МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«ТЮМЕНСКИЙ индустриальный университет»

Высшая школа цифровых технологий

Кафедра Математики и прикладных информационных технологий

КУРСОВОЙ ПРОЕКТ

Дисциплина: «Интеллектуальные системы и их проектирование»

Тема: «Телеграм-бот по поиску мест отдыха в городе Тюмень»

Выполнил:

Студент 4 курса группы РИСб-20-1

направления 45.03.04 «Интеллектуальные

системы в гуманитарной сфере»

Лисейцев А.Д.

Руководитель:

Доцент, канд. пед. наук

Спирин И.С.

Тюмень, 2023 г.

**АННОТАЦИЯ**

Целью данной курсовой работы является создание Telegram-bot.

Объектом работы является Telegram-bot с пасрером данных.

Предметом курсовой работы является процесс разработки Телеграм бота на Python.

Информационная база исследования: электронные ресурсы, статьи, учебные пособия.

Результат: разработан бот для Telegram, который предоставляет информацию о местах отдыха в городе Тюмени, основываясь на данных, полученных путем парсинга веб-сайтов.

Курсовая работа состоит из введения, трёх глав, заключения и списка использованных источников.

**ТЕХНИЧЕСКОЕ ЗАДАНИЕ**

Для реализации курсового проекта необходимо решить ряд задач:

− анализ и описание предметной области;

− определение предметной области и темы курсового проекта;

− оформление технического задания и его утверждение;

− реализация чат-бота с функцией парсера;

− оформление курсового проекта и документации проекта;

− предоставление курсовой проекта и документации проекта.

СОДЕРЖАНИЕ

ВВЕДЕНИЕ 5

1 Теоретическая часть про Telegram и Парсер 7

1.1 Ознакомление с Telegram 7

1.2 Каналы и боты Telegram 8

1.3 Определение понятий "парсинг", "веб-скрапинг", "веб-краулинг" и их взаимосвязь 10

1.4 Работа Парсера 11

2 Аналитический аспект разработки Telegram-bot с парсером 14

2.1 Выбор ПО, используемого в разработке 14

3 Практическая часть 15

3.1 Главный блок кода 15

3.2 Код парсера информации сайта 16

3.3 Код телеграмм бота 20

ЗАКЛЮЧЕНИЕ 26

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 27

# ВВЕДЕНИЕ

Парсинг сайтов - процесс автоматизированного сбора и анализа данных с веб-страниц, используя специальные программные инструменты или библиотеки. Он позволяет получать актуальные и структурированные данные для анализа в различных сферах, таких как маркетинг, бизнес-аналитика, исследования рынка и прогнозирование трендов. Также может быть использован для создания базы данных или архива веб-страниц и мониторинга информации в интернете.

В современном обществе высокая популярность мессенджеров и чат-ботов, которые упрощают ежедневные рутинные задачи, делает актуальным использование парсинга. Однако, его использование может быть ограничено законом в зависимости от страны и типа информации, которую собирает парсер. Некоторые сайты могут иметь механизмы защиты от парсинга, такие как CAPTCHA или блокировки IP-адресов. Поэтому необходимо соблюдать правовые и этические ограничения при использовании парсинга. Рассмотрим существующие подходы к созданию чат-ботов для образовательных учреждений, разработаем концепцию чат-бота для поддержки абитуриентов в процессе поступления, и изучим потенциальные выгоды и сложности, связанные с его внедрением.

Решение: создать чат-бота на доступном каждому мессенджере Telegram, который будет в удобном виде предоставлять варианты мест отдыха.

Актуальность – телеграмм боты по подбору мест могут быть полезны для пользователей, желающих найти интересные места для отдыха и развлечений. Такие боты могут предоставить информацию о различных местах, таких как рестораны, бары, клубы, кинотеатры, музеи, парки развлечений и друге места.

Степень разработки: в интернете имеются подобные телеграмм-боты, но по нашей теме отсутствуют.

Цель – разработать бота для Telegram, который будет предоставлять информацию о местах отдыха в городе Тюмени, основываясь на данных, полученных путем парсинга веб-сайтов.

Объект данного курсового проекта – чат-бот с функцией парсинга данных на мессенджере «Telegram».

Предмет – процесс разработки чат-бот с функцией парсинга в мессенджере «Telegram».

Для достижения поставленной цели необходимо решить следующие задачи:

* анализ выбранной предметной области;
* выбор технологий и среды разработки;
* собрать информацию по поводу парсеров;
* проанализировать возможности и ограничения парсеров;
* изучить процесс создания парсеров;
* разработка чат-бота на платформе Telegram.

