МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

МАТЕМАТИЧНИЙ ФАКУЛЬТЕТ
Кафедра програмної інженерії

КВАЛІФІКАЦІЙНА РОБОТА МАГІСТРА

на тему: **«РЕАЛІЗАЦІЯ ДОДАТКУ АНАЛІЗУ АУДІОДАНИХ НА ОСНОВІ ВЕБТЕХНОЛОГІЙ»**

Виконав: студент 2 курсу, групи 8.1212-іпз-1
спеціальності 121 інженерія програмного забезпечення
(шифр і назва спеціальності)
освітньої програми інженерія програмного забезпечення
(назва освітньої програми)
А.В. Левченко
(ініціали та прізвище)

Керівник доцент кафедри програмної інженерії, доцент, к.ф.-м.н. Горбенко В.І.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Рецензент доцент кафедри комп'ютерних наук, доцент, к.т.н. Борю С.Ю.
(посада, вчене звання, науковий ступінь, прізвище та ініціали)

Запоріжжя – 2023

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ

ЗАПОРІЗЬКИЙ НАЦІОНАЛЬНИЙ УНІВЕРСИТЕТ

Факультет математичний

Кафедра програмної інженерії

Рівень вищої освіти магістр

Спеціальність 121 інженерія програмного забезпечення
(шифр і назва)

Освітня програма інженерія програмного забезпечення

ЗАТВЕРДЖУЮ
Завідувач кафедри програмної інженерії, к.ф.-м.н., доцент

_____ Лісняк А.О.
(підпис)

“ ____ ” _____ 2023 р.

З А В Д А Н Н Я

НА КВАЛІФІКАЦІЙНУ РОБОТУ СТУДЕНТОВІ

Левченку Артему Володимировичу
(прізвище, ім’я та по-батькові)

1. Тема роботи Реалізація додатку аналізу аудіоданих на основі вебтехнологій

керівник роботи Горбенко Віталій Іванович, к.ф.-м.н. доцент
(прізвище, ім’я та по-батькові, науковий ступінь, вчене звання)

затверджені наказом ЗНУ від « 01 » травня 2023 року № 642-с

2. Строк подання студентом роботи 27.11.2023 р.

3. Вихідні дані до роботи 1. Постановка задачі.
2. Перелік літератури.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити)
1. Постановка задачі.
2. Основні теоретичні відомості.
3. Реалізація вебдодатку.

5. Перелік графічного матеріалу (з точним зазначенням обов’язкових креслень)
презентація

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання 03.05.2023 р.

К А Л Е Н Д А Р Н И Й П Л А Н

№	Назва етапів кваліфікаційної роботи	Строк виконання етапів роботи	Примітка
1.	Розробка плану роботи.	15.05.2023	
2.	Обробка методичних та теоретичних	20.06.2023	
	джерел.		
3.	Вибір засобів реалізації.	14.07.2023	
4.	Розробка першого розділу.	28.07.2023	
5.	Проєктування вебдодатку.	14.08.2023	
6.	Створення макетів вебсайту	04.09.2023	
7.	Розробка другого розділу.	25.09.2023	
8.	Розробка вебдодатку.	16.10.2023	
9.	Розробка третього розділу.	10.11.2023	
10.	Оформлення та нормоконтроль	20.11.2023	
	кваліфікаційної роботи.		
11.	Захист кваліфікаційної роботи	14.12.2023	

Студент _____ А.В. Левченко
(підпис) (ініціали та прізвище)

Керівник роботи _____ В.І. Горбенко
(підпис) (ініціали та прізвище)

Нормоконтроль пройдено

Нормоконтролер _____ А.В. Столярова
(підпис) (ініціали та прізвище)

РЕФЕРАТ

Кваліфікаційна робота магістра «Реалізація додатку аналізу аудіоданих на основі вебтехнологій»: 56 с., 42 рис., 4 табл., 18 джерел, 2 додатки.

AUDIO DATA ANALYSIS, DATA VISUALIZATION, DIGITAL SIGNAL PROCESSING, MACHINE LEARNING, REAL-TIME ANALYTICS, SIGNAL PROCESSING, SPECTRAL REPRESENTATIONS, WEB AUDIO API, WEB PLATFORM.

