Міністерство освіти і науки України
Чернівецький національний університет
імені Юрія Федьковича
Інститут фізико-технічних та комп'ютерних наук
(повна назва інституту/факультету)
кафедра інформаційних технологій та комп'ютерної фізики
(повна назва кафедри)

Онлайн агрегатор кінофільмів та новин про кіно

Дипломна робота
Рівень вищої освіти - перший (бакалаврський)

Виконав:
студент 4 курсу, групи 417
спеціальності
126 – інформаційні системи та технології
(шифр і назва спеціальності)
Дребот Д. С. _____
(прізвище та ініціали)
Керівник: доктор технічних наук, доцент
Баловсяк Сергій Васильович _____
(науковий ступінь, вчене звання, прізвище, ім'я, по-батькові)
Рецензент_____
_____
(науковий ступінь, вчене звання, прізвище, ім'я, по-батькові)

До захисту допущено:
Протокол засідання кафедри № 19
від „ 17 ” червня 2021 р.
зав. кафедри _____ доц. Борча М.Д.

Чернівці – 2021

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЧЕРНІВЕЦЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ
ІМЕНІ ЮРІЯ ФЕДЬКОВИЧА

Інститут фізико-технічних та комп'ютерних наук
Кафедра інформаційних технологій та комп'ютерної фізики

ЗАТВЕРДЖУЮ
Завідувач кафедри
докт. фіз.-мат. наук, доц.
_____М. Д. Борча
" ___" _____ 2021 р.

ОНЛАЙН АГРЕГАТОР КІНОФІЛЬМІВ ТА НОВИН ПРО КІНО

ЛИСТ ЗАТВЕРДЖЕННЯ

УЗГОДЖЕНО
Керівник роботи
докт. техн. наук, доцент
_____ С.В. Баловсяк
"___" _____ 2021 р.

Виконавець
студент 4-го курсу
_____ Д. С. Дребот
"___" _____ 2021 р.

2021

ЗАВДАННЯ

НА БАКАЛАВРСЬКУ РОБОТУ СТУДЕНТУ

Дреботу Дмитру Святославовичу
(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) Онлайн агрегатор кінофільмів та новин про кіно

керівник роботи *Баловсяк Сергій Васильович, докт. техн. наук, доцент*,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)
затверджена наказом вищого навчального закладу від "__"_____202_ року №_____

2. Строк подання студентом проекту (роботи) *27 травня 2021 р.*

3. Вихідні дані до проекту (роботи) *Мета роботи – створення онлайн інтегратора та спрощення вибору кінофільмів користувачем. Інформацію про фільми отримувати з веб-сайтів, для взаємодії з програмою використати Telegram-бот. При виборі кінофільмів враховувати їх рейтинг. Мова програмування – Python.*

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1) описати застосування Telegram-ботів
2) описати структуру та функції системи
4) створити програму для вибору кінофільмів
5) дослідити ефективність розробленої програми

5. Перелік графічного матеріалу
1) Структура та функції системи
2) Приклад роботи програми

Студент _____ *Дребот Д.С.*
(підпис) (прізвище та ініціали)

Керівник проекту (роботи) _____ *Баловсяк С.В.*
(підпис) (прізвище та ініціали)

ЗМІСТ

АНОТАЦІЯ

Кваліфікаційна робота була виконана студентом 417 групи, Дреботом Дмитром Святославовичом. Тема «Онлайн агрегатор кінофільмів та новин про кіно». Робота направлена на здобуття ступеня бакалавр за спеціальністю 126 «Інформаційні системи на технології». Результатом виконання кваліфікаційної роботи є інформаційна система з двох парсерів, які зчитують інформацію з веб-сайту та Telegram-бот для взаємодії з програмою. Створений програмний продукт дозволяє отримувати новини з індустрії кіно в зручному форматі у вигляді повідомлень в месенджері Telegram. Також другою функцією інформаційної системи є функція рандомного підбору кінофільмів з рейтигового списку веб-сайту.

Робота містить: 29 сторінок, 3 рисунки, 4 посилання на літературні джерела.

Ключові слова: Telegram-бот, парсер, інформаційна система, підписка на новини.

ABSTRACT

Qualification work was performed by a student of the 417 group Drebot Dmytro. Topic "Online aggregator of films and news about movie industry". The work is aimed at obtaining a bachelor's degree in specialty 126 "Information Systems and Technologies". The result of the qualification work is an information system of two scrapers that read information from the website and Telegram-bot to interact with the program. The created software product allows to receive news from the film industry in a convenient format in the form of messages in the Telegram messenger. Also, the second function of the information system is the function of random selection of movies from the ranking list of the website.

Bachelor thesis contains 22 pages, 3 figures and 4 references to literary sources.

Keywords: Telegram-bot, scraper, information system, news subscrabition.

ПЕРЕЛІК УМОВНИХ ЗНАЧЕНЬ

ПЗ – програмне забезпечення.

ПК – персональний комп’ютер.

TG – Telegram

БД – база даних

ID – ідентифікатор користувача в Telegram

ВСТУП

Велика кількість людей час-від часу переглядають кінофільми. Для багатьох з них питання вибору хорошої кінокартини досить важливе. Більше того, їх різноманіття змушує довго перглядати десятки сторінок з запропонованими варіантами, сортувати по акторам, країнам, жанрам та іншим характеристикам. Саме тому мною було обрано цей напрямок для розробки інформаційної системи. Програма дозволяє користувачеві обрати один із рейтингових списків, після чого вже робить вибір за нього і видає один із сотні кінофільмів. Також було реалізовано можливість підписки на новини зі сфери кіно. У зв'язку з тим, що користувачі обирають програмні продукти з легким доступом, було обрано розробку Telegrma-бота для швидкої та простої взаємодії.