# Теоретическая часть про Telegram и Парсер

## Ознакомление с Telegram

Telegram - это мессенджер, который был создан братьями Павлом и Николаем Дуровыми. Он представляет собой платформу для обмена сообщениями и файлами, обладающую рядом особенностей и преимуществ:

* защита данных – Telegram известен своим усиленным шифрованием, которое обеспечивает конфиденциальность переписки и передаваемых файлов;
* многофункциональность – помимо обмена текстовыми сообщениями, Telegram позволяет отправлять голосовые сообщения, видео, файлы различных форматов, а также создавать групповые чаты и каналы;
* боты и стикеры – в Telegram есть возможность создания и использования ботов для автоматизации различных задач, а также обширная библиотека стикеров для выражения эмоций;
* кроссплатформенность – приложение Telegram доступно на различных устройствах и платформах, что обеспечивает удобство использования для пользователей с разными предпочтениями.
* самоуничтожающиеся сообщения – в Telegram можно отправлять сообщения с установленным временем самоуничтожения, что обеспечивает дополнительный уровень конфиденциальности.
* открытость и развитие – Telegram является открытым исходным кодом, что позволяет разработчикам создавать собственные приложения и расширения для платформы.
* безлимитный облачный хранитель – в Telegram доступен облачный хранитель, который позволяет пользователям хранить файлы и медиаконтент без ограничений по объему.
* групповые видеозвонки – Telegram предоставляет возможность проведения групповых видеозвонков, что делает его удобным инструментом для коммуникации в больших группах.

Telegram продолжает развиваться и предлагать новые функции, сохраняя при этом фокус на безопасности и конфиденциальности пользователей.

## Каналы и боты Telegram

Каналы и боты Telegram предлагают ряд преимуществ и широко используются для различных целей. Вот некоторые из ключевых преимуществ и вариантов использования:

* взаимодействие и поддержка клиентов. Ботов Telegram можно использовать для оказания поддержки клиентам, ответов на запросы и взаимодействия с клиентами. Они более экономичны и эффективны, чем агенты-люди, поскольку могут участвовать в сложных разговорах и оказывать поддержку 24 часа в сутки, 7 дней в неделю, что приводит к повышению удовлетворенности клиентов и повышению эффективности.
* продвижение бизнеса и маркетинг. Каналы и боты используются для продвижения бизнеса, маркетинга и привлечения потенциальных клиентов. Их можно использовать для рассылки информационных бюллетеней, демонстрации портфолио и продвижения продуктов и услуг среди широкой аудитории. Ботов также можно использовать для сбора потенциальных клиентов и взаимодействия с потенциальными клиентами, способствуя продажам и маркетингу.
* доставка информации и автоматизация. Боты Telegram могут доставлять информацию из различных доменов и выполнять команды в чате Telegram, запуская действия или запрашивая информацию. Их можно использовать для автоматизации задач, доставки обновлений в режиме реального времени и предоставления ценной информации, что делает их полезным дополнением к интеллектуальному общению, особенно в профессиональных областях.
* распространение контента и коммуникация: каналы используются для распространения контента среди широкой аудитории, и они использовались журналистами в репрессивных режимах для создания независимых новостных сетей. Боты и каналы обеспечивают общение между людьми и могут использоваться для обмена историями, создания больших публичных групп и проведения частных и групповых голосовых или видеозвонков, а также публичных прямых трансляций.

Популярные телеграм-боты включают в себя разнообразные категории, такие как боты для поиска людей, боты для знакомств, боты для автоматизации административных задач в каналах и группах, боты для обработки фото и видео, боты для игр, а также боты для повседневных задач, такие как напоминания, конвертация видео в GIF и многое другое. Некоторые из популярных ботов включают в себя Usersbox, Quick OSINT, ControllerBot, RusDate, Ls, Video to GIF Converter, Akinator, Emojinate, и многие другие.

Каналы Telegram — это функция приложения обмена сообщениями Telegram, которая позволяет пользователям транслировать сообщения большой аудитории. Каналы могут создаваться любым пользователем и иметь неограниченное количество подписчиков. Их часто используют в новостных, развлекательных и образовательных целях. Каналы Telegram могут быть публичными или частными, и пользователи могут присоединиться к ним или покинуть их в любое время. Каналы также можно использовать для обмена мультимедийным контентом, например, фотографиями, видео и аудиофайлами. Каналы Telegram использовались журналистами репрессивных режимов для создания независимых новостных сетей. Каналы также используются для продвижения бизнеса, маркетинга и привлечения потенциальных клиентов.

В целом каналы и боты Telegram являются ценными инструментами для предприятий, организаций и частных лиц, предлагая широкий спектр преимуществ, таких как улучшение взаимодействия с клиентами, эффективная автоматизация и эффективное распространение контента. Они стали неотъемлемой частью экосистемы Telegram, способствуя ее универсальности и широкому распространению.

## Определение понятий "парсинг", "веб-скрапинг", "веб-краулинг" и их взаимосвязь

Парсинг, веб-скрапинг и веб-краулинг – это термины, фактически связанные с извлечением данных из веб-ресурсов, но с некоторыми существенными отличиями в подходе и целях.

Парсинг является процессом синтаксического анализа текста или данных, выполняемым с помощью компьютерной программы – парсера. Целью парсинга является распознавание структуры данных и представление ее в удобной форме для последующей обработки или анализа. В контексте веб-разработки парсинг обычно относится к извлечению информации из HTML-кода веб-страниц.

