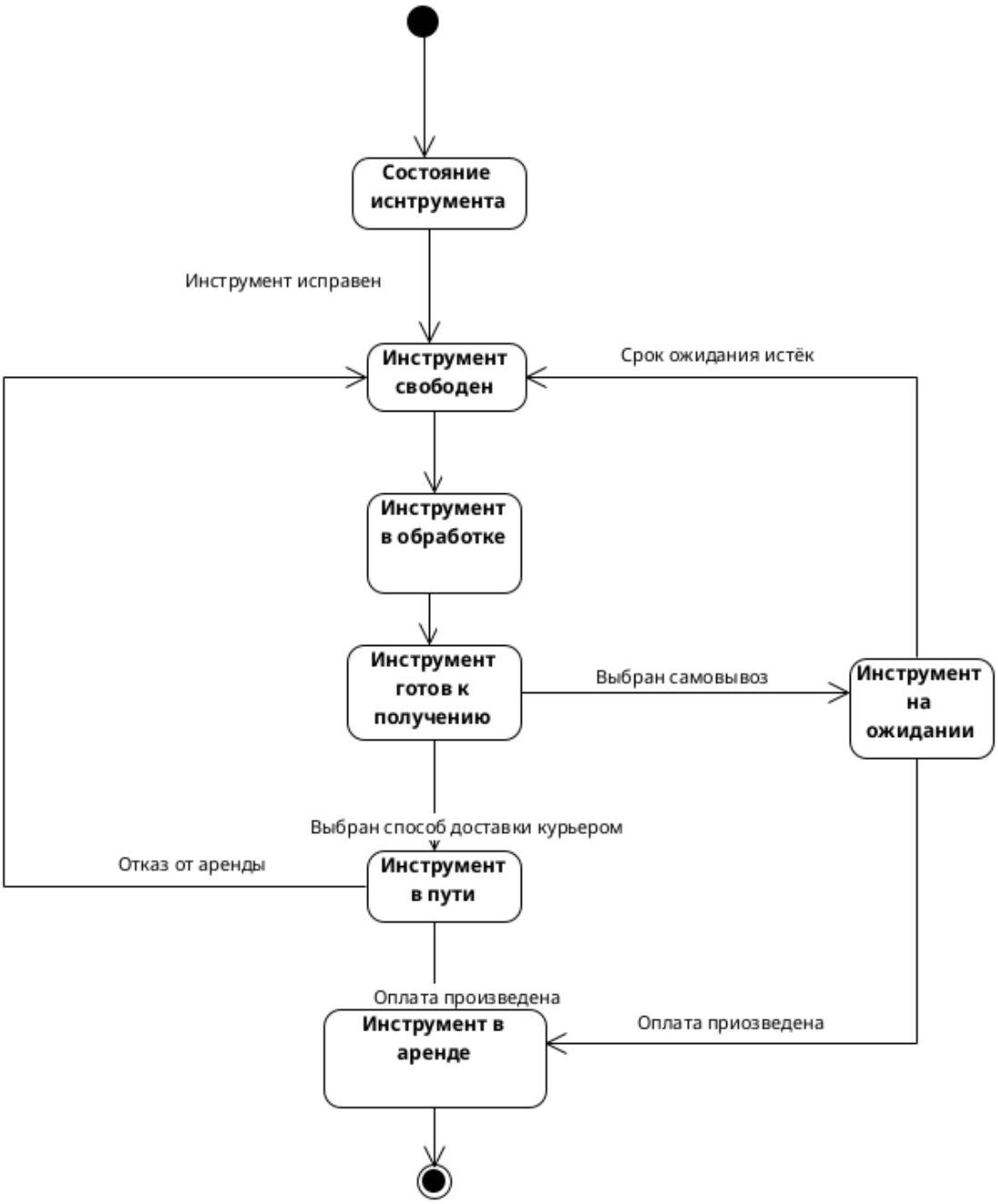

Рисунок 17 – Диаграмма состояний инструмента

3.6. Диаграмма развертывания

На рисунке 19 представлена UML диаграмма развертывания системы.

