**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**федеральное государственное автономное образовательное   
учреждение высшего образования   
«Санкт-Петербургский политехнический университет Петра Великого»**

**ИНСТИТУТ СРЕДНЕГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ**

|  |  |
| --- | --- |
|  | **Допустить к защите**  Заместитель директора по  учебно-методической работе  \_\_\_\_\_\_\_\_ Е.Г. Конакина\_\_\_\_\_  (Подпись) (И.О.Ф.)  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_2023 г. |
|  |

**Выпускная квалификационная работа**

Тема Разработка умного помощника для рестораторов

специальность 09.02.07 группа 42919/5

Студентка Постникова Л.В.

(подпись) (ФИО)

Руководитель Стариков А.Ю

(подпись) (ФИО)

Санкт-Петербург

2023

СОДЕРЖАНИЕ

[Введение 4](#_Toc136897907)

[1 Общая часть 5](#_Toc136897908)

[1.1 Анализ предметной области 5](#_Toc136897909)

[1.2 Анализ рынка существующих решений 6](#_Toc136897910)

[1.3 Анализ СУБД 7](#_Toc136897911)

[1.4 Обоснование и выбор методики, технологии и инструментальных средств разработки 8](#_Toc136897912)

[2 Специальная часть 9](#_Toc136897913)

[2.1 Постановка задачи 9](#_Toc136897914)

[2.1.1 Основания для разработки 10](#_Toc136897915)

[2.1.2 Наименование программы 10](#_Toc136897916)

[2.1.3 Назначение разработки 10](#_Toc136897917)

[2.1.4 Требования к функциональным характеристикам 10](#_Toc136897918)

[2.1.5 Требования к надежности 11](#_Toc136897919)

[2.1.6 Требования к условиям эксплуатации 11](#_Toc136897920)

[2.2 Описание схем 11](#_Toc136897921)

[2.2.1 Бизнесс процесс 11](#_Toc136897922)

[2.2.2 Инфологическая модель 12](#_Toc136897923)

[2.3 Написание чат-бота на Python 15](#_Toc136897924)

[2.3.1.1 Функция для обработки команды 16](#_Toc136897925)

[2.3.1.2 Создание кнопок на клавиатуре 16](#_Toc136897926)

[2.4 Руководство по управлению чат-ботом для официанта 18](#_Toc136897927)

[2.5 Руководство по управлению чат-ботом для ресторатора 21](#_Toc136897928)

[3 Экономическая часть 25](#_Toc136897929)

[3.1 Область применения программного продукта и его преимущества перед аналогичным программным продуктом 25](#_Toc136897930)

[3.2 Трудоемкость разработки программного продукта, квалификация исполнителя и его оклад 26](#_Toc136897931)

[3.3 Расчет затрат на разработку 28](#_Toc136897932)

[3.3.1 Затраты на оплату труда разработчику (Зот), руб. 30](#_Toc136897933)

[3.3.2 Затраты по страховым взносам (Зст), руб. 30](#_Toc136897934)

[3.3.3 Затраты по содержанию Эвм (Зэвм), руб. 31](#_Toc136897935)

[3.3.4 Затраты на расходные материалы 31](#_Toc136897936)

[3.3.5 Затраты по накладным расходам (Знр) 32](#_Toc136897937)

[3.4 Расчет цены и прибыли 32](#_Toc136897938)

[4 Техника безопасности и охрана труда 35](#_Toc136897939)

[4.1 Анализ условий труда программиста 35](#_Toc136897940)

[4.2 Расчёт искусственного освещения в помещении 36](#_Toc136897941)

[4.3 Электробезопасность на предприятии 38](#_Toc136897942)

[4.4 Пожарная безопасность на предприятии 40](#_Toc136897943)

[Заключение 43](#_Toc136897944)

[Список использованных источников 44](#_Toc136897945)

[Приложение А (справочное) Исходный код 45](#_Toc136897946)

# ВВЕДЕНИЕ

Telegram — программа для обмена сообщениями. Ее суть такая же, как у WhatsApp, Skype или Viber. Приложение может быть установлено на любые платформы и устройства, от настольных компьютеров до смартфонов. Работает идеально даже при очень плохом интернет-соединении, расход трафика при его использовании — минимальный.

Одной из особенностей Telegram13 является платформа для создания ботов. Это аккаунты, которые управляются программно, а не людьми. Они обладают функцией искусственного интеллекта.

Бот общается с пользователями в чате Telegram и может выполнять любые полезные действия, которые мы в него заложим.

Благодаря своей функциональности боты очень интересны бизнесу. Фактически, они являются своеобразными сотрудниками, которые отлично, без сбоев выполняют свою работу: принимают заказы, консультируют, продают, записывают на услугу, обрабатывают заявки и т.д. Причем делают это 24 часа в сутки, 7 дней в неделю, без праздников, выходных и даже перерывов.

Именно поэтому рынок Telegram-ботов сегодня очень активен. Их разработка — это тренд, который уже закрепился и устоялся. Они активно используются сейчас, а в будущем станут применяться еще шире и чаще.

## Общая часть

## Анализ предметной области

Начнем с изучения существующих решений для автоматизации деятельности работников в ресторане. Некоторые рестораны используют для этого специальные приложения, которые предоставляют работникам возможность выполнять свою работу в электронном виде. Однако не все рестораны имеют ресурсы или желание разрабатывать свои собственные программы и технологии. В данном случае использование чат-бота может оказаться более привлекательным вариантом, поскольку нет необходимости в разработке отдельного приложения.

Важным аспектом является сравнение различных чат-ботов, которые могут быть использованы для разработки умного помощника для рестораторов. Для этого необходимо изучить инструменты для создания чат-ботов и сравнить функционал, который они предоставляют. Рассмотреть достоинства и недостатки каждого инструмента, а также определить, какие из них наиболее подходят для решения поставленной задачи.

Для того, чтобы определить преимущества и недостатки чат-ботов в ресторанном бизнесе, необходимо изучить и проанализировать опыт использования данной технологии в других ресторанах. Рассмотреть, какие функции чат-боты могут выполнять для оптимизации работы ресторана и повышения удобства для работников. Анализ должен учесть, как позитивные, так и негативные аспекты использования чат-ботов, исходя из опыта и отзывов пользователей.

В результате проведенного анализа предметной области в дипломной работе будет разработан и реализован чат-бот для ресторана, позволяющий работникам удобно выполнять свою работу. Бот будет сравнен с существующими решениями, проведен анализ его преимуществ и недостатков, а также оценена эффективность его использования в ресторанном бизнесе.

## Анализ рынка существующих решений

Анализ рынка существующих решений для дипломной работы на тему «Разработка умного помощника для рестораторов» позволяет определить сильные и слабые стороны конкурирующих чат-ботов для ресторанов, а также выявить потребности потенциальной аудитории.

Существует большое количество чат-ботов для ресторанов с функцией бронирования мест в Telegram, некоторые из которых пользуются большой популярностью у пользователей. Среди них можно выделить следующие:

«ChatFoodBot» Этот бот предоставляет персоналу ресторана инструменты для управления заказами и доставкой. Он позволяет принимать заказы от клиентов, отслеживать их статус, управлять меню и ингредиентами, а также обеспечивает связь с курьерами для доставки заказов.

«RestroBot» Этот бот предоставляет функционал для управления бронированием столов и обслуживанием посетителей в ресторане. Персонал может принимать запросы на бронирование, проверять наличие свободных столов, устанавливать предпочтения и особые требования клиентов, а также предоставлять информацию о меню и акциях.

«WaiterioBot» Этот бот предназначен для организации работы официантов в ресторане. Он позволяет персоналу принимать заказы, отправлять их на кухню или бар, отслеживать состояние заказов, управлять временными задержками и обеспечивать эффективное взаимодействие между официантами и кухней.

«FoodBot» Этот бот предоставляет персоналу ресторана возможность управлять инвентарем и снабжением. Он позволяет отслеживать запасы продуктов, оформлять заказы у поставщиков, контролировать срок годности и расходы на ингредиенты, а также вести аналитику по потреблению и заказам.

Каждый из этих ботов имеет свои особенности и функциональность, которые могут быть адаптированы под конкретные потребности и процессы работы в ресторане. В результате проведенного анализа рынка существующих решений становится ясно, что чат-боты уже активно применяются в ресторанной сфере.

## Анализ СУБД

SQLite9 - это встроенная реляционная база данных, которая часто используется для хранения небольших объемов данных на клиентских устройствах и встроенных системах. Она не обладает такой же производительностью, как MySQL11 или PostgreSQL12, и не обладает возможностями для обработки нескольких потоков.