Об'єктом дослідження є засоби створення Telegram-бота і поєднання його з декількома парсерами, які зчитують інформацію з веб-сайтів.

Предметом дослідження є способи побудови Telegram-бота та орагнізація його роботи для зручного та просто користування.

Мета роботи: створення програмного продукту з простим і швидким доступом з різних операційних систем для перегляду новин зі сфери кіно та підбору кінофільмів.

Методи дослідження. В роботі використано платформу для розробки програмних продуктів repl.it. Мова програмування – Python. Також було використано ряд бібліотек для оргізації коректної роботи парсера та його взаємодії з Telegram API.

РОЗДІЛ 1. ЗАГАЛЬНІ ПОЛОЖЕННЯ

1.1 Системний аналіз об’єкта дослідження та предметної області

Інформаційна система повинна мати мету функціонування та певні критерії якості для відповідної роботи. Таким чином перш ніж розпочати розробку власного програмного забезпечення з перегляду новин та підбору кінофільмів, потрібно визначитись з формою представлення даного продукту та проаналізувати нішу на наявність аналогів та конкурентів. Дослідивши напрямок Telegram-ботів, прийшов до висновку, що ця ніша тільки розвивається, з кожним роком показує хороший ріст і чудово підходить під дану інформаційну систему.

1.1.1 Аналіз мети функціонування системи

Для розуміння мети функціонування системи варто розпочати з розгляду сфери та галузі застосування даного продукту. Як я зазначив раніше, ринок Telegram-ботів розвивається та зростає з кожним роком. Більше того на даний момент ще немає аналогів даного продукту, що значно полегшує вихід на ринок.

Також сервіс з підбору фільмів і підписки на новини про кіно пропонує простий та зрозумілий інтерфейс для взаємодії, а сфера кіно цікава широкій аудиторії. Тому вирішення такої проблеми як вибір фільму буде актуальне для багатьох потенційних користувачів.

1.1.2 Забезпечення якості

Для забезпечення достатньої якості має бути складено ряд вимог до програмного продукту, як зі сторони програми, так і зі сторони користувача. Користувач має чітко розуміти можливості програмного продукту та його цінність.

1.2 Постановка задачі

Отже для забезпечення відповідноїї якості було складено ряд вимог до продукту. Доступ до програмного продукту повинен бути максимально простим та швидким. Адже тілки, як в потенційного користувача виникають певні проблеми з доступом, він перестає бути зацікавленим. Також після запуску Telegram-бота мають бути інтуітивно зрозумілі наступні кроки взяємодії з інформаційною системою. Бот має працювати безперебійно. Баги повинні бути усунені. Програмний продукт має првильно зчитувати та видвати відповідну інформацію. Також бот повинен мати можливість для розширення функіоналу чи зміни напрямку на інший при необхідності, що значно понизить економічні ризики. В програмний продукт повинен бути закладений екномічний інтерес для його подальшої підтримки.

1.2.1 Вимоги до системи

Першою вимогою до системи є її стабільна робота та правильне зчитування даних з вебсайту. Також система повинна витримувати навантаження в декілька тисяч активних користувачів. Не менш важлию вимогою є її надійність та приватність.

1.2.2 Очікувані ефекти від впровадження

Беручи до уваги всі перераховані вимоги, очікується що продукт буде надійним, стабільним в роботі та простим у використанні.

Висновок до розділу 1

Отже так як програма буде розважальною, вона має бути спрямована в першу чергу на задоволення користувача, простоту використання та швидкий доступ. Також інформаційна система повинна повина працювати стабільно, щоб не втратити користувачів і заохочувати їх користуватися нею довше. Основними функціями повинні бути підписка на новини та підбір фільмів по

запиту користувача, також важливою є можливість відписатися від новин, якщо користувач цього забажає.

РОЗДІЛ 2. ПОБУДОВА СИСТЕМИ

2.1 Моделювання системи

Для розробки інформаційною системи було обрану онлайн платформу repl.it, яка дозволяє коп'ютеру зі слабкими характеристками використовувати хмарні сервіси для більш продуктивної роботи. Інформаційна система скоадається з декількох модулів. Перший модуль – це Telehram-бот для взаємодії користувача з програмою, два інших модулі – це два окремих парсери, один отримує інформацію про новини, а інший збирає інформаію про фільми.

2.1.1 Вхідні дані

Для повноцінної роботи програмного продукту, потрібно організувати взаємодію двох парсерів з Telegram-ботом. Також було створено ще два додаткових модулі. Один модуль записує в базу даних тих, хто підписався на новини, видаляє тих хто відписався, та відслідковує тих хто заблокував бота. Другий модуль з певною періодичністю робить запити парсеру новин на перевірку актуальних новин і передає інформацію боту. Таким чином у нас є один основний модуль – це бот, завдяки якому виводиться вся інформація користувачу, два парсера і два допоміжних модулі для парсера новин.

2.1.2 Вихідні дані

Парсер новин з певним інтервалом перевіряє сторінку з новинами. Запис про останню новину зберігається в нереляційній базі даних, також в ній зберігаються ID користувачів, які підписались на новини. Якщо є нова публікація, бот парсер відправляє посилання через бота користувачу.

Парсер фільмів працює дещо по-іншому. Спочатку користувач в боті натискає кнопку "Film" та обирає одну з категорій рейтингів. Після цього бот по заданій категорії передає запит парсеру і парсер по даному списку обирає рандомно один із фільмів і виводить користувачеві.

2.1.3 Функції та структура системи

Основні функції – це надсилання новин тим користувачам, які підписались на розсилку та вибір фільмів по запиту користувача в боті.