Веб-скрапинг – это процесс автоматического извлечения данных или информации с веб-страниц, используя специализированные программы или боты. Суть веб-скрапинга заключается в обращении к веб-странице, анализе ее содержимого и извлечении нужной информации. Веб-скрапинг может быть использован для различных целей, таких как анализ конкурентов, сбор данных для исследований, мониторинг цен и других.

Веб-краулинг, также известный как веб-индексация, – это процесс автоматического обхода интернета перебором ссылок и сбором информации о веб-страницах. Основная цель веб-краулинга заключается в создании поисковых индексов для поисковых систем. Краулеры, также называемые ботами или пауками, обходят веб-страницы, анализируют их содержимое, извлекают ссылки на другие страницы, затем рекурсивно следуют этим ссылкам и повторяют процесс снова. Веб-краулинг широко применяется для составления крупных поисковых систем, а также для создания архивов веб-страниц.

Таким образом, парсинг, веб-скрапинг и веб-краулинг представляют собой различные аспекты работы с веб-ресурсами и извлечения информации из них. Парсинг относится к процессу анализа и разбора структуры данных, веб-скрапинг – к автоматическому извлечению данных с веб-страниц, а веб-краулинг – к обходу и индексации веб-ресурсов в целях создания поисковых индексов. В то же время, эти термины взаимосвязаны, так как часто используются вместе для решения задач, связанных с обработкой информации из веб-ресурсов.

## Работа Парсера

На сайте имеется обширное количество информации, однако вам нужно только несколько существенных деталей. Эта информация представлена в виде графиков, PDF-файлов и баз данных. Задача программного обеспечения для Web-парсинга заключается в преобразовании этих неструктурированных данных в более понятные и удобочитаемые формы, которые могут быть использованы писателями и аналитиками. Обычно данные структурируются в виде HTML-файлов с помощью тегов, которые затем отображаются в Web-браузерах. Затем эти теги интерпретируются инструментами Web-парсинга, которые собирают информацию, содержащуюся в этих тегах.

Теоретически Web-парсинг может показаться удобным, однако на практике его выполнение для новичка может быть непростой задачей. Человек, желающий заниматься Web-парсингом, должен обладать знаниями программирования, поскольку иногда для сбора необходимых данных требуется написание кода. Поэтому для успешного Web-парсинга необходимо быть опытным программистом или нанимать специалистов для сбора данных. Однако существуют простые инструменты Web-парсинга, которые помогают справиться с простыми проектами.

Как работает инструмент Web-парсинга? Пользователь указывает URL-адрес инструмента, который извлекает все данные с Web-сайта, позволяя самостоятельно извлечь информацию с любого сайта. Для работы с Web-сайтами пользователю не требуется знание программирования. Инструмент Web-парсинга нужен для извлечения информации с сайтов, которая может быть полезна в научных исследованиях. Некоторые менеджеры социальных сетей желают собирать сообщения и комментарии с Web-сайтов социальных сетей. Для этого необходимо выбрать URL-адрес для инструмента Web-парсинга, который выберет посты и комментарии и извлечет их. Эти инструменты позволяют менеджерам социальных сетей сэкономить много времени и усилий.

Часть информации на сайтах организована в виде диаграмм, а другая часть не имеет структуры и представлена в виде PDF-файлов. Многие тексты на сайтах структурированы с использованием тегов XHTML или HTML, которые дают инструкции браузерам для их правильного отображения. Эти теги определены таким образом, чтобы сделать текст более читабельным на сайте. Инструменты Web-парсинга определяют эти теги и работают над тем, как извлечь содержащуюся в них информацию. Процесс работы начинается с отправки браузеру запроса GET, на что мы получаем ответ в виде содержимого сайта.

На протяжении исследований особую роль играют время и усилия, оказывающие значительное влияние на итоговый результат. Например, при анализе специфического PDF-файла, включающего около 800 страниц информации, необходима только информация, относящаяся к определенной теме. Для этой цели лучше всего использовать инструменты парсинга, которые способны извлекать только необходимые для исследования данные, связанные с интересующей нас темой. Web-парсинг может быть очень полезным при сборе данных.

Однако, следует помнить, что не все заинтересованы в получении полезной информации. Существуют некоторые лица, которые нарушают законодательство путем копирования и использования содержимого других Web-сайтов без согласия их владельцев. Парсеры не должны применять такие методы, поскольку это преимущественно выгодно для парсеров, но не для владельцев Web-сайта. Кроме того, такое поведение является недопустимым с этической точки зрения.

Поскольку Web-парсинг является эффективным и не требует больших затрат времени, мы полюбили его. Он позволяет сэкономить время парсеров при изучении больших объемов информации, а также обходится дешевле. Тем не менее, мы всегда должны использовать этот инструмент с добрыми намерениями и никоим образом наносить ущерб владельцам Web-сайтов или их содержимому.

