Міністерство освіти і науки України
Університет митної справи та фінансів

Факультет інноваційних технологій
Кафедра комп'ютерних наук та інженерії програмного забезпечення

Кваліфікаційна робота бакалавра

на тему: "Створення боту для завантаження відео на YouTube з оптимізацією та аналітикою"

Виконав: студентка групи ІПЗ20-2

Спеціальність 121 «Інженерія програмного забезпечення»

Зубко Дар'я Анатоліївна
(прізвище та ініціали)

Керівник к.т.н., доцент Чупілко Т.А.
(науковий ступінь, вчене звання, прізвище та ініціали)

Рецензент Університет митної справи та фінансів,
(місце роботи)

доцент кафедри кібербезпеки та інформаційних технологій
(посада)

к.т.н. Савченко Ю.В.

Дніпро – 2024

АНОТАЦІЯ

Зубко Д.А. Створення боту для завантаження відео на YouTube з оптимізацією та аналітикою.

Кваліфікаційна робота на здобуття освітнього ступеня бакалавр за спеціальністю 121 «Інженерія програмного забезпечення». – Університет митної справи та фінансів, Дніпро, 2024.

Дана кваліфікаційна робота присвячена створенню боту для завантаження відео на YouTube з оптимізацією та аналітикою, що полегшують процес завантаження відео та знижує потребу у ручному втручанні. Розглянуті методи та технічні засоби, необхідні для реалізації боту, сформульовані до вимог програмного забезпечення. Було проведено проектування, розробку системи та її тестування. Отримані результати мають практичне значення, оскільки сприяють автоматизації та поліпшенню завантаження контенту, зменшуючи час та ресурси, необхідні для завантаження та оптимізації відео. Створена система є надійною та ефективною, застосовує сучасні технології, такі як штучний інтелект, та є важливим кроком у покращенні роботи з соціальними мережами для інфлюенсерів та просто зацікавлених осіб.

Розроблена система має серверну та клієнтська частини. Система отримує дані про канал на YouTube та показує статистику переглядів та підписників. Цей додаток має зручний дизайн.

Ключові слова: розробка, оптимізація, штучний інтелект, JavaScript.

ABSTRACT

Zubko D.A. Development of bot for uploading videos to YouTube with optimization and analytics.

Qualification work for obtaining the bachelor`s degree in the specialty 121 "Software Engineering." – University of Customs and Finance, Dnipro, 2024.

This qualification work is dedicated to the development of a bot for uploading videos to YouTube with optimization and analytics, which simplifies the video uploading process and reduces the need for manual intervention. The methods and technical means necessary for the imülementation of the bot are considered, and the software requirements are formulated. The design, development, and testing of the system have been carried out. The obtained results have practical significance as they promote the automation and improvement of content uploading, reducing the time and resources needed for uploading and optimizing videos. The created system is reliable and efficient, employing modern technologies such as artificial intelligence, and represents an important step in improving social media management for influencers and interested individuals.

The developed system consists of server and client parts. The system data about the YouTube channel and displays statistics on views and subscribers. This application has a user-friendly design.

Keywords: development, optimization, artificial intelligence, JavaScript.

ЗМІСТ

ВСТУП

Актуальність дослідження. У сучасному цифровому світі автоматизація процесів стає все більш важливою, особливо в контексті створення та управління контентом у соціальних мережах, наприклад YouTube. YouTube є однією з найбільших і найпопулярніших платформ для перегляду відео у світі, який привертає щодня увагу мільярдів користувачів. Він є не тільки засобом для розваг, але й потужним інструментом для навчання, саморозвитку та бізнесу. В умовах сучасного цифрового середовища управління YouTube-каналом стає дедалі складнішим і вимагає значних зусиль для регулярного завантаження відео, їх оптимізації та аналізу ефективності контенту. Це створює потребу у високоефективних інструментів для автоматизації цих процесів.

З кожним роком обсяги контенту на YouTube стрімко зростають, що збільшує конкуренцію серед контент-творців. Для того, щоб залишатися у полі зору та приваблювати нову аудиторію, творці контенту мають забезпечувати регулярне завантаження якісних відео та їх належну оптимізацію. В умовах високої конкуренції важливо не лише створювати цікавий контент, але й ефективно його просувати, це вимагає часу та зусиль, що може бути особливо складним для індивідуальних контент-мейкерів та малих команд.

Автоматизація процесів завантаження та оптимізації відео на YouTube стає необхідним для багатьох користувачів. Вона дозволяє знизити витрати часу на рутинні завдання, зменшити кількість людських помилок та забезпечити більш ефективне управління ресурсами. Автоматизовані інструменти можуть значно полегшити процеси створення та публікації контенту, дозволяючи творцям зосередитися на творчих аспектах своєї роботи. Крім того, інтеграція з аналітичними інструментами надає можливість більш детально аналізувати ефективність відео, що сприяє покращенню контент-стратегій.

Використання сучасних технологій, таких як штучний інтелект, відкриває нові можливості для оптимізації контенту та аналізу даних. Ці технології дозволяють автоматизувати процеси, які раніше вимагали значних зусиль.

Наприклад, штучний інтелект може використовуватися для автоматичного генерування метаданих, аналізу тенденцій та визначення найкращого часу для публікації відео. Це дозволяє створювачам контенту бути більш ефективними та адаптивними до змін у середовищі YouTube.

Мета дослідження. Метою цього дослідження є розробка бота для автоматизованого завантаження відео на платформу YouTube з оптимізацією та аналітикою. Цей бот повинен забезпечити спрощення та прискорення процесу завантаження контенту, мінімізуючи необхідність ручного втручання. Окрім того, що сприятиме підвищенню видимості та залученості контенту. Інтеграція з аналітичними інструментами YouTube дозволить збирати та контенту. Інтеграція з аналітичним інструментами YouTube дозволить збирати та обробляти дані про ефективність відео, що надасть можливість користувачам проводити детальний аналіз і вдосконалювати свої контент-стратегії.

Методи дослідження: аналіз вимог до розробки програмного забезпечення, проектування архітектури додатку, програмування із застосування мови JavaScript, тестування та налагодження розробленого додатку.

У відповідності до поставленої мети в кваліфікаційній роботі ставились та вирішувались наступні завдання дослідження:

1. Аналіз вимог та проектування архітектури системи, що включає в себе вивчення існуючих рішень для автоматизації завантаження відео на YouTube, визначення вимог до функціоналу бота та формування технічного завдання, розробка архітектури системи, включаючи серверну та клієнтську частини.

2. Розробка модулів для завантаження відео та управління метаданими, реалізація функцій для автоматизованого завантаження відео на YouTube, створення алгоритмів для автоматичної генерації та оптимізації метаданих (заголовків, описів, тегів, мініатюр).

3. Інтеграція з аналітичними інструментами YouTube, розробка методів для збору даних про перегляди, лайки, коментарі та інші показники взаємодії, створення інтерфейсу для візуалізації аналітичних даних та надання рекомендацій щодо покращення контент-стратегії.

4. Тестування та оптимізація роботи боту, проведення тестування системи для виявлення та усунення помилок, оптимізація продуктивності та надійності боті, оцінка ефективності роботи бота на основі зібраних аналітичних даних.

5. Впровадження та оцінка практичної значущості системи, випробування ступеня на реальних каналах YouTube, аналіз результатів використання бота для завантаження відео, оптимізації та аналітики, визначення ступеня зниження витрат часу та ресурсів на управління контентом.

Об'єктом дослідження є процес розробки боту для завантаження відео на YouTube з оптимізацією та аналітикою.

Предметом дослідження є програма, з реалізованим веб-додатком для перегляду аналітики на YouTube-каналі. Написання програми та веб-сайту реалізовано в інтегрованому середовищі Visual Studio Code.

Практичне значення отриманих результатів. Створення бота для завантаження відео на YouTube з оптимізацією та аналітикою. Такий бот може стати незамінним інструментом для контент-створювачів, маркетологів та бізнесів, які активно використовують YouTube для просування своїх продуктів та послуг. Він дозволить значно знизити витрати часу та ресурсів на управління контентом, підвищити ефективність роботи та забезпечити кращі результати. Автоматизація цих процесів також сприятиме зростанню аудиторії та покращенню взаємодії з нею, що є ключовим фактором успіху на YouTube.

В результаті виконання кваліфікаційної роботи було набуто фахові компетентності та підтверджені наступні результати навчання, які відповідають Освітньо-Професійній програмі за спеціальністю 121«Інженерія програмного забезпечення»:

1. Здатність застосовувати знання у практичних ситуаціях.
2. Здатність спілкуватися державною мовою як усно, так і письмово.
3. Здатність до пошуку, оброблення та аналізу інформації з різних джерел.
4. Здатність ідентифікувати, класифікувати та формулювати вимоги до програмного забезпечення;

5. Здатність брати участь у проектуванні програмного забезпечення, включаючи проведення моделювання (формальний опис) його структури, поведінки та процесів функціонування;

6. Здатність формулювати та забезпечувати вимоги щодо якості програмного забезпечення у відповідності з вимогами замовника, технічним завданням та стандартами.

7. Володіння знаннями про інформаційні моделі даних та системи, здатність створювати програмне забезпечення для зберігання, видобування та опрацювання даних;

8. Здатність накопичувати, обробляти та систематизувати професійні знання щодо створення і супроводження програмного забезпечення та визнання важливості навчання протягом всього життя;

9. Здатність до алгоритмічного та логічного мислення.

Практичні навички:

1. Аналізувати, цілеспрямовано шукати і вибирати необхідні для вирішення професійних завдань інформаційно-довідникові ресурси і знання з урахуванням сучасних досягнень науки і техніки.

2. Знати і застосовувати відповідні математичні поняття, методи доменного, системного і об'єктно-орієнтованого аналізу та математичного моделювання для розробки програмного забезпечення.

3. Уміння вибирати та використовувати відповідну задачі методологію створення програмного забезпечення.

4. Вміти розробляти людино-машинний інтерфейс.

5. Застосовувати на практиці ефективні підходи щодо проектування програмного забезпечення.

6. Знати і застосовувати методи розробки алгоритмів, конструювання програмного забезпечення та структур даних і знань.

7. Застосовувати на практиці інструментальні програмні засоби доменного аналізу, проектування, тестування, візуалізації, вимірювань та документування програмного забезпечення.

8. Знати, аналізувати, вибирати, кваліфіковано застосовувати засоби забезпечення інформаційної безпеки (в тому числі кібербезпеки) і цілісності даних відповідно до розв'язуваних прикладних завдань та створюваних програмних систем.

Структура роботи:

Розділ 1 Дослідження предметної області. Постановка завдань дослідження. У даному розділі буде виконано пошук та аналіз публікацій стосовно теми завантаження відео на платформу YouTube та методи розробки боту для автоматизації, також ідеї для створення відео.

Розділ 2 Аналіз засобів реалізації створення програми та ідея для пробного контенту на власному YouTube-каналі. У даному розділі буде проаналізовано та обрано технології для реалізації системи завантаження відео та створення власного контенту для тестування.

Розділ 3 Розробка власного боту для завантаження відео на YouTube. У даному розділі буде спроектовано та розроблено програму для завантаження відео на YouTube, а також оптимізація та аналітика контенту.

РОЗДІЛ 1

ДОСЛІДЖЕННЯ ПРЕДМЕТНОЇ ОБЛАСТІ. ПОСТАНОВКА ЗАВДАННЯ ДОСЛІДЖЕННЯ

1.1. Аналіз публікацій щодо розробки боту для завантаження відео

У сучасному цифровому світі відеоконтент стає однією з найпотужніших форм комунікації. Платформи, такі як YouTube, перетворилися на справжні майданчики для глобальних трансляцій, де кожен може знайти свою аудиторію. З кожним днем мільйони людей діляться своїми ідеями, знаннями та творчістю через відео. В епоху, коли відео стає основним носієм інформації, автоматизація процесів створення і розповсюдження контенту є надзвичайно важливою.

Боти для завантаження відео на YouTube стають незамінними помічниками для творців контенту. Вони забезпечують автоматизацію процесів, що значно підвищує ефективність роботи. Автори можуть планувати завантаження відео, оптимізувати час публікацій, автоматично додавати описи і теги. Це не лише економить час, але й дозволяє збільшити продуктивність і регулярність виходу нових відео.

Сучасні боти для завантаження відео на YouTube стали незамінними помічниками для творців контенту завдяки своїй здатності автоматизувати різноманітні процеси, що значно підвищує ефективність роботи. Вони здатні виконувати численні завдання, які раніше вимагали значних витрат часу та зусиль від авторів.

