Министерство цифрового развития, связи и массовых коммуникаций РФ
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Сибирский государственный университет телекоммуникаций и информатики»
(СибГУТИ)

Кафедра ПМиК
Допустить к защите
зав. кафедрой: д.т.н.

_____ Нечта И.В.

ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА БАКАЛАВРА

Приложение для обработки базы инфраструктуры города

Пояснительная записка

Студент Енокян Ваге Арташесович /……/

Институт ИВТ Группа ИП-812

Руководитель к.т.н., доцент Янченко Е.В. / ……..……/

Новосибирск 2022 г.

Оглавление

ВВЕДЕНИЕ

На сегодняшний день информационные технологии оказывают огромное влияние на общество, меняя уклад и образ жизни. Одним из таких технологий являются боты.

Бот – умная программа, которая работает в мессенджерах и выполняет разные функции. Основными функциями бота является поддержка пользователя 24 часа в сутки; является маркетинговым инструментом для распространения контента, продвижения бизнеса и т.д. Правильное использование бота, поможет в продажах, а также предоставит необходимую информацию пользователям. Бот отличается от живого человека тем, что способен давать ответы большому количеству пользователей одновременно.

В настоящее время, большинство людей тратят не мало времени, в поисках общественных мест, которые хотят посетить. Есть не мало приложений где это можно осуществить, но для чего скачивать ещё одно дополнительное приложение, если это возможно осуществить с помощью приложения Telegram и telegram бота.

При выборе платформы, для разработки бота, был выбран Telegram.

Быстрый рост Telegram и претензии к конфиденциальности, безопасности помогли ему стать важной платформой для бизнеса и брендов. Многие компании и бренды используют приложение Telegram, для предоставления услуг поддержки клиентов. Telegram стал важной платформой, на которой нужно сосредоточиться, с увеличенной базой пользователей. Месячная аудитория Telegram в России, превышает 30 миллионов пользователей

Telegram – это приложение с открытым исходным кодом. Это позволило многим внешним разработчикам добавить функции в платформу.

Telegram bot – это стороннее приложение, которое работает внутри Telegram. Бот в Telegram может взаимодействовать с пользователями, отправляя сообщения. Telegram-бот – является компьютерной программой, которая может обслуживать компании или бренды многими функциями, такими как отправка информации, напоминания, воспроизведение мелодий, заказ и многое другое.

Пользователи могут взаимодействовать с telegram-ботом, отправляя сообщения. Telegram предоставляет API для создания ботов для социальных взаимодействий, производительности, игр и услуг электронной коммерции.

Несколько причин, по которым было принято решение разработки telegram бота:

1. Telegram – это бесплатная платформа с открытым исходным кодом, что означает, что Telegram позволяет использовать свой API и код любому человеку.
2. Каждый обмен сообщениями между пользователем и Telegram-ботом зашифрован до конца.
3. Мессенджер telegram охватывает все основные платформы, такие как Android, iOS, Windows phone, а также является десктопным приложением для Mac, Linux и Windows. Кроме того, он также имеет веб-

версию, которая позволит ориентироваться на потенциальных клиентов в широком масштабе

4. Telegram – бот может позволить настроить клавиатуру пользователя. Боты позволяют его создателю настраивать взаимодействие пользователя с ботом в telegram, заменяя традиционную клавиатуру опциями, связанными с функциональностью этого бота.

Также к telegram боту, было реализовано веб-приложение, где возможно также найти необходимое общественное место, с выводим его местоположения на карте.

Веб-приложение — это своего рода компьютерная программа. Он использует онлайн-технологии (включая браузеры) для выполнения огромного количества различных задач. Многие приложения используются для целей онлайн-ритейла. Тем не менее, они могут служить для самых разных целей, от заказа еды на вынос до поиска общественных мест.

Веб-приложения извлекают и хранят информацию с помощью сценариев на стороне сервера (на языках сценариев, таких как PHP и ASP), а сценарии на стороне клиента (на JavaScript и HTML5) представляют соответствующую информацию в пользовательском интерфейсе. Эта информация может принимать любую форму.

Основной идеей стало – поиск общественных мест, с простой формой просмотра их через мессенджер Telegram, либо через веб-приложение. То есть пользователю не придется устанавливать на свое устройство сторонних приложений, достаточно браузера и Telegram’а.

1 ПОСТАНОВКА ЗАДАЧИ

1.1 Основные требования к программному продукту

В telegram боте должны быть реализованы следующие функциональные возможности:

- Регистрация пользователя;
- Удаления пользователя из базы;
- Изменение данных пользователя:
    1) имя;
    2) фамилия;
    3) возраст;
    4) город;
    5) пол;
- Добавление общественного места в базу;
- Удаление общественного места из базы;
- Поиск общественного места: категории, по радиусу поиска;
- Вывод всех общественных мест;
- Получение контакта общественного места;
- Вывод информации об аккаунте пользователя;
- Вывод информации для пользователей и компаний;
- Вывод списка допустимых команд;
- Получении ссылки веб-приложения;

В веб-приложении должны быть реализованы следующие функциональные возможности:

- На странице категории выводятся все общественные места из базы;
- На странице общественного места, выводится следующая информация: Описание, время работы, номер телефона, расположение на карте;

При разработке должны учитываться следующие этапы:

- Реализация удобного, легко воспринимаемого дизайна начальной страницы и других разделов;
- Разработка правильной структуры веб-приложения;
- Создание удобного, простой навигации по веб-приложению;
- Правильно расположенный текст и изображения на страницах;

1.2 Выбор программных средств

Для реализации поставленных задача, были выбраны следующие средства:

1. Операционная система Linux
2. Среда разработки Visual Studio Code
3. Языки программирования Python, PHP, JS
4. СУБД MySQL

1.2.1 Используемые технологии для разработки Telegram bot

Python - популярный язык программирования общего назначения, который может использоваться для широкого спектра приложений. Он включает в себя высокоуровневые структуры данных, динамическую типизацию, динамическую привязку и многие другие функции, которые делают его столь же полезным для разработки сложных приложений, как и для сценариев или "клеевого кода", соединяющего компоненты вместе. Он также может быть расширен для выполнения системных вызовов практически всех операционных систем и запуска кода, написанного на C или C ++. Благодаря своей повсеместности и способности работать практически на любой системной архитектуре, Python является универсальным языком, который можно найти в самых разных приложениях.

- Он может подключаться к системам баз данных и может читать и изменять файлы.
- Он поддерживает быстрое прототипирование и разработку программного обеспечения.
- Это интерпретируемый язык, который означает, что интерпретатор реализует код построчно за один раз.
- Он может быть легко интегрирован с такими языками, как C, JAVA, C ++.
- Это очень простой в освоении язык, что делает его удобным для разработчиков.

1.2.2 Используемые технологии для разработки веб-приложения

PHP - это серверный язык сценариев, который в основном используется при создании динамических веб-страниц в Интернете, но эволюционировал, чтобы иметь возможность выполнять сценарии командной строки и клиентские приложения. Он может использоваться на всех основных операционных системах и поддерживает большинство веб-серверов. Как и во многих языках программирования, одна из распространенных вещей для разработчиков - это доступ к данным в базе данных.

PHP приобрел большую часть своей популярности благодаря своей способности подключаться к широкому числу баз данных. Поскольку PHP рассматривается как свободное программное обеспечение (под лицензией PHP), неудивительно, что одной из наиболее распространенных баз данных, используемых с ним, является MySQL.

JavaScript - это текстовый язык программирования, используемый на стороне клиента, а также и на стороне сервера, который позволяет делать веб-страницы интерактивными. Где HTML (HyperText Markup Language) используется для разметки различных видов содержимого (например, таблиц, абзацев, списков, гиперссылок, изображений, видео и т. Д.) На веб-странице с тегами. Пользовательский интерфейс (UI) построен с использованием HTML. CSS (каскадные таблицы стилей) определяет стиль документа разметки. Например, цвета, шрифты, выравнивания, макет и т. Д. Каждый

презентабельный HTML-элемент имеет набор свойств стиля, изменяемых с помощью CSS.

Обычно клиент или пользователь отправляет запрос на веб-сервер сайта, который он хочет посетить. Веб-сервер, имеющий собственный IP-адрес, хранит все файлы в бэкэнде, которые могут быть написаны на PHP, Python или Node.js. Веб-сервер отправляет ответ клиенту в виде HTML, CSS и JavaScript.

1.2.3 Среда разработки

Интегрированная среда разработки (IDE) - это программное обеспечение, которое облегчает разработку приложений.

В качестве IDE был выбран Visual Studio Code. Это текстовый редактор с открытым исходным кодом, разработанный Microsoft и бесплатный в использовании. Он известен тем, что относительно легкий, а также включает в себя ключевые функции, найденные в современных IDE, такие как интеграция Git и обширный отладчик. Это делает VS Code отличным для всего, от простых сценариев Python до более плотных проектов разработки программного обеспечения.

1.2.4 Операционная система

ОС - это фоновое программное обеспечение, которое подключается и взаимодействует со всеми другими программами на компьютере или интеллектуальном устройстве.

Linux - это операционная система на базе Unix, она включает в себя весь потенциал Unix, интернет-утилиты, среды разработки, полнофункциональный интерфейс рабочего стола и множество приложений. Традиционное монолитное ядро используется в ядре Linux для повышения производительности. Его модульная функция позволяет большинству драйверов динамически загружаться и выгружаться во время выполнения

Данная операционная система была выбрана, по ряду причин:

а) Удобство в использовании для разработчиков. Установка программного обеспечения в Linux особенно проста по сравнению с другими ОС. Данный процесс значительно улучшает рабочий процесс при разработке.
б) В Linux представлено множество удобных и полезных инструментов программирования.
в) Время разработки драгоценно, поэтому использование Linux делает разработку проще и интереснее.
г) Так же, Python предустановлен в Linux, поэтому нет необходимости его устанавливать.

1.3 Технологии разработки для Telegram bot

Bot API - это HTTP-интерфейс, который создан для разработчиков, желающих создавать ботов для Telegram. Каждый бот получает уникальный токен при создании.

Токен - это строка 120001621:FFHdtYegCH1vGWJxfSeofSAs0T5qDORgwe, которая необходима для авторизации бота и отправки запросов в API бота.

Каждый бот является специальным аккаунтом, созданный для автоматического обрабатывания и отправления сообщений.

Есть два взаимоисключающих способа получения обновлений для вашего бота — метод getUpdates с одной стороны и Webhooks с другой. Входящие обновления хранятся на сервере, пока их не получит бот. Но они не будут храниться более 24 часов. Внезависимо от способа получения обновлений в ответ отправляется объект Update, сериализованный в JSON.

Все запросы к Telegram Bot API должны обслуживаться через HTTPS и должны быть представлены в следующей форме: https://api.telegram.org/bot<token>/METHOD_NAME.

BotFather отвечает за реализацию оболочки и выдает токен для работы с Telegram API. В случаи необходимости токен может быть заменен на другой.

Для изменения внешнего вида бота, используются различные команды, представленные в таблице 1.1.

Таблица 1.1 – Список команд редактирование ботов

/setname	Изменение имени бота
/setdescription	Изменение описания бота, короткий текст до 512 символов, описывающий бота.
/setabouttext	Изменение информации о боте. Пользователи увидят этот текст на странице профиля бота.
/setuserpic	Изменение фотографии профиля бота
/setcommands	Изменения списка команд, поддерживаемых ботом.
/deletebot	Удаление бота.

Для создаваемого бота найдены необходимые библиотеки, которые понадобятся в создании данного проекта.

1.3.1 Библиотека pytelegrambotapi

Разработка бота содержит большое количество стандартной работы, это:

- Запуск цикла;
- Анализ полученного JSON;
- Обработка сообщений с учётом значения offset;
- Ответ на сообщения;
- Отправка запросов, для ответа на сообщение;

Одна из наиболее популярных библиотек — pytelegrambotapi (telebot).

1.3.2 Библиотека geopy

При работе с геолокациями, использовалась библиотека – geopy. Geopy позволяет разработчикам Python легко находить координаты адресов, городов, стран и достопримечательностей по всему миру с помощью сторонних геокодеров и других источников данных.

Для получения координат, используется прямое геокодирование, для определения координат по названию адреса объекта или его названия. При отправке пользователем геолокации, данная геолокация преобразовывается в координаты и записывается в базу данных.

Для получения адреса из координат используется обратный геокодинг. Обратное геокодирование - процесс преобразования местоположения, описанного географическими координатами (широта, долгота), в удобный адрес или название места. Данный процесс также используется для вывода местоположение объекта на карте.

Для работы с геолокацией необходимо зарегистрировать Яндекс аккаунт и перейти на страницу разработчика, для дальнейшего получения API. API - это аббревиатура интерфейса прикладного программирования, который представляет собой программный посредник, позволяющий двум приложениям общаться друг с другом. После выполнения нескольких шагов, разработчик получает ключ геокодирования.

1.3.3 Библиотека pymysql

В качестве СУБД был выбран всеми любимый MySQL. Для подключения к серверу баз данных MySQL из python, использовался – pymysql. PyMySQL - это клиентская библиотека, основанная на PEP 249. Большинство публичных API совместимы с mysqlclient и MySQLdb. PyMySQL работает с MySQL 5.5+ и MariaDB 5.5+.

В ходе реализации взаимодействия с БД, выполнили следующие шаги:

1) Зная адрес хостинга, логин администратора, имя базы и пароль, реализовывать подключение;
2) Формирование запросы к базе;
3) Реализация функции добавления/удаления/перезаписи;
4) Реализация получения данных из базы;

1.3.4 Библиотека telegram_bot_pagination

Библиотека обеспечивает простой способ создания нумерации номеров с помощью встроенной клавиатуры для telegram bot на python.

Пример работы изображен на рисунке 1.1.

