МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное бюджетное образовательное учреждение высшего образования
«Тольяттинский государственный университет»

Кафедра Прикладная математика и информатика
(наименование)

09.03.03 Прикладная информатика
(код и наименование направления подготовки / специальности)

Разработка программного обеспечения
(направленность (профиль)/специализация)

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА (БАКАЛАВРСКАЯ РАБОТА)

на тему «Разработка программного обеспечения чат-бота для информационной поддержки клиентов компании Уралпромстрой»

Обучающийся М.А. Долгополова
(Инициалы Фамилия)
(личная подпись)

Руководитель д.т.н., доцент, С.В. Мкртычев
(ученая степень (при наличии), ученое звание (при наличии), Инициалы Фамилия)

Тольятти 2024

Аннотация

Тема выпускной квалификационной работы – «Разработка программного обеспечения чат-бота для информационной поддержки клиентов компании Уралпромстрой».

Цель работы: разработать программное обеспечение чат-бота для упрощения взаимодействия клиентов с ООО «УРАЛПРОМСТРОЙ».

Актуальность темы обусловлена потребностью предприятия ООО «УРАЛПРОМСТРОЙ» автоматизировать процесс обработки обращений клиентов для сокращения ресурсных затрат и повышения клиентского сервиса.

Практическая значимость работы заключается в автоматизации обработки обращений клиентов с целью экономии ресурсов и увеличения скорости обслуживания.

Работа состоит из введения, трех глав, заключения и списка используемой литературы.

В первой главе проанализирован бизнес-процесс общения с клиентами в ООО «УРАЛПРОМСТРОЙ», составлены требования к программного обеспечения разрабатываемого чат-бота, проведен сравнительный анализ аналогов.

Во второй главе проанализированы и выбраны наиболее технологии для разработки программного обеспечения чат-бота, описаны логическая модель и архитектура.

В третьей главе описаны процессы реализации и тестирования программного обеспечения чат-бота.

В заключении описаны выводы и результаты.

Объем выполненной работы – 51 страница, 36 рисунков, 7 таблиц, 20 источников информации.

Оглавление

Введение

Высокие скорости в принятии решений, многозадачность, необходимость снижения рисков требуют автоматизации бизнес-процессов. Автоматизация бизнеса – это перевод некоторых операций под управление информационной системы [1]. Результат – повышение производительности труда, увеличение скорости выполнения операций, экономия ресурсов (временных, человеческих, финансовых).

Обработка обращений клиентов является одним из бизнес-процессов, которые можно автоматизировать. Обращающемуся за консультацией клиенту нужно в удобном и доступном виде получить информацию. Но для этого специально обученный сотрудник должен выделять время на проведение консультации. Из-за ограниченности ресурсов компании возможности общения с клиентами ограничены. При увеличении объема обращений снижается скорость и качество их обработки. Автоматизация этого процесса позволяет разгрузить сотрудников, что приводит к улучшению клиентского сервиса предприятия.

Использование чат-ботов – это способ автоматизировать многие действия, в том числе общение с клиентами. Все больше компаний начинают пользоваться этой технологией, так как это удобно и для предприятия, и для клиентов. Боты имитируют человеческую речь, для пользователя переписка с ботом выглядит, как диалог с реальным человеком. При этом сотрудники компании не тратят время на разговоры с клиентами.

Актуальность темы обусловлена потребностью предприятия ООО «УРАЛПРОМСТРОЙ» автоматизировать процесс обработки обращений клиентов для сокращения ресурсных затрат и повышения клиентского сервиса.

Объект исследования – чат-бот для упрощения взаимодействия клиентов с ООО «УРАЛПРОМСТРОЙ».

Предмет исследования – программное обеспечение чат-бота для упрощения взаимодействия клиентов с ООО «УРАЛПРОМСТРОЙ».

Цель работы: разработать программное обеспечение чат-бота для упрощения взаимодействия клиентов с ООО «УРАЛПРОМСТРОЙ».

Для достижения поставленной цели необходимо решение следующих задач:

- выполнить постановку задачи на разработку программного обеспечения чат-бота;
- спроектировать программного обеспечение чат-бот;
- выполнить реализацию и тестирование программного обеспечение чат-бота.

Практическая значимость работы заключается в разработке программного обеспечения чат-бота для упрощения взаимодействия клиентов с ООО «УРАЛПРОМСТРОЙ».

Глава 1 Постановка задачи на разработку программного обеспечения чат-бота

1.1 Функциональные и архитектурные особенности чат-ботов

Для сокращения ресурсных затрат на общение с клиентами в ООО «УРАЛПРОМСТРОЙ» было принято решение внедрить чат-бот для предоставления информации клиентам.

Чат-бот – это компьютерная программа, которая имитирует человеческую речь (устную или письменную) и позволяет общаться с цифровыми устройствами так, как если бы они были живыми людьми [9]. Например, это могут быть аккаунты в социальных сетях, которые управляются не человеком, а программой. При этом для пользователя переписка с аккаунтом-ботом выглядит как диалог с реальным человеком.

Работа чат-ботов строится состоит из 3 основных действий [2]:

- получение и вывод информации (они происходят через определенные каналы связи, например, социальные сети);
- распознавание намерения (анализ полученной информации для формирования ответа);
- обработка действия (любая работа, проведенная на серверной стороне, необходимая для подготовки ответа).

С помощью чат-ботов можно автоматизировать многие действия, например:

- ответы на типовые вопросы,
- поиск данных в базе,
- сбор отзывов,
- запись на услугу,
- сбор данных клиентов.

Популярность чат-ботов постоянно растет. Это связано с тем, что чат-боты обладают большим количеством преимуществ по сравнению с другими решениями (колл-центры, личное общение с клиентами, веб-сайты, мобильные приложения). Можно выделить следующие плюсы чат-ботов:

- чат-боты позволяют сократить время, необходимое для обслуживания клиента;
- количество клиентов, которое можно обслуживать одновременно, не ограничено количеством персонала, как при личном обращении клиентов к сотрудникам компании;
- чат-боты позволяют компаниям отказаться от расходов на рассылку смс-сообщений и электронных писем, ведь рассылка может выполняться с помощью бота;
- боты могут работать в тех же соцсетях и мессенджерах, где пользователь переписывается с друзьям. Пользователю не нужно переключаться между разными приложениями или устанавливать какие-то дополнительные программы;
- как только пользователь запустил чат-бота, он сразу сохраняется в контактах мессенджера, и пользователь сможет вернуться к нему в любое время;
- чат-ботами удобно пользоваться с помощью смартфона, в отличие от сайтов, которые часто даже не адаптированы для мобильных устройств;
- разрабатывать чат-ботов быстрее и дешевле, чем мобильные приложения.

Благодаря данным преимуществам чат-боты активно используются бизнесе. Например, они могут применяться в центрах поддержки клиентов для ответов на популярные вопросы. Могут предоставлять разные сервисы, такие как оформление заказов в интернет-магазине, бронирование номеров в гостинице, поиск подходящих билетов.

1.2 Разработка требований к программному обеспечению чат-бота

Для того, чтобы более детально понять потребности сотрудников и сформулировать требования, был проанализирован бизнес-процесс предоставления информации клиентам.

На данный момент большинство клиентов компании – частные лица, обращающиеся в ООО «УРАЛПРОМСТРОЙ» впервые. Они узнают контакты компании от знакомых или из рекламы, но не знают полного перечня услуг, условий оказания услуг, стоимости, графика работы. У компании нет сайта или страницы в соцсетях, где можно было бы узнать эту информацию. Поэтому прежде чем оставить заявку, потенциальные клиенты приходят в офис компании, звонят сотрудникам или связываются с ними в соцсетях, чтобы задать интересующие вопросы. Контекстная диаграмма IDEF0 данного процесса представлена на рисунке 1.

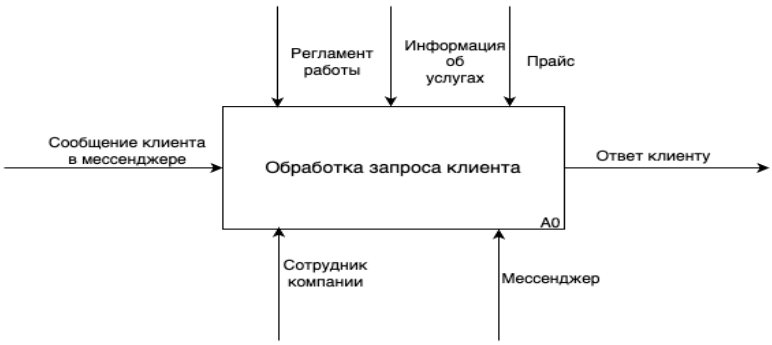

Рисунок 1 – IDEF0 диаграмма процесса «Общение с клиентами»

Для наглядности детальная декомпозиция блока A0 представлена на рисунке 2.

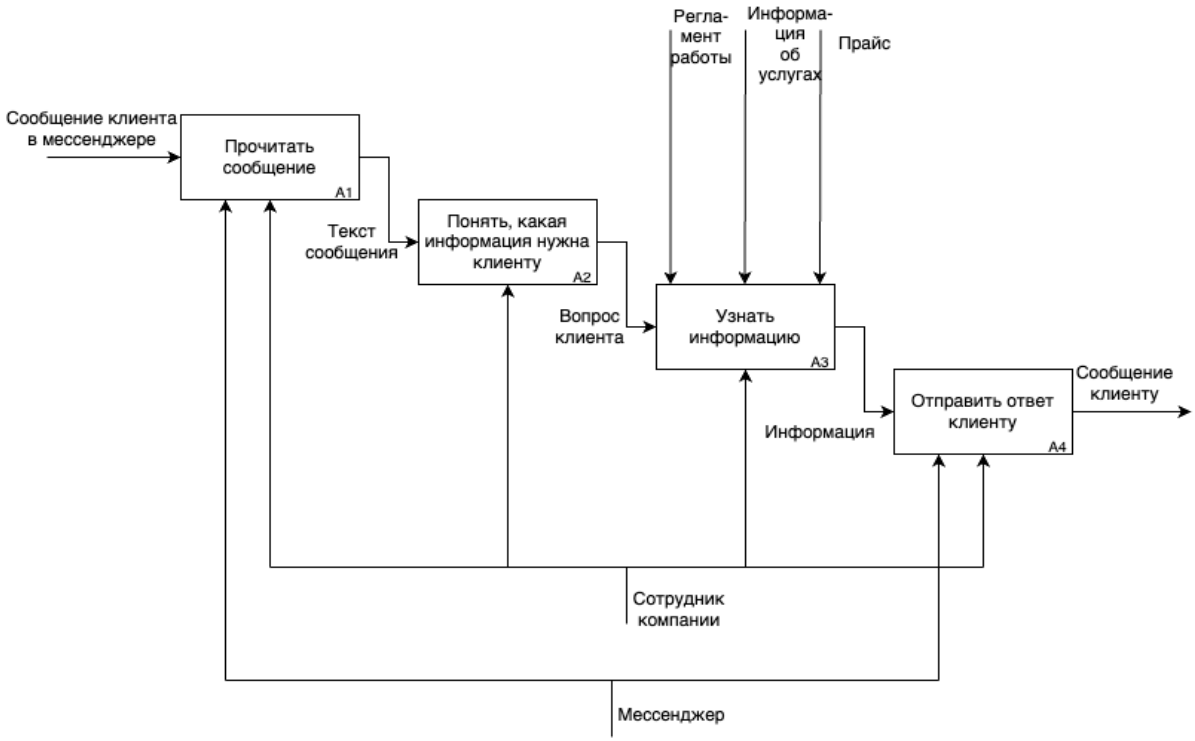

Рисунок 2 – Декомпозиция блока A0 процесса «Общение с клиентами»

На рисунке 3 представлена модель данного бизнес-процесса «Как есть» в формате BPMN.