Однією з головних переваг використання ботів є можливість автоматизації процесів. Це означає, що автори контенту можуть налаштувати боти для виконання повторюваних завдань без потреби постійного ручного втручання. Завдяки цьому, творці контенту можуть зосередитися на креативній складовій своєї роботи, залишивши технічні аспекти на ботів.

Автоматизовані системи можуть аналізувати активність аудиторії та рекомендувати найкращий час для публікацій. Це дозволяє авторам налаштувати

свої відео для завантаження у найвигідніші години, що сприяє максимальному охопленню та взаємодії з глядачами.

Розробка боту для автоматизації завантаження відео на YouTube є важливою темою, яка знаходить широке застосування серед контент-мейкерів та маркетологів. Ця тема охоплює кілька важливих аспектів, такі як автоматизація самого процесу завантаження відео, оптимізація метаданих [1] та аналітику ефективності контенту.

Метадані [1] – це структуровані дані, що представляють собою характеристики описуваних сутностей для цілей їх ідентифікації, пошуку, оцінки, управління ними, також це набір допустимих структурованих описів, які доступні в явному вигляді і призначення яких допомогти знайти об'єкт. Призначення метаданих концентрується на пошуку об'єктів, сутностей, ресурсів. За допомогою ботів можна забезпечити стабільну якість та послідовність контенту. Вони допомагають автоматично застосовувати стандартні шаблони для описів, тегів і заголовків, що робить контент більш професійним і привабливим для аудиторії. Крім того, боти можуть проводити аналіз ефективності відео та надавати рекомендації для покращення майбутніх публікацій.

Ідея створення боту для завантаження відео на YouTube з'явилася після перегляду відео [21], у якому розповідалося про створення великого обсягу відео для YouTube та завантаження через сайт Buffer. Цей сайт дійсно комфортний для завантаження контенту у різні соціальні мережі, але потрібно оформляти підписку. Ідея була в тому щоб, тим же способом робити відео, але завантажувати їх безкоштовно, через програму, яка буде створення спеціально для цього.

Існують сервіси для автоматизованого завантаження відео, такі як Buffer [3], які використовуються для автоматизації завантаження контенту та аналітики, але для користування цією платформою потрібно оформлювати підписку, тому ідеєю було створити схожу програму і яка буде безкоштовна для користування. Ідея [4] створення безкоштовного аналога такого сервісу може значно змінити

ринок і надати широкі можливості для тих, хто не має змоги платити за підписку. Основними функціональними можливостями є автоматизація завантаження відео, аналітика та моніторинг, інтеграція з іншими сервісами.

1. Автоматизація завантаження відео

Автоматизація допоможе у плануванні завантаження відео, з-за допомогою якої користувачі можуть налаштовувати розклад, коли вони хочуть завантажити відео на власний YouTube-канал, також надається можливість вирішувати, яку кількість відео одночасно публікувати та у який проміжок часу. З-за допомогою вбудованих функцій можна додати опис, хештеги тощо.

2. Аналітика та моніторинг

З-за допомогою аналітики можна відстежити кількість переглядів, лайків та дізнатися дані про географічне положення глядачів.

3. Інтеграція з іншими сервісами

Програму для завантаження відео можна інтегрувати з відео-сервісом YouTube для покращення процесу використання.

Створення безкоштовного сервісу для автоматизованого опублікування відео та аналітики має великий потенціал зміни цифрового маркетингу та соціальних мереж. Це не лише забезпечує доступ до потужних інструментів для тих, хто не може дозволити собі платні сервіси, але й стимулюватиме конкуренцію та інновації у цій галузі. Таким чином, розробка ботів для завантаження відео на YouTube не тільки актуальна, але й має величезне значення для сучасних творців контенту. Вони стають невід'ємною частиною процесу створення, забезпечуючи ефективність, якість та розширення можливостей для творчого самовираження.

Завдяки автоматизації рутинних завдань, боти значно економлять час авторів. Це дозволяє їм зосередитися на створенні якісного контенту, а не на технічних аспектах його публікації. Як результат, продуктивність підвищується, а регулярність виходу нових відео стає більш стабільною, що позитивно впливає на зростання аудиторії та її залученість.

1.2. Аналіз методів розробки боту для завантаження відео на YouTube та ідеї для створення відео

У сучасному цифровому світі, де відеоконтент стає все більш популярним засобом комунікації, платформа YouTube набуває величезного значення для розповсюдження інформації, розваг та навчання. Розробка ботів для автоматизації завантаження відео на YouTube стає актуальною темою, оскільки дозволяє оптимізувати процеси керування контентом, зменшити кількість рутинних завдань і підвищити продуктивність авторів. Далі будуть розглянуті переваги та недоліки методів розробки ботів для завантаження відео на YouTube.

Метод 1: Використання Google API

YouTube Data API[7] v3 надає широкий спектр функцій для взаємодії з YouTube, включаючи завантаження відео, керування плейлистами, коментарями та іншими аспектами каналу. Для використання API необхідно створити проект у Google Cloud Console, активувати YouTube Data API та налаштувати OAuth 2.0 для авторизації користувачів. Це дозволяє безпечно отримувати доступ до облікових записів користувачів і завантажувати відео від їхнього імені.

API YouTube – це ключ, який забезпечує безпомилковий і надійний обмін даними, а також використання YouTube за межами додатку Google. Зазвичай цей ключ використовується разом з переданням HTTP-запитів і вбудований в заголовки або параметри запиту. API-ключ може обмежувати доступ за певними параметрами, такими як обсяг запитів або типи дозволених операцій.

Для автоматизації завантаження відео одним з ключових аспектів є використання API YouTube для автоматизації процесів, що раніше використовувалися вручну.

OAuth [6] – це відкритий протокол, який був розроблений для захищеної API-авторизації. Зазвичай він використовується на веб-сайтах, щоб надати стороннім користувачам доступ до даних користувача, без постійного запиту паролю. Термін розшифровується як відкрита авторизація.

Переваги цього методу полягають у тому, що отримується офіційна та надійна підтримка з боку Google також надаються широкі можливості управління контентом, включаючи завантаження відео, керування плейлистами та аналітику.

Недоліками є високі вимоги до технічних знань для налаштування API та авторизації також обмеження на використання квот API, що може стати проблемою для великих проектів.

Метод 2: Використання готових бібліотек і фреймворків

Для багатьох мов програмування існують бібліотеки, які значно спрощують роботу з YouTube API [2]. Наприклад, “google-api-python-client” для Python[8] та “googleapis” для Node.js. Ці бібліотеки дозволяють легко інтегрувати функціональність YouTube у додаток.

Перевагами Google API Python Client для Python є легка інтеграція з існуючими Python проектами та велика кількість документації та прикладів коду, що полегшує процес розробки.

Недоліками є необхідність налаштування OAuth 2.0 для авторизації, що може бути складним для новачків та обмежена костомізація в порівнянні з прямим використанням API.

Плюсами використання Google API Client для Node.js є швидка інтеграція з Node.js додатками та підтримка асинхронного програмування, що підвищує ефективність.

Мінуси полягають у необхідності налаштування OAuth 2.0 для авторизації та обмеженій гнучкості у налаштуваннях.

Метод 3: Використання веб-інтерфейсу автоматизації

Інструменти автоматизації веб-інтерфейсу, такі як Selenium, дозволяють автоматизувати процес завантаження відео через веб-інтерфейс YouTube. Цей метод корисний у випадках, коли необхідно обійти обмеження API або виконати дії, які не підтримуються API.

Перевагами є можливість автоматизації завантаження відео без використання API та гнучкість у налаштуванні взаємодії з веб-інтерфейсом YouTube, що дозволяє виконувати більш складніші завдання.

Із недоліків можна помітити меншу стабільність та надійність у порівнянні з використанням API і більша складність у налаштуванні та обслуговуванні автоматизації.

Метод 4: Використання серверних рішень

Розгортання серверного додатка для завантаження відео на YouTube може бути реалізовано на хмарних сервісах, таких як AWS або Google Cloud, або на власному сервері. Це дозволяє створити більш гнучку та масштабовану систему для завантаження відео.

Перевагами є висока костомізація та можливість масштабування системи відповідно до потреб проекту та можливість інтеграції з іншими системами та сервісами, що дозволяють створити комплексте рішення.

Недоліками є високі вимоги до технічних знань налаштування та підтримки серверної інфраструктури та витрати на інфраструктуру та підтримку, що можуть бути значиними для невеликих проектів.

Розробка бота для завантаження відео на YouTube є ключовою задачею для сучасних творців контенту. Вибір методу залежить від специфічних вимог проекту, технічних навичок та доступних ресурсів. Кожен з методів має свої унікальні переваги та недоліки, що потребує ретельного аналізу перед початком реалізації. Зрештою, автоматизація процесів не лише звільняє час для творчої роботи, але й підвищує ефективність управління YouTube-каналом, сприяючи регулярному виходу високоякісного контенту.

Для тестування завантаження відео на YouTube, потрібно також ідеї для створення цих відео, які вони повинні бути, для якої категорії людей вони підходять та на яких мовах буде легше їх просувати.

Для дипломної роботи розглядалося три мови, на яких створювалися потенційні відео, такі як німецька, англійська та українська. Для кожної мови створення контенту є свої плюси та мінуси.

1. Англійська мова є міжнародною мовою спілкування тому створення відео на YouTube цією мовою має як значні переваги, так і певні недоліки.

Перевагами є широка аудиторія, що допомагає охопити велику кількість переглядів від глядачів з різних країн. Це особливо важливо для контенту, який не обмежується географічними рамками також допомагає збільшенню кількості потенційних підписників, оскільки велика частина населення розмовляє англійською.

Якщо розглядати монетизацію на каналі, то це може підвищити вірогідність доходів від реклами, так як більша кількість переглядів може призвести до вищих доходів від рекламних оголошень. Також з глобальною аудиторією є більше можливостей для стабільного доходу від монетизації.

Створюючи контент на англійській мові, легше впроваджувати нові формати та ідеї, які можуть стати популярними.

З мінусів це високий рівень конкуренції, бо англомовний ютуб має багату кількість користувачів, що ускладнює виділення серед інших творців контенту, також багато популярних тем вже покриті великою кількістю каналів, що ускладнює знайти свою нішу.

Мовний бар'єр має теж свої мінуси, для тих, хто не володіє англійською, бо для створення контенту потрібно ще розуміння змісту цього контенту.

Також присутні культурні нюанси, необхідно враховувати культурні відмінності, щоб зробити контент зрозумілим і прийнятним для глобальної аудиторії, можливість непорозумінь через культурні відмінності та особливості гумору.

Вибір на користь англійської мови відкриває двері до ширшої аудиторії, покращених можливостей для монетизації та міжнародної співпраці. Водночас, творці повинні бути готові до високої конкуренції, мовних бар'єрів та культурних нюансів. Оцінка всіх цих факторів допоможе прийняти зважене рішення щодо мовної політики каналу.

2. Німецька мова

Плюсами створення контенту на німецькій мові є те, що німецькомовний контент менш конкурентний у порівнянні з англомовним, що полегшує виділитись серед інших творців та легше привернути увагу цільової аудиторії, яка шукає контент рідною мовою. Глядачі більш схильні до взаємодії з контентом рідною мовою, що підвищує рівень залученості та лояльності, та аудиторія частіше залишає коментарі, ділиться контентом і підписується на канали, що розмовляють їхньою мовою.

Зі сторони монетизації, більш точне таргетування реклами для німецькомовної аудиторії, що може призвести до вищих доходів від реклами та можливості для співпраці з місцевими брендами, які шукають німецькомовних інфлюенсерів.

З мінусів це проблематика збільшення аудиторії за допомогою глядачів з різних країн, так як німецькомовна аудиторія значно менша за англомовну, що може обмежити потенційне охоплення та кількість переглядів. Також менше можливостей для міжнародної співпраці та деякі глобальні тренди та теми можуть бути менш популярними серед німецькомовної аудиторії. Створення контенту на німецькій мові має свої унікальні переваги, такі як менша конкуренція та лояльна цільова аудиторія, що може сприяти успіху. Однак, є і певні виклики, включаючи обмежену аудиторію та менші можливості для міжнародної співпраці. Важливо ретельно зважити ці фактори при виборі мови для створення контенту на YouTube

3. Українська мова.

Створення контенту українською мовою має також свої переваги та недоліки. Перевагами є цілеспрямована аудиторія, яка цінує контент рідною мовою, що підвищує рівень залученості та лояльності. Контент українською мовою створює почуття національної ідентичності та єдності.