Тем не менее, у SQLite есть ряд преимуществ, которые могут быть полезны при разработке чат-бота:

* Простота использования и установки. SQLite легко устанавливается и настраивается, что удобно при разработке на локальном компьютере.
* Низкие требования к ресурсам. SQLite работает быстро на устройствах с ограниченными ресурсами и не требует большого количества системных ресурсов или мощного оборудования.
* Быстрый доступ к небольшим объемам данных. SQLite хорошо работает с статическими и малыми объемами данных (до нескольких гигабайт), что может быть достаточным для разработки чат-бота ресторана.

В целом, выбор между SQLite и другими СУБД будет зависеть от конкретных требований проекта. В данном проекте не будет храниться большое количество данных и ввиду выше перечисленных особенностей, SQLite станет хорошим выбором.

## Обоснование и выбор методики, технологии и инструментальных средств разработки

Для разработки чат-бота в Telegram, доступны различные языки программирования, среды разработки и платформы. Ниже представлены наиболее подходящие варианты.

Язык программирования:

Python4 – это широко используемый язык программирования, который находит применение во многих областях, включая разработку чат-ботов. Python обладает простым, понятным и лаконичным синтаксисом, что делает его особенно привлекательным для новых разработчиков.

Среда разработки:

PyCharm10 – это мощная среда разработки, которая поддерживает работу с Python. Она обладает богатым набором функций для удобного написания кода, включая подсветку синтаксиса, автодополнение и интеграцию с системой контроля версий.

Платформа разработки:

Для разработки чат-бота ресторана с бронированием мест в Telegram рекомендуется выбрать платформу, которая поддерживает работу с Python. Одним из популярных вариантов является Aiogram, который предлагает обширную документацию и активное сообщество разработчиков. Aiogram также обладает встроенной системой хранения данных для каждого пользователя бота и асинхронной архитектурой, что позволяет ему эффективно обрабатывать запросы от нескольких пользователей одновременно.

Исходя из вышеизложенного, можно сделать вывод, что Python, PyCharm и Aiogram5 являются отличным выбором для разработки чат-бота ресторана с функцией бронирования мест в Telegram.

## Специальная часть

## Постановка задачи

Необходимо разработать чат-бота для Telegram, обладающего понятным и удобным интерфейсом и способного автоматизировать процесс добавления клиентов и их обслуживания. Чат-бот предоставит официантам возможность удобно и быстро добавить нового клиента, выбрать зал и стол, выбрать блюда, которые заказал клиент, а так же, пометить стол как обслуженный. Помимо этого, бот должен предоставить рестораторам возможность просматривать доход ресторана за месяц и за все время, просматривать прибыль от каждого сотрудника, список необслуженных заказов, просмотр добавленных в стоп-лист блюд, просматривать рейтинг из популярных блюд.

Задачи проекта:

* Реализация добавления нового клиента;
* Реализация выбора зала и столика;
* Реализация добавления в заказ списка блюд;
* Реализация подтверждения выполнения заказа;
* Реализация просмотра выручки за все время и за месяц;
* Реализация просмотра прибыли от каждого официанта;
* Реализация просмотра списка заказов;
* Реализация просмотра списка блюд;
* Реализация добавления блюда в стоп лист;
* Реализация просмотра списка популярных блюд;
* Реализация просмотра информации о ресторане.

## Основания для разработки

Основанием для разработки является задание руководителя дипломного проектирования Старикова А. Ю., представителя «Информационно-аналитический центр» (ИАЦ).

## Наименование программы

Полное наименование программной разработки: «Разработка умного помощника для рестораторов».

## Назначение разработки

Назначение разработки умного помощника для рестораторов заключается в создании удобного и интуитивно понятного инструмента для работников, позволяющего производить оформление клиентов, формирование заказа, его обслуживания, просмотра информации о выручке и возможность добавления блюда в стоп-лист через мессенджер Telegram. Благодаря его использованию, официанты смогут быстро и удобно выполнять свою работу, а рестораторы смогут контролировать весь процесс.

## Требования к функциональным характеристикам

Необходимо реализовать следующие функции:

* Реализация добавления нового клиента;
* Реализация выбора зала и столика;
* Реализация добавления в заказ списка блюд;
* Реализация подтверждения выполнения заказа;
* Реализация просмотра выручки за все время и за месяц;
* Реализация просмотра прибыли от каждого официанта;
* Реализация просмотра списка заказов;
* Реализация просмотра списка блюд;
* Реализация добавления блюда в стоп лист;
* Реализация просмотра списка популярных блюд;
* Реализация просмотра информации о ресторане.

## Требования к надежности

Надежность приложения должна заключаться к стабильной работе. Бот не должен вылетать и зависать во время работы. Данные без проблем должны сохраняться в базу данных, интерфейс должен отображаться корректно.

## Требования к условиям эксплуатации

Для работы с приложением необходимо иметь базовые навыки работы с телефоном или ПК, особые знания от пользователя не требуются.

## Описание схем

## Бизнес процесс

Чат-бот предоставляет работникам функционал, соответствующий их роли. Бизнес процесс чат бота для роли «Официант» представлен на рисунке 1, а для роли «Ресторатор» на рисунке 2.

Рисунок 1 – бизнес-процесс чат-бота для пользователя с ролью «Официант»

Рисунок 2 – бизнес-процесс чат-бота для пользователя с ролью «Ресторатор»

## Инфологическая модель

На рисунке 3 изображена инфологическая модель базы данных, с которой осуществляется работа бота.



Рисунок 3 – Инфологическая модель базы данных

1. Словарь данных для таблицы «Залы»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип** | **Описание** |
| ID | Int | Первичный ключ, уникальный идентификатор таблицы «Залы» |
| Зал | Text | Зал |

1. Словарь данных для таблицы «Заказы»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип** | **Описание** |
| ID | Int | Первичный ключ, уникальный идентификатор таблицы «Заказы» |
| ID Стола | Int | Внешний ключ, ID Стола |
| ID Клиента | Int | Внешний ключ, ID Клиента |
| ID Официанта | Int | Внешниц ключ, ID Официанта |
| Дата | Text | Дата и время сделки |
| Итого | Int | Итоговая стоимость заказа |
| Обслужено | Int | Статус заказа |

1. Словарь данных для таблицы «Клиенты»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип** | **Описание** |
| ID | Int | Первичный ключ, уникальный идентификатор таблицы «Клиенты» |
| ФИО | Text | ФИО Клиента |
| Номер телефона | Text | Номер телефона клиента |

1. Словарь данных для таблицы «Выбранные блюда»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип** | **Описание** |
| ID | Int | Первичный ключ, уникальный идентификатор таблицы «Выбранные Блюда» |
| ID Заказа | Int | Внешний ключ, ID Заказа |
| ID Блюда | Int | Внешний ключ, ID Блюда |

1. Словарь данных для таблицы «Блюда»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип** | **Описание** |
| ID | Int | Первичный ключ, уникальный идентификатор таблицы «Блюда» |
| Блюдо | Text | Название блюда |
| Стоимость | Int | Стоимость блюда |
| Активно | Int | Статус активности блюда |

1. Словарь данных для таблицы «Столы»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип** | **Описание** |
| ID | Int | Первичный ключ, уникальный идентификатор таблицы «Столы» |
| ID Зала | Int | Внешний ключ, ID Зала |
| Номер стола | Text | Номер стола |
| Количество мест | Int | Количество мест стола |

1. Словарь данных для таблицы «Должности»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип** | **Описание** |
| ID | Int | Первичный ключ, уникальный идентификатор таблицы «Должности» |
| Должность | Text | Буква класса |

1. Словарь данных для таблицы «Работники»

|  |  |  |
| --- | --- | --- |
| **Название** | **Тип** | **Описание** |
| ID | Int | Первичный ключ, уникальный идентификатор таблицы «Работники» |
| ID Должности | Int | Внешний ключ, ID Должности работника |
| TG ID | Text | Telegram ID работника |
| ФИО | Text | ФИО работника |

## Написание чат-бота на Python

Для написания чат-бота на Python нужно установить и подключить необходимые библиотеки (Рискнок 4) и указать токен созданного бота (Рисунок 5).