BPMN – это нотация моделирования бизнес-процессов, которая является промежуточным звеном между формализацией/визуализацией и воплощением бизнес-процесса [5].

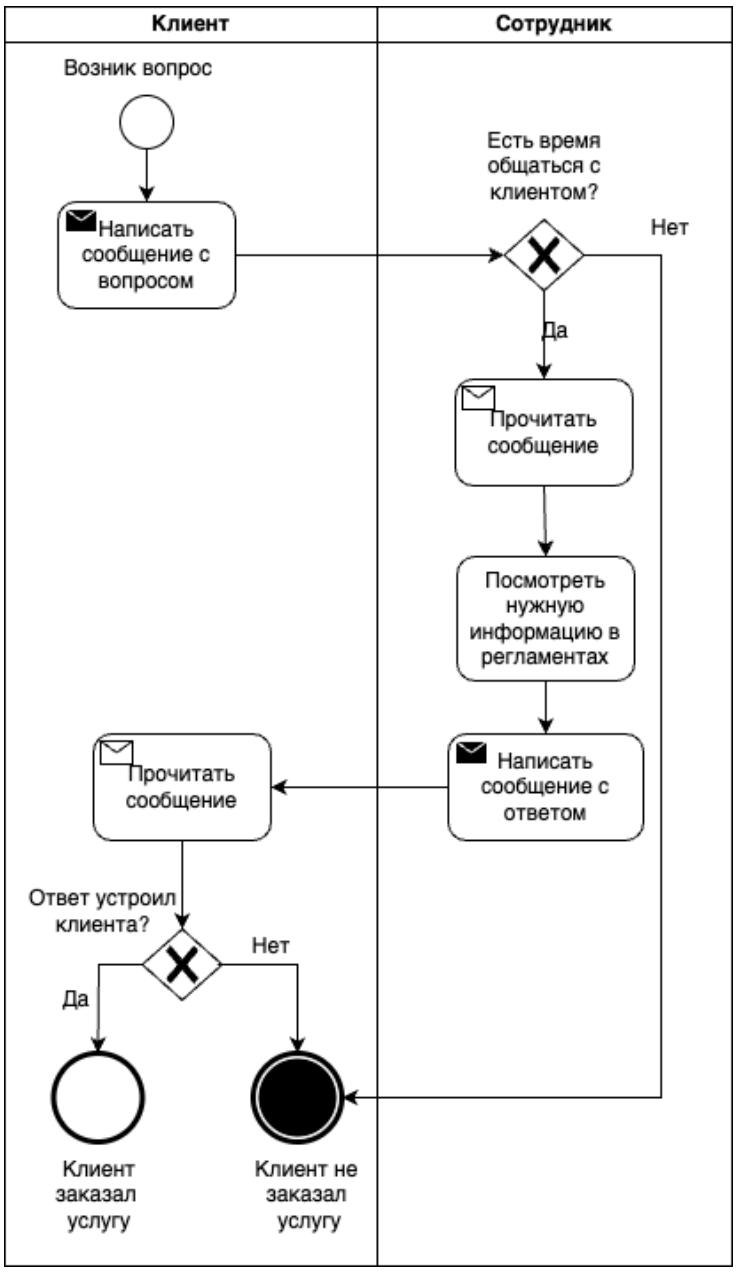

Рисунок 3 – Модель бизнес-процесса «Как есть»

Существующий процесс имеет существенные недостатки – низкую скорость обслуживания клиентов и большие временные затраты сотрудников. На общение с потенциальными заказчиками сотрудники компании тратят много времени. Для автоматизации этого процесса необходим чат-бот, который будет вести диалог с клиентами и рассказывать им об услугах, которые предоставляет ООО «УРАЛПРОМСТРОЙ».

Модель бизнес-процесса поддержки клиентов «Как должно быть» представлена на рисунке 4.

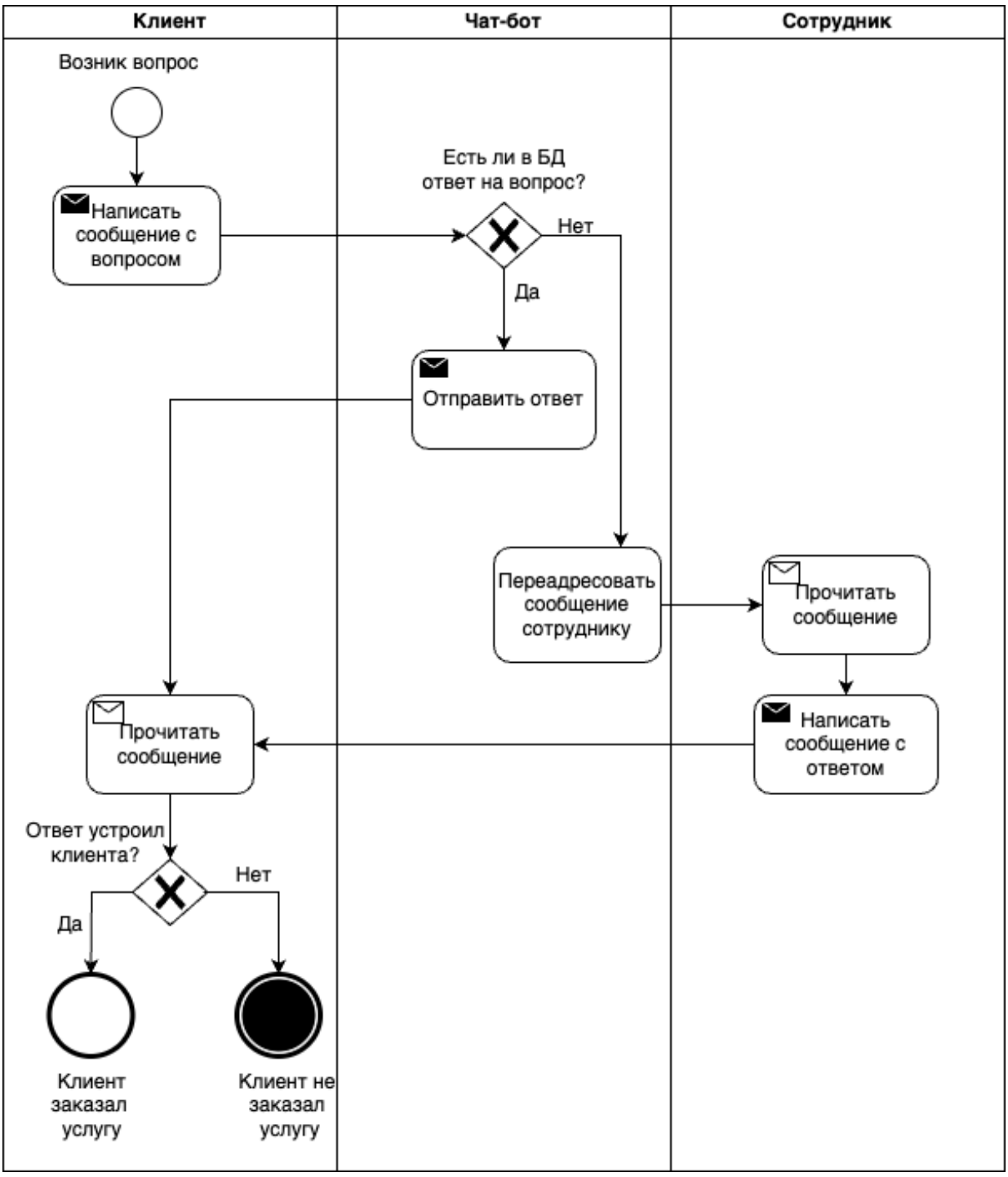

Рисунок 4 – Модель бизнес-процесса «Как должно быть»

Усовершенствованный бизнес-процесс общения с клиентами организован следующим образом:

- в базе данных хранится информация, которую клиенты спрашивают чаще всего;
- когда клиент начинает диалог, бот предлагает ему информацию из базы данных, организованную в удобном виде, которым можно управлять с помощью кнопок;
- с большой вероятностью клиент найдет ответ на свой вопрос в предложенной информации и вмешательство сотрудника не потребуется;
- если клиент не находит ответ на свой вопрос, бот предлагает ему оставить запрос для специалиста;
- клиент отправляет запрос и бот пересылает его сотруднику компании.

Таким образом, большая часть обращений обрабатывается без участия сотрудника. Вмешательство сотрудника требуется, только если вопрос нетипичный и ответа нет в базе данных.

Еще одной необходимой функцией чат-бота является возможность делать рассылки для клиентов (например, сообщать об акциях).

Ограничения, накладываемые спецификой работы компании – это низкая стоимость разработки и поддержки, простота поддержки и внесения изменения в программное обеспечение, отсутствие необходимости нанимать новый персонал.

Проанализировав бизнес-процесс и пожелания сотрудников ООО «УРАЛПРОМСТРОЙ», были сформулированы требования к программному обеспечения чат-бота, согласованные с главным инженером ООО «УРАЛПРОМСТРОЙ».

Для разработки требований к чат-боту используется методология FURPS+. FURPS+ – это одна из классификаций требований к

информационным системам, которая расшифровывается как Functionality (функциональность), Usability (удобство использования), Reliability, (надежность), Performance (производительность), Supportability, (поддерживаемость). Символ «+» говорит о том, что помимо требований есть и ограничения.

Требования к функциональности – это описание конкретных функций системы, которые нужно реализовать, чтобы пользователи могли с помощью этой системы выполнять свои задачи [4].

Удобство использования включает в себя логичность и понятность пользовательского интерфейса, эстетическую привлекательность.

Надежность определяет режим доступности системы, отказоустойчивость.

Производительность – это скорость работы системы, время отклика, число пользователей, которые могут работать с системой одновременно.