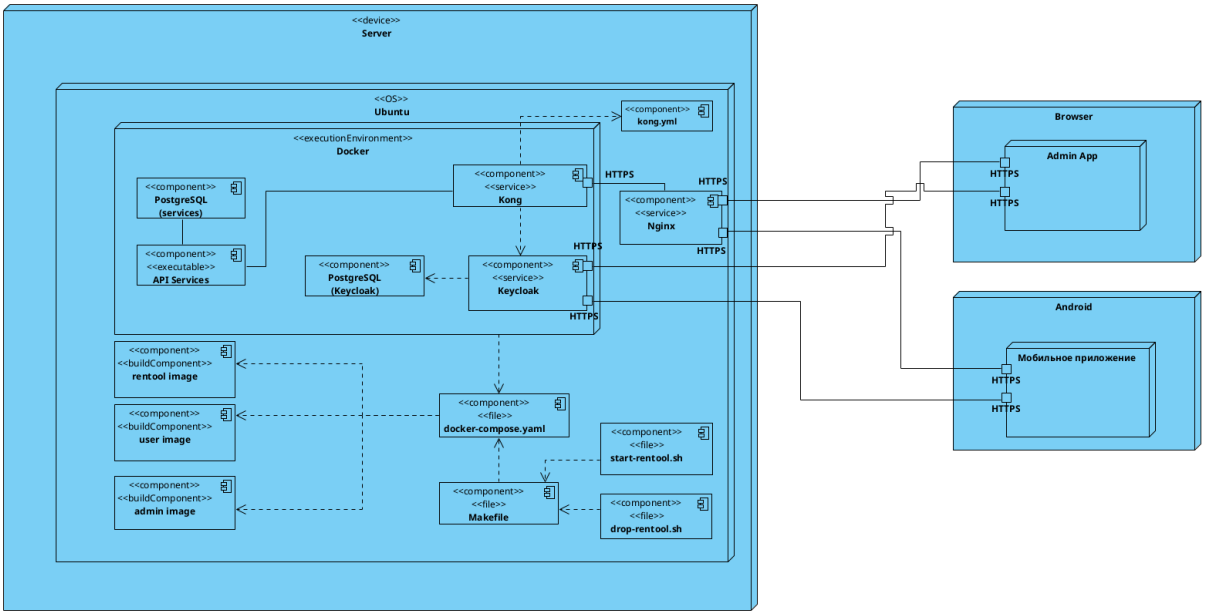

Рисунок 18 – Диаграмма развертывания

3.7. Диаграмма последовательности

Диаграммы последовательности для анонимного пользователя представлены на рисунках 20-21.

