МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: 121 «Інженерія програмного забезпечення»

Освітня програма: «Розробка програмного забезпечення»

Група: 4РП-06

Дипломний проект

здобувача освіти денної форми навчання
РП.06.15.000.ДП

МАКОВІЯ ДМИТРА ВІКТОРОВИЧА

м. Одеса
2023 р.

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Спеціальність: **121 «Інженерія програмного забезпечення»**

Освітня програма: «**Розробка програмного забезпечення»**

Група: **4РП-06**

ПОЯСНЮВАЛЬНА ЗАПИСКА

до дипломного проекту (роботи) на тему:

Розробка Telegram-боту та сайту для відстежування оновлень фільмів та серіалів

Проектний матеріал складається з пояснювальної записки на 54 сторінках та графічного (презентаційного) матеріалу на 11 аркушах (слайдах).

Дипломник _____ (Маковій Д.В.)

Керівник _____ (Томаченко О.Є.)

Консультанти:

з економічної частини _____ (Копайгородська Т.Г.)

з охорони праці _____ (Чорновол Н.І.)

з дотримання вимог ЄСКД _____ (Петрашова Т.І.)

старший консультант _____ (Кунуп Т.В.)

До захисту допущений

Голова циклової комісії _____ (Кривченко Ю.В.)

Завідувач відділення _____ (Скорнякова О.В.)

Захист «23» 06 2023 р. Протокол ДКК № 2

Оцінка ДКК 5 (відмінно)

Секретар ДКК _____

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

Відділення комп'ютерних систем Комісія КТ та ПІ

Спеціальність 121 Інженерія програмного забезпечення

Освітня програма Інженерія програмного забезпечення

ЗАТВЕРДЖУЮ:

Заст. дир. з НВР Беркань І.В

"_____" 2023 р.

ЗАВДАННЯ

на дипломний проект (роботу)

Здобувачеві (здобувачці) освіти Маковій Дмитро Вікторович

(прізвище, ім'я, по батькові)

1. Тема проекту (роботи) Розробка Telegram-боту та сайту для відстежування оновлень фільмів та серіалів

затверджена наказом по коледжу від "14" жовтня 2022 р. № 235-А2-06

2. Термін здачі закінченого проекту (роботи)

3. Вихідні дані до проекту (роботи) Система створення запитів до API, отримання потрібних даних після обробки результатів запиту до API, створення функції та змінних для прийому та збереження отриманих даних. Робота циклу функції, завдяки якому отримуються потрібні дані з мережі через певний проміжок часу за деякими умовами.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які необхідно розробити)

Актуальність даної теми дипломної роботи;. Які основні функції має виконувати Telegram-бот та сайт; Розробка архітектури проєкту, вибір використовуваних технологій та інструментів; Проектування бази даних для зберігання інформації про фільми, серіали;. Розробка коду для взаємодії з API для отримання інформації про фільми та серіали.

5. Перелік графічного (презентаційного) матеріалу (з точним зазначенням обов'язкових креслень, кількості слайдів)

Телеграм-бот та сайт для відстеження оновлень фільмів та серіалів. Телеграм-бот, який відстежує оновлення серіалів, дає змогу користувачам отримувати повідомлення про нові серії або сезони їхніх улюблених телесеріалів. Він може бути корисний для тих, хто стежить за кількома серіалами і хоче бути в курсі останніх подій, не заходячи на сайти з розкладом виходу нових серій.Огляд наявних рішень у сфері телеграм-ботів і відстеження оновлення фільмів і серіалівОпис функціоналу бота, включно з підпискою на сповіщення про нові серії та фільми, отримання сповіщень і адміністрування підписок.Розробка архітектури проєкту, вибір використовуваних технологій та інструментів.Проектування бази даних для зберігання інформації про фільми, серіали та підписки користувачів.Розробка коду для взаємодії з API TMDB для отримання інформації про фільми та серіали

Реалізація користувацького інтерфейсу бота, включно з командами, які користувач може використовувати для взаємодії з ботомТестування та налагодження застосунку, включно з тестуванням функціональності та продуктивності бота.Оптимізація продуктивності та надійності бота, включно з оптимізацією бази даних і коду бота

6. Консультанти по проекту (роботі), із зазначенням розділів проекту, що їх стосується

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв
Технологічний	Томаченко О.Є		
Економічна частина	Копайгородська Т.Г		
Охорона праці	Черновол Н.І		
Нормоконтроль	Петрашова В.І		
Старший консультант	Кунуп Т.В		

7. Дата видачі завдання 01.05.2023

Керівник _____
(підпис)

Завдання прийняв до виконання _____
(підпис)

КАЛЕНДАРНИЙ ПЛАН

№ з/р	Назва етапів дипломного проекту (роботи)	Термін виконання етапів дипломного проекту (роботи)	Відмітка про виконання
1	Вступ. Вивчення теорії та аналіз наявних рішень у сфері телеграм-ботів і відстеження оновлення фільмів і серіалів.	24.02.2023	Виконано
2	Визначення вимог до проекту і розробка технічного завдання.	02.03.2023	Виконано
3	Аналітичний розділ. Огляд можливих рішень	21.03.2023	Виконано
4	Планування. Вибір елементної бази/середи розробки	12.04.2023	Виконано
5	Проектування архітектури проекту і вибір технологій.	03.05.2023	Виконано
6	Написання коду для взаємодії з API для отримання інформації про фільми та серіали	12.05.2023	Виконано
7	Реалізація функціоналу надсилання повідомлень про нові серії та фільми користувачам	23.05.2023	Виконано
8	Розробка користувацького інтерфейсу для роботи з ботом	03.06.2023	Виконано
9	Тестування та налагодження бота.	8.06.2023	Виконано
10	Оптимізація продуктивності та надійності бота, розробка інтерфейса сайта	12.06.2023	Виконано
11	Налагодження роботи сайта	14.06.2023	Виконано
12	Питання з охорони праці та економічні розрахунки	17.06.2023	Виконано
13	Підготовка звіту та презентації проекту для захисту	19.06.2023	Виконано
14	Отримання рецензії, відповідей на зауваження	21.06.2023	Виконано
15	Захист дипломного проекту	за графіком	Виконано

Дипломник _____
(підпис)

Керівник _____
(підпис)

Формат	Зона	Поз.	Позначення	Назва	Кіл.	Примітка
				Документація		
			РП 06. 15 000. 00 ДП	Дипломний проект		
А4			РП 06. 15 000. 00 ДП ПЗ	Пояснювальна записка	1	

Зм.	Арк.	№ докум.	Підпис	Дата	РП 06. 02 000. 00 ДП		
Розробив		Маковій Д.В			Розробка Telegram-боту та сайту для відстежування оновлень фільмів та серіалів	Літ.	Аркуш 4 / Аркушів ?
Перевірив		Томаченко О. Є.				Н Д П	
Н. Контр.		Петрашова В.І.				ВСП “ОТФК ОНТУ” гр.4РП-06	
Затверд.		Кривченко Ю.В.					

ЗМІСТ

ВСТУП

Сучасний світ безперервно перетворюється під впливом стрімкого розвитку технологій. Інновації та зміни, які стали невід'ємною частиною нашого буття, мають глибокий вплив на різні аспекти нашого життя, включно зі сферою комунікацій.

Технологічний прогрес призвів до появи безлічі комунікаційних платформ, які змінили наше уявлення про взаємодію з навколишнім світом. Месенджери стали незамінним інструментом для миттєвого обміну повідомленнями, даючи нам змогу залишатися на зв'язку з друзями, сім'єю і колегами в будь-який час і в будь-якому місці.

Однак важливість месенджерів не обмежується лише обміном текстовими повідомленнями. Вони надають можливості для передачі мультимедійного контенту, голосових повідомлень, відеодзвінків і навіть створення групових чатів для колективного спілкування.

Особливе місце серед месенджерів посідає Telegram - одна з найпопулярніших і найбільш швидкозростаючих платформ комунікацій у світі. Telegram пропонує не тільки зручність і безпеку, а й розширений функціонал, що містить такі можливості, як канали для публікації контенту, групові чати, стікери, боти та багато іншого.

Телеграм-боти, зі свого боку, являють собою програми, які автоматизують певні завдання і надають додаткові функції всередині Telegram. Вони дають змогу користувачам отримувати різну інформацію, робити замовлення, отримувати сповіщення, грати в ігри та багато іншого, всього лише взаємодіючи з ботом через текстові повідомлення.

На сьогоднішній день Telegram належить до числа найбільш популярних месенджерів в Україні. Його початкова мета полягала в простому обміні повідомленнями між користувачами, але з плином часу Telegram претерпів значні зміни та вдосконалення. Тепер у співрозмовника будь-якого користувача може виступати не лише людина, а й автоматична програма - бот.

1 ТЕХНОЛОГІЧНИЙ РОЗДІЛ

1.1 Аналіз предметної області

1.1.1 Види чат ботів

Одним із популярних месенджерів, який широко використовується для розробки чат-ботів, є Telegram. Telegram надає розробникам зручне API для створення та інтеграції ботів у платформу. Це відкриває можливості для розроблення різноманітних функціональностей і сервісів із використанням Telegram-ботів.

Інформаційні боти, покупові боти, розважальні боти та боти підтримки клієнтів - це лише кілька видів ботів, які стали невід'ємною частиною нашого цифрового світу. Завдяки швидкому розвитку технологій і штучного інтелекту, ці боти стають все популярнішими і корисними для користувачів(табл. 1.1).

Таблиця 1.1. Типи ботів за функціональністю

За функціональністю	**За способом взаємодії**
Інформаційні боти: надають користувачам інформацію про певну тему	Текстові боти: взаємодіють з користувачами через текстові повідомлення
Покупові боти: забезпечують можливість здійснення покупок через месенджер	Голосові боти: взаємодіють з користувачами за допомогою голосових команд або розпізнавання голосу
Розважальні боти: надають розважальний контент, такий як ігри, загадки, вікторини	Візуальні боти: використовують графічний інтерфейс для взаємодії з користувачем
Підтримки клієнтів: забезпечують підтримку користувачам, відповідають на запитання та допомагають розв'язати проблеми	Гібридні боти: комбінують різні способи взаємодії, такі як текст, голос, зображення тощо

Бот, що розробляється – інформаційний, саме тому, що надає інформацію про кінофільми, повідомляє користувачів про вихід нових серій, тобто надають певну інформацію для користувачів.

1.1.2 Платформа для бота

Телеграм (Telegram) - це популярний месенджер, що надає можливість обміну повідомленнями, аудіо- та відео-контентом, файлами та іншою інформацією. Він був створений у 2013 році Павлом Дуровим і його братом Миколою Дуровим, і відтоді став однією з найпопулярніших платформ для комунікації та обміну інформацією.

1.1.3 Telegram API

Однією з основних переваг чат-ботів у Телеграмі є їхня простота у використанні для звичайних користувачів. Боти можуть бути додані в контакти користувачів або підписані на канали, що дає змогу їм отримувати сповіщення та інформацію від ботів. Взаємодія з ботами відбувається через текстовий інтерфейс, що робить його інтуїтивно зрозумілим і доступним для широкого кола користувачів.

Телеграм-боти, як автоматизовані системи, мають деякі відмінності в порівнянні зі звичайними користувачами у месенджері Telegram. Ось кілька з таких відмінностей:

1. Бот не має можливості показати свою останню активність, адже ця функція в нього відсутня. Зате ви можете бачити його ім’я. Так ми розуміємо, що маємо справу з штучним інтелектом, а не живою людиною.

2. Ботам не виділено безмежного простору на сервері. Вони будуть стерті через певний час після обробки;

3. Користувач повинен дати боту команду, що він бажає розпочати спілкування. Бот не зможе сам знайти вас і почати діалог без вашого дозволу.

4. Назва бота повинна бути написана латинськими літерами та може включати цифри і деякі інші символи, але завжди на кінці має бути «…_bot» або «…Bot» (@WeatherBot).

5. Коли бот додається до бесіди, він не може бачити всі попередні повідомлення, які були відправлені до його приєднання. Бот почне зчитувати

інформацію лише після запиту на діалог та отримання нових повідомлень.

1.1.4 Актуальність

Вибір теми проєкту, пов'язаного з розробкою телеграм-бота для відстеження оновлень серіалів і фільмів, зумовлений низкою чинників і має високу актуальність у сучасному інформаційному суспільстві. У даному розділі буде обґрунтовано обрану тему проєкту та пояснено її актуальність.

Актуальність цього проєкту підтверджується підвищеним інтересом користувачів до серіалів і фільмів, а також необхідністю забезпечити їм зручний і централізований доступ до інформації та оновлень. Такий бот може істотно спростити і поліпшити досвід користувачів, допомагаючи їм бути в курсі останніх новин і оновлень у світі серіалів і фільмів.

Бот має в своєму арсеналі ключові функціональні можливості, такі як:

- Пошук фільмів
- Пошук серіалів
- Виведення постера, опис кінофільму та інформація про дати оновлень
- Підписка на оновлення
- Можливість переглянути підписки
- Видалення з підписок
- Перегляд підписків з детальною інформацією та рекомендаціями на підписки у веб-додатку
- Зміна мови інтерфейсу

Аналоги розроблюваного телеграм-бота для відстеження оновлень серіалів і фільмів існують. У цьому разі ми розглянемо кілька з них, а також їхні переваги та недоліки. Ознайомившись з аналогами, можна отримати корисну інформацію для оптимізації та поліпшення розроблюваного бота.

TrackSeries - ще один популярний веб-сервіс для відстеження серіалів і фільмів. Він пропонує можливість додавати серіали і фільми в список перегляду, отримувати повідомлення про нові випуски і дізнаватися подробиці про твори. Однак, на відміну від телеграм-бота, TrackSeries вимагає доступу до веб-сервісу і

не надає можливості взаємодії через Telegram.

Мабуть найкращим з додатків буде TV Time (рис. 1.1) – це мобільний додаток, який призначений для керування і відстеження вашого кіно- та телевізійного досвіду. Цей додаток надає користувачам зручний спосіб відстежувати та організовувати свої перегляди фільмів і серіалів. Функції та можливості додатку TV Time досить обширні, хоча є свої недоліки: реклама, обмеженість контенту, відсутність синхронізації.

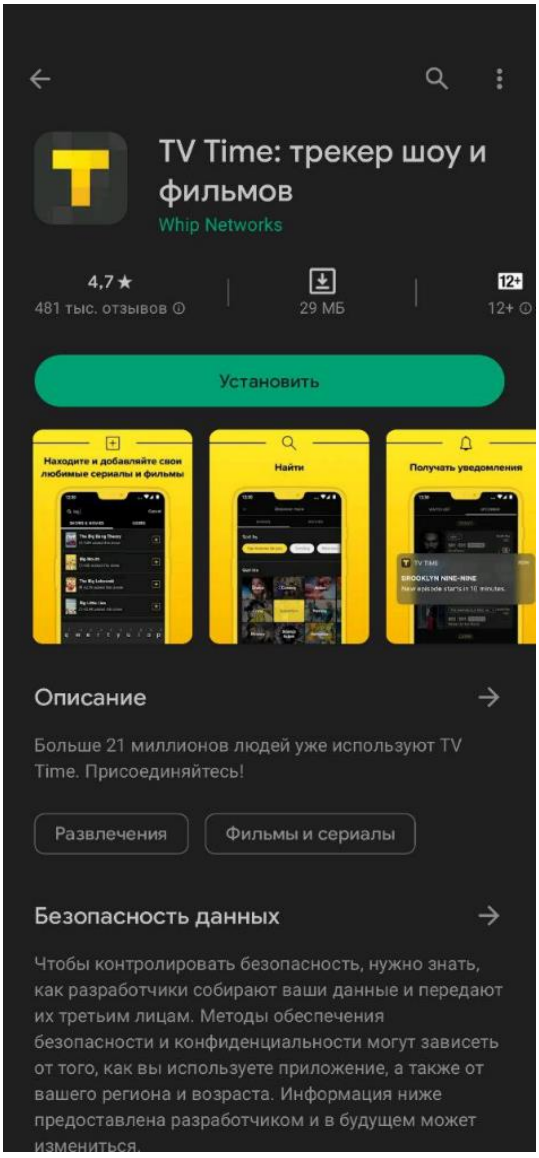


Рисунок 1.1. Додаток TV Time у Google маркет

У телеграм є безліч телеграм ботів (рис. 1.2) з фільмами, деякі навіть дають можливість дивитись фільми безпосередньо у телеграмі, але для цього потрібно завантажити фільм на пристрій. Також у них є досить цікава можливість підібрати випадковий фільм для перегляду, якщо у користувача немає ідей що поглянути.

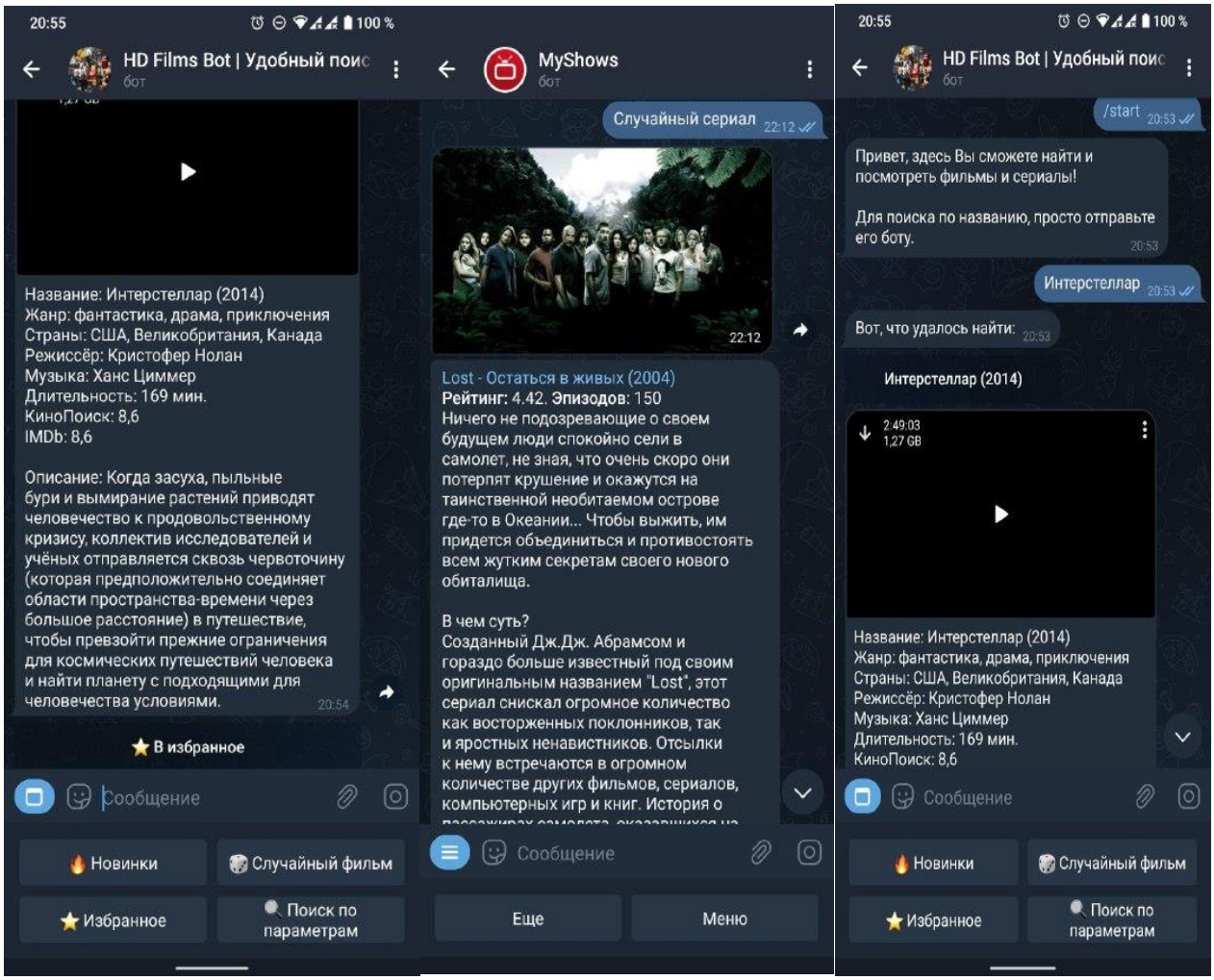


Рисунок 1.2. Аналоги бота

Деякі люди розробили телеграм боти фільмотеки, які можуть по назві фільму знайти фільм для перегляду відразу в телеграмі, або подальшому завантаженню його на пристрій, наприклад у галерею за замовчуванням. Також є боти у яких добавили можливість вибивати випадковий фільм, якщо у користувача немає своїх ідей щодо фільму.

