МИНОБРНАУКИ РОССИИ
Федеральное государственное бюджетное образовательное учреждение высшего образования
«САРАТОВСКИЙ НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ ИМЕНИ Н.Г. ЧЕРНЫШЕВСКОГО»

Кафедра математической теории упругости и биомеханики

Построение интегративного сервиса на базе социальных сетей

АВТОРЕФЕРАТ БАКАЛАВРСКОЙ РАБОТЫ

студента 4 курса 442 группы
направления 09.03.03 – Прикладная информатика
механико-математического факультета

Козорез Андрея Денисовича

Научный руководитель
к.ю.н., доцент _____ Р.В. Амелин
подпись, дата

Зав. кафедрой
д.ф.-м.н., профессор _____ Л.Ю. Коссович
подпись, дата

Саратов 2022

ВВЕДЕНИЕ

Социальные сети и другие источники информации стали спутниками жизни многих людей за последние годы. Почти все люди каждый день пользуются подобными новостными ресурсами, и большинство использует сразу несколько подобных источников.

В некоторых социальных сетях есть лента новостей[1], она позволяет быстро получить актуальную сводку новостей из интересных пользователю источников. Но новостных сайтов, мессенджеров и социальных сетей слишком много и на каждой из этих платформ разные группы, что затрудняет пользователям поиск нужной информации. Данную проблему раньше могли решить RSS-агрегаторы[2], но их популярность ушла очень быстро из-за платного использования и других немаловажных факторов.

В настоящее время люди ежедневно отслеживают любимые группы, каналы и подобные источники в разных приложениях или же сайтах. Это забирает у пользователей лишнее время на поиск актуальных новостей, так как, как было написано ранее, не каждая онлайн-платформа имеет новостную ленту.

Цели и задачи. В связи с вышеописанным, целью данной работы является разработка RSS-агрегатора, поддерживающего работу на современных платформах, для отслеживания различных истоников информации.

Для достижения этой цели должны быть выполнены следующие задачи:

1. Сформулировать требования к системе;
2. Спроектировать систему;
3. Выбрать платформу и инструменты реализации;
4. Реализовать целевой продукт;
5. Провести анализ проделанной работы.

Объекты исследования. В качестве объекта исследования выступает Telegram, как способ удобного представления актуальной информации, а также язык программирования Python.

Научная новизна. Разработан бот RSS-агрегатор для формирования

[1]Новостная лента — формат данных, используемый для доставки пользователям часто обновляемой информации. Распространители этой информации предоставляют новостную ленту, позволяя пользователям подписаться на неё.

[2]RSS-агрегатор — клиентская программа или веб-приложение для автоматического сбора сообщений из источников, экспортирующих в форматы RSS или Atom, например, заголовки новостей, блогов, подкастов и видеохостингов.

удобной новостной ленты из множества источников информации.

Научная значимость. Полносттю реализованные сервис отслеживания новостной информации позволит пользователям просматривать новостную ленту из разнообразных источников в удобном формате, а также позволит экономить время на просмотр той же информации в разных приложениях.

Структура и объем работы. Выпускная квалификационная работа состоит из введения, 3 глав, заключения, списка использованных источников, включающего 13 наименований, работа изложена на 39 листах машинописного текста, содержит 17 рисунков. Далее приведены наименования глав:

1. Поставновка задачи
2. Реализация бота
3. Демонстрация работы Telegram бота

ОСНОВНОЕ СОДЕРЖАНИЕ РАБОТЫ

Во введении обоснована актуальность выбора объектов исследования, сформулированы цели и задачи, а также представлена научная новизна.

Первая глава посвящается постановке задач, здесь рассмотрена проблема отсутствия новостной ленты во многих социальных сетях. Приведено решение на основе бота RSS-агрегатора в Telegram (рисунок 1).