Менша конкуренція, порівняно з іншими мовами. У порівнянні з англомовним контентом, українськомовний має меншу конкуренцію, що полегшує виділення серед творців та легше привабити українську аудиторію, яка

шукає локалізований контент. Таж це можливість створювати контент, що враховує національні традиції, культуру та особливості, роблячи його більш релевантним для українських глядачів. Зростання популярності YouTube серед українців відкриває нові можливості для розвитку каналів.

З мінусів, це так як і з німецькою, обмежена аудиторія, що може вплинути на кількість переглядів і підписників. Обмежена кількість навчальних матеріалів та інструментів українською мовою для розвитку каналів, також часто нові тренди та технології спочатку з'являються англійською мовою, що може затримати їх адаптацію в українському сегменті. Та якщо канал має глобальні амбіції, додаткові витрати можуть виникнути через необхідність перекладу та адаптації контенту на інші мови.

Створення контенту на українській мові має значні переваги, включаючи меншу конкуренцію та лояльну аудиторію, яка шанує рідну мову. Однак, є й певні виклики, такі як обмежена аудиторія та менші можливості для монетизації. Важливо ретельно зважити всі плюси та мінуси при виборі мови для створення контенту на YouTube, щоб забезпечити успішний розвиток каналу. Щоб забезпечити успішний розвиток каналу, важливо ретельно досліджувати свою цільову аудиторію, щоб розуміти її потреби та інтереси. Це допоможе створювати контент, який буде затребуваним. Варто ставити акцент на якість, оригінальність та регулярність випуску відео. Це сприятиме зростанню каналу навіть в умовах обмеженої аудиторії. Використовуйте всі доступні канали для просування контенту, включаючи соціальні мережі, співпрацю з іншими блогерами та участь у тематичних заходах.

Згідно з поставленою метою та аналізом методів створення та автоматизації боту для завантаження відео, було вирішено наступні завдання дослідження:

1. Проаналізувати методи та технічні засоби, що застосовуються для створення боту, та обрати необхідні методи реалізації програми.

2. Розробити вимоги до програмного забезпечення для боту та аналітики.

3. Спроектувати та реалізувати програму для завантаження відео та вирішити питання з відео-ідеями для пробних відео завантажень.

4. Провести тестування програми. Проведення тестування програми є критично важливим етапом у процесі розробки, який допомагає забезпечити стабільну роботу програмного забезпечення, відповідність вимогам та відсутність критичних помилок.

1.3. Висновки до першого розділу

У сучасному цифровому світі відеоконтент стає найпотужнішою формою комунікації, що робить автоматизацію процесів його створення та розповсюдження надзвичайно важливою. Боти для завантаження відео на YouTube значно підвищують ефективність роботи контент-мейкерів, дозволяючи оптимізувати час публікацій, автоматично додавати описи та теги, а також забезпечувати стабільну якість та послідовність контенту.

Аналіз методів розробки боту показує, що кожен метод має свої переваги та недоліки, що потребують ретельного аналізу перед реалізацією. Використання Google API, готових бібліотек і фреймворків, веб-інтерфейсів автоматизації та серверних рішень надає різні підходи до досягнення мети.

Розгляд створення контенту на англійській, німецькій та українській мовах показує, що кожна мова має свої унікальні переваги та виклики.

Таким чином, розробка ботів для завантаження відео на YouTube є актуальною та важливою задачею, яка сприяє підвищенню ефективності та якості контенту, забезпечуючи розширення можливостей для творчого самовираження. Вибір методів розробки та мови створення контенту повинен ґрунтуватися на специфічних потребах та можливостях проекту, що забезпечить успішний розвиток YouTube-каналу.

РОЗДІЛ 2

АНАЛІЗ ЗАСОБІВ СТВОРЕННЯ БОТУ ДЛЯ ЗАВАНТАЖЕННЯ ВІДЕО НА YOUTUBE

2.1. Огляд існуючих методів реалізації програми та пошук мови програмування для створення боту

Існує багато методів реалізації боту для завантаження відео на YouTube, наприклад такі, як бібліотеки та фреймворки, інструменти для автоматизації веб-інтерфейсів, використання YouTube Data API.

Використання готових бібліотек і фреймворків значно спрощують роботу з YouTube API [2], так як для багатьох мов програмування вже існують бібліотеки, які дозволяють швидко і легко інтегрувати функціональність YouTube у додаток без необхідності писати багато коду з нуля.

Прикладами бібліотек є Google API Python Client та Googleapis для Node.js. Google API Python Client спрощує інтеграцію YouTube API[2] з Python-додатками. Вона забезпечує простий інтерфейс для виконання API-запитів, таких як завантаження відео та управління плейлистами. Googleapis для Node.js дозволяє легко використовувати YouTube API[2] в JavaScript-додатках. Вона підтримує асинхронне програмування, що підвищує ефективність роботи з API.

Інструменти автоматизації веб-інтерфейсів дозволяють автоматизувати взаємодію з веб-сторінками, включаючи процес завантаження відео на YouTube. Один із найпопулярніших інструментів для цього – Selenium. Selenium дозволяє створювати скрипти, які можуть імітувати дії користувача у веб-браузері, такі як заповнення форм, натискання кнопок та завантаження файлів.

Але більш вдалий варіант для створення боту це – використання YouTube Data API.

YouTube Data API v3 є потужним інструментом для взаємодії з платформою YouTube. Цей API дозволяє автоматизувати процеси завантаження

відео, управління плейлистами, коментарями та іншими аспектами каналу. Щоб скористатися YouTube Data API, необхідно створити проект у Google Cloud Console, активувати YouTube Data API та налаштувати OAuth 2.0 для авторизації користувачів.

Для того, щоб зробити програму, потрібно вирішити, яка мова програмування буде використовуватися при написанні боту. Приклади мов програмування наведено нижче.

- Python

Python є однією з найпопулярніших мов програмування для створення ботів завдяки своїй простоті та великій кількості бібліотек. Python має офіційні бібліотеки для роботи з YouTube API, що значно полегшує процес розробки. Основними бібліотеками є google-api-python-client для інтеграції з YouTube API та Selenium для автоматизації веб-інтерфейсів.

- JavaScript (Node.js)

JavaScript з використанням Node.js дозволяє створювати високопродуктивні та асинхронні додатки. Node.js має чудові бібліотеки для роботи з YouTube API та автоматизації завдань. Основними бібліотеками є googleapis, офіційна бібліотека для роботи з YouTube API, та Puppeteer[12], для автоматизації дій у браузері.

- Java

Java є потужною мовою програмування, яка часто використовується для великих проектів і серверних додатків. Існують офіційні та сторонні бібліотеки для інтеграції з YouTube API. Основними бібліотеками є google-api-java-client, офіційна бібліотека для роботи з YouTube API, та також, як і для Python, Selenium.

- C#

C# використовується для створення додатків на платформі .NET. Це популярний вибір для розробки корпоративних додатків, включаючи ботів для автоматизації завантаження відео. Основними бібліотеками є Google.Apis.YouTube.v3 та Selenium.

- PHP

PHP часто використовується для розробки веб-додатків і скриптів на серверній стороні. Існують бібліотеки для інтеграції з YouTube API, що дозволяє створювати боти для автоматизації завантаження відео. Основними бібліотеками є Google API Client Library for PHP та cURL[15].

- Ruby

Ruby [16], зокрема з використанням фреймворку Ruby on Rails, є відомою мовою програмування для розробки веб-додатків. Існують бібліотеки для інтеграції з YouTube API. Основними бібліотеками є google-api-client[13] та Capybara.

- Go (Golang)

Go, або Golang, є сучасною мовою програмування, яка набирає популярності завдяки своїй продуктивності та простоті. Вона добре підходить для створення високопродуктивних серверних додатків. Основними бібліотеками є google-api-go-client та chromedp.

Вибір мови програмування для створення бота для завантаження відео на YouTube залежить від досвіду, потреб проекту та технічних вимог. Кожна з перерахованих мов має свої переваги та екосистему бібліотек, що полегшує інтеграцію з YouTube API та автоматизацію процесів. Тому у цьому випадку буде вдалим варіантом використовувати JavaScript (Node.js), так як мова JavaScript була більш детальніше вивчена.

JavaScript[17] є поширеною мовою програмування. Вона використовується переважно для додавання динамічних елементів на веб-сторінки. Це можуть бути випадаючі меню або спливаючі вікна. Також веб-розробники часто використовують функції JavaScript для обробки введених користувачами даних. Наприклад, у полях форм за допомогою цієї мови програмування можна перевірити, чи не було зроблено помилкових або нелогічних введень. Зазвичай JavaScript використовується разом з мовою розмітки HTML та мовою стилів CSS.

HTML забезпечує створення основного каркасу веб-сторінки, а за допомогою CSS здійснюється форматування.

Node.js[18] – це середовище виконання для JavaScript, яке не залежить від хост-додатка, такого як веб-браузер. Це означає, що JavaScript можна виконувати на стороні сервера (також відомій як бекенд) і використовувати для розробки серверних скриптів, інструментів та веб-додатків. Завдяки цьому JavaScript стає привабливим для розробників бекенду, адже для таких завдань зазвичай потрібні знання інших мов програмування, таких як PHP[14], Python, Ruby або ASP.NET.

Node.js спрямований на уніфікацію веб-розробки, надаючи можливість використовувати одну мову програмування як на фронтенді, так і на бекенді. Це приносить деякі переваги:

- Можна використовувати ті самі іменні конвенції як на фронтенді, так і на бекенді, що дозволяє мати акуратну та послідовну кодову базу.
- Розробникам більше не потрібні різні інструменти для розробки, вони можуть продовжувати використовувати вже знайомі інструменти.

Це робить його ідеальним для програм, які потребують високу продуктивність при роботі з мережею або файлами. Однопоточна архітектура дозволяє Node.js обробляти численні паралельні запити ефективніше, ніж багатопотокові моделі. Завдяки цьому розробники можуть легко використовувати сторонні рішення та швидко інтегрувати їх у свої проекти. Крім того, Node.js дозволяє створювати серверні додатки різної складності, від простих API до складних веб-додатків. Відкрите та кросплатформне середовище робить Node.js доступним для використання на різних операційних системах, включаючи Windows, macOS і Linux. Завдяки широкому спектру бібліотек та інструментів, Node.js дозволяє розробникам створювати високоефективні додатки з мінімальними зусиллями. Активна спільнота розробників постійно вдосконалює та розширює можливості платформи, що сприяє її швидкому розвитку та підтримці сучасних стандартів JavaScript.

Також, для розробки програми потрібна платформа, яка допоможе у запуску, доставки та розробки додатку у контейнерах, це дозволить ізолювати

додаток і всі його залежності в одному пакеті, який можна подалі відкрити. Для цього зручно буде скористатися Docker.

Docker[19] — це програмна платформа, яка дозволяє швидко створювати, тестувати та розгортати додатки. Docker упаковує програмне забезпечення в стандартизовані одиниці, які називаються контейнерами і містять все необхідне для запуску програмного забезпечення, включаючи бібліотеки, системні інструменти, код і середовище виконання. За допомогою Docker можна швидко розгортати і масштабувати додатки в будь-якому середовищі, будучи впевненими, що код працюватиме.

Використання Docker для створення бота для завантаження відео на YouTube має кілька переваг. Docker забезпечує ізоляцію середовища, простоту розгортання та масштабування додатків, а також полегшує управління залежностями та конфігураціями.

Використання Docker для створення бота для завантаження відео на YouTube спрощує процес розробки, тестування та розгортання, забезпечує консистентність середовища та дозволяє легко масштабувати додаток. Це робить Docker ідеальним інструментом для розробників, які прагнуть створювати стабільні, надійні та легко керовані рішення.

Для створення сайту, для перегляду статистики каналу, буде використано React.js.

React JS це найпопулярніший фронтенд-фреймворк для веб-додатків. Це бібліотека JavaScript з відкритим вихідним кодом для створення користувацьких інтерфейсів (UI, User Interface). Як основа для односторінкових додатків (SPA), використовуються багаторазові UI-компоненти. React JS, або просто React, був розроблений Джорданом Волке, програмним інженером, який працює у Facebook. У 2011 році програмне забезпечення вперше було використане для стрічки новин Facebook, а у 2012 році — для Instagram. Завдяки віртуальному DOM (Document Object Model), React JS мінімізує операції з реальним DOM, що значно підвищує швидкість роботи додатка. Це особливо корисно для складних і високоінтерактивних додатків, де кожна мілісекунда важлива. Однією з

основних особливостей React JS є його компонентний підхід. Кожен елемент інтерфейсу розробляється як окремий компонент, що може бути повторно використаний в різних частинах додатку. Це дозволяє легко підтримувати і масштабувати код, оскільки зміни в одному компоненті не впливають на інші частини додатку. Компоненти можна комбінувати та вкладати один в одного, що робить структуру додатку більш організованою та логічною.