Але такі боти мають досить мало можливостей по пошуку та налаштуванню. Тут як правило не можливо обрати потрібну мову запиту, наприклад - українську Також подивитися рекомендації. Також майже усі боти не мають опції, щоб обрати українську мову для користувачів з України. Такі боти зазвичай не вміють відслідковувати за оновленнями серіалів чи аніме. Більшість таких ботів акцентує увагу на конкретному жанрі чи озвучки. Існує досить якісний і зручний бот -Бот AniLibria (рис. 1.3) Він вміє відслідковувати тільки аніме і тільки однієї російськомовної озвучки від AniLibria, що є значним недоліком.

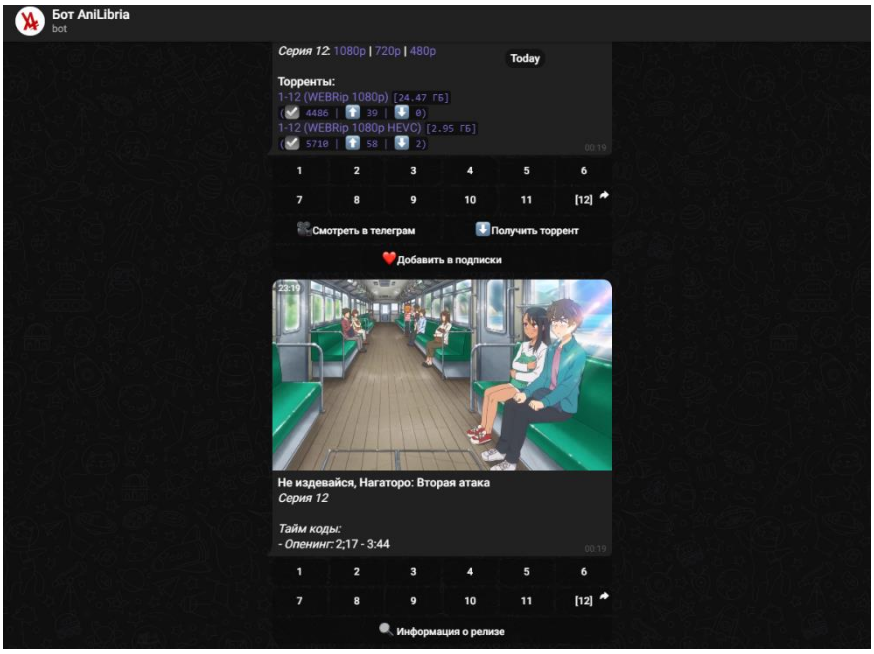


Рисунок 1.3. Anilibria

Також є телеграм бот LostFilm updates(рис. 1.4), який також відслідковує оновлення серіалів, але у нього досить обмежена база контенту і він також не має української мови інтерфейсу.

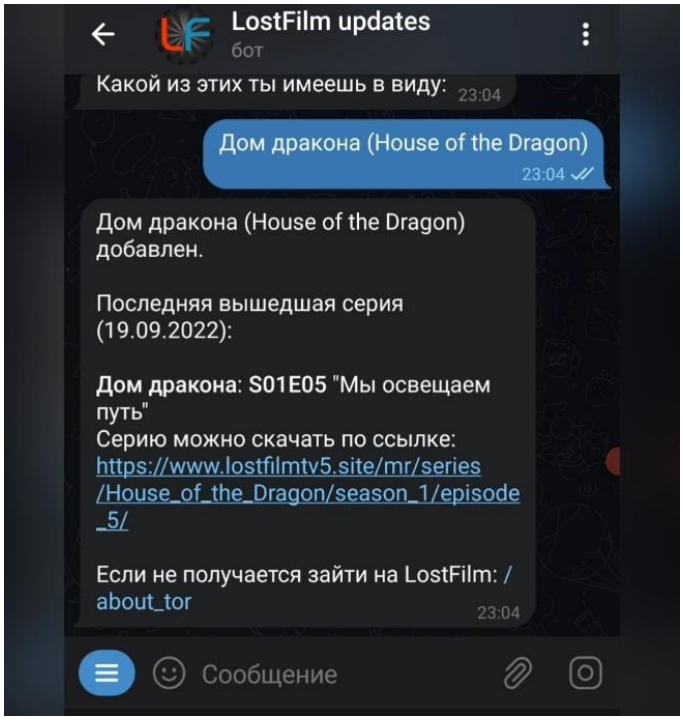


Рисунок 1.4. LostFilm

Загалом телеграм-бот, що розробляється, пропонує зручне рішення для відстеження оновлень серіалів і фільмів через популярну платформу Telegram. Він

має свої переваги, такі як зручність використання та інтеграція з Telegram, а також інтеграція з TMDB API і AniList API для отримання актуальної інформації. Однак, слід враховувати обмеження функціоналу та доступності на різних пристроях.

Телеграм бот працює за допомогою онлайн сервісів, парсингу сторінок та арі. Бот має відслідковувати оновлення на фільмотеках та різних відеохостингах, повідомляти про вихід нових серій серіалів, аніме, та новинок сучасних фільмів.

Переваги розробленого телеграм-бота:

- Зручність використання – Телеграм-бот забезпечує простий та інтуїтивно зрозумілий інтерфейс для взаємодії з користувачем. Користувачі можуть легко підписатися на серіали та фільми, налаштовувати сповіщення та отримувати актуальну інформацію.
- Інтеграція з Telegram – Телеграм-бот використовує Telegram як платформу для обміну повідомленнями. Це дає змогу користувачам отримувати сповіщення та взаємодіяти з ботом без необхідності встановлення додаткових застосунків або відвідування веб-сайтів.
- Використання TMDB API і AniList API – Інтеграція з TMDB API і AniList API дає змогу отримувати актуальну інформацію про серіали, фільми та аніме. API надають доступ до великих баз даних і пропонують різноманітні можливості для відстеження оновлень.

1.2 Розробка алгоритму відстеження оновлень фільмів та серіалів

Алгоритм відстеження оновлень фільмів і серіалів є важливою частиною телеграм-бота і сайту для повідомлення користувачів про нові епізоди. Алгоритм складається з трьох етапів: збір і зберігання даних про фільми та серіали з різних платформ і джерел; запит і аналіз даних про статус фільмів і серіалів, а також про дати виходу нових епізодів або сезонів; визначення змін у статусі та надсилання повідомлень користувачам. Алгоритм дає змогу надавати актуальну та достовірну інформацію про фільми та серіали, а також своєчасно інформувати користувачів про появу оновлень.

1.2.1 Збір і збереження даних про фільми та серіали з різних платформ

Для того, щоб телеграм-бот і сайт могли надавати користувачам актуальну і повну інформацію про фільми і серіали, необхідно збирати і зберігати дані про них з різних платформ. Як основне джерело даних використовується TMDB API - інтернет-сервіс, що надає доступ до бази даних про фільми, телепрограми та комп'ютерні ігри. TMDB API дає змогу отримувати дані про фільми та серіали за їхніми ідентифікаторами, назвами, жанрами, акторами, режисерами та іншими параметрами. Дані містять назву, дату виходу, останній епізод, постер, короткий опис та іншу інформацію.

Для збору та збереження даних про фільми та серіали використовується SQLite Database(рис. 1.5) - легковажна вбудована реляційна база даних. Для збору даних про фільми та серіали використовується функція Request, яка надсилає запит до TMDB API за назвою фільму або серіалу. Функція отримує відповідь у форматі JSON і витягує з нього необхідні дані. Потім функція зберігає дані в таблицю shows бази SQLite Database.

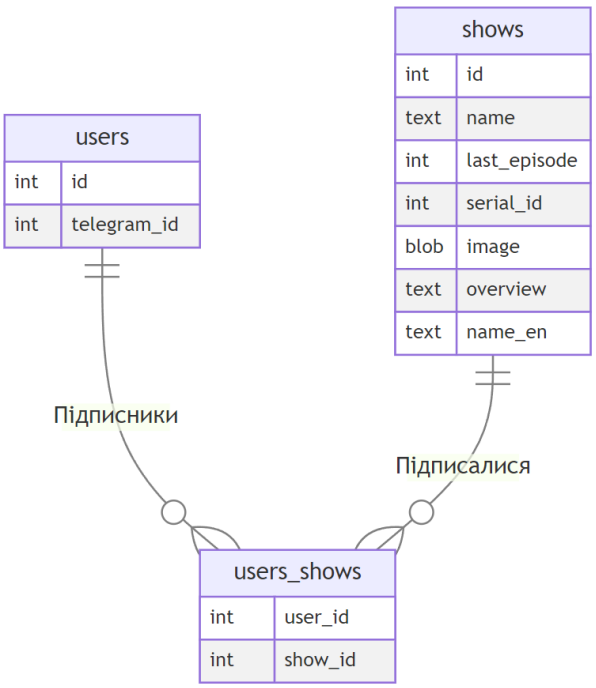


Рисунок 1.5. Структура бази даних бота

Для збереження даних про користувачів і підписки використовується функція update_tracked_series, яка отримує від користувача його telegram_id і

show_id фільму або серіалу, на який він хоче підписатися. Функція перевіряє наявність користувача в таблиці users бази SQLite Database. Якщо користувача немає, то функція додає його в таблицю users. Потім функція додає запис про підписку (з усіма даними, які дістаються за допомогою TMDB API) у таблицю users_shows бази SQLite Database.

Таким чином, збір і збереження даних про фільми та серіали з різних платформ здійснюється за допомогою TMDB API і SQLite Database. Це забезпечує актуальність і повноту інформації для користувачів телеграм-бота і сайту.

Запит даних здійснюється з використанням клієнтської сесії та підключення до бази даних. Потім витягуються необхідні дані, такі як ідентифікатори користувачів, серіалів і останні епізоди. Після цього код виконує запит до API TMDB для отримання актуальної інформації про останній епізод серіалу.

Аналіз даних здійснюється шляхом порівняння отриманого номера останнього епізоду з номером останнього епізоду в базі даних. Якщо номери не збігаються, код оновлює значення останнього епізоду в базі даних і надсилає повідомлення користувачам.

Таким чином, функція дає змогу автоматично відстежувати й оновлювати інформацію про статус фільмів і серіалів, забезпечуючи користувачів актуальною інформацією про нові епізоди та релізи.

1.2.2 Визначення змін у статусі та надсилання повідомлень користувачам

Запит даних здійснюється з використанням клієнтської сесії та підключення до бази даних(рис. 1.7). Потім витягуються необхідні дані, такі як ідентифікатори користувачів, серіалів і останні епізоди. Після цього код виконує запит до API TMDB для отримання актуальної інформації про останній епізод серіалу.

Аналіз даних здійснюється шляхом порівняння отриманого номера останнього епізоду з номером останнього епізоду в базі даних. Якщо номери не збігаються, код оновлює значення останнього епізоду в базі даних і надсилає повідомлення користувачам.

Таким чином, функція дає змогу автоматично відстежувати й оновлювати

інформацію про статус фільмів і серіалів, забезпечуючи користувачів актуальною інформацією про нові епізоди та релізи.

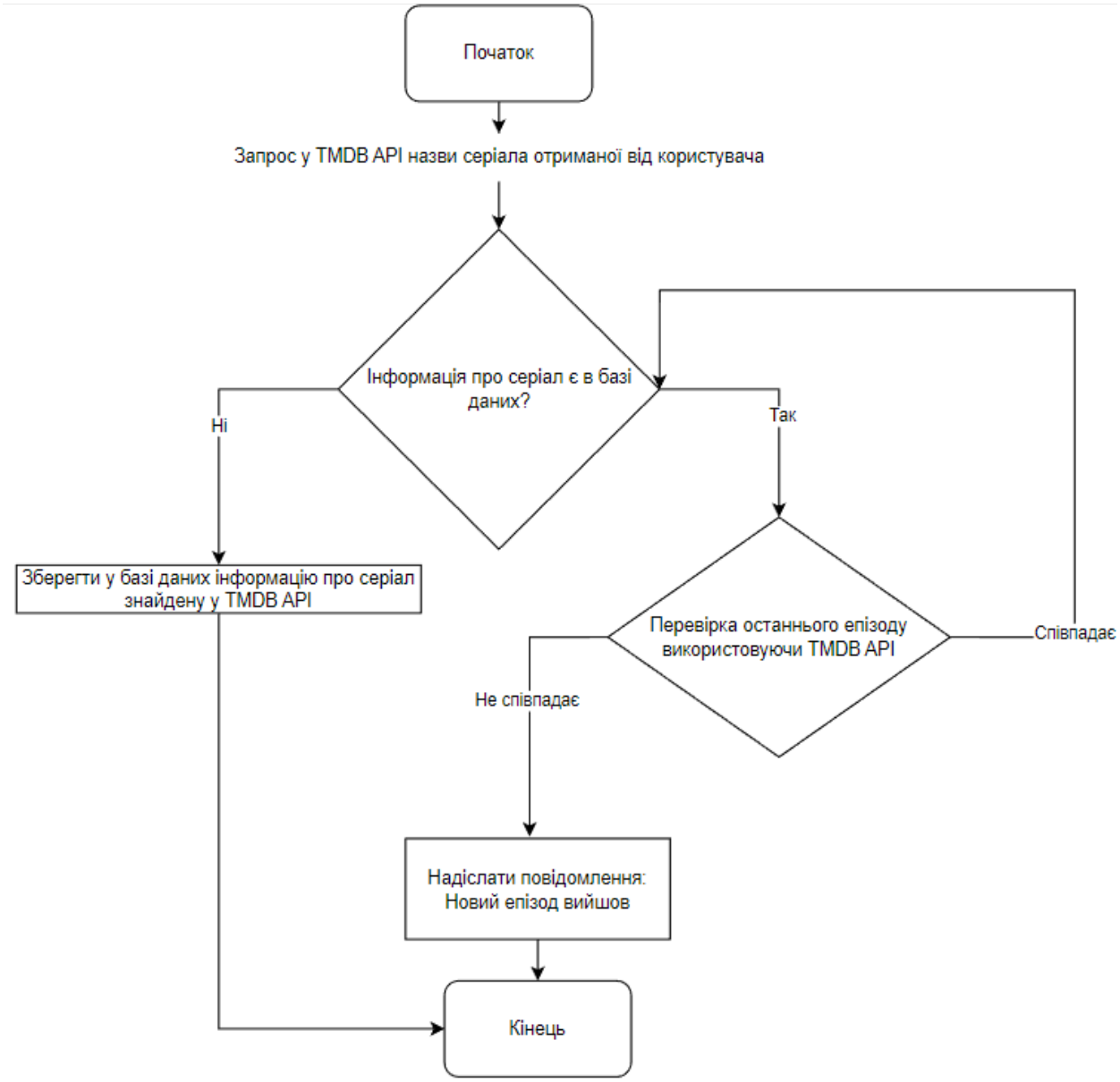


Рисунок 1.6. Блок-схема роботи системи сповіщення

1.2.3 Надання користувачу інформації про підписки

Функція get_tracked_series(user_id) призначена для отримання відстежуваних серіалів користувача з бази даних і повернення відповідного повідомлення. Після встановлення з'єднання з базою даних, створюється курсор, який виконує SQL-запит до бази даних. Запит SELECT вибирає ідентифікатор та назву серіалу з таблиці shows, які відповідають користувачеві з вказаним user_id. Запит також використовує з'єднання через таблиці users_shows і users для зв'язку

серіалів з користувачем за їх ідентифікаторами. Результат запиту зберігається в змінній result.

Якщо result містить дані, тобто відстежувані серіали знайдено, то формується повідомлення message на основі мови (lang) і додаються рядки з ідентифікатором і назвою серіалу для кожного знайденого запису. Кінцеве повідомлення повертається з функції.

Якщо result порожній, тобто немає відстежуваних серіалів, то функція повертає повідомлення "havent" із відповідного словника повідомлень (messages) залежно від мови (lang).

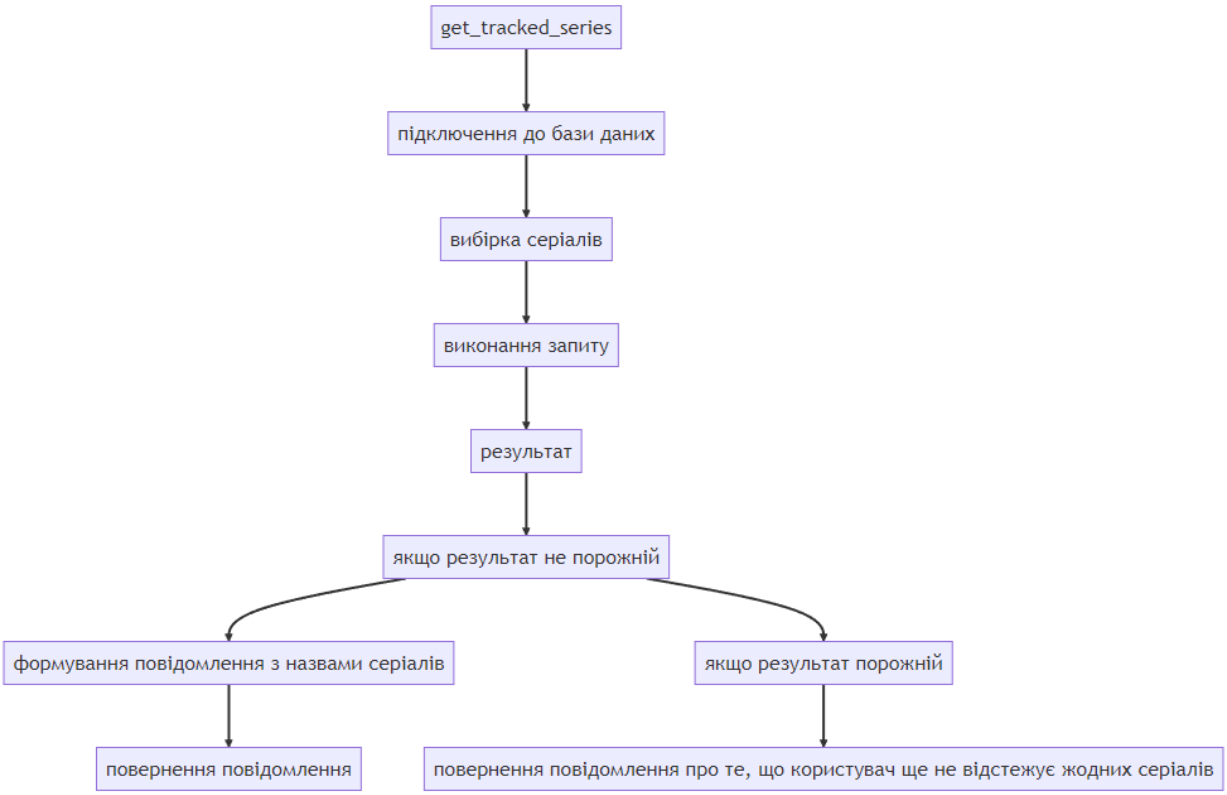


Рисунок 1.7. Блок-схема роботи списку підписок

1.2.4 Функціональні можливості

Після проведення всебічного аналізу проблем і визначення основних потреб користувачів, ми розробили діаграму прецедентів (рис. 1.8). Цей інструмент необхідний для чіткого визначення функціональних можливостей, які ми повинні надати як користувачам, так і адміністратору.