# Аналитический аспект разработки Telegram-bot с парсером

## Выбор ПО, используемого в разработке

Python - это язык программирования общего назначения, завоевавший популярность благодаря своей простоте и универсальности. Впервые он был представлен в 1991 году, что делает его старше таких языков, как Java, C#, JavaScript и PHP, но моложе Cсамого большого сообщество из трех вариантов.

Python обладает простым синтаксисом, что облегчает его изучение для начинающих и обеспечивает приятный опыт программирования. Он широко используется в веб-разработке, анализе данных, машинном обучении, искусственном интеллекте и автоматизации.

Python поддерживает Юникод и имеет упрощенный механизм итерации по спискам. Он был разработан с акцентом на читабельность и простоту, что делает его отличным выбором для новичков в программировании.

Универсальность Python проявляется в его использовании в различных приложениях - от серверной части Instagram до Google, Spotify, Netflix, Uber, Dropbox, Pinterest, Reddit и многих других популярных платформ. Он также используется в веб-скреппинге, разработке настольных приложений и научных вычислениях.

Сообщество Python активно и активно поддерживает пользователей, предоставляя множество ресурсов для обучения и развития.

Однако Python не лишен недостатков. Он, как правило, медленнее некоторых других языков программирования, что может быть недостатком в областях, где производительность имеет решающее значение, таких как разработка игр или высокопроизводительные вычисления.

Несмотря на это, Python остается популярным и востребованным языком благодаря своей универсальности, простоте и обширной экосистеме библиотек и инструментов.

# Практическая часть

## Главный блок кода



Рисунок 3.1 – Код из блока main.py

Этот блок кода устанавливает основную точку входа для бота Telegram, используя библиотеку python-telegram-bot. Вот разбивка кода:

* load\_dotenv(): Эта функция загружает переменные окружения из файла .env;
* token = os.getenv('BOT\_TOKEN'): Эта строка извлекает токен бота Telegram из переменных окружения. Токен бота телеграмм сохранен в файле с расширением .env;
* bot = Bot(token=token, default=DefaultBotProperties(parse\_mode=ParseMode.HTML)): Эта строка инициализирует экземпляр бота Telegram, используя токен, полученный из переменных окружения. Для параметра parse\_mode установлено значение ParseMode.HTML, что позволяет использовать HTML-теги в сообщениях бота;
* dp = Dispatcher(): Эта строка инициализирует новый экземпляр Dispatcher для обработки входящих обновлений;
* dp.include\_router(start\_router): Эта строка добавляет start\_router в диспетчер, позволяя ему обрабатывать входящие обновления;
* asyncdefstart(): ...: Это основная точка входа для выполнения бота. Он запускает цикл опроса входящих обновлений с помощью экземпляра dp и закрывает сессию бота по завершении цикла;
* if \_\_name\_\_ == '\_\_main\_\_': asyncio.run(start()): Это точка входа для скрипта при запуске в качестве автономной программы. Он запускает функцию start() с помощью функции asyncio.run(), которая запускает цикл событий и выполняет функцию start() до его завершения.

## Код парсера информации сайта

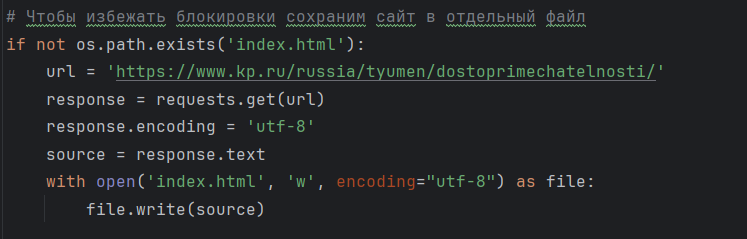


Рисунок 3.2 - Часть кода из parser.py ответственный за запрос и токен

Этот блок кода проверяет, существует ли файл с именем "index.html" в текущей директории. Если он не существует, код переходит к загрузке HTML-содержимого URL 'https://www.kp.ru/russia/tyumen/dostoprimechatelnosti/' с помощью библиотеки запросов Python.

Загруженное HTML-содержимое кодируется в формате UTF-8, чтобы обеспечить правильную обработку символов. Затем содержимое записывается в новый файл с именем "index.html" с помощью функции open с режимом 'w', что означает режим записи. Параметр 'encoding' имеет значение 'utf-8', чтобы убедиться, что содержимое сохраняется в правильном формате кодировки.

В общем, этот блок кода проверяет существование файла, загружает содержимое с URL-адреса, если файл отсутствует, и сохраняет содержимое в файл с именем "index.html".