React JS має численні переваги, серед яких:

- Простота у вивченні та застосуванні: Будь-який розробник з досвідом роботи з JavaScript може зрозуміти та застосовувати React JS. Існує багато документації, навчальних матеріалів і туторіалів.
- Спільнотне екосередовище: React JS є бібліотекою з відкритим вихідним кодом, яку використовують і розширюють тисячі користувачів по всьому світу. Розробники, які працюють з React JS, постійно оновлюють бібліотеку, додають нові функції та виправляють помилки. Оскільки React JS був розроблений компанією Facebook, кілька розробників з Facebook постійно займаються оптимізацією коду.
- Економія часу та ефективність: Усі розробники, що використовують React JS, змушені працювати з тим самим кодом. Це гарантує, що при одночасній роботі кількох розробників з React JS не виникатимуть різні версії коду.
- Стабільний код: З React JS інформація завжди передається в одному напрямку, що робить код стабільнішим. Завдяки цьому односпрямованому потоку даних гарантовано, що при оновленні або зміні коду React JS основний код не буде порушений.

Таким чином, React JS є надзвичайно корисним інструментом для створення сучасних веб-додатків. Його компонентний підхід, швидкість, ефективність, гнучкість і велика спільнота роблять його ідеальним вибором для розробників, які прагнуть створювати високоякісні, масштабовані та продуктивні інтерфейси. Використання React JS спрощує процес розробки, забезпечує стабільність коду і дозволяє легко адаптувати додаток до змінюваних вимог користувачів та бізнесу.

Також у написанні програми використовуються файли JSON. Формат JSON (JavaScript Object Notation) використовується для обміну даними між сервером та клієнтом, а також для зберігання структурованої інформації. JSON популярний завдяки своїй легкості та зручності читання, як людиною, так і машиною. Структура JSON відображає об'єкти та масиви у зрозумілому вигляді, що спрощує розробку та налагодження. JSON підтримується багатьма мовами програмування, включаючи JavaScript, Python, Java, C#, Ruby, PHP та багато інших, що робить його ідеальним для передачі даних між різними системами та мовами програмування. JSON використовує пари ключ-значення та списки, що забезпечує його універсальність та простоту використання в різних контекстах. Завдяки своїм перевагам JSON став стандартом для обміну даними в сучасних веб-додатках, API та багатьох інших областях програмування.

2.2. Ідеї та концепт відео для контенту на YouTube-каналі

Створення якісного контенту для YouTube вимагає планування, зйомки, редагування та оптимізації.

Для того, щоб тестувати програму, потрібні короткі відео у великому обсязі. Тестування програми вимагає великого обсягу коротких відео, оскільки такі відео легше створити та швидше завантажити. Відео тривалістю до 60 секунд мають численні переваги, зокрема їх можна оперативно виробляти та публікувати. Короткі відео не тільки спрощують процес тестування, але й дозволяють швидше оцінювати результати та вносити необхідні корективи. Короткі відео до 60 секунд можна швидше створити, також вони завантажуються у форматі YouTube-Shorts, що допомагає швидше поширити контент та привернути увагу глядачів. Короткі відео, тривалість яких не перевищує 60 секунд, можуть бути завантажені у форматі YouTube Shorts. Цей формат спеціально розроблений для короткого контенту, що дозволяє легко захопити увагу глядачів. YouTube Shorts мають окрему стрічку переглядів, що підвищує ймовірність швидкого поширення відео та залучення нових глядачів. Це також

сприяє зростанню популярності каналу, оскільки короткі відео мають вищі шанси бути поміченими та переглянутими великою кількістю людей. Для цього допоможе онлайн-інструмент Canva[20], з-за допомогою цього додатку можна у короткий проміжок часу створити велику кількість відео.

Canva — це сервіс, який дозволяє отримати доступ до величезної бібліотеки онлайн-шаблонів та зображень, які можна налаштовувати, змінювати та зберігати відповідно до потреб і уподобань користувача. Для створення великої кількості коротких відео у стислі строки можна використовувати онлайн-інструмент Canva. Canva пропонує широкий спектр інструментів для редагування відео, що дозволяє створювати професійно виглядаючі ролики без необхідності мати спеціальні знання у сфері відеомонтажу. З Canva можна швидко додати ефекти, музику, текст та анімації до відео, що значно спрощує процес виробництва контенту.

Відео будуть завантажуватися кожні чотири години, тому що YouTube-правила не дозволяють завантажувати більш ніж шість відео на добу.

Концептуально кажучи, є різні теми, які потрібно віднести до певних категорій. У жодному разі ці теми чи категорії не будуть завантажені на один YouTube-канал. Причини: ліміт на завантаження 6 відео щодня.

Структура розподілу відео виглядає так: Основна категорія - Підкатегорія.

Основні категорії:

- Тривія/Факти
- Лайфстайл
- Інформаційна(новини тощо)
- Розваги
- тощо

Наразі наступні теми є успішними для створення оригінального контенту:

- Швидкі факти - Тривія/Факти

Цей формат дозволяє передавати важливу інформацію у невеликих, легкозасвоюваних порціях глядачам.

Причина: Широка різноманітність тем; короткі факти легко поширювати у соціальних мережах, що сприяє розповсюдженню контенту та залученню нових глядачів; безперервний потік контенту.

- Рекомендації книг/фільмів - Лайфстайл

Унікальний та ефективний спосіб поділитися короткими, але захоплюючими рекомендаціями книг та фільмів.

Причина: Рекомендації книг та фільмів надають захоплюючий контент, адже глядачі завжди шукають нові твори для читання та перегляду.

- Мотиваційні відео - Лайфстайл

Ця категорія на YouTube є популярним відеоформатом, який мотивує та надихає глядачів.

Причина: Широка аудиторія; емоції у відео можуть сильно вплинути на глядачів, зробити їх щасливими та спонукати до дій.

- Челенджі – Лайфстайл

Ця категорія може містити веселі челенджі, які розважають людей, або завдання, що допомагають розвивати нові звички.

Причина: Взаємодія з аудиторією; дуже популярний формат на YouTube.

- Страшні або смішні історії - Розваги

У цьому відео розповідаються смішні або страшні історії.

Причина: Фактор розваги; обидва варіанти є цікавими та приваблюють багато нових глядачів на канал.

- Для створення контенту з існуючих відео:

Ностальгічний флешбек - Лайфстайл

Ця категорія нагадує людям про музику або фільми, які були популярні раніше.

Причина: Широка аудиторія, почуття ностальгії.

- Вирізки з подкастів, фільмів, інтерв'ю - Розваги

Цей контент може містити цікаві сегменти з фільмів, подкастів або інтерв'ю.

Причина: Цікавий контент, скорочення матеріалу фільму тощо.

- Переклад популярних відео з однієї мови на іншу

Популярні відео з YouTube перекладаються на інші мови, наприклад, з англійської на німецьку.

Причина: Культурний обмін та залучення більшої кількості глядачів з різних країн.

Перші пробні відео будуть створюватися на німецькій мові, причиною є та, що місце, де зараз створюється программа є Німеччита, таким чином можна визначити тренди, які зараз частно переглядають німці.

Тренди можна відслідковувати за допомогою Google Trends [21]. Цей веб-додаток дозволяє переглядати, що зараз у тренді у всьому світі чи у країні, яка цікавить (див. рис. 2.1).

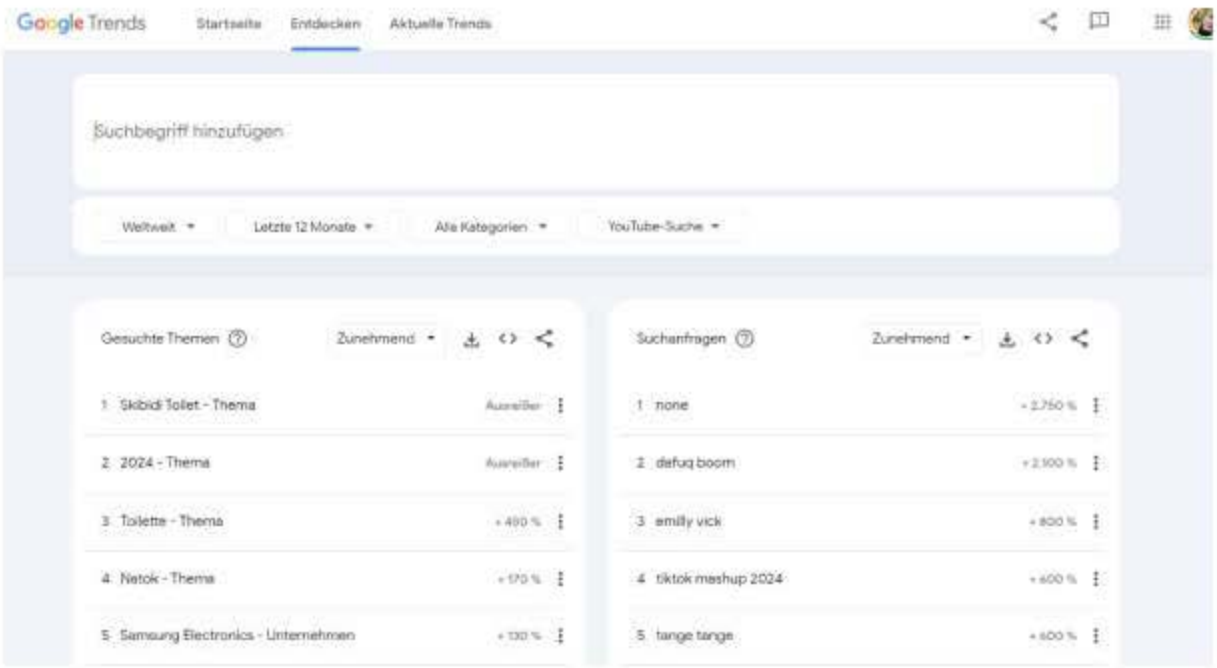

Рисунок 2.1 – Вид сторінки веб-додатку Google Trands

Google Trends є потужним інструментом для аналізу та відстеження популярних пошукових запитів в Інтернеті. Його використання має численні переваги, які роблять його незамінним для дослідників ринку, маркетологів, журналістів та будь-кого, хто цікавиться актуальними тенденціями. Ключовими причинами, чому Google Trends зручно використовувати для перегляду трендів,

це глобальне охоплення, актуальність та реальність даних, простота використання, порівняння пошукових запитів, виявлення сезоних та довгострокових трендів, інтеграція з іншими інструментами та безкоштовний доступ.

Таким чином, Google Trends є зручним та ефективним інструментом для аналізу популярності пошукових запитів, виявлення актуальних трендів та прийняття обґрунтованих рішень у різних сферах діяльності. Його використання допомагає залишатися в курсі останніх тенденцій та краще розуміти потреби та інтереси аудиторії.

2.3. Висновок до другого розділу

В огляді існуючих методів реалізації боту для завантаження відео на YouTube було розглянуто кілька підходів, зокрема використання готових бібліотек і фреймворків, інструментів для автоматизації веб-інтерфейсів та YouTube Data API. Кожен метод має свої переваги, проте використання YouTube Data API виявилося найбільш ефективним і зручним для автоматизації процесу завантаження відео та управління каналом.

Вибір мови програмування для створення бота є критичним і залежить від технічних вимог проекту, наявного досвіду розробників та доступності бібліотек. Серед розглянутих мов (Python, JavaScript (Node.js), Java, C#, PHP, Ruby, Go) кожна має свої переваги. Однак, JavaScript (Node.js) виглядає найбільш перспективним варіантом завдяки своїй популярності, потужним можливостям асинхронного програмування та широкій підтримці в індустрії.

Використання Docker для розгортання додатку забезпечує ізоляцію середовища, простоту масштабування та управління залежностями. Це є критично важливим для стабільної роботи бота та дозволяє швидко розгортати і масштабувати додатки в будь-якому середовищі. React.js, як фронтенд-фреймворк, є ідеальним вибором для створення інтерфейсу для перегляду статистики каналу завдяки своїй простоті та ефективності. Docker є потужним

інструментом, який революціонізував підхід до розробки, тестування та розгортання додатків. Його використання має численні переваги, що робить його надзвичайно корисним для розробників, системних адміністраторів та DevOps-інженерів.