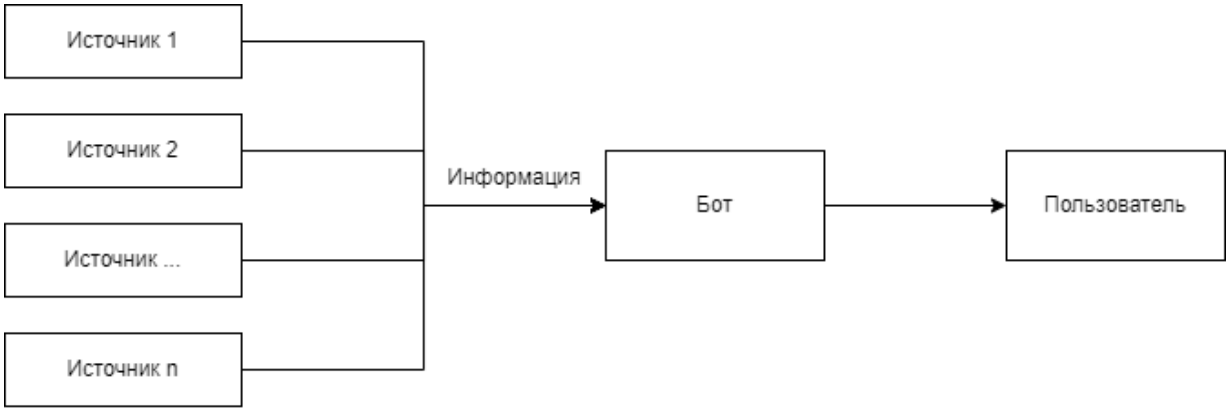

Рисунок 1 — Схема работы бота

Глава разделяется на четыре раздела:

Первый раздел первой главы посвящается технологиям создания чат-ботов. Приведены сферы применения, основной функционал, классфикация по способу функционирования и основные недостатки.

Во втором разделе первой главы рассмотрены языки программирования, с помощью которых возможна реализация поставленной задачи, в том числе:

1. JavaScript;
2. PHP;
3. Python.

Для реализации выбран высокоуровневый язык программирования Python.

В третьем разделе первой главы рассмотрены платформы, на основе которых может использоваться целевой программный продукт, а также способы представления удобного интерфейса пользователя. В качестве целевой платформы выбран вариант реализации бота в мессенджере Telegram.

В четвертом разделе первой главы приведен перечень необходимых для реализации модулей и библиотек для языка Python, в их числе:

1. pyTelegramBotApi;
2. telethon;
3. asycio;
4. subprocess;
5. re;
6. json.

Вторая глава посвящается реализации Telegram-бота и разделяется на семь разделов:

1. **Этапы разработки.** В данном разделе приведены этапы разработки поставленной задачи;
2. **Возможности Telegram ботов.** Данный раздел описывает функциональные возможности Telegram ботов в общем случае и конкретно для бота RSS-агрегатора;
3. **Регистрация бота.** В данном разделе описан алгоритм регистрации нового Telegram бота при помощи виртуального помощника-бота – @BotFather;
4. **Подключение библиотеки pyTelegramBotApi.** В разделе начинается реализация меню бота с использованием бибилиотеки pyTelegramBotApi, здесь же приведен исходный код с пояснениями (рисунок 2);

```
31  @bot.message_handler(commands=['start'])
32  def start(message):
33
34      markup = types.ReplyKeyboardMarkup(resize_keyboard=True)
35      btn1 = types.KeyboardButton("👋 Поздороваться")
36      btn2 = types.KeyboardButton("Меню")
37      markup.add(btn1, btn2)
38      bot.send_message(message.chat.id, text="Привет, {0.first_name}!
```

Рисунок 2 — Код начального меню

5. **Подключение библиотеки json.** В данном разделе описаны методы хранения и обработки данных в формате json при помощи одноименной

библиотеки. Главным хранилищем выступает файл data.json, чтение и запись в который происходит в функциях-обработчиках запросов;

6. **Библиотека telethon.** Данный раздел описывает особенности использования бибилиотеки telethon для сбора и дальнейшего парсинга данных с Telegram-каналов. Описан алгоритм создания Telegram приложения (рисунок 3);

Рисунок 3 — Создание Telegram приложения

7. **Подключение библиотеки subprocess.** В данном разделе описан метод взаимодействия приложения-агрегатора и чат-бота при помощи библиотеки subprocess.

В третьей главе приведена демонстрация работы реализованного Telegram-бота. Глава разделяется на два раздела:

В первом разделе третьей главы рассмотрено начало работы с ботом: его запуск и добавление источников (рисунок 4).