Структура полягає в взаємодії двох парсерів з ботом Telegram в якості інтерфейсу. Функції, шлях користувача та структура представлені на рисунках нижче (Рисунки 2.1-2.2).

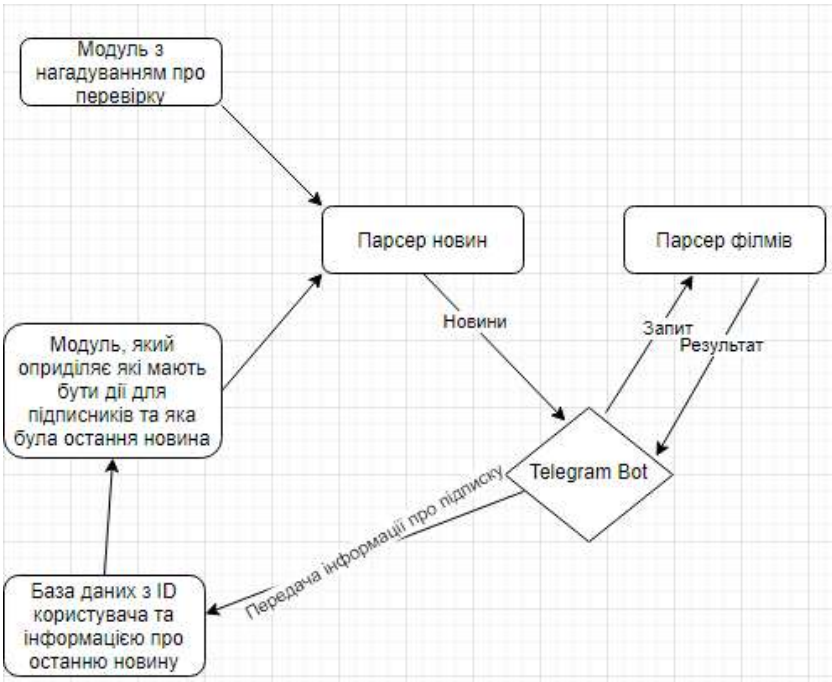

Рисунок 2.1 (Структура системи)

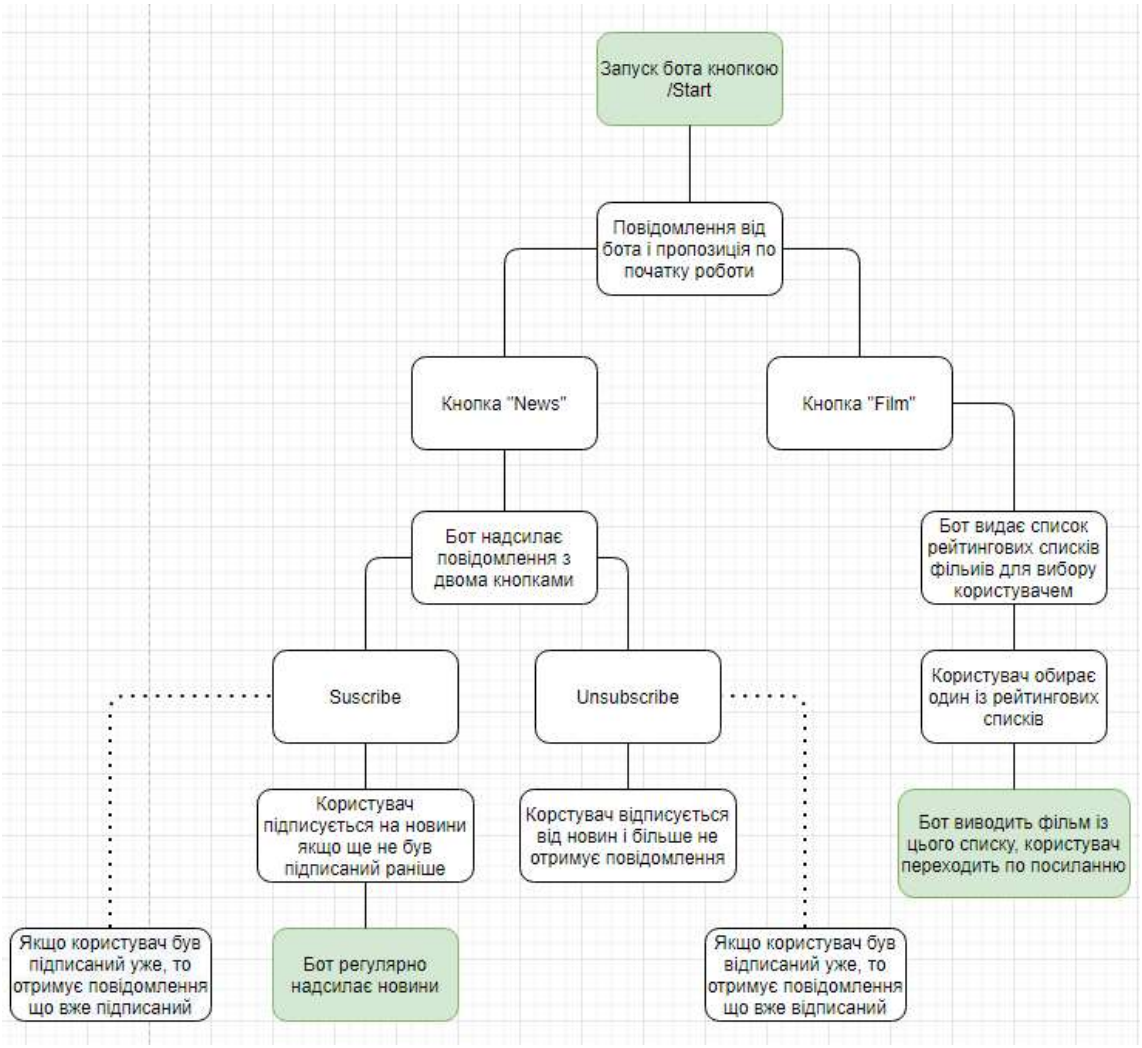

Рисунок 2.2 (Функції бота та шлях користувача)