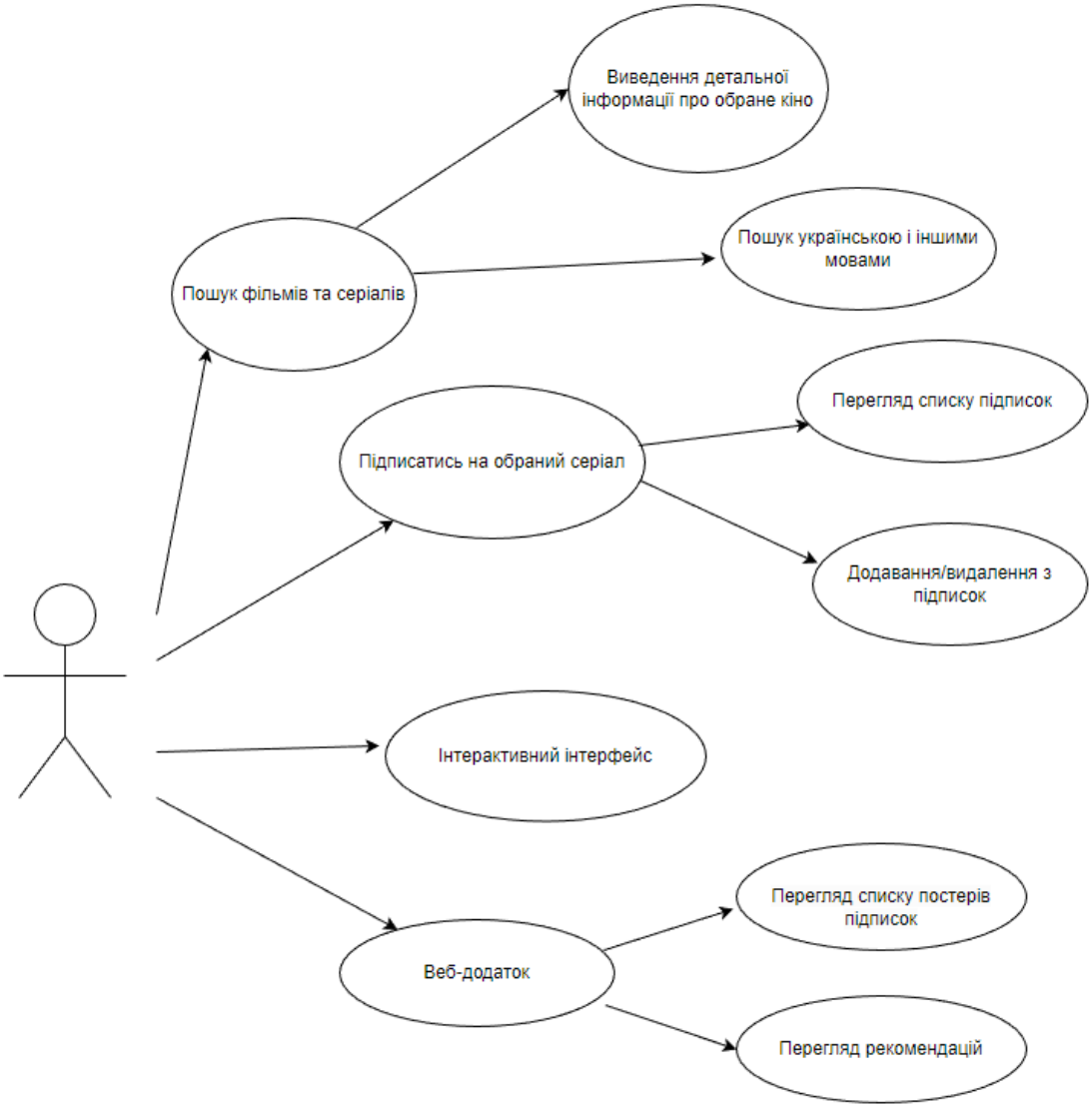


Рисунок 1.8. Діаграма прецедентів функціональних можливостей

1.3 Розробка сайту та бота

1.3.1 Визначення середовища розробки

VS Code (Visual Studio Code) є одним з найпопулярніших та широко використовуваних редакторів коду у світі програмування. Його популярність пояснюється низкою функцій, продуктивністю та високою кастомізацією, що дозволяє задовольнити потреби багатьох програмістів. У цьому тексті розглянемо основні причини, чому VS Code є більш універсальним та швидким в порівнянні з іншими редакторами коду і чому він підходить для більшості програмістів.

VS Code підтримує багато мов програмування та платформ, що робить його універсальним і гнучким інструментом для більшості програмістів. Він надає

можливість працювати з такими мовами, як JavaScript, Python, C++, Java, HTML/CSS, Ruby, PHP та багато інших. Незалежно від того, яку мову програмування використовувати, VS Code зможе задовольнити ваші потреби.

VS Code відомий своєю високою продуктивністю та швидкодією. Він запускається швидко, і його робота з проектами різного розміру є плавною і без затримок. Це особливо важливо при роботі з великими проектами або при використанні ресурсоємних мов програмування. Завдяки оптимізованому ядру та ефективному використанню ресурсів, VS Code забезпечує швидкість редактора, що дозволяє програмістам зосередитися на своїй роботі без зайвих перешкод.

Враховуючи універсальність, швидкість, розширюваність та інтеграцію, Visual Studio Code (VS Code) вважається більш універсальним та швидким інструментом порівняно з іншими і буде більш пріоритетним вибором для розобки.

1.3.2 Визначення засобів розробки

Для реалізації дипломного проєкту було обрано мову Python, оскільки вона має такі переваги:

- Простий і лаконічний синтаксис, який спрощує написання і читання коду.
- Багата стандартна бібліотека і безліч сторонніх модулів, які розширюють можливості мови.
- Переносимість коду між різними операційними системами і платформами.
- Підтримка багатопотокових обчислень і обробки виключень.
- Можливість використовувати інтерполяцію рядків і зрізи для зручної роботи з текстовими даними.
- Наявність потужних фреймворків для веб-розробки, таких як Django та Flask, що дає змогу створювати сайти з використанням шаблонів, баз даних і системи аутентифікації.

Бібліотека Telebot є однією з популярних бібліотек для розробки телеграм ботів на мові програмування Python. Вона надає зручні та прості засоби для взаємодії з API Telegram і дозволяє легко створювати та налаштовувати ботів з різноманітною функціональністю.

Широкий спектр документації: Telebot має добре документовану API та багато прикладів використання. Розробники можуть легко знайти документацію, приклади та поради щодо використання цієї бібліотеки. Крім того, існує активна спільнота користувачів, яка готова надати допомогу та відповісти на питання.

Бібліотека Django є одним з найпопулярніших фреймворків для розробки веб-додатків на мові програмування Python. Вона була створена в 2003 році і швидко набула широкого розповсюдження завдяки своїй потужності, гнучкості і зручності використання.

Django надає повноцінну платформу для розробки веб-додатків, яка включає в себе все необхідне для створення високоякісних проектів. Вона базується на концепції моделі-представлення-контролера (Model-View-Controller, MVC), яка дозволяє розділити логіку додатку на окремі компоненти і полегшує розробку і підтримку проекту.

Django має вбудовану адміністративну панель (рис. 1.9), яка автоматично створюється на основі моделей додатку. Ця панель надає готові інструменти для додавання, редагування і видалення даних з бази даних, що дозволяє адміністраторам легко керувати контентом своєї програми без необхідності писати додатковий код.

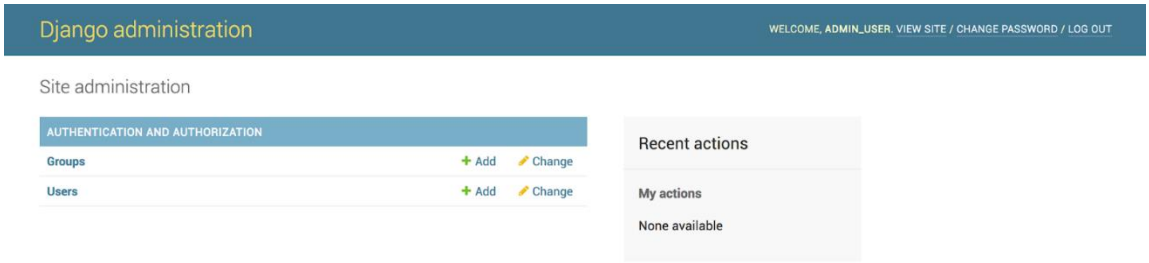


Рисунок 1.9. Адміністративна панель Django

Для розробки веб-додатків Django використовує патерн проектування Model-View-Template (MVT), який є варіацією класичного MVC. Моделі в Django представляють схему бази даних і методи для роботи з даними, Представлення (Views) відповідають за обробку запитів і відображення даних, а Шаблони

(Templates) використовуються для оформлення веб-сторінок.

Django також надає засоби для роботи з URL-ами, формами, файлами, кешуванням, міжнародизацією та багатьма іншими функціями, що полегшують розробку веб-додатків. Крім того, Django має активну спільноту розробників, яка постійно вносить внески в розвиток фреймворку і надає багато корисних пакетів і розширень.

Отже зазначимо основні переваги Django:

1. Швидкість розробки – Django надає готовий набір компонентів та інструментів, що дозволяють розробникам прискорити процес розробки.
2. Масштабованість – Django підтримує масштабування веб-додатків на будь-якому рівні.
3. Безпека – Django має вбудовану систему безпеки, що допомагає уникати багатьох типових атак, таких як внедрення кодування, підбір паролів, міжсайтовий скриптінг тощо.
4. Розширюваність –- Django підтримує модульну архітектуру, що дозволяє розширювати функціональність додатку за допомогою сторонніх розширень і плагінів.
5. Документація та спільнота – Django має добре структуровану та докладну документацію, яка допомагає розробникам ознайомитися з фреймворком та використовувати його ефективно.

У підсумку, Django є потужним фреймворком для розробки веб-додатків, який надає широкі можливості для швидкої і ефективної розробки. Його зручний синтаксис, велика кількість вбудованих функцій і підтримка спільноти роблять Django ідеальним вибором для професійних розробників і початківців в галузі веб-розробки.

TMDB (The Movie Database) - це велика база даних, що зберігає інформацію про фільми, серіали та акторів. TMDB API надає розробникам доступ до цієї бази даних і дозволяє отримувати різноманітну інформацію про фільми, включаючи назви, рейтинги, описи, постери, трейлери та багато іншого.

TMDB API є потужним і зручним інструментом для отримання даних про

фільми та серіали. Велика кількість доступної інформації, простий інтерфейс, підтримка трейлерів та афіш, а також безкоштовний доступ роблять TMDB API популярним серед розробників. Варто враховувати обмеження даних та API використання, а також залежність від зовнішнього сервісу. При розробці додатків, які використовують TMDB API, необхідно забезпечувати альтернативні шляхи отримання даних і контролювати використання запитів, щоб уникнути обмежень.

AniList API є інтерфейсом, який надає доступ до функціональності та даних AniList платформи для розробників. Цей API дозволяє розробникам створювати додатки, програми або інтеграції, що використовують дані AniList.

За допомогою AniList API розробники можуть отримувати доступ до різноманітних функцій, таких як отримання списків аніме, деталей про серіали, оцінок користувачів, списків бажаного та багато іншого. Це надає можливості для створення додатків, які інтегруються з AniList та надають користувачам додаткові функції, наприклад, автоматичне оновлення списків перегляду аніме або отримання рекомендацій на основі їхніх уподобань.

AniList API використовується широкою спільнотою розробників, а також слугує основою для створення різноманітних додатків та сервісів, пов'язаних з аніме та манґо. За допомогою цього API розробники можуть легко отримати доступ до даних AniList та інтегрувати їх у свої проекти.

SQL (Structured Query Language) є широко використовуваною мовою програмування для керування реляційними базами даних. Вона надає набір інструкцій для створення, зміни та управління даними в базах даних, а також для виконання запитів та аналізу даних. Однією з найпопулярніших реляційних баз даних, яка підтримує SQL, є SQLite.

SQLite - це компактна, вбудована база даних, яка є самодостатньою і не потребує окремого сервера для роботи. Вона відома своєю простотою, надійністю та ефективністю. SQLite широко застосовується в мобільних додатках, вбудованих системах, веб-браузерах та інших місцях, де потрібна компактна і швидка база даних.

SQLite пропонує низку переваг для розробників. Він є крос-платформеним і

може працювати на різних операційних системах без необхідності внесення змін до коду. SQLite також не потребує складного налаштування або адміністрування, що робить його зручним для використання в простих проєктах або на пристроях з обмеженими ресурсами.

Однак, незважаючи на всі свої переваги, SQLite також має деякі обмеження. Він не призначений для роботи з величезними базами даних або високонавантаженими системами, оскільки може мати проблеми з продуктивністю за великих обсягів даних і одночасних записів. SQLite також не підтримує деякі просунуті функції, які можуть бути доступні в інших реляційних базах даних.

SQLite є чудовим вибором для невеликих і середніх проєктів, які потребують простоти, компактності та низьких вимог до ресурсів. Він забезпечує надійне зберігання даних і хорошу підтримку SQL, хоча є деякі обмеження на масштабованість і функціональні можливості. Під час вибору між SQLite та іншими базами даних важливо враховувати конкретні вимоги проекту та очікування від бази даних.

1.3.3 Створення чат-боту

Основним завданням, яке стоїть при розробці цього програмного продукту, було вирішення проблеми зручного пошуку фільмів, серіалів та аніме, можливість їх усі відслідковувати, зробити цей процес максимально зручним та простим. Програмний продукт пропонує зручний спосіб відслідковувати оновлення нових фільмів і серіалів, отримувати сповіщення про вихід нових серій та забезпечує користувачам зручну інтерактивну платформу для занурення яскравий світ кіношоу.

Основні завдання та функції, які виконує програмний продукт, включають наступне:

1. Відстеження оновлень – Основна задача телеграм-боту полягає у відстеженні оновлень нових серій фільмів та серіалів. Він періодично перевіряє наявність нових епізодів у визначених серіалах і отримує актуальну інформацію про вихід нових серій з відповідного джерела.

2. Сповіщення користувачів – Телеграм-бот надсилає сповіщення

користувачам, які відстежують певні серіали, коли виходить нова серія. Це дозволяє користувачам бути в курсі останніх оновлень і не пропустити нові епізоди їх улюблених серіалів.

3. Система рекомендацій – Телеграм-бот також має веб-сайт, на якому користувачі можуть переглянути список серіалів, які вони відстежують. Вони можуть додавати нові серіали до свого списку або видаляти вже відслідковані. Крім того, на сайті є рекомендації щодо нових серій та фільмів на підставі інтересів користувача.

4. Можливість перегляду деталей – Телеграм-бот надає можливість користувачам переглянути деталі про кожну серіал або фільм, такі як назва, опис, остання серія, дата виходу наступної серії.. Це дозволяє користувачам знайти більше інформації про відстежувані ним фільми та серіали.

5. Підтримка інтерактивності – Телеграм-бот може надавати інтерактивні функції, такі як кнопки для вибору конкретного серіалу або зміна мови інтерфейсу. Це сприяє зручному та ефективному взаємодії користувачів з ботом.

@BotFather є спеціальним помічником у месенджері, який допомагає створювати і налаштовувати власних ботів, а також керувати ними:

1. Для початку введемо команду /newbot у діалозі з BotFather. Вона допоможе нам створити нового бота.

2. BotFather запросить ім'я для бота.

3. Потім BotFather попросить обрати унікальне ім'я користувача для нашого бота. Ім'я бота має закінчуватися на "bot" (наприклад, AbobaBot або SuperBot123). Придумаємо ім'я користувача (бота) і введемо його, наприклад Animuvie_bot.

4. Після успішного створення бота, BotFather видасть токен. Це важливий код, який потрібно зберегти, оскільки він буде використовуватися для взаємодії з ботом через API.

5. Тепер бот зареєстрований! Можна налаштувати його, додати команди, функції та навчити його відповідати на повідомлення.

У BotFather є багато команд для налаштування, але ми розглянемо лише деякі з них, які корисні та потрібні нам для розробки:

1. /mybots - Ця команда дає змогу переглянути список усіх наших ботів. Ми можемо побачити інформацію про кожного бота, таку як ім'я, username і токен.

2. /setname - За допомогою цієї команди ми можемо змінити ім'я нашого бота. Просто дотримуємося інструкцій BotFather і вводимо нове ім'я.

3. /setdescription - Ця команда дає змогу встановити опис для нашого бота. Опис відображатиметься під час пошуку бота.

4. /setabouttext - За допомогою цієї команди можна встановити додатковий текст про бота. Цей текст також буде відображатися під час пошуку бота.

5. /setuserpic- Ця команда дає змогу встановити зображення профілю для бота. Досить надіслати йому зображення і слідувати інструкціям BotFather.

6. /setcommands- За допомогою цієї команди ми можемо налаштувати список команд, які відображатимуться під полем введення бота в Telegram.

7. /deletebot – Якщо необхідно видалити свого бота, можна використовувати цю команду.

Після завершення цих кроків, BotFather повідомить нас про успішне створення нашого бота. Щоб навчити його виконувати певні дії, необхідно написати відповідний код програми.

Розглянемо основні кроки розробки функцій бота. Першим кроком в розробці бота є імпорт необхідних модулів. Спочатку імпортуємо досить важливі модулі(рис. 1.10), без яких неможлива робота бота.

```
import os
import base64
from io import BytesIO
from PIL import Image
import telebot
from telebot import types
from telebot.types import InlineKeyboardMarkup, InlineKeyboardButton
import pars
from wsgiref import headers
import requests
from bs4 import BeautifulSoup as BS
import sqlite3
import asyncio
import aiosqlite
import aiohttp
import threading
```

Рисунок 1.10. Імпортовані модулі

Модуль `os` надає функції для взаємодії з операційною системою. Можна використовувати його для роботи з файлами і папками, керування шляхами, запуску зовнішніх програм і багато чого іншого.

Модуль `base64` надає функції для кодування та декодування даних у форматі Base64. Можна використовувати його для перетворення даних у рядок, який можна безпечно передавати або зберігати.

Модуль `io` надає класи для роботи з введенням-виведенням (I/O). Підмодуль `BytesIO` надає можливість роботи з даними в пам'яті як з файлом у байтовому поданні.

Модуль `PIL` (Python Imaging Library) надає функціональність для роботи із зображеннями. Підмодуль `Image` дає змогу відкривати, обробляти та зберігати зображення в різних форматах.