Об’єкт дослідження – використання вебтехнологій для створення додатків аналізу аудіоданих.

Мета роботи: дослідження створення вебдодатків на базі фреймворку Fastapi та взаємодії серверної частини з компонентами фреймворку.

Процес створення вебдодатку триває достатньо довго. Для реалізації проєктів на Python використовують різні фреймворки, які значно полегшують і прискорюють розробку. Фреймворк – це готовий каркас або бібліотека коду. У кваліфікаційній роботі розглянуто методи створення вебдодатків на базі фреймворку Fastapi, а також був проведений огляд і порівняння інших актуальних фреймворків. У практичній частині роботи реалізовано вебплатформу на базі даних зі створенням додатку аналізу аудіоданих.

SUMMARY

Master’s qualification paper «Implementation of the Audio Data Analysis Application Based on Web Technologies»: 56 pages, 42 figures, 4 tables, 18 references, 2 supplements.

AUDIO DATA ANALYSIS, DATA VISUALIZATION, DIGITAL SIGNAL PROCESSING, MACHINE LEARNING, REAL-TIME ANALYTICS, SIGNAL PROCESSING, SPECTRAL REPRESENTATIONS, WEB AUDIO API, WEB PLATFORM.

The object of research is the use of the Audio Data Analysis Application Based on Web Technologies.

The aim of the study is the research on the creation of web applications based on the Fastapi framework and the interaction of the server part with the components of the framework.

The process of creating a web application takes quite a long time. Various frameworks are used to implement Python projects, which greatly facilitate and speed up development. A framework is a ready framework or library of code. In the qualification work, the methods of creating web applications based on the Fastapi framework were considered, as well as a review and comparison of other relevant frameworks was conducted. In the practical part of the work, a web platform based on a database was implemented with the creation of an audio data analysis application.

ЗМІСТ

ВСТУП

У сучасному цифровому середовищі, що швидко розвивається, сфера програмної інженерії пережила зміни у бік розробки складних додатків, керованих даними. До них відносяться програми аналізу аудіоданих. Ці програми, які використовують потужність вебплатформ, знаходяться в авангарді інновацій, відіграють ключову роль у трансформації таких галузей, як музика, охорона здоров’я та розваги.

Поєднання вебплатформ і програм аналізу аудіоданих відкрило можливості, що дозволяють отримувати знання про звук і музику, створювати звукові враження, покращувати доступність для людей з різними можливостями та покращити спосіб взаємодії і сприйняття аудіоконтенту. Незалежно від того, чи йдеться про розшифровку музичних патернів, проведення аналізу звуку на аудіозаписах або сприяння розпізнаванню мовлення в реальному часі, застосування цієї сфери, що розвивається, завжди покращуватиметься.

Вебплатформи надають доступ до інструментів і технологій аналізу аудіоданих, які локалізовано на інших комп’ютерних системах, дозволяючи будь-яким стартапам, дослідникам і організаціям використовувати потужності цих систем та збережені на них аудіодані. Ця доступність породила активну екосистему проєктів з відкритим вихідним кодом, API та бібліотек, що дозволяє розробникам програмного забезпечення створювати власні проєкти і співпрацювати в глобальному масштабі.

Метою дослідження є застосування web платформи для реалізації додатку аналізу аудіоданих.

1 РОБОТА З ВЕБФРЕЙМВОРКОМ FASTAPI

1.1 Аналіз та опис сутностей

Багатьом людям не подобається розробляти вебсайти на PHP, тому що PHP не має гарного менеджера пакетів або найкращої структури для написання вебсайтів.

Статистику всесвітньо відомого сервісу w3techs, показано на рисунку 1.1.