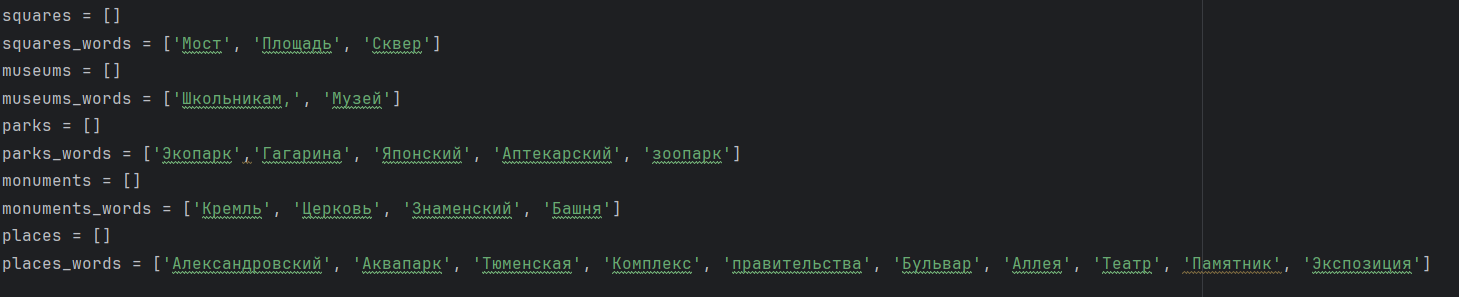


Рисунок 3.3 - Часть кода из parser.py отвественныйза создание массива для записи данных

Этот блок кода инициализирует несколько пустых списков и присваивает их различным переменным: squares, squares\_words, museums, museums\_words, parks, parks\_words, monuments, monuments\_words и places.

Список squares будет использоваться для хранения информации о площадях, а squares\_words - это список ключевых слов, которые могут быть использованы для идентификации площадей в HTML-контенте. Также существуют следующие списки:

* в списке museums будет хранить информацию о музеях, а museums\_words содержит ключевые слова для идентификации музеев;
* в списке parks будет храниться информация о парках, а parks\_words содержит ключевые слова для идентификации парков;
* в списке monuments будет храниться информация о памятниках, а в monument\_words - ключевые слова для идентификации памятников;
* в списке places будет храниться информация о различных местах, а places\_words содержит ключевые слова для идентификации этих мест.

Эти списки будут использоваться для извлечения релевантной информации из загруженного HTML-контента на основе наличия определенных ключевых слов.



Рисунок 3.4 – Часть кода из parser.py отвественный за очистку лишних данных

Этот блок кода определяет функцию add\_list, которая принимает три аргумента: places, words и new\_list.

Функция инициализирует пустой список indices\_to\_remove для отслеживания индексов элементов списка places, которые будут удалены после обработки.

Затем функция выполняет итерацию по каждому элементу списка places, используя цикл for с функцией enumerate для получения индекса и самого элемента. Для каждого элемента внутренний цикл for проверяет, присутствует ли какое-либо слово из списка слов. Если слово найдено, элемент обрабатывается путем удаления всех неперекрывающихся пробельных символов (\xa0) и замены их на обычные пробелы. Обработанный элемент добавляется в список new\_list с форматированной строкой, содержащей индекс и название места.

Индекс обработанного элемента добавляется в список indices\_to\_remove, и внешний цикл for переходит к следующему элементу.

После обработки всех элементов функция сортирует список indices\_to\_remove в порядке убывания и удаляет элементы из списка мест с помощью цикла for с оператором del.

Наконец, функция возвращает обновленный список new\_list.

Эта функция предназначена для обработки списка географических названий и извлечения из него географических названий, соответствующих ключевым словам, указанным в списке слов. Извлеченные географические названия добавляются в новый список с помощью форматированной строки, содержащей индекс и название места. Исходный список географических названий также обновляется путем удаления обработанных элементов.

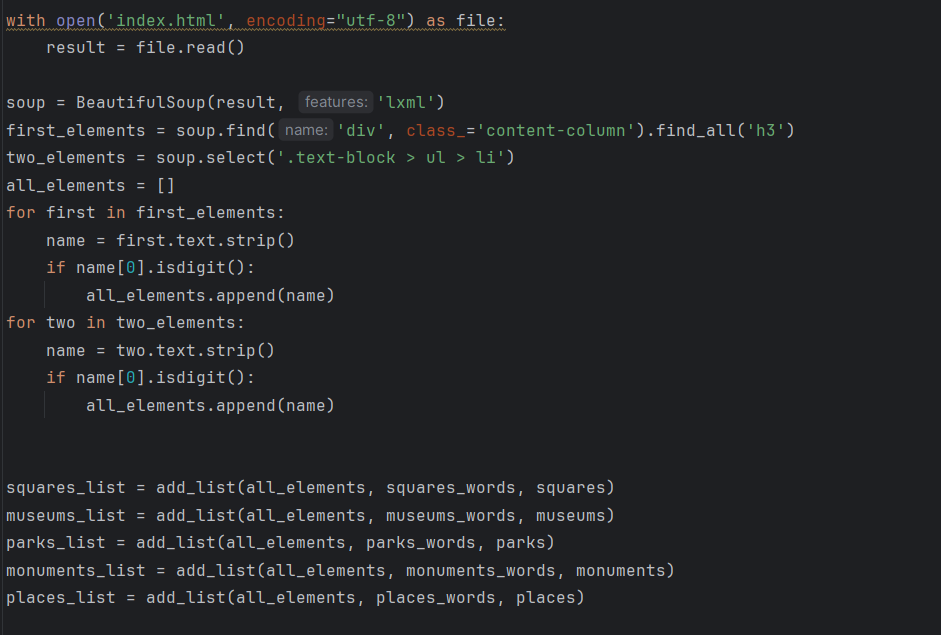


Рисунок 3.5 – Часть кода из parser.py отвественный за заполнение пустых массивов данными