2.2 Методи та засоби побудови системи

Система базується на Telegram-боті, до якого ми отримуємо доступ за допомогою TOKEN ключа. Після цього ми прописуємо логіку бота та під’єднуємо парсери, до одного з них додаємо ще два модулі, які допомагають краще організувати роботу програмного продукту. Так як єдина інформація, яка зберігається - це ID користувачів та останні новини, потреби використання складної бази даних не було. На даний момент цілком достатньо двох ключів для повноцінної роботи інформаційної системи. Для представлення логіки бази даних та її можливостей, я прикріпив фото з платформи Repl.it (Рисунки 2.3-2.4).

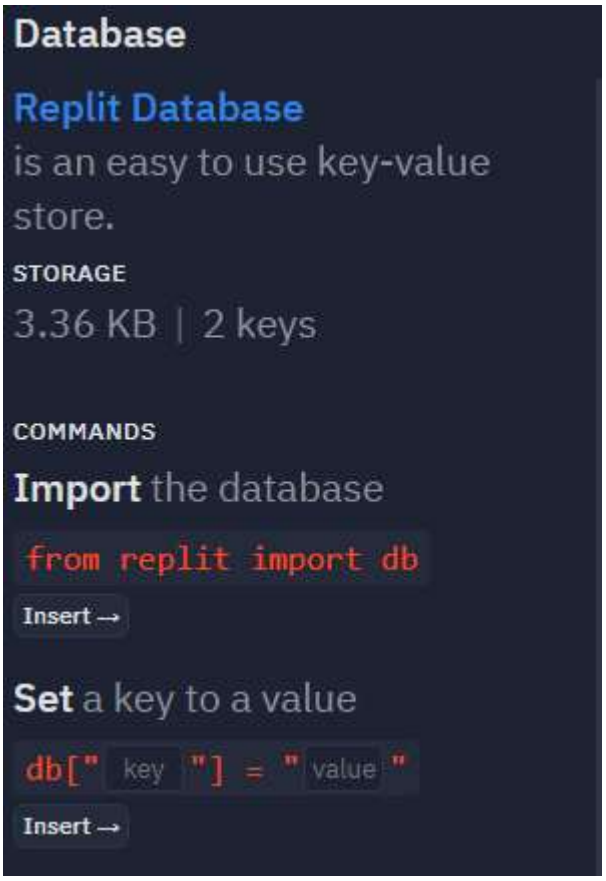

Рисунок 2.3 (Застосування бази даних)

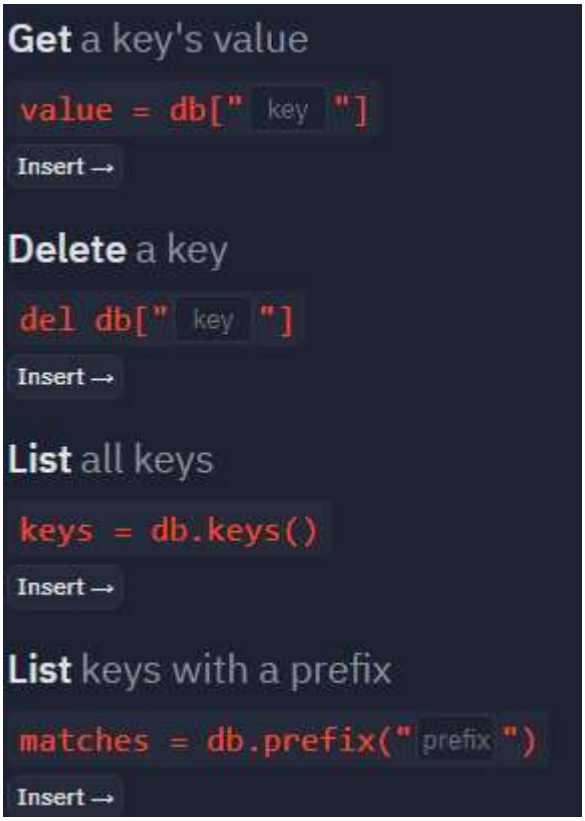

Рисунок 2.4 (Команди бази даних)

Для взаємодії бази даних з програмою, був розроблений окремий модуль, який відслідковує хто з користувачі підписаний на новини. Знімок екрану прикріплено нижче (Рисунок 2.5).

```
1  from replit import db
2
3  SUBSCRIBERS = 'subs'
4
5  def subscribe(user_id):
6    subs = list_subs()
7    if not user_id in subs:
8      subs.append(user_id)
9      db[SUBSCRIBERS] = subs
10     return True
11   return False
12
13 def unsubscribe(user_id):
14   subs = list_subs()
15   if user_id in subs:
16     subs.remove(user_id)
17     db[SUBSCRIBERS] = subs
18     return True
19   return False
20
21 def list_subs():
22   if not list_exists():
23     db[SUBSCRIBERS] = []
24   return db[SUBSCRIBERS]
25
26 def list_exists():
27   return SUBSCRIBERS in db.keys()
```

Рисунок 2.5 (Використання бази даних в модулі ‘subs')

Також для кращої роботи застосував бібліотеку Beautiful Soup, яка допомагає краще організувати роботу парсера, та бібліотеку aiogram, яка дозволяє розроблятита взаємодіяти з ботом Telegram.