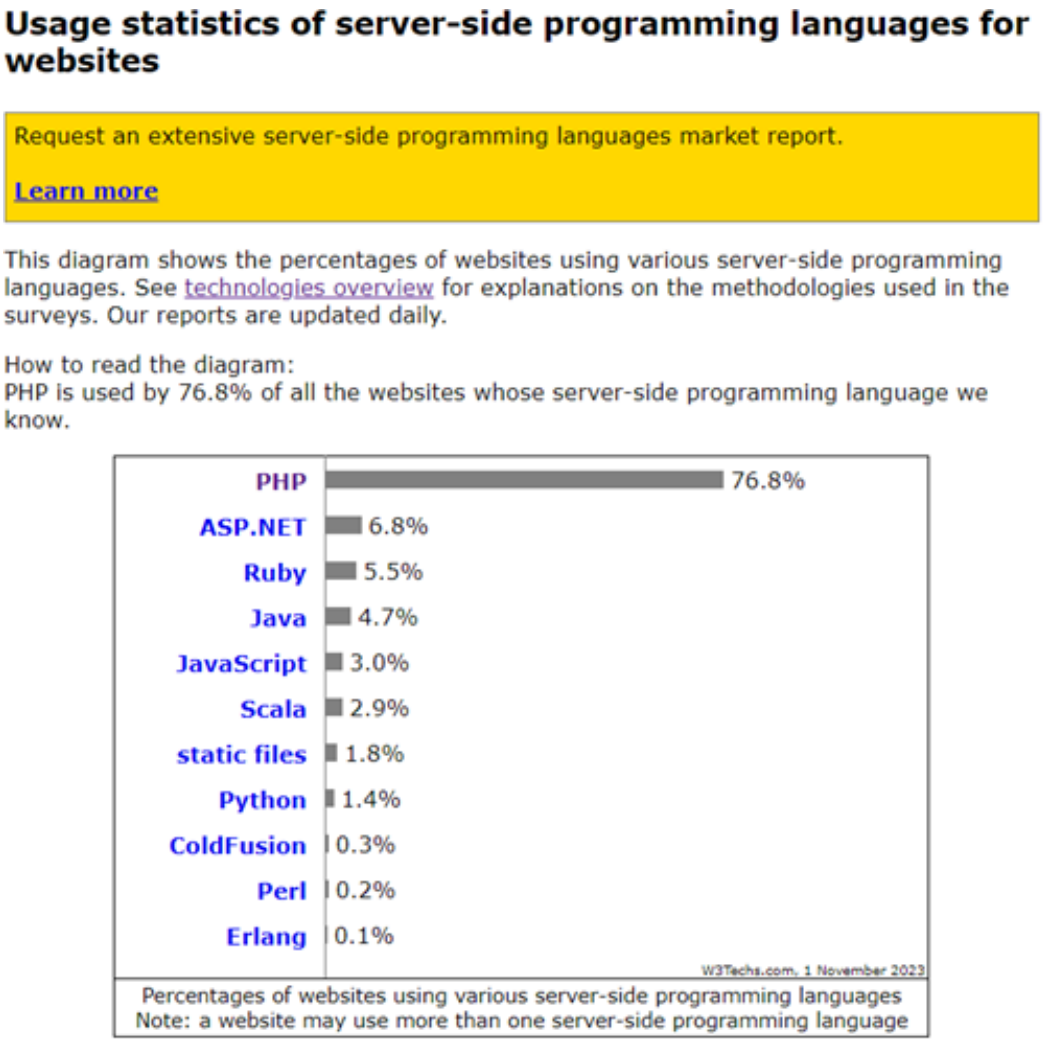

Рисунок 1.1 – Статистика з w3techs [11]

Як можна побачити, близько 77% вебсайтів, які зараз розробляються або знаходяться на стадії розробки, використовують мову PHP. Не дивлячись на

популярність PHP мова, яка буде використовуватися у цій дипломній роботі і є більш зручною з точки зору розробки, це Python. Самі ж темплейти сторінок будуть написані на типовому html.

Для створення зручних сайтів на Python, одним з найпопулярніших є фреймворк FastAPI. Який отримав багато нагород за свою простоту, функціональність і добре написану документацію.

FastAPI – це сучасна, високопродуктивна вебплатформа для створення API за допомогою Python. Він розроблений таким чином, щоб бути простим у використанні, швидким для розробки, і він використовує підказки типу Python 3.7+ для забезпечення перевірки даних і автоматичного створення інтерактивної документації API.