Этот блок кода считывает содержимое файла «index.html», обрабатывает его с помощью библиотеки BeautifulSoup и извлекает информацию о площадях, музеях, парках, памятниках и других местах.

Сначала код открывает файл «index.html» с кодировкой UTF-8 и считывает его содержимое в строковую переменную с именем result.

Затем библиотека BeautifulSoup используется для анализа содержимого HTML и извлечения элементов <h3> первого уровня и элементов <li> второго уровня, являющихся прямыми потомками элементов <ul> в классе .text-block. Эти элементы хранятся в переменных first\_elements и two\_elements соответственно.

Затем код инициализирует пустой список с именем all\_elements и перебирает списки first\_elements и two\_elements, чтобы извлечь имена мест. Если имя начинается с цифры, оно добавляется в список all\_elements.

После этого код обрабатывает список all\_elements для извлечения названий площадей, музеев, парков, памятников и других мест с помощью функции add\_list. Функция add\_list принимает три аргумента: список all\_elements, список ключевых слов (squares\_words, museums\_words, parks\_words, monuments\_words или places\_words) и целевой список (квадраты, музеи, парки, памятники или места).

Функция add\_list проверяет, присутствует ли какое-либо из ключевых слов в списке слов в каждом элементе списка all\_elements. Если ключевое слово найдено, элемент обрабатывается и добавляется в целевой список с отформатированной строкой, которая включает индекс и имя места. Исходный список all\_elements обновляется путем удаления обработанных элементов.

Наконец, код присваивает результат функции add\_list соответствующему целевому списку (squares\_list, museums\_list, parks\_list, monuments\_list или places\_list).

## Код телеграмм бота

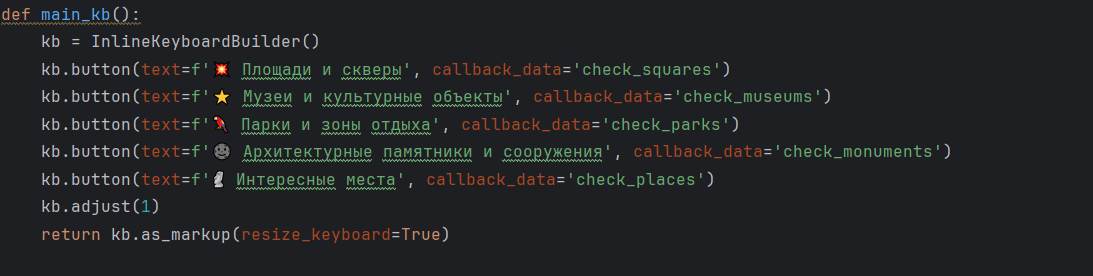


Рисунок 3.5 – Главное меню

Этот фрагмент кода определяет функцию main\_kb, которая создает встроенную клавиатуру с помощью класса InlineKeyboardBuilder. Функция создает пользовательскую клавиатуру с кнопками для различных категорий, связанных с достопримечательностями.

Вот описание того, что делает функция:

Инициализирует экземпляр InlineKeyboardBuilder как kb.

Добавляет на клавиатуру кнопки для различных категорий:

* "Площади и скверы" с данными обратного вызова 'check\_squares';
* "Музеи и культурные объекты" с данными обратного вызова 'check\_museums';
* "Парки и зоны отдыха" с данными обратного вызова 'check\_parks';
* "Архитектурные памятники и сооружения" с данными обратного вызова 'check\_monuments';
* "Интересные места" с данными обратного вызова 'check\_places'.

Настраивает раскладку клавиатуры так, чтобы на каждый ряд приходилось по одной кнопке, используя kb.adjust(1).

Преобразует клавиатуру в формат разметки с возможностью изменения размера клавиатуры с помощью kb.as\_markup(resize\_keyboard=True).

Возвращает отформатированную разметку клавиатуры.

Эта функция предназначена для создания удобного интерфейса встроенной клавиатуры для выбора различных категорий, связанных с местами, что облегчает пользователям навигацию и выбор интересующей их информации.