Листинг 1.1 – Пример кода пагинации в telegram боте

```
def send_character_page(message, page=1):
        character_pages = places_base.get_pages()
        id = places_base.get_my_id()
```

```
            photo = places_photo.print_photo(*id[page-1])
            paginator = InlineKeyboardPaginator(
                    len(character_pages),
                    current_page=page,
                    data_pattern='character#{page}'
            )

            config.bot.send_photo(chat_id=message.chat.id,
photo=open(photo,        'rb'),        caption=character_pages[page-
1],reply_markup=paginator.markup)


    @config.bot.callback_query_handler(func=lambda           call:
call.data.split('#')[0]=='character')
    def characters_page_callback(call):
        page = int(call.data.split('#')[1])

        config.bot.delete_message(
            call.message.chat.id,
            call.message.message_id
        )
        send_character_page(call.message, page)
```

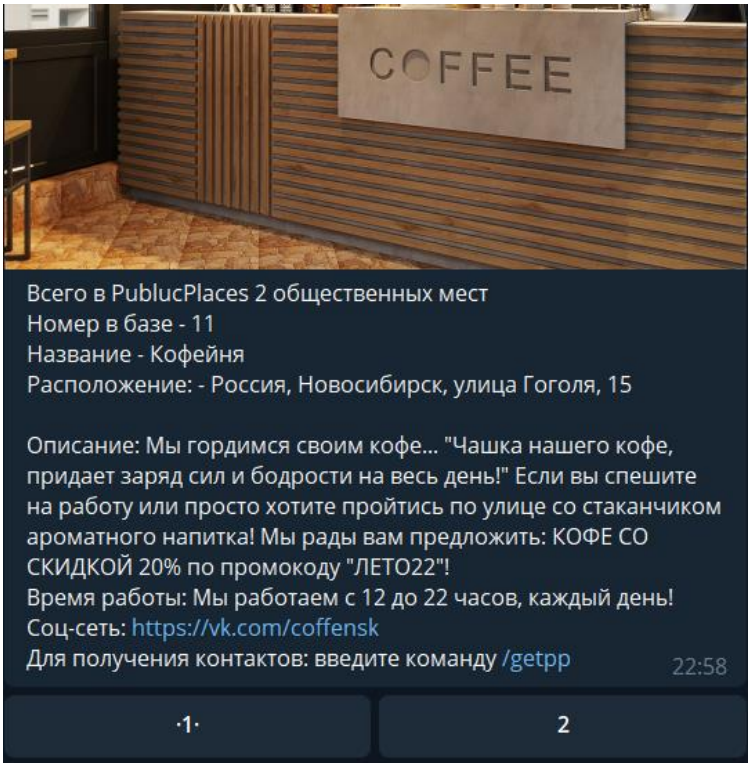

Рисунок 1.1 – Пример работы пагинации в telegram

1.3.5 Библиотека requests

Requests - это элегантная и простая библиотека Python, созданная для легкой обработки HTTP-запросов в python. Он позволяет делать GET, POST, PUT и другие типы запросов и обрабатывать полученный ответ гибким Pythonic

способом. Данная библиотека использовалась для получения данных геолокации.

1.4 Технологии разработки для веб-приложения

При создании веб-приложения, использовались такие языки программирования как:

- JavaScript - использовался для создания анимации на странице веб-приложения, получения и вывода данных на карте. Для данных случаев использовалась многофункциональная библиотека JQuery;
- PHP - использовался для работы с база данными;

1.4.1 Библиотека JQueri

jQuery - это библиотека JavaScript с открытым исходным кодом, которая упрощает создание и навигацию веб-приложений. В частности, jQuery упрощает манипулирование объектной моделью HTML-документа (DOM), асинхронный JavaScript и XML (Ajax), а также обработку событий. Кроме того, jQuery включает в себя функции JavaScript, манипулируя свойствами CSS, чтобы добавить эффекты, такие как затухание и выходы для элементов веб-сайта. jQuery - это широко используемая библиотека JavaScript, поддерживаемая тысячами пользовательских плагинов.

1.5 Сервер базы данных MySQL

База данных - это логически организованный набор информации, разработанный таким образом, что информация внутри может быть доступна для последующего использования компьютерной программой.

Язык структурированных запросов, широко известный как SQL, является стандартным языком программирования для реляционных баз данных. Несмотря на то, что он старше многих других типов кода, это наиболее широко реализованный язык баз данных. SQL используется для запроса и управления базовыми реляционными базами данных, такими как SQL Server, Oracle, MySQL, PostgreSQL, SQLite и т. Д.

SQL - это стандартный язык ANSI (American National Standards Institute) и ISO (International Organization for Standardization). Однако не все базы данных поддерживают один и тот же SQL, но есть небольшие различия. Кроме того, большинство баз данных включают свое собственное дополнение к SQL.

MySQL - это бесплатная система управления реляционными базами данных с открытым исходным кодом, которая использует язык структурированных запросов (SQL), самый популярный язык для добавления, доступа и обработки данных в базе данных. MySQL известен своей скоростью, надежностью и гибкостью и широко используется для веб-приложений с середины 1990-х годов.

Данные в базе данных MySQL хранятся в таблицах. Таблица - это набор связанных данных, состоящий из столбцов и строк.

Если сравнивать MySQL с другими СУБД с открытым исходным кодом, то основное отличие заключается в плавной работе интерфейса API. Это программное обеспечение позволяет любому пользователю получить доступ к системе управления базами данных, даже если для написания использовался любой другой язык программирования. Для оптимального администрирования сайта MySQL хорошо сочетается с таким языком программирования как PHP.

Данный тип базы данных, был выбран из-за ряда причин:

1) Простота эксплуатации
2) Богатая функциональность;
3) Безопасность;
4) Может работать с большими объемами данных;
5) Высокая скорость работы;

Существует три типа метолов в PHP для подключения базы данных MySQL через бэкенд:

- MySQL
- MySQLi
- PDO

MySQLi - это API, используемый в качестве функции соединителя для связи бэкэнда приложения PHP с базой данных MySQL. Он работает так же, как и предыдущая версия, но он безопаснее и быстрее, а также предоставляет лучший набор функций и расширений. MySQLi был представлен с PHP 5.0.0, а драйверы были установлены в 5.3.0. API был разработан для поддержки MySQL от версии 4.1.13 до более новых.

PDO - PHP Data Object универсальный интерфейс данных, что-то вроде mysqli, но не специфичный для MySQL, его можно использовать для соединения с другими СУБД

Основное различие между PDO и Mysqli заключается в том, что PDO поддерживает различные базы данных, а mysqli поддерживает только MySQL.

2 ОПИСАНИЕ РАЗРАБОТКИ

2.1 Технология разработки telegram bot

2.1.1 Регистрация бота

Первым шагом при реализации приложения является регистрация у бота «BotFather». Данные бот помогает создать оболочку нашего бота.

Рисунок 2.1 - BotFather – самый простой способ для регистрации, настройки и управления другими telegram-ботами

Первым этапом регистрации реализуется с помощью команды «/newbot», после чего BotFather предлагает вести название нашего бота, при этом имя должно выполнить обязательно условие: заканчивается на «Bot» или «_bot». В нашем случаи получилось - «PublicPlacesBot».

После выполнения данных условий, BotFather выдает URL-адрес для доступа к боту и уникальный токен (набор символов для доступа к HTTP API). С помощью команды «/mybots» добавляется фотография боту, описание, которое отображается при первом открытии бота и добавляется список команд.

После настроек бота и получения специального токена, начинается этап разработки программной части бота. Для реализации бота использовались специальные библиотеки языка программирования Python.

2.1.2 Реализация начала работы бота

Начала работы бота начинается с команды «/start». Данная команда инициализирует общение бота с пользователем. При этом проверяет

пользователей на раннюю регистрацию. Как начала работа приводится в листинге 2.1.

Листниг 2.1 - Код приветственной команды «/start»

```
@config.bot.message_handler(commands=['start'])
def start_message(message):
    if antiddos(message) == False:
        if    (user_base.check_user(message.from_user.id)    !=
False)  or  user_base.users_dictionary.get(message.from_user.id)  !=
None:
            config.bot.send_message(message.from_user.id,  "Мы
уже стартанули ;)")
        else:
         config.bot.send_message(message.from_user.id,
config.HELLO_REG)
```

После выполнения команды бот отправляет нам приветственно сообщение, в соответствии с рисунком 2.2.

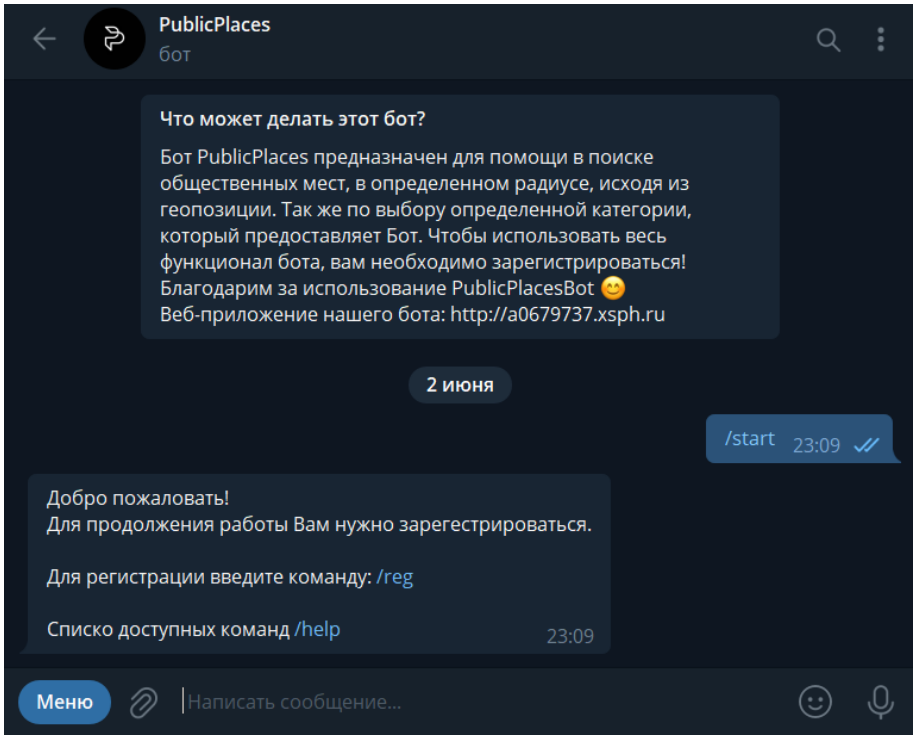

Рисунок 2.2 - Выполение команды «/start»

Для того, чтобы пользователи не нагружали бота спамами, была реализована функция «antiddos» - при многочисленном спаме, бот выдает предупреждение пользователю, в противном случае пользователь блокируется на пять минут.

2.1.3 Реализация регистрации или удаление пользователя

Для дальнейшего использования функционала бота, пользователю необходимо зарегистрироваться. Регистрация осуществляется с помощью

команды «/reg». При выполнении команды, осуществляется проверка, зарегистрирован ли пользователь или нет.

Если раннее был зарегистрирован, то выводит соответствующее сообщение – «Этап регистрации уже пройден».

Регистрация выполняется в несколько этапов, в соответствии с рисунком 2.3:

1. Ввод имени
2. Ввод фамилии
3. Ввод возраста
4. Ввод пола
5. Ввод города

В функции получении имени и фамилии стоит валидация на проверку ввода, чтобы пользователь кроме латинских и русских букв, не мог ввести другие символы. Если введенные данные удовлетворяет валидиацию, тогда осуществляет переход к следующей функции. В противном случаи, повторно вызывается функция ввода имени.

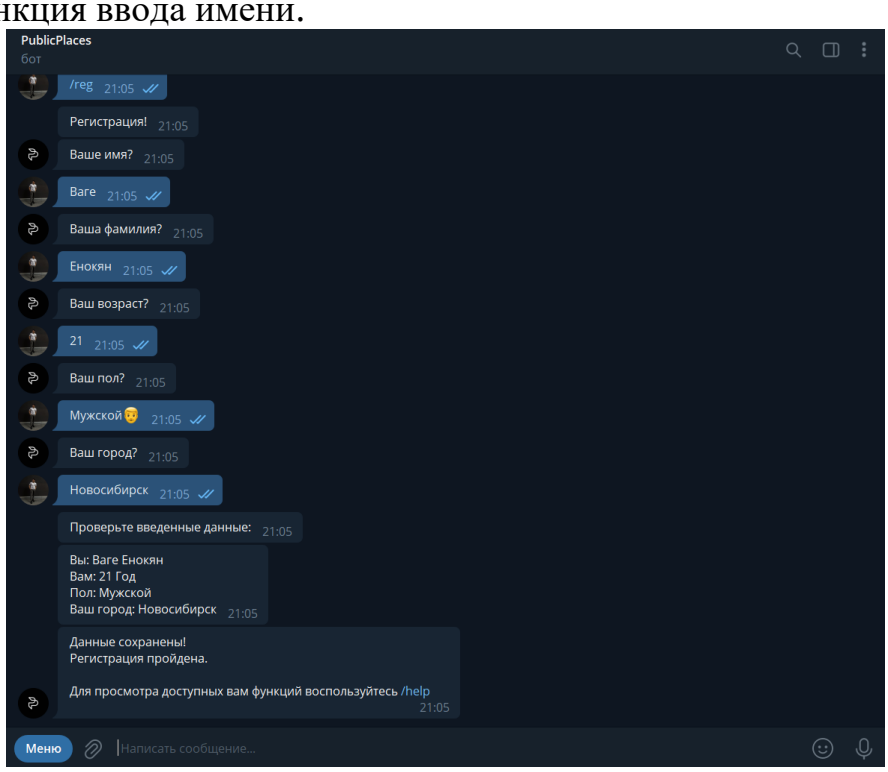

Рисунок 2.3 – Демонстрация регистрации пользователя в боте

Для того, чтобы из одной функции вызвать другую, используется register_next_step_handler, принимающая в аргументы объект отправленного сообщения и имя следующей функции, листинг 2.2.