Для тестування програми та створення великої кількості коротких відео зручно використовувати онлайн-інструмент Canva. Короткі відео формату YouTube-Shorts дозволяють швидко залучити аудиторію та отримати зворотний зв'язок, що є важливим для оцінки ефективності бота та контенту.

Розробка контенту для YouTube вимагає ретельного планування та вибору тем. Успішними темами для створення відео є швидкі факти, рекомендації книг і фільмів, мотиваційні відео, челенджі, страшні або смішні історії, ностальгічні флешбеки, вирізки з подкастів, фільмів та інтерв'ю, а також переклад популярних відео з однієї мови на іншу. Ці теми дозволяють створювати контент, який буде цікавий і залучатиме широку аудиторію.

Створення контенту на англійській, німецькій та українській мовах має свої унікальні переваги та виклики. Англійська мова забезпечує доступ до широкої міжнародної аудиторії, проте має високу конкуренцію. Німецька мова менш конкурентна та має лояльну аудиторію, але обмежене охоплення. Українська мова забезпечує лояльну аудиторію та меншу конкуренцію, проте обмежена у розмірі потенційної аудиторії та можливостях монетизації. Вибір мови залежить від цільової аудиторії та стратегічних цілей каналу.

Таким чином, розробка бота для завантаження відео на YouTube є складним, але надзвичайно перспективним завданням, яке дозволяє оптимізувати процеси створення і розповсюдження відеоконтенту. Використання JavaScript (Node.js), Docker, React.js та інструментів для автоматизації допоможе створити надійну та ефективну систему, яка задовольнить потреби сучасних контент-мейкерів. Це дозволить значно підвищити ефективність управління YouTube-каналом, забезпечити стабільний вихід якісного контенту та розширити можливості для творчого самовираження.

РОЗДІЛ 3

РОЗРОБКА БОТУ ДЛЯ ЗАВАНТАЖЕННЯ ВІДЕО НА YOUTUBE З ОПТИМІЗАЦІЄЮ ТА АНАЛІТИКОЮ

3.1. Створення YouTube каналу та відео для пробного завантаження

Ця кваліфікаційна робота присвячена вирішенню задачі автоматизації процесу завантаження відео на YouTube. У ній розглядалися різні методи реалізації бота, зокрема використання YouTube Data API, бібліотек і фреймворків, інструментів для автоматизації веб-інтерфейсів та інших технологій. Після обрання мови програмування та обрання методики створення проекту, потрібно починати з створення каналу на YouTube та створення пробних відео. Створення YouTube каналу займає небагато часу, для цього потрібно мати Google пошту, та зайти на платформу YouTube і натиснути на кнопку “Створити канал”. Після цього з'явиться вікно, у якому потрібно ввести ім’я та псевдонім (рисунок 3.1).

Рисунок 3.1 - Вікно створення каналу на YouTube

Назва для каналу була створена за допомогою штучного інтелекту. Було згенеровано 50 назв, з яких була обрана назва “Logik Lords”. Також за допомогою ШІ було створено логотип для каналу (рисунок 3.2).

Рисунок 3.2 - Логотип каналу

На рисунку 3.3 зображений остаточний вигляд каналу.

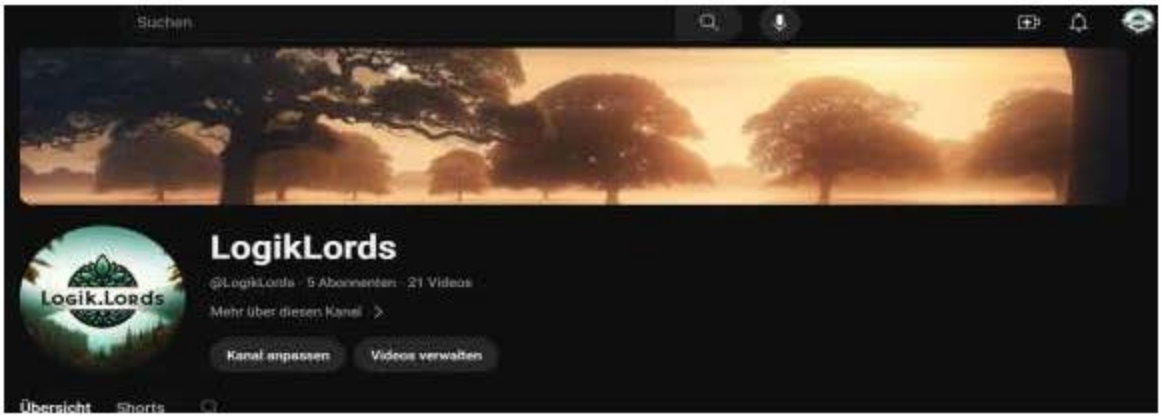

Рисунок 3.3 - Вигляд YouTube-каналу

Наступним кроком було створення відео. Для того, щоб генерувати відео у великому обсязі та за короткий проміжок часу було вирішено використовувати застосунок Canva.

Але перед тим, як почати створювати відео, потрібно вибрати жанр відео. Для пробних відео буде використовуватися тема Опитування (Quiz). За

допомогою Chat GPT згенеровано таблицю (рисунок 3.4), у першій колоні будуть теми для опитування, друга колона – самі питання, третя – відповіді.

Topic	Quiz Question	Answer
Motivation	What superhero analogy represents positive thoughts?	A cheerleader doing victory dances.
Crush Facts	What's the emotional zoo staging a fluttery parade in your stomach during a crush?	Butterflies.
Male Facts	What metaphorical shed might men have in their brains?	A "tool shed" for gadgets.
Girls Facts	What imaginative elements might girls' minds involve when navigating emotions?	Magical forests, unicorns, and storms.
Relationship Facts	What analogy compares relationships to gardens?	Watering them with communication helps love blossom.
Psychology	What's the mental trampoline that makes understanding bounce higher?	Curiosity.
Social Anxiety	What's the trick to overcome social anxiety according to the metaphor?	Realizing most people are too busy worrying about their own performance.
Success	What relay metaphor describes the journey to success?	Passing the baton of hard work and perseverance.
Self-Love	What cozy place is your heart compared to when filled with self-love furniture?	A home.
Curiosity	What explorer's backpack should you pack with curiosity for life's jo ... ?	Your brain.

Рисунок 3.4 - Таблиця з згенерованими ідеями

Після того, як була згенерована таблиця ми переходимо у застосунок Canva, та генеруємо відео для фону.

Обираємо шаблон “Мобільні відео” та переходимо до редагування. У застосунку можна обрати відео для фону, текст та шрифти і т.д (рисунок 3.5).

Після того, завантажуємо таблицю з ідеями та генеруємо відео (було згенеровано 20 відео). Після завершення редагування відео наступним кроком є завантаження таблиці з ідеями. Це може бути таблиця у форматі Excel або Google Sheets, де зберігаються різні ідеї або сценарії для відео. Така таблиця може містити різні теми, тексти, зображення або інші дані, які будуть використані для автоматичної генерації відео.

Застосунок використовує цю таблицю для автоматичного створення кількох відео на основі заданих ідей. У цьому випадку було згенеровано 20 відео.

Це означає, що застосунок автоматично обробив дані з таблиці, застосував шаблон "Мобільні відео" і створив 20 різних відео, кожне з яких відповідає одній з ідей.

Рисунок 3.5 - Приклад фонового відео

3.2. Побудова програми

Першим кроком було створення ключа API.

Щоб створити API-ключ[5] потрібно перейти до Google Cloud Console та створити новий проект. Далі перейти до “YouTube Data API v3” та активувати його для проекту. Подалі для створення API потрібно створити облікові дані для додатку та обрати “OAuth Client ID” та налаштувати його.

Другим кроком була розробка додатку в Visual Studio Code.

Для цього потрібно було запустити Visual Studio Code та створити нову папку для свого проекту та відрити його у VS Code. Для роботи також нам знадобиться Node.js та npm[11].

Node.js – це виконавче середовище для JavaScript, яке дозволяє виконувати код JavaScript за межами веб-браузера. В основному використовується для створення серверних застосунків, але також може використовуватись для розробки інших видів програм. Він базується на движку V8, розробленому Google для браузера Chrome. Завдяки Node.js розробники можуть

використовувати JavaScript для написання серверного коду, що раніше було обмежено виключно клієнтськими браузерами.

npm (Node Package Manager) – це стандартний менеджер пакетів для середовища Node.js. Він дозволяє розробникам легко встановлювати, оновлювати, видаляти та керувати залежностями в їхніх проектах Node.js. npm дозволяє швидко знаходити, встановлювати та використовувати тисячі пакетів JavaScript, які використовуються для розробки різноманітних застосунків та бібліотек. Він входить до складу стандартного пакету Node.js і доступний для використання в терміналі або командного рядку.

З допомогою команди у консолі “npm -v” можна перевірити чи встановлений npm на комп’ютері та команда виводить версію, яка у зараз встановлена (рисунок 3.6).

Рисунок 3.6 - Встановлена версія npm

Далі було написано код, який ініціалізує OAuth2-клієнта з обліковими даними. Цей код генерує URL-адресу, за допомогою якої можна авторизувати свою програму. Після авторизації отримується код, який потрібно ввести в консоль для отримання токену доступу.

Після отримання токену можна вбудувати функціонал для завантаження відео.

Структура проекту — це організована система розташування та взаємодії файлів і компонентів в межах програмного проекту. Вона визначає логічну організацію коду та ресурсів, що спрощує розробку, управління та розширення програми. Добре спроектована структура дозволяє легко знаходити та розуміти код, покращує ефективність розробки, полегшує співпрацю між командами

розробників та забезпечує стабільність та масштабованість проекту. Загалом, структура проекту є важливим інструментом, який сприяє ефективному управлінню кодовою базою та робить розробку програмного забезпечення більш організованою та прозорою. Тут вказана структура проекту:

- Структура проекту.
- models/ - Містить моделі бази даних.
- controllers/ - Містить логіку для обробки операційної системи даних.
- services/ - Містить бізнес-логіку або інтеграції з API.
- utils/ - Допоміжні функції, наприклад, для читання файлів з папки.
- config/ - Конфігураційні файли, наприклад, для з'єднання з базою даних.
- app.js - Точка входу у додаток.

- Моделі. Папка models/ містить класи або функції, які представляють структуру таблиць бази даних та надають методи для основних операцій, таких як додавання, оновлення або запит
  - Приклад (models/video.js)

```
class Video {
constructor(id, path, category, isUploaded) {
this.id = id;
this.path = path;
this.category = category;
this.isUploaded = isUploaded;
}
// Методи для інтеграції з базою даних, на пр. save(), update()
  }
```

- Контролери. У папці controllers/ реалізується логіка, яка обробляє запити і взаємодіє з моделями. Вони є мостом між моделями та логікою застосунку.
  - Наприклад (controllers/videoController.js):

```
const Video = require('../models/video');
```

```
const videoController = {
uploadVideo: async (videoPath) => {
// Логіка для завантаження завантаження відео та актуалізації моделі },
};
```

- Сервіси. У папці services/ можна розміщувати логіку для конкретних завдань, таких як взаємодія зовнішніми API або складніша бізнес-логіка.

- Приклад (services/youtubeService.js)

```
const youtubeService = {
uploadToYouTube: (videoPath) => {
// Логіка для завантаження завантаження відео та актуалізації
  },
};
```

- Допоміжні функції. У папці utils/ можна розміщувати загальні допоміжні функції, такі як функція для читання файлів з папки.

– Приклад (utils/fileReader.js):

```
const fs = require('fs');
const fileReader = {
readUploadFolder: (folderPath) => {
// Читання даних з папки
    },
// Наступні допоміжні функції
};
```

- Конфігурація. У папці config/ розміщуються конфігураційні файли, наприклад для з'єднання з базою даних.

– Приклад (config/database.js):

```
module.exports = {
host: 'localhost',
user: 'мійюзер',
password: 'мійпароль',
database: 'моябазаданих'
```

```
};
```

- app.js – це точка входу у застосунок, де ініціалізується та запускається застосунок.

```
const express = require('express');
const videoController = require('./controllers/videoController');
const app = express();
app.post('/upload', videoController.uploadVideo);
ff
app.listen(3000, () => {
console.log('Server läuft auf Port 3000');
```

На рисунку 3.7 вказаний код, що реалізує бота для автоматичного завантаження відео на YouTube з використанням Google APIs, зокрема YouTube Data API. Основними частинами коду ініціалізація OAuth2-клієнта, завантаження та збереження токену, отримання токену доступу, завантаження контенту відео, завантаження відео на YouTube, запуск зворотнього відліку та запуск програми.