Модуль `telebot` надає API для взаємодії з Telegram Bot API. Він полегшує створення та налаштування ботів, а також обробку вхідних і вихідних повідомлень.

Модуль `requests` надає простий і зручний інтерфейс для надсилання HTTP-запитів і отримання відповідей. Він часто використовується для взаємодії з веб-серверами та отримання даних з інтернету.

Модуль `bs4` (або BeautifulSoup) надає функціональність для розбору HTML-коду веб-сторінки. Він дає змогу витягувати дані з HTML-документів і виконувати різні операції з елементами веб-сторінки.

Модуль `sqlite3` надає функціональність для роботи з базами даних SQLite. Ви можете використовувати його для створення, підключення, виконання запитів і обробки даних у SQLite базі даних.

Модуль `asyncio` надає можливості для асинхронного програмування в Python. Він дає змогу створювати асинхронні задачі, корутини та керувати подіями і планувальником подій.

Модуль `aiosqlite` надає асинхронний інтерфейс для роботи з базами даних SQLite. Він дає змогу виконувати асинхронні запити та маніпулювати даними в SQLite базі даних.

Модуль `aiohttp` надає можливості для асинхронної HTTP-взаємодії. Він

дозволяє виконувати асинхронні запити до веб-серверів і обробляти отримані відповіді.

Модуль `threading` надає функціональність для створення та керування потоками виконання в Python. Він дає змогу виконувати кілька операцій паралельно та координувати їхнє виконання.

Імпорт модулів дозволяє мати доступ до їх функціональності та використовувати їх у подальшій розробці функцій бота.

Наступним кроком створюємо базу данних за допомогою моделей Django, а також використовуючи мову SQL (рис. 1.11).

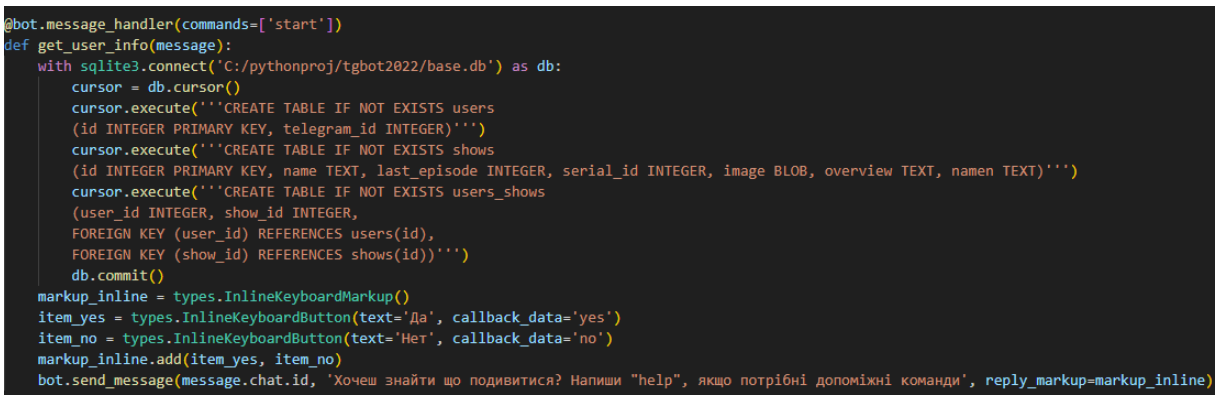


Рисунок 1.11. База даних

Потрібно підключити токен до боту, для збереження токену і інших важливих даних, його зберігаємо в окремий файл: “bot = telebot.TeleBot(token)”. Використовуємо модуль `sqlite3` для роботи з базою даних SQLite. У цьому випадку створюється файл бази даних `base.db` у зазначеному шляху `'C:/pythonproj/tgbot2022/base.db`. Далі виконуються запити на створення трьох таблиць: `users`, `shows` і `users_shows`.

таблиця `users`:

– `id` (INTEGER PRIMARY KEY): Унікальний ідентифікатор користувача.
– telegram_id` (INTEGER): Ідентифікатор користувача в Telegram.

таблиця `shows`:

– `id` (INTEGER PRIMARY KEY): Унікальний ідентифікатор шоу.
– name` (TEXT): Назва шоу.
– last_episode` (INTEGER): Номер останнього епізоду.

– serial_id` (INTEGER): Ідентифікатор шоу в сервісі.
– image` (BLOB): Зображення шоу в бінарному форматі.
– overview` (TEXT): Опис шоу.
– `namen` (TEXT): Додаткова назва шоу на англійській мові.

таблиця `users_shows`:

– user_id` (INTEGER): Ідентифікатор користувача.
– show_id` (INTEGER): Ідентифікатор шоу.

Таблиця `users_shows` встановлює зв'язок між користувачами та шоу. Вона використовується для зберігання інформації про те, які користувачі підписані на які шоу. Виклик методу `db.commit()` зберігає зміни в базі даних після виконання всіх запитів. Цей код також створює базу даних але за допомогою моделей Django у файлі models.py (рис. 1.12).

```python
from django.db import models

class User(models.Model):
    telegram_id = models.IntegerField()

    class Meta:
        db_table = 'users'

class Show(models.Model):
    name = models.TextField()
    last_episode = models.IntegerField()
    serial_id = models.IntegerField()
    image = models.BinaryField()
    overview = models.TextField()
    namen = models.TextField()

    class Meta:
        db_table = 'shows'

class UserShow(models.Model):
    user = models.ForeignKey(User, on_delete=models.CASCADE)
    show = models.ForeignKey(Show, on_delete=models.CASCADE)

    class Meta:
        db_table = 'users_shows'
```

Рисунок 1.12. База даних фільмів та серіалів, що відстежує користувач у Django

Створюємо клавіатуру(кнопки інтерфейсу) за допомогою бібліотеки Telebot (рис. 1.13).

```python
def buttons(lang):
    markup_reply = types.ReplyKeyboardMarkup(resize_keyboard= True)
    item_film = types.KeyboardButton('🎞' + messages[lang]['film'])
    item_username = types.KeyboardButton('📺' + messages[lang]['serial'])
    item_list = types.KeyboardButton('📜' + messages[lang]['subscriptions'])
    item_follow = types.KeyboardButton('🔔' + messages[lang]['subscribe'])
    item_id = types.KeyboardButton('🆔' + messages[lang]['telegram_id'])
    item_delete = types.KeyboardButton('🗑')
    item_language = types.KeyboardButton('🌐')
    markup_reply.add(item_film, item_username, item_list, item_id, item_delete, item_language, item_follow)
    return markup_reply
```

Рисунок 1.13. Клавіатура інтерфейса бота

Створюємо команди:

– /start – команда завдяки якої користувач запускає бот.

– /help – команда яка надає інструкції користувачу для роботи з ботом.

Також реалізуємо функцію для зміни мови інтерфейсу (рис. 1.14), для цього ми використовуємо словник ‘messages’ (рис. 1.15).

```
def language_buttons():
    markup_reply = types.ReplyKeyboardMarkup(resize_keyboard=True)
    item_ukrainian = types.KeyboardButton('українська мова')
    item_russian = types.KeyboardButton('русский язык')
    item_english = types.KeyboardButton('english')
    markup_reply.add(item_ukrainian, item_russian, item_english)
    return markup_reply
```

Рис. 1.14 – Функція зміни мови інтерфейсу

```
messages = {
    'ru-RU': {'film': 'Фильм',
    'serial': 'сериал/аниме',
    'subscriptions': 'Мои подписки',
    'subscribe': 'Подписаться',
    'telegram_id': 'Мой телеграмм id',
    'name': 'Название',
    'desc': 'Описание',
    'ls': 'Последняя серия',
    'next': 'Следующая',
    'seria': 'серия',
    'data': 'сезона выходит',
    },

    'uk-UA': {'film': 'Фільм',
    'serial': 'серіал/аніме',
    'subscriptions': 'Мої підписки',
    'subscribe': 'Підписатися',
    'telegram_id': 'Мій телеграм id',
    'name': 'Назва',
    'desc': 'Опис',
    'ls': 'Остання серія',
    'next': 'Наступна',
    'seria': 'серія',
    'data': 'сезона виходить',
    },

    'en-EN': {'film': 'film',
    'serial': 'serial',
    'subscriptions': 'subscriptions',
    'subscribe': 'subscribe',
    'telegram_id': 'My telegram_id',
    'name': 'Name',
    'desc': 'Description',
    'ls': 'Last Seria',
    'next': 'Next',
    'seria': 'seria',
    'data': 'coming out',
    }
}
```

Рисунок 1.15. Словник мови інтерфейсу

Функція, з якої починається робота з ботом (рис. 1.16).

```python
@bot.message_handler(commands=['start'])
def get_user_info(message):
    with sqlite3.connect('C:/pythonproj/tgbot2022/base.db') as db:
        cursor = db.cursor()
        cursor.execute('''CREATE TABLE IF NOT EXISTS users
        (id INTEGER PRIMARY KEY, telegram_id INTEGER)''')
        cursor.execute('''CREATE TABLE IF NOT EXISTS shows
        (id INTEGER PRIMARY KEY, name TEXT, last_episode INTEGER, serial_id INTEGER, image BLOB, overview TEXT, namen TEXT)''')
        cursor.execute('''CREATE TABLE IF NOT EXISTS users_shows
        (user_id INTEGER, show_id INTEGER,
        FOREIGN KEY (user_id) REFERENCES users(id),
        FOREIGN KEY (show_id) REFERENCES shows(id))''')
        db.commit()
    markup_inline = types.InlineKeyboardMarkup()
    item_yes = types.InlineKeyboardButton(text='Да', callback_data='yes')
    item_no = types.InlineKeyboardButton(text='Нет', callback_data='no')
    markup_inline.add(item_yes, item_no)
    bot.send_message(message.chat.id, 'Хочеш знайти що подивитися? Напиши "help", якщо потрібні допоміжні команди', reply_markup=markup_inline)
```

Рисунок 1.16. Функція запуску бота

Створення команди /start, ця команда запускає наш бот.

Функція def fio_step(message) обробляє запити від користувачів бота та запускає необхідні нам функції залежно від тексту повідомлень (рис. 1.17).

```python
@bot.message_handler(content_types= ['text'])
def fio_step(message):
    global lang
    user_id = message.from_user.id
    if '✍' in message.text:
        bot.send_message(message.chat.id, messages[lang]["enter_the_name"]) #вводимо назву
        bot.register_next_step_handler(message, name)
    elif '📺' in message.text:
        bot.send_message(message.chat.id, messages[lang]["enter_the_name"]) #вводимо назву
        bot.register_next_step_handler(message, name_1)
    elif '📋' in message.text:
        bot.send_message(message.chat.id, get_tracked_series(user_id))
    elif '🔔' in message.text:
        try:
            bot.send_message(message.chat.id, update_tracked_series(user_id, results_1[0], results_1[4], series_id, results_1[8], results_1[1], results_1[9], results_1[10]))
        except IndexError:
            bot.send_message(message.chat.id, messages[lang]["a"]) #Вибачте, це кіно не знайдено😔
        except NameError:
            bot.send_message(message.chat.id, messages[lang]["b"]) ##Вибачте, але оновлення фільмів неможливо відслідковувати😯
    elif '🗑' in message.text:
        bot.send_message(message.chat.id, messages[lang]["delete"])
        bot.register_next_step_handler(message, lambda msg: bot.send_message(user_id, text=remove_tracked_series(user_id, int(msg.text))) if msg.text.isdigit() else bot.send_message(me
    elif '🆔' in message.text:
        bot.send_message(message.chat.id, user_id)
    elif '🌐' in message.text:
        bot.send_message(message.chat.id, messages[lang]["lang"], reply_markup=language_buttons())
    elif message.text.lower() == 'русский язык':
        lang = "ru-RU"
        bot.send_message(message.chat.id, "выбран русский язык", reply_markup=buttons(lang))
    elif message.text.lower() == 'українська мова':
        lang = "uk-UA"
        bot.send_message(message.chat.id, "обрано українську мову", reply_markup=buttons(lang))
    elif message.text.lower() == 'english':
        lang = "en-EN"
        bot.send_message(message.chat.id, "English is selected", reply_markup=buttons(lang))
```

Рисунок 1.17. Функція обробки запитів

Наприклад, elif 'емоджи' in message.text: bot.send_message(message.chat.id, get_tracked_series(user_id)) - якщо користувач відправить це емоджі, бот запустить функцію get_tracked_series і передасть їй значення змінної ‘user_id’, яка містить в собі id користувача: user_id = message.from_user.id.

Напишемо основні функції які потрібні для роботи бота, ось саме вони:

1. update_tracked_series - ця функція спрацьовує коли користувач обравши серіал, натискає на кнопку ‘Підписатись’, вона оновлює базу даних додаючи інформацію про серіал яка потрібна для відстеження оновлень цього серіалу. Саме ця функція додає дані про серіал у базу даних, коли користувач підписується на його оновлення.
2. get_tracked_series - ця функція спрацьовує коли користувач натискає на кнопку - емоджі у вигляді списку і виводить підписки користувача.
3. remove_tracked_series - ця функція спрацьовує коли користувач натискає на кнопку - емоджі у вигляді корзини, тоді бот дає можливість ввести номер у списку підписок і тоді обраний серіал буде видалений з підписаних.
4. track_new_episodes - ця функція починає працювати після того як користувач підпишиться на оновлення серіалу. Задача цієї функції відслідковувати оновлення серіалів, вона перевіряє за допомогою TMDB API, чи співпадає остання серія серіалу в базі даних з даними в інтернеті на сервері TMDB, якщо ні, то відправляє повідомлення користувачу який підписаний на цей серіал, що вийшла нова серія.
5. Film та Serial (рис. 1.19)- ці функції розпочинає свою роботу після натискання кнопки ‘serial/anime’, відправляється повідомлення користувачу, щоб той ввів назву серіалу, тоді спрацьовує функція name_ (рис. 1.18)., яка у свою чергу, яка чекає назву від користувача і зберігає її у змінну, яку передає до функції Serial (рис. 1.20), яка робить запит через TMDB API і збирає необхідну інформацію про серіал та виводить її у чат користувачу. Також видно функцію run_async_function_in_thread(loop), ця функція запускає асинхронну функцію у потоці. Вона встановлює цикл подій asyncio і запускає його до тих пір, поки не буде завершена функція track_new_episodes().

   Цикл подій asyncio - це основний компонент бібліотеки asyncio. Він використовується для обробки подій в асинхронному коді. Цикл подій обробляє події, такі як введення / виведення та таймери, і запускає функції зворотного виклику, коли стануть доступними нові дані або коли таймер

закінчиться. asyncio.set_event_loop(loop) - це функція, яка встановлює цикл подій asyncio для поточного потоку.

```python
def name(message):
    try:
        global film_name
        film_name = str(message.text.lower())
        results = Film(film_name)
        bot.send_photo(message.chat.id, photo=results[2], caption=f'{messages[lang]["name"]}: {results[0]}\n{messages[lang]["desc"]}: {results[1]}')
    except IndexError:
        bot.send_message(message.chat.id, messages[lang]["a"]) #Вибачте, це кіно не знайдено
    except NameError:
        bot.send_message(message.chat.id, messages[lang]["b"]) #Вибачте, але оновлення фільмів неможливо відслідковувати


def name_1(message):
    try:
        global series_name
        global results_1
        series_name = str(message.text.lower())
        results_1 = Serial(series_name)
        if results_1[5] == "не найдено":
            bot.send_photo(message.chat.id, photo=results_1[2], caption=f'{messages[lang]["name"]}: {results_1[0]}\n{messages[lang]["desc"]}: {results_1[1]}\n {messages[lang]["ls"]} {results_1[3]}: {results_1[4]}')
        else:
            bot.send_photo(message.chat.id, photo=results_1[2], caption=f'{messages[lang]["name"]}: {results_1[0]}\n{messages[lang]["desc"]}: {results_1[1]}\n {messages[lang]["ls"]} {results_1[3]}: {results_1[4]}\n {m
            # запускаем асинхронную функцию в отдельном потоке
            loop = asyncio.new_event_loop()
            t = threading.Thread(target=run_async_function_in_thread, args=(loop,))
            t.start()
    except IndexError:
        bot.send_message(message.chat.id, messages[lang]["a"]) #Вибачте, це кіно не знайдено

def run_async_function_in_thread(loop):
    asyncio.set_event_loop(loop)
    loop.run_until_complete(track_new_episodes())
```

Рисунок 1.18. Функції обробки запиту назви серіала чи фільма

```python
def Film(film_name):
    print(film_name)
    # Формируем URL-адрес для поиска сериала на TMDB
    url = f"https://api.themoviedb.org/3/search/movie?api_key={API_KEY}&query={film_name}"

    # Отправляем запрос к TMDB API
    response = requests.get(url)

    # Извлекаем ID сериала из ответа
    series_id = response.json()["results"][0]["id"]

    url = f"https://api.themoviedb.org/3/movie/{series_id}?api_key={API_KEY}&language={lang}"

    response = requests.get(url)

    poster_path = response.json().get("poster_path", "Постер не найден")

    original_title = response.json().get("original_title", "Не найдено")

    overview = response.json().get("overview", "Не найдено")

    poster_url = f"https://image.tmdb.org/t/p/w500/{poster_path}"
    directory = "images" # имя папки, в которую нужно сохранить изображение
    if not os.path.exists(directory):
        os.makedirs(directory)

    responsed = requests.get(poster_url)
    image = Image.open(BytesIO(responsed.content))
    filepath = os.path.join(directory, "image.jpg")
    image.save(filepath)
    poster = Image.open(filepath)

    #выводим название, описание, url изображения

    return(original_title, overview, poster)
```

Рис. 1.19. Функція Film

```python
def Serial(series_name):
    print(series_name)
    #Формируем URL-адрес для поиска сериала на TMDB
    url = f"https://api.themoviedb.org/3/search/tv?api_key={API_KEY}&query={series_name}"

    # Отправляем запрос к TMDB API
    response = requests.get(url)

    # Извлекаем ID сериала из ответа
    global series_id
    series_id = response.json()["results"][0]["id"]

    print(series_id)

    url = f"https://api.themoviedb.org/3/tv/{series_id}?api_key={API_KEY}&language={lang}"

    en = f"https://api.themoviedb.org/3/tv/{series_id}?api_key={API_KEY}&language=en-EN"

    response = requests.get(url)

    response_1 = requests.get(en)

    poster_path = response.json().get("poster_path", "Постер не найден")

    overview = response.json().get("overview", "Не найдено")

    name = response.json().get("name", "Не найдено")

    namen = response_1.json().get("name", "Не найдено")

    poster_url = f"https://image.tmdb.org/t/p/w500/{poster_path}"
    directory = "images" # имя папки, в которую нужно сохранить изображение
    if not os.path.exists(directory):
        os.makedirs(directory)
    responsed = requests.get(poster_url)
    image = Image.open(BytesIO(responsed.content))
    filepath = os.path.join(directory, "image.jpg")
    image.save(filepath)
    poster = Image.open(filepath)
    with open(filepath, "rb") as f:
        image_bytes = f.read()
    image_b64 = base64.b64encode(image_bytes).decode('utf-8')

    aboba = []
    # Извлекаем номер последней серии из ответа
    last_episode_number = response.json().get("last_episode_to_air", {}).get("episode_number", "не найдено")
    aboba.append(last_episode_number)
    if response.json() is not None:
```

Рисунок 1.20. Функція Serial

Сайт працює за допомогою бібліотеки Django, база данних створюється у моделях, а саме у файлі models.py (рис. 1.21).

```python
from django.db import models

class User(models.Model):
    telegram_id = models.IntegerField()

    class Meta:
        db_table = 'users'

class Show(models.Model):
    name = models.TextField()
    last_episode = models.IntegerField()
    serial_id = models.IntegerField()
    image = models.BinaryField()
    overview = models.TextField()
    namen = models.TextField()

    class Meta:
        db_table = 'shows'

class UserShow(models.Model):
    user = models.ForeignKey(User, on_delete=models.CASCADE)
    show = models.ForeignKey(Show, on_delete=models.CASCADE)

    class Meta:
        db_table = 'users_shows'
```

Рисунок 1.21. Моделі Django

Файл `models.py` в Django використовується для визначення моделей даних, які будуть використовуватися в додатку. Він визначає структуру бази даних і

спосіб взаємодії з нею.

