**Министерство профессионального образования и занятости населения Приморского края**

**краевое государственное автономное профессиональное образовательное учреждение**

**«ПРИМОРСКИЙ ПОЛИТЕХНИЧЕСКИЙ КОЛЛЕДЖ»**

**(КГА ПОУ «ППК»)**

**ОТЧЕТ ПО УЧЕБНОЙ ПРАКТИКЕ**

**ПМ 05 «ЧТО ТУТ НАПИСАТЬ Я НЕ ЗНАЮ»**

**09.02.07 «Информационные системы и программирование»**

|  |  |
| --- | --- |
| **Выполнил** | **Руководитель практики** |
| студент группы 1135 | Пушкин А.А. |
| Рябушенко М.С. |  |
|  |  |

Оценка, полученная на защите практики \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Владивосток 2022**

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc102609933)

[Описание базы данных 4](#_Toc102609934)

[Функционал бота 5](#_Toc102609935)

[1. Изначальное состояние 5](#_Toc102609936)

[2. Оформление заказа 6](#_Toc102609937)

[2.1. Ввод адреса 6](#_Toc102609938)

[2.2. Выбор времени 6](#_Toc102609939)

[2.3. Подтверждение и оформление заказа 8](#_Toc102609940)

[3. Список текущих заказов 9](#_Toc102609941)

[3.1. Просмотр и редактирование заказа. 10](#_Toc102609942)

[ЗАКЛЮЧЕНИЕ 11](#_Toc102609943)

**ВВЕДЕНИЕ**

Данная программа представляет собой бота, созданного для использования в мессенджере Telegram. Бот – это тот же пользователь Telegram с уникальным идентификатором, только с условием, что им нельзя управлять напрямую через клиент Telegram.

Основная функция данного бота – обработка входящих сообщений пользователей, на основе которых формируется заказ на выполнение какого-либо рода работ.

Для связи с ботом и для реализации его функционала используется доступный в свободном доступе Telegram API – интерфейс, позволяющий посылать HTTPS запросы на сервер Telegram для последующей обработки и выполнения их ботом. API позволяет задавать боту интерфейс для взаимодействия пользователей с ним.

Проект построен на архитектуре .NET с использованием языка программирования C#, а для связи с Telegram API была использована библиотека Telegram.Bot, доступная в менеджере пакетов NuGet. Для хранения данных выбрана технология Entity Framework, входящая в состав .NET 5 и предоставляющая удобный интерфейс для работы с СУБД MSSQL, представленной в данном проекте.

**Описание базы данных**

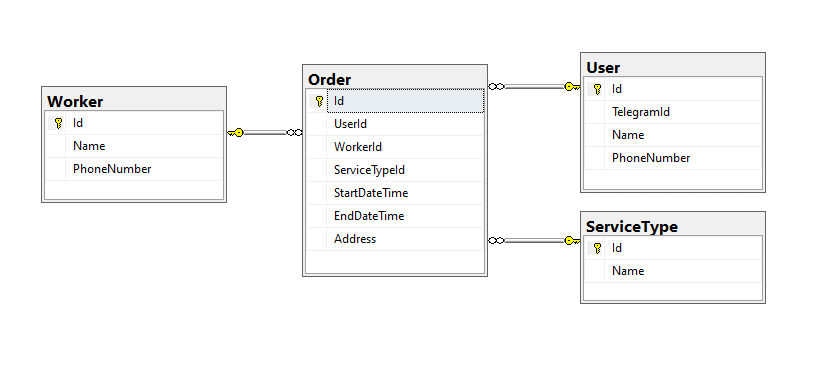


Рисунок 1. Схема базы данных приложения.

База данных содержит всего 4 таблицы, чего достаточно для такого небольшого бота.

Основная сущность Order – это заказ, формируемый на основе введенных пользователем данных. В заказе присутствуют: идентификатор заказа, идентификатор пользователя, идентификатор рабочего, идентификатор типа поставляемых работ, адрес выполнения работ, а также время начала и окончания работ.

Worker – сущность, представляющая собой рабочего, способного брать заказы. Содержит в себе следующие поля: идентификатор, имя рабочего и его номер телефона.

User – таблица, содержащая информацию о всех клиентах (пользователях), вступавших в контакт с ботом, пользовавшихся его функциями. Поля таблицы: идентификатор, идентификатор пользователя в Telegram, имя пользователя, его номер телефона.

ServiceType – таблица, содержащая все возможные типы выполняемых работ. Поля таблицы: идентификатор, наименование оказываемой услуги.

**Функционал бота**

Бот выполняет различные функции по работе с заказами. Каждый шаг оформления заказа, а также побочные шаги, не входящие в оформление заказа, представлены в коде отдельными состояниями, в которых может находиться пользователь в данный момент, реализованными с помощью паттерна состояния. Рассмотрим каждое возможное состояние по порядку.

**1. Изначальное состояние**

Когда пользователь впервые напишет сообщение боту (по умолчанию – команда /start), ему будет присвоено это состояние.

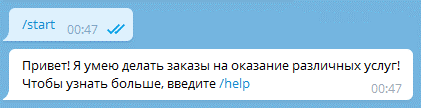


Рисунок 2. Ответ бота на команду /start.

Из этого состояния доступны следующие команды:

* /order – начало оформления заказа, переход в состояние оформления заказа (2);
* /list – вывод информации о текущих заказах, переход в состояние списка заказов (3).

Также доступны стандартные для всех состояний команды:

* /help – вывод информации о боте и инструкции по работе с ним;
* /cancel – отмена текущего действия, возврат в изначальное состояние.