Поддерживаемость включает в себя удобство и простоту масштабирования, внесения изменений, тестирования.

Представим требования к чат-боту, сформулированные по методологии FURPS+, в таблице 1.

Таблица 1 – Требования к чат-боту по методологии FURPS+

Требование	Полезность
Функциональные требования	
Предоставление информации клиентам	критическое
Возможность оставлять заявки для связи со специалистом	критическое
Передача сотрудникам заявок, оставленных клиентами	критическое

Продолжение таблицы 1

Требование	Полезность
Функциональные требования	
Управление заявками (обработка, удаление)	важное
Редактирование информации в базе данных бота	важное
Рассылка сообщений клиентам	важное
Ограничение доступа: просмотр и обработка заявок, редактирование информации в базе и запуск рассылок должны быть доступны только для сотрудников	критическое
Удобство использования	
Удобное меню для управления ботом	важное
Современный привлекательный интерфейс	важное
Тексты составлены грамотно, не содержат ошибок	важное
Работа на базе популярного мессенджера	важное
Надежность	
Доступность 24/7	важное
Производительность	
Максимальное время реакции на запрос клиента: 30 секунд	важное
Поддерживаемость	
Возможность удобно и быстро изменять и добавлять тексты в базе данных чат-бота	важное
Ограничения	
Отсутствие необходимости нанимать новый персонал	критическое
Низкая стоимость разработки и поддержки	критическое
Простота поддержки и внесения изменений в функциональность ПО	критическое

Разработанный перечень требований является основой для проектирования программного обеспечения чат-бота.

1.3 Анализ аналогов разрабатываемого чат-бота

DIRECTUM Bot – программное обеспечение, которое позволяет с помощью чат-бота предоставить доступ пользователям к корпоративным сервисам [14]. Диалог с пользователями происходит в мобильном приложении Viber.

Программное обеспечение состоит из чат-бота в Viber, сервисов интеграции и обмена и компонентов DIRECTUM для разработки и настройки процессов.

Пример использования чат-бота на базе DIRECTUM показан на рисунке 5.

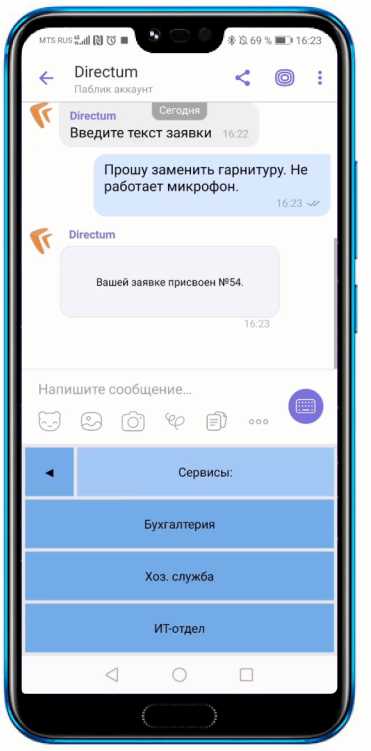

Рисунок 5 – Пример использования DIRECTUM

Недостатками DIRECTUM являются высокая сложность настройки и подключения, а также невозможность контролировать надежность и

безопасность бота, так как она полностью зависит от внешнего программного обеспечения.

Chatim – это чат-бот и онлайн-чат для повышения продаж на сайте [13]. С помощью него можно автоматизировать общение с клиентами, отвечать на популярные вопросы, использовать готовые сценарии для разных типов бизнеса.

На рисунке 6 показан пример использования Chatim.

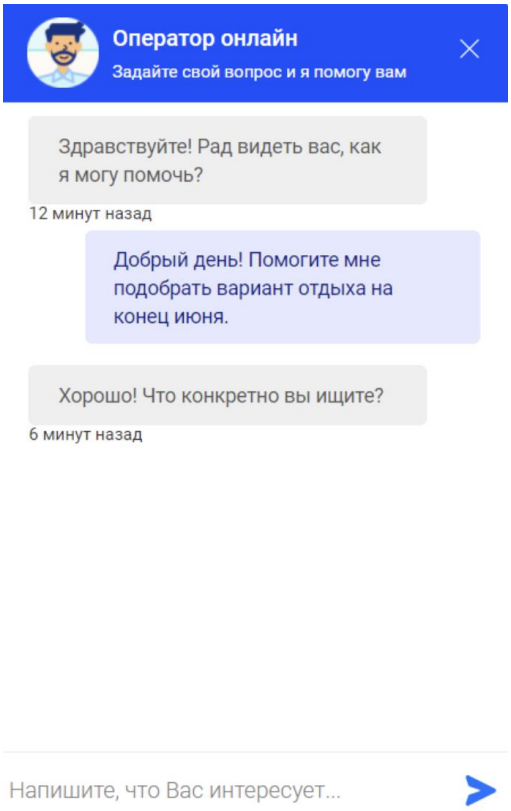

Рисунок 6 – Пример использования Chatim

Недостатками Chatim являются высокая стоимость и функционирование на базе сайта, а не мессенджера.

СберБизнесБот – платформа, с помощью которой можно подключать чат-ботов с искусственным интеллектом для автоматических ответов на вопросы клиентов [7]. Общение с пользователями может происходить на

сайтах или в мессенджерах. Данное программное обеспечение предоставляет много готовых сценариев обслуживания клиентов для разного бизнеса, в том числе, для строительства.

Пример использования чат-бота СберБизнесБот продемонстрирован на рисунке 7.

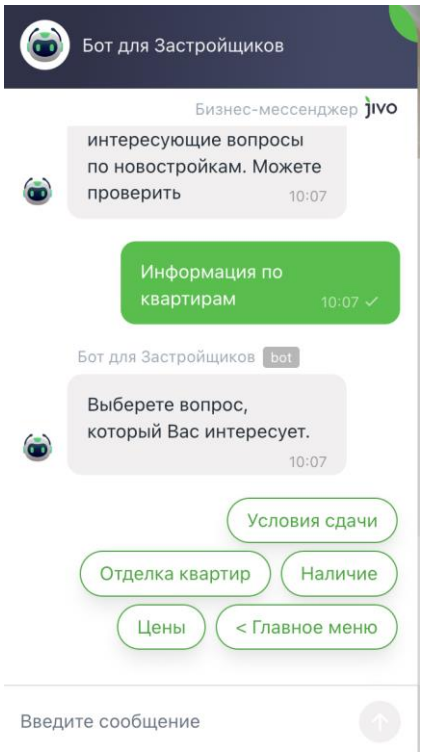

Рисунок 7 – Пример использования СберБизнесБот

Основным недостатком данного решения является высокая стоимость.

Проанализируем, соответствуют ли аналоги требованиям, которые были разработаны для чат-бота. Для этого составим таблицу (таблица 2).

Таблица 2 – Сравнительный анализ аналогов

Характеристика/балл (0-3)	DIRECTUM Bot	Chatim	СберБизнесБот
Предоставление информации клиентам	3	3	3
Возможность оставлять заявки для связи со специалистом	3	3	3
Рассылка сообщений клиентам	3	1	3
Работа на базе популярного мессенджера	2	0	3
Доступность 24/7	3	3	3
Максимальное время реакции на запрос клиента: 30 секунд	3	3	3
Низкая стоимость разработки и поддержки	1	1	1
Простота поддержки и внесения изменений в функциональность ПО	1	1	1

Анализ показал, что хоть аналоги и выполняют основную функцию – предоставление информации клиентам, но все они имеют недостатки, которые являются критичными для заказчиков. Поэтому есть необходимость в создании системы, которая будет лишена всех этих недостатков.

Выводы по главе 1

Был проанализирован бизнес-процесс общения с клиентами в ООО «УРАЛПРОМСТРОЙ», его недостатки и потребности сотрудников. На основании этого были составлены требования к программного обеспечения разрабатываемого чат-бота.

Был проведен сравнительный анализ аналогов, который показал, что они не соответствуют всем предъявляемым требованиям.

Глава 2 Проектирование программного обеспечения

2.1 Выбор технологий для реализации

2.1.1 Выбор мессенджера

В России используются в основном мессенджеры WhatsApp, Вконтакте, Telegram и Viber, поэтому бот должен работать на одной из этих площадок. Проанализируем эти мессенджеры и составим сравнительную таблицу (таблица 3).

Таблица 3 – Сравнительный анализ мессенджеров

Характеристика	WhatsApp	Вконтакте	Telegram	Viber
Популярность мессенджера растет	-	-	+	-
Есть удобные инструменты для создания ботов	-	+	+	-
Удобно использовать ботов (наличие кнопок, меню, команд и т.д.)	-	-	+	-

Для разработки бота была выбрана платформа Telegram. Telegram – это один из самых популярных и быстрорастущих мессенджеров [8]. Он был создан в 2013 году Павлом Дуровыми, известным также как основатель соцсети ВКонтакте. Telegram доступен для использования на различных платформах: Windows, Linux, MacOS, IOS, Android, также существует веб-версия.

Согласно исследованию Mediascope [3], в России Telegram стал быстро набирать популярность в 2022 году, а к марту 2023 года мессенджер вышел на

5 место среди российских интернет-ресурсов по объему дневной аудитории. Более 40% россиян пользуются Telegram ежедневно, более 60% – хотя бы раз в месяц.

Telegram отлично подходит для создания чат-ботов, так как имеет открытое Telegram API [18], которое могут использовать все разработчики. Существует даже специальное Telegram Bot API – отдельное API поверх Telegram API, предназначенное для создания ботов. Это позволяет разрабатывать чат-ботов любой сложности и реализовывать множество креативных идей.

2.1.2 Выбор способа создания бота

На данный момент есть два основных способа создать бота в Телеграме: с помощью кода и с помощью ноукод-платформ. В первом случае разработчик программирует бота на python, java или другом языке программирования, а затем загружает его на сервер. Во втором – собирает бота из готовых элементов в специальном конструкторе. Существует много конструкторов для создания ботов, например Manybot [15], PuzzleBot [16], Unisender [20], BotTap [12]. Все они похожи по функциональности.

К преимуществам ноукод-платформ относится простота использования и низкий порог входа. Создать бота можно быстро и это может сделать человек без навыков программирования. Поэтому такие платформы обычно используются, если нужно в сжатые сроки создать небольшого бота без сложной логики.

Однако такой подход имеет и недостатки. Во-первых, использование ноукод-платформ предполагает плату за создание бота, а также абонентскую плату в течении всего времени эксплуатации. Если бот будет использоваться долгое время, расходы могут оказаться большими.

Вторым недостатком является то, что функциональность и логика работы бота ограничены возможностями, которые предусмотрены

платформой. Поэтому некоторые функции может быть невозможно добавить в бота, созданного в конструкторе.

Еще одним недостатком является ненадежность ботов, созданных в ноукод-платформах. При проблемах в функционировании платформы бот может просто перестать работать.