Цей код завантажує відео на YouTube з використанням YouTube API та OAuth2 для автентифікації. Далі йде детальне пояснення кожного блоку коду.

На початку коду виконується console.log('app.js wird gestartet...'), щоб повідомити про запуск програми. Потім імпортуються необхідні бібліотеки: googleapis для роботи з API Google, fs для роботи з файловою системою, readline для взаємодії з користувачем через консоль. Створюється екземпляр OAuth2-клієнта з допомогою google.auth.OAuth2, де треба вставити свої значення для Client ID, Client Secret і Redirect URI.

```
function getRandomVideo() {
    const randomIndex = Math.floor(Math.random() * videoContent.length);
    return videoContent[randomIndex];
}

function getAccessToken(oauth2Client) {
    const authUrl = oauth2Client.generateAuthUrl({
        access_type: 'offline',
        scope: ['https://www.googleapis.com/auth/youtube.upload'],
    });
    console.log('Autorisieren Sie diese App, indem Sie diesen URL besuchen:', authUrl);
    const rl = readline.createInterface({
        input: process.stdin,
        output: process.stdout,
    });

    rl.question('Geben Sie den Code von der Seite hier ein: ', (code) => {
        rl.close();
        oauth2Client.getToken(code, (err, tokens) => {
            if (err) {
                console.error('Fehler beim Einlösen des Autorisierungscodes', err);
                return;
```

Рисунок 3.7 – Код реалізації бота

Далі визначається шлях до файлу з токеном TOKEN_PATH = './token.json'. Функція main читає цей файл, щоб спробувати використати завантажений токен. Якщо файл не існує або в ньому помилка, викликається getAccessToken для отримання нового токена.

Функція loadVideoContent читає файл videocontent.json, що містить дані про відео для завантаження, і зберігає їх у змінній videoContent. Якщо файл не знайдено або виникає помилка при читанні, виводиться відповідне повідомлення та програма завершує роботу.

Функція getRandomVideo повертає випадкове відео з масиву videoContent. Це відео буде використане для завантаження.

Функція getAccessToken генерує URL для авторизації користувача в Google та виводить його у консоль. Користувач повинен відвідати цей URL, отримати код авторизації та ввести його в консоль. Потім програма обмінює цей код на токен доступу, зберігає його у файл token.json та викликає uploadVideo для завантаження відео.

Функція storeToken зберігає отриманий токен у файл token.json.

Функція uploadVideo обирає випадкове відео за допомогою getRandomVideo, створює розширений опис відео з тегами та завантажує його на YouTube з використанням API. Якщо завантаження успішне, виводиться повідомлення з ідентифікатором відео, а також запускається зворотний відлік до наступного завантаження.

Функція startCountdown запускає зворотний відлік, який відображається у консолі. Після завершення зворотного відліку викликається main для завантаження наступного відео.

На завершення викликаються loadVideoContent для завантаження вмісту відео та main для старту програми і першого завантаження відео (рисунок 3.8).

```
function startCountdown(duration) {
    let timer = duration;
    const countdownInterval = setInterval(function () {
        let hours = parseInt(timer / 3600, 10);
        let minutes = parseInt((timer % 3600) / 60, 10);
        let seconds = parseInt(timer % 60, 10);

        hours = hours < 10 ? "0" + hours : hours;
        minutes = minutes < 10 ? "0" + minutes : minutes;
        seconds = seconds < 10 ? "0" + seconds : seconds;

        process.stdout.write("\rNächstes Video wird hochgeladen in: " + hours + ":" + minutes + ":"

        if (--timer < 0) {
            clearInterval(countdownInterval);
            main(); // Rufen Sie main erneut auf, um das nächste Video hochzuladen
        }
    }, 1000);
}

loadVideoContent();
// Starten Sie das Programm zum ersten Mal
main();
```

Рисунок 3.8 – Друга частина коду для реалізації бота

Цей код виконує завдання автоматичного завантаження відео на YouTube, використовуючи OAuth2 для авторизації, вибираючи випадкове відео з набору даних, завантажуючи його на YouTube та відраховуючи час до наступного завантаження. Цей код виконує завдання автоматичного завантаження відео на YouTube, використовуючи OAuth2 для авторизації. OAuth2 є стандартом для авторизації, який забезпечує безпечний доступ до облікових записів користувачів без необхідності передавати паролі. Завдяки цьому, бот може

завантажувати відео на YouTube від імені користувача без необхідності вручну вводити облікові дані кожного разу. Після успішної авторизації, бот вибирає випадкове відео з заданого набору даних. Набір даних може включати різні відеофайли, які попередньо підготували для завантаження. Вибір випадкового відео забезпечує різноманітність контенту та зменшує ймовірність повторного завантаження одного і того ж відео. Для запуску боту потрібно у консолі ввести спочатку команду «npm install» для завантаження всіх залежностей, які є у проекті, та потім ввести команду «npm start» для запуску завантаження відео на YouTube канал (дивитись рисунок 3.9).

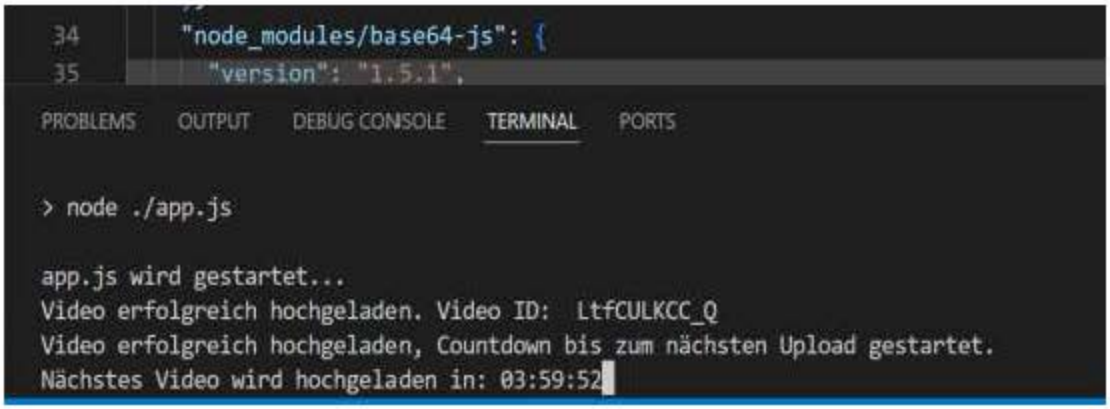

Рисунок 3.9 – Запуск програми

У консолі відображається ID відео, яке було завантажене на канал та таймер, до завантаження наступного відео.

Далі пояснення моделі зв'язків сутностей бази даних. На рисунку 3.10 вказана діаграма моделі зв'язків сутностей для бази даних, що описує структуру та взаємозв'язки між сутностями в системі управління відео контентом.

Основні сутності та їх атрибути:

Video (Відео)

Id: Унікальний ідентифікатор відео.

Title: Назва відео.

Description: Опис відео.

Path: Шлях до файлу відео.

Boolean: isUploaded: Логічний показник, що вказує, чи було відео завантажено.

array: id_tags: Масив ідентифікаторів тегів, пов'язаних з відео.

id_maincategory: Ідентифікатор головної категорії.

id_subcategory: Ідентифікатор підкатегорії.

Main Category (Головна категорія)

Id: Унікальний ідентифікатор головної категорії.

name: Назва головної категорії.

Sub Category (Підкатегорія)

Id: Унікальний ідентифікатор підкатегорії.

name: Назва підкатегорії.

Tags (Теги)

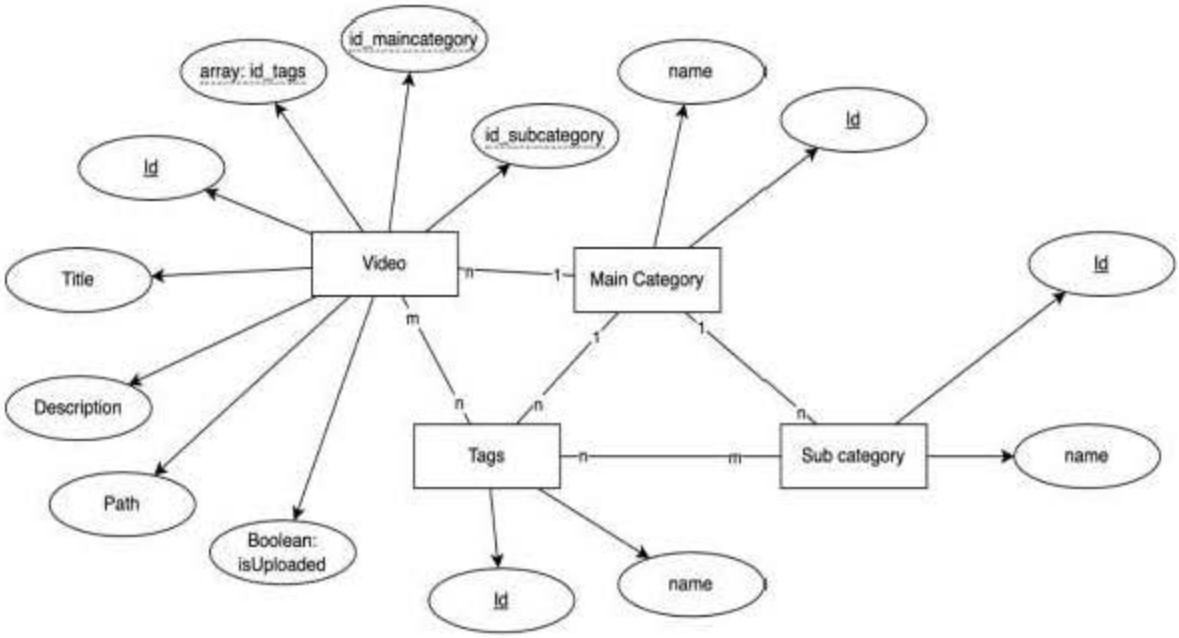

Рисунок 3.10 – Діаграма сутностей зв’язків

Id: Унікальний ідентифікатор тегу.

name: Назва тегу.

Взаємозв'язки між сутностями:

Video має зв'язок "багато-до-одного" з Main Category. Це означає, що кожне відео відноситься до однієї головної категорії, але одна головна категорія може включати багато відео.

Video має зв'язок "багато-до-багатьох" з Tags. Це означає, що кожне відео може мати багато тегів, і кожен тег може відноситися до багатьох відео.

Tags має зв'язок "багато-до-багатьох" з Sub Category. Це означає, що кожен тег може відноситися до багатьох підкатегорій, і кожна підкатегорія може мати багато тегів.

Пояснення зв'язків:

Зв'язок між Video та Main Category: Одне відео може мати одну головну категорію, але одна головна категорія може мати багато відео.

Зв'язок між Video та Sub Category: Одне відео може мати одну підкатегорію, але одна підкатегорія може мати багато відео.

Зв'язок між Main Category та Sub Category: Одна головна категорія може мати багато підкатегорій, але кожна підкатегорія належить до однієї головної категорії.

Ця діаграма допомагає зрозуміти структуру даних та взаємозв'язки між різними елементами в системі управління відео контентом.

Діаграма включає такі основні сутності та їх атрибути: відео, основна категорія, підкатегорія, теги.

- Відео: Репрезентує центральну одиницю в базі даних, яка представляє відео.
  - id: Унікальний ідентифікатор для кожного відео.
  - Title: Назва відео.
  - Description: Текстовий опис того, що показано або міститься у відео.
  - Path: Шлях до файлу, де зберігається відео.
  - Boolean: isUploaded: Логічне значення, яке вказує, чи було відео вже завантажено (істина/хибність).
- Основна категорія: Класифікує відео в одну з основних категорій.
  - id: Унікальний ідентифікатор для кожної основної категорії.

- name: Назва основної категорії.

- Підкатегорія: Додаткова диференціація в межах основної категорії.
- id: Унікальний ідентифікатор для кожної підкатегорії.
- name: Назва підкатегорії.

- Теги: Ключові слова або мітки, які присвоюються відео.
- id: Унікальний ідентифікатор для кожного тегу.
- name: Текст тегу.

Відносини між сутностями.