Висновок до розділу 2

Отже була побудована система, яка працює в двох напрямках: парсинг новин і парсинг фільмів з рейтингового списку. На новини користувач може підписатись, а при бажанні відписатись. Також було додано два додаткових модулі для кращої роботи парсера новин. З їх допомогою парсер перевіряє веб-сайт в назначений час і оприділяє кому потрібно їх відправити. Дана структура дозволяє працювати двом парсерам незалежно один від одного з більшою ефективністю.

РОЗДІЛ 3. ПРАКТИЧНА РЕАЛІЗАЦІЯ

3.1 Засоби розробки

Для розробки інформаційної системи я використовував платформу repl.it, яка дозволяє недостатньо потужним комп'ютерам користуватись хмарним середовищем розробки. Мовою програмування обрав Python, тому що саме для Python є безліч бібліотек для роботи з Telegram. Для взаємодії з користувачем вибір був між веб-сайтом та Telegram-ботом, проте вирішив що такий формат як веб-сайт значно устарівший ніж бот. Більше того в Telegram люди проводять досить багато часу і при використанні бота їм не потрібно буди виходити з додатку. Саме тому мій вибір зупинився на Telegram. Використовував базу даних запропоновану платформою repl.it. Також використовув ряд бібліотек, найважливіші з них – це Beautiful Soup для зручнішої організації парсера та Aiogram для розробки бота на Python.

3.2 Опис реалізації системи

Система складається з декількох модулів. Перший модуль – це інтерфейс і логіка бота Telegram. Він відповідає за зручну комунікацію користувача з програмою. Два інші модулі - парсери. Перший парсер відповідає за зчитування фільмів з веб-сайту, а другий за зчитування новин. Додатково до основних модулів є також модуль, який перевіряє чи користувач підписаний на розсилку новин і чи потрібно відправляти йому новини.

Інформацію про підписаних користувачів він отримує з бази даних, також в базі даних зберігається інформація про останні новини, щоб бот не повторювався і не надсилав повторно.

Інший додатковий модуль - це нагадування для парсера про перевірку веб-сайту з певним інтервалом.

Це були основні модулі для роботи Telegram-бота, код яких я додам в додатку.

3.3 Аналіз отриманих результатів

Отримали бота-асистента, який може обрати фільм за вас та надсилати актуальні новини зі світу кіно. Бот стабільний, працює без помилок та багів. Підчас розробки виникали деякі проблеми з парсингом неправильних даних, але їх вдалось вирішити. Також в першій тестовій версії можна було підписатися безліч раз на розсилку новин. Зараз цей баг прибрали, і користувач отрмує зауваження, що вже підписаний. В усіх інших напрямках бот працює плавно та стабільно.

3.3.1 Контрольний приклад

Для контрольного прикладу я прикріпив декілька зображень, які демонструють роботу програмного продукту. Інтерфейс зрозумілий та дружелюбний. API Telegram дозволяє інтегрувати безліч функцій, але при цьому не перевантажуючи користувача кнопками, даними та списками. В боті є декілька основних видів взаємодії: команди (/start), outline кнопки (кнопки, які розміщуються внизу екрану «News», «Film»), inline кнопки (кнопки, які з’являються в діалоговому вікні, в моєму випадку це вибір списку фільмів, та кнопки «Subscribe», «Unsubscribe». Детальніше можна розглянути функціонал на фото нижче (Рисунки 3.1-3.6).

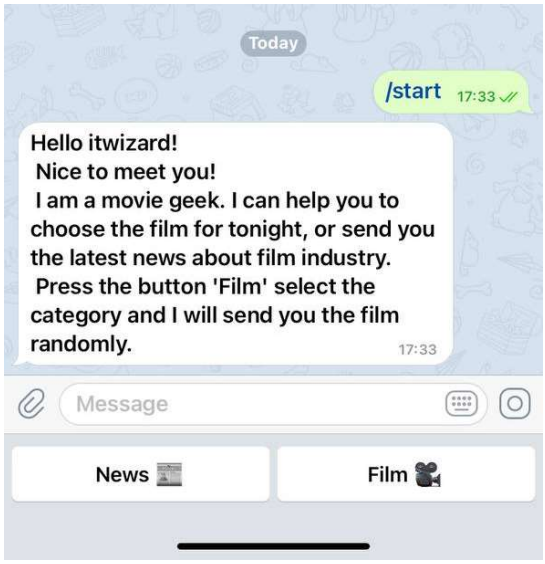

Рисунок 3.1 (Початок роботи бота за допомогою команди '/start’)

Рисунок 3.2 (Натискання на кнопку «News» і можливість підписатися на новини)