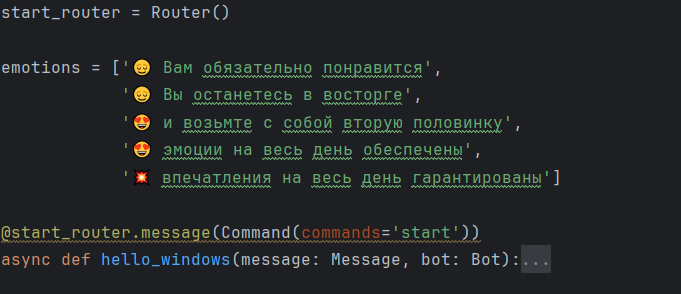


Рисунок 3.6 - Часть кода из parser.py отвественный за небольшие комментарие к местам и начало работы бота с команды start

Этот фрагмент кода создает экземпляр Router с именем start\_router и определяет список положительных эмоций под названием emotions. Затем он добавляет обработчик для команды /start.

Вот пошаговое описание кода:

* start\_router = Router(): Создает экземпляр Router с именем start\_router, который будет использоваться для маршрутизации обработчиков сообщений и колбэков;
* emotions = ['Вам обязательно понравится', ' Вы останетесь в восторге', 'и возьмите с собой вторую половинку', 'эмоции на весь день обеспечены', 'впечатления на весь день гарантированы']: Определяет список положительных эмоций, которые будут использоваться в сообщениях бота;
* @start\_router.message (Command (commands='start')): Определяет обработчик для команды /start. Когда пользователь отправляет эту команду, выполняется функция hello\_windows;
* asyncdefhello\_windows(message: Message, bot: Bot):: Это асинхронная функция, которая принимает объект сообщения message и объект бота bot. Она отправляет сообщение пользователю с приветствием и предложением узнать, куда сходить в Тюмени, используя клавиатуру main\_kb();
* awaitbot.send\_message(message.from\_user.id, 'Здравствуйте \n\n расскажу Вам куда сходить в Тюмене', reply\_markup=main\_kb()): Отправляет сообщение пользователю с приветствием и предложением узнать, куда сходить в Тюмени, используя клавиатуру main\_kb().

Этот фрагмент кода устанавливает начальную точку для бота, обрабатывает команду /start и отправляет приветственное сообщение с предложением узнать о местах для посещения в Тюмени.

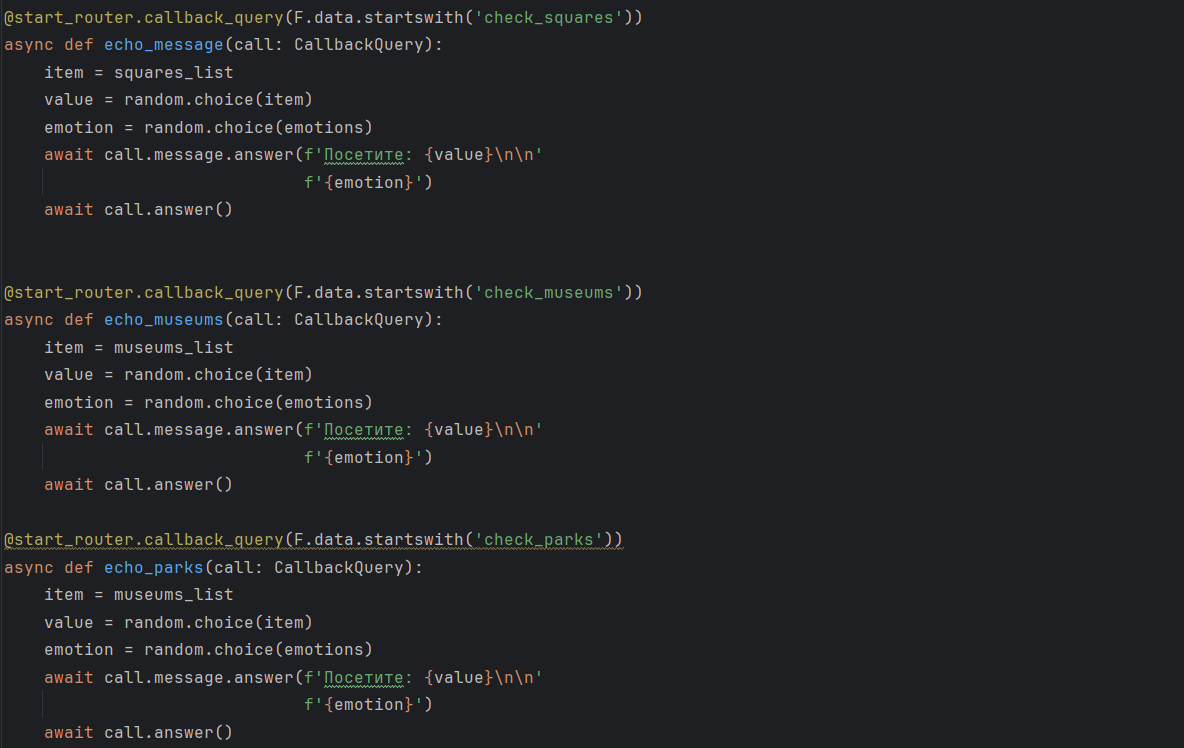


Рисунок 3.7 - Часть кода из parser.py отвественный обратный вызов и действия пользователя