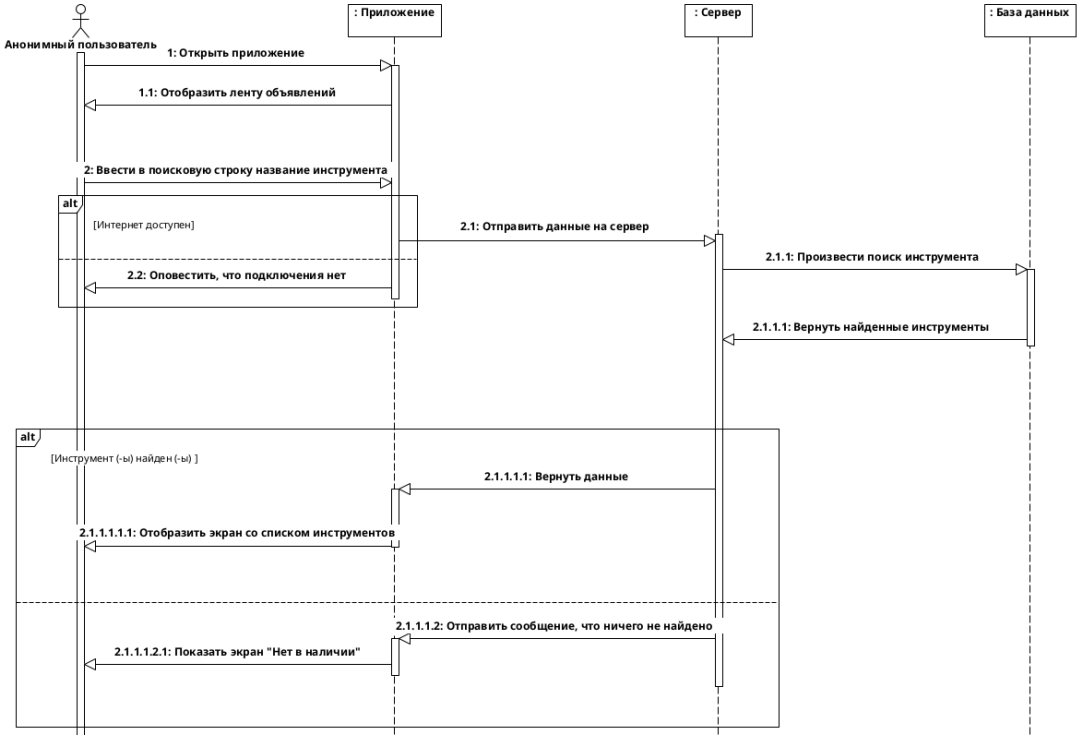

Рисунок 19 – Диаграмма последовательности поиска инструмента

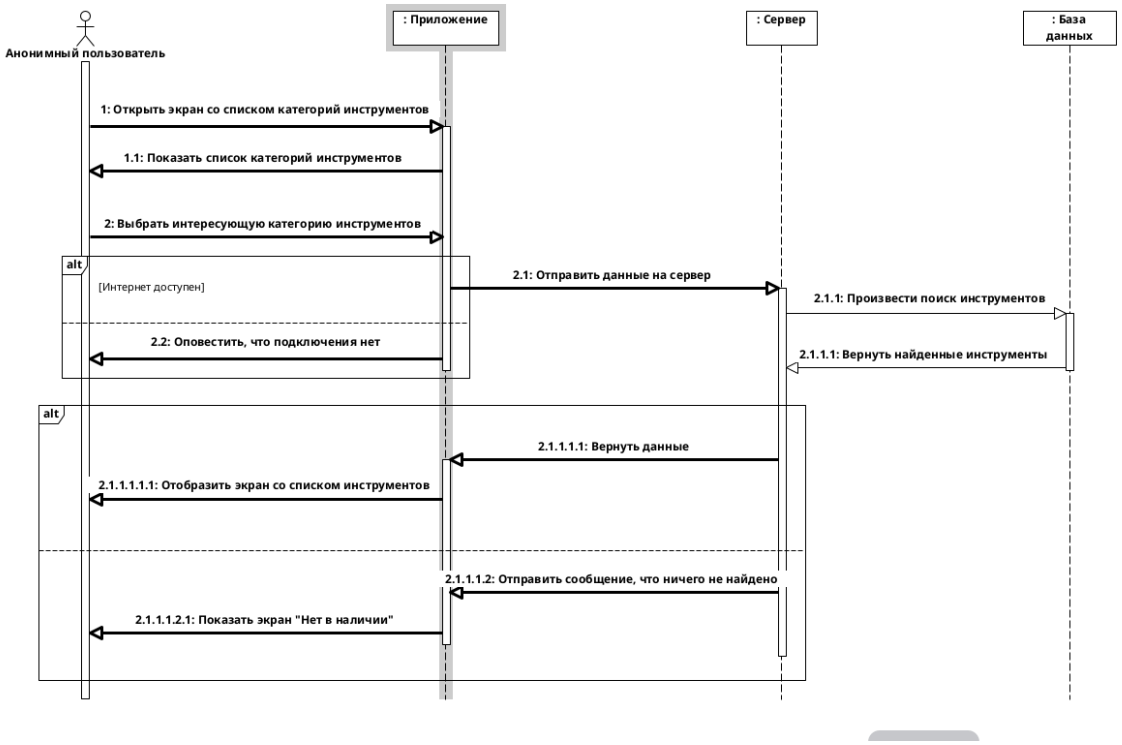

Рисунок 20 – Диаграмма последовательности фильтра по категориям

На рисунках 22-24 представлены диаграммы последовательностей для арендатора.