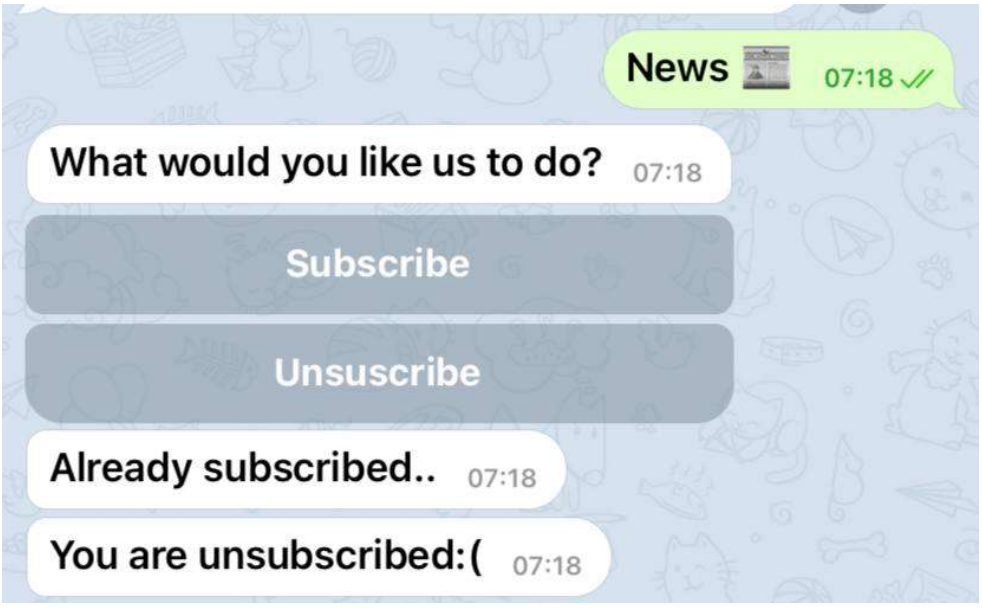

Рисунок 3.3 (Користувач, який підписаний на новини, натискає ще раз «Subscribe» і отримує повідомлення що вже підписаний, після чого відписується і отримує повідомленні про те, що відписався)

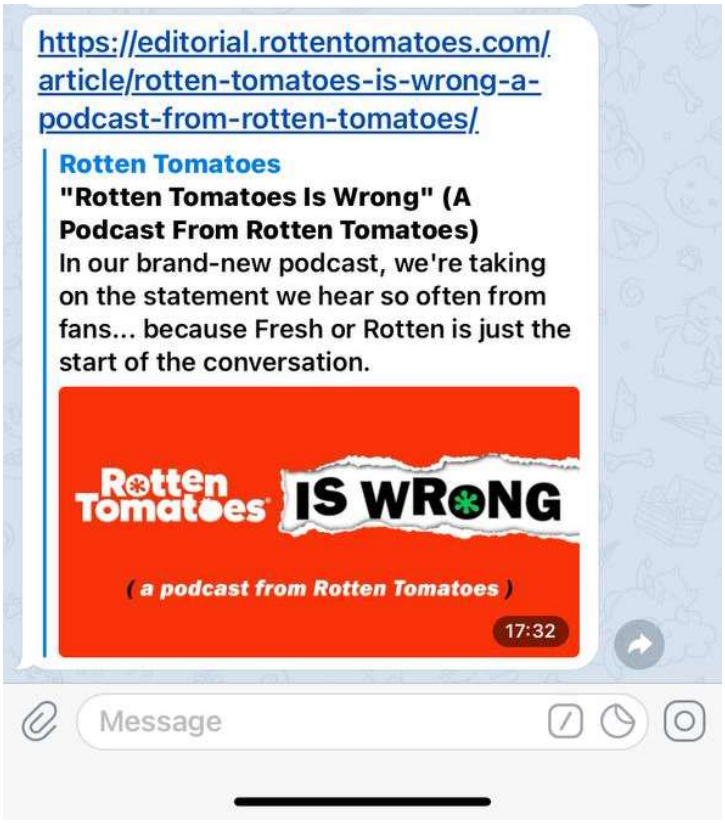

Рисунок 3.4 (Надсилання новин ботом підписаним користувачам)

Рисунок 3.5 (Результат натискання на кнопку «Film»)

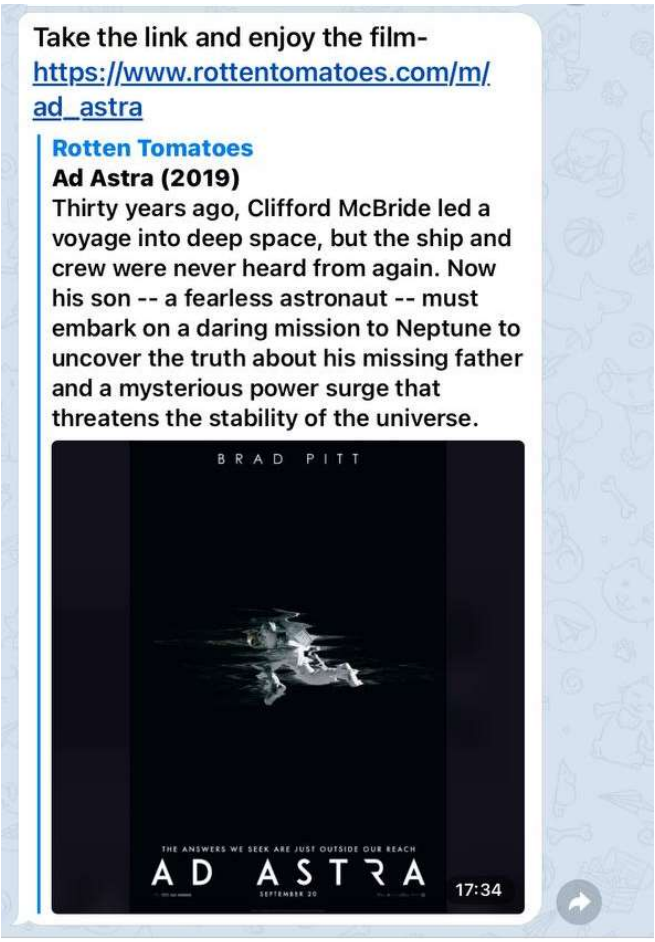

Рисунок 3.6 (Корстувач натиснувши на один із списків отримує фільм від бота)

3.3.2 Аналіз критеріїв якості

Telegram – бот відповідає всім умовам, поставленим раніше. Виконує всі функції, які були зазначені в початкових умовах. За тиждень користування ботом у своїх цілях помилок та багів не було. Відповідно Telegram-бот відповідає всі критеріям якості.

Висновок до розділу 3

Отже провівши аналіз готового додатку, ми зрозуміли, що він відповідає всім вимогам, які були поставлені. Бот складається з декількох модулів і за рахунок того, що в ньому є два окремих парсери, він працює стабільно і без помилок. Інтерфейс зручний та інтуітивно зрозумілий.