Рисунок 3.8 - Часть кода из parser.py отвественный обратный вызов и действия пользователя

Этот фрагмент кода определяет несколько функций обратного вызова в start\_router для обработки различных действий, вызванных взаимодействием пользователя с Telegram-ботом. Каждая функция обратного вызова реагирует на определенное действие, связанное с различными категориями мест в Тюмени. Вот описание каждой функции обратного вызова:

Обратный вызов для "check\_squares":

* выбирает случайный элемент из списка squares\_list;
* выбирает случайную положительную эмоцию из списка эмоций;
* отправляет пользователю сообщение с рекомендацией выбранного квадрата и связанной с ним эмоцией.

Обратный вызов для 'check\_museums':

* выбирает случайный элемент из списка museums\_list;
* выбирает случайную положительную эмоцию из списка эмоций;
* отправляет пользователю сообщение с рекомендацией выбранного музея и связанной с ним эмоцией.

Обратный вызов для 'check\_parks':

* выбирает случайный элемент из списка\_музеев (должен быть список\_парков);
* выбирает случайную положительную эмоцию из списка эмоций;
* отправляет пользователю сообщение с рекомендацией выбранного парка и связанной с ним эмоцией.

Обратный вызов для 'check\_monuments':

* выбирает случайный элемент из списка\_памятников;
* выбирает случайную положительную эмоцию из списка эмоций;
* отправляет пользователю сообщение с рекомендацией выбранного памятника и связанной с ним эмоцией.

Обратный вызов для функции 'check\_places':

* выбирает случайный элемент из списка places\_list;
* выбирает случайную положительную эмоцию из списка эмоций.

Отправляет пользователю сообщение с рекомендацией выбранного места и связанной с ним эмоцией.

Эти функции обратного вызова повышают вовлеченность пользователей, предоставляя случайные рекомендации из различных категорий мест в Тюмени вместе с положительными эмоциями, создавая интерактивный и приятный опыт для пользователей, взаимодействующих с Telegram-ботом.

# ЗАКЛЮЧЕНИЕ

В ходе выполнения курсовой работы был получен огромный набор знаний о Телеграм ботах и парсинге данных. Разработка парсинга и создание телеграмм бота затрагивает многие сферы программирования, так и решения разных проблем в целом. Также были изучены возможности разных библиотек языка Python. За счет полученных знаний уже сейчас есть возможность значительного улучшения кода.

В дальнейшем планируется доработка программы, включающая улучшение визуальной составляющей, увеличение данных парсинга, добавление новых функций.

Для достижения данной цели были решены следующие задачи:

* проведен анализ выбранной предметной области;
* выбрана технология и среда разработки;
* собрана информация по поводу парсеров;
* проанализированы возможности и ограничения парсеров;
* изучен процесс создания парсеров;
* разработан чат-бота на платформе Telegram.

Все поставленные задачи были решены, цель достигнута.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

* 1. «GeekBrains» [Электронныйресурс]. Режим доступа: – https://gb.ru/blog/chto-takoe-telegram/ (Дата обращения: 20.10.2023).

1. «Pampadu» [Электронныйресурс]. Режим доступа: – https://pampadu.ru/blog/3788-kak-zarabotat-na-telegram-kanale-instruktsiya-dlya-arbitrazhnika/ (Дата обращения: 20.10.2023).
2. «Skillbox» [Электронный ресурс]. Режим доступа: – https://skillbox.ru/media/code/parsing-sayta-vmeste-s-python-i-bibliotekoy-beautiful-soup-prostaya-instruktsiya-v-tri-shaga/ (Дата обращения: 20.10.2023).
3. «Startpack» [Электронныйресурс]. Режим доступа: – https://startpack.ru/application/telegram-messenger (Дата обращения: 20.10.2023).
4. «Timeweb» [Электронный ресурс]. Режим доступа: – https://timeweb.com/ru/community/articles/chto-takoe-parser (Дата обращения: 20.10.2023).
5. «Xakep» [Электронныйресурс]. Режим доступа: – https://xakep.ru/2021/11/28/python-telegram-bots/ (Дата обращения: 23.10.2023).
6. Zvonobot» [Электронный ресурс]. Режим доступа: https://zvonobot.ru/blog/parser(Дата обращения: 25.10.2023).
7. Книга «Разработка чат-ботов и разговорных интерфейсов» автор Джанарсанам Срини.
8. Статья «Cyberleninka» [Электронный ресурс]. Режим доступа: https://cyberleninka.ru/article/n/issledovanie-pravilnosti-i-effektivnosti-sredstv-parsinga-informatsii-na-veb-resursah/viewer(Дата обращения: 20.10.2023).
9. «Хабр» [Электронный ресурс]. Режим доступа: – https://habr.com/ru/companies/click/articles/494020/ (Дата обращения: 20.10.2023).