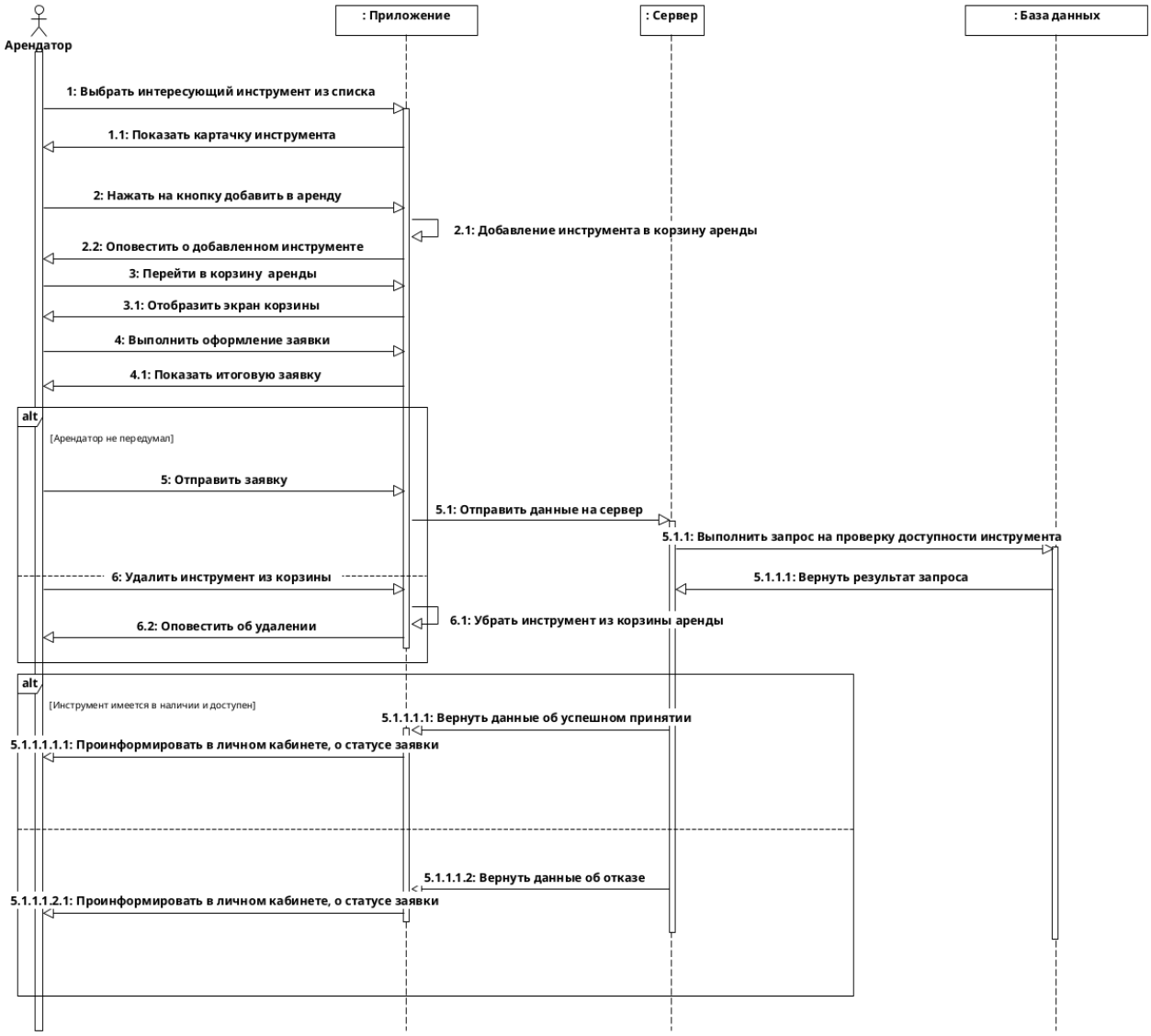

Рисунок 21 – Диаграмма последовательности аренды инструмента

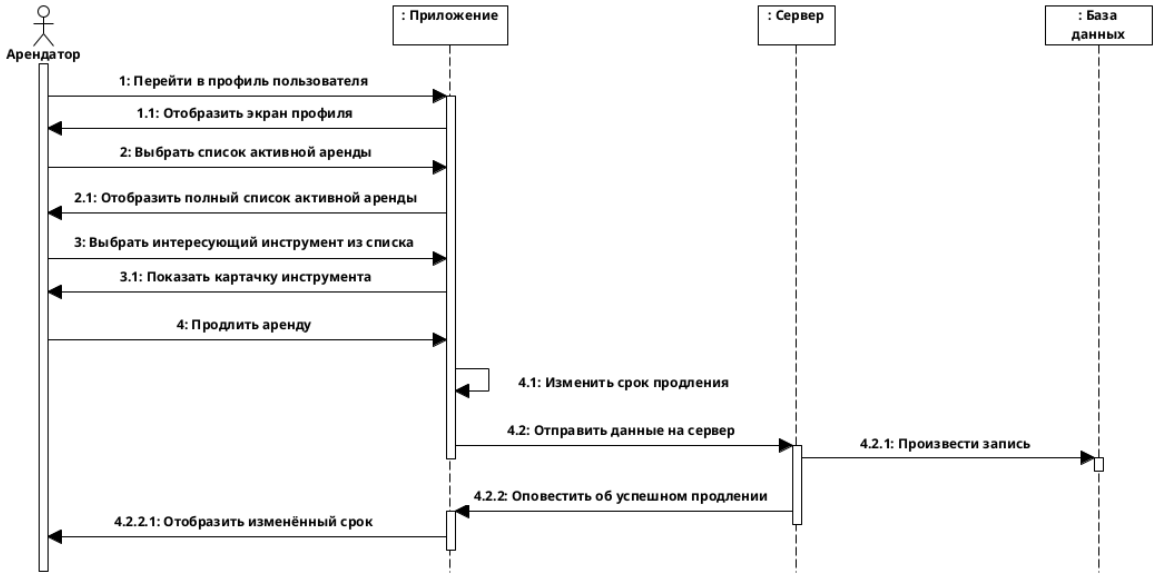

Рисунок 22 – Диаграмма последовательности продления аренды

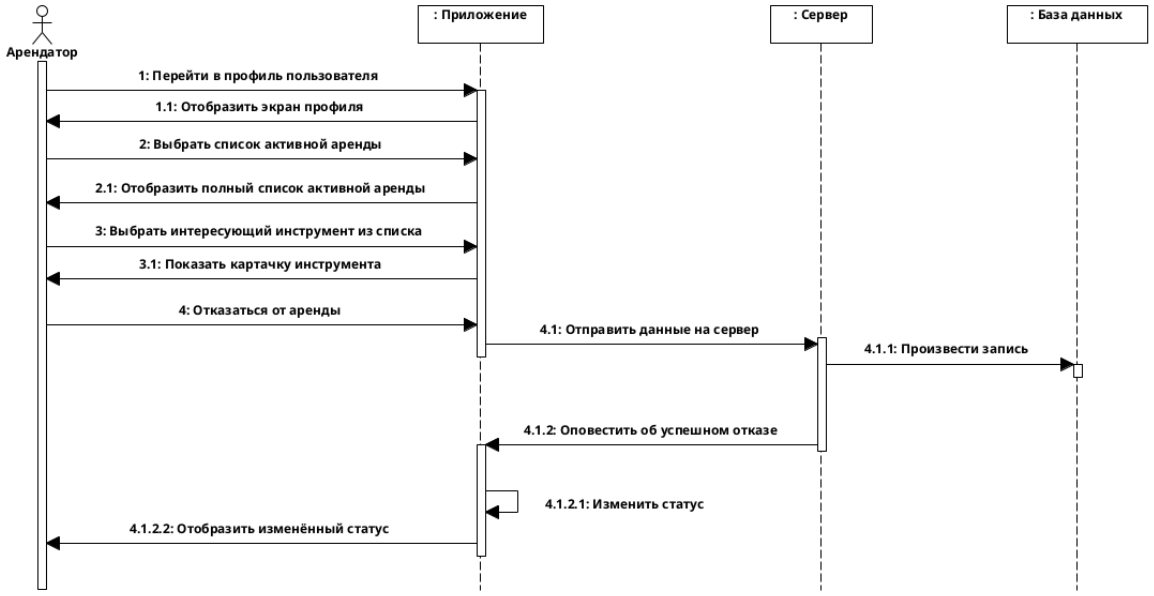

Рисунок 23 – Диаграмма последовательности отказа от аренды

3.8. Серверная часть

Для реализации серверной части использовался микросервисный подход. Все сервисы являются независимыми компонентами, которые могут взаимодействовать между собой через API, что повышает масштабируемость и надежность системы в целом.

При реализации каждого сервиса использовался паттерн MVC(Model-View-Controller), который позволяет разделить логику приложения на три основные части: модель, представление и контроллер. Модель отвечает за работу с данным, представление за отображение информации пользователю, а контроллер – за обработку запрос и управление приложением. Такое разделение упрощает разработку, тестирование и поддержку приложения.

3.8.1. Сервис инструментов

Данный сервис разделен на три основных слоя: api, service, repository.

На рисунке 25 показана UML диаграмма пакетов, которая иллюстрирует структуру взаимосвязи между пакетами сервиса tool.