У цьому коді визначено три моделі: `User`, `Show` і `UserShow`.

Модель `User` представляє користувача додатка. Вона містить одне поле `telegram_id`, яке є цілочисельним полем. Клас `User` успадковується від `models.Model` і використовує клас `IntegerField` з модуля `models` для визначення типу поля.

Модель `Show` представляє шоу (серіал або аніме). Вона містить кілька полів, таких як `name`, `last_episode`,`serial_id`, `image`, `overview` і `namen`. Кожне поле визначено з використанням відповідного типу поля з модуля `models`, наприклад, `TextField` для текстових полів і `BinaryField` для бінарних даних. Клас `Show` також успадковується від `models.Model`.

Модель `UserShow` представляє відношення між користувачем і шоу. Вона містить два поля: `user` і `show`, які є зовнішніми ключами, що пов'язують об'єкти з моделями `User` і `Show`. Це дає змогу встановити зв'язок між конкретним користувачем і конкретним шоу. Клас `UserShow` також успадковується від `models.Model`.

Класи моделей визначаються з використанням успадкування від `models.Model` і містять метаклас `Meta`, який дає змогу задати додаткові налаштування моделі, як-от ім'я таблиці в базі даних (`db_table`). Коли моделі визначені, Django використовує їх для створення відповідних таблиць у базі даних і надає зручний спосіб взаємодії з даними, включно зі створенням, читанням, оновленням і видаленням (CRUD) записів у таблицях бази даних.

Файл views.py (рис. 1.22) в Django є частиною MVC (Model-View-Controller) архітектури та використовується для визначення уявлень (views) додатка. Він містить функції (або класи, якщо використовується клас-спадкоємець View) для оброблення HTTP-запитів і формування HTTP-відповідей.

Функція show_list(request):

- Приймає об'єкт запиту (request) як параметр.
- Витягує user_id із сесії користувача.
- Якщо user_id існує:

– Отримує об'єкт користувача (user) з бази даних на основі user_id.
– Фільтрує об'єкти Show за відношенням usershow__user, щоб отримати пов'язані з користувачем шоу.
– В іншому випадку:
– Присвоює змінній shows порожній запит (об'єкт Show.objects.none()).
– Повертає відтворення шаблону my_template.html з передачею змінної shows в контекст шаблону.

Функція recommendations(request, show_id):

– Приймає об'єкт запиту (request) та ідентифікатор шоу (show_id) як параметри.
– Намагається виконати такі дії:
– Створює порожній список recommended_shows.
– Отримує об'єкт шоу (show) з бази даних на основі show_id.
– Витягує ім'я шоу (show_name) з об'єкта шоу.
– Встановлює URL і заголовки для запиту до API AniList.
– Задає GraphQL-запит для пошуку шоу за ім'ям і отримання рекомендацій.
– Виконує запит до API AniList.
– Перевіряє, чи були отримані будь-які результати від API AniList.
– Якщо результати були отримані:
– Присвоює recommended_shows списку рекомендацій із даних API AniList.
– В іншому разі:
– Встановлює ключ API TMDB і URL для запиту до API TMDB на ім'я шоу.
– Виконує запит до API TMDB для пошуку шоу за іменем.
– Перевіряє, чи були отримані будь-які результати від API TMDB.
– Якщо результати були отримані:
– Отримує ідентифікатор шоу (tmdb_show_id) з даних API TMDB.
– Виконує запит до API TMDB для отримання схожих шоу.
– Присвоює recommended_shows результатам із даних API TMDB.
– В іншому випадку:

– Встановлює recommended_shows порожнім списком.
– Повертає відтворення шаблону recommendations.html з передачею об'єктів show і recommended_shows в контекст шаблону.

Функція login(request):

– Перевіряє метод запиту (POST або будь-який інший).
– Якщо метод запиту дорівнює POST:
– Витягує значення telegram_id з POST-параметрів запиту.
– Намагається отримати об'єкт користувача (user) з бази даних на основі telegram_id.
– Якщо користувач існує:
– Встановлює user_id у сесії користувача.
– Перенаправляє на подання з ім'ям movie_list.
– В іншому випадку:
– Повертає відтворення шаблону login.html з передачею повідомлення про помилку в контекст шаблону.

```python
from django.shortcuts import render, redirect
from .models import User, Show, UserShow
import requests

def show_list(request):
    user_id = request.session.get('user_id')
    if user_id:
        user = U          t(id=user_id)
        shows = Show.objects.filter(usershow__user=user)
    else:
        shows = Show.objects.none()
    return render(request, 'my_template.html', {'shows': shows})


def recommendations(request, show_id):
    try:
        recommended_shows = [] # Создаем переменную и устанавливаем ее пустым списком
        # Fetch show name from database
        show = Show.objects.get(id=show_id)
        show_name = show.namen
        print(show_name)

        # Set AniList API URL and headers
        url = 'https://graphql.anilist.co'
        headers = {'Content-Type': 'application/json', 'Accept': 'application/json'}

        # Set GraphQL query to search for show by name and fetch recommendations
        query = '''
        query ($name: String) {
            Media (search: $name, type: ANIME) {
                recommendations {
                    edges {
                        node {
                            mediaRecommendation {
                                id
                                title {
                                    romaji
                                    english
                                    native
                                }
```

(class) Show

Рисунок 1.22. Функція, що виводить на сторінку рекомендації серіалів

Релізація HTML шаблонів (HTM-сторінки), усього їх 3. “login.html” (рис. 1.23), сторінка відображає форму входу користувача. Користувач авторизується за допомогою telegram id, який індивідуальний у кожного користувача месенджеру телеграм та відображається в боті при натисканні відповідної кнопки. “my_template.html” (рис. 1.24) ,ця HTML-сторінка являє собою список постерів різних шоу або фільмів, які відслідковує користувач бота. Самі постери, назви та інша інформація дістається з бази даних. “recommendations.html” (рис. 1.25), ця HTML-сторінка відображає рекомендації щодо шоу або фільмів. Вона складається з таких елементів:

– Кнопка перемикання теми між світлим і темним режимами.
– Контейнер `.container`, в якому розміщуються рекомендації.
– Обране шоу або фільм відображається в блоці `.selected-show` із зображенням, назвою та описом.
– Заголовок "Рекомендуємо переглянути" і рекомендації відображаються в блоках `.show`, кожен з яких містить зображення і назву рекомендованого шоу або фільму. Частину коду можна побачити на рисунку 3.15.

Сторінка також підключає зовнішній JavaScript-файл `main.js` і використовує змінні та цикли для відображення даних про рекомендації, які беруться з TMDB API та Anilist API.

```
<body>
    {% load static %}
{% load now %}
<script src="{% static 'main.js' %}?t={{ now|date:'U' }}"></script>
<button class="sun-moon"  onclick="changeBackgroundAndTextColor(); toggleSunMoon(this);"></button>
    <div class="container">
        <h1>Увійти</h1>
        {% if error %}
        <div class="error">{{ error }}</div>
        {% endif %}
        <form method="post">
            {% csrf_token %}
            <input type="text" name="telegram_id" placeholder="Введіть свій Telegram ID">
            <input type="submit" value="Увійти">
        </form>
    </div>
    <a href="https://t.me/Animuvie_bot" class="telegram-button">
        <i class="fa fa-telegram"></i>
        <span></span>
      </a>
</body>
```

Рисунок 1.23. Тіло login.html(body)

```
<body>
  {% load static %}
{% load now %}
<script src="{% static 'main.js' %}?t={{ now|date:'U' }}"></script>
<div class="container">
{% for show in shows %}
<div class="poster" onclick="window.location.href='{% url 'recommendations' show.id %}'">
    <img src="{{ show.image }}" alt="{{ show.name }}">
<div class="description">
<p>{{ show.overview }}</p>
</div>
</div>
{% endfor %}
</div>
</body>
```

Рисунок 1.24. Тіло my_template.html(body)

```
<body>
    {% load static %}
    {% load now %}
    <script src="{% static 'main.js' %}?t={{ now|date:'U' }}"></script>
    <button class="sun-moon"  onclick="changeBackgroundAndTextColor(); toggleSunMoon(this);"></button>
    <div class="container">
        <div class="selected-show">
            <div class="selected-show-image">
                <img src="{{ show.image }}" alt="{{ show.name }}">
            </div>
            <div class="selected-show-details">
                <div class="selected-show-title">{{ show.name }}</div>
                <div class="selected-show-overview">{{ show.overview }}</div>
            </div>
        </div>

        <h1>Рекомендуємо переглянути</h1>

        {% if recommended_shows %}
          {% if recommended_shows.0.node %}
              {% for recommendation in recommended_shows %}
                  <div class="show">
                      <div class="show-image">
                          <img src="{{ recommendation.node.mediaRecommendation.coverImage.large }}" alt="{{ recommendation.node.mediaRecommendation.title.romaji }}">
                      </div>
                      <div class="show-details">
                          <div class="show-title">{{ recommendation.node.mediaRecommendation.title.romaji }}</div>
                      </div>
                  </div>
              {% endfor %}
          {% else %}
              {% for recommendation in recommended_shows %}
                  <div class="show">
                      <div class="show-image">
                          <img src="https://image.tmdb.org/t/p/w500{{ recommendation.poster_path }}" alt="{{ recommendation.name }}">
                      </div>
                      <div class="show-details">
                          <div class="show-title">{{ recommendation.name }}</div>
                      </div>
                  </div>
              {% endfor %}
          {% endif %}
      {% endif %}
    </div>


</body>
```

Рисунок 1.25. Тіло recommendations.html(body)

Файл urls.py (рис. 1.26) в Django використовується для визначення відповідності між URL-адресами (адресами сторінок) і функціями, які повинні обробляти ці адреси. Він є частиною механізму маршрутизації Django.

Код у urls.py дає змогу нам налаштувати маршрути для різних сторінок вашого веб-додатку. Ми визначили маршрут для гололовної сторінки, сторінки реєстрації та сторінки рекомендацій. Коли Django отримує запит із певною URL-адресою, він звертається до файлу urls.py, щоб знайти відповідну функцію обробки запиту і повернути відповідну сторінку у відповіді.

```
urlpatterns = [
    path('admin/', admin.site.urls),
    path('', login, name='login'),
    path('recommendations/<int:show_id>/', views.recommendations, name='recommendations'),
    path('shows/', show_list, name='movie_list'),
]
```

Рисунок. 1.26. Файл маршрутів (urls.py)

1.3.4 Тестування чат-бота та веб-додатка

Ураховуючи той факт, що чат-бот розроблений для платформи месенджера Telegram, під час тестування використовуються пристрої, що підтримують дану платформу. Telegram може бути використаний на різних пристроях, таких як персональні комп'ютери, планшети, ноутбуки або мобільні телефони. Крім того, існує також веб-сайт web.telegram.org, що дає змогу користуватися Telegram без необхідності встановлення додатку на пристрій.

Були проведені тестування на персональному комп’ютері з такими характеристиками:

- Операційна система: Microsoft Windows 10
- Оперативна пам'ять: 16 ГБ
- Відеокарта: Nvidia GTX 1050 TI
- Процесор: Intel® Core™ i3-8100
- Вбудована пам'ять: HHD -1ТБ, SSD- 480ГБ
- Telegram для Android v9.6.5

Процедура тестування:

1. Знаходимо чат-бот у телеграмі, для цього знайдемо у телеграм та напишемо animovie.
2. Розпочинаємо спілкування за допомогою команди /start.
3. У результаті, бачимо (рис. 1.27), що бот відповідає. Натиснувши - Так, з’являється клавіатура.

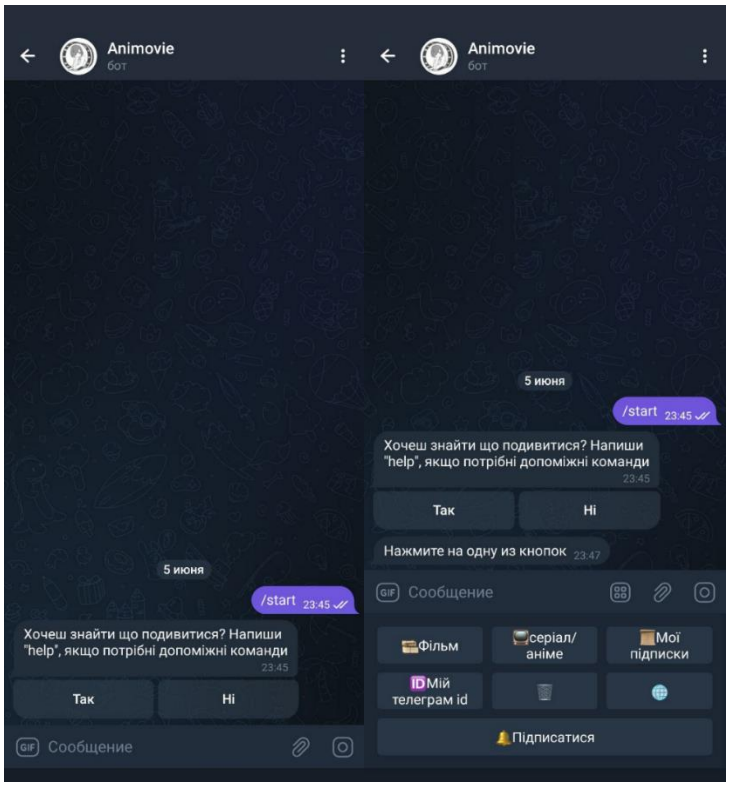


Рисунок 1.27. Початок роботи чат боту

Ми бачимо різні кнопки за допомогою яких користувач користується ботом.

Тепер потрібно натиснути на одну з кнопок на клавіатурі, наприклад натиснувши кнопку - Фільм, бот попросить увести назву фільму, ввівши його ми бачимо постер, назву та короткий опис фільму (рис. 1.28).

Рисунок 1.28. Знаходження фільмів.

Також можна натиснути кнопку Серіал/аніме, щоб знайти, наприклад, новий серіал або аніме. У цьому разі різниця ще в тому, що відображається (рис. 1.29) остання актуальна серія та дата наступної серії, яка незабаром вийде.

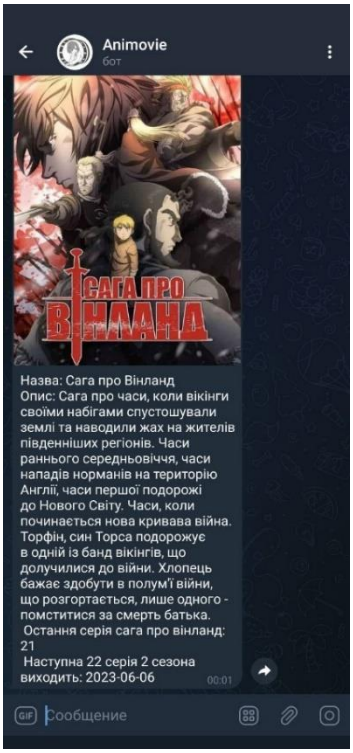


Рисунок 1.29. Знаходження фільмів.

Наприклад якщо ввести неіснуючий серіал чи фільм, то буде відповідне повідомлення (рис. 1.30).

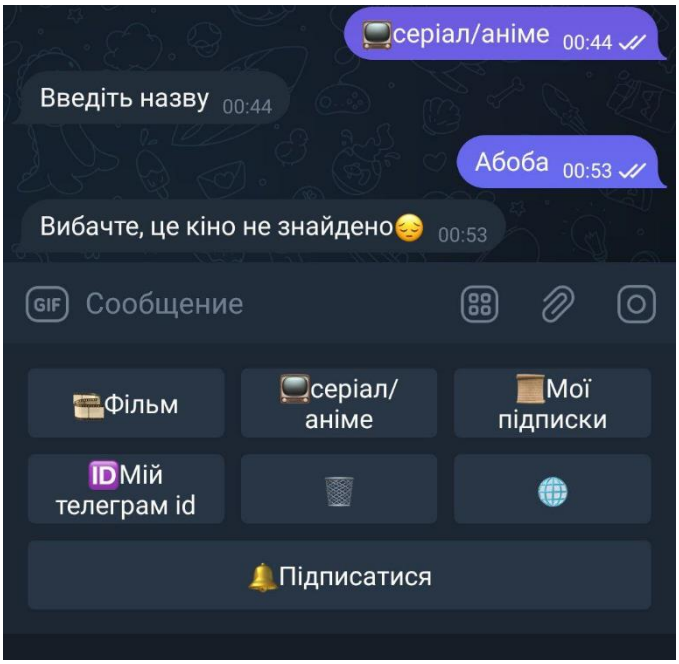


Рисунок 1.30. Обробка помилки

Натиснувши кнопку - Підписатися, оформиться підписка на оновлення (рис. 1.31).

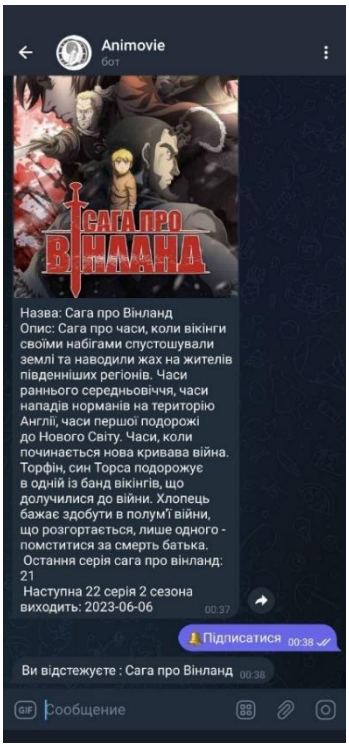


Рисунок 1.31. Відстежування

Якщо нажати кнопку мої підписки, можна побачити список серіалів, на які підписався користувач, а якщо нажати кнопку у вигляді корзини можна видалити з підписок увівши унікальний номер серіалу, який видно у списку (рис. 1.32).

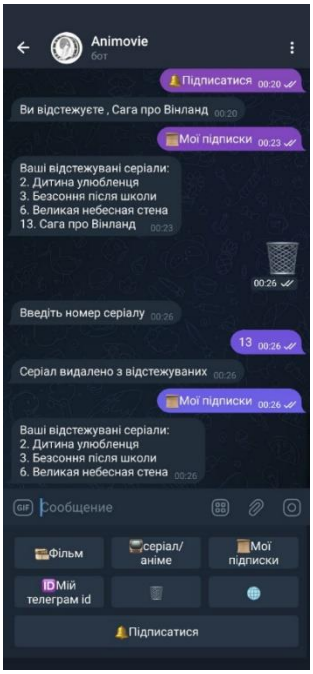


Рисунок 1.32. Видалення з підписок

Якщо натиснути на кнопку у вигляді глобуса, можна обрати мову інтерфейсу (рис. 1.33, 1.34).