Листинг 2.2 – функция, отвечающая за получение имени пользователя

```
def get_name(message):
    name = message.text.title()
    if message.text == 'Стоп' + emoji.emojize(':raised_hand:'):
        stop_reg(message.from_user.id)
    elif validation.validateName(name):
```

```
user_base.users_dictionary[message.from_user.id].append(name)
            config.bot.send_message(message.from_user.id,    "Ваша
фамилия?")

config.bot.register_next_step_handler(message,get_surname)
        else:
            config.bot.send_message(message.from_user.id,     "Имя
введено не корректно! Повторите попытку.")

config.bot.register_next_step_handler(message,get_name)
```

Также валидацией проверяется ввод возраста, чтобы пользователь не вводил буквы или другие символы, при этом пользователю должно быть больше 12 лет. При выборе пола, у пользователя появляются две кнопки выбора: Мужской, Женский. Пользователю запрещено вводить другие слова в это поле.

Для создания кнопки необходимо создать переменную(rmk) и поместить в нее types.ReplyKeyboardMarkup(resize_keyboard=True). В котором resize_keybord=True выполняет функцию адаптации. Создаем переменную с кнопками, добавляем их текст, при желании можно добавить смайлики. После всего проделанного добавляем эти переменные командой rmk.add(имя перемнной). Чтобы кнопки отображались, необходимо в методе send_message, добавить reply_markup = rmk. Чтобы украсить кнопки, с помощью библиотеки emoji, были добавлены подходящие картинки. Пример создание кнопок приведен ниже, в листинге 2.3.

Листинг 2.3 – создание кнопок в боте

```
rmk = types.ReplyKeyboardMarkup(resize_keyboard=True)
rmk.add(types.KeyboardButton("Мужской"  +  emoji.emojize(':man:')),
types.KeyboardButton("Женский"     +     emoji.emojize(':woman:')),
types.KeyboardButton("Стоп" + emoji.emojize(':raised_hand:')))
            config.bot.send_message(message.from_user.id,     "Ваш
пол?", reply_markup=rmk)
```

Во время этапа регистрации, в любой момент, пользователь может нажать на кнопку «Стоп» и регистрация будет остановлена. При повторной регистрации, все этапы регистрации придётся начать заново.

В конце ввода всех данных, бот предлагает пользователю сохраниться или отменить регистрацию. При сохранении данные пользователя добавляются в базу данных. После регистрации, пользователь получает доступ ко всему функционалу бота.

Если же пользователь захочет изменить свои данные, то для этого предусмотрен ряд команд, которые перезаписывают данные пользователя.

- /name – перезаписывает имя
- /surname - перезаписывает фамилию
- /age - перезаписывает возраст

– /city - перезаписывает город
– /gender - перезаписывает пол

Удаление пользователя из базы, осуществляется с помощью команды «/del». Данная команда удаляется пользователя и все добавленные им записи из базы.

2.1.4 Реализация информационных функций

Далее реализуются ряд команд, служащие для вывода нужной информации:

– «/web» - вывод url-адрес веб-приложение. Итог смотреть на рисунке 2.4;
– «/help» - список доступных команд. Итог изображен на рисунке 2.5;
– «/info» - информация о боте. Итог смотреть на рисунке 2.6;
– «/infoempl» - необходимая информацию для пользователей, которые хотят добавить общественное место в базу. Итог смотреть на рисунке 2.7;
– «/infopp» - информацию для пользователей. Итог смотреть на рисунке 2.8;
– «/mypp» - вывод всех добавленных записи пользователя. Итог смотреть на рисунке 2.9;

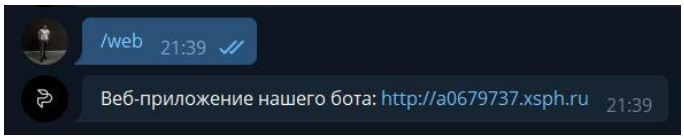

Рисунок 2.4 – выполнение команды /web

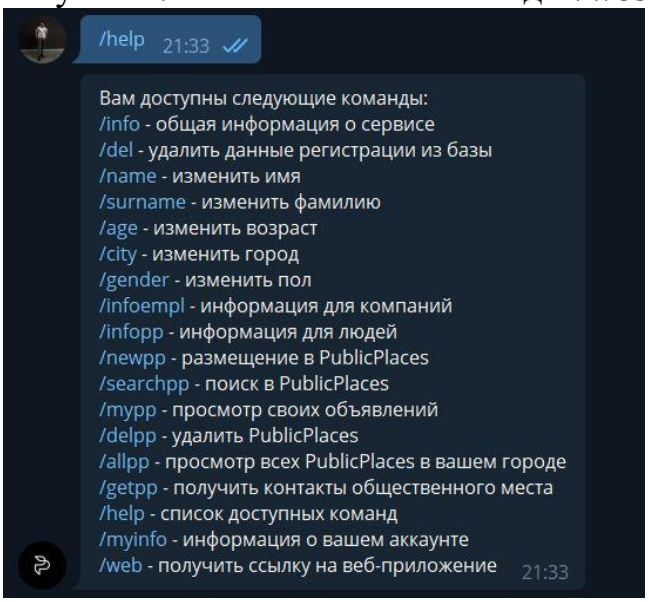

Рисунок 2.5 – выполнение команды /help

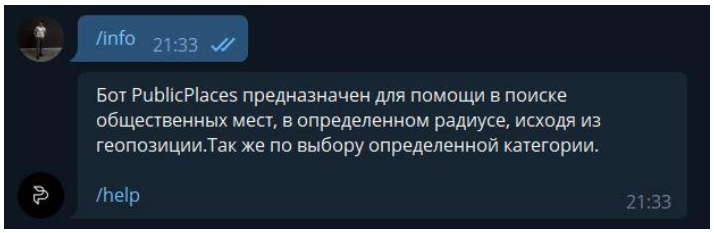

Рисунок 2.6 – выполнение команды /info

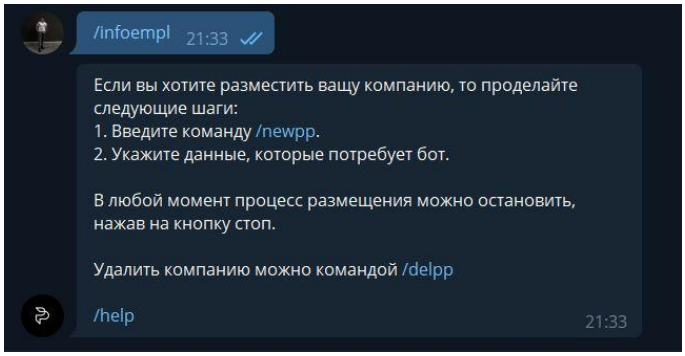

Рисунок 2.7 – выполнение команды /infoempl

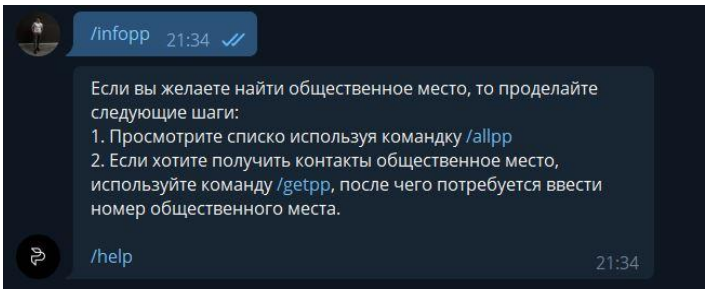

Рисунок 2.8 – выполнение команды /infopp

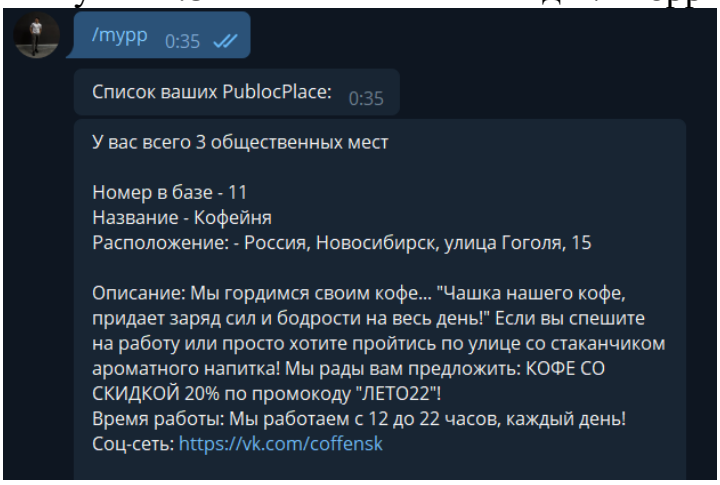

Рисунок 2.9 – выполнение команды /mypp

2.1.5 Реализация добавления и удаления общественных мест

После того как пользователь зарегистрировался, у него появляется возможность добавлять общественные места. Для это необходимо воспользоваться командой «/newpp». Добавление осуществляется несколькими этапами, блок-схемы изображено на рисунке 2.10.

Рисунок 2.10 – Блок схемы работы команды «/newpp»

Как видно на блок-схеме, добавление начинается с ввода названия общественного места. После этого пользователю предлагают ввести номер телефона, при этом ввод должен удовлетворить условие валидации. Номер телефона пользователя должно начинаться с чисел +7 или 8, длина номера не должно превышать 11 чисел.

Следующий этап, это отправка геолокация. Пользователю позволяется вводить местоположение в формате – Новосибирск, ул. Гурьевская, д. 51. При использовании мобильного устройства, пользователь имеет возможность отправить геолокацию с помощью выбора геопозации на карте, для этого

необходимо нажать по иконке скрепки, выбираем место на карте и прикрепить геопозицию (Location).

Геолокация пользователя записывается в формате – долгота, ширина. Это необходимо, чтобы в дальнейшем было удобно находить расстояние до общественного места. Ниже приведен кода, отвечающее за получение отправленной геолокции - Листинг 2.4

Листинг 2.4 – Функция получения геолокации от пользователя

```
    def get_location(message):
        if message.text == 'Стоп' + 
emoji.emojize(':raised_hand:'):
            stop_newpp(message.from_user.id)
            return
        coords = ""
        print("Геолокация от ")
        print(message.from_user.id)
        print(str(message.text) + "\n\n\n")
        if message.text == None:
            current_position = (message.location.longitude, 
message.location.latitude)
            coords = 
f"{current_position[0]},{current_position[1]}"
        else:
            coords = check_coords(message.text)

        street = location.get_address_from_coords(coords)
        if street == False:
            config.bot.send_message(message.from_user.id, 
config.ERROR_LOCATION)
            config.bot.send_message(message.from_user.id, 
config.LOCATION)
            config.bot.register_next_step_handler(message, 
get_location)
        else:

places_base.pps_dictionary[message.from_user.id].append(coords)
            config.bot.send_message(message.from_user.id, 
"Опишите общественное место")
            config.bot.register_next_step_handler(message, 
get_about)
```

Следующий этап, это описание общественного места, времени работы. После этого, необходимо выбрать категорию, к которому она относиться. В выборе категории, доступны следующее: поесть, красота, цветы, медицина, развлечение. Выбор категории, также даёт возможность в будущем легко находить общественное место.

После всего этого, необходимо отправить фотографию, при этом если пользователь добавляет через персональный компьютер, ему необходимо выбрать сжатый формат. Фотографии в таблице хранятся в формате «BLOB».

BLOB (большой двоичный объект) – тип данных MySQL, который может содержать переменный объем данных. Он используется для хранения двоичных данных и для столбцов двоичного хранилища большой емкости.

В конце пользователю, необходимо добавить социальную сеть, принадлежащее общественному месту. После завершения заполнения данных, необходимо нажать опубликовать.

На каждом этапе заполнения, есть возможно остановить процесс, при повторном добавлении, всю информацию необходимо заполнить заново. После того как пользователь, нажал опубликовать, данные записываются в базу. При этом, добавление фотографии хранятся в отдельной таблице.

Удаление осуществляется с помощью команды «/delpp», данная команда удаляет запись из базы данных. В листинге 2.5 приведен код удаления данных из базы. В функцию подается номер записи в базе, которую пользователь ранее добавил. Функция проверяет, есть ли в базе данный номер, в случаи успеха запись удаляется, в противные случаи выводится сообщение, что данная запись не найдена. Удаление осуществляется после нажатия на кнопку - «Удалить», при отмене нажимается кнопка - «Оставить».

Листинг 2.5 – Функция, отвечающая за удаление общественного место из базы

```
    def del_places(message):
        flag_number = True
        number = 0
        if message.text == 'Стоп' + 
emoji.emojize(':raised_hand:'):
config.bot.send_message(message.from_user.id,"Остановил.",reply_ma
rkup=del_keyb)
            return
        try:
            number = int(message.text)
        except Exception:
            flag_number = False
            config.bot.send_message(message.from_user.id, 
"Цифрами, пожалуйста")
            number = int(0)
        if number > 0 and flag_number == True:
            contact = 
places_base.get_my_pp_id(message.from_user.id,number)
            if contact == False:
                config.bot.send_message(message.from_user.id, 
"Общественное место с таким номером не найдено!\nПовторите 
попытку.")
                config.bot.register_next_step_handler(message, 
del_places)
            else:
                dictionary_del[message.from_user.id] = []

dictionary_del[message.from_user.id].append(number)
                keyboard = types.InlineKeyboardMarkup()
```

```
                key_del                                                 =
types.InlineKeyboardButton(text='Удалить', callback_data='delw')
                keyboard.add(key_del)
                key_leav=
types.InlineKeyboardButton(text='Оставить', callback_data='leave')
                keyboard.add(key_leav)
                config.bot.send_message(message.from_user.id,
contact, reply_markup=keyboard)
        else:
            config.bot.send_message(message.from_user.id,
"Общественное место с таким номером не найдено!\nПовторите
попытку.")
            config.bot.register_next_step_handler(message,
del_places)
```

2.1.6 Реализация поиска общественных мест

Поиск осуществляется с помощью команды «/searchpp». Ниже на рисунке блок-схемы 2.11, отображаются этапы поиска.