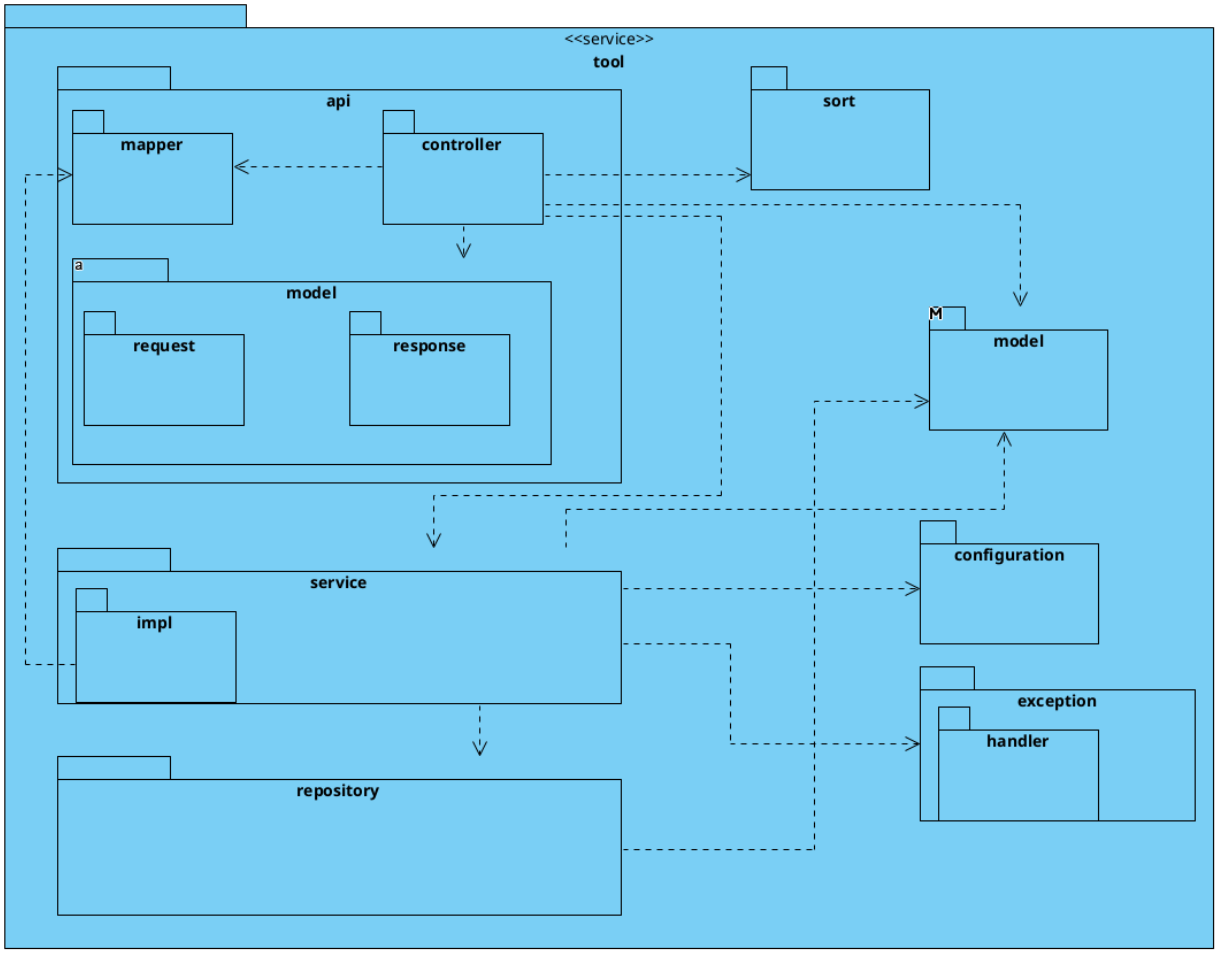

Рисунок 24 – Диаграмма пакетов сервиса tool

Слой api является входной точкой для взаимодействия с сервисом и содержит все необходимые компоненты для обработки входящих запросов и формирования ответов. Он включает в себя контроллеры, которые отвечают за прием запросов и передачу их на обработку в слой сервисов, а также DTO модели запросов и ответов, которые используются для валидации и преобразования данных. Кроме того, в слое API присутствуют мапперы, которые преобразуют модели данных между слоями приложения.

Слой сервисов является центральным компонентом сервиса и содержит бизнес-логику приложения. Он включает в себя сервисы, которые отвечают за обработку запросов и выполнение CRUD операций, а также модели данных, которые используются для представления сущностей приложения.

Слой репозиториев является нижним слоем сервиса и отвечает за взаимодействие с базой данных. Он включает в себя репозитории, которые предоставляют интерфейс для выполнения операций с данными.

Помимо этих основных слоев присутствуют дополнительные пакеты, которые обеспечивают корректную работу сервиса. Одним из них является пакет exception, содержащий иерархию классов исключений, которые могут возникнуть в процессе работы сервиса.

Другим дополнительным пакетов является пакет configuration, содержащий в себе настройку для работы с сервисом MinIO.

Пакет model содержит в себе основные модели, отражающие их физическое представление в базе данных. На рисунке 26 показана физическая схема базы данных сервиса tool.