ЗАГАЛЬНІ ВИСНОВКИ

Telegram-бот – це інформаційна система, яка була створена для перегляду новин про кіно та для швидкого пошуку та підбору фільмів для пергляду. Даний програмний продукт був об’єктом дослідження напрямку розробки Telegram-ботів. Так як ринок Telegram росте, це був безумовно ціний проект. Мені вдалось дослідити новий напрямок та застосувати знання, набуті за чотири роки навчання.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

Інтернет-ресурси:

1. Telegram-бот на Python и Aiogram: Урок 1 - https://surik00.gitbooks.io/aiogram-lessons/content/chapter1.html
2. Telegram-бот на Python и Aiogram: Урок 2 - https://surik00.gitbooks.io/aiogram-lessons/content/chapter2.html
3. Telegram-бот на Python и Aiogram: Урок 3 - https://surik00.gitbooks.io/aiogram-lessons/content/chapter3.html
4. Telegram-бот на Python и Aiogram: Урок 5 - https://surik00.gitbooks.io/aiogram-lessons/content/chapter5.html
5. Select Movies with Python – Web Scraping Tutorial - https://youtu.be/FoPPgcpSmNs
6. Пишем Telegram бота на Python - https://youtu.be/M8fhrtvedHA
7. Пишем реальный TELEGRAM бот на Python | БД + Парсинг - https://youtu.be/bXxa9IkAPew
8. Асинхронный Telegram-бот на Python / Введение в aiogram - https://youtu.be/I_m2jQ-8p3A
9. Документація бібліотеки Beautiful Soup для парсингу даних з сайтів - https://www.crummy.com/software/BeautifulSoup/bs4/doc.ru/bs4ru.html
10. Документація бібліотеки Aiogram для взаємодії з ботом Telegram - https://docs.aiogram.dev/en/latest/v
11. Документація Telegram bot API - https://tlgrm.ru/docs/bots/api

ДОДАТКИ

В додатки я додам інформацію про основні файли прогрпмного продукту та їх взаємодію між собою.

Файли:

1. main.py
2. buttons.py
3. link.txt
4. scheduler.py
5. scraperfilm.py
6. scrapernews.py
7. subs.py

Файл «main.py», як можна зрозуміти, головний в даному програмному продукті, в ньому прописана вся логіка взаємодії користувача з ботом Telegram та логіка взаємодії Telegram-бота з іншими модулями.

```
import os
from aiogram import Bot, types
from aiogram.dispatcher import Dispatcher
from aiogram.utils import executor
import buttons
import subs
import threading

from scheduler import main

from scraperfilm import scrap_random_film

my_secret = os.environ['TOKEN']

bot = Bot(token=my_secret)
dp = Dispatcher(bot)
```

```
@dp.message_handler(commands=['start'])
async def on_message(message: types.Message):
    await bot.send_message(message.from_user.id, f"Hello
{message.from_user.username}!\n Nice to meet you!\n I am a
movie geek. I can help you to choose the film for tonight, or
send you the latest news about film industry.\n Press the
button 'Film' select the category and I will send you the film
randomly.\n  ", reply_markup=buttons.btns)

@dp.message_handler(content_types=['text'])
async def on_message_2(message: types.Message):
  if message.chat.type =='private':
    if message.text =='Film 📽':
      await bot.send_message (message.from_user.id,"Choose the
list!", reply_markup=buttons.inline_kb1)
    elif message.text == 'News 📰':
      await bot.send_message (message.from_user.id, "What would
you like us to do?", reply_markup=buttons.inline_kb2)

@dp.callback_query_handler(text='button1')
async def process_callback_inline_btn_1(callback_query:
types.CallbackQuery):
    await bot.answer_callback_query(callback_query.id)
    url = scrap_random_film('Top movies 2021')
    await bot.send_message(callback_query.from_user.id, 'Ось
посилання - ' + url)

@dp.callback_query_handler(text='button2')
```

```
async def process_callback_inline_btn_2(callback_query:
types.CallbackQuery):
    await bot.answer_callback_query(callback_query.id)
    url = scrap_random_film('Top movies 2020')
    await bot.send_message(callback_query.from_user.id, 'Take
the link and enjoy the film- ' + url)
@dp.callback_query_handler(text='button3')
async def process_callback_inline_btn_3(callback_query:
types.CallbackQuery):
    await bot.answer_callback_query(callback_query.id)
    url = scrap_random_film('Top movies 2019')
    await bot.send_message(callback_query.from_user.id, 'Take
the link and enjoy the film- ' + url)
@dp.callback_query_handler(text='button4')
async def process_callback_inline_btn_4(callback_query:
types.CallbackQuery):
    await bot.answer_callback_query(callback_query.id)
    url = scrap_random_film('Top movies alltime')
    await bot.send_message(callback_query.from_user.id, 'Take
the link and enjoy the film- ' + url)
@dp.callback_query_handler(text='button10')
async def process_callback_inline_btn_10(callback_query:
types.CallbackQuery):
    await bot.answer_callback_query(callback_query.id)

    if not subs.subscribe(callback_query.from_user.id):
      await bot.send_message(callback_query.from_user.id,
'Already subscribed..')
    else:
```

```
      await bot.send_message(callback_query.from_user.id,
'Thank you! You are subscribed)')

@dp.callback_query_handler(text='button20')
async def process_callback_inline_btn_20(callback_query:
types.CallbackQuery):
    await bot.answer_callback_query(callback_query.id)

    if not subs.unsubscribe(callback_query.from_user.id):
      await bot.send_message(callback_query.from_user.id,
'Already unsubscribed')
    else:
      await bot.send_message(callback_query.from_user.id, 'You
are unsubscribed:(')


if __name__ == '__main__':
    print('Bot started.')
    thread = threading.Thread(target=main, args=())
    thread.daemon = True
    thread.start()
    executor.start_polling(dp)
```