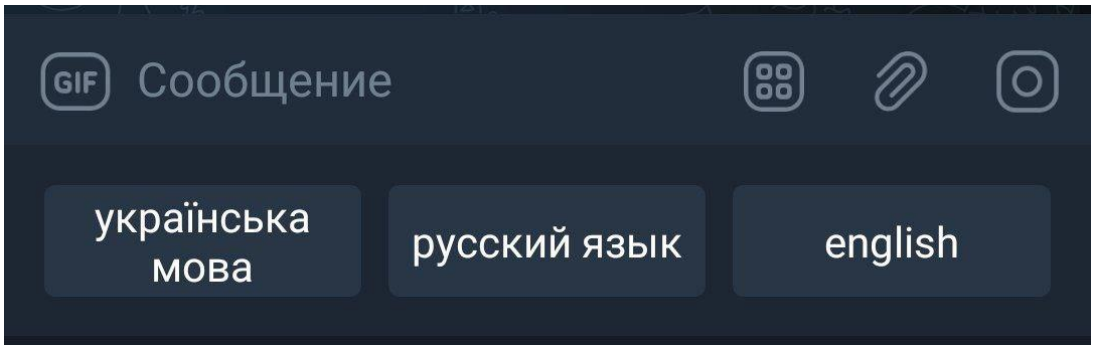


Рисунок 1.33. Доступні мови інтерфейсу

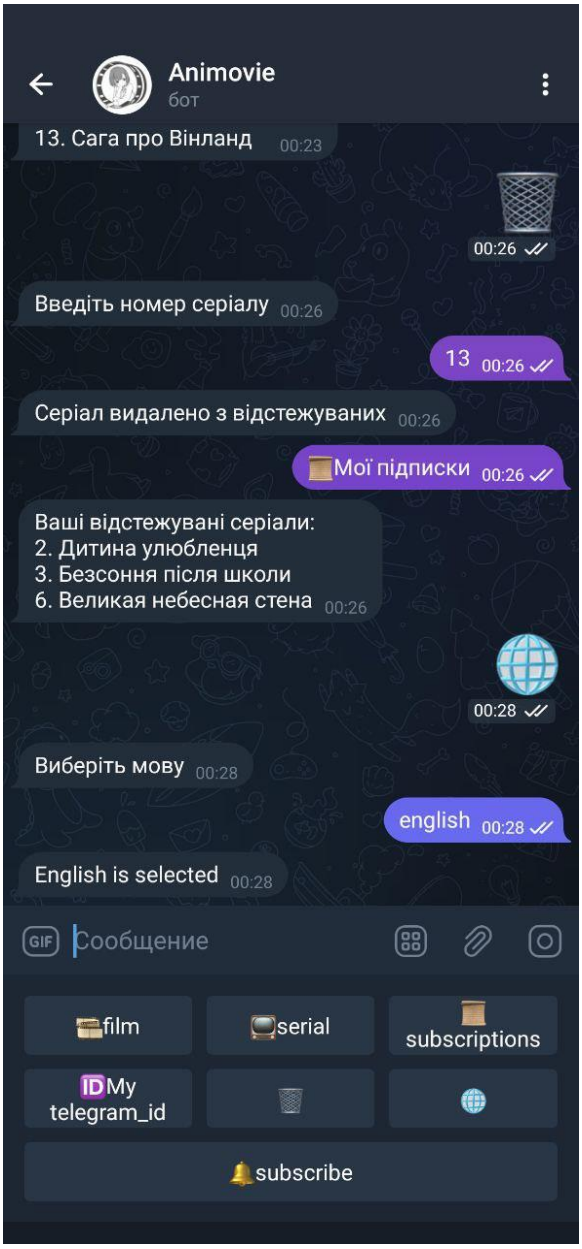


Рисунок 1.34. Обрана англійська мова

Також можливо подивитись свій телеграм id за допомогою якого можливо

зайти на наш сайт animovie (рис. 1.35).

Рисунок 1.35. Сторінка реєстрації

На сторінці реєстрації користувач вводить свій телеграм id, після цього відкривається сторінка з його підписками (рис.1.36). Це саме ті серіали на які підписаний користувач у телеграм боті. Якшо навести мишу на постер, то можна побачити коротку інформацію про серіал.

Рисунок 1.36. Сторінка підписок

Якщо натиснути на один з постерів ми побачимо назву, опис та рекомендації схожих серіалів для перегляду (рис 1.37).

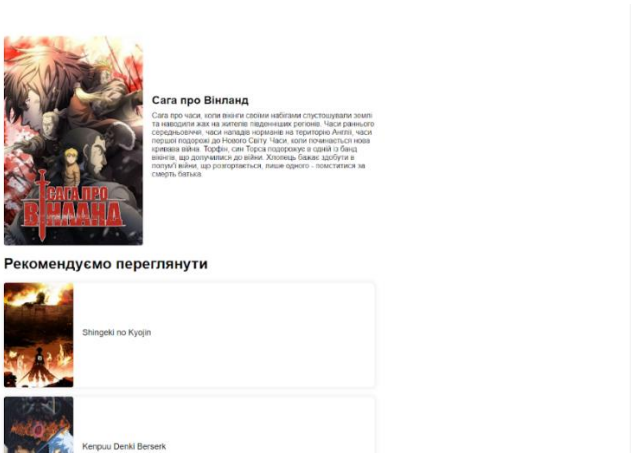


Рисунок 1.37. Сторінка рекомендацій

Також, якщо натиснути на кнопку у вигляді сонця, можна змінити тему сайту на нічну, тоді фон та інші елементи стають темними, а текст білим.

У результаті проведеного тестування було підтверджено, що програмний продукт відповідає поставленим вимогам та функціональним вимогам. У процесі тестування були перевірені основні функції і задачі чат-бота для пошуку фільмів та серіалів, а також його взаємодія з користувачами, відстеження оновлень та інші функції. Були виявлені та виправлені деякі недоліки та помилки, що дозволило досягти більшої ефективності та надійності роботи програмного продукту. Загальний результат тестування підтвердив його готовність до використання в реальних умовах та задоволення потреб користувачів.

2 ЕКОНОМІЧНА ЧАСТИНА

В умовах розвитку сучасного інформаційного суспільства та економічних відносин, які в ньому встановилися, дуже важливою є роль інформаційних ресурсів, до яких відносять веб-сайти. Вдале використання інформаційного забезпечення та інформаційних ресурсів може значно підвищити ефективність функціонування підприємства. Як правило, веб-сайт створюється підприємством для надання основної інформації щодо його діяльності, характеристик товарів і послуг, що надаються, розміщення реклами, публікації контактів тощо. Основна мета створення телеграм боту та сайту – це забезпечення максимально можливого прибутку за умов мінімізації витрат.

При оцінці ефективності створюваного сайту слід виходити з того, що залежно від характеру ефекту, що досягається, можуть бути визначені наступні види ефективності сайту: економічна, функціональна та соціальна ефективність. Ці види ефективності в свою чергу залежать від технологічних, технічних, маркетингових та економічних показників веб-сайту. Між доходом від сайту і витратами на його створення існує певна залежність. Слід також враховувати, що на величину доходу від сайту, окрім інших чинників ефективності, можуть вплинути витрати на його оптимізацію та просування.

Розрахунок економічної ефективності розробки створеного web-сайту.

Загальні витрати ($В_з$) на створення сайту складаються з декількох параметрів:

$$В_з = В_р + В_в + В_е \ ,$$

де $В_р$ – витрати на розробку сайту;

$В_в$ – витрати на впровадження сайту;

$В_е$ – витрати на експлуатацію сайту;

Витрати на розробку сайту ($В_р$) є одноразовими та складаються з вартості наступних видів робіт зі створення сайту Розробка дизайну сайту: розробка макетів дизайну для головної та внутрішньої сторінок сайту; розробка фірмового стилю, логотипуРеалізація на сайті меню: звичайного багаторівневого, «випадає»

«розсовують» багаторівневого Підготовка сторінок-шаблонів Наповнення сайту інформацією: наповнення та форматування web-сторінок; обробка малюнків для публікації на web-сторінках, верстка (переклад в HTML-формат) web-сторінок Програмна розробка сайту: створення програмного коду сайту, програмування динамічних елементів Реалізація пошукових можливостей сайту: настройка модуля пошуку по сайту; створення карти сайту; настройка виведення шляху по сайту Налаштування модулів: модуля новин, модуля документів (створення лістингу документів), модуля каталогу товарів, модулів інтернет-магазину (каталог, корзина, замовлення), модуля листа розсилки, модуля конструктора форм, модуля e-mail форм, модуля питань і відповідей, модуля опитувань і голосувань, модуля web-форуму, модуля на багато користувачів доступу до системи і т.п.

Склад видів робіт зі створення сайту може бути уточнений з урахуванням типу створюваного сайту і реалізованих їм функціональних можливостей.

Для визначення витрат на розробку сайту ($В_р$) необхідно розрахувати оплату праці виконавців, безпосередньо притягнених до її виконання. Для реалізації проекту Web-системи використовуються програміст.

Для визначення трудомісткості розробки сайту ($В_р$) складено план-графік по розробці web-сайту і тривалості виконання робіт. Розподіл робіт по етапах і видах виконавців наведено в таблиці 3.1.

Таблиця 2.1. План-графік по розробці Web-сайту

№	**Назва етапу**	**Час виконання (годин)**	**Посада виконавця**
1	Аналіз технічного завдання	14	Програміст
2	Розробка алгоритму роботи серверної частини	8	Програміст
3	Розробка програмного забезпечення бота	54	Програміст
4	Розробка серверної частини веб-додатку	29	Програміст
5	Розробка користувальницької частини веб-додатка	18	Програміст
6	Тестування програмного забезпечення	12	Програміст
ВСЬОГО:		135	

Розрахунок трудомісткості здійснений в наступній послідовності:

1. Складений перелік всіх етапів і видів робіт, які необхідно виконати в ході даної розробки. Після узгодження з керівником проекту допущено виключення, доповнення, об'єднання окремих етапів і видів робіт;

2. По кожному виду робіт визначений кваліфікаційний рівень виконавців. В разі виконання однієї роботи виконавцями різної кваліфікації, робота розподілена на ряд паралельних конкретних робіт для кожної категорії виконавця.

В умовах відсутності нормативної бази тривалість виконання окремих робіт розраховується на основі вірогідних оцінок робіт, що задаються виконавцями.

Розмір заробітної плати розраховується виходячи з чисельності різних категорій виконавців, трудомісткості, що витрачається ними на виконання різних видів робіт, а також їх середньої заробітної плати (ставки) за годину (або один робочий день).

При визначенні вартості виконуваних робіт можливо орієнтуватися на ціни, представлені на сайтах фірм, що спеціалізуються в сфері створення та модернізації web-ресурсів або на мінімальну заробітну плату, встановлену Відповідно до «Закону про Державний бюджет України» (станом на 1.01 поточного року), враховуючи кваліфікацію виконавців, Витрати на заробітну плату приведені в таблиці 3.2. (мінімальна заробітна плата в місяць - 6700 грн; в годину - 40,43 грн)

Таблиця 2.2 – Витрати на заробітну плату

№	Персонал	Етапи розробки	Кількість робочих днів	Погодинна ставка, грн.	Заробітна плата, грн.
1	Програміст	1, 2, 3, 4, 5	5	140	21 840
ВСЬОГО:					$В_{зп}$ =21 840

До складу витрат на оплату праці також включаються податки, збори і інші обов'язкові платежі, встановлені системою оподаткування що діє. Розмір єдиного соціального внеску складає 22% від заробітної плати, розраховується за наступною

формулою:

$$В_{єсв} = В_{зп} \times 0,22$$

Загальні витрати ($В_р$) на розробку веб-сайту розраховуються як сума витрат на заробітну плату праці персоналу ($В_{зп}$) та єдиного соціального внеску ($В_{єсв}$):

$$В_р = В_{зп} + В_{єсв}$$

Витрати на впровадження сайту ($В_в$) складаються з двох складових :

- витрати на реєстрацію доменного імені на 1 рік ($В_{в1}$);

- витрати на реєстрацію в пошукових системах ($В_{в2}$), наприклад, Yandex, Google, Rambler и т.п.)

$$В_в = В_{в1} + В_{в2}$$

Витрати на експлуатацію сайту (Ве) включають вартість робіт з підтримки сайту в робочому стані і вартість послуг по продовженню доменного імені на 1 рік.

Роботи по підтримці сайту в робочому стані включають в себе:

1. Оновлення даних на сайті;
2. Створення нових розділів на сайті;
3. Видалення застарілої інформації з сайту;
4. Додавання потрібної інформації на сайт;
5. Налаштування параметрів сервера хостингу;
6. Моніторинг роботи сервера хостингу;
7. Забезпечення щомісячного захисту сайту;
8. Створення резервних копій сайту та ін.

Підтримка сайту в робочому стані може здійснювати сама організація або спеціалізована фірма, яка виконує ці види робіт за договором. У першому випадку витрати розраховуються виходячи із заробітної плати (або доплати) співробітникові організації, призначеного для виконання цього виду робіт. Для певних робіт з цього переліку може використовуватися обслуговуючий персонал (адміністратор web-сайту). У другому випадку слід скористатися прайс-листами на відповідні послуги спеціалізованих фірм. У таблиці 3.3 визначаються постійні

витрати як сума витрат на впровадження та експлуатацію сайту протягом року.

Таблиця 2.3 – Постійні витрати

№	Стаття витрат	Вартість за рік, грн.
1	Розробка бота	17 000
2	Розробка серверної частини сайта	12 000
3	Хостинг	1 000
4	Доменне ім’я	200
Всього:		30 200

Загальні витрати ($В_з$) на розробку, впровадження та експлуатацію веб-сайту розраховуються за наступною формулою:

Вз = Вр + (Вв + Ве), де Вр - витрати на розробку сайту, Вв - витрати на впровадження сайту, а Ве - витрати на експлуатацію сайту, можна розрахувати загальні витрати на створення сайту: \

Вз = 21 840 грн + (1000 грн + 30200 грн) = 53 040 грн.

Отже, загальні витрати на створення сайту складають 53 040 грн.

3 ОХОРОНА ПРАЦІ

Безпека праці є важливою галуззю, спрямованою на створення безпечних умов праці. У контексті сучасного виробництва, ця проблематика набуває ще більшого значення. Забезпечення безпечних умов праці на виробництві вимагає значних витрат, впровадження науково-дослідних знань і рішень у галузі охорони праці. Особливу роль у цьому процесі відіграють інженерно-технічні працівники з глибокими знаннями в галузі охорони праці. Компанії та організації повинні вкласти зусилля в розробку і впровадження політик безпеки праці, надавати належну підтримку та навчання працівників, щоб створити безпечне та здорове робоче середовище.

3.1 Аналіз та безпека умов працівника на робочому місці

Питання розробки Telegram – боту та сайту для відстежування оновлень фільмів та серіалів передбачає працю робітника з використанням ПК Тому до розгляду беремо робоче місце програміста в приміщенні офісу.

Високий рівень уважності і свідомого підходу до вирішення питань безпеки праці у програмістів допоможе забезпечити їхнє здоров'я, добробут і ефективність у роботі.

Аналіз шкідливих та небезпечних факторів виробничого середовища є необхідною складовою частиною процесу охорони праці. При оцінці умов праці програміста важливо враховувати зазначені фактори та приймати заходи для мінімізації їх впливу. Це може включати застосування ізольованих робочих місць, встановлення звукоізоляційних матеріалів, регулювання температури та вологості повітря, використання антивібраційних матеріалів та екранів, а також забезпечення належної освітленості робочої зони.

3.2 Гігієнічні вимоги до виробничого середовища

Розробка заходів з охорони праці є важливим етапом в забезпеченні безпеки та здоров'я працівників. Цей процес включає систематичний підхід до виявлення потенційних ризиків, їх аналізу та прийняття необхідних заходів для запобігання

можливим нещасним випадкам та впливу шкідливих факторів на здоров'я працівників.

Це аналіз рівня шуму, вібрацій, температурних умов, рівня освітленості, хімічних речовин тощо.

На основі результатів аналізу формулюються конкретні заходи з охорони праці, спрямовані на зменшення або усунення виявлених ризиків.

Загальною метою розробки заходів з охорони праці є забезпечення безпечних та нешкідливих умов праці для всіх працівників. Це допомагає знизити ризик виникнення нещасних випадків, професійних захворювань та покращити загальну продуктивність та добробут організації.

3.3 Вимоги до приміщення

Вимоги до приміщення великого значення для безпеки та комфорту працівників. Робочий простір повинен бути достатньо просторим, щоб працівник мав місце для обладнання та зручного виконання роботи. Освітлення повинно бути достатнім і якісним, щоб уникнути напруження очей та зниження продуктивності. Температура повинна бути комфортною для працівників, уникати занадто високих або низьких значень, які можуть впливати на їх здоров'я та роботу.

Умови праці для програмістів мають відповідати вимогам безпеки праці, згідно з нормативно-правовими актами, такими як ДСаПіН 3.3.2.-007-98 Державні санітарні правила і норми. Гігієнічні вимоги до організації роботи з візуальними дисплейними терміналами електронно-обчислювальних машин.

Робоче місце програміста включає персональний комп'ютер з відповідним програмним забезпеченням та може бути організоване в офісному приміщенні. Під час роботи на програміста можуть впливати різні шкідливі та безпечні фактори, такі як підвищена або знижена температура повітря, шум, вологість, рухливість повітря, іонізація, вібрація, електромагнітні випромінювання. Також важливо забезпечити достатнє природне освітлення робочої зони, оскільки недостатня освітленість може впливати на зорову функцію та спричиняти зморшки. Крім того, нервово-психічні перевантаження є ще одним аспектом, який потрібно враховувати. Вимоги до безпеки праці належить дотримуватися, щоб забезпечити

здоров'я та безпеку працівника

Організація робочого місця важлива для ефективності, комфорту та безпеки працівників. Це включає оптимальне розташування обладнання, належну організацію робочого простору та розміщення матеріалів та інструментів. Розташування обладнання повинно бути зручним та доступним, а робочий простір має мати раціональну розкладку та враховувати фактори безпеки. Організація робочого простору включає встановлення раціональної розкладки робочих зон, столів, стільців та інших елементів робочого середовища. Розташування робочих зон повинно сприяти зручності та продуктивності працівників. Також необхідно враховувати фактори безпеки, наприклад, забезпечити достатні просторові межі між робочими станціями, щоб уникнути зіткнень та травм.

3.4 Освітлення робочого місця

Освітлення робочого місця відіграє важливу роль у забезпеченні комфорту, безпеки та продуктивності працівників. Добре сплановане та належне освітлення допомагає запобігати напруженню очей, покращує видимість та сприяє збільшенню концентрації та продуктивності працівників. При проектуванні освітлення робочого місця слід враховувати багато факторів:

Природне освітлення - максимальне використання природного світла допомагає знизити використання штучного освітлення та сприяє створенню природного та комфортного робочого середовища. Важливо забезпечити належну кількість вікон та їх розміщення, щоб уникнути прямих блисків на робочу поверхню та додаткового навантаження на очі працівників.

Штучне освітлення - там, де природне освітлення недостатнє, слід використовувати штучне освітлення. Важливо встановити освітлювальні прилади, які забезпечують належну якість та інтенсивність світла. Розміщення світильників повинно бути рівномірним та розподіллено по всьому робочому простору, щоб уникнути тіней та недостатнього освітлення деяких зон.