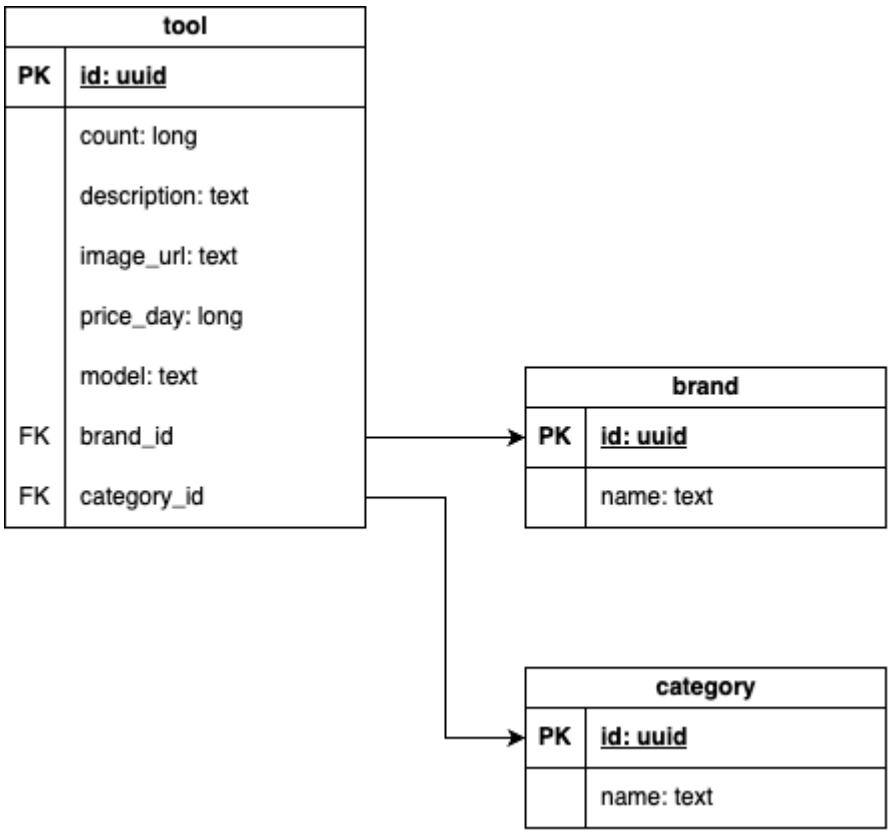

Рисунок 25 – Физическая схема базы данных сервиса tool

3.8.2. Сервис аренды

Данный сервис разделен на три основных слоя: api, service, repository.

В этом сервисе находится вся основная бизнес-логика нашего приложения.

На рисунке 27 показана UML диаграмма пакетов, которая иллюстрирует структуру взаимосвязи между пакетами сервиса rent.

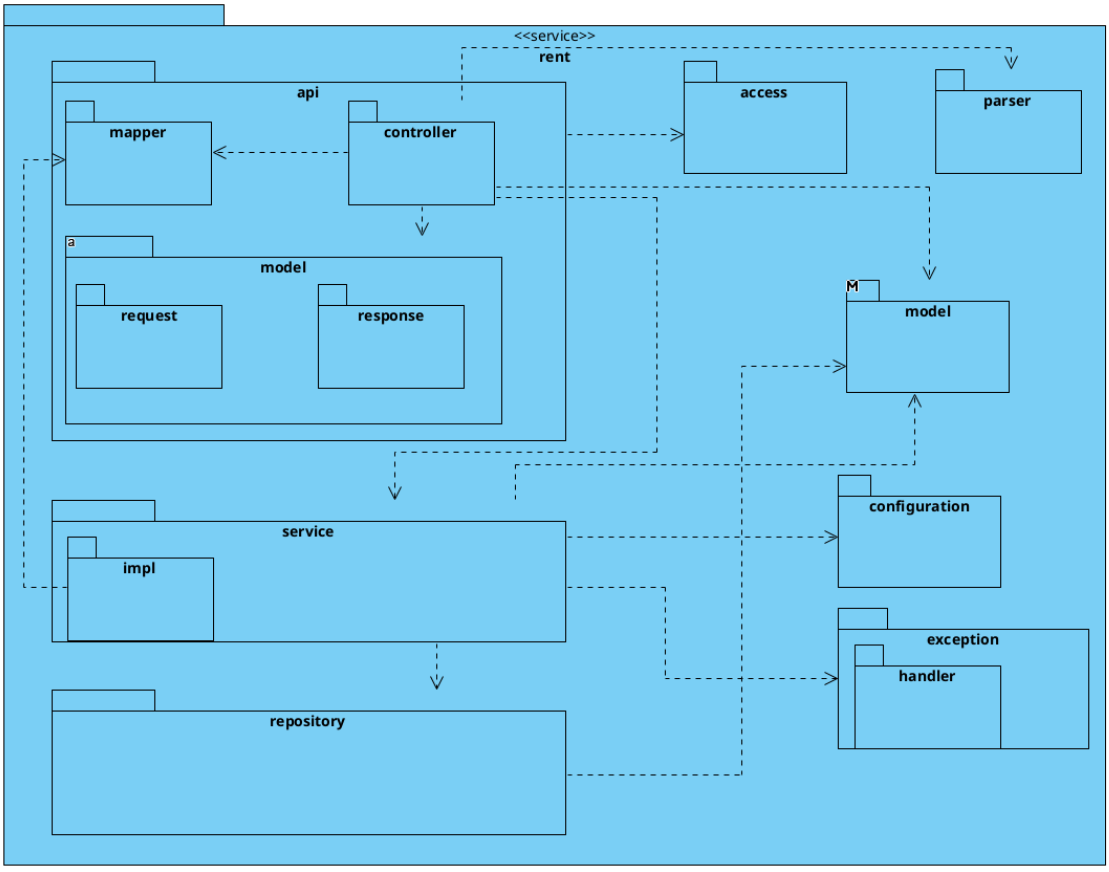

Рисунок 26 – Диаграмма пакетов сервиса rent

На рисунке 28 показана физическая схема базы данных сервиса rent.