Рисунок 2.11 – Блок-схема работы команды /searchpp

В начале поиска необходимо выбрать категорию, таким способом диапозон поиска уменьшается. В выборе категории доступно следующее: поесть, красота, цветы, медицина, развлечение.

Следующим этапом, является отправка геолокации и выбор радиуса поиска. Данный способ дает возможность находить уменшить диапозон поиска, чтобы не выводить все записи. Зная координаты, бот выводит записи с указанным расстоянием до общественного места. Это поможет пользователю, знать как далеко находиться желаемое место.

Поиск растоянния осуществляется с помощью функции location_radar, привиденний в листинге 2.6.

Листинг 2.6 – Функция, для нахождения расстояния между двумя точками

```
    def location_radar(loc1: str, loc2: str):
        if    validation.validationGeo(loc1)    ==    False    or
validation.validationGeo(loc1) == False:
            return -1
        coord1 = loc1.split(',')
        coord2 = loc2.split(',')
        latitude_1 = float(coord1[0])
        longitude_1 = float(coord1[1])
        latitude_2 = float(coord2[0])
        longitude_2 = float(coord2[1])
        longitude_1,   latitude_1,   longitude_2,   latitude_2   =
map(radians, [longitude_1, latitude_1, longitude_2, latitude_2])
        dlon = longitude_2 - longitude_1
        dlat = latitude_2 - latitude_1
        a = sin(dlat/2)**2 + cos(latitude_1) * cos(latitude_2) *
sin(dlon/2)**2
        c = 2 * asin(sqrt(a))
        r = 6371
        res = c * r

        return res
```

В функцию подаются две локации, в первое подается геолокация общественно места, во второе указнная пользователем. С помощью метода split, получаем список слов в строке. Далее используем формула хаверсина, которая определяет расстояние по большой окружности между двумя точками на сфере с учетом их долготы и широты. Dlon и dlat – являются расстоянием между двумя точками, где r – является радиусом земли(6371 км).

2.1.7 Реализация просмотра всех объявлений

Для того, чтобы посмотреть все объявления, необходимо прописать команду «/allpp». Данная команда выводит все записи, вне зависимости от категории и локации. Код данной команды приведен в листинге 2.7

Листинг 2.7– Код функции команды «allpp»

```
@config.bot.message_handler(commands=['allpp'])
def start_message(message):
    if antiddos(message) == False:
```

```
            if    (user_base.check_user(message.from_user.id)    !=
False):
                config.bot.send_message(message.from_user.id,
"Список доступных PublicPlaces в вашем городе:")
                cw = places_base.count_places()
                if cw > 0:
                    allpp.send_character_page(message)
                else:
                    config.bot.send_message(message.from_user.id,
"К сожаления в вашем городе нет доступных PublicPlaces" +
emoji.emojize(':pensive_face:'))
            else:
                config.bot.send_message(message.from_user.id,
"Сначала зарегистрируйтесь! " + emoji.emojize(':grinning_face:') +
" /reg")
```

При запуске команды, с помощью оператора «count» получаем количество строк из таблицы общественных мест, если количество больше нуля, производим вывод, в противном случаи, бот выводит ошибку. Если пользователь не зарегистрирован, бот предлагает сначала выполнить регистрацию. Для удобства, записи выводится по страницам, благодаря пагинации.

2.1.8 Реализация получения контактов

Для получения контактов записи, пользователю необходимо прописать команду «/getpp» и отправить номер общественного места. Номер можно посмотреть при выводе всех записей или поиске. После отправки, бот выдает следующие данные: имя и фамилию, номер и город пользователя, который добавил данное общественное место.

2.1.9 Реализация базы данных

Весь функционал работы с базой данных осуществляет с помощью библиотеки pymysql. Подключение к базе происходит с помощью функции get_connetction, листниг 2.8

Листинг 2.8 – функция, отвечающая за подключение к базе данных

```
def get_connection():
global __connection
if __connection == None:
__connection = pymysql.connect(**config)
return __connection
```

Для подключения к базе, необходимы следующие значения:

- Имя сервера
- Имя пользователя
- Пароль
- Имя базы данных

Создание таблиц происходит при запуске бота, в случаи, если ранее они не были созданы. При этом, таблица «category» заполняется автоматически. Таким способом, в базе можно легко записать номер определенной категории.

Иерархия базы данных отображен на рисунке 2.12.

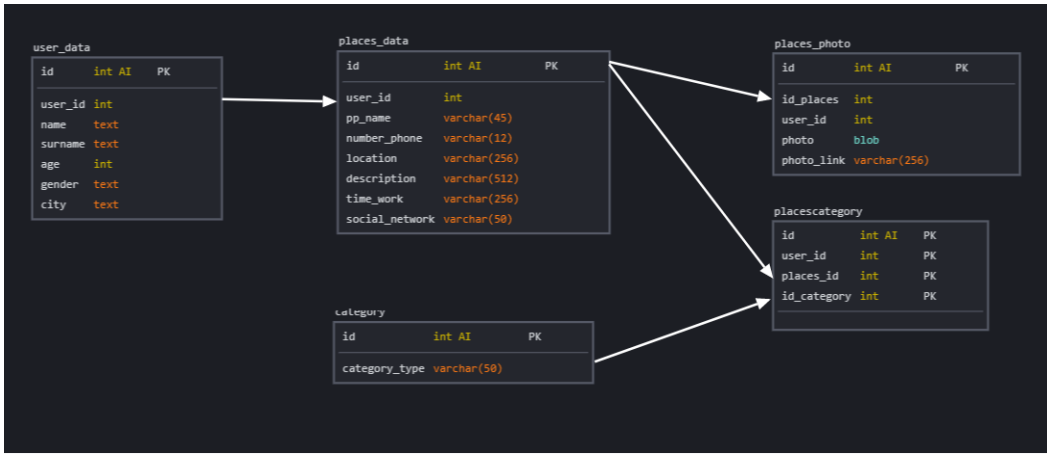

Рисунок 2.12 – Иерархия базы данных PublicPlaces

Создание базы данных осуществляется с помощью функций init_db. Создание остальных таблиц, происходит идентичным способом, листинг 2.9.

Листинг 2.9 – Создание таблицы пользователей

```
def init_db(forse: bool = False):
    conn = get_connection()
    c = conn.cursor()

    if forse:
        c.execute('''
            CREATE TABLE IF NOT EXISTS places_data (
                id    INTEGER    PRIMARY    KEY    not    null
AUTO_INCREMENT,
                user_id INTEGER not null,
                pp_name VARCHAR(40) not null,
                number_phone VARCHAR(12) not null,
                location VARCHAR(256) not null,
                description VARCHAR(512) not null,
                time_work VARCHAR(256) not null,
                social_network VARCHAR(50) not null
                )
        ''')

    c.close()
    conn.commit()
```

Перезапись данных в базе, осуществляет с помощью оператора «Update» и «Set», чтобы указать, какие столбцы и значения должны быть обновлены в таблице, пример отображается в листинге 2.10. Возможность обновлять записи в базе данных это ключ к тому, чтобы данные велись аккуратно, и был полный порядок. Перезапись других записей, выполнен идентичным способом.

Листинг 2.10 – изменение имени пользователя в базе данных

```
def update_name(name:str, id:int):
    conn = get_connection()
    c = conn.cursor()
    sql = 'UPDATE users_data SET name = %s WHERE user_id =
%s'
    val = (name, id)
    c.execute(sql, val)
    c.close()
    conn.commit()
```

Удаление записей из таблиц, осуществляется с помощью оператора «DELETE». Оператор удаляет из таблицы строки, удовлетворяющие заданным в «where» условиям. Образец удаления записы из таблицы, приведен ниже в листниге 2.11.

Листинг 2.11 – Удаление пользователя из таблицы «users_data»

```
def delete_user(user_id: int):
    conn = get_connection()
    c = conn.cursor()
    sql = 'DELETE FROM users_data WHERE user_id = %s'
    val = user_id
    c.execute(sql,val)
    sql = 'DELETE FROM places_data WHERE user_id = %s'
    val = user_id
    c.execute(sql,val)
    c.close()
    conn.commit()
```

2.2 Технология разработки веб-приложения

На сегодняшний день существует немало способов настройки веб-приложения. Но все же, многие из них следуют одной и той же базовой структуре: клиент, сервер и база данных.

Клиент - это то, с чем взаимодействует пользователь. Исходя из этого, «клиентский» код отвечает за большую часть того, что на самом деле видит пользователь. Клиентский код включает в себя:

1) Определение структуры веб-страницы
2) Настройка внешнего вида веб-страницы
3) Реализация механизма реагирования на взаимодействие с пользователем, например, ввод текста, нажатие кнопки и т.п.

Макет и содержимое веб-страницы определяются HTML. Он позволяет описать базовую физическую структуру документа с помощью HTML-тегов. Каждый HTML-тег описывает определенный элемент документа.

Для определения внешнего вида веб-приложения, используется CSS, что означает каскадные таблицы стилей. Каскадные таблицы стилей (CSS) - это

язык таблиц стилей, который используется для описания представления документа, написанного в HTML или XML (включая XML-диалекты, такие как SVG, MathML или XHTML). CSS описывает, как элементы должны отображаться на экране, на бумаге, в речи или на других носителях.

Для улучшения анимации на странице веб-приложения, используется язык программирования JavaScript с подключением библиотеки JQueri. С помощью неё было реализован появление фиксированного заголовка. Реализован плавный прокрутки якорю, это метка, до которой мы осуществляем переход, с какой-либо ссылки в которой указан этот якорь, листинг 2.12.

Листинг 2.12 – Код, отвечающий за плавную прокрутку

```
$("[data-scroll]").on("click", function(event) {
        event.preventDefault();
        var $this = $(this),
            blockId = $this.data('scroll'),
            blockOffset = $(blockId).offset().top;
        $("#nav a").removeClass("active");
        $this.addClass("active");

        $("html, body").animate({
            scrollTop:  blockOffset
        }, 500);
    });
```

Для пользователей мобильных телефонов, был реализован меню «гамбургер», которой изображен на рисунке 2.13 и в листинге 2.13.

Листинг 2.13 - Код, отвечающий за меню в формате «гамбургер»

```
$("#nav_toggle".on("click", function(event) {
        event.preventDefault();

        $(this).toggleClass("active");
        $("#nav").toggleClass("active");
    });
```

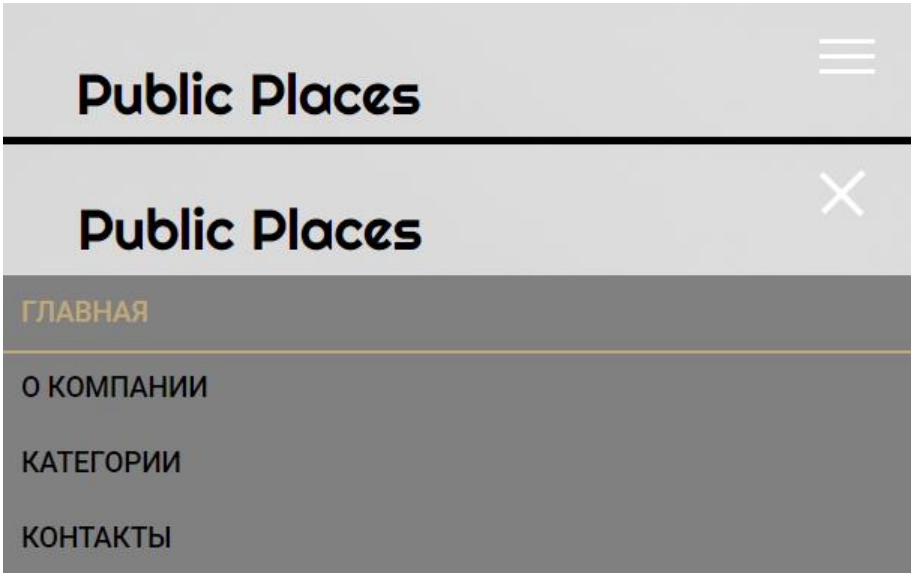

Рисунок 2.13 – Отображение меню, под названием «гамбургер»

Для того чтобы сэкономить место и отображать контент только по востребованию, было решено применять раскрывающиеся панели. Он позволяет определить HTML-элементы, которые нужно будет спрятать, и которые затем можно будет раскрыть, посредством простой функции обратного вызова, листинг 2.14.

Листинг 2.14 - Код, отвечающий за раскрывающиеся панели

```
$("[data-collapse]").on("click", function(event) {
    event.preventDefault();
    var $this = $(this),
        blockId = $this.data('collapse');

    $this.toggleClass("active");
});
```

Для удобства пользователей, добавлена интерактивная карта Яндекса в веб-приложение, чтобы показать на карте точное местоположение общественного места. Это было реализовано с помощью Яндекс карты, так как у них есть специальный сервис, который помогает сделать интерактивный фрагмент карты местности и добавить туда нужные обозначения. Отображение карты на странице веб-приложение, изображен на рисунке 2.14.

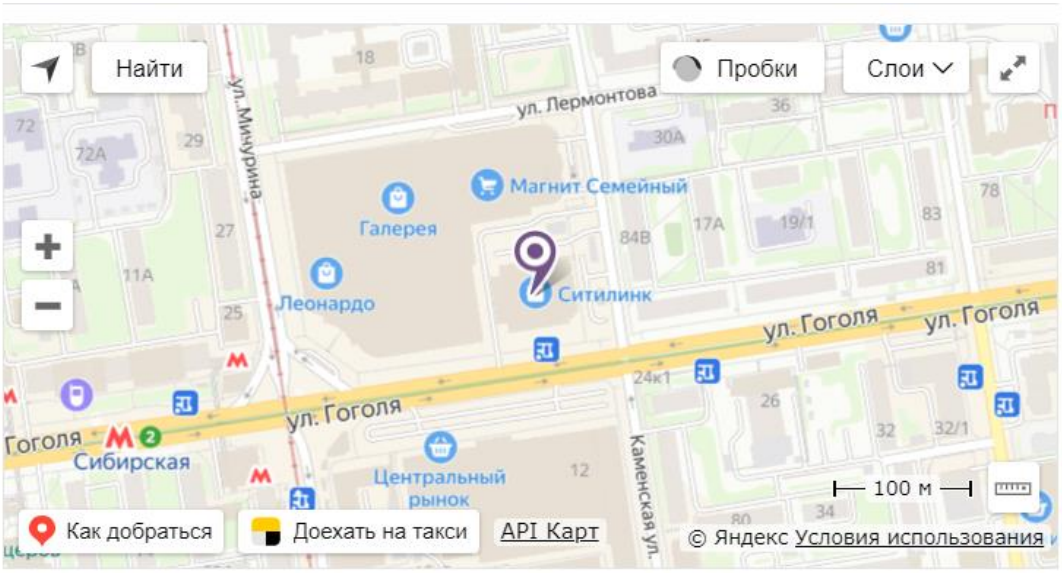

Рисунок 2.14 – Карта с отметкой местоположения общественного места.