1.2 Фреймворк FastAPI

FastAPI побудований на основі Starlette [12] для вебчастин і Pydantic [13] для аналізу та перевірки даних, що забезпечує високу продуктивність.

Простий у використанні, він поставляється з простим, інтуїтивно зрозумілим API, який дозволяє розробникам створювати API швидко та ефективно.

FastAPI використовує підказки типу Python для перевірки вхідних даних і генерації схеми JSON.

Інтерактивна документація API, автоматично створює інтерактивну документацію на основі типів даних і параметрів функцій, до яких можна отримати доступ через браузер.

Усі блоки коду можна скопіювати та використовувати безпосередньо. Щоб запустити будь-який із прикладів, потрібно перенести код у файл main.py і запустити uvicorn, як показано на рисунку 1.2.

Для наглядного прикладу створення простого коду за допомогою FastAPI рекомендовано писати, переносити, редагувати та запустити код локально.

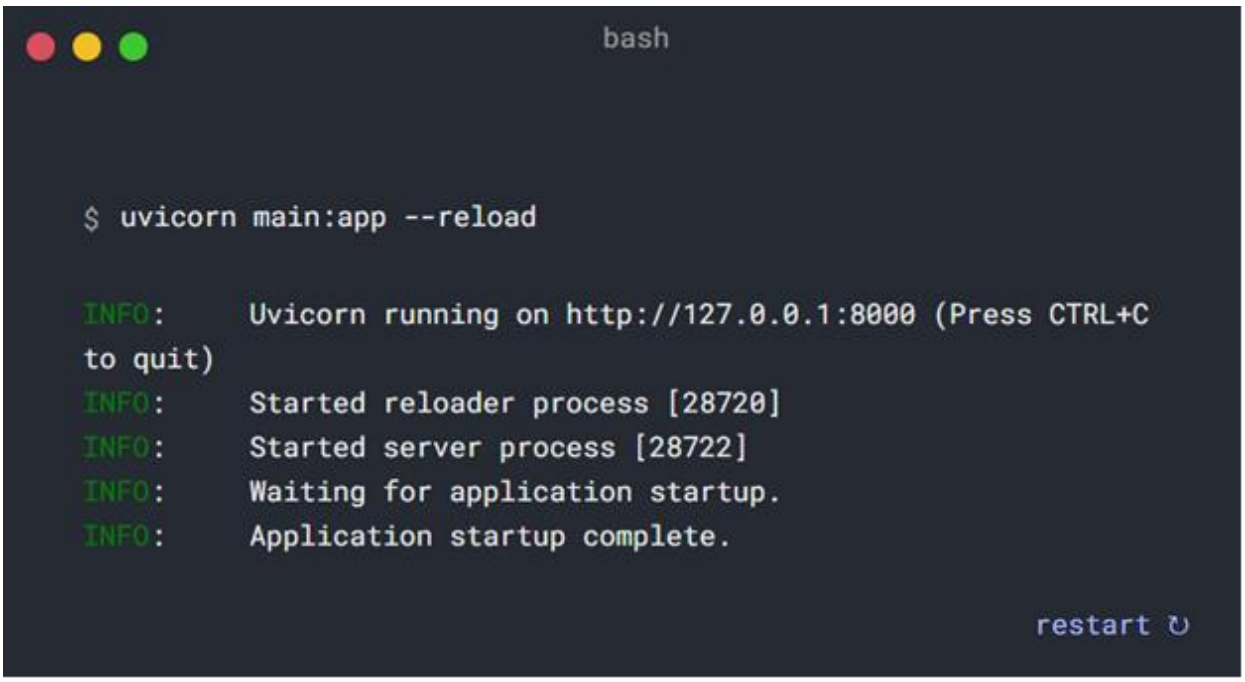

Рисунок 1.2 – Запуск прикладу коду FastAPI

Використання FastAPI у редакторі – це те, що дійсно показує переваги FastAPI, побачивши, як мало коду потрібно, щоб написати усі перевірки типів, автозавершення тощо.