Файл «buttons.py» створений для розробки кнопок Telegram-бота:

```
from aiogram.types import ReplyKeyboardMarkup, KeyboardButton,
InlineKeyboardMarkup, InlineKeyboardButton

btnNews = KeyboardButton ("News 📰")
btnFilm = KeyboardButton ("Film 📽")
```

```
btns = ReplyKeyboardMarkup(resize_keyboard=True).row(btnNews
,btnFilm)


inline_btn_1 = InlineKeyboardButton('Top movies 2021',
callback_data='button1')
inline_btn_2 = InlineKeyboardButton('Top movies 2020',
callback_data='button2')
inline_btn_3 = InlineKeyboardButton('Top movies 2019',
callback_data='button3')
inline_btn_4 = InlineKeyboardButton('Top movies alltime',
callback_data='button4')
inline_kb1 =
InlineKeyboardMarkup().add(inline_btn_1).add(inline_btn_2).add(
inline_btn_3).add(inline_btn_4)


inline_btn_10 = InlineKeyboardButton('Subscribe',
callback_data='button10')
inline_btn_20 = InlineKeyboardButton('Unsuscribe',
callback_data='button20')
inline_kb2 =
InlineKeyboardMarkup().add(inline_btn_10).add(inline_btn_20)
```

Файл «scheduler.py» створений як допоміжний файл для файлу «scrapernews.py». Він надає команду парсеру зчитувати новини з сайту з певним періодом і перевіряти на наявність нових новин і якщо такі є, то дається команди відправлення повідомлення з новинами підписникам.

```
import time
import os
```

```
import asyncio
from scrapernews import scrap_news
from subs import list_subs

from aiogram import Bot

INTERVAL = 3600   # 1 hour

bot = Bot(token=os.environ['TOKEN'])

async def do():
  while True:
    news = scrap_news()
    subs = list_subs()
    for sub in subs:
      for n in news:
        try:
          await bot.send_message(sub, str(n))
        except Exception:
          print('User ' + str(sub) + ' blocked our bot :(')

    time.sleep(INTERVAL)

def main():
  asyncio.run(do())
```

Файл «scrapernews.py» один із основних модулів програмного продукту. Він відповідає за зситування інформації про фільми з сайту Rotten Tomatoes. Саме цей парсер відповідає за вілпрвлення рандомного фільму зі списку, який обрав користувач.

```
import random
import requests
from bs4 import BeautifulSoup


HOSTNAME = 'https://www.rottentomatoes.com'

URLS = {
  'Top movies 2020':
'https://www.rottentomatoes.com/top/bestofrt/?year=2020',
  'Top movies alltime':
'https://www.rottentomatoes.com/top/bestofrt/',
  'Top movies 2019':
'https://www.rottentomatoes.com/top/bestofrt/?year=2019',
  'Top movies 2021':
'https://www.rottentomatoes.com/top/bestofrt/?year=2021'
}

def scrap_random_film(category):
    URL = URLS[category]

    response = requests.get(URL)

    soup = BeautifulSoup(response.text, 'html.parser')

    if category == 'Top movies today':
      titles = soup.select('.movieTitle')
      links = [ title.parent for title in titles ]
    else:
      links = soup.select('td > a.unstyled.articleLink')
```

```
    aTags = filter_movies([ aTag['href'] for aTag in links ])

    id = random.randint(0, len(aTags))
    href = aTags[id]
    return href if href.startswith(HOSTNAME) else (HOSTNAME +
href)

def filter_movies(ls):
  f = []
  for l in ls:
    if is_movie(l):
      f.append(l)
  return f

def is_movie(href):
  return not 'tv' in href
```

Файл «scrapernews.py» ще один із головних модулів прогрпмного продукту. Це парсер, який зчитує дані про новини і відпрвляє підписникам. В цьому йому допомагають файли «scheduler.py», «subs.py». Які відповідають за організацію постійного зчитування актуальних новин та витягування інформації з бази даних про підписників та визначають, які дії мають бути.

```
import requests
from bs4 import BeautifulSoup
from replit import db

NEWS = 'news'

URL = "https://editorial.rottentomatoes.com/news/"
```

```
# кількість остнніх новин які відправляться ще раз
BACKTRACK = 2

def scrap_news():
    response = requests.get(URL)

    soup = BeautifulSoup(response.text, 'html.parser')

    links = soup.select('a.unstyled.articleLink')

    hrefs = []
    bt_counter = 0
    for l in links:
      href = l['href']

      if not href in news_seen():
        seen = news_seen()
        seen.append(href)
        db[NEWS] = seen
        hrefs.append(href)
      elif bt_counter < BACKTRACK:
        hrefs.append(href)
        bt_counter += 1

    return hrefs

def news_seen():
  if NEWS in db.keys():
    return db[NEWS]
  return []
```

Файл «subs.py» витягує дані з бази даних про наявних підписників на новини і передає цю інфорацію файлу «scheduler.py». Також цей файл відповідає за функцію підписки та відписки на новини.

```
from replit import db

SUBSCRIBERS = 'subs'

def subscribe(user_id):
  subs = list_subs()
  if not user_id in subs:
    subs.append(user_id)
    db[SUBSCRIBERS] = subs
    return True
  return False

def unsubscribe(user_id):
  subs = list_subs()
  if user_id in subs:
    subs.remove(user_id)
    db[SUBSCRIBERS] = subs
    return True
  return False

def list_subs():
  if not list_exists():
    db[SUBSCRIBERS] = []
  return db[SUBSCRIBERS]

def list_exists():
  return SUBSCRIBERS in db.keys()
```