С помощью языка программирования php, были полученные необходимые данные из базы. Специальные встроенные функции для работы с MySQL позволяют легко и эффективно работать с ней: выполнять любые запросы, считывать и записывать данные, обрабатывать ошибки.

Для работы с MySQL не понадобилось ничего дополнительно устанавливать или настраивать; все, что нужно, уже доступно в стандартном дистрибутиве PHP. Перед началом работы необходимо подключиться к базе данных. Чтобы подключиться к базе, необходимо использовать стандартную функцию PHP – mysqli_connect(). Функция возвращает результат — ресурс соединения. При этом, необходимо выполнить проверку, чтобы исключить ошибку при подключении. Для проверки возникновения ошибки с помощью mysqli_connect_error(), можно вернуть текстовое описание последней ошибки. Код подключения к базе приведен в листинге 2.15.

Листниг 2.15 – Код подключения к базе данных

```
    $link = mysqli_connect("a0679737.xsph.ru", "
a0679737_publicplaces ", "password", " a0679737_publicplaces ");
    if (mysqli_connect_errno())
    {
        printf("He удалось подключиться: %s\n",
mysqli_connect_error());
        exit();
    }
```

3 ПОЛЬЗОВАТЕЛЬСКИЙ ИНТЕРФЕЙС

3.1 Графический интерфейс telebram bot

При первом запуске бота, пользователя встречает описание бота, на верхнем углу, название. Всё это можно прочитать на рисунке 3.1.

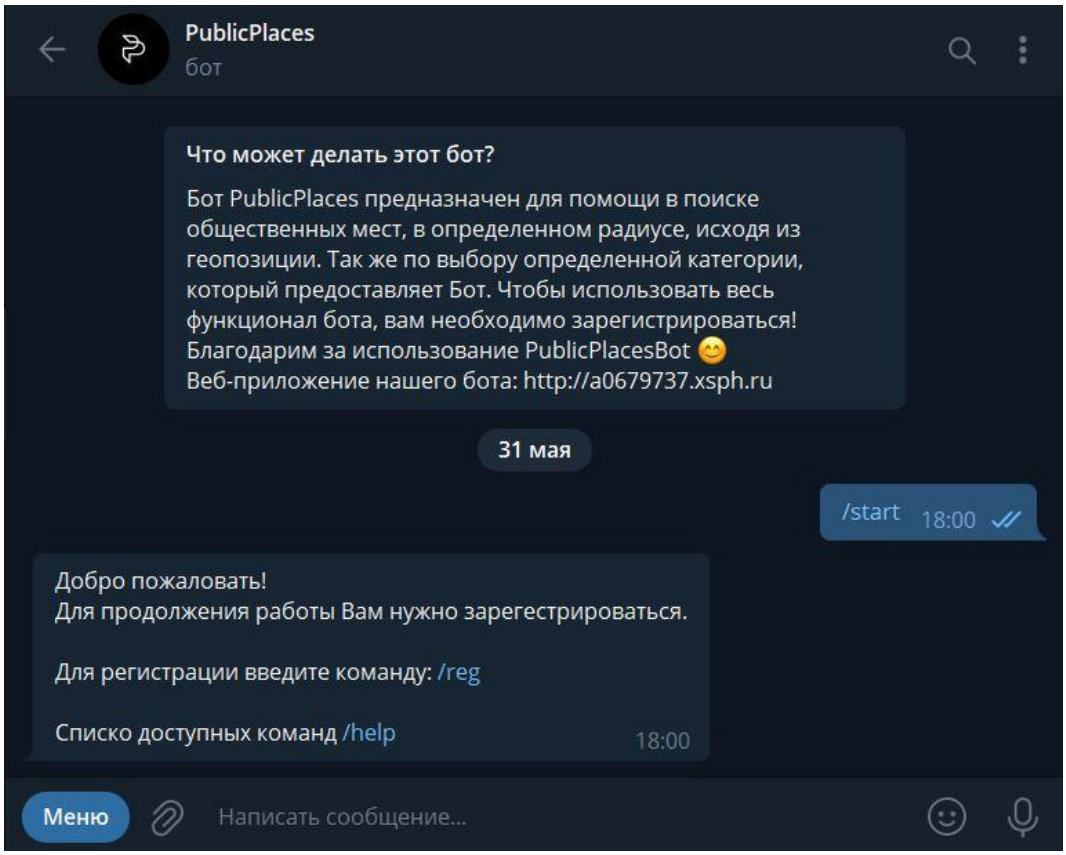

Рисунок 3.1 – Первый запуск бота

Далее мы нажимаем «/start», бот поприветствует пользователя и предложит ему зарегистрироваться.

Перед регистрацией, пользователю доступен ряд команд (см. рисунок 3.2)

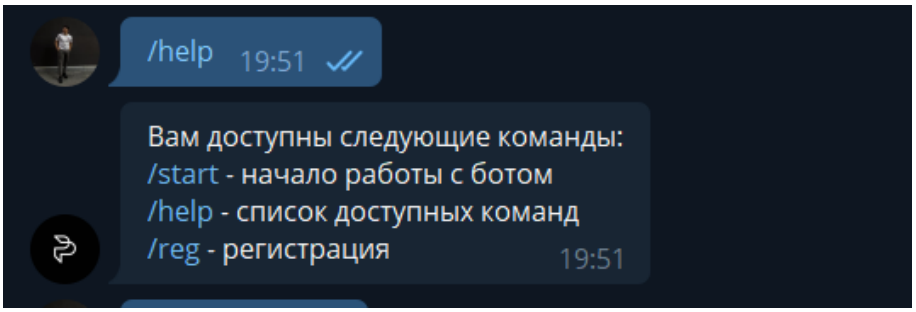

Рисунок 3.2 – Команды, доступные до регистрации

После ввода команды «/reg», запускается этап регистрации, который можно увидеть на рисунке 3.3.

При завершении регистрации, бот уведомляет пользователя об успешном сохранении данных.

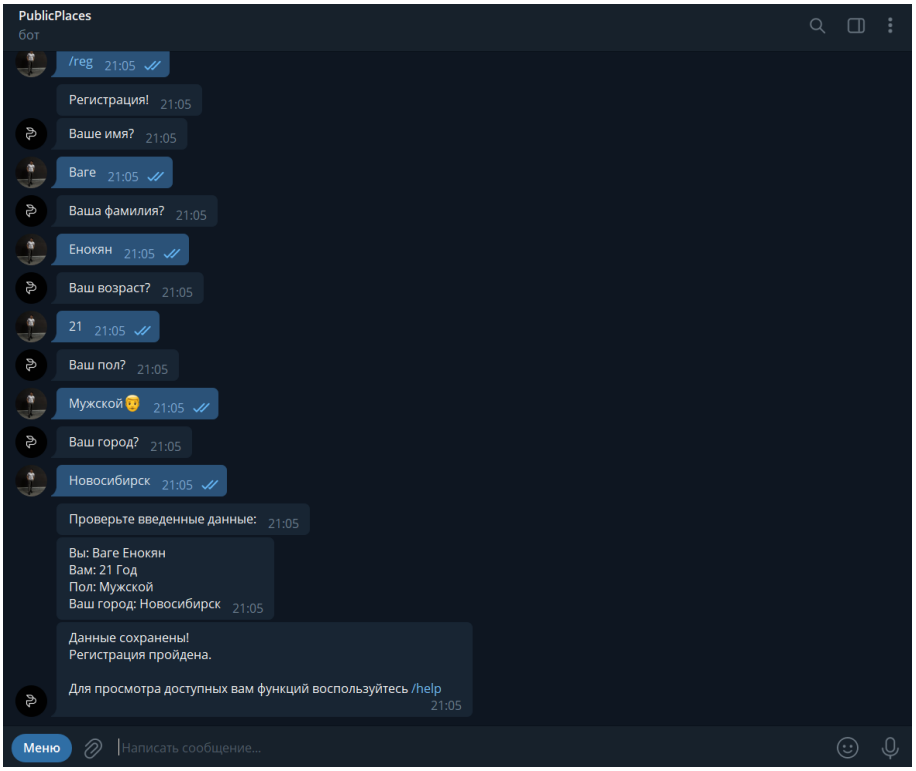

Рисунок 3.3 – Этапы регистрации пользователя

После регистрации, как мы видим на рисунке 3.4, пользователю открывается доступ ко всему функционалу бота.

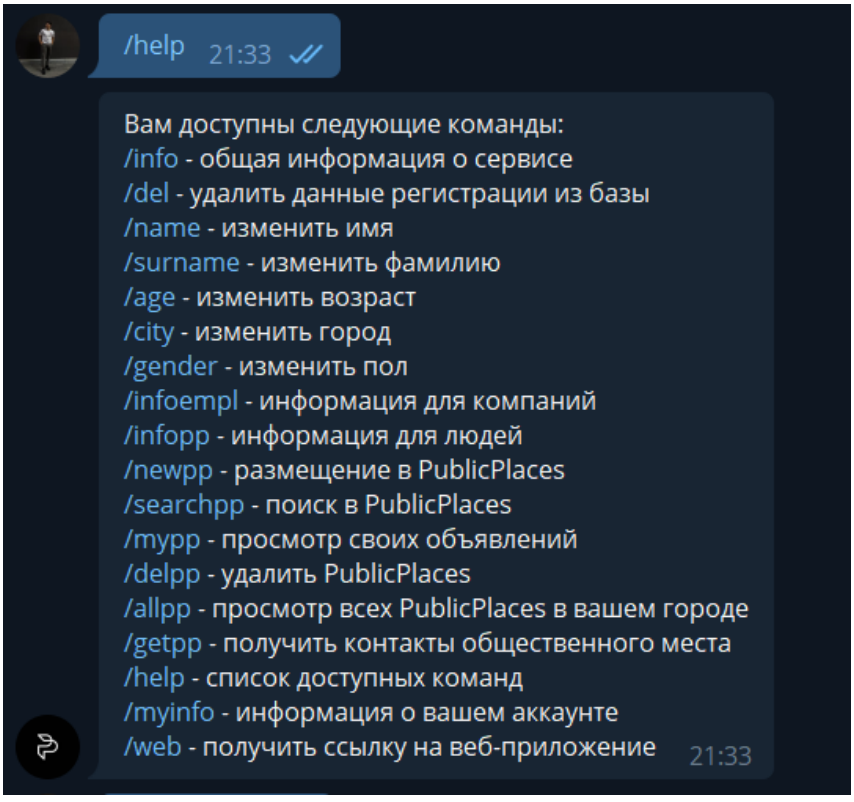

Рисунок 3.4 - Команды, доступные после регистрации

При вводе команды «/searchpp», пользовтель начинает поиск в базе, общестенных мест. Для начале бот предлагает пользователю выбрать категорию, после этого необходимо вести определенное местоположение, где он хочет начать поиск. После этого необходимо ввести радиус поиска. После вычисления необходимых данных, бот выдает варианты поиска. Удобство просмотра улучшает пагинация, которая выводит общественные места по отдельным страницам. Работу команды поиска изображены на рисунках 3.5, 3.6.