Першим кроком є встановлення FastAPI. Потрібно встановити його з усіма додатковими залежностями та функціями, як показано на рисунку 1.3.

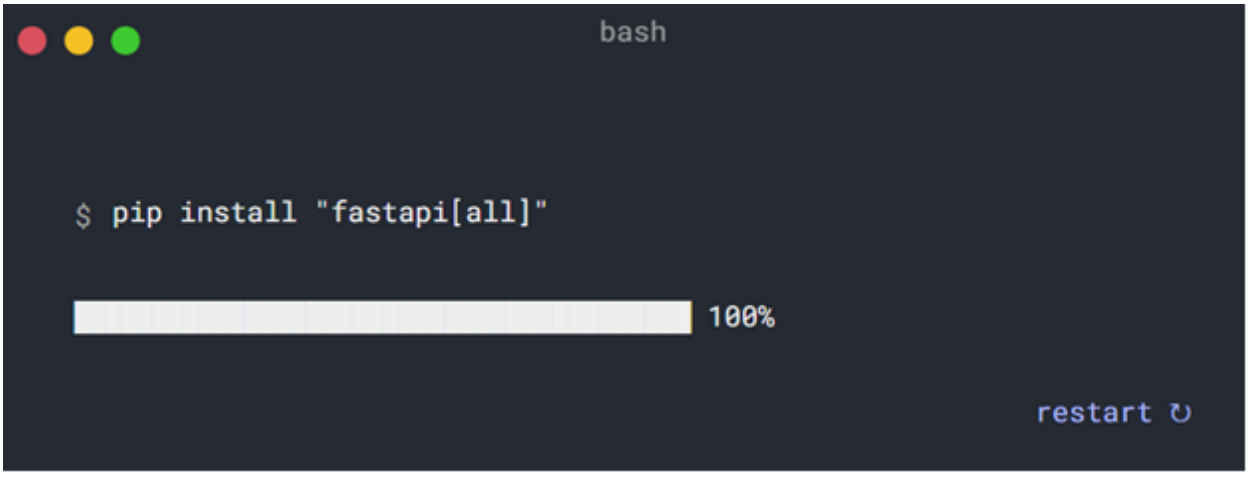

Рисунок 1.3 – Запуск прикладу коду FastAPI

Для наглядного прикладу створення простого коду за допомогою FastAPI рекомендовано писати, переносити, редагувати та запустити код локально.

Використання FastAPI у редакторі – це те, що дійсно показує переваги FastAPI, побачивши, як мало коду потрібно, щоб написати усі перевірки типів, автозавершення тощо.

Першим кроком є встановлення FastAPI. Потрібно встановити його з усіма додатковими залежностями та функціями, як показано на рисунку 1.4.