Колірна температура - вибір правильної колірної температури світла є важливим фактором. Тепле біле світло (низька колірна температура) може

створювати затишну та розслаблюючу атмосферу, підходить для приміщень, де вимагається відпочинок, наприклад, перерви. Холодне біле світло (висока колірна температура) надає більш яскраве і енергійне освітлення, підходить для робочих приміщень, де необхідна більша концентрація та активність.

Рівномірне розподілення світла на робочому місці важливе. Уникайте різкого контрасту між освітленою та затемненою областями. Регульоване освітлення дозволяє пристосувати його до потреб працівника.

Загальний рівень освітлення також має бути достатнім. Недостатня освітленість може напружувати очі та знижувати увагу. Рекомендований рівень освітленості залежить від типу роботи.

При роботі з комп'ютером важливо уникати відблисків на моніторі та забезпечувати достатню контрастність між шрифтами та фоном. Регулярні перерви для відпочинку очей також можуть бути корисними.

Належне освітлення робочого місця є важливим для комфорту, безпеки та продуктивності працівників.

3.5 Мікроклімат

Мікроклімат в робочому середовищі має важливе значення для комфорту, здоров'я та продуктивності працівників. Він включає параметри, такі як температура, вологість повітря, швидкість руху повітря і якість повітря. Оптимальна температура робочого середовища залежить від характеру роботи та сезону. Зазвичай рекомендована діапазон температур для комфорту працівників становить від 20 до 24 градусів Цельсія.

Рекомендована вологість повітря в робочому середовищі зазвичай становить від 40% до 60%. Надлишок вологості або недостатній рівень вологості можуть негативно впливати на здоров'я та комфорт працівників.

Швидкість руху повітря також має значення. Занадто сильний потік повітря або його недостатній рух можуть створювати дискомфорт.

Важливо забезпечити належну якість повітря в робочому середовищі, усунути або зменшити концентрацію шкідливих речовин. Для цього можна

проводити регулярну вентиляцію приміщення, контролювати рівень вологості, перевіряти якість повітря та дотримуватись нормативів безпеки.

Організація зон відпочинку є важливим аспектом забезпечення здорового та безпечного робочого середовища, що дозволяє працівникам відпочивати і відновлювати сили.

Іонізація повітря

Іонізація повітря - це процес, за якого нейтральні молекули повітря набувають додатнього або від'ємного заряду, перетворюючись на іони. Цей процес може бути досягнутий за допомогою спеціальних пристроїв, відомих як іонізатори або іонізуючі фільтри.

Іонізація повітря має деякі потенційні переваги для робочого середовища:

Очищення повітря – Іонізатори сприяють зменшенню концентрації алергенів, пилу, пилових кліщів, бактерій та вірусів у повітрі. Вони приводять до осадження шкідливих частинок на поверхні та утворення їх агломератів, що полегшує їх видалення з повітря.

Важливо зауважити, що при використанні іонізаторів необхідно дотримуватися встановлених норм щодо випуску іонів, оскільки певні рівні іонів можуть бути шкідливими для здоров'я людей. Рівні позитивних та негативних іонів у повітрі робочих приміщень з ВДТ мають відповідати санітарно-гігієнічним нормам N 2152-80. Тому рекомендується обирати сертифіковані пристрої і контролювати їхню роботу.

3.6 Пожежна безпека

Пожежна безпека є важливим аспектом у робочому середовищі. Забезпечення належних заходів пожежної безпеки запобігає виникненню пожеж, захищає майно та людей.

В приміщенні дотримуються всіх вимог пожежної безпеки відповідно до нормативного акта НАПБ А.0.001-2014 "Правила пожежної безпеки в Україні". Також у приміщенні розроблений план евакуації у разі пожежі. Час, необхідний для повної евакуації, відповідає вимогам будівельного стандарту СНиП 2.01.02-85, а максимальна відстань від робочих місць до евакуаційних виходів відповідає

СНиП 2.09.02-85.

На робочому місці для гасіння пожеж застосовуються вуглекислотні та порошкові вогнегасники. Кількість та вид вогнегасників відповідають вимогам національного стандарту ГОСТ 12.4.009-75 та міжнародного стандарту ISO3941-77.

Безпека праці є надзвичайно важливою галуззю практичної діяльності, особливо в контексті сучасного розвитку виробництва. Створення безпечних та нешкідливих умов праці вимагає значних матеріальних витрат і впровадження науково-дослідних розробок в галузі охорони праці. Знання з питань охорони праці відіграють важливу роль серед інженерно-технічних працівників.

У контексті робочого місця програміста, вимоги до безпеки праці повинні відповідати нормативно-правовим актам, зокрема НПАОП 0.00-7.15-18 та НПАОП 0.00-1.28-10. Аналіз шкідливих та небезпечних факторів виробничого середовища вказує на підвищену або знижену температуру повітря, рівень шуму, вологість, рухливість та іонізацію повітря, вібрацію, рівень електромагнітних випромінювань та освітленість.

Важливо розробити та впровадити заходи з охорони праці, які забезпечать комфортні та безпечні умови для програмістів. Це можуть бути заходи щодо контролю температури, шуму, вологості, вентиляції, освітленості та інших факторів. Також важливо уникати нервово-психічних перевантажень працівників.

Узагалі, ефективна організація робочого місця, враховуючи вимоги безпеки праці, сприяє збереженню здоров'я працівників, підвищенню їх продуктивності та задоволеності від роботи. Постійне вдосконалення умов праці та впровадження сучасних підходів до безпеки є важливим завданням як для роботодавців, так і для працівників у галузі програмування та інформаційних технологій.

ВИСНОВОК

У рамках дипломної роботи було успішно розроблено та реалізовано комплексний проект, який включає в себе два основні компоненти: чат-бот для пошуку фільмів за різними критеріями та веб-сайт для виведення оновлень фільмів та серіалів, на які підписані користувачі, а також рекомендації на основі їхньої схожості. Чат-бот надає зручний інтерфейс для користувачів, де вони можуть швидко знайти актуальні фільми та серіали за жанром, роком, рейтингом, країною тощо, переглянути рекомендації на основі їх вподобань, а також відстежувати оновлення своїх улюблених фільмів та серіалів, отримуючи сповіщення про нові серії. Веб-сайт надає користувачам можливість переглядати список їх підписаних фільмів та серіалів у зручному форматі, а також отримувати рекомендації на основі схожих серіалів.

У проекті розроблено наступне:

1) Аналізовано загальні принципи та способи побудови чат-бота для обміну повідомленнями Телеграм та вивчено програмні засоби для його розробки.
2) Розроблено та реалізовано чат-бота для пошуку фільмів за різними критеріями, з можливістю перегляду детальної інформації про фільми, отримання рекомендацій та відстеження оновлень улюблених фільмів та серіалів.
3) Створено веб-сайт, який виводить оновлення фільмів та серіалів, на які підписані користувачі, і надає рекомендації на основі їхніх вподобань.
4) Забезпечено можливість користувачам зручно шукати та відстежувати свої улюблені фільми та серіали, отримувати сповіщення про нові серії та отримувати детальну інформацію про них.
5) Забезпечено можливість користувачам зручно шукати та відстежувати свої улюблені фільми та серіали, отримувати сповіщення про нові серії та отримувати детальну інформацію про них.

Цей комплексний проект успішно реалізовує ідею створення зручних інструментів для пошуку, відстеження та отримання рекомендацій щодо фільмів та серіалів. Він відповідає потребам користувачів, які шукають зручні інструменти для відстежування серіалів.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Real Python - онлайн-ресурс з навчання Python та Django [Електронний ресурс]. URL: https://realpython.com/

2. Праворська Н.І. Інформатика та комп'ютерна техніка: Навчальнометодичний посібник для студентів вищих навчальних закладів. – Хмельницький, 2002. – 312с

3. Скрапинг веб-сайтов с помощью Python. / Райан Митчелл. – ЛитРес, 2022 г. – 336 с.

4. Копайгородська Т.Г.Методичні вказівки до виконання економічного розділу. ВСП ОТФК ОНТУ, 2023

5. Документація Django [Електронний ресурс]. URL: https://docs.djangoproject.com/

6. Грофф Д. SQL: полное руководство / Д. Грофф, П. Вайнберг. – К.: BHV, 2005. – 608 с.

7. Python is a programming language, [Електронний ресурс]. URL: **https://www.python.org**

8. Stack Overflow - платформа для питань та відповідей з програмування [Електронний ресурс]. URL: https://stackoverflow.com/

9. Документація SQLite [Електронний ресурс]. URL: **https://www.sqlite.org/docs.html**

10. Two Scoops of Django 3.x: Best Practices for the Django Web Framework, 2020 р. - 550 с.

Додаток А Програмний код

Tgbot:

```
def buttons(lang):
        markup_reply = types.ReplyKeyboardMarkup(resize_keyboard= True)
        item_film = types.KeyboardButton('🎞' + messages[lang]['film'])
        item_username = types.KeyboardButton('📺' + messages[lang]['serial'])
        item_list = types.KeyboardButton('📜' + messages[lang]['subscriptions'])
        item_follow = types.KeyboardButton('🔔' + messages[lang]['subscribe'])
        item_id = types.KeyboardButton('🆔' + messages[lang]['telegram_id'])
        item_delete = types.KeyboardButton('🗑')
        item_language = types.KeyboardButton('🌐')
        markup_reply.add(item_film, item_username, item_list, item_id, item_delete,
item_language, item_follow)
        return markup_reply
def language_buttons():
    markup_reply = types.ReplyKeyboardMarkup(resize_keyboard=True)
    item_ukrainian = types.KeyboardButton('українська мова')
    item_russian = types.KeyboardButton('русский язык')
    item_english = types.KeyboardButton('english')
    markup_reply.add(item_ukrainian, item_russian, item_english)
    return markup_reply
@bot.message_handler(commands=['start'])
def get_user_info(message):
        with sqlite3.connect('C:/pythonproj/tgbot2022/base.db') as db:
                cursor = db.cursor()
                cursor.execute('''CREATE TABLE IF NOT EXISTS users
                (id INTEGER PRIMARY KEY, telegram_id INTEGER)''')
                cursor.execute('''CREATE TABLE IF NOT EXISTS shows
                (id  INTEGER PRIMARY KEY, name TEXT, last_episode INTEGER,
```

```
serial_id INTEGER, image BLOB, overview TEXT, namen TEXT)''')
            cursor.execute('''CREATE TABLE IF NOT EXISTS users_shows
            (user_id INTEGER, show_id INTEGER,
            FOREIGN KEY (user_id) REFERENCES users(id),
            FOREIGN KEY (show_id) REFERENCES shows(id))''')
            db.commit()
    markup_inline = types.InlineKeyboardMarkup()
    item_yes = types.InlineKeyboardButton(text='Так', callback_data='yes')
    item_no = types.InlineKeyboardButton(text='Hi', callback_data='no')
    markup_inline.add(item_yes, item_no)
    bot.send_message(message.chat.id, messages[lang]["start"],
reply_markup=markup_inline)
    # запускаем асинхронную функцию в отдельном потоке
    loop = asyncio.new_event_loop()
    t = threading.Thread(target=run_async_function_in_thread, args=(loop,))
    t.start()
def run_async_function_in_thread(loop):
    asyncio.set_event_loop(loop)
    loop.run_until_complete(track_new_episodes())
@bot.message_handler(content_types= ['text'])
def fio_step(message):
    global lang
    user_id = message.from_user.id
    if '📝' in message.text:
        bot.send_message(message.chat.id, messages[lang]["enter_the_name"])
#вводимо назву
        bot.register_next_step_handler(message, name)
    elif '📺' in message.text:
        bot.send_message(message.chat.id, messages[lang]["enter_the_name"])
#вводимо назву
```

```
            bot.register_next_step_handler(message, name_1)
        elif '📜' in message.text:
            bot.send_message(message.chat.id, get_tracked_series(user_id))
        elif '🔔' in message.text:
            try:
                bot.send_message(message.chat.id, update_tracked_series(user_id, results_1[0], results_1[4], series_id, results_1[8], results_1[1], results_1[9], results_1[10]))
            except IndexError:
                bot.send_message(message.chat.id, messages[lang]["a"]) #Вибачте, це кіно не знайдено😔
            except NameError:
                bot.send_message(message.chat.id, messages[lang]["b"]) ##Вибачте, але оновлення фільмів неможливо відслідковувати🙃
        elif '🗑' in message.text:
            bot.send_message(message.chat.id, messages[lang]["delete"])
            bot.register_next_step_handler(message, lambda msg: bot.send_message(user_id, text=remove_tracked_series(user_id, int(msg.text))) if msg.text.isdigit() else bot.send_message(message.chat.id, "Некорректный ввод. Пожалуйста, введите целое число."))
        elif '🆔' in message.text:
            bot.send_message(message.chat.id, user_id)
        elif '🌐' in message.text:
            bot.send_message(message.chat.id, messages[lang]["lang"], reply_markup=language_buttons())
        elif message.text.lower() == 'русский язык':
            lang = "ru-RU"
            bot.send_message(message.chat.id, "выбран русский язык", reply_markup=buttons(lang))
```

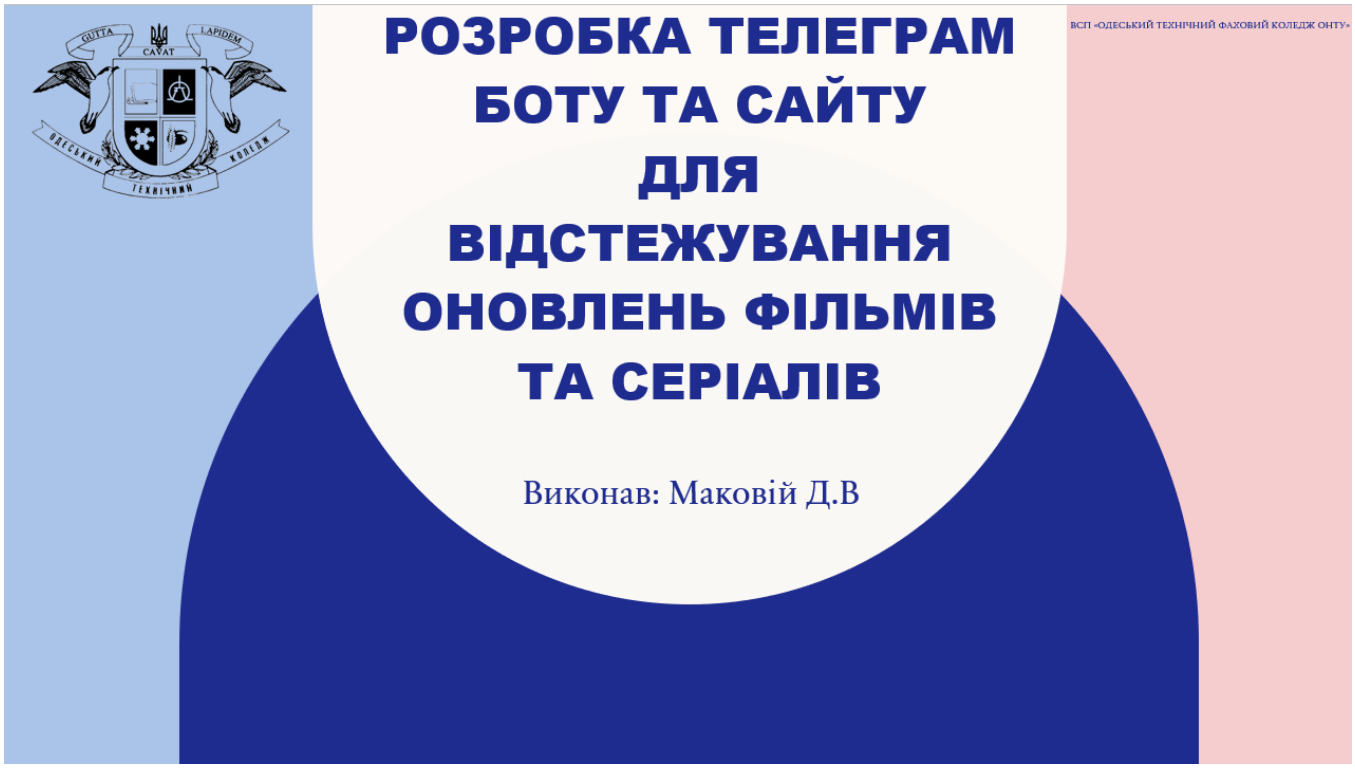
```
        elif message.text.lower() == 'українська мова':
                lang = "uk-UA"
                bot.send_message(message.chat.id, "обрано українську мову",
reply_markup=buttons(lang))
        elif message.text.lower() == 'english':
                lang = "en-EN".
```

Models.py:

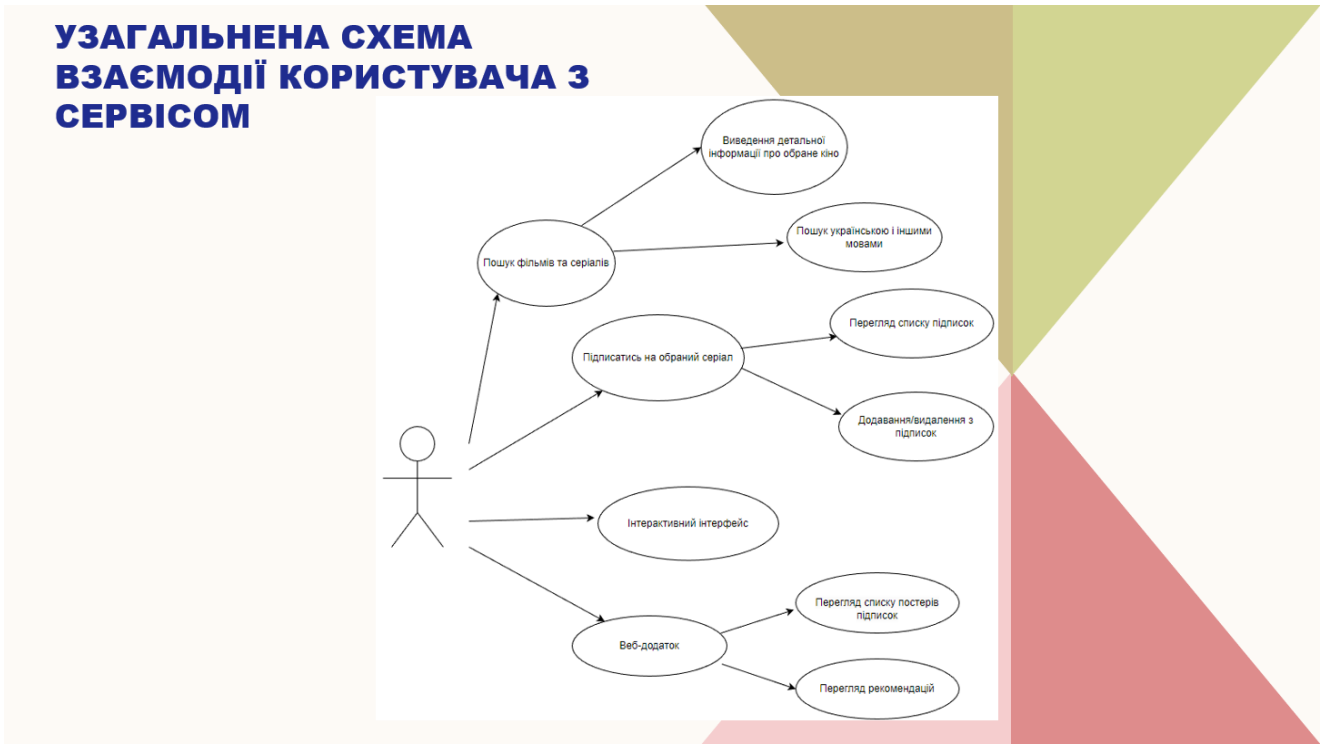
```
from django.db import models
class User(models.Model):
    telegram_id = models.IntegerField()
    class Meta:
        db_table = 'users'
class Show(models.Model):
    name = models.TextField()
    last_episode = models.IntegerField()
    serial_id = models.IntegerField()
    image = models.BinaryField()
    overview = models.TextField()
    namen = models.TextField()
    class Meta:
        db_table = 'shows'
class UserShow(models.Model):
    user = models.ForeignKey(User, on_delete=models.CASCADE)
    show = models.ForeignKey(Show, on_delete=models.CASCADE)

    class Meta:
        db_table = 'users_shows'
```