Составим таблицу для сравнения подходов (таблица 4).

Таблица 4 – Сравнительный анализ подходов к созданию telegram ботов

Характеристика	С помощью кода	Ноукод-платформа
Простота и высокая скорость создания бота	-	+
Создание бота бесплатное	+	-
Можно реализовать любые функции, нет ограничений	+	-
Высокая надежность	+	-

Из-за серьезных недостатков ноукод-платформ выбор был сделан в пользу разработки бота на языке программирования.

2.1.3 Выбор языка программирования и базы данных

Telegram-бота можно разработать на Java, JavaScript, Go, Python и других языках программирования. Все они обладают нужными возможностями. Составим сравнительную таблицу для анализа нескольких популярных языков (таблица 5).

Таблица 5 – Сравнительный анализ языков программирования

Характеристика	Python	Java	JavaScript	Go
Область применения	очень обширная, различные приложения, наука, игры и другое	различные приложения, игры	веб-приложения	бэкенд
Типизация	динамическая	статическая	динамическая	статическая
Большое количество обучающих материалов в свободном доступе	+	+	+	-
Простота синтаксиса	+	-	-	-
Наличие удобных библиотек для разработки telegram-ботов	+	-	+	-

Был выбран Python [17]. Этот язык широко применяется в разработке веб-приложений и игр, машинном обучении, тестировании и в других областях. Согласно рейтингу TIOBE, на данный момент Python является самым востребованным языком программирования [19].

Python обладает несколькими преимуществами, благодаря которым хорошо подходит для реализации telegram-бота.

Во-первых, язык Python обладает простотой синтаксиса и визуального восприятия кода. Это позволяет реже обращаться к документации, писать меньше строк кода и быстрее разбираться в чужом коде. В результате достигается высокая скорость разработки.

Так как Python очень популярен, большое количество программистов знает этот язык. Можно найти много учебных материалов, статей, видео, обсуждений на форумах, посвященных тому, как решить ту или иную задачу

на Python. Это существенно упрощает решение проблем, возникающих в ходе разработки.

Еще одним плюсом Python является обширная библиотека, в которой содержатся оптимизированные и многократно используемые фрагменты кода для решения практически любых задач. В том числе, есть специальные библиотеки, предназначенные для разработки telegram-ботов.

Для взаимодействия с Telegram Bot API было решено использовать фреймворк aiogram [10]. Это асинхронный фреймворк, предназначенный для разработки чат-ботов. С помощью него можно создавать ботов со сложной логикой. К основным преимуществам фреймворка относятся асинхронность и удобная обработка состояний, позволяющая создавать ботов с многошаговыми диалогами.

В качестве библиотеки для работы с базой данных была выбрана SQLAlchemy. Ее достоинством является то, что библиотека позволяет при работе с базами данных вместо SQL-запросов использовать обычные объекты и их методы. SQLAlchemy поддерживает различные СУБД.

В качестве базы данных была выбрана SQLite. Это популярная СУБД с открытым исходным кодом и большим комьюнити. SQLite обладает несколькими преимуществами относительно других СУБД. Основным из них является простота и компактность – она занимает всего один файл и может встраиваться в приложения и работать автономно, без использования сервера.

2.2 Выбор технологии моделирования

Самым популярным способом создания логических моделей информационных систем является UML. Unified Modeling Language – это унифицированный язык моделирования, который используется для разработки диаграмм и схем. Существует много видов UML диаграмм, все их можно разделить на две основные группы: структурные и поведенческие [6].

Структурные диаграммы показывают иерархию компонентов и модулей, а также их взаимосвязи и то, как они взаимодействуют друг с другом. Примеры структурных диаграмм:

- диаграмма классов – визуально представляет классы системы их атрибуты, методы и взаимосвязи;
- диаграмма компонентов – демонстрирует разбиение программной системы на структурные компоненты;
- диаграмма развертывания – моделирует физическое развертывание системы.

Поведенческие диаграммы ориентированы не на структуру системы, на ее поведение. Они показывают функциональные возможности и демонстрируют, что и как должно происходить. Примеры поведенческих диаграмм:

- диаграмма вариантов использования – демонстрирует, какие функции разрабатываемой системы должны быть доступны каждой группе пользователей;
- диаграмма последовательности – показывает последовательность событий, происходящих в системе, в хронологическом порядке;
- диаграмма деятельности – визуализирует набор операций, которые должны быть выполнены для достижения цели.

Для логического моделирования программного обеспечения чат-бота могут помочь диаграмма вариантов использования и диаграмма последовательности. Диаграмма вариантов использования позволит описать функции и поведение системы, а диаграммы последовательности будут подробно визуализировать хронологический порядок действий.

Для проектирования архитектуры программного обеспечения будем использовать диаграмму компонентов. Она поможет представить сложную систему в более простом виде, разбив ее на компоненты.

2.3 Разработка логической модели

При создании любого программного продукта важно ответить на вопрос «кто и как будет пользоваться этим приложением?». Для этого используется диаграмма вариантов использования, которая показывает функциональность разрабатываемой системы, доступную разным категориям пользователей. Данная диаграмма представлена на рисунке 8.

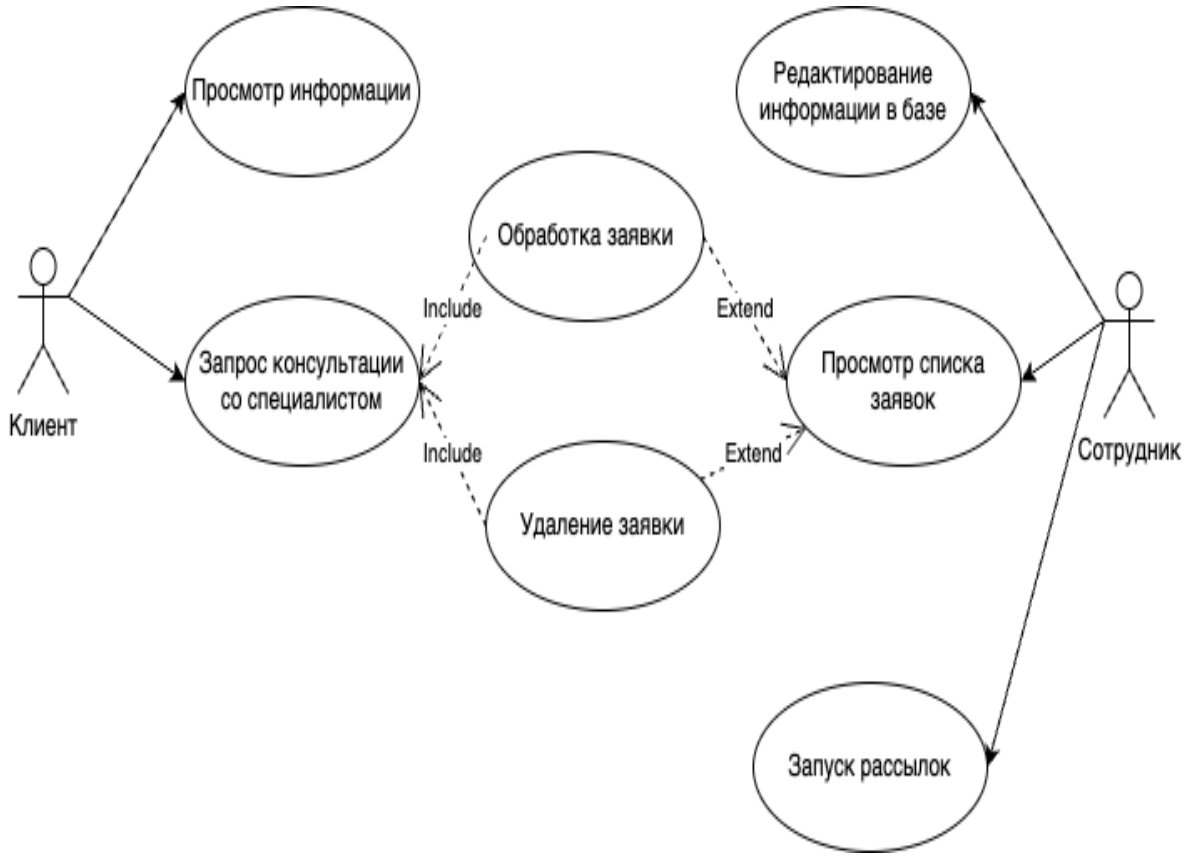

Рисунок 8 – Диаграмма вариантов использования чат-бота

Описание каждого прецедентов, изображенных на диаграмме вариантов использования, представлено в таблице 6.

Таблица 6 – Описание прецедентов

Прецедент	Описание
Просмотр информации	Клиент запрашивает информацию. Бот возвращает соответствующую информацию из базы данных
Запрос консультации со специалистом	Клиент вводит текст своего запроса. Бот сохраняет в базу данных запрос, состоящий из текста и контакта клиента
Редактирование информации	Сотрудник редактирует информацию. Бот обновляет данные в базе данных
Просмотр списка заявок	Сотрудник запрашивает список заявок. Бот выводит из базы данных все необработанные заявки. Заявка состоит из текста и контакта пользователя
Обработка заявки	Сотрудник нажимает на кнопку обработки заявки. Бот в базе данных устанавливает заявке статус Обработана
Удаление заявки	Сотрудник нажимает на кнопку удаления заявки. Бот в базе данных устанавливает заявке статус Удалена
Запуск рассылок	Сотрудник вводит текст для рассылки и запускает рассылку. Бот отправляет сообщение с заданным текстом всем пользователям, контакты которых есть в базе данных

Для уточнения диаграммы вариантов использования можно использовать диаграммы последовательности. Диаграммы последовательности визуально демонстрируют порядок взаимодействия, используя вертикальную ось диаграммы для представления времени.

На рисунке 9 представлена диаграмма последовательности просмотра информации клиентом.

Рисунок 9 – Диаграмма последовательности просмотра информации

Сценарий получения информации в чат-боте организован следующим образом:

- клиент, используя кнопки меню, выбирает, какая информация его интересует;
- бот берет нужную информацию из базы данных;
- бот отправляет клиенту сообщение с нужной информацией.

Диаграмма последовательности процесса работы с заявкой представлена на рисунке 10.

Рисунок 10 – Диаграмма последовательности работы с заявкой

Работа с заявкой происходит по следующему сценарию:

- клиент через чат-бот отправляет запрос на консультацию со специалистом;
- бот сохраняет запрос в базу данных;
- бот отправляет клиенту сообщение о том, что заявка принята;
- сотрудник через чат-бот получает список необработанных заявок;
- сотрудник консультирует клиента и в чат-боте отмечает, что заявка обработана;
- бот сохраняет в базу данных информацию о том, что заявка обработана.

На рисунке 11 изображена диаграмма последовательности процесса рассылок.