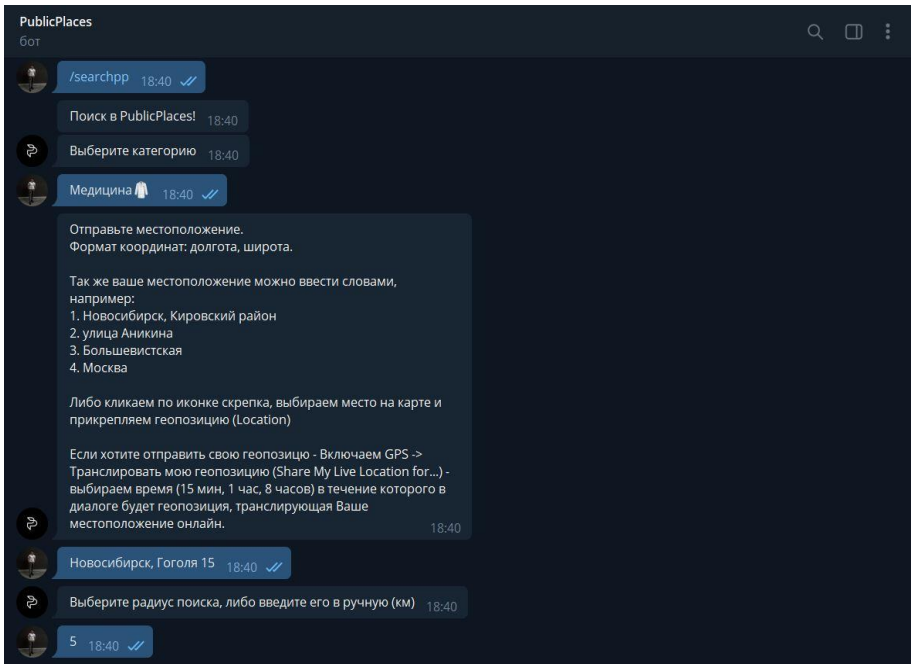

Рисунок 3.5 – Этап поиск общественных мест

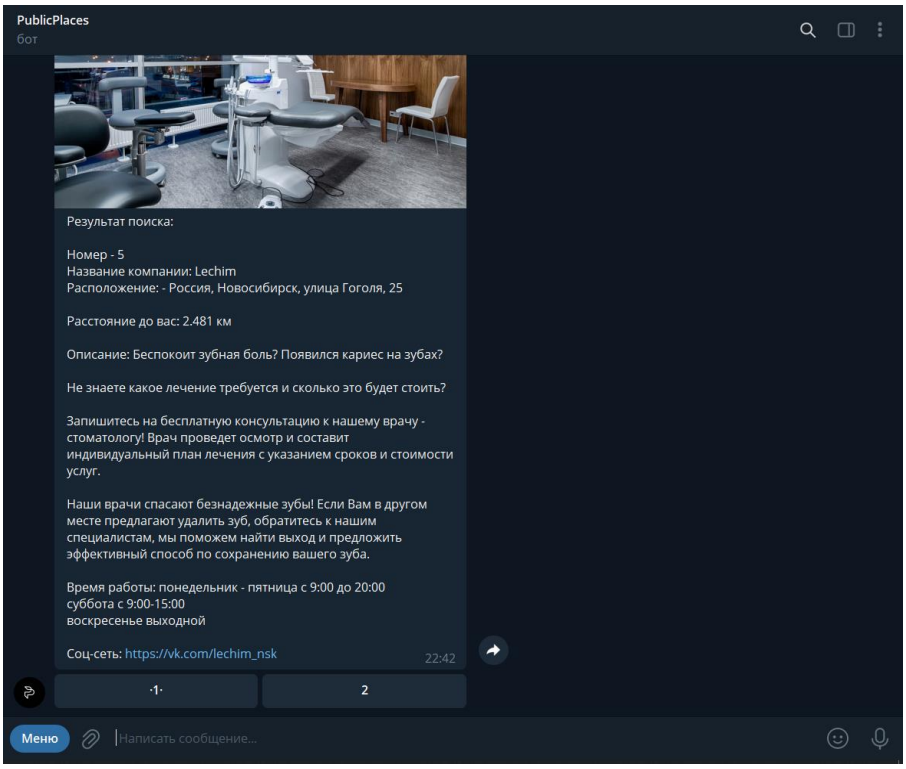

Рисунок 3.6 – Вывод поиска общественных мест

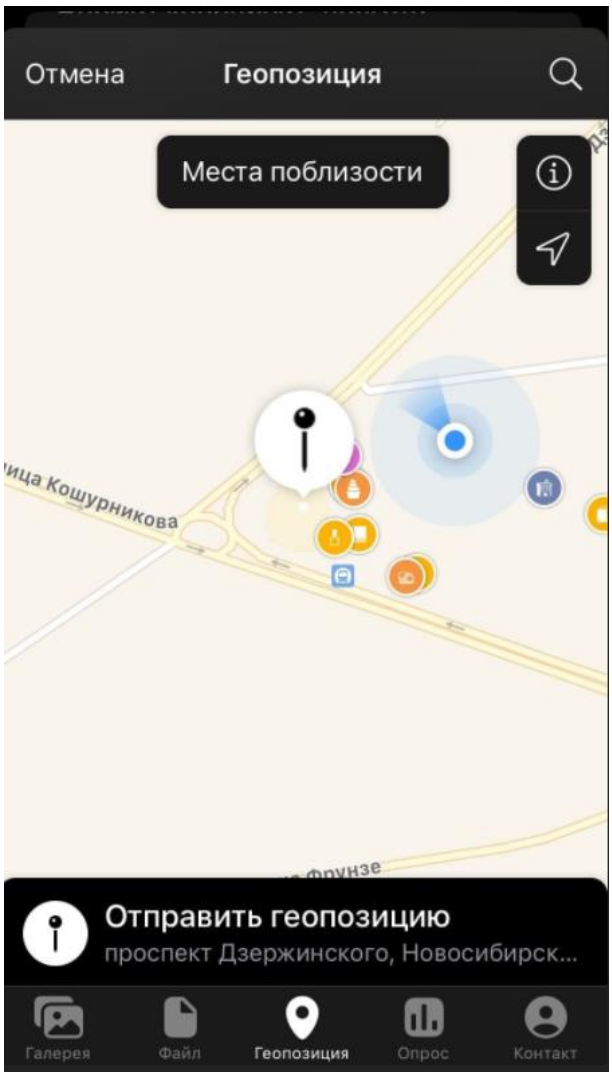

Рисунок 3.7 – Отправка местоположение, через мобильный телефон

На рисунке 3.7, изображен способ добавление местоположения через мобильный телефон. Для этого необходимо было нажать на значок скрепки в левом нижнем углу телефона и выбрать «Геопозиция».

Чтобы пользователю узнать номер телефона общественного места, необходимо ввести команду «/getpp». Как видно на рисунке 3.8, данная команда выводит номер телефона, имя и фамилию, и город.

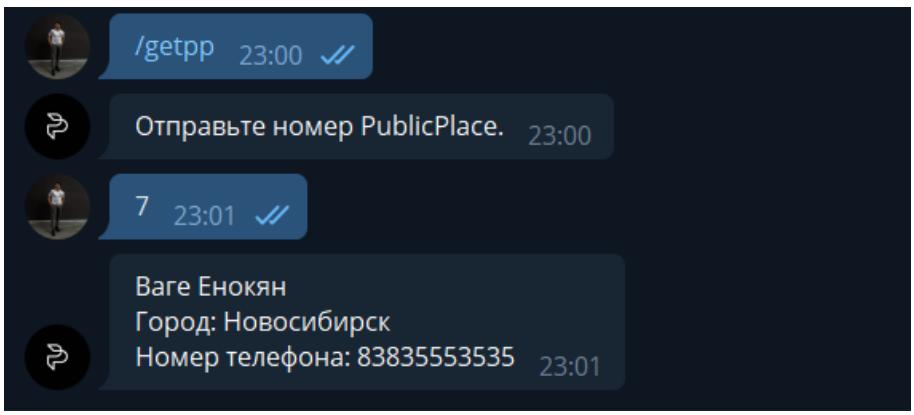

Рисунок 3.8– Вывод работы команды «/getpp»

Добавление общественного места, начинается с команды «/newpp». Первым делом бот предлагает ввести необходимые данные об общественном месте (см. рисунок 3.9 и 3.10).

Данные которые необходимо ввести:

– Название;
– Номер телефона;
– Местоположение;
– Описание;
– Время работы
– Фотография
– Социальная сеть

Как видно на рисунке 3.11, бот выводит заполненную запись и предлагает её опубликовать или отменить публикацию. После публикация, бот выдает готовую запись (см. рисунок 3.12).

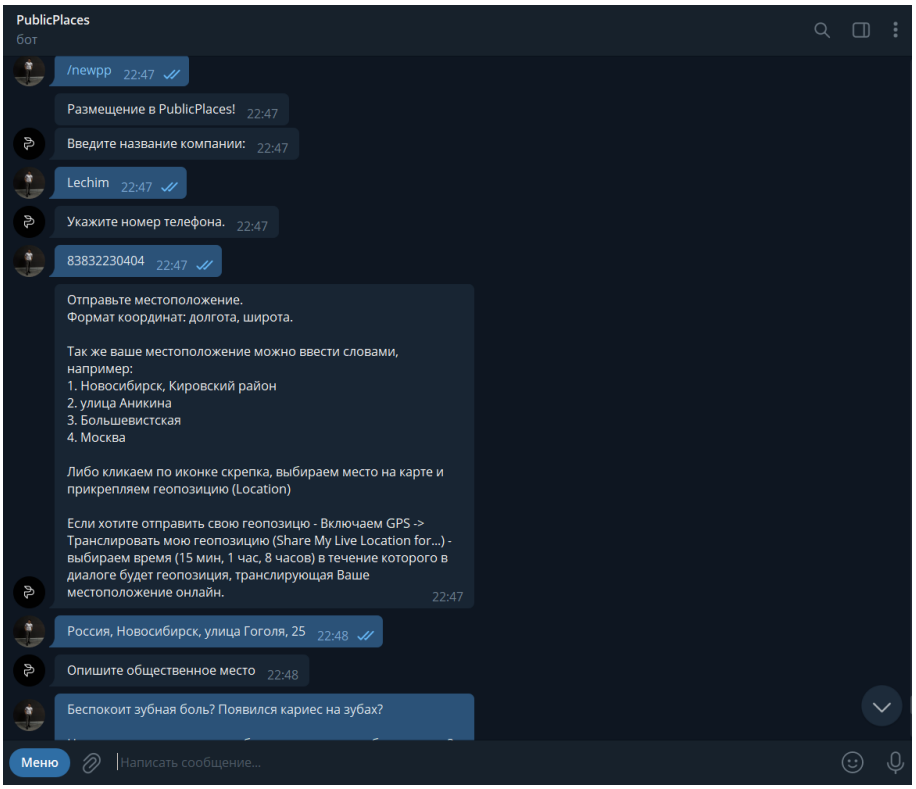

Рисунок 3.9 – Заполнение названия, номера телефона, координат, описания общественного места

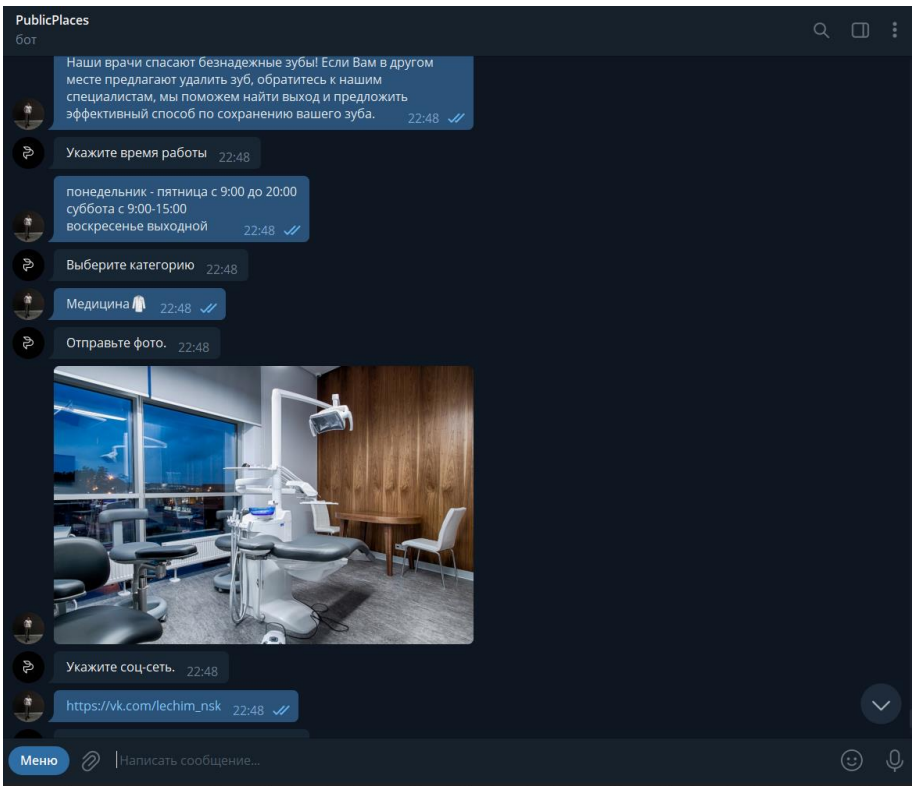

Рисунок 3.10 – Заполнение времени работы, выбора категории, добавление фотографии и ввода соц-сети.