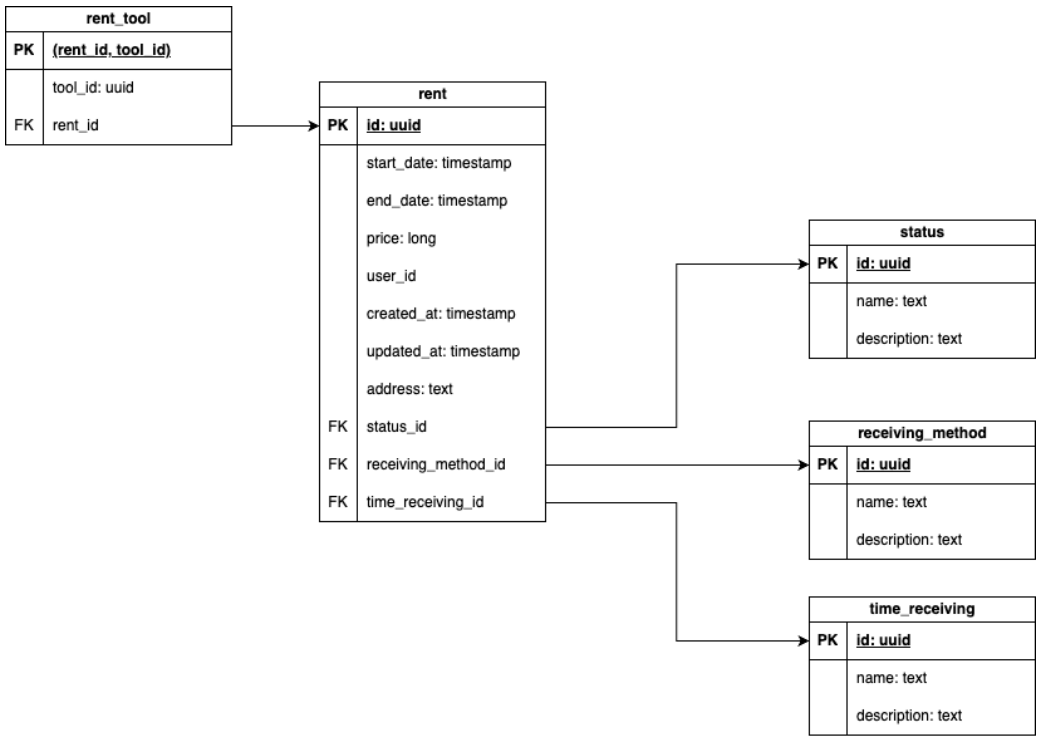

Рисунок 27 – Физическая схема базы данных сервиса rent

3.8.3. Сервис user

Данный сервис разделен на три основных слоя: api, service, repository.

Главной особенностью является интеграция с системой авторизации Keycloak, что реализовано за счет написания клиента к этому сервису.

На рисунке 29 показана UML диаграмма пакетов, которая иллюстрирует структуру взаимосвязи между пакетами сервиса user.

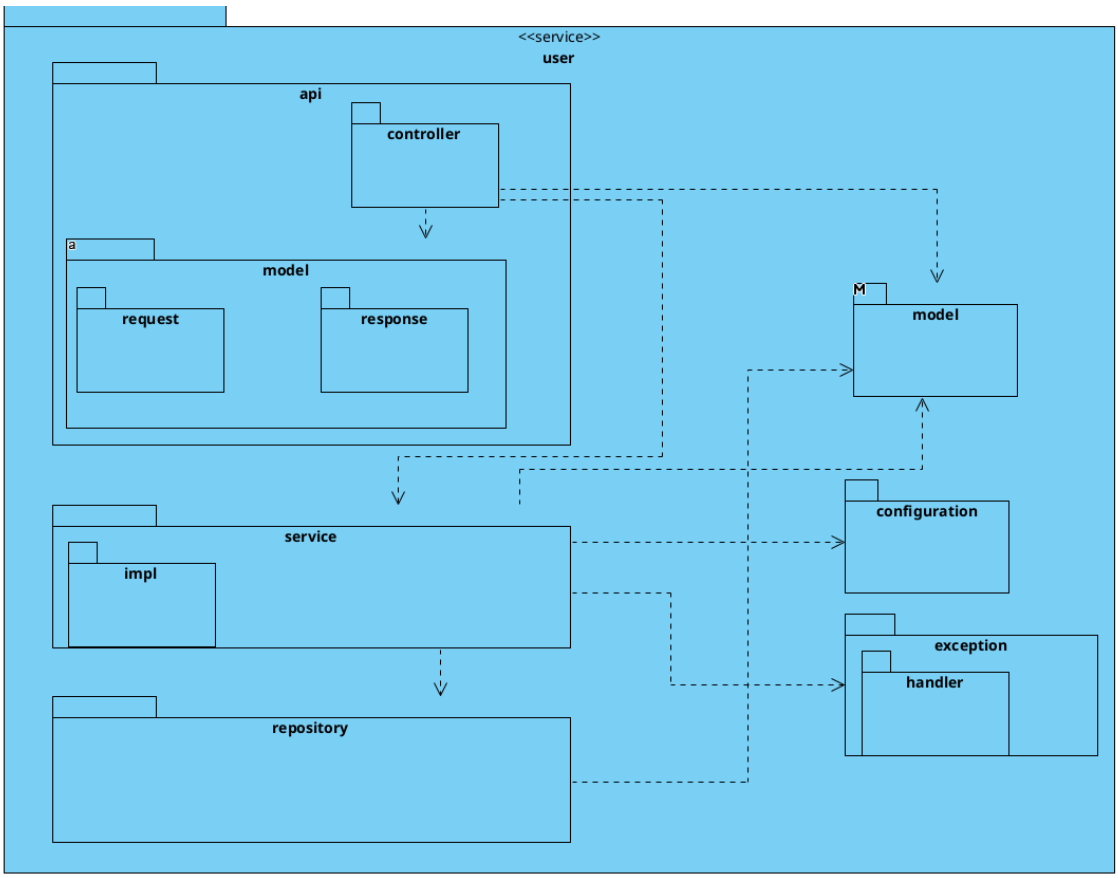

Рисунок 28 – Диаграмма пакетов сервиса user

На рисунке 30 показана физическая схема базы данных сервиса user.