Рисунок 11 – Диаграмма последовательности процесса рассылок

- сотрудник через чат-бот вводит текст и запускает рассылку;
- бот берет из базы данных контакты клиентов;
- бот отправляет сообщение с текстом рассылки всем клиентам.

Построенные диаграммы позволяют увидеть порядок взаимодействия с системой.

2.4 Архитектура программного продукта

Работа чат-ботов строится состоит из 3 основных действий [2]:

- получение и вывод информации;
- распознавание намерения;
- обработка действия.

Получение и вывод информации в нашем случае будут происходить с помощью мессенджера telegram. Для хранения информации понадобится база данных. Таким образом, нужно реализовать:

- взаимодействие с API мессенджера telegram для получения сообщений, отправки ответов, обработки нажатия на кнопки и ввода команд;
- взаимодействие с базой данных;
- распознавание и анализ полученной информации (входящие сообщения, нажатия на кнопки, ввод команд);
- обработка различных действий, которые можно совершать с ботом (получение информации, обработка заявок, рассылки и так далее).

Для наглядного представления архитектуры информационных систем можно использовать диаграмму компонентов. Диаграмма компонентов – это структурная UML диаграмма, которая описывает особенности физического представления программного продукта. Компонентом в этой диаграмме называется часть системы, например класс, модуль, физическое устройство.

Диаграмма компонентов программного обеспечения разрабатываемого чат-бота представлена на рисунке 12.

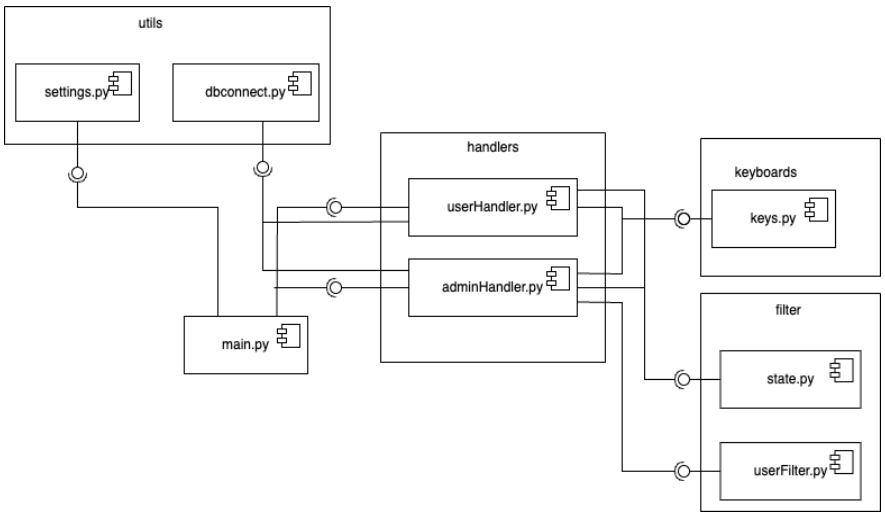

Рисунок 12 – Диаграмма компонентов чат-бота

В системе будет два типа пользователей – клиент (обычный пользователь) и сотрудник (пользователь с правами администратора). Поэтому нам потребуется два обработчика сообщений (хэндлера) – userHandler и adminHandler. Нам нужно уметь хранить состояния пользователей и определять, является ли пользователь администратором. Это будет происходить в модулях state.py и userFilter.py соответственно.

В боте будет меню с кнопками. Для работы с кнопками будем использовать модуль keys.py.

Настройки бота, такие как токен и список id администраторов, будем хранить в settings.py. А для подключения к базе данных понадобится модуль dbconnect.py.

Данные будут храниться в базе данных, состоящей из трех таблиц:

- таблица users для хранения контактов пользователей, которые будут использоваться для рассылок;
- таблица texts для хранения текстов сообщений;
- таблица requests для хранения заявок.

Диаграмма структуры БД представлена на рисунке 13.

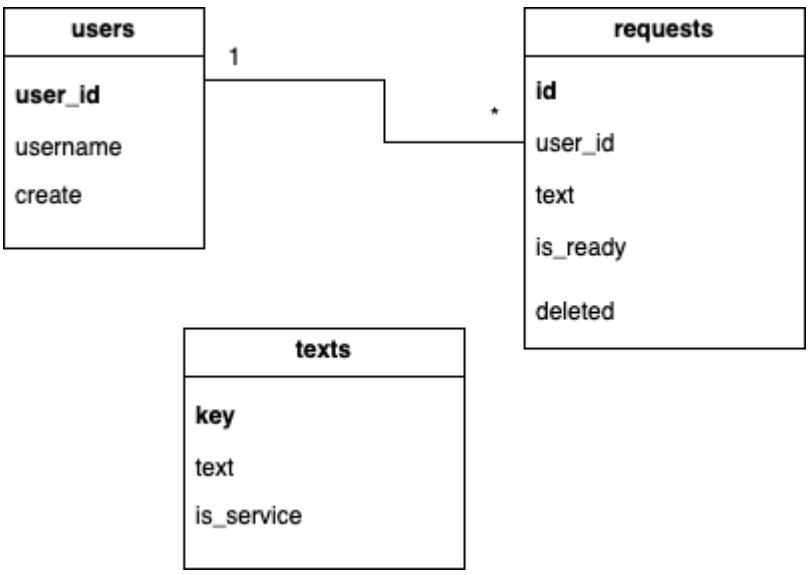

Рисунок 13 – Диаграмма структуры базы данных

Таблицы users и requests связаны друг с другом полем user_id.

Таблица texts с ними не связана, так как тексты, которые использует бот, зависят только от поступившего запроса и не зависят от пользователя, сделавшего этот запрос.

Выводы по главе 2

Были проанализированы и выбраны:

- наиболее подходящие мессенджер;
- способ создания бота, язык программирования и база данных.

Для наглядного представления архитектуры информационных систем можно использована диаграмма компонентов.

Были выбраны технологии моделирования, спроектирована логическая модель, архитектура, построены диаграммы вариантов использования, последовательности и компонентов.

Глава 3 Реализация и тестирование программного обеспечения

3.1 Регистрация чат-бота

Сначала необходимо зарегистрировать бота в Telegram и получить токен. С помощью этого токена можно будет взаимодействовать с ботом из программного кода. Для регистрации нужно обратиться к специальному боту BotFather [11]. BotFather – это основной бот в телеграме, который управляет остальными ботами. С помощью него можно создавать новых ботов и редактировать существующих. Основные функции BotFather:

- /newbot – создать нового бота,
- /mybots – редактировать своих ботов,
- /deletebot – редактировать своих ботов,
- /setname – изменить название бота,
- /setdescription – изменить описание бота,
- /setuserpic – изменить аватарку бота,
- /token – сгенерировать новый токен.

Для создания бота нужно запустить BotFather с помощью команды /start, а затем ввести команду /newbot и указать желаемое название. BotFather создает бота и генерирует для него уникальный токен. С помощью этого токена можно будет управлять ботом. Его нельзя размещать в публичном пространстве, так как если токен станет известен злоумышленникам, они смогут получить доступ к боту. Если токен все же стал кому-то известен, можно получить новый токен с помощью команды /token.

После регистрации бота можно приступить к разработке программной части.

3.2 Работа с базой данных

Для хранения данных используется база данных SQLite. База данных чат-бота состоит из трех трех таблиц:

- users – предназначена для хранения контактов пользователей, которые когда-либо взаимодействовали с ботом, содержит id пользователя, его контакт в telegram и дату добавления;
- requests – предназначена для хранения заявок, которые оставляют пользователи для консультации со специалистом, содержит id заявки, id пользователя, текст заявки и два булевых значения, указывающие, что заявка обработана и удалена;
- texts – предназначена для текстов сообщений, которые бот отправляет пользователям, содержит идентификатор текста, сам текст и булевое значение is_service, указывающее, что это описание услуги.

Работа с базой данных происходит с помощью библиотеки SQLAlchemy, которая позволяет взаимодействовать с реляционными базами данных с помощью ORM (Object-Relational Mapping). Технология ORM позволяет управлять базами данных не с помощью SQL, а с помощью объектов и их методов.

Например, чтобы в таблице texts обновить текст по ключу keyword используем следующее выражение (рисунок 14):

```
db.query(Text).filter(Text.key == keyword).update({Text.text: message.text})
```

Рисунок 14 – Обновление текста по ключу keyword в таблице texts

Аналогично происходят все взаимодействия с базой данных (получение и редактирование текстов, добавление новых заявок, получение списка заявок,

удаление и обработка заявок, добавление пользователей и так далее). Стоит отметить, что при удалении заявки она не удаляется из базы данных, а просто помечается как deleted=True. Это сделано для возможности восстановления, так как одним из требований была возможность восстановления в случае случайного удаления.

3.3 Разработка серверной части бота

Точка входа, из которой будет запускаться наш бот – это модуль main.py. Сначала требуется импортировать нужные модули и классы (рисунок 15):

```
import asyncio
import logging
from aiogram import Bot, Dispatcher
from core.utils.settings import BOT_TOKEN
from core.handlers import userHandler, adminHandler
```

Рисунок 15 – Импорт модулей и классов

Модуль asyncio нужен для асинхронной работы. logging – для логирования. Bot – это класс для взаимодействия с Telegram Bot API. Dispatcher – класс для обработки входящих событий. В константе BOT_TOKEN у нас хранится токен, который выдал BotFather. userHandler и adminHandler – обработчики входящих событий для обычного пользователя и для администратора.