A screenshot of a computer program

Description automatically generated with medium confidence

Рисунок 4 – подключенные библиотеки

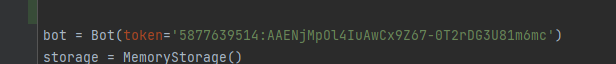


Рисунок 5 – указание токена бота

## Функция для обработки команды

Для обработки команды /start обращаемся к боту, указываем декоратор bot.message\_handler() и указываем те команды, которые будет отслеживать, после чего пишем саму функцию def (Рисунок 6).

A picture containing text, screenshot, font

Description automatically generated

Рисунок 6 – функция для обработки команды /start

## Создание кнопок на клавиатуре

Создание нового файла markups с функциями get\_officiant\_buttons (Рисунок 7) и get\_restorator\_buttons (Рисунок 8).

A picture containing text, screenshot, font

Description automatically generated

Рисунок 7 – функция для получения кнопок официанта

A screen shot of a computer

Description automatically generated with medium confidence

Рисунок 8 – функция для получения кнопок ресторатора

Отображение кнопок, созданных при помощи вышеописанных функций нем, которые будут возвращать клавиатуры для каждых ролей (Рисунок 9, Рисунок 10).

A picture containing text, screenshot, font, line

Description automatically generated

Рисунок 9 – кнопки официанта

A picture containing text, screenshot, font, green

Description automatically generated  
Рисунок 10 – кнопки ресторатора

## Руководство по управлению чат-ботом для официанта

Для начала работы, необходимо перейти по ссылке <https://t.me/RestDiplomBot> или по названию чат-бота, после чего, нажать на кнопку «Начать» или отправить команду /start (Рисунок 11).

A screenshot of a chat

Description automatically generated with medium confidence

Рисунок 11 – кнопки официанта

Для добавления нового клиента, необходимо нажать на кнопку «Новый заказ”, после чего написать контактные данные клиента (Рисунок 12).

A screenshot of a chat

Description automatically generated with medium confidence

Рисунок 12 – ввод контактной информации о клиенте

После ввода контактных данных, официант может выбрать из списка зал (Рисунок 13) и стол (Рисунок 14).

A picture containing text, screenshot, font, green

Description automatically generated

Рисунок 13 – список залов

A screenshot of a phone

Description automatically generated with low confidence

Рисунок 14 – список столов

После выбора стола, официанту станет доступен список блюд, не находящихся в стоп-листе (Рисунок 15). При нажатии на кнопку, соответствующее блюдо будет добавлено к заказу (Рискнок 16).

A screenshot of a computer

Description automatically generated with low confidence

Рисунок 15 – список блюд

A screenshot of a chat

Description automatically generated with medium confidence

Рисунок 16 – выбор блюд

Для добавления заказа в базу данных нужно нажать кнопку «Оформить», после чего, будут выведены сообщения о добавлении и начальные кнопки официанта (Рисунок 17), и заказ появится в таблице (Рисунок 18).

A screenshot of a phone

Description automatically generated with low confidence

Рисунок 17 – сообщение о добавлении и вывод стартовых кнопок

A screenshot of a computer

Description automatically generated with medium confidence

Рисунок 18 – добавление записи в таблицу

Добавленный заказ имеет поле «Обслужено», равное нулю, что означает, что заказ не обслужен. Когда официант принесет на стол весь заказ, он должен нажать на кнопку «Подтвердить заказ», и выбрать нужный из списка (Рисунок 19), после чего, значение поля изменится (Рисунок 20) и заказ будет учитываться в расчётах дохода.

A screenshot of a phone

Description automatically generated with low confidence

Рисунок 19 – выбор стола для обслуживания

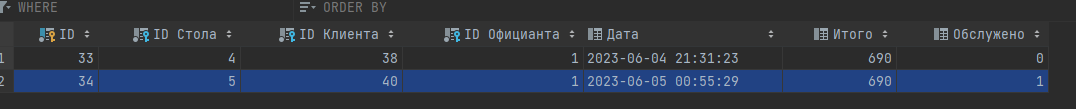


Рисунок 20 – обновление поля «Обслужено»

Далее добавляем эти переменные командой keyboard.add. В засисимости от того как будут добавлены переменные, так и будут отображаться структура кнопок. Переменные вводятся через запятую (Рисунок 15), чтобы они были сгруппированы (Рисунок 16).

## Руководство по управлению чат-ботом для ресторатора

Для начала работы, все так же необходимо перейти по ссылке <https://t.me/RestDiplomBot> или по названию чат-бота, после чего, нажать на кнопку «Начать» или отправить команду /start (Рисунок 21).

A screenshot of a phone

Description automatically generated with medium confidence

Рисунок 21 – вывод кнопок ресторатора

При нажатии кнопки «Вводная информация» будет выведена вводная информация о ресторане (Рисунок 22).

A screenshot of a computer

Description automatically generated with medium confidence

Рисунок 22 – вводная информация о ресторане

При нажатии кнопки «Выручка ресторана», будет выведена выручка ресторана за последний месяц и за все время (Рисунок 23). Учитываются только обслуженные заказы.

A screenshot of a chat

Description automatically generated with low confidence

Рисунок 23 – выручка ресторана

При нажатии кнопки «Выручка по официантам», будет выведена выручка каждого официанта за последний месяц (Рисунок 24). Учитываются тольо обслуженные заказы.

A screenshot of a phone

Description automatically generated with low confidence

Рисунок 24 – выручка по официантам

При нажатии кнопки «Столы с открытыми заказами», будут выведены все еще не обслуженные заказы (Рисунок 25).

A picture containing text, screenshot, font

Description automatically generated

Рисунок 25 – столы с невыполненными заказами

При нажатии кнопки «Стоп лист» будет выведен список блюд с эмодзи, означающий статус блюда (Рисунок 26). Крестик – в стоп листе, галочка – доступно для заказа.

A screenshot of a phone

Description automatically generated with low confidence

Рисунок 26 – список блюд

При нажатии кнопки «Рейтинг блюд», будут выведены блюда и количество заказов за месяц (Рисунок 27)

A screenshot of a computer

Description automatically generated with low confidence

Рисунок 27 – список популярных блюд с количеством заказов

Бот работает только с пользователями, ID которых находится в базе данных (Рисунок 28). Если боту попробует написать человек, ID которого в базе данных нет, то бот выведет соответствующее сообщение (Рисунок 29).

A screenshot of a computer

Description automatically generated with medium confidence

Рисунок 28 – таблица работников ресторана

A screenshot of a phone

Description automatically generated with low confidence

Рисунок 29 – сообщение пользователю, не находящемуся в базе данных

## Экономическая часть

## Область применения программного продукта и его преимущества перед аналогичным программным продуктом

Область применения информационного помощника для рестораторов включает использование его для автоматизации процессов работы ресторана и улучшения качества обслуживания клиентов. Этот помощник может быть использован для следующих задач:

* Автоматизация обработки заказов. Информационный помощник может обрабатывать запросы клиентов и автоматически формировать, и передавать заказ на кухню или бар.
* Управление клиентским сервисом. Информационный помощник может использоваться для обработки и ответа на запросы клиентов в реальном времени.
* Управление персоналом. Информационный помощник может использоваться для автоматического управления персоналом в ресторане, например, для контроля рабочего графика, расчета заработной платы и т.д.
* Аналитика бизнеса. Информационный помощник может быть использован для сбора и анализа данных о производительности ресторана, доходах, расходах и других показателях.

Преимущества информационного помощника перед аналогичными программными продуктами заключаются в следующем:

* Более точная обработка данных. Использование технологий машинного обучения и искусственного интеллекта позволяет информационному помощнику лучше распознавать запросы и более точно обрабатывать данные.
* Улучшенное качество обслуживания клиентов. Информационный помощник может быстро и точно реагировать на запросы клиентов, что позволяет улучшить уровень обслуживания в ресторане.
* Автоматическое управление. Информационный помощник может проводить автоматическое управление различными процессами в ресторане, что позволяет избежать ошибок человеческого фактора и повысить эффективность работы.
* Аналитика и отчётность. Информационный помощник может собирать, обрабатывать и анализировать информацию, что позволяет владельцам ресторанов принимать правильные решения на основе данных.
* Экономия времени и денег. Информационный помощник может автоматизировать многие процессы, что позволяет сэкономить время и снизить затраты на персонал и процессы в ресторане.

В целом, создание информационного помощника для рестораторов является эффективным способом для улучшения бизнес-процессов и повышения качества обслуживания клиентов в ресторане.

## Трудоемкость разработки программного продукта, квалификация исполнителя и его оклад

Трудоемкость разработки можно определить в таблице 9. Строка «Всего» отображает общую трудоемкость разработки.