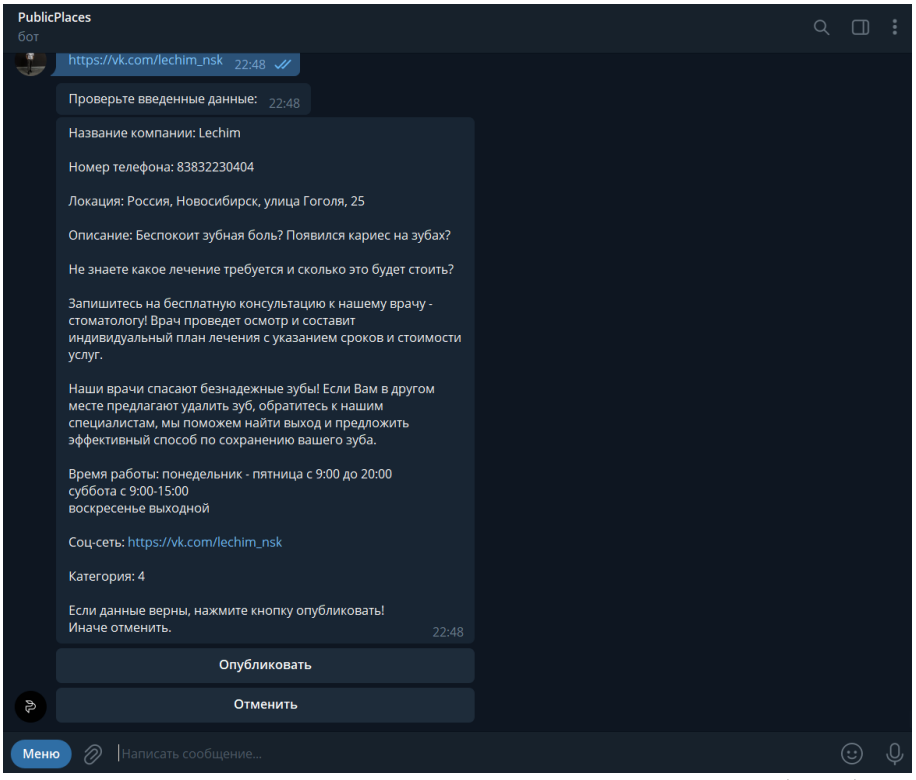

Рисунок 3.11 – Вывод записи, с предложение об публикации

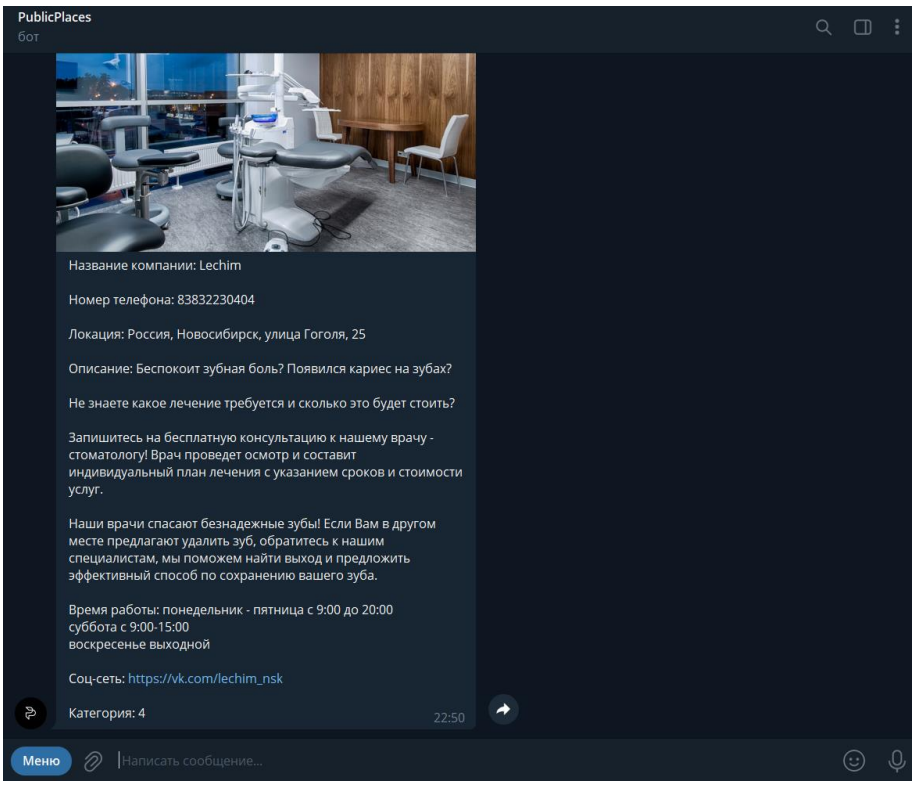

Рисунок 3.12 - Вывод опубликованной записи

На рисунке 3.13, отображается выполнение команды «/age». При желании, данную команду можно остановить, при нажатии на кнопку «Стоп».

Команды по типу – «name», «surname», «cit» , «gender» , имеют идентичный интерфейс.

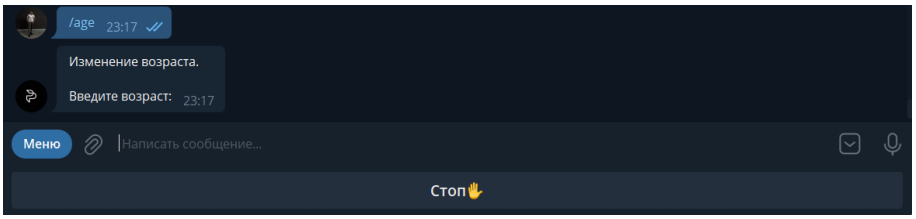

Рисунок 3.13 – Выполнение команды «/age»

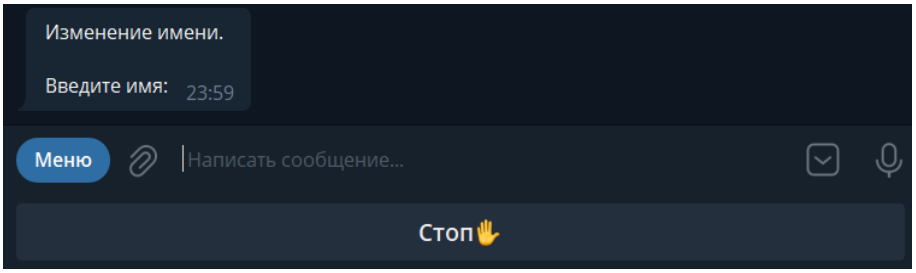

Рисунок 3.14 – Выполнение команды «/name»

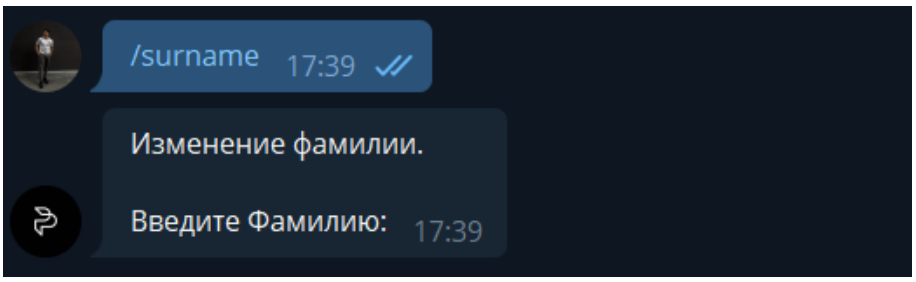

Рисунок 3.15 – Выполнение команды «/surname»

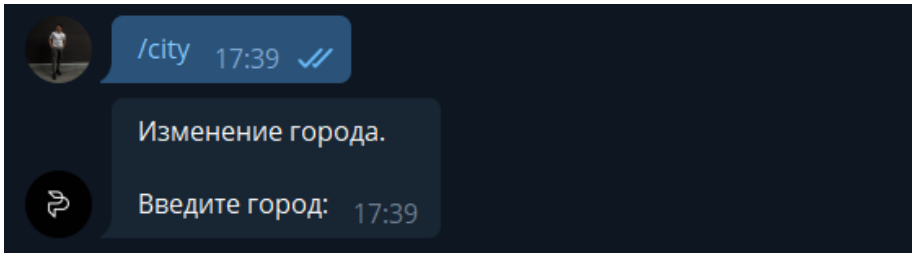

Рисунок 3.16 – Выполнение команды «/city»

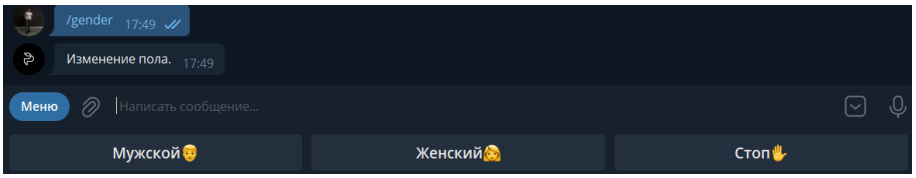

Рисунок 3.17 – Выполнение команды «/gender»

3.2 Графический интерфейс веб-приложения

Интерфейс страниц веб-приложения были изначально разработаны в виде макета и позже был использован при создании интерфейса веб-приложения.

Главная страница веб-приложения представляет собой приветственную страницу, содержит навигационные ссылки и показан на рисунке 3.18.

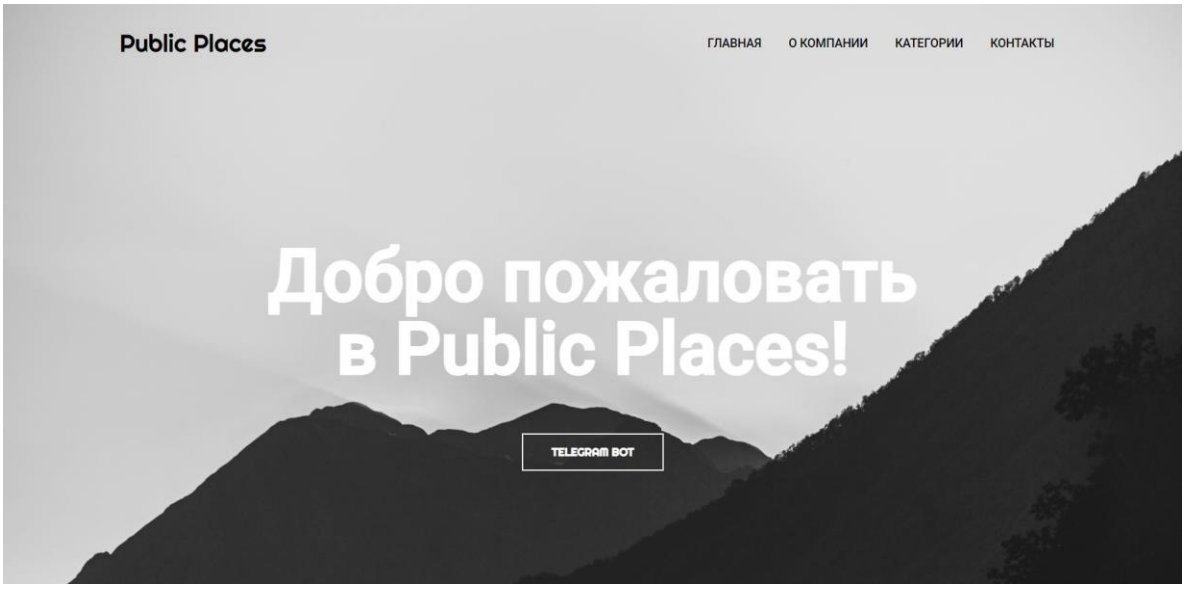

Рисунок 3.18 - Дизайн главной страницы веб-приложения.

Каждая навигационная ссылка является якорем, что позволяет быстро переместить на поле веб-страницы. Якорь в HTML – это закладка на поле веб-страницы. Отсюда можно совершать переход в telegtam бота благодаря кнопки «Telegtam Bot».

На веб-старнице «О компании» пользователь видит текстовую информацию о telegram боте, а также информации о каждой доступной категории. Для каждой категории в удобной форме представлена информация о количестве записей на текущий момент в базе данных. Статистика автоматически обновляются, при добавлении новой записи. Данная информация отображается на рисунке 3.19.

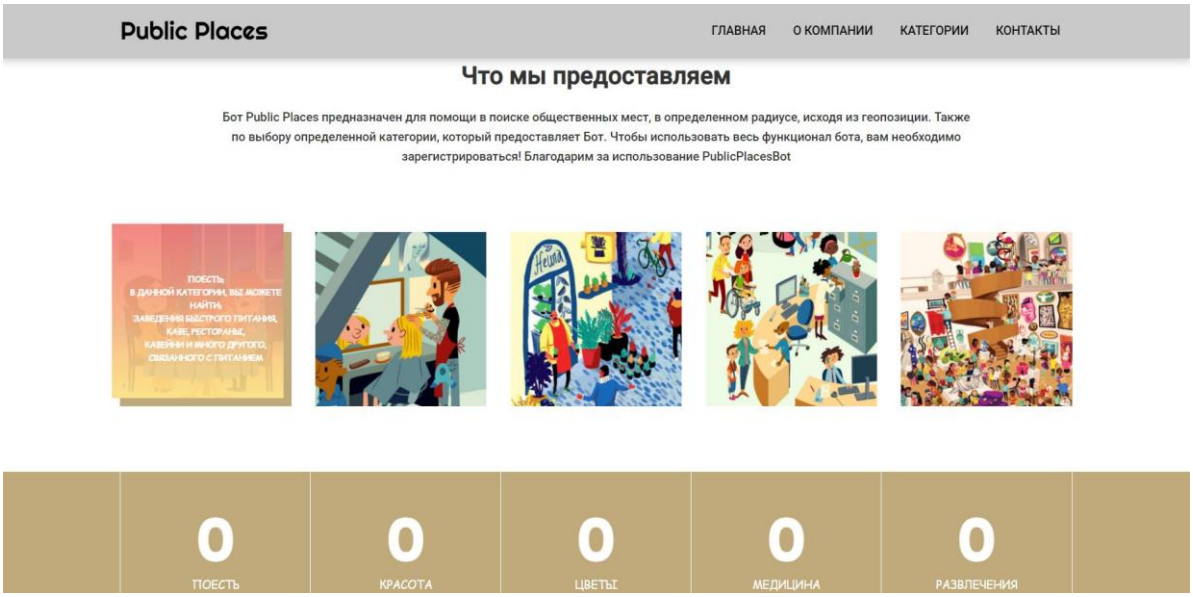

Рисунок 3.19 – Дизайн страницы «О компании» веб-приложения.

Когда пользователю необходимо прейти на страницу определенной категории, он должен на главном экране нажать на ссылку «Категории», рисунок 3.20. После этого, пользователь перейдет на веб-страницу и при выборе нажмёт кнопку «Перейти».

При нажатии, пользователь попадет на страницу категории, при выборе уже определенной записи, открывается новая страница подробного просмотра информации. На странице содержится следующая информация:

- Описание компании;
- Время работы;
- Местоположение;
- Фотография;
- Номер телефона;

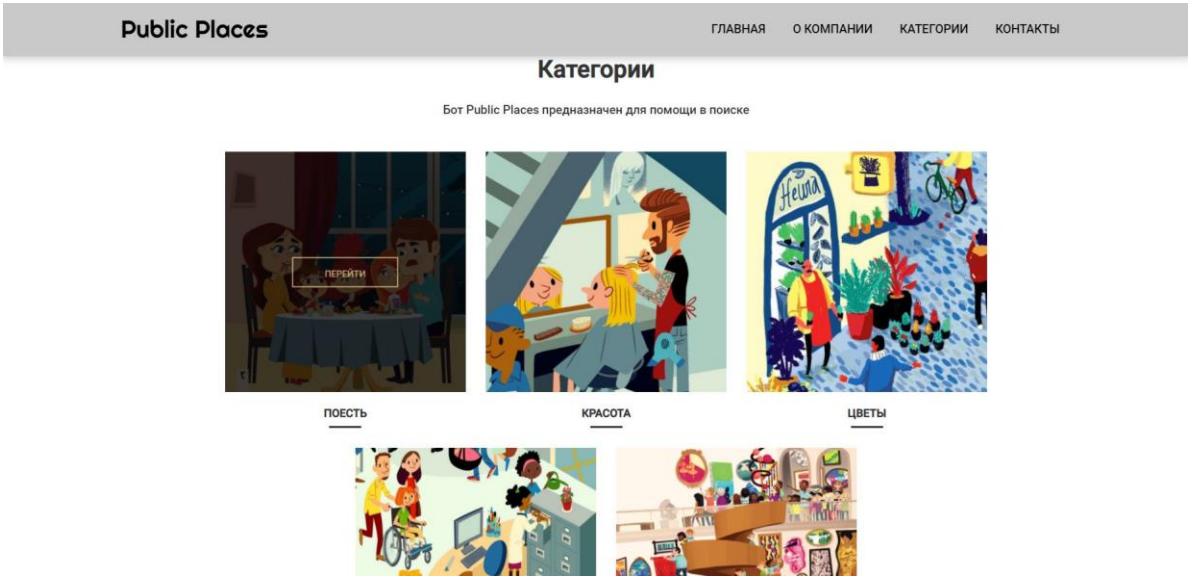

Рисунок 3.20 - Дизайн страницы «Категории» веб-приложения.