Функция для запуска бота выглядит следующим образом (рисунок 16):

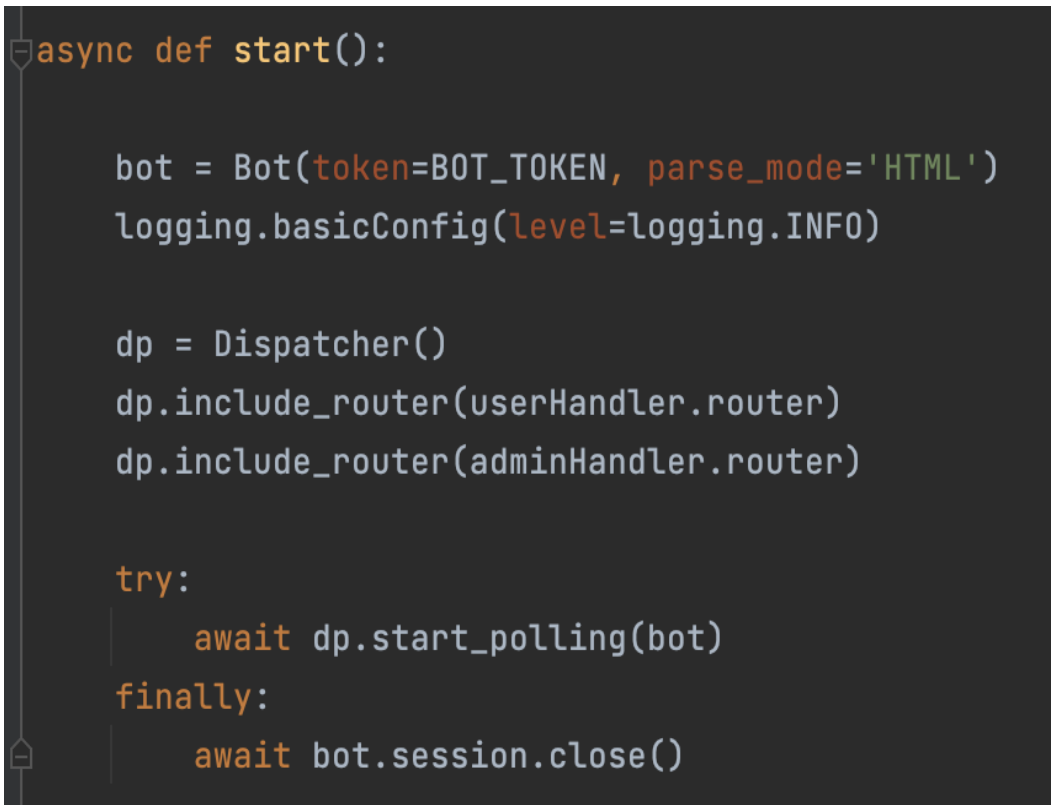

Рисунок 16 – Реализация функции запуска бота

Здесь создается объект класса Bot, настраивается логирование, создается диспетчер, к нему подключаются обработчики и происходит запуск бота.

Теперь напишем код в модулях userHandler.py и adminHandler.py. В них будут находиться функции входящих событий. Например, рассмотрим функцию для команды /start (рисунок 17):

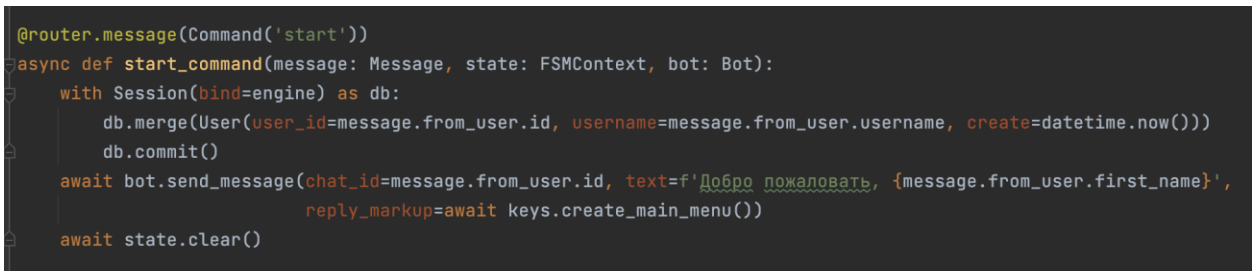

Рисунок 17 – Реализация обработчика команды /start

Декоратор @router.message указывает на то, что функция является обработчиком входящих событий. В декоратор передается фильтр Command('start'). Он означает, что данный обработчик используется только для события «Команда /start». В теле функции мы сначала сохраняем в базу данных нового пользователя, который вступил в диалог с ботом. После этого отправляем ему приветственное сообщение и меню. id и имя пользователя мы получаем из объекта message.

Похожим образом происходит обработка всех событий (сообщений, нажатий на кнопки, ввода команд) в модулях userHandler.py и adminHandler.py.

Бот должен уметь определять, кто с ним общается – администратор (в нашем случае – сотрудник компании) или обычный пользователь, так как функции администратора должны быть недоступны обычным пользователям. Для этого в функции модуля adminHandler.py в декоратор @router.message передается дополнительный фильтр UserFilter(). Этот класс определен в модуле userFilter.py. Он наследуется от BaseFilter и работает следующим образом: возвращает True, если id пользователя есть в списке администраторов, иначе возвращает False (рисунок 18).

```
class UserFilter(BaseFilter):
    def __init__(self):
        pass

    async def __call__(self, message: Message) -> bool:
        return True if message.from_user.id in ADMINS else False
```

Рисунок 18 – Реализация фильтра

Есть сценарии работы с ботом, состоящие из нескольких действий. Например, сценарий «Рассылка»:

- администратор говорит боту, что хочет провести рассылку,

– бот просит ввести текст рассылки,
– администратор вводит текст,
– бот отправляет введенный текст всем пользователям,
– бот сообщает администратору, что рассылка проведена.

Для корректной обработки таких сценариев требуется хранить состояние пользователя. Список возможных состояний хранится в модуле state.py, текущее состояние передается в декоратор функций-обработчиков.

3.4 Разработка клиентской части бота

Есть 3 основных способа взаимодействия с telegram ботами:
– отправка текстовых сообщений,
– отправка команд (например, /start),
– нажатие на кнопки встроенных клавиатур.

Большинство действий с нашим ботом будет происходить с помощью встроенных клавиатур. Поэтому в рамках разработки клиентской части требуется реализовать их.

Например, реализация клавиатуры для главного меню выглядит так (рисунок 19):

```
async def create_main_menu():
    builder = InlineKeyboardBuilder()
    builder.button(text='Услуги', callback_data=MenuFactory(keyword='services'))
    builder.button(text='О компании', callback_data=MenuFactory(keyword='company'))
    builder.button(text='Контакты', callback_data=MenuFactory(keyword='contacts'))
    builder.button(text='Прайс', callback_data=MenuFactory(keyword='price'))
    builder.button(text='Связаться со специалистом', callback_data=MenuFactory(keyword='callback'))
    builder.adjust(2, 2, 1)
    return builder.as_markup()
```

Рисунок 19 – Реализация клавиатуры для главного меню

Для создания клавиатуры используем билдер InlineKeyboardBuilder. Добавляем несколько кнопок, указав для них text и callback_data. Параметр callback_data нужен, чтобы обработчики событий понимали, какая именно кнопка была нажата. builder.adjust позволяет задавать расположение кнопок. В данном случае на первых двух строчках будет по две кнопки, а на последней – одна. То есть меню будет выглядеть так (рисунок 20):

Рисунок 20 – Главное меню

Аналогично реализованы все клавиатуры бота.

3.5 Тестирование

Тестирование – это проверка того, соответствует ли реальное поведение программы ожидаемому. Чтобы не забыть ничего проверить, перед началом тестирования составляются тест-кейсы или чек-листы. Тест-кейсы – это подробные инструкции с шагами, ожидаемыми результатами, тестовыми данными и другой информацией. Чек-листы – это списки необходимых проверок. В отличии от тест-кейсов, чек-листы гораздо компактнее, они содержат только названия проверок без подробной информации.

На основе требований к чат-боту был составлен чек-лист для его тестирования. Чек-лист представлен в таблице 7.

Таблица 7 – Чек-лист для тестирования чат-бота

Клиент
При запуске бота появляется приветственное сообщение и главное меню
При нажатии на кнопки «О компании», «Контакты», «Прайс» бот присылает сообщение с описанием
При нажатии на кнопку «Услуги» появляются кнопки с названиями услуг, при нажатии на них бот присылает сообщение с описанием
При нажатии на кнопку «Связаться со специалистом» есть возможность оставить заявку
Сотрудник
Сотруднику доступно меню для администраторов
При нажатии на кнопку «Заявки» появляются кнопки с необработанными заявками, каждую заявку можно посмотреть, пометить выполненной или удалить
При нажатии на кнопки «Редактирование ответов» и «Редактирование услуг» есть возможность редактировать сообщения, которые бот присылает клиентам
При нажатии на кнопку «Рассылка» есть возможность запустить рассылку
Пользователь, который не является сотрудником, не видит меню для администраторов

Преимущества чек-листов состоят в том, что их составление и поддержка в актуальном состоянии не требуют много времени.

3.5.1 Функциональность для клиентов

Протестируем функциональность для клиента по составленному чек-листу.

Для запуска telegram бота необходимо ввести команду /start. Видим, что при вводе данной команды бот присылает приветственное сообщение и меню (рисунок 21).

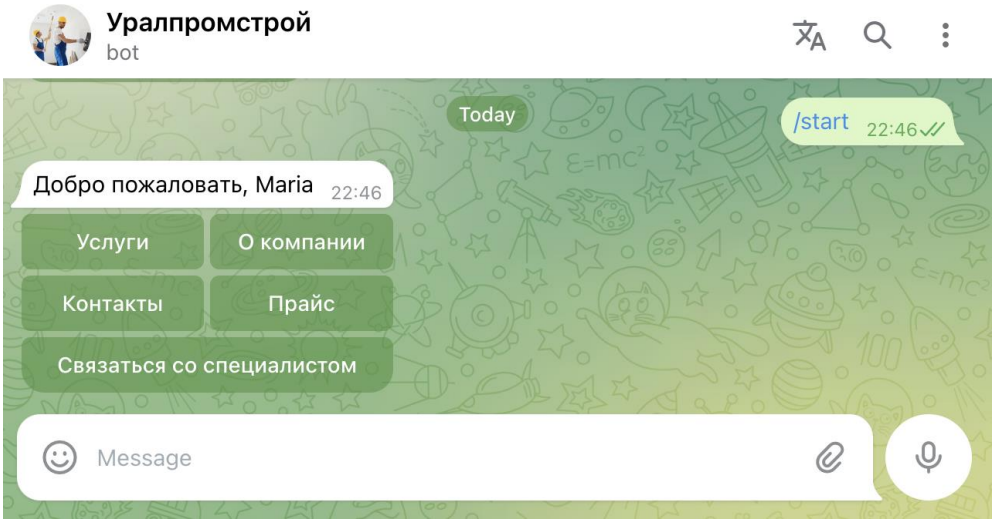

Рисунок 21 – Приветствие и меню

При нажатии на кнопки «О компании», «Контакты», «Прайс» бот присылает сообщение с описанием. Например, нажимаем на кнопку «Контакты» бот присылает сообщение с контактами (рисунок 22) и кнопку для возврата в главное меню.

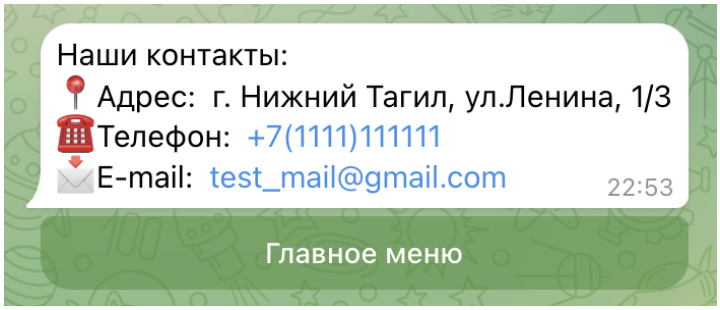

Рисунок 22 – Контакты

При нажатии на кнопку «Услуги» появляются кнопки с названиями услуг (рисунок 23).