1. Трудоемкость разработки программного продукта

|  |  |  |
| --- | --- | --- |
| **Наименование этапа** | **Условное обозначение** | **Трудоемкость выполнения этапа, час** |
| Сбор информации, описание задания | То | 40 |
| Разработка архитектуры чат-бота | Тра | 16 |

|  |  |  |
| --- | --- | --- |
| **Наименование этапа** | **Условное обозначение** | **Трудоемкость выполнения этапа, час** |
| Разработка интерфейса чат-бота | Три | 65 |
| Разработка функционала чат-бота | Тфп | 120 |
| Отладка чат-бота | Топ | 48 |
| Оценка качества чат-бота | Тоц | 20 |
| Оформление документации | Тд | 43 |
| Всего | Тобщ | 352 |

Оклад и тарифная ставка разработчика программного продукта представлены в таблице 10.

1. Разработчики программного продукта

|  |  |  |  |
| --- | --- | --- | --- |
| **Исполнители** | **Оклад, руб** | **Часовая тарифная ставка, руб./час** | **Количество сотрудников** |
| Разработчик-программист | 50 000 | 312,5 | 1 |

Часовая тарифная ставка ЧТС, руб./час, определяется исходя из месячного оклада, количества рабочих дней в месяце и продолжительности рабочего дня по формуле:

, (1)

где Ом – оклад исполнителя в месяц, руб./мес;

Д – количество рабочих дней в месяце (для расчета Д = 20 раб. дней);

Тс – продолжительность рабочего дня (для расчета Тс = 8час.).

В таблице 11 показана стоимость технических средств для разработки системы.

1. Стоимость технических средств разработки ИСУ

|  |  |  |  |
| --- | --- | --- | --- |
| **Наименование компонента** | **Цена, руб.** | **Количество, шт.** | **Стоимость, руб.** |
| Ноутбук ASUS Vivobook Go 15 E510MA-BQ509W, 15.6", синий | 30290 | 1 | 30290 |
| Принтер лазерный Pantum P2516/P2518, ч/б, A4, черный | 8215 | 1 | 8215 |
| Итого: | | | 38505 |

В таблице 12 представлены затраты на расходные материалы.

1. Планируемые затраты на расходные материалы

|  |  |  |  |
| --- | --- | --- | --- |
| **Затраты** | **Стоимость** | **Количество** | **Сумма, руб.** |
| Интернет | 500 руб./мес. | 2 месяца | 1000 |
| Электричество | 4,98 руб./КВт\*ч | 1170 КВт\*ч | 5826,6 |
| Бумага | 294,9 руб. | 2 упаковки | 589,8 |
| Ручка | 45 руб. | 1 штука | 45 |
| Заправка МФУ | 2700 руб. | 1 раз | 2700 |
| Итого: | | | 10161,4 |

## Расчет затрат на разработку

Исходные данные, связанные с разработкой программного продукта приведены в таблице 13.

1. Исходные данные

|  |  |  |
| --- | --- | --- |
| **Наименование** | **Обозначение** | **Значение** |
| Оклад разработчика | Ор | 50 000 руб. |
| Время разработки | Трп | 2 месяца |
| Машинное время разработки | Тмч | 1,5 месяца |
| Коэффициент дополнительной заработной платы | Кд | 0,12 |

|  |  |  |
| --- | --- | --- |
| **Наименование** | **Обозначение** | **Значение** |
| Коэффициент страховых взносов | Кст | 0,3 |
| Количество единиц техники | Q | 1 шт |
| Себестоимость содержания техники | См/ч | 15 руб./час |
| Коэффициент готовности техники | Кгт | 0,95 |
| Число рабочих дней в месяце | ЧРД | 20 дней |
| Продолжительность смены | Тст | 8 часов |
| Коэффициент сменности | Ксм | 1 |
| Коэффициент транспортных расходов | Кт | 0,15 |
| Коэффициент накладных расходов | Кнр | 0, 54 |

Произведя вычисления было выявлено, что полные затраты на разработку составляют 280221,86 рублей. Промежуточные результаты вычислений представлены в таблице 14.

1. Результаты вычислений

|  |  |  |
| --- | --- | --- |
| **Наименование** | **Обозначение** | **Значение** |
| Оплата труда | Зот | 112000 руб.; |
| Страховые взносы | Зст | 33600 руб.; |
| **Наименование** | **Обозначение** | **Значение** |
| Содержание ЭВМ | Зэвм | 3420 руб.; |
| Затраты на специальные программы | Зсп | 58590 руб.; |
| Расходные материалы | Зрм | 12131,86 руб.; |
| Накладные расходы | Знр | 60480 руб.; |
| Итого затрат на разработку | Зрп | 280221,86 руб. |

Расчет полных затрат на разработку проектного решения (программного продукта) осуществляется по формуле:

Зрп = Зот + Зст + Зэвм + Зрм + Знр + Зсп (2)

где Зот – затраты на оплату труда разработчика (разработчиков), руб.;

Зст – страховые взносы по оплате труда во внебюджетные фонды, руб.;

Зэвм – затраты, связанные с содержанием вычислительной техники, руб.;

Зрм – затраты на расходные материалы, необходимые при разработке программного продукта, руб.;

Знр – затраты по накладным расходам, приходящиеся на разработку программного продукта, руб.

Зрп = 112000 + 36960 + 3420 + 58590 +12131,86 + 66528 = 300 829,86

## Затраты на оплату труда разработчику (Зот), руб.

Размер фонда оплаты труда разработчиков (Зот) рассчитывается по формуле:

Зот = Ор \* Трп \* (1 + Кд), (3)

где Ор – месячный оклад разработчика проектного решения, руб./мес.;

Трп – время разработки проектного решения разработчиком, мес.(час) включает в себя машинное время работы над проектом (Тмрп);

Кд – коэффициент дополнительной заработной платы разработчика;

Если время разработки программного продукта менее месяца, т.е. определено в часах, то в формулу подставляется часовая тарифная ставка, определяется по формуле 1.

Зот = 50000 \* 2 \* (1 + 0,12) = 112000

## Затраты по страховым взносам (Зст), руб.

Сумма страховых взносов определяется по формуле:

Зст = Кст \* Зот, (4)

где Кст – коэффициент страховых взносов для расчета отчислений во внебюджетные фонды.

Зст = 0,3 \* 112000 = 33600

## Затраты по содержанию Эвм (Зэвм), руб.

Затраты, связанные с эксплуатацией и содержанием ЭВМ, определяются по формуле:

Зэвм = Тмрп \* Кгт \* Q \* См/ч, (5)

где Тмрп – машинное время на разработку проектного решения, час;

Кгт – коэффициент готовности техники;

Q – количество условных единиц используемой техники;

См/ч – стоимость машино-часа, эксплуатации оборудования, руб. в час.

Зэвм = 240 \* 0,95 \* 1 \* 15 = 3420

Так как машинное время может измеряться в месяцах, а себестоимость машино-часа за один час, то машинное время необходимо перевести в часы.

Перевод рабочего времени в часы осуществляется по формуле:

Тмрп = Тмч \* Чрд \* Тсм \* Ксм, (6)

где Тмч рабочее время в месяцах;

Чрд – число рабочих дней в месяце;

Тсм – продолжительность рабочей смены;

Ксм – количество рабочих смен.

Тмрп = 1,5 \* 20 \* 8 \* 1 = 240

## Затраты на расходные материалы

Затраты на расходные материалы, необходимые для разработки проектного решения определяются по формуле:

, (7)

где Qi – количество i – го вида материала, шт.;

Рi – цена i – го вида материала, руб.;

n – количество видов материалов;

Кт – коэффициент транспортных расходов.

Зрм = 12131,86

## Затраты по накладным расходам (Знр)

Затраты по накладным расходам определяются по формуле:

Знр = Кнр \* Зот, (8)

где Кнр – коэффициент накладных расходов (или процент) принимается для расчета по данным предприятия;

Зот – затраты по оплате труда, руб.

Знр = 0,54 \* 112000 = 60480

## Расчет цены и прибыли

Исходные данные, связанные с разработкой программного продукта приведены в таблице 15.

1. Исходные данные

|  |  |  |
| --- | --- | --- |
| **Наименование** | **Обозначение** | **Значение** |
| Коэффициент рентабельности | Кр | 0,25 |
| Коэффициент налога на добавочную стоимость | Кндс | 0,2 |
| Ставка налога | Кнп | 0,2 |
| Стоимость затрат на разработку и внедрение | Квпр | 280221,86 руб. |

Произведя вычисления было выявлено, что чистая прибыль будет составлять 60 165,972 рублей. Промежуточные результаты вычислений представлены в таблице 16.