- Відео до Основної категорії: Кожне відео може бути віднесено тільки до однієї основної категорії. Цей зв'язок представлений як "1 до n", що означає, що одна основна категорія може мати декілька відео.
- Відео до Підкатегорії: Кожне відео може бути віднесено тільки до однієї підкатегорії. Подібно до основних категорій, одна підкатегорія може мати декілька відео. Це також представлено як відношення "1 до n".
- Відео до Тегів: Одне відео може мати декілька тегів, і кожен тег може бути присвоєний кількома відео. Це представлено як відношення "m до n", що означає багато-до-багатьох.
- Основна категорія до Підкатегорії: Кожна основна категорія може мати декілька підкатегорій, але кожна підкатегорія належить тільки до однієї основної категорії. Це представлено як відношення "1 до n".

На рисунку 3.11 вказаний код, який створює сервер Express, ініціалізує базу даних, налаштовує та запускає панель адміністрування AdminJS з автентифікацією. Він забезпечує зручний інтерфейс для керування даними додатку через браузер.

Також, вказаний код, який є конфігураційним файлом Docker Compose, який використовується для налаштування та запуску кількох контейнерів Docker одночасно. У цьому випадку налаштовуються два сервіси: PostgreSQL база даних і Adminer (інструмент для управління базою даних через веб-інтерфейс).

```
import express from 'express';
import AdminJS from 'adminjs';
import { buildAuthenticatedRouter } from '@adminjs/express';

import provider from './admin/auth-provider.js';
import options from './admin/options.js';
import initializeDb from './db/index.js';

const port = process.env.PORT || 3000;

const start = async () => {
  const app = express();

  await initializeDb();

  const admin = new AdminJS(options);

  if (process.env.NODE_ENV === 'production') {
    await admin.initialize();
  } else {
    admin.watch();
  }

  const router = buildAuthenticatedRouter(
```

Рисунок 3.11 – Створення бази даних

Цей код реалізує веб-додаток на основі Express.js, що включає панель адміністратора з використанням AdminJS, та налаштовує базу даних PostgreSQL через Docker Compose. Давайте розглянемо кожну частину коду та його пояснення.

Основний код на Node.js:

Спочатку імпортуються необхідні модулі: Express для створення веб-сервера, AdminJS для побудови адміністративної панелі, модуль для аутентифікації з AdminJS, конфігураційні опції для AdminJS та функція ініціалізації бази даних.

Задається порт для сервера, який береться з змінної оточення або за замовчуванням встановлюється на 3000.

У функції start створюється новий екземпляр Express додатку. Потім викликається функція ініціалізації бази даних. Після цього створюється екземпляр AdminJS з вказаними опціями. Якщо сервер працює в режимі production, викликається метод initialize, в іншому випадку викликається метод watch.

Далі створюється маршрутизатор для AdminJS з функцією аутентифікації, використовуючи налаштування для куків, включаючи пароль та ім'я куків, а також провайдер аутентифікації. Цей маршрутизатор додається до Express додатку.

Нарешті, сервер запускається на заданому порту, і у консолі виводиться повідомлення з посиланням на адміністративну панель.

Docker Compose файл налаштовує два сервіси: PostgreSQL та Adminer.

PostgreSQL: Використовується офіційний образ PostgreSQL. Налаштовується автоматичний рестарт сервісу. Задаються змінні оточення для налаштування користувача, пароля та імені бази даних. Вказується перенаправлення порту 3308 на хост-машині до порту 5432 всередині контейнера.

Adminer: Використовується офіційний образ Adminer для керування базою даних через веб-інтерфейс. Налаштовується автоматичний рестарт сервісу. Вказується перенаправлення порту 8080 на хост-машині до порту 8080 всередині контейнера.

Таким чином, цей код та конфігурація Docker Compose дозволяють розгорнути веб-додаток з адміністративною панеллю та базою даних PostgreSQL, яку можна керувати через веб-інтерфейс Adminer.

Версія Docker Compose 3.1, яка використовується для створення додатку, підтримує більш нові можливості та синтаксис Docker Compose.

Коли запускається Docker Compose за допомогою цієї конфігурації (docker-compose up), Docker створює два контейнери: один для PostgreSQL і один для Adminer. PostgreSQL контейнер налаштовується за допомогою змінних середовища для створення користувача, пароля та бази даних при першому запуску. Adminer контейнер забезпечує веб-інтерфейс для управління базою даних PostgreSQL. Він буде доступний на порту 8080 хост-машини (наприклад, http://localhost:8080). Перенаправлення портів дозволяє вам підключатися до PostgreSQL через порт 3308 хост-машини, а до Adminer через порт 8080. Це

зручний спосіб налаштування середовища для розробки або тестування, оскільки всі необхідні сервіси запускаються одночасно за допомогою однієї команди.

На рисунку 3.12 вказаний код створення веб-додатку для перегляду статистики каналу, де вказується кількість підписників, кількість відео та кількість переглядів. Основним елементом цього веб-додатку є компонент React. Компоненти React дозволяють розробникам будувати багаторазові, ізольовані частини інтерфейсу, які можуть бути легко керовані та змінювані. Для виконання HTTP-запитів і отримання даних з API використовується бібліотека Axios. Axios є популярною бібліотекою, яка спрощує процес здійснення асинхронних запитів до серверів. У цьому випадку компонент відповідає за відображення статистики каналу. Цей код є компонентом React, який використовує бібліотеку Axios для виконання HTTP-запитів та зберігає отримані дані у стані компонента.

```
import React, { useState, useEffect } from 'react';
import axios from 'axios';
import ChannelStats from './components/ChannelStats';
import './App.css';

interface Stats {
  subscriberCount: string;
  viewCount: string;
  videoCount: string;
}

const App: React.FC = () => {
  const [stats, setStats] = useState<Stats | null>(null);
  const [error, setError] = useState<string | null>(null);
  const apiKey = import.meta.env.VITE_YOUTUBE_API_KEY as string;
  const channelId = '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'; // Вставте ваш channelId

  useEffect(() => {
    const fetchChannelStats = async () => {
      try {
        console.log('Отримання даних для каналу:', channelId);
        console.log('Використовується API ключ:', apiKey);

        const url = `https://www.googleapis.com/youtube/v3/channels?part=statistics&id=${channelId}&
        console.log('URL запиту:', url);
```

Рисунок 3.12 – Код створення веб-додатку

Цей код налаштовує React компонент для отримання та відображення статистики YouTube-каналу. Він використовує YouTube API для отримання даних, зберігає їх у стані компонента і відображає результати. Якщо виникає помилка при отриманні даних, вона відображається користувачеві.

Цей код створює простий веб-додаток на основі React, який отримує статистику YouTube-каналу за допомогою YouTube Data API і відображає її. У файлі App.tsx імпортуються необхідні модулі: React, useState та useEffect з react, axios для HTTP-запитів, компонент ChannelStats для відображення статистики та файл стилів App.css. Описується інтерфейс Stats, який містить структуру об'єкта статистики з полями subscriberCount, viewCount та videoCount.

Основний компонент App використовує стан stats для зберігання даних статистики каналу та стан error для зберігання повідомлень про помилки. API ключ береться зі змінних оточення, а channelId є ідентифікатором YouTube-каналу. У useEffect визначається асинхронна функція fetchChannelStats, яка виконує HTTP-запит до YouTube API для отримання статистики каналу. Якщо запит успішний і дані отримані, вони зберігаються в стан stats. Якщо виникає помилка або канал не знайдено, в стан error записується відповідне повідомлення. Компонент рендерить заголовок та компонент ChannelStats, який відображає статистику або повідомлення про помилку.

Файл App.css містить стилі для додатку. Стилі визначають загальні параметри для тексту тіла документа, шрифт для коду, оформлення основного контейнера додатку з вирівнюванням, фоном та кольором тексту, а також стилі для заголовка та контейнера для відображення статистики. Окремі стилі визначають оформлення для кожного пункту статистики.

Файл ChannelStats.tsx містить компонент для відображення статистики каналу. Описується інтерфейс ChannelStatsProps, який містить пропси для компонента, що включають об'єкт stats або null. Компонент ChannelStats приймає об'єкт stats як пропс. Якщо stats дорівнює null, відображається повідомлення "Немає даних для відображення". Якщо stats містить дані, відображається кількість підписників, переглядів та відео.

На рисунку 3.13 зображено вигляд веб-сторінки для аналітики даних YouTube-каналу.

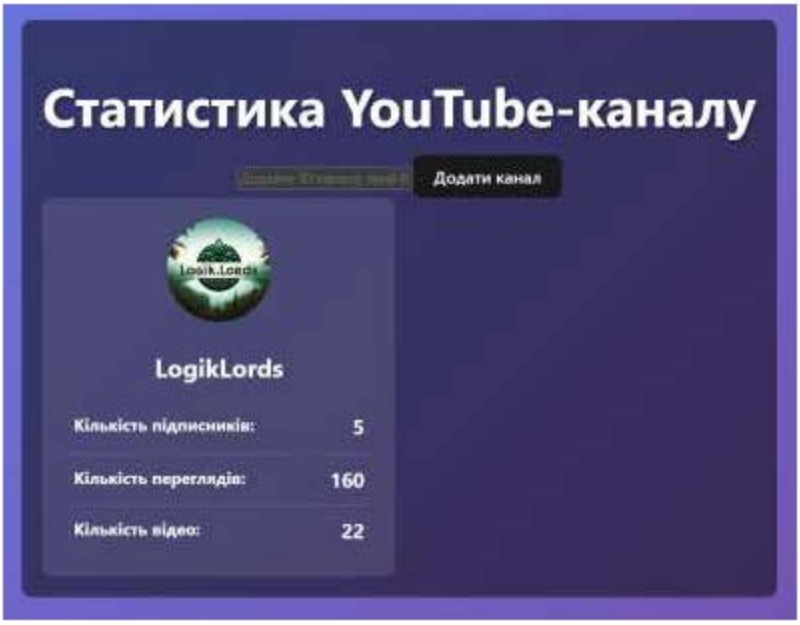

Рисунок 3.13 - Вигляд веб-сторінки для перегляду статистики каналу

Був розроблений веб-додаток для перегляду статистики каналу, де відображаються кількість підписників, кількість відео та кількість переглядів. Цей додаток використовує React для створення користувацького інтерфейсу та Axios для виконання HTTP-запитів. Також на сторінці можна ввести ID-каналу, статистику якого можна передивитися, наприклад канал, який має велику аудиторію, чи якогось популярного YouTube-контентмейкера.

Структура проекту

Структура проекту була організована таким чином, щоб забезпечити логічний розподіл коду та ресурсів, що спрощує управління та розширення програми. Ось детальніше про кожен компонент структури:

1. models/

Ця папка містить моделі бази даних, які визначають структуру таблиць та взаємодію з ними. Моделі використовуються для створення, читання, оновлення та видалення записів у базі даних. Наприклад, модель відео може включати ідентифікатор, шлях до файлу, категорію та статус завантаження.

2. controllers/

Папка controllers/ містить логіку, яка обробляє запити та взаємодіє з моделями. Контролери є посередниками між моделями та рештою програми.

Вони обробляють запити на завантаження відео, оновлення даних і виконують інші операції, необхідні для функціонування додатку.

3. services/

У цій папці знаходиться бізнес-логіка або інтеграція з зовнішніми API. Сервіси містять код, який виконує основні функції програми, такі як завантаження відео на YouTube, управління токенами доступу та інші операції, пов'язані з бізнес-логікою.

4. utils/

Папка utils/ містить допоміжні функції, які використовуються в різних частинах програми. Це можуть бути утиліти для роботи з файлами, обробки даних або інші загальні функції, що спрощують розробку та підтримку коду.

5. config/

У цій папці зберігаються конфігураційні файли, такі як налаштування для підключення до бази даних, параметри авторизації та інші конфігурації, необхідні для роботи програми. Вони дозволяють легко змінювати налаштування без внесення змін до основного коду.

6. app.js

Цей файл є точкою входу в додаток. Він ініціалізує сервер, налаштовує маршрути для обробки запитів та запускає додаток. В app.js підключаються всі необхідні модулі та конфігурації для коректної роботи програми.

JSON (JavaScript Object Notation) використовується для зберігання та передачі даних у структурованому форматі. JSON часто використовується для передачі даних між веб-клієнтами та серверами. Наприклад, коли заповнюється форма на веб-сторінці і натискається "Відправити", дані можуть бути відправлені на сервер у форматі JSON. Сервер обробляє ці дані і може повернути відповідь також у форматі JSON, яку клієнт використовує для оновлення веб-сторінки без перезавантаження. JSON використовується для зберігання конфігураційних файлів у багатьох програмах. Наприклад, файл конфігурації може містити налаштування бази даних, параметри мережі, налаштування користувацького інтерфейсу тощо. JSON підтримується багатьма мовами програмування,

включаючи JavaScript, Python, Java, C#, PHP та інші. Це робить JSON універсальним форматом для обміну даними між різними системами. JSON став стандартом де-факто для обміну даними в сучасних веб-додатках і сервісах завдяки своїй простоті, гнучкості та широкій підтримці. JSON має простий і зрозумілий синтаксис, що робить його легким для читання і написання як людиною, так і машиною. Це також сприяє легкості обробки і маніпуляції даними в різних мовах програмування.

