1 слайд - Приветствие

Здравствуйте дорогие члены комиссии и слушвтели.  
  
Меня зовут Мацук Владислав Сергеевич  
  
Тема моей выпускной квалификационной работы: «Разработка программного виртуального помощника по работе с клиентами и заказами сервисного центра»

2 слайд - Актуальность

В условиях роста количества обращений в сервисные центры возрастает необходимость в автоматизации обработки заказов и взаимодействия с клиентами. **Дипломный проект выполнялся для** организации ООО «Термо-Мастер сервис», где внедрение виртуального помощника позволит сократить время отклика, повысить качество обслуживания и оптимизировать работу сотрудников. **ООО «Термо-Мастер сервис»** — это сервисная организация, специализирующаяся на обслуживании, ремонте кондиционеров и холодильного оборудования в сети супермаркетах **«Командор»** и **«Аллея», а также** частных лиц и организаций. Основные направления деятельности включают приём и обработку заказов от клиентов, проведение выездных работ, техническое консультирование, а также послегарантийное обслуживание оборудования. ООО «Термо-Мастер сервис» ориентировано на высокое качество сервиса, оперативное реагирование на обращения клиентов и надёжное техническое сопровождение оборудования.

3 слайд - Цель и задачи

**Цель:**

Разработать программного виртуального помощника для обработки клиентских обращений и заказов в сервисном центре.

**Основные задачи работы:**

1. Проанализировать текущий процесс работы с заказами в сервисном центре
2. Спроектировать архитектуру и логику работы виртуального помощника
3. Разработать программного виртуального помощника для автоматизации обработки заказов
4. Сделать выводы

4 слайд - Сравнительный анализ программного обеспечения

На данном слайде представлена таблица по сравнению программного обеспечения.

В таблице видны восемь ПО (программное обеспечение), в самом низу расположился мой виртуальный помощник, который выигрывает все предыдущие ПО (программное обеспечение) по поддержке ИИ (Есть (через OpenAI/API); Интеграции (Огромное количество API, Webhooks, CRM, таблицы); Сложности настройки (Средняя (no-code + логика)); Стоимости (Бесплатно (open-source)); Гибкости настройки (Очень высокая). А также мой ВП (виртуальный помощник) идеален для кастомных решений, и с его помощью можно получить мощную автоматизацию.

Данный виртуальный помощник был создан на базе n8n.io.  
  
5 Процесс выбора заказов до внедрения виртуального помощника

6 слайд - Процесс выбора заказов после внедрения виртуального помощника

На данном слайде представлена таблица по сравнению программного обеспечения.

В таблице видны восемь ПО (программное обеспечение), в самом низу расположился мой виртуальный помощник, который выигрывает все предыдущие ПО (программное обеспечение) по поддержке ИИ (Есть (через OpenAI/API); Интеграции (Огромное количество API, Webhooks, CRM, таблицы); Сложности настройки (Средняя (no-code + логика)); Стоимости (Бесплатно (open-source)); Гибкости настройки (Очень высокая). А также мой ВП (виртуальный помощник) идеален для кастомных решений, и с его помощью можно получить мощную автоматизацию.   
  
Данный виртуальный помощник был создан на базе n8n.io.

7 слайд – Виртуальный помощник

Разработка виртуального помощника велась на платформе **n8n.io**, которая позволяет создавать автоматические сценарии.

**Что такое n8n.io:**

1. Это **workflow-платформа (конструктор сценариев)**, позволяющий автоматизировать процессы;
2. Работает по принципу **no-code/low-code** — для настройки не требуется глубоких знаний в программировании;
3. Поддерживает **более 350 интеграций** (Telegram, Google Drive, OpenAI, CRM, email и др.);
4. Основана на визуальной сборке блоков — процессы строятся как схемы (что достаточно удобно).

**Почему выбрана:**

1. Позволяет **гибко обрабатывать входящие запросы**, соединять их с ИИ и отправлять результат в нужный канал (например, обратно в Telegram);
2. Поддерживает работу с ИИ, в том числе через **OpenAI, ChatGPT, Pinecone** и др.;
3. Позволяет легко **добавлять ветвления логики**, обработку ошибок, условия (if, switch), память (memory) и хранение данных;
4. Возможность **разворачивания на своём сервере** — полная автономия и безопасность;
5. Можно создавать **сценарии, которые масштабируются**: от простых уведомлений до сложной бизнес-логики.

8 слайд – Создание Telegram-бота для взаимодействия с виртуальным помощником

Для общения с виртуальным помощником был создан специальный **Telegram-бот**, который выполняет функцию основного интерфейса между клиентами и автоматизированной системой n8n. **Telegram-бот** — **э**то удобный и привычный интерфейс, через который клиент или заказчик может оформлять заказы, получать статус заказов, задавать вопросы и взаимодействовать с системой.