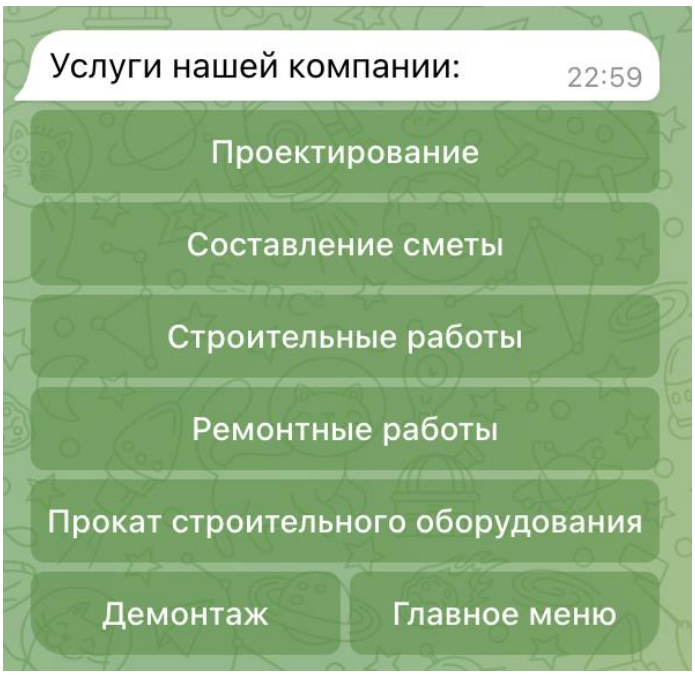

Рисунок 23 – Услуги

При нажатии на услугу бот присылает сообщение с ее описанием. Например, нажимаем на «Демонтаж» и получаем описание услуги демонтажа (рисунок 24).

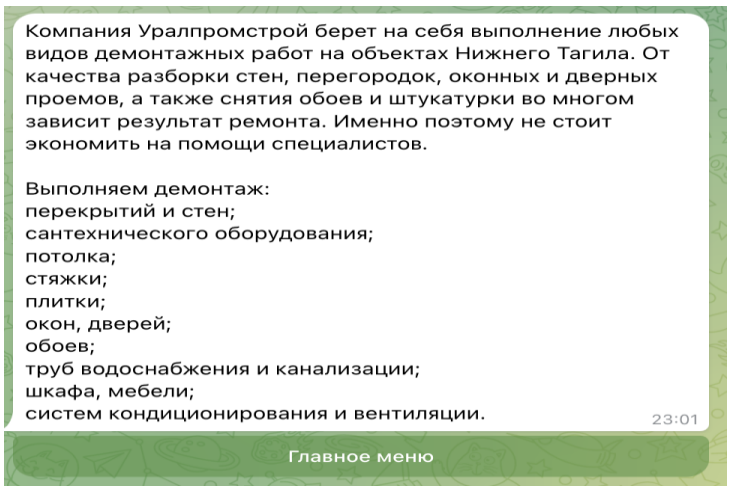

Рисунок 24 – Демонтаж

Если в главном меню нажать на «Связаться со специалистом», бот спрашивает, хочет ли клиент оставить заявку (рисунок 25).

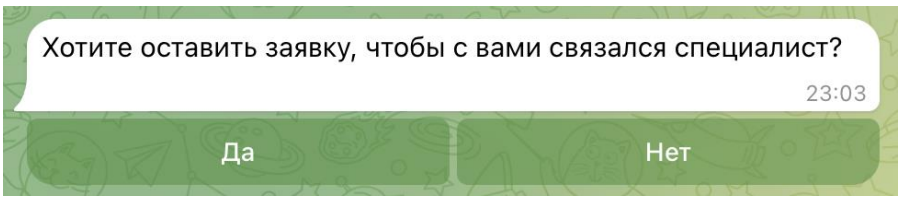

Рисунок 25 – Прием заявок

Нажимаем на «Да» и бот просит ввести вопрос (рисунок 26).

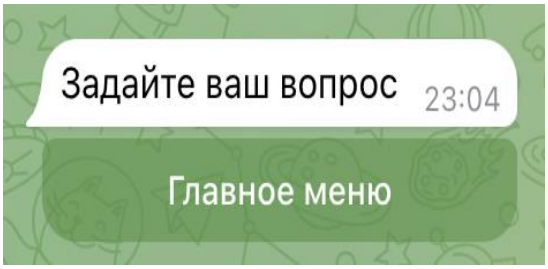

Рисунок 26 – Прием заявок

Вводим вопрос, бот оповещает, что обращение зафиксировано (рисунок 27).

Рисунок 27 – Прием заявок

Тестирование показало, что все функции для клиентов работают правильно.

3.5.2 Функциональность для сотрудников

Протестируем функциональность для сотрудника по составленному чек-листу.

Для получения доступа к меню администратора необходимо ввести команду /admin. Видим, что при вводе данной команды открывается меню администратора с кнопками: «Заявки», «Редактирование ответов», «Редактирование услуг», «Рассылка» (рисунок 28).

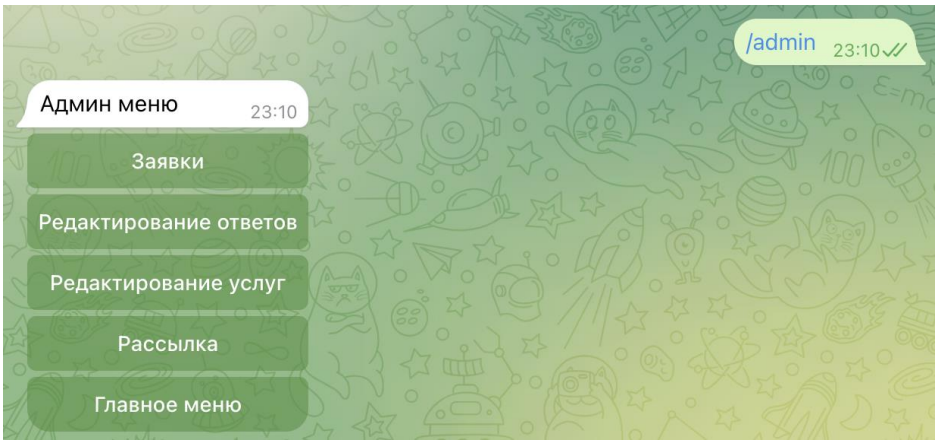

Рисунок 28 – Меню администратора

При нажатии на кнопку «Заявки» появляются кнопки с id необработанных заявок и кнопка для возврата в меню администратора (рисунок 29).

Рисунок 29 – Список заявок

Выберем последнюю заявку. Видим, что это наша заявка, которую мы создали при тестировании функциональности клиента. Есть возможность обработать или удалить заявку (рисунок 30).

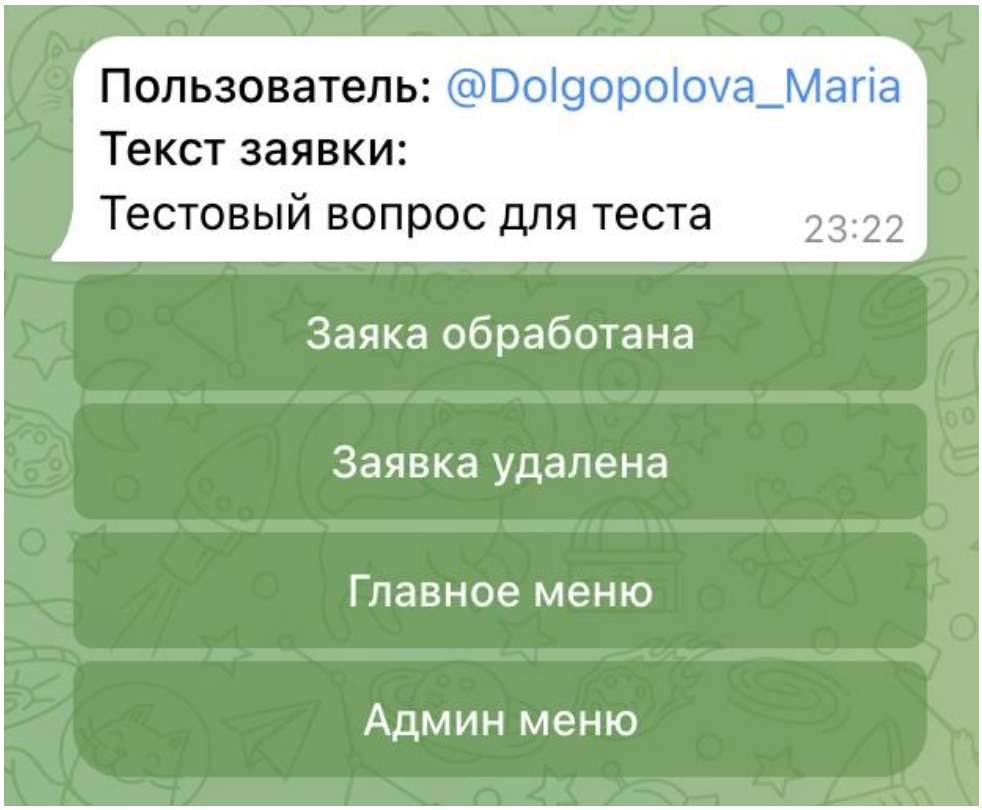

Рисунок 30 – Заявка

При нажатии на кнопку «Редактирование ответов» есть возможность редактировать сообщения, которые бот присылает клиентам. Сначала он спрашивает, какое именно сообщение мы хотим отредактировать (рисунок 31).

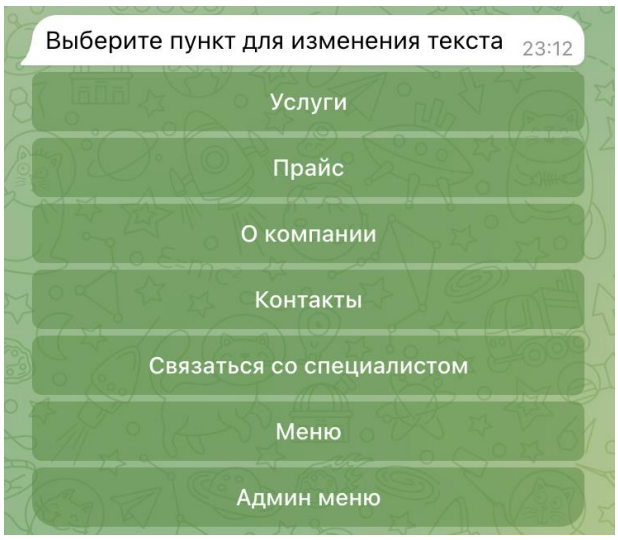

Рисунок 31 – Редактирование ответов

Выберем «Контакты» и отредактируем текст (рисунок 32).