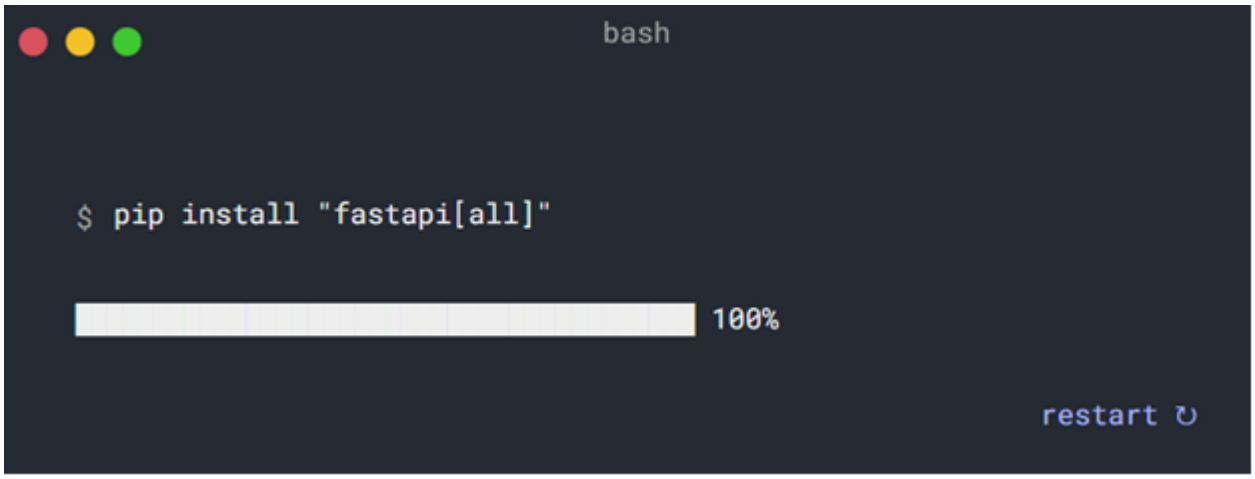

Рисунок 1.4 – Встановлення FastAPI

Щоб використовувати вбудовані можливості для запуску API-сервера, треба встановити сервер ASGI, наприклад Uvicorn, який може запускати програми FastAPI.

FastAPI пропонує вичерпну документацію, яка охоплює різні аспекти фреймворку, зокрема репозиторій GitHub.

Вихідний код і відстеження проблем для FastAPI можна знайти в його репозиторії GitHub: FastAPI GitHub [19].

Офіційна документація містить посібник користувача з підручниками, прикладами та поясненнями щодо різних функцій.

FastAPI має активну спільноту на GitHub та інших платформах, де можна поставити запитання та отримати підтримку.

Доступно кілька розширень та інтеграцій для розширення функцій FastAPI, і вони часто задокументовані або в офіційній документації, або у відповідних репозиторіях.

1.3 Бібліотека pyAudioAnalysis для Python з відкритим вихідним кодом для аналізу звукових сигналів

Зростаюча доступність аудіовмісту завдяки широкому розповсюдженню каналів призвела до потреби в системах, здатних автоматично аналізувати цей вміст. Залежно від окремих типів каналів розповсюдження, типів аудіокласів (мовлення, музика тощо), існування інших засобів масової інформації (наприклад, візуальна інформація) та вимог до конкретної програми протягом останнього періоду з'явився широкий спектр різноманітних програм: пошук музичної інформації, виявлення аудіоподій, аналіз мовлення та динаміка, розпізнавання мовних емоцій, мультимодальний аналіз тощо. Метою бібліотеки pyAudioAnalysis є надання широкого спектру функціональних можливостей аналізу аудіо через просте у використанні та комплексне програмування дизайну.

Бібліотеку pyAudioAnalysis можна використовувати для виділення аудіофункцій, навчання та застосування аудіокласифікаторів, сегментації аудіопотоку за допомогою контрольованих або неконтрольованих методологій і візуалізації зв'язків вмісту. Бібліотека написана мовою Python, яка є мовою програмування високого рівня, яка протягом останніх кількох років привертає все більший інтерес, особливо в академічній та науковій спільноті.

На рисунку 1.5 показано кілька скріншотів використання бібліотеки pyAudioAnalysis.

У бібліотеці реалізовано кілька аудіофункцій як з часової, так і з частотної області.

Контрольовані знання використовуються для навчання класифікаторів. Процедура перехресної перевірки також реалізована для того, щоб оцінити оптимальний параметр класифікатора (наприклад, параметр вартості в Support Vector Machines або кількість найближчих сусідів, що використовуються в класифікаторі kNN). Результатом цієї функції є модель класифікатора, яку можна зберегти у файлі. Крім того, у цьому контексті також надаються оболонки, які класифікують невідомий аудіофайл.