Додаток Б Графічний матеріал

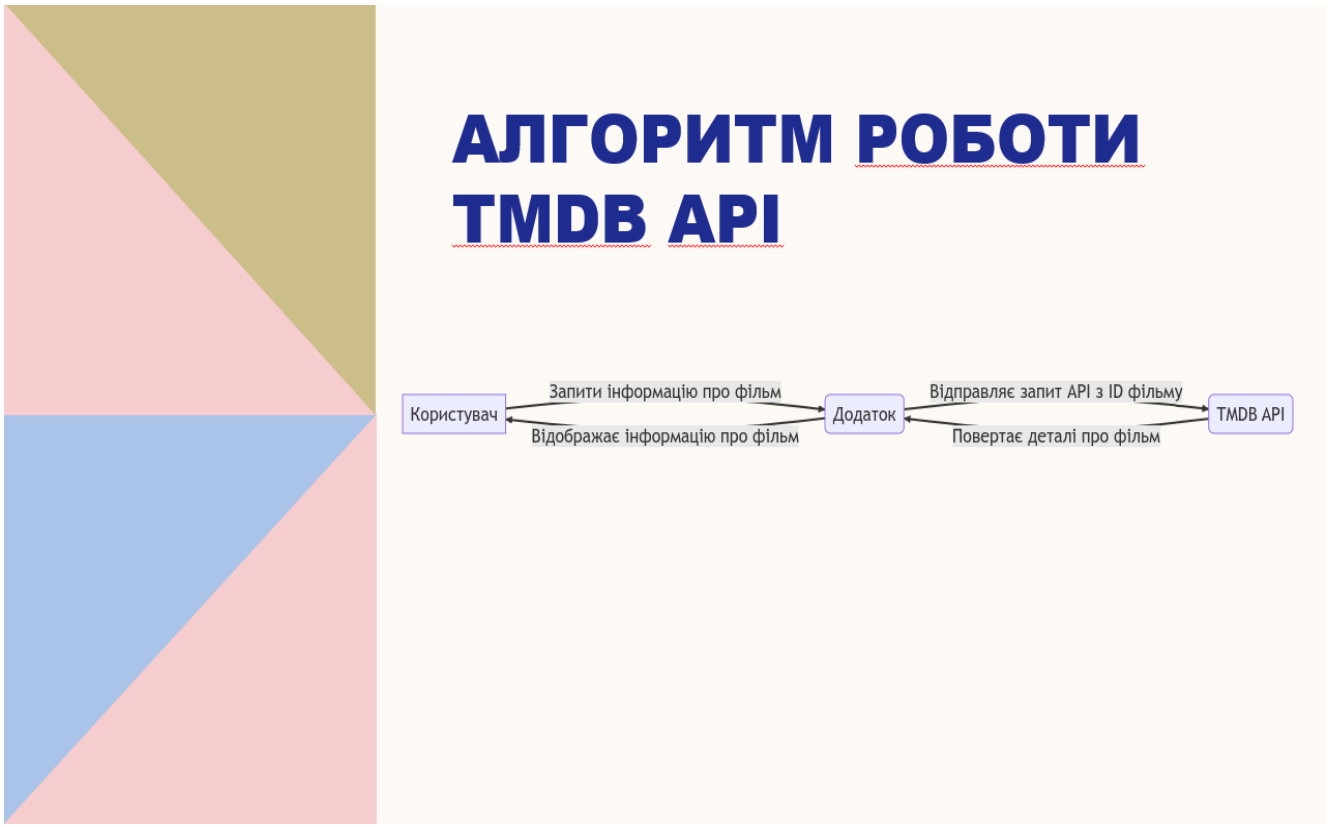
Слайд 1



Слайд 2



Слайд 3



Слайд 4

Слайд 5

ВИВЕДЕННЯ СПИСКУ ПІДПИСОК

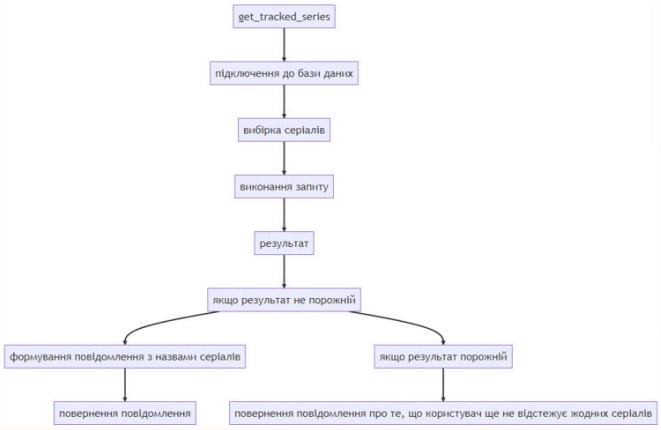
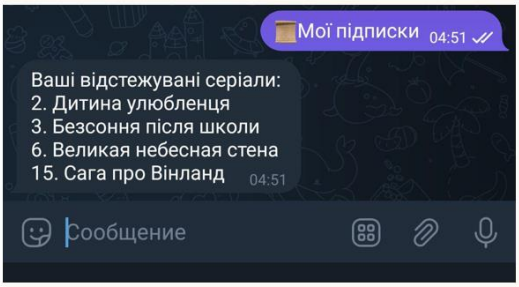


Схема роботи функції “get_tracked_series”

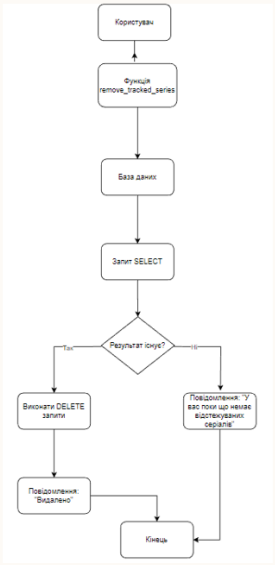


Відображення користувачу підписок

Слайд 6

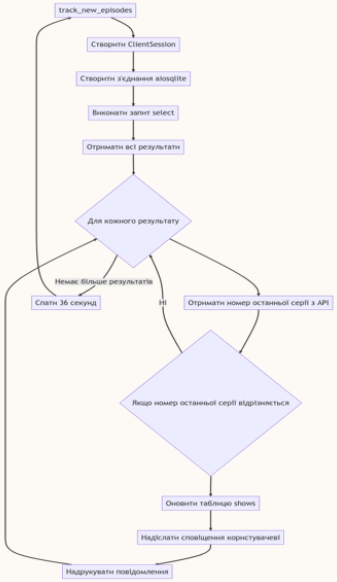
ФУНКЦІЯ “REMOVE_TRACKED_SERIES”

6



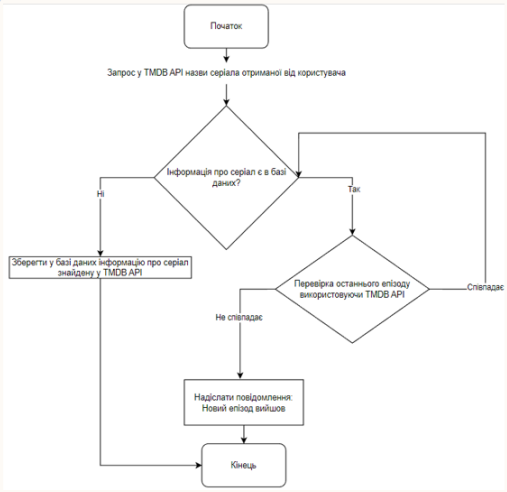
Слайд 7

АЛГОРИТМ РОБОТИ ФУНКЦІЇ ВІДСТЕЖЕННЯ

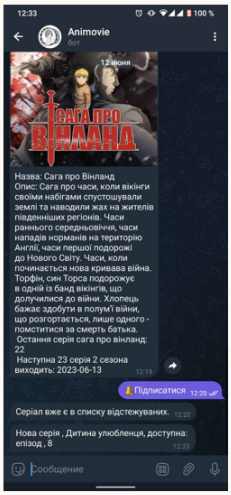


Слайд 8

СИСТЕМА СПОВІЩЕННЯ



Блок-схема роботи системи сповіщення

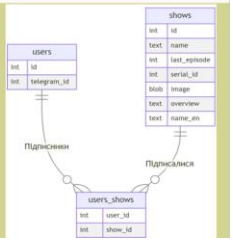


Як це бачить користувач

Слайд 9

СТРУКТУРА БАЗИ ДАНИХ ТА ЇЇ РОБОТА

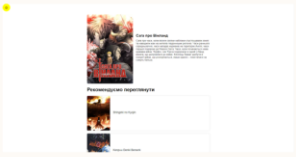
Для збереження даних про користувачів і підписки використовується функція update_tracked_series, яка отримує від користувача його telegram_id і show_id фільму або серіалу, на який він хоче підписатися. Функція перевіряє наявність користувача в таблиці users бази SQLite Database. Якщо користувача немає, то функція додає його в таблицю users. Потім функція додає запис про підписку (з усіма даними, які дістаються за допомогою TMDB API) у таблицю users_shows бази SQLite Database.



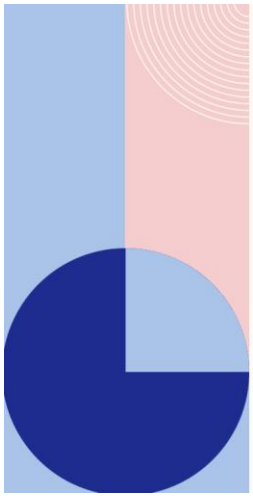
Слайд 10

РОБОТА САЙТУ

Сайт відображає відстежувані серіали для кожного користувача, а також для кожного з відстежуваних серіалів дає рекомендації щодо перегляду чогось схожого



Слайд 11



ВИСНОВКИ

- У рамках дипломної роботи було успішно розроблено та реалізовано комплексний проект, який включає в себе два основні компоненти: чат-бот для пошуку фільмів за різними критеріями та веб-сайт для виведення оновлень фільмів та серіалів, на які підписані користувачі, а також рекомендації стосовно них на основі їхніх переваг. Чат-бот надає зручний інтерфейс для користувачів, де вони можуть швидко знайти актуальні фільми та серіали за жанром, роком, рейтингом, країною тощо, переглянути рекомендації на основі їх вподобань, а також відстежувати оновлення своїх улюблених фільмів та серіалів, отримуючи сповіщення про нові серії. Веб-сайт, з свого боку, надає користувачам можливість переглядати список їх підписаних фільмів та серіалів у зручному форматі, а також отримувати рекомендації щодо них на основі аналогічних факторів.

UNICHECK

Ім'я користувача:
Наталія Вікторівна Копусь

ID перевірки:
1015598145

Дата перевірки:
14.06.2023 11:27:45 EEST

Тип перевірки:
Doc vs Internet + Library

Дата звіту:
14.06.2023 11:40:14 EEST

ID користувача:
100011688

Назва документа: 4РП-06 Маковій Дмитро

Кількість сторінок: 52 Кількість слів: 7991 Кількість символів: 59709 Розмір файлу: 7.37 MB ID файлу: 1015246775

Виявлено модифікації тексту (можуть впливати на відсоток схожості)

1.1%
Схожість

Найбільша схожість: 0.2% з Інтернет-джерелом (https://ask.roboflow.ai/question/20286032)

1.1% Джерела з Інтернету	184	Сторінка 54

Не знайдено джерел з Бібліотеки

0% Цитат

Вилучення цитат вимкнене

Вилучення списку бібліографічних посилань вимкнене

0%
Вилучень

Немає вилучених джерел

Модифікації

Виявлено модифікації тексту. Детальна інформація доступна в онлайн-звіті.

Підозріле форматування	10 сторінок

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

РЕЦЕНЗІЯ

на дипломний проект (роботу) здобувача (здобувачки) освіти

відділення комп'ютерних систем

Маковій Дмитро Вікторович

(прізвище, ім'я та по батькові)

Спеціальність **121 «Інженерія програмного забезпечення»**

Освітня програма **Розробка програмного забезпечення**

Керівник дипломного проекту (роботи) **Томаченко Олександр Євгенович**

(прізвище, ім'я та по батькові)

Тема дипломного проекту (роботи):

Розробка telegram-боту та сайту для відстежування оновлень фільмів та серіалів

Обсяг розрахунково-пояснювальної записки 78 сторінок

Обсяг графічної (презентаційної) частини 11 аркушів (слайдів)

ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЕКТУ (РОБОТИ)

а) заключення про ступінь відповідності виконаного дипломного проекту (роботи) завданню

Дипломний проект відповідає поставленим завданням і має добрий рівень відповідності вимогам, встановленим для даного проекту. Розрахунково-пояснювальна записка та графічна частина проекту майже відповідає запланованим обсягам і якості.

б) характеристика виконання кожного розділу дипломного проекту (роботи)

В технологічному розділі автор детально описав технологічний процес розробки та впровадження чат-бота. Подані алгоритми, методи та інструменти, що використовувалися під час реалізації проекту. Розділ містить достатньо інформації для розуміння технічних аспектів проекту. У економічній частині автор провів економічний аналіз проекту, оцінив його ефективність та рентабельність. У розділі охорони праці, автор описав заходи з охорони праці. Подана оцінка можливих ризиків та пропозиції щодо їх запобігання.

в) оцінка якості виконання пояснювальної записки та графічної частини дипломного проекту (роботи)

Пояснювальна записка містить всі необхідні розділи, досить повно описує хід роботи, виконані етапи проекту та досягнуті результати. Вона структурована, логічна та зрозуміла. Графічна частина проекту включає блок-схеми та ілюстративні матеріали, які належним чином підтримують пояснювальну записку та поліпшують сприйняття матеріалу.

г) перелік позитивних якостей дипломного проекту (роботи) До переваг можна віднести повністю виконану розробку боту, відсутність повноцінних аналогів, детально описаний етап розробки, простота в користуванні розробленого боту, Зручний інтерфейс

д) основні недоліки дипломного проекту (роботи) 1. Алгоритми складено з зауваженнями, 2. Недостатня адаптивність - сайт не пристосований до різних пристроїв та розмірів екранів,

Оцінка розрахункової частини Добре

Оцінка графічної частини Добре

Загальна оцінка Добре

Прізвище, ім'я, по батькові рецензента Василіу Євген Вікторович

Місце роботи і посада рецензента Державний університет інтелектуальних технологій і зв'язку, д.т.н., проф. кафедри КБ та ТЗІ, декан факультету інформаційних технологій та кібербезпеки

Підпис:

« 16 » червня 2023 р.

ПІДПИС ПОСВІДЧУЮ
НАЧАЛЬНИК ВІДДІЛУ КАДРІВ ДУІТЗ

ВІДДІЛ КАДРІВ

ВСП «ОДЕСЬКИЙ ТЕХНІЧНИЙ ФАХОВИЙ КОЛЕДЖ ОНТУ»

ВІДГУК

керівника на дипломний проект здобувача (здобувачки) освіти

відділення комп'ютерних систем

Маковій Дмитро Вікторович

(прізвище, ім`я та по батькові)

Спеціальність: 121 «Інженерія програмного забезпечення»

Освітня програма: Розробка програмного забезпечення

Тема дипломного проекту: Розробка telegram-боту та сайту для відстежування оновлень фільмів та серіалів

ХАРАКТЕРИСТИКА ДИПЛОМНОГО ПРОЕКТУ

а) обсяг і якість виконання проекту (графічного матеріалу і розрахунково-пояснювальної записки) Дипломний проект виконаний в повному обсязі згідно з вимогами і рекомендаціями. Графічний матеріал, включаючи, блок-схеми та інші граф. елементи, представлені чітко і зрозуміло. Розрахунково-пояснювальна записка містить повну і детальну інформацію про проект, включаючи опис проблеми, постановку задачі, методи дослідження та отримані результати. Вона написана задовільно. Пояснювальна записка містить 78 сторінок, Графічний матеріал містить 11 слайдів

б) самостійність роботи над проектом: Студент виявив високий рівень самостійності та ініціативи в роботі над проектом. Він проявив здатність до самостійного пошуку інформації, використання актуальних джерел та методів дослідження.

в) теоретична підготовка випускника (випускниці): Студент виявив задовільні знання у сфері, пов'язаній з дипломним проектом. Він продемонстрував розуміння теоретичних концепцій, методів та моделей, що стосуються проекту. Теоретична база, що була використана в дипломному проекті, є актуальною і відповідає сучасним вимогам.

г) вміння розв'язувати виробничі та конструкторські питання Студент продемонстрував вміння застосовувати теоретичні знання для розв'язання практичних завдань і проблем, пов'язаних з проектом. Він виявив здатність до аналізу та вирішення виробничих та конструкторських питань, включаючи вибір оптимальних рішень та використання інноваційних підходів.

Оцінка розрахункової частини (4)Добре

Оцінка графічної частини (3)Задовільно

Загальна оцінка (4)Добре

Прізвище, ім'я, по батькові керівника дипломного проекту Томаченко Олександр Євгенович

Місце роботи і посада керівника дипломного проекту ВСП «Одеський технічний коледж ОНТУ», Викладач спеціальних дисциплін комісії КП та ПІ

Підпис

«09» 06 2023 р.

ДОЗВІЛ
НА РОЗМІЩЕННЯ
ВИПУСКНОЇ КВАЛІФІКАЦІЙНОЇ РОБОТИ
В ЕЛЕКТРОННОМУ РЕПОЗИТАРІЇ ВСП «ОТФК ОНТУ»

Ми, що нижче підписалися,

Маковій Дмитро Вікторович,
здобувач освіти гр. 4РП-06, та

Томаченко Олександр Євгенович,
керівник дипломного проекту,

не заперечуємо щодо розміщення електронного варіанту пояснювальної записки до випускної кваліфікаційної роботи фахового молодшого бакалавра на тему:

«Розробка Telegram-боту та сайту для відстежування оновлень фільмів та серіалів» (автор роботи – Маковій Д.В., керівник роботи – Томаченко О.Є.)

виконаного у ВСП «Одеський технічний фаховий коледж Одеського національного технологічного університету» в 2023 році, у повному обсязі в електронному репозитарії ВСП «ОТФК ОНТУ» для вільного доступу через мережу Інтернет.

Несемо відповідальність за ідентичність електронного та друкованого варіантів випускної кваліфікаційної роботи, і даємо згоду на обробку персональних даних.

Виконавець _____ / Маковій Д.В. /

Керівник _____ / Томаченко О.Є./

«09» 06 2023 р.