Docker Compose використовувався для налаштування та запуску контейнерів PostgreSQL та Adminer, що спрощує процес управління базою даних через веб-інтерфейс. Docker Compose забезпечує ізоляцію середовища, що означає, що кожен контейнер працює в окремому ізольованому середовищі. Це запобігає конфліктам між різними версіями програмного забезпечення та бібліотек, що можуть виникати під час розробки. Ізоляція дозволяє створювати відтворювані середовища, де додаток може працювати однаково на різних системах. Завдяки Docker Compose процес розгортання додатка значно спрощується. Все, що потрібно зробити, це написати один конфігураційний файл (docker-compose.yml), який визначає всі необхідні контейнери і їх налаштування. Після цього додаток можна розгорнути однією командою: docker-compose up. Це значно зменшує час і зусилля, необхідні для налаштування середовища.

3.3. Висновки до третього розділу

Ця кваліфікаційна робота присвячена автоматизації процесу завантаження відео на YouTube. У ході дослідження розглядалися різні методи реалізації бота, зокрема використання YouTube Data API, бібліотек і фреймворків, а також інструментів для автоматизації веб-інтерфейсів. Після вибору мови програмування та методики створення проекту було здійснено кілька важливих кроків, описаних нижче.

Створення YouTube-каналу та пробних відео

Першим кроком було створення YouTube-каналу та пробних відео. Процес створення каналу включає наявність облікового запису Google і заповнення необхідної інформації на платформі YouTube. Назва каналу "Logik Lords" та його логотип були згенеровані за допомогою штучного інтелекту.

Побудова програми

Наступним кроком було створення ключа API у Google Cloud Console для роботи з YouTube Data API. Цей процес включав створення проекту, активацію API та налаштування OAuth 2.0 для авторизації користувачів.

Розробка додатку виконувалася у Visual Studio Code із використанням Node.js та npm. Node.js забезпечує середовище для виконання JavaScript-коду на серверній стороні, а npm використовується для керування залежностями проекту. Було перевірено встановлення npm та Node.js на комп'ютері, після чого розпочато розробку коду.

Код програми включав ініціалізацію OAuth2-клієнта, генерацію URL-адреси для авторизації, отримання токену доступу та реалізацію функціоналу для завантаження відео на YouTube. Структура проекту була організована таким чином, щоб забезпечити логічний розподіл коду та ресурсів, що спрощує управління та розширення програми.Інтеграція та розгортання

Для інтеграції з базою даних використовувалася модель зв'язків сутностей (ERD). Діаграма ERD описує структуру та взаємозв'язки між сутностями, такими як відео, основні категорії, підкатегорії та теги.

Ця кваліфікаційна робота демонструє ефективність використання сучасних технологій для автоматизації завантаження відео на YouTube. Вибір інструментів, таких як YouTube Data API, Node.js, Docker та React, забезпечує стабільність, продуктивність та легкість у використанні. Результати роботи підтверджують, що автоматизація процесів дозволяє значно підвищити ефективність та якість управління відеоконтентом.

ВИСНОВОК

З метою автоматизації та покращення завантаження відео на платформу YouTube, було проаналізовано та розроблено програму для оптимізації завантаження відео та відстеження аналітики каналу. При створенні програму, враховувались сучасні технології, такі як штучний інтелект, бази даних та клієнт-серверна архітектура.

Дослідження на тему автоматизації завантаження відео на YouTube досі є актуальним з багатьох причин, найпопулярнішими є:

- Зростання популярності відеоконтенту.

Відеоконтент стає все більш популярним форматом інформації в Інтернеті. YouTube, як одна з найбільших платформ для розміщення відео, надає можливість мільйонам користувачів створювати та ділитися своїми відео. Автоматизація цього процесу дозволяє спростити та прискорити завантаження великої кількості відео, що є важливим для творців контенту, які постійно оновлюють свої канали.

- Підвищення ефективності роботи.

Ручне завантаження відео може бути довготривалим та рутинним завданням, особливо для великих каналів, які публікують нові відео щоденно. Автоматизація цього процесу звільняє час і ресурси, які можуть бути використані для створення більш якісного контенту, маркетингу або взаємодії з аудиторією.

- Використання сучасних технологій.

Дослідження та впровадження автоматизації завантаження відео за допомогою YouTube Data API та інших сучасних технологій, таких як Node.js та інструменти для автоматизації веб-інтерфейсів, дозволяє тримати руку на пульсі технологічного прогресу. Це особливо важливо для розробників та компаній, які прагнуть залишатися конкурентоспроможними в сучасному світі.

- Потреба у швидкій адаптації до ринку.

Ринок відео контенту дуже динамічний і вимагає від творців швидкої реакції на тренди та попит аудиторії. Автоматизовані системи дозволяють

швидко адаптуватися до змін та випускати новий контент у найкоротші терміни, що є важливим конкурентною перевагою.

- Інноваційний підхід до контент-маркетингу.

Автоматизація процесів завантаження відкриває нові можливості для контент-маркетингу, включаючи більш персоналізовані та своєчасні публікації, що можуть бути краще налаштовані під потреби аудиторії.

Тому дослідження автоматизації процесу завантаження відео на YouTube є актуальним і важливим як для окремих творців контенту, так і для великих компаній, які прагнуть ефективно управляти своїми ресурсами, зменшувати витрати часу і підвищувати якість та швидкість роботи. Це дозволяє залишатися на передовій технологічного розвитку і успішно відповідати вимогам сучасного світу.

Таким чином у результаті виконання кваліфікаційної роботи була досягнута її мета – автоматизація завантаження контенту та перегляд статистики каналу. У роботі було виконано наступне:

- було проведено пошук та аналіз публікацій стосовно теми завантаження відео на платформу YouTube та методи розробки боту для автоматизації, також ідеї для створення відео.
- проаналізовано та обрано технології для реалізації системи завантаження відео та створення власного контенту для тестування.
- спроектовано та розроблено програму для завантаження відео на YouTube, а також оптимізація та аналітика контенту.

У результаті дослідження були визначені методи та технічні засоби для реалізації програми автоматичного завантаження відео та веб-сторінки для перегляду статистики каналу. У межах практичної частини проекту було успішно реалізовано програму для автоматизації роботи завантаження відео. Проект складався з двох основних компонентів – серверної та клієнтської частини. У серверній частині була розроблена програмна модель, яка завантажує відео, які були додані до програми та завантажує їх у певний часовий проміжок (кожні 4 години). Програма була написана на мові програмування JavaScript з

використанням платформи Node.js, для створення контейнерів, оптимізації розробки та створення бази даних було використано Docker. Веб сторінка для перегляду статистики каналу, таких як кількість переглядів, підписників та кількість відео на каналі, було використано React.js.

Під час тестування системи було підтверджено її відповідність очікуванням. Отримані результати мають практичне значення, оскільки сприяють автоматизації та поліпшенню роботи програми, так як програма потребує подальшої оптимізації та створення інтерфейсу завантаження самих відео для тих користувачів, які не розуміються з мовами програмування, бо запуск та додавання відео для завантаження на даний момент створюється у середовищі Visual Studio Code. На даний момент програма функціонує та готова для використання.

Отже, дослідження та реалізація автоматизації процесу завантаження відео на YouTube є важливим кроком вперед для творців контенту та компаній, що прагнуть залишатися на передовій технологічного розвитку. Автоматизація дозволяє зменшити рутинні задачі, підвищити ефективність роботи та якість контенту, що в кінцевому рахунку сприяє досягненню успіху на сучасному динамічному ринку відеоконтенту.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Документація метадані [Електронний ресурс] – Режим доступу до ресурсу: https://moodle.znu.edu.ua/pluginfile.php/486123/mod_resource/content/1/%D0%9B%D0%B5%D0%BA%D1%86%D1%96%D1%8F%203.pdf

2. Документація YouTube API ключ [Електронний ресурс] – Режим доступу до ресурсу: https://www.ionos.de/digitalguide/websites/web-entwicklung/youtube-api-key/#:~:text=YouTubes%20API%20sorgt%20f%C3%BCr%20einen,Authentifizierungsschl%C3%BCssel%20namens%20YouTube%20API%2DKey.

3. Документація Буффер, сайт-приклад [Електронний ресурс] – Режим доступу до ресурсу: https://buffer.com/

4. Відео-ідея для проекту [Електронний ресурс] – Режим доступу до ресурсу: https://www.youtube.com/watch?v=oM6BfVbJ4OU

5. Документація створення YouTube API ключа [Електронний ресурс] – Режим доступу до ресурсу: https://www.kodi-tipps.de/youtube-api-schluessel-einrichten/

6. Документація OAuth [Електронний ресурс] – Режим доступу до ресурсу: https://blog.hubspot.de/website/oauth

7. Документація API [Електронний ресурс] – Режим доступу до ресурсу: https://wolf-of-seo.de/was-ist/api/

8. Документація поєднання Python та YouTube API [Електронний ресурс] – Режим доступу до ресурсу: https://www.amalytix.com/blog/python-youtube-kanal-recherche/

9. Документація Selenium [Електронний ресурс] – Режим доступу до ресурсу: https://www.selenium.dev/

10. Документація npm [Електронний ресурс] – Режим доступу до ресурсу: https://www.npmjs.com/package/googleapis

11. Документація Puppeteer [Електронний ресурс] – Режим доступу до ресурсу: https://learn.microsoft.com/de-de/microsoft-edge/puppeteer/

12. Документація порівняння Puppeteer та Selenium [Електронний ресурс] – Режим доступу до ресурсу: https://brightdata.de/blog/proxys-101/puppeteer-vs-selenium

13. Документація Google-api-client [Електронний ресурс] – Режим доступу до ресурсу: https://cloud.google.com/java/docs/reference/google-api-client/latest/overview

14. Документація бібліотеки PHP [Електронний ресурс] – Режим доступу до ресурсу: https://developers.google.com/photos/library/guides/get-started-php?hl=

15. Документація cURL [Електронний ресурс] – Режим доступу до ресурсу: https://www.hosteurope.de/blog/was-ist-curl/

16. Документація Ruby email library [Електронний ресурс] – Режим доступу до ресурсу: https://www.mailslurp.com/examples/ruby-email-library-capybara-cucumber-selenium/

17. Документація JavaScript [Електронний ресурс] – Режим доступу до ресурсу: https://www.seo-kueche.de/lexikon/javascript/

18. Документація Node.js [Електронний ресурс] – Режим доступу до ресурсу: https://www.mittwald.de/blog/hosting/was-ist-eigentlich-node-js

19. Документація Docker [Електронний ресурс] – Режим доступу до ресурсу: https://aws.amazon.com/de/docker/

20. Документація Canva [Електронний ресурс] – Режим доступу до ресурсу: https://praxistipps.chip.de/canva-was-ist-das-einfach-erklaert_135353

21. Офіційна веб-сторінка Google Trends [Електронний ресурс] – Режим доступу до ресурсу: https://trends.google.com/trends/explore?gprop=youtube

ДОДАТОК А

```
const { google } = require('googleapis');
const fs = require('fs');
const readline = require('readline');
const OAuth2 = google.auth.OAuth2;
//ініціалізація OAuth2 Client
const oauth2Client = new OAuth2(
  'YOUR_CLIENT_ID', // Замінити Client ID
  'YOUR_CLIENT_SECRET', // Замінити Client Secret
  'YOUR_REDIRECT_URI' // Замінити Redirect URI);
// Авторизація URL
const SCOPES = ['https://www.googleapis.com/auth/youtube.upload'];
const authUrl = oauth2Client.generateAuthUrl({
  access_type: 'offline',
  scope: SCOPES,
});
console.log('Autorisieren Sie diese App, indem Sie diesen URL besuchen:', authUrl);
const rl = readline.createInterface({
  input: process.stdin,
  output: process.stdout,
});
// Отримати токен
rl.question('Geben Sie den Code von der Seite hier ein: ', (code) => {
  rl.close();
  oauth2Client.getToken(code, (err, token) => {
    if (err) return console.error('Fehler beim Abrufen des Access Tokens', err);
    oauth2Client.setCredentials(token)
```