1. Результаты вычислений

|  |  |  |
| --- | --- | --- |
| **Наименование** | **Обозначение** | **Значение** |
| Плановая прибыль | Ппл | 75 207,465 |
| Налог на добавочную стоимость | НДС | 75 207,465 |
| Цена программного продукта | Цпп | 451 244,79 |
| Налог на прибыль | НП | 15 041,493 |
| Чистая прибыль | ЧП | 60 165,972 |
| Поступление в бюджет | ПБ | 90 248,958 |

Цена программного продукта, который разработан одной организацией по заказу другой и не предназначен для тиражирования, определяется по формуле:

Цпп = Квпр +Ппл + НДС, руб., (9)

Цпп = 280221,86 + 70055,456 + 70055,456 = 420332,772

где Ппл – планируемая прибыль рассчитывается по формуле:

Ппл = Квпр \* Кр, руб исходя из нормативной рентабельности (коэффициента) Кр = 0,25.

Ппл = 280221,86 \* 0,25 = 70055,456

НДС – налог на добавленную стоимость определяется исходя из Кндс = 0,2 (ставка налога 20%) по формуле:

НДС = (Квпр + Ппл) \* Кндс, руб. (10)

НДС = (280221,86 + 70055,456) \* 0,2 = 70055,465

Каждое предприятие с полученной прибыли перечисляет государству налог на прибыль. На сегодня ставка налога 20% (Кнп = 0,2) от полученной прибыли, и определяется по формуле:

НП = Ппл \* Кнп, руб. (11)

НП = 70055,465 \* 0,2 = 14011,093

Чистая прибыль, остающаяся в распоряжении предприятия:

ЧП = Ппл – НП, руб. (12)

ЧП = 70055,465 – 14011,093= 56044,372

Поступление в бюджет складываются из налога на прибыль и НДС.

ПБ = НП + НДС, руб. (13)

ПБ = 14011,093 + 70055,465 = 84066,558

Исходя из расчетов экономической части затраты на разработку составили 280221,86 рублей, цена программного продукта составляет 420332,772 рублей, а чистая прибыль продукта составляет 56044,372 рублей.

## Техника безопасности и охрана труда

## Анализ условий труда программиста

Рабочее место программиста – это часть помещения, где он проводит большую часть рабочего времени. Совокупность показателей технических и программных средств определяет качество и производительность работы, наряду с габаритными показателями мебели и её удобством.

Помещение для работы представляет собой кабинет длиной 5 м, шириной 4 м и высотой потолков 2,5 м.

Согласно санитарным правилам монитор на столе нужно располагать на расстоянии 60 – 70 см от глаз пользователя, но не ближе 50 см с учетом размеров алфавитно-цифровых знаков и символов. При использовании жидкокристаллических дисплеев на каждый компьютер должно приходиться не менее 4,5 м2 площади. На дисплей ПК не должен попадать прямой солнечный свет во избежание бликов и повышенной нагрузки на зрение.

Для снижения нагрузки на технические средства, влекущей за собой перегрев компьютера и ноутбука, используются программные средства, позволяющие снизить нагрузку на технические средства за счёт отказа в использование ненужных служб.

При идентификации вредных производственных факторов было установлено, что опасными могут являться:

* электромагнитные поля и излучения от технических средств;
* статическое электричество;
* статические перегрузки, вызываемые неподвижной работой за компьютером;
* перенапряжение органов зрения.

В целях выявления вредных и (или) опасных факторов производственной среды и трудового процесса, оценке уровня их воздействия на работника с учетом отклонения их фактических значений от установленных уполномоченным Правительством Российской Федерации федеральным органом исполнительной власти гигиенических нормативов условий труда и применения средств индивидуальной и коллективной защиты работников проводится специальная оценка условий на рабочих местах. Оценке подлежат все имеющиеся в организации рабочие места.

Работа, по специальной оценке, условий труда заключается в исследовании следующих факторов на рабочих местах организации:

* химические факторы;
* физические факторы;
* биологические факторы;
* тяжесть и напряженность трудового процесса.

Для уменьшения нагрузки на органы зрения, а также снятия мышечного напряжения во время рабочего дня проводится проветривание в течении 10-15 минут каждые 3 часа.

## Расчёт искусственного освещения в помещении

Через зрительный анализатор человек получает около 80% из общего объема информации. Качество поступающей, в основном, зависит от освещения в помещении. Если оно неудовлетворительное, это приводит к утомлению организма в целом. Кроме того, нерациональное освещение может явиться причиной травматизма: плохо освещенные опасные зоны, слепящие источники света и блики от них, резкие тени ухудшают видимость настолько, что вызывают полную потерю ориентировки работающих, снижают производительность труда и увеличивают брак продукции. Поэтому необходимо достаточное искусственное освещение для рабочего места.

Условия работы для расчета показателя освещенности рабочего места программиста являются:

* помещение размером 20 квадратных метров;
* высота помещения 2,5 метра, длина 5 метров, ширина 4 метра;
* лампы люминесцентные (дневного света) в количестве 2 штук, новые;
* крупногабаритная мебель отсутствует.

Основным критерием, по которому определяется необходимое количество осветительных приборов, является нормируемый уровень освещенности.

Базовым методом проектирования осветительной установки является метод коэффициентов использования, позволяющий вручную проводить все вычислительные процедуры при решении относительно простых светотехнических задач.

По этому методу необходимое количество светильников в осветительной установке определяется с помощью следующей формулы:



где N – количество светильников, шт;

Е – нормируемая освещенность, лк, определяется по СП 52.13330.2016 «Естественное и искусственное освещение»;

S – площадь помещения, м2;

Кз – коэффициент запаса = 1,4;

U – коэффициент использования = 0,54;

Фл – световой поток одной лампы в светильнике, 2600 лм;

n – количество ламп в светильнике = 2.

Коэффициент использования характеризует эффективность использования светового прибора в помещении. Для его определения необходимо знать индекс помещения φ и коэффициент отражения стен, пола и потолка.

Индекс помещения:



где  – индекс помещения;

S – площадь помещения, м2;

h1 – высота подвеса светильников над рабочей поверхностью, м;

h2 – высота рабочей поверхности, м (принять 0,8 м);

а – длина помещения, м;

b – ширина помещения, м

Полученное количество светильников совпадает с числом светильников, установленных в помещении, что соответствует нормативным показателям искусственного освещения согласно СП 52.13330.2016 «Естественное и искусственное освещение».

## Электробезопасность на предприятии

При поступлении на работу сотрудник должен пройти инструктаж по технике безопасности и электробезопасности. Работника знакомят с основными правилами по технике безопасности, предлагают внимательно прочитать действующие на предприятии инструкции, поясняя при этом отдельные правила и требования.

Инструктаж по технике безопасности при выполнении конкретной работы проводит руководитель соответствующего производственного участка, показывая инструктируемому правильные безопасные приемы работы.

Повторный инструктаж проводится для рабочих независимо от их квалификации, стажа и опыта работы, не реже 1 раза в 6 месяцев по программе инструктажа на рабочем месте.

При нарушении работающими правил и инструкций по технике безопасности, технологической и производственной дисциплины, а также в случаях изменения технологического процесса или вида работы проводят дополнительные инструктажи.

Все инструктажи оформляются записями в специальном журнале с указанием номеров или шифров инструкций. Журнал о проведении инструктажа хранится у руководителя подразделения.

Работник на рабочем месте должен соблюдать общие меры электробезопасности. При этом запрещается:

* прикасаться к задней панели системного блока при включенном питании;
* допускать попадание влаги на поверхность системного блока (процессора), монитора, рабочую поверхность клавиатуры, дисководов, принтеров и др. устройств;
* производить самостоятельно вскрытие и ремонт оборудования;
* пользоваться неисправными электроприборами и электропроводкой;
* ремонтировать электроприборы самостоятельно;
* подвешивать электропровода на гвоздях, металлических и деревянных предметах, перекручивать провод, закладывать провод и шнуры на водопроводные трубы и батареи отопления, вешать что-либо на провода, вытягивать за шнур вилку из розетки;
* прикасаться одновременно к персональному компьютеру и к устройствам, имеющим соединение с землей (радиаторы отопления, водопроводные краны, трубы и т.п.);
* пользоваться самодельными электронагревательными приборами и электроприборами с открытой спиралью;
* наступать на переносимые электрические провода, лежащие на полу.