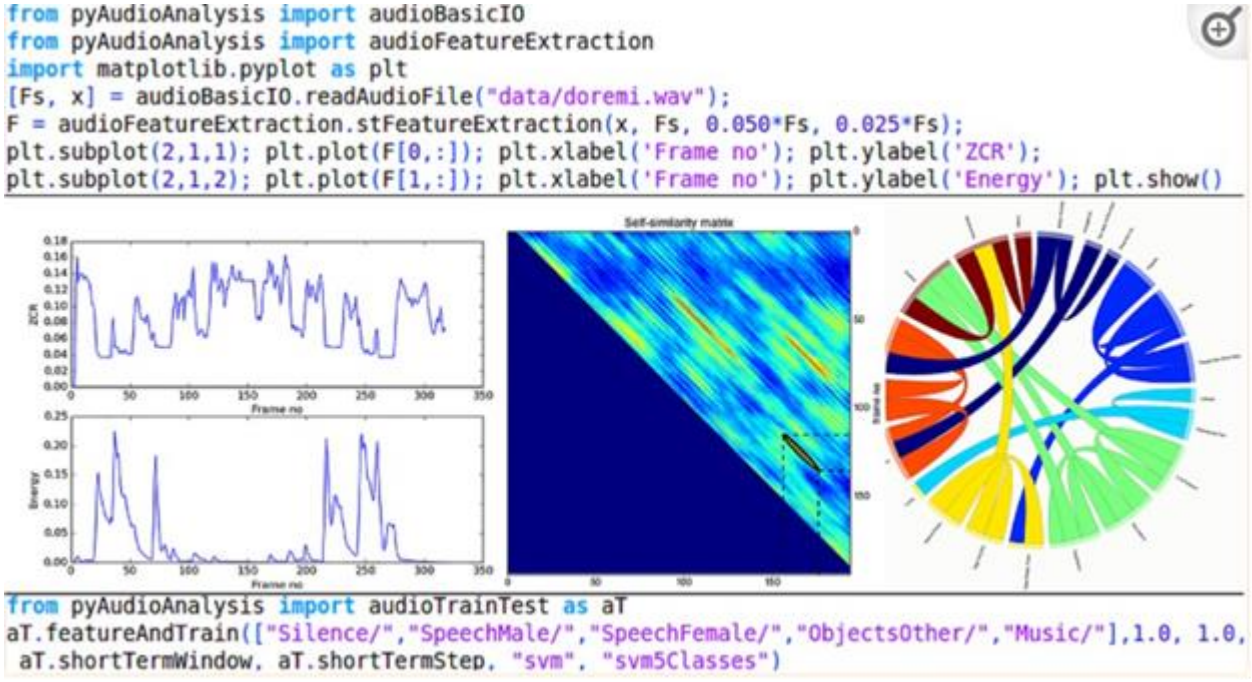

Рисунок 1.5 – Використання бібліотеки pyAudioAnalysis [15]

Моделі, які відображають аудіофункції на змінні з реальними значеннями, також можна навчити в контрольованому контексті. Знову ж таки, перехресна перевірка реалізована для оцінки найкращих параметрів регресійних моделей.

У бібліотеці реалізовано наступні контрольовані або неконтрольовані завдання сегментації: сегментація та класифікація фіксованого розміру, видалення мовчання, щоденник мовця та ескізи аудіо. За потреби навчені моделі використовуються для класифікації аудіосегментів за попередньо визначеними класами або для оцінки однієї чи кількох змінних, які вивчаються.

За наявності колекції аудіозаписів pyAudioAnalysis можна використовувати для отримання візуалізацій зв’язків вмісту між цими записами.

Автоматична тактна індукція, тобто задача визначення темпу музичних ударів у часі, є досить важливою задачею, особливо для прикладних програм пошуку музичної інформації. У цій бібліотеці реалізовано простий підхід для розрахунку темпу. Він використовує процедуру виявлення локальних максимумів (див. рис. 1.6), застосовану до набору короткочасних послідовностей ознак. Агрегована гістограма (див. рис. 1.7) часових відстаней між послідовними локальними максимумами також обчислюється, і її максимальний елемент

відповідає найбільш домінантній часовій відстані між послідовними биттями. Отримане значення використовується для обчислення швидкості BPM (beats per minute).

Крім самого значення BPM, як характеристика використовується відношення максимального значення гістограми до загальної суми значень гістограми, що відповідає загальному «домінуванню» виявленої частоти биття.