person	
PK	id: uuid
	login: text
	phone: text
	email: text
	first_name: text
	last_name: text

Рисунок 29 – Физическая схема базы данных сервиса user

3.9. Клиентская часть

Для реализации мобильного приложения использовался паттерн проектирования BLoC. Это паттерн проектирования, который помогает разделить бизнес-логику и пользовательский интерфейс в приложении. Для каждой новой функциональности создаются отдельные блоки для обработки событий, генерируемых пользовательским интерфейсом, и отправки новых состояний в пользовательский интерфейс. Это позволяет легко управлять состоянием приложения и масштабировать его в будущем.

Для реализации паттерна BLoC использовался пакет fltter_bloc. Этот пакет предоставляет набор инструментов и утилит, которые помогают разработчикам реализовывать паттерн BLoC в своих приложениях. Он включает в себя генераторы кода, middleware для обработки событий, и инструменты для тестирования.

На рисунке 31 показана общая структура проекта.

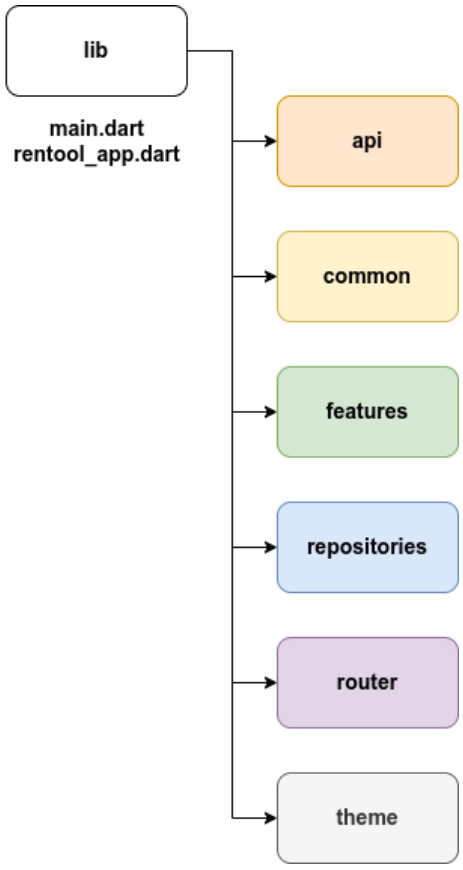

Рисунок 30 – Архитектура мобильного приложения

Пакет api: отвечает за взаимодействие с внешними сервисами и предоставляет данные для приложения. Реализует обращение к API сервисам и обрабатывает полученные данные для последующего использования в приложении.

Пакет common: содержит общие виджеты и компоненты, которые используются на различных экранах приложения. Предоставляет набор готовых решений для создания интерфейса приложения и способствует единообразию дизайна.

Пакет features: содержит экраны, виджеты, блоки и другие компоненты, относящиеся к конкретной функциональности приложения. Организует код приложения по функциональным блокам и облегчает навигацию по проекту.

Пакет repositories: отвечает за взаимодействие с локальным хранилищем данных Realm. Реализует операции по сохранению, извлечению и удалению данных из хранилища для конкретных сущностей, а так же предоставляет их для использования в приложении.

Пакет router: содержит настройки маршрутов для приложения и отвечает за навигацию между экранами.

Пакет theme: содержит настройки темы приложения и отвечает за ее отображение.

Заключение

В ходе выполнения данного курсового проекта были выполнены все поставленные цели:

— предоставить пользователям возможность арендовать инструмент;

— сокращение простоя оборудования;

— расширение клиентской базы.

Список используемых источников

1. Документация SpringBoot [Электронный ресурс]. – Режим доступа: URL:https://spring.io/projects/spring-boot – Заглавление с экрана. – (Дата обращения 15.03.2024).
2. Документация Dart [Электронный ресурс]. – Режим доступа: URL:https://dart.dev/guides – Заглавление с экрана. – (Дата обращения 20.03.2024).
3. Документация Flutter [Электронный ресурс]. – Режим доступа: URL:https://docs.flutter.dev/ – Заглавление с экрана. – (Дата обращения 25.03.2024).
4. Документация Atlas Device SDK [Электронный ресурс]. – Режим доступа: URL:https://www.mongodb.com/docs/atlas/device-sdks/sdk/flutter – Заглавление с экрана. – (Дата обращения 01.04.2024).
5. Документация Kong [Электронный ресурс]. – Режим доступа: URL:https://docs.konghq.com/ – Заглавление с экрана. – (Дата обращения 25.04.2024).
6. Документация Keycloak [Электронный ресурс]. – Режим доступа: URL:https://www.keycloak.org/documentation – Заглавление с экрана. – (Дата обращения 25.04.2024).