При работе с электроприборами и оргтехникой (персональные компьютеры, принтеры, сканеры, копировальные аппараты, факсы, бытовые электроприборы, приборы освещения):

* Автоматические выключатели и электрические предохранители должны быть всегда исправны;
* Изоляция электропроводки, электроприборов, выключателей, штепсельных розеток, ламповых патронов и светильников, а также шнуров, с помощью которых включаются в электросеть электроприборы, должны быть в исправном состоянии;
* Электроприборы необходимо хранить в сухом месте, избегать резких колебаний температуры, вибрации, сотрясений;
* Для подогрева воды пользоваться сертифицированными электроприборами с закрытой спиралью и устройством автоматического отключения, с применением несгораемых подставок.

## Пожарная безопасность на предприятии

Инструктаж по пожарной безопасности проводится по программе, разработанной ответственным по охране труда организации, с учетом требований стандартов, правил, норм и инструкций о мерах пожарной безопасности. Продолжительность инструктажа устанавливается в соответствии с утвержденной программой. Инструктаж по пожарной безопасности, как правило, проводится совместно с инструктажем по технике безопасности и в те же сроки.

Инструктаж по пожарной безопасности проходят все работники организации, независимо от их образования, стажа работы по данной профессии или должности, временные работники, командированные, обучающиеся и студенты, прибывшие на практику.

О проведении инструктажа по пожарной безопасности работник, проводивший инструктаж, делает запись в журнале учета проведения инструктажей по пожарной безопасности, с обязательной подписью инструктируемого и инструктирующего. При регистрации внепланового инструктажа по пожарной безопасности указывают причину его проведения.

Все работники организаций должны допускаться к работе только после прохождения противопожарного инструктажа, а при изменении специфики работы проходить дополнительное обучение предупреждению и тушению возможных пожаров в порядке, установленном руководителем.

На проведение первичного противопожарного инструктажа необходимо отводить не менее 1 часа. Инструктируемые должны ознакомиться:

* С действующими на объекте правилами пожарной безопасности и инструкциями;
* С производственными участками, наиболее опасными в пожарном отношении, где запрещается курить, применять открытый огонь;
* С возможными причинами возникновения пожара и мерами его предупреждения;
* С практическими действиями в случае возникновения пожара - вызов пожарной помощи, использование первичных средств пожаротушения, место расположения ближайшего телефона и ознакомление с правилами поведения в случае возникновения пожара, эвакуации людей и материальных ценностей.

При первичном инструктаже инструктирующий обязан рассказать о производственных установках с повышенной пожарной опасностью, мерах предотвращения пожаров и загораний, указать место курения, ознакомить вновь поступившего с имеющимися на объекте средствами пожаротушения, показать ближайший телефон и объяснить правила поведения в случае возникновения пожара.

Проведение противопожарного инструктажа в обязательном порядке должно сопровождаться практическим показом способов использования имеющихся на объекте средств пожаротушения (противогазы, респираторы, огнетушители и т.д.).

Соблюдение рассмотренных в данном разделе мероприятий позволяет снизить утомляемость и травматизм, повысить производительность труда, обеспечить комфортные условия трудовой деятельности.

# ЗАКЛЮЧЕНИЕ

Результатом преддипломной практики является разработанный чат бот для работников ресторана, обладающий понятным и удобным интерфейсом и способного автоматизировать процесс добавления клиентов и их обслуживания. Чат-бот предоставляет официантам возможность удобно и быстро добавить нового клиента, выбрать зал и стол, выбрать блюда, которые заказал клиент, а также, пометить стол как обслуженный. Помимо этого, бот предоставляет рестораторам возможность просматривать доход ресторана за месяц и за все время, просматривать прибыль от каждого сотрудника, список необслуженных заказов, просмотр добавленных в стоп-лист блюд, просматривать рейтинг из популярных блюд. В результате, были разработаны следующие функции:

* Добавления нового клиента;
* Выбор зала и столика;
* Добавления в заказ списка блюд;
* Подтверждение выполнения заказа;
* Просмотр выручки за все время и за месяц;
* Просмотр прибыли от каждого официанта;
* Просмотр списка заказов;
* Просмотр списка блюд;
* Добавления блюда в стоп лист;
* Просмотр списка популярных блюд;
* Просмотр информации о ресторане.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Python. Карманный справочник / Марк Лутц / ДМК Пресс, 2019 — 272 с.
2. Изучаем Python / Марк Лутц / Символ-Плюс, 2013 — 1616 с.
3. Dive Into Python 3 / Марк Пилгрим / Apress, 2009 — 413 с.
4. Техническая документация Python.org [Электронный ресурс]: [https://docs.python.org](https://docs.python.org/)
5. Aiogram: Документация / Кирилл Волков / GitHub, 2022 [Электронный ресурс]: [https://docs.aiogram.dev](https://docs.aiogram.dev/)
6. Aiogram: Telegram Bot Framework for Python / Кирилл Волков / Medium, 2021 [Электронный ресурс]: <https://medium.com/aiogram>
7. SQLite: Карманный справочник / Марк Лутц / ДМК Пресс, 2019 — 176 с.
8. SQLite. Самоучитель / Алексей Кузнецов / Питер, 2015 — 240 с.
9. Техническая документация SQLite.org [Электронный ресурс]: <https://www.sqlite.org/docs.html>
10. Техническая документация PyCharm [Электронный ресурс]: <https://www.jetbrains.com/pycharm/guide/tips/quick-docs/>
11. Техническая документация MySQL [Электронный ресурс]: <https://dev.mysql.com/doc/>
12. Техническая документацяия PostgreSQL [Электронный ресурс]: <https://www.postgresql.org/>
13. Техническая документация TelegramBotFather [Электронный ресурс]: https://core.telegram.org/bots/tutorial
14. ГОСТ 28195 – 89. Оценка качества программных средств. Общие положения. // М.: Издательство стандартов, 1989
15. ГОСТ 19.102-77. ЕСПД. Стадии разработки.
16. ГОСТ 19.105-78 ЕСПД. Общие требования к программным документам.
17. ГОСТ 19.106-78 ЕСПД. Требования к программным документам, выполненным печатным способом.
18. ГОСТ 19.402-78 ЕСПД. Описание программы.
19. ГОСТ 19.404-79 ЕСПД. Пояснительная записка. Требования к содержанию и оформлению.
20. ГОСТ 19.505-79 ЕСПД. Руководство оператора.

# ПРИЛОЖЕНИЕ А

(справочное)

**Исходный код**

main.py:

import markups as nav  
  
from DataBase import DataBase  
from aiogram import Bot, Dispatcher  
from aiogram.contrib.fsm\_storage.memory import MemoryStorage  
from aiogram.dispatcher import FSMContext  
from aiogram.dispatcher.filters.state import State, StatesGroup  
from telebot import types  
  
  
class States(StatesGroup):  
 Await\_Officiant\_Buttons = State()  
 Await\_Restorator\_Buttons = State()  
 Ask\_Phone = State()  
 Ask\_Fio = State()  
 Ask\_Zal = State()  
 Ask\_Table = State()  
 Ask\_Food = State()  
 Ask\_Stop\_List = State()  
 Get\_Service\_Status = State()  
  
  
counter = 1  
  
bot = Bot(token='5877639514:AAENjMpOl4IuAwCx9Z67-0T2rDG3U81m6mc')  
storage = MemoryStorage()  
dp = Dispatcher(bot, storage=storage)  
  
bd\_path = 'E:\SQLLiteDBs\TGRestDiplomBot.db'  
  
data\_base = DataBase(bd\_path)  
  
  
@dp.message\_handler(commands=['start'], state='\*')  
async def start(message: types.Message, state: FSMContext):  
 result = data\_base.get\_worker\_data(message.from\_user.id)  
  
 worker\_result = await send\_welcome\_message(result, message)  
  
 async with state.proxy() as data:  
 data["worker\_id"] = worker\_result[0]  
  
 await show\_buttons(worker\_result[1], message)  
  