Само создание Telegram-бота выполнялось с помощью **BotFather** — это официальный бот в Telegram, через которого создаются и настраиваются все остальные боты. При создании нового бота, **BotFather выдаёт уникальный API-токен**, который используется для подключения к внешним системам, также можно настроить имя, описание, аватар, команды и многое другое.

### Реализация взаимодействия:

1. Через BotFather был создан Telegram-бот и получен **API-токен**;
2. Этот токен был использован в платформе **n8n.io**, чтобы связать Telegram-бота с внутренней логикой помощника;
3. В n8n был создан специальный блок **Telegram Trigger**, который принимает входящие сообщения от клиентов;
4. После обработки сообщений с помощью ИИ и логики, ответ возвращается обратно в **Telegram**.

9 слайд – Архитектура взаимодействия с виртуальным помощником

(Клиент / Заказчик) заходит в Telegram-бота (это интерфейс для общения с виртуальным помощником, так как у него нет определённых команд, и он работает на основе интеграции с искусственными интеллектуальными (AI) моделями), после чего (Клиент / Заказчик) составляет (оформляет) заказ, и он поступает в (Виртуального помощника), если же (Клиент / Заказчик) вносит корректировки или отменяет заказ, то вся информация автоматически попадает в виртуального помощника. Если же заказ был сделан, то виртуальный помощник:   
1. Получение данных от Telegram-бота (через Webhook).

2. Создание и запись данных в систему (новый заказ,

данные клиента / заказчика, статус и т.п.).

3. Проверка доступности запчастей (интеграция со складской

системой).

4. Назначение свободного инженера (по расписанию и компетенциям).

5. Отправка уведомлений всем участникам через Telegram-бот:

· Клиенту / Заказчику — о приёме заказа, назначении инженера,

выезде, завершении;

· Инженеру — новый заказ, адрес, время, детали;

· Менеджеру — сводка действий (по запросу или при

отклонениях).

6. Контроль выполнения:

· Автоматические напоминания;

· Фиксирование завершения;

· Обновление статуса.

Также вся нужная информация передаётся инженеру, и он в свою очередь подтверждает выполнение заказа или отмечает что заказ выполнен (только после выполнения), вся информация также передаётся обратно в виртуального помощника. Также менеджер, который следит за работой, может попросить сводку данных, и они моментально придут ему. Так как у самого виртуального помощника есть “мозги - память” (это его хранилище данных), с которой он взаимодействует во всех процессах. Также вся информация об оповещениях поступает всем участникам системы.

10 слайд – Блок-схема алгоритма поставки запчастей

Для клиента:

“Начало”, нам “поступает заказ от клиента”, после чего нам нужно узнать “требуется ли запчасти?”, если “нет”, то “конец”, если “да”, то уточняем для чего нужна запчасть “для кондиционеров?”, если “да”, то идёт “поиск запчастей для кондиционеров” и после чего “конец”, а если “нет”, то идёт “поиск запчастей для холодильного оборудования”, после чего “конец”.

Для заказчика:

“Начало”, нам “поступает заказ от заказчика”, после чего мы проводим “анализ заказа”, после чего мы проводим “поиск запчастей на складе или магазине”, затем мы узнаём “если ли запчасть на складе?”, если “нет”, то направляется “заявка в магазин запчастей”, после чего производится “доставка запчастей к заказчику”, потом “конец”, а если “да”, то мы узнаем “далеко объект от слада?”, если “нет”, то “сообщаем инженеру, что нужно приехать на склад”, поле чего “конец”, а если “да”, то идёт “создание заявки курьеру”, после чего производится “доставка курьером запчасти к заказчику”, а потом “конец”.

11 слайд – Блоки виртуального помощника по работе с заказами

На данном слайде представлена первая часть блока, отвечающая за нашу информацию, которую мы заранее подгрузили в векторную базу данных нашего виртуального помощника.

Говоря о внутренних блоках, то

* Подключение к хранилищу данных осуществлялось через **Google Drive**;
* Загрузка и обработка текстов с помощью **OpenAI Embeddings** и **Recursive Text Splitter**;
* Для сохранения в **векторную базу данных использовался Pinecone**, что позволяет ИИ эффективно анализировать запросы;
* А подключение моделей OpenAI было сделано для генерации ответов.

12 слайд – Блоки виртуального помощника по работе с заказами

На следующем слайде представлена вторая часть блока, которая отвечает за подключение нашего виртуального помощника (который будет общаться с клиентами, заказчиками, инженерами, менеджером) к заранее созданному Telegram-боту (который выступает в роли мессенджера), который в свою очередь будет отвечать исходя из запросов пользователя.