На веб-старнице «Контакты» отображается количество пользователей, информация и ссылка на telegram бота, рисунок 3.21.

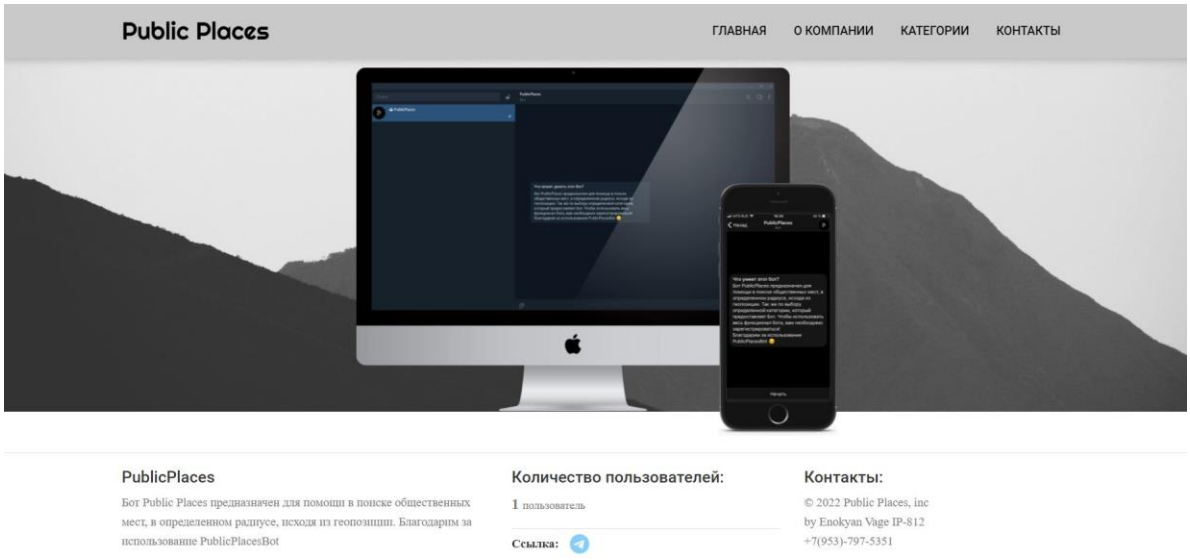

Рисунок 3.21 - Дизайн страницы «Контакты» веб-приложения

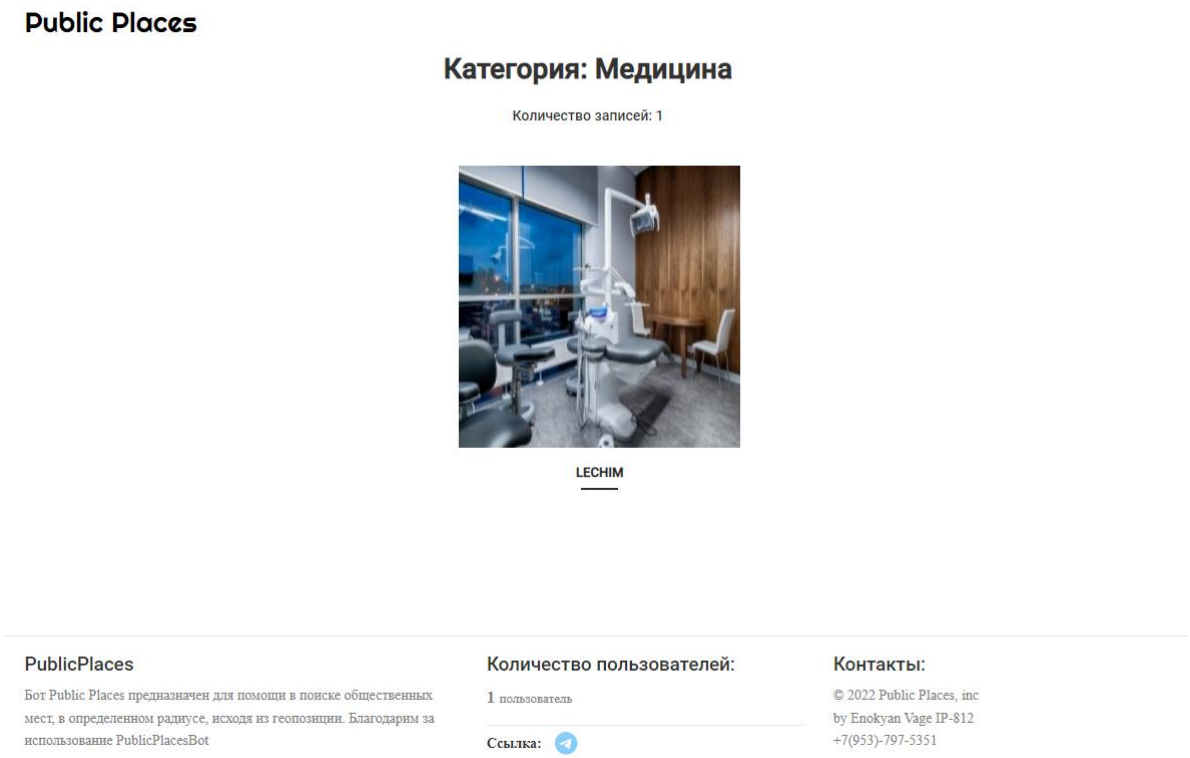

Рисунок 3.22 - Дизайн страницы категории

На веб-старнице категори отображается количество записей, фотографию общественного места и ссылку на её страницу, рисунок 3.22.

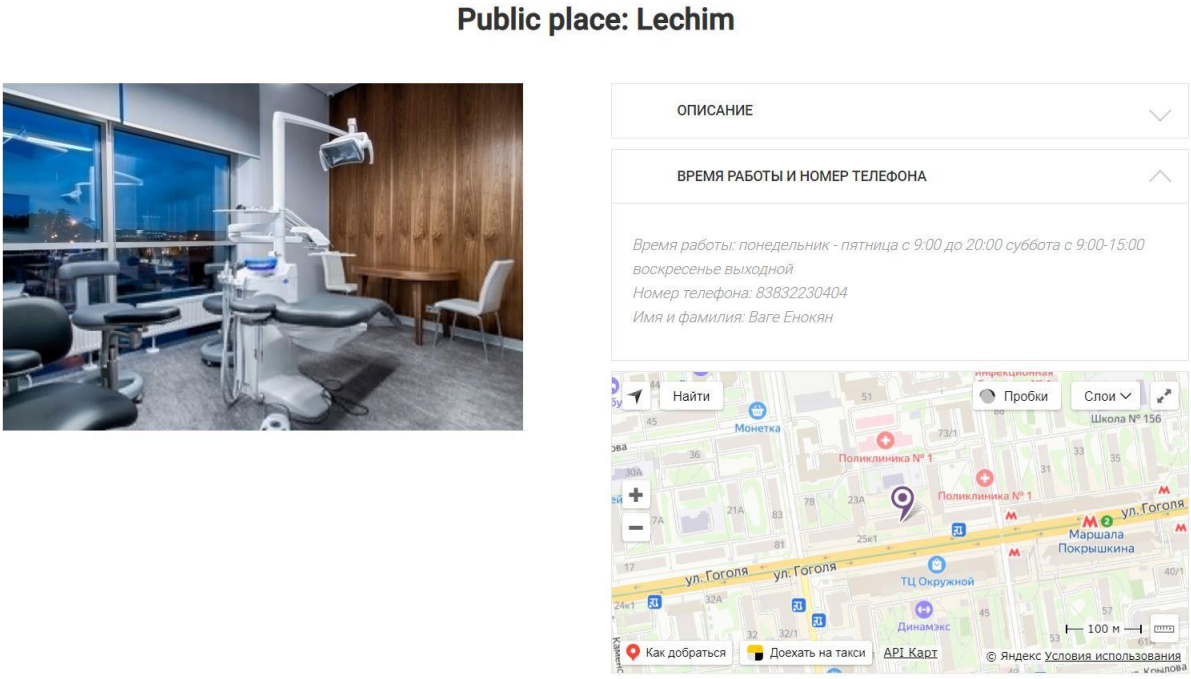

Рисунок 3.23 - Дизайн страницы общественного места

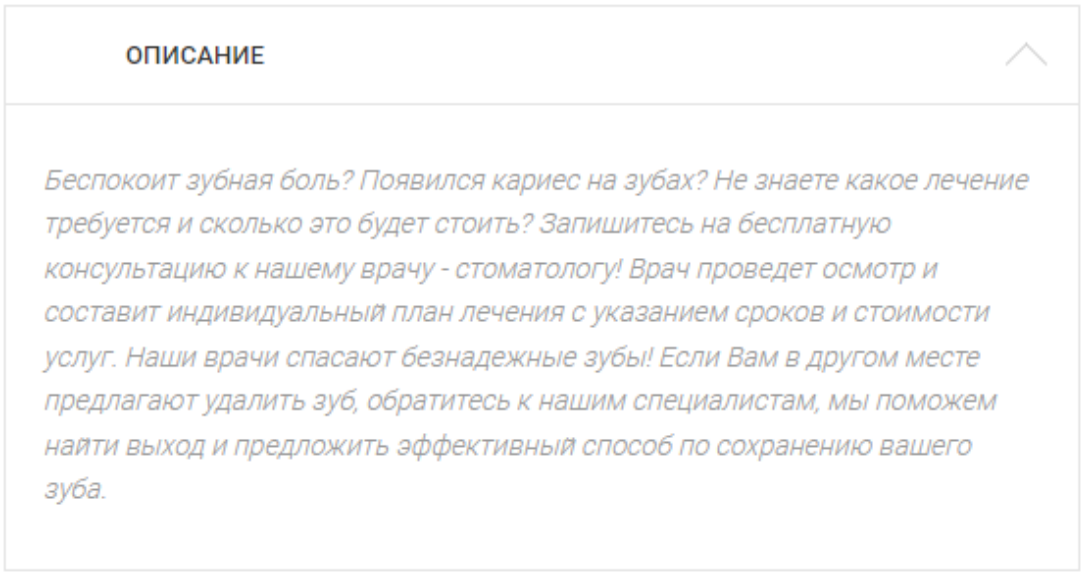

Рисунок 3.24 - Раскрывающиеся панель

На веб-старнице общественнго места отображается фотография общественного места на левой стране веб-страницы и на правой стороне вся необходимая информация (см. рисунок 3.23). Также присутствует карта, на которой помечена местоположение. При желании прочитать описание, необходимо нажать на раскрывающиеся панель (см. рисунок 3.24).

ЗАКЛЮЧЕНИЕ

Целью бакалаврской работы являлась разработка telegram бота, для осуществления добавления, поиска и просмотра общественных мест. А также разработка веб-приложения, отображающее все записи из базы и просмотра каждой по отдельности.

В результате разработки были получены множество практических навыков по созданию telegram бота, а также веб-приложения и работе с базой данных. Были изучены средства для реализации проекта. Улучшены навыки работы, с такими языками программирования как python, php. Разработан алгоритм реализации проекта. Была проделана работа с Yandex API для получения координат, а также вывода на карте местоположения.

Разработанный бот предоставляет пользователям возможность добавлять общественные места в базу. С возможностью добавления фотографии, описания, геолокации, времени работы, выбора категории, а также социальной сети. Осуществлять поиск, с возможностью выбора местоположения и категории, что позволяет уменьшить радиус поиска.

Веб-приложения позволяет просматривать все общественные места. На каждой странице отображается вся необходимая информация: описание, время работы, номер телефона. Для удобного просмотра местоположения, добавлена карта с указателем.

Для того, чтобы начать пользоваться ботом, достаточно иметь аккаунт в Telegram. Все данные пользователей хранятся на удаленном сервере.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. pyTelegramBotAPI [Электронный ресурс]. URL https://pypi.org/project/pyTelegramBotAPI/
2. pyTelegramBotAPI [Электронный ресурс]. URL https://github.com/eternnoir/pyTelegramBotAPI
3. Geopy [Электронный ресурс] URL:https://geopy.readthedocs.io/en/stable/
4. PyMySql [Электронный ресурс]. URL:https://pymysql.readthedocs.io/en/latest/ python-telegram-bot-pagination.
5. python-telegram-bot-pagination [Электронный ресурс]. URL: https://github.com/ksinn/python-telegram-bot-pagination
6. Мейкшан В.И. Основы языка SQL в примерах и задачах: Учебно-методическое пособие / Сибирский государственный университет телекоммуникаций и информатики; Кафедра телекоммуникационных сетей и вычислительных средств. – Новосибирск, 2013. – 25 с. – 40 с.
7. PHP: Documentation [Электронный ресурс]. URL: https://www.php.net/docs.php
8. PHP Tutorial [Электронный ресурс]. URL: https://www.w3schools.com/php/default.asp
9. CSS [Электронный ресурс]. URL: https://developer.mozilla.org/ru/docs/Web/CSS