  
@dp.message\_handler(commands=['switch'], state='\*')  
async def switch(message: types.Message, state: FSMContext):  
 global counter  
  
 counter += 1  
 if counter > 2:  
 counter = 1  
  
 await message.answer(f"Вы сменили роль на {counter}, напишите команду /start")  
  
 data\_base.switch\_role(918386646, counter)  
  
  
async def show\_buttons(role\_id, message):  
 if role\_id == 1:  
 await States.Await\_Officiant\_Buttons.set()  
 await bot.send\_message(message.from\_user.id, "Выберите действие", reply\_markup=nav.get\_officiant\_buttons())  
 elif role\_id == 2:  
 await States.Await\_Restorator\_Buttons.set()  
 await bot.send\_message(message.from\_user.id, "Выберите действие", reply\_markup=nav.get\_restorator\_buttons())  
 else:  
 await message.answer("Вы не являетесь членом рабочего персонала")  
  
  
async def send\_welcome\_message(result, message: types.Message):  
 if result is not None:  
 role\_id = result[1]  
 worker\_name = result[2]  
 else:  
 await message.answer("Вы не являетесь работником ресторана")  
 return  
  
 role\_text = ""  
 if role\_id == 2:  
 role\_text = "ваша роль - ресторатор"  
 elif role\_id == 1:  
 role\_text = "ваша роль - официант"  
  
 await message.answer(f"Добро пожаловать в систему, {worker\_name}, {role\_text}")  
 return result  
  
  
# обработчик кнопок официанта  
@dp.callback\_query\_handler(text\_contains="officiant", state=[States.Await\_Officiant\_Buttons])  
async def officiant\_buttons(call: types.CallbackQuery):  
 await bot.delete\_message(call.from\_user.id, call.message.message\_id)  
  
 if call.data == "officiant\_new\_client":  
 await bot.send\_message(call.from\_user.id, "Для начала, укажите номер телефона")  
 await States.Ask\_Phone.set()  
  
 elif call.data == "officiant\_stop\_list":  
 await bot.send\_message(call.from\_user.id, "Выберите стол, который был полностью обслужен",  
 reply\_markup=nav.get\_unserved\_orders())  
 await States.Get\_Service\_Status.set()  
  
  
# обработчик кнопок ресторатораВыбранные блюда  
@dp.callback\_query\_handler(text\_contains="restorator\_", state=[States.Await\_Restorator\_Buttons])  
async def restorator\_buttons(call: types.CallbackQuery):  
 button = int(call.data.replace("restorator\_", ""))  
  
 if button == 1:  
 await bot.send\_message(call.from\_user.id,  
 "Морской бриз расположен в самом сердце города на берегу прекрасной реки. "  
 "Гости могут наслаждаться прекрасным видом на реку и атмосферой морского курорта.")  
 elif button == 2:  
 month = data\_base.show\_revenue\_for\_the\_month()  
 total = data\_base.show\_total\_revenue()  
 await bot.send\_message(call.from\_user.id, f"Доход за этот месяц: {month}р \nДоход за все время: {total}р")  
  
 elif button == 3:  
 rows = data\_base.get\_officiants\_revenue()  
 message = "Выручка по работникам:\n\n"  
  
 if rows:  
 for row in rows:  
 worker\_id, fio, revenue = row  
 message += f"Работник {fio}: {revenue} руб.\n"  
 else:  
 message += "Нет данных о выручке по работникам."  
  
 await bot.send\_message(call.from\_user.id, message)  
  
 elif button == 4:  
 rows = data\_base.get\_orders\_list()  
 message = "Невыполненные заказы:\n\n"  
  
 if rows:  
 for row in rows:  
 table, zal, food = row  
 message += f"| {zal} | {table} | Заказ - {food}\n"  
 else:  
 message += "Нет данных о невыполненных заказах"  
  
 await bot.send\_message(call.from\_user.id, message)  
  
 elif button == 5:  
  
 await bot.delete\_message(call.from\_user.id, call.message.message\_id)  
 await bot.send\_message(call.from\_user.id, "Нажмите на кнопку, чтобы добавить или убрать еду из меню",  
 reply\_markup=nav.get\_food\_and\_statuses())  
 await States.Ask\_Stop\_List.set()  
  
 elif button == 6:  
  
 count = 5  
  
 rows = data\_base.get\_most\_popular\_food(count)  
 message = f"Топ {count} самых популярных блюд:\n\n"  
  
 if rows:  
 for row in rows:  
 food, amount = row  
 message += f"| Количество: {amount} | Блюдо: {food}\n"  
  
 await bot.send\_message(call.from\_user.id, message)  
  
  
@dp.callback\_query\_handler(text\_contains="food\_status\_", state=[States.Ask\_Stop\_List])  
async def food\_statuses(call: types.CallbackQuery):  
 if call.data == "food\_status\_back":  
 await States.Await\_Restorator\_Buttons.set()  
 await bot.send\_message(call.from\_user.id, "Выберите действие", reply\_markup=nav.get\_restorator\_buttons())  
  
 else:  
 food\_id = int(call.data.replace("food\_status\_", ""))  
 data\_base.update\_food\_status(food\_id)  
  
 await call.message.edit\_text("Нажмите на кнопку, чтобы добавить или убрать еду из меню",  
 reply\_markup=nav.get\_food\_and\_statuses())  
  
  
@dp.callback\_query\_handler(text\_contains="unserved\_order\_", state=[States.Get\_Service\_Status])  
async def get\_tables\_served(call: types.CallbackQuery):  
 if call.data == "unserved\_order\_back":  
 await bot.delete\_message(call.from\_user.id, call.message.message\_id)  
 await show\_buttons(1, call)  
 else:  
 table\_id = call.data.replace("unserved\_order\_", "")  
 data\_base.serve\_order(table\_id)  
 await bot.send\_message(call.from\_user.id, "Стол обслужен")  
 await bot.delete\_message(call.from\_user.id, call.message.message\_id)  
 await bot.send\_message(call.from\_user.id, "Выберите стол, который должен быть обслужен",  
 reply\_markup=nav.get\_unserved\_orders())  
  
  
@dp.message\_handler(state=[States.Ask\_Phone])  
async def ask\_phone(message: types.Message, state: FSMContext):  
 user\_id = message.from\_user.id  
 phone = message.text  
  
 client\_id = len(data\_base.try\_get\_user(phone))  
  
 if bool(client\_id):  
 await bot.send\_message(message.from\_user.id, "Выберите зал", reply\_markup=nav.get\_zals())  
  
 async with state.proxy() as data:  
 data["client\_id"] = client\_id  
  
 await States.Ask\_Zal.set()  
 else:  
 async with state.proxy() as data:  
 data["phone"] = phone  
  
 await bot.send\_message(user\_id, "Введите ФИО клиента")  
 await States.Ask\_Fio.set()  
  
  
@dp.message\_handler(state=[States.Ask\_Fio])  
async def ask\_fio(message: types.Message, state: FSMContext):  
 user\_id = message.from\_user.id  
 fio = message.text  
  
 async with state.proxy() as data:  
 phone = data.get("phone")  
  
 client\_id = data\_base.add\_user(fio, phone)  
  
 data["client\_id"] = client\_id  
  
 await bot.send\_message(message.from\_user.id, "Выберите зал", reply\_markup=nav.get\_zals())  
 await States.Ask\_Zal.set()  
  
  
# обработчик кнопок зала  
@dp.callback\_query\_handler(text\_contains="zal\_", state=[States.Ask\_Zal])  
async def zal\_buttons(call: types.CallbackQuery, state: FSMContext):  
 await bot.delete\_message(call.from\_user.id, call.message.message\_id)  
  
 zal = call.data.replace("zal\_", "")  
  
 async with state.proxy() as data:  
 data["zal"] = zal  
  
 await bot.send\_message(call.from\_user.id, "Выберите стол", reply\_markup=nav.get\_tables(zal))  
 await States.Ask\_Table.set()  
  
  
# обработчик кнопок стола  
@dp.callback\_query\_handler(text\_contains="table\_", state=[States.Ask\_Table])  
async def table\_buttons(call: types.CallbackQuery, state: FSMContext):  
 await bot.delete\_message(call.from\_user.id, call.message.message\_id)  
 table\_id = call.data.replace("table\_", "")  
  
 async with state.proxy() as data:  
 data["table\_id"] = table\_id  
  
 await bot.send\_message(call.from\_user.id, "Выберите что хотите заказать", reply\_markup=nav.get\_food())  
 await States.Ask\_Food.set()  
  