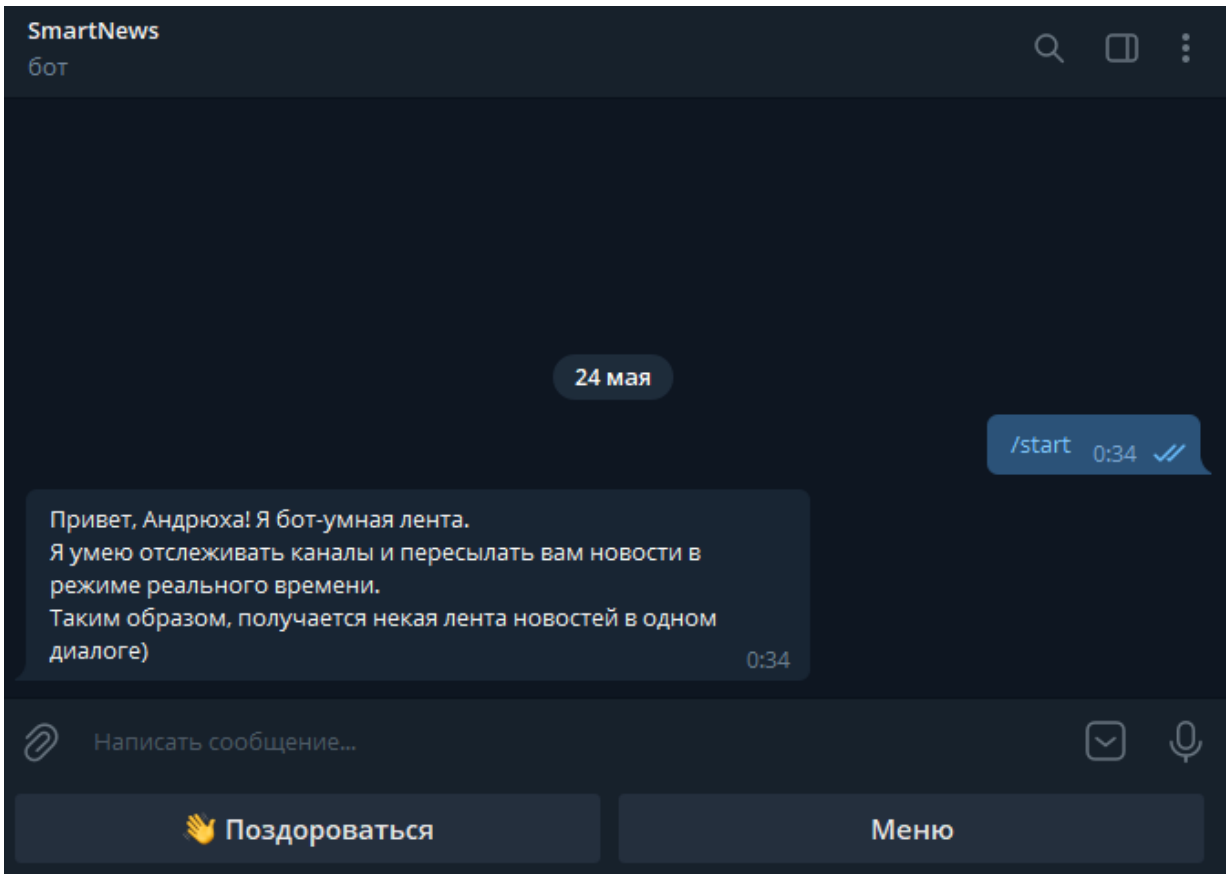

Рисунок 4 — Начало работы с ботом

Во втором разделе третьей главы приведена демонстрация работоспособности функций парсера. Раздел содержит в себе четыре подраздела:

1. **Сообщения-альбомы.** Подраздел включает в себя демонстрацию обработки «сообщений-альбомов» – сообщений с несколькими медиа-вложениями;
2. **Пересланные сообщения.** Демонстрирует обработку пересланных сообщений, также указывается невозможность подписи такого вида сообщения;
3. **Голосовые и видео-сообщения.** В данном подразделе продемонстрирована обработка голосовых и видео-сообщений;
4. **Удаление источника.** Заключительный подраздел демонстрирует удаление источника с последующим изменением json файла.

ЗАКЛЮЧЕНИЕ

В рамках выполнения выпускной квалификационной работы был создан телеграм-бот выполняющий сбор актуальной новостной информации из различных источников в хронологическом порядке.

Данный продукт поможет пользователям оптимизировать свой процесс получения новостей. Сделает его более быстрым и удобным.

Для достижения цели, я проделал следующие работы:

1. Были сформированы требования к боту;
2. Были выбраны необходимые инструменты для реализации;
3. Реализован бот RSS-агрегатор в приложении Telegram, написанный на языке Python;
4. Была проведена демонстрации работы реализованного бота.

Пользователь различных приложений, который ежедневно просматривает множество источников информации, теперь может в удобном и доступном для него виде получать актуальные новости из интересных ему ресурсов. В конечном итоге мы решили проблему использования разных приложений для отслеживания любимых источников. При увеличении пользователей бота не снизится качество отслеживания новостей.

Представшая проблема неэффективного распределения времени на просмотр свежих новостей решилась автоматизацией. При этом важность того, чтобы автоматизированное решение было надежным сохранилась. Бот представляет простой и удобный интерфейс для пользователя, возможность добавления новых источников в любой момент.

Считаю, что работа успешно закончена, все цели, поставленные в начале работы, достигнуты, а задачи решены.