* Как уже ранее говорил, блок **Telegram Trigger**, позволяет принимать сообщения от бота, которые были в него отправлены;
* Далее через блоки **If/Switch** определяется суть запроса и ветка его обработки;
* Подключается **AI Agent**, который с учётом логики и базы знаний формирует ответ;
* Используются **OpenAI Chat Model**, блок памяти (**Simple Memory**) и подключение к Pinecone;
* После обработки, результат возвращается обратно в Telegram через **выходной блок Telegram Message**.

13 слайд – Схема дополнительных возможностей для ООО «Термо-Мастер сервис»

А сейчас представлена схема дополнительных возможностей для ООО «Термо-Мастер сервис».

1. Голосовое управление и распознавание речи

**Описание:** Пользователем отправляется голосовое сообщение, затем оно распознаётся, после чего из текста формируется заказ.

**Используемые блоки такие как:**

* **Webhook** — точка входа, куда попадает аудиофайл от клиента (например, с Telegram-бота или сайта).
* **HTTP Request** — отправляет аудиофайл в внешний API для распознавания речи (например, Whisper, Google Speech-to-Text).
* **Set** — создаёт структуру заказ: вставляет распознанный текст, дату, статус.
* **IF / Switch** — определяет по ключевым словам, что именно нужно клиенту (кондиционер, ремонт и т.д.).

1. Автоматическая генерация отчётов

**Описание:** Система по расписанию отправляет сводки (например, каждую пятницу — сколько заказов, по какому оборудованию и тому подобное).

### Используемые блоки:

* **Cron** — задаёт периодичность (ежедневно, еженедельно).
* **Google Sheets** — собирает и читает заявки.
* **Function** — обрабатывает, агрегирует данные: сколько было заказ, какие типы и т.д.
* **Email** или **Telegram** — отправляет отчёт получателю (руководителю, тебе, клиенту).

1. Геолокация заявок и выездов

**Описание:** При создании заказа добавляется координата адреса — для карты или маршрута мастера.

### Используемые блоки:

* **HTTP Request** — используется для запроса координат у API геокодирования (например, Google Maps, Яндекс Гео).
* **Function** — форматирует адрес клиента в корректную строку запроса.
* **Set** — сохраняет полученные координаты.
* **Google Sheets** / **Firebase** — хранит заказы вместе с геоданными.
* Дополнительно можно отправлять координаты мастеру через **Telegram**.

1. Интеграция с CRM или телефонией

**Описание:** Автоматическая синхронизация с AmoCRM, Bitrix24 и автоматические звонки клиентам.

### Используемые блоки:

* **HTTP Request** — обращение к REST API CRM или телефонии.
* **Webhook** — для приёма новых лидов или заказов из CRM-системы.
* **IF / Set** — проверка данных, форматирование, добавление полей (например, ID клиента из CRM).
* **Email** или **Telegram** — уведомление оператору о новом лидe или необходимости созвона.

1. Автогенерация договоров и PDF-документов

**Описание:** Автоматически формируются договора, счета, акты — и отправляются клиенту.

### Используемые блоки:

* **HTTP Request** — обращение к сервису генерации PDF (например, PDFMonkey, DocSpring, Google Docs API).
* **Function** — сбор данных для шаблона документа (имя клиента, адрес, сумма, дата).
* **Email** / **Telegram** — отправка документа клиенту или мастеру.

14 слайд – Экономическая эффективность

**Затраты на реализацию проекта:**

1. Затраты проектировщику – 20 000₽
2. Затраты разработчику – 50 000₽
3. Лицензия – 500₽

**Итоговые затраты:** **70 500₽  
  
  
Потенциал оптимизации затрат:**

1. Среднемесячные затраты на выполнение задач менеджером: 50 000₽
2. Годовая экономия за счёт автоматизации процессов: 600 000₽

**Потенциал оптимизации затрат:**

1. Инвестиции в автоматизацию процессов с аналогичной рыночной стоимостью: 100 000₽
2. Предполагаемый срок окупаемости: 2 месяца

15 слайд – Выводы

**Цель достигнута:**  
Разработан программный виртуальный помощник для обработки клиентских обращений и заказами в сервисном центре.

**Все задачи выполнены:**

1. Проанализирован текущий процесс работы с заказами в сервисном центре
2. Спроектирована архитектура и логика работы виртуального помощника
3. Разработан программный виртуальный помощник для автоматизации обработки заказов
4. Сделаны выводы

16 слайд – Завершение

Спасибо за внимание!