  
# обработчик кнопок выбора еды  
@dp.callback\_query\_handler(text\_contains="food\_", state=[States.Ask\_Food])  
async def food\_buttons(call: types.CallbackQuery, state: FSMContext):  
 food\_id = call.data.replace("food\_", "")  
  
 if (call.data == "food\_confirm"):  
  
 async with state.proxy() as data:  
 selected\_foods = data.get("selected\_foods", [])  
 table\_id = data.get("table\_id")  
 client\_id = data.get("client\_id")  
 worker\_id = data.get("worker\_id")  
  
 if not selected\_foods:  
 await bot.send\_message(call.from\_user.id, "Вы не выбрали ни одного пункта еды.")  
 return  
  
 total\_price = (data\_base.get\_total\_price(selected\_foods))  
  
 order\_id = data\_base.add\_order(table\_id, client\_id, worker\_id, total\_price)  
  
 # добавление еды в бд  
 for food\_id in selected\_foods:  
 data\_base.add\_food(order\_id, food\_id)  
  
 # Очистка списка выбранных пунктов еды  
 data["selected\_foods"] = []  
  
 await bot.send\_message(call.from\_user.id, "Заказ добавлен")  
  
 await show\_buttons(1, call)  
  
 else:  
 async with state.proxy() as data:  
 if "selected\_foods" not in data:  
 data["selected\_foods"] = []  
  
 data["selected\_foods"].append(food\_id)  
 await bot.send\_message(call.from\_user.id, f"Добавлено в заказ")  
  
  
# Запускаем бота  
if \_\_name\_\_ == '\_\_main\_\_':  
 from aiogram import executor  
  
 executor.start\_polling(dp)

DataBase.py:

import datetime  
import sqlite3  
  
today = datetime.datetime.today()  
  
start\_of\_month = datetime.datetime(today.year, today.month, 1)  
start\_of\_month\_str = start\_of\_month.strftime('%Y-%m-%d')  
  
class DataBase:  
  
 def \_\_init\_\_(self, db\_file):  
 self.connection = sqlite3.connect(db\_file)  
 self.cursor = self.connection.cursor()  
  
 def switch\_role(self, tg\_id, role\_id):  
 with self.connection:  
 result = self.cursor.execute("UPDATE Работники SET [Id Должности] = ? WHERE [TG ID] = ?", (role\_id, tg\_id,))  
 return result  
  
 def try\_get\_user(self, phone):  
 with self.connection:  
 result = self.cursor.execute("select id from Клиенты where [Номер телефона] = ?", (phone,)).fetchall()  
  
 return result  
  
 def get\_worker\_data(self, user\_id):  
 with self.connection:  
 result = self.cursor.execute("SELECT id, [ID Должности], ФИО FROM Работники WHERE [TG ID] = ?",  
 (user\_id,)).fetchone()  
 return result  
  
 def get\_zals(self):  
 with self.connection:  
 result = self.cursor.execute("select id, Зал from Залы").fetchall()  
  
 return result  
  
 def get\_tables(self, zal):  
 with self.connection:  
 result = self.cursor.execute("SELECT DISTINCT Столы.id, Столы.Стол "  
 "FROM Столы "  
 "LEFT JOIN Заказы ON Заказы.[ID Стола] = Столы.ID "  
 "WHERE Заказы.[ID Стола] IS NULL "  
 "AND Столы.[ID Зала] = ?", (zal,)).fetchall()  
  
 return result  
  
 def get\_food(self):  
 with self.connection:  
 result = self.cursor.execute("Select ID, Блюдо from Блюда where Активно == 1")  
  
 return result  
  
 def get\_total\_price(self, food\_ids):  
 total\_price = 0  
 with self.connection:  
 for food\_id in food\_ids:  
 result = self.cursor.execute("SELECT Стоимость FROM Блюда WHERE ID = ?", (food\_id,)).fetchone()  
 if result:  
 total\_price += result[0]  
 return total\_price  
  
 def get\_unserved\_orders(self):  
 with self.connection:  
 result = self.cursor.execute(  
 "SELECT Заказы.ID, Столы.Стол, Залы.Зал "  
 "FROM Заказы "  
 "JOIN Столы ON Заказы.[Id Стола] = Столы.Id "  
 "JOIN Залы ON Столы.[ID Зала] = Залы.ID "  
 "WHERE Заказы.Обслужено = 0")  
  
 return result  
  
 def add\_user(self, fio, phone):  
 with self.connection:  
 self.cursor.execute("INSERT INTO Клиенты (ФИО, [Номер телефона]) VALUES (?, ?)", (fio, phone,))  
 last\_insert\_id = self.cursor.lastrowid  
  
 return last\_insert\_id  
  
 def add\_order(self, table\_id, client\_id, worker\_id, total\_price):  
 with self.connection:  
 current\_datetime = datetime.datetime.now()  
 current\_datetime\_str = current\_datetime.strftime("%Y-%m-%d %H:%M:%S")  
  
 self.cursor.execute(  
 "Insert into Заказы ([ID Стола], [Id Клиента], [Id Официанта], Дата, Итого, Обслужено ) "  
 "values (?, ?, ?, ?, ?, ?) ", (  
 table\_id, client\_id, worker\_id, current\_datetime\_str, total\_price, 0,)).fetchall()  
  
 last\_insert\_id = self.cursor.lastrowid  
  
 return last\_insert\_id  
  
 def add\_food(self, order\_id, food\_id):  
 with self.connection:  
 self.cursor.execute("insert into [Выбранные блюда] ([ID Заказа], [ID Блюда]) values (?, ?) ",  
 (order\_id, food\_id,))  
  
 def serve\_order(self, order\_id):  
 with self.connection:  
 self.cursor.execute("UPDATE Заказы SET Обслужено = 1 WHERE ID = ?", (order\_id,))  
  
 def show\_revenue\_for\_the\_month(self):  
  
  
 with self.connection:  
 result = self.cursor.execute(  
 "SELECT SUM(Итого) FROM Заказы WHERE Дата >= date(?) AND Дата < date(?, '+1 month') and Обслужено = 1",  
 (start\_of\_month\_str, start\_of\_month\_str,)).fetchone()[0]  
  
 return result  
  
 def show\_total\_revenue(self):  
 with self.connection:  
 result = self.cursor.execute("SELECT SUM(Итого) FROM Заказы WHERE Обслужено = 1").fetchone()[0]  
  
 return result  
  
  
 def get\_officiants\_revenue(self):  
 with self.connection:  
 result = self.cursor.execute("SELECT Работники.Id, Работники.ФИО, SUM(Заказы.Итого) AS Выручка "  
 "FROM Заказы "  
 "JOIN Работники ON Заказы.[ID Официанта] = Работники.Id "  
 "Where Обслужено = 1 "  
 "and Дата >= date(?) AND Дата < date(?, '+1 month') "  
 "GROUP BY Работники.Id, Работники.ФИО",  
 (start\_of\_month\_str, start\_of\_month\_str,)).fetchall()  
  
 return result  
  
 def get\_orders\_list(self):  
 with self.connection:  
 result = self.cursor.execute("SELECT DISTINCT Столы.Стол, Залы.Зал, Блюда.Блюдо "  
 "FROM Столы "  
 "JOIN Заказы ON Столы.ID = Заказы.[Id Стола] "  
 "JOIN Залы ON Залы.ID = Столы.[ID Зала] "  
 "JOIN [Выбранные блюда] ON [Выбранные блюда].[ID Заказа] = Заказы.id "  
 "JOIN Блюда ON [Выбранные блюда].[Id Блюда] = Блюда.Id "  
 "WHERE Заказы.Обслужено = 0"  
 ).fetchall()  
  
 return result  
  
 def get\_most\_popular\_food(self, count):  
 with self.connection:  
 result = self.cursor.execute("SELECT Блюда.Блюдо, COUNT([ID Блюда]) AS Количество "  
 "FROM [Выбранные блюда], Блюда "  
 "where [Выбранные блюда].[Id Блюда] = Блюда.id "  
 "GROUP BY [ID Блюда] "  
 "ORDER BY Количество DESC "  
 "LIMIT ?;",(count,)).fetchall()  
  
 return result  
  
 def get\_food\_and\_statuses(self):  
 with self.connection:  
 result = self.cursor.execute("Select ID, Блюдо, Активно from Блюда").fetchall()  
  
 return result  
  
 def update\_food\_status(self, food\_id):  
 with self.connection:  
 result\_status = self.cursor.execute("select Активно from Блюда where id = ?", (food\_id,)).fetchone()[0]  
  
 if result\_status == 1:  
 self.cursor.execute("Update Блюда set Активно = 0 where id = ?", (food\_id,))  
 else:  
 self.cursor.execute("Update Блюда set Активно = 1 where id = ?", (food\_id,))