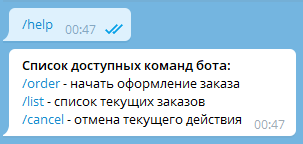


Рисунок 3. Ответ бота на команду /help.

**2. Оформление заказа**

В этом состоянии пользователю предлагается выбрать услугу для заказа. Список интерактивных кнопок формируется в коде из данных, содержащихся в базе данных в таблице ServiceType.

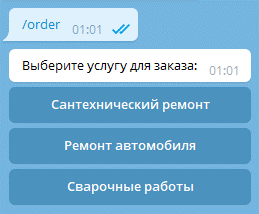


Рисунок 4. Ответ бота на команду /order.

**2.1. Ввод адреса**

По нажатию одной из кнопок пользователь переходит в состояние ввода адреса, по которому предполагается выполнение выбранного типа работ. В это же время клавиатура скрывается и в предыдущем сообщении отображается выбранная пользователем услуга.

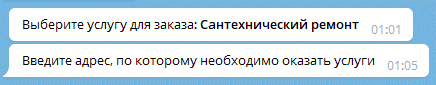


Рисунок 5. Ответ бота на выбор услуги из списка.

**2.2. Выбор времени**

После отправки адреса в ответном текстовом сообщении пользователю предлагается выбрать доступное время для заказа.

Реализован переход по страницам, где каждая страница – новый список, содержащий доступное время в пределах одного дня.

|  |  |  |
| --- | --- | --- |
| Рисунок 6. Ответ бота на ввод адреса.  Первая страница. | Рисунок 7.  Страница в середине. | Рисунок 8.  Последняя страница. |

Доступное время – время больше текущего на 2 часа. Если следующий день, то с 9:00 до 20:00. При этом определенное время может быть недоступно, если все рабочие в это время заняты. В этом случае недоступное время будет пропущено.

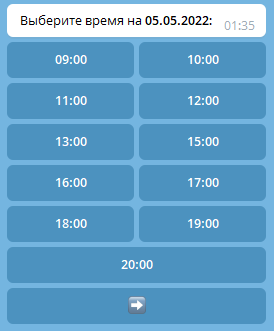


Рисунок 9. Недоступно время 14:00.

**2.3. Подтверждение и оформление заказа**

После выбора времени, пользователь переходит в состояние подтверждения заказа. В чат присылается вся информация о готовом заказе с просьбой подтвердить введенные данные. При этом клавиатура прошлого сообщения скрывается и выбранное время отображается в самом сообщении.

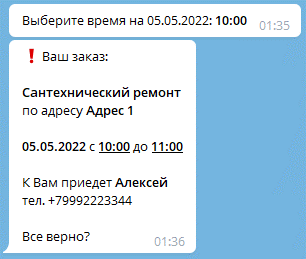


Рисунок 10. Ответ бота на выбор времени.

Одновременно с этим в чате появляется встроенная клавиатура с выбором из двух вариантов: подтвердить или вернуться назад.

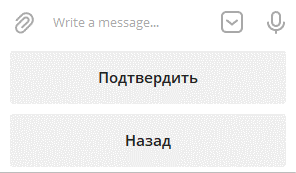


Рисунок 11. Кнопки подтверждения заказа.

По нажатии кнопки «Назад» пользователь возвращается к предыдущему состоянию (2.2) и ему предлагается выбрать другое время.

По нажатии кнопки «Подтвердить» в чат отправляется сообщение об успешном оформлении заказа. При этом в коде программы составляется заказ на основе введенных пользователем данных и записывается в базу данных, а пользователь возвращается в изначальное состояние (1).

**3. Список текущих заказов**

В этом состоянии пользователь работает со списком своих текущих (актуальных) заказов. В чат присылается сообщение со списком, выполненным в виде интерактивной клавиатуры, в которой кнопки содержат краткую информацию о каждом заказе. Заказы отсортированы по времени по возрастанию.

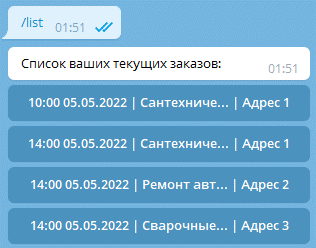


Рисунок 12. Ответ бота на команду /list.

При переполнении доступного на одной странице количества кнопок (в данном случае для удобства их шесть) в самом низу списка отображаются кнопки навигации.

|  |  |
| --- | --- |
| Рисунок 13.  Первая страница заказов. | Рисунок 14.  Последняя страница заказов. |

Когда пользователь нажимает на одну из кнопок, он перемещается в состояние просмотра и редактирования заказа (3.1).

**3.1. Просмотр и редактирование заказа.**

В этом состоянии пользователю выводится полная информация о выбранном заказе, а также предлагается отменить заказ или вернуться назад.

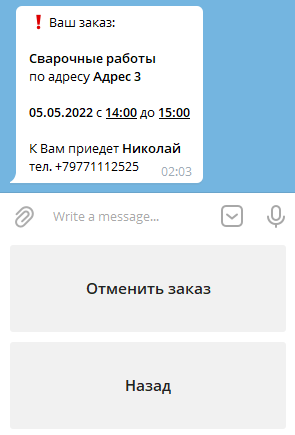


Рисунок 15. Диалог просмотра заказа.

По нажатии кнопки «Назад» пользователь возвращается к списку заказов (3).

По нажатии кнопки «Отменить заказ» в чат отправляется сообщение об успешной отмене заказа. В этот момент из базы данных удаляется запись, соответствующая выбранному заказу, а пользователь переходит в изначальное состояние (1).

**ЗАКЛЮЧЕНИЕ**

В боте для Telegram был успешно реализован функционал, соответствующий основным функциям по работе с записями в базе данных, таким как составление, добавление и удаление пользовательских заказов.