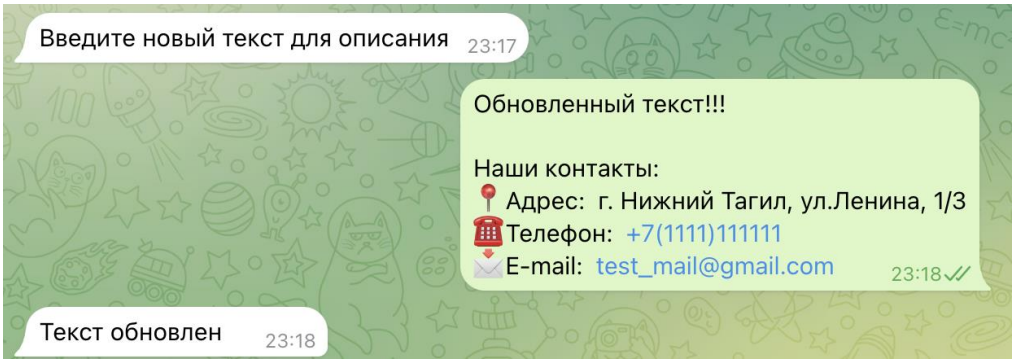

Рисунок 32 – Редактирование ответов

Теперь от имени клиента попросим бота показать информацию о контактах. Видим, что бот прислал обновленный текст (рисунок 33).

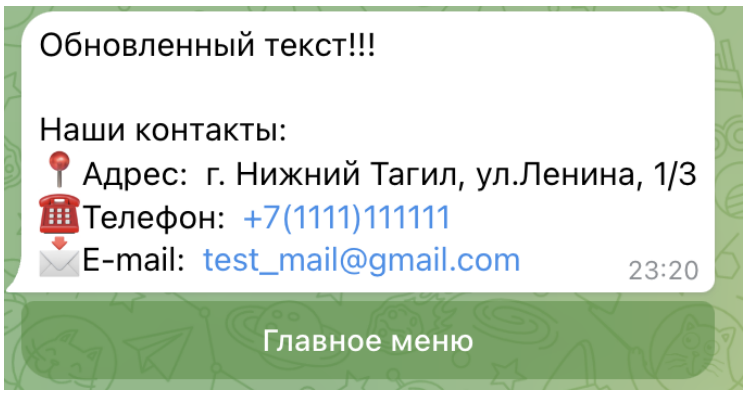

Рисунок 33 – Редактирование ответов

Аналогично работает кнопка «Редактирование услуг».

При нажатии на кнопку «Рассылка» бот просит ввести текст для рассылки. Вводим текст (рисунок 34).

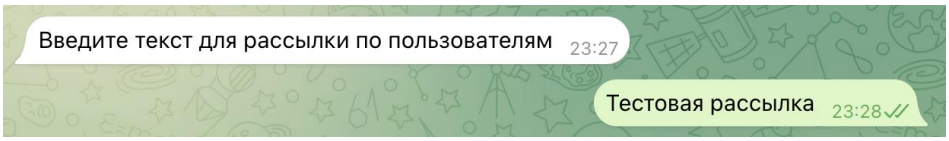

Рисунок 34 – Рассылка

Бот отправляет данный текст всем пользователям, контакты которых есть в базе данных (рисунок 35).

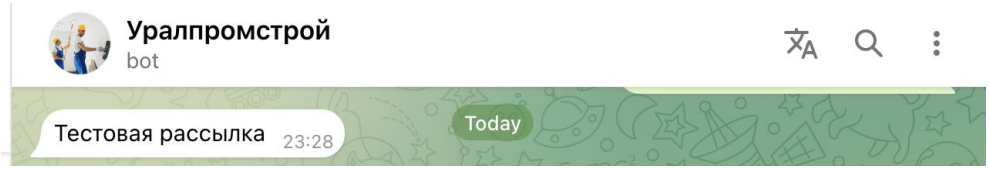

Рисунок 35 – Рассылка

Теперь проверим, что пользователи, которые не является сотрудниками, не видят меню для администраторов. Запустим бота из аккаунта пользователя, который не является сотрудником, и введем команду /admin. Ничего не происходит, меню администратора на появляется (рисунок 36).

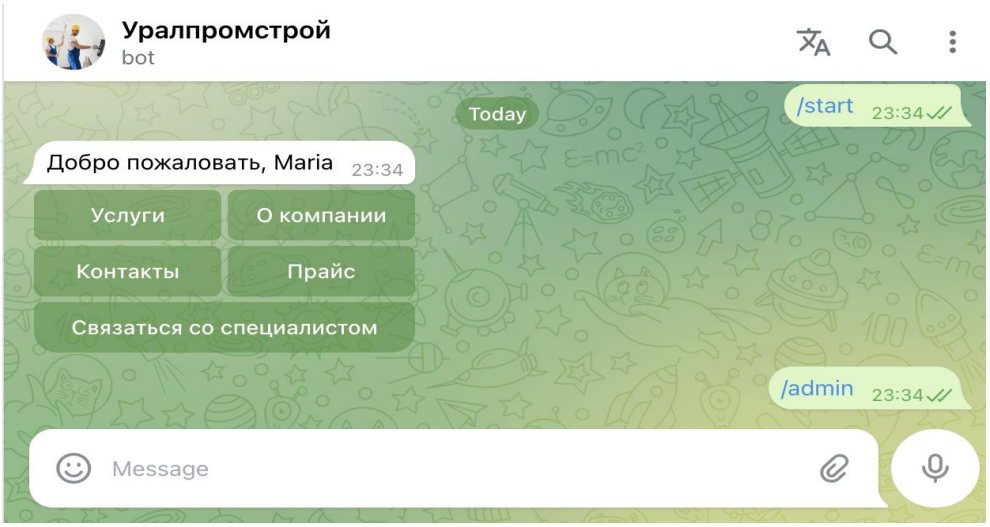

Рисунок 36 – Меню администратора не доступно обычным пользователям

Были проведены все проверки из чек-листа. Тестирование показало, что все функции работают правильно. Созданный чат-бот соответствует ожиданиям пользователей.

Выводы по главе 3

Было реализовано программное обеспечение чат-бота, включающее бэкенд, фронтенд и взаимодействие с базой данных.

Составлен чек-лист для тестирования и проведены все проверки из чек-листа. Тестирование показало, что все функции работают правильно.

Заключение

В рамках выпускной квалификационной работы было разработано программное обеспечение чат-бота для информационной поддержки клиентов строительной компании «УРАЛПРОМСТРОЙ».

Для достижения поставленной в работе цели были выполнены следующие задачи:

- проанализирован бизнес-процесс до внедрения чат-бота, построены IDEF0 и BPMN диаграммы;
- собраны и проанализированы требования к чат-боту;
- изучены функции чат-ботов и возможности их применения;
- изучены технологии, которые можно использовать при создании чат-ботов, и выбраны наиболее подходящие;
- спроектирован чат-бот. Построены диаграмма вариантов использования, диаграммы последовательности, диаграмма компонентов и диаграмма структуры базы данных;
- выполнена реализация чат-бота для мессенджера telegram на языке программирования python;
- реализованный бот протестирован. Тестирование подтвердило его соответствия требованиям заинтересованных сторон.

Таким образом, результатом ВКР является разработанный чат-бот для информационной поддержки клиентов строительной компании «УРАЛПРОМСТРОЙ».

Список используемой литературы

1. Автоматизация бизнеса [Электронный ресурс]. URL: https://www.kp.ru/guide/avtomatizatsija-biznesa.html (дата обращения: 05.01.2024).

2. Академия Selectel, Ботоводство [Электронный ресурс]. URL: https://selectel.ru/blog/botovodstvo/ (дата обращения: 29.12.2023).

3. Аудитория Telegram: исследование Mediascope [Электронный ресурс]. – URL: https://mediascope.net/news/1601603/ (дата обращения: 03.01.2024).

4. Вигерс К., Разработка требований к программному обеспечению. 3-е изд., дополненное / Битти Д., Вигерс К., Издательство «Русская редакция», 736 стр.

5. Нотация BPMN 2.0: ключевые элементы и описание [Электронный ресурс]. URL: https://www.comindware.com/ru/blog-нотация-bpmn-2-0-элементы-и-описание (дата обращения: 29.12.2023).

6. Руководство по UML-диаграммам и моделированию баз данных [Электронный ресурс]. URL: https://www.microsoft.com/ru-ru/microsoft-365/business-insights-ideas/resources/guide-to-uml-diagramming-and-database-modeling/ (дата обращения: 01.04.2024).

7. СберБизнесБот [Электронный ресурс]. URL: https://sberbb.ru/ (дата обращения: 01.04.2024).

8. Телеграм: социальная сеть или что-то другое? [Электронный ресурс]. URL: https://uchet-jkh.ru/i/telegram-socialnaya-set-ili-cto-to-drugoe/ (дата обращения: 03.01.2024).

9. Что такое чат-бот? [Электронный ресурс]. URL: https://www.oracle.com/cis/chatbots/what-is-a-chatbot/ (дата обращения: 29.12.2023).

10. aiogram [Электронный ресурс]. URL: https://aiogram.dev/ (дата обращения: 01.03.2024).

11. BotFather [Электронный ресурс]. URL: https://telegram.me/BotFather (дата обращения: 03.03.2024).

12. BotTap [Электронный ресурс]. URL: https://bottap.ru/ (дата обращения: 01.02.2024).

13. Chatim [Электронный ресурс]. URL: https://chatim.io/ (дата обращения: 01.04.2024).

14. DIRECTUM Bot [Электронный ресурс]. URL: https://directum.ru/blog-post/directum_bot__novoe_reshenie_dlja_cifrovogo_vzaimodejjstvija (дата обращения: 01.04.2024).

15. Manybot [Электронный ресурс]. URL: https://manybot.io/ (дата обращения: 01.02.2024).

16. PuzzleBot [Электронный ресурс]. URL: https://puzzlebot.top/ (дата обращения: 01.02.2024).

17. Python [Электронный ресурс]. URL: https://www.python.org/ (дата обращения: 04.01.2024).

18. Telegram APIs [Электронный ресурс]. URL: https://core.telegram.org/ (дата обращения: 03.01.2024).

19. TIOBE Index for December 2023 [Электронный ресурс]. URL: https://www.tiobe.com/tiobe-index/ (дата обращения: 04.01.2024).

20. Unisender (Chat-bot Telegram) [Электронный ресурс]. URL: https://www.unisender.com/ru/features/chat-bot/telegram/ (дата обращения: 01.02.2024).