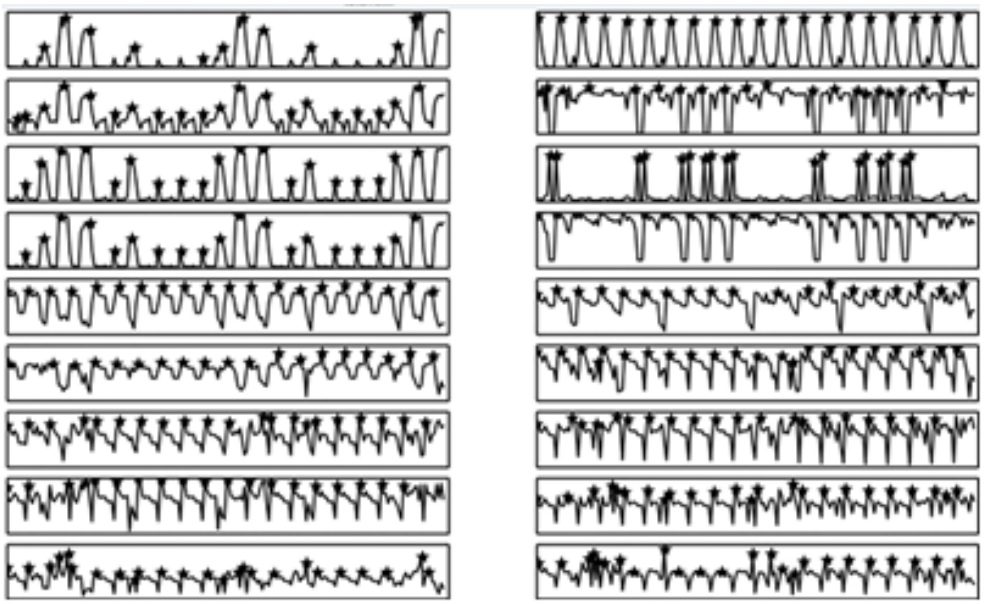

Рисунок 1.6 – Набор короткочасних послідовностей [15]

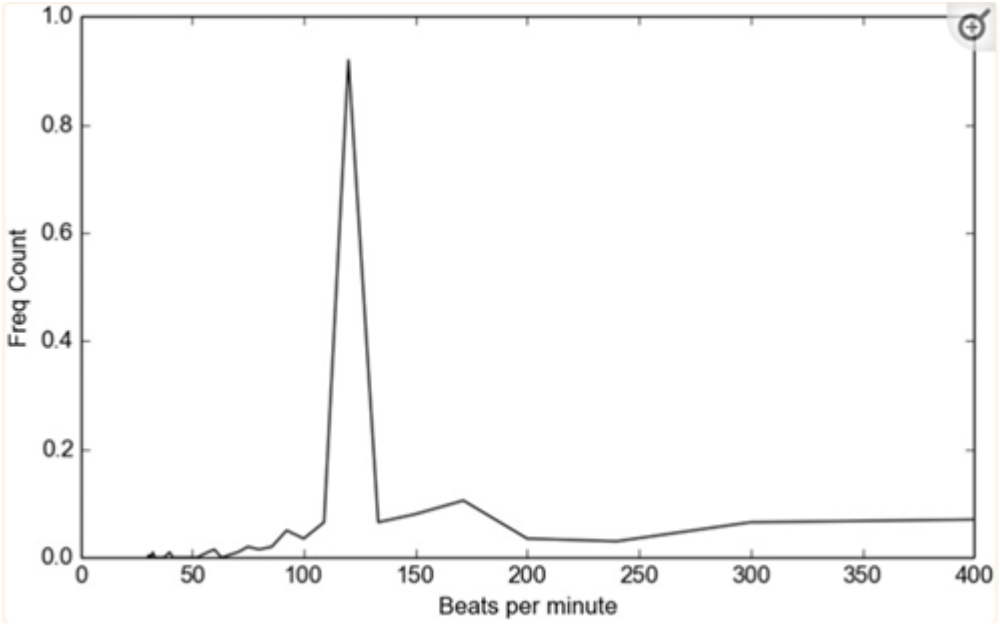

Рисунок 1.7 – Агрегована гістограма часових відстаней [15]

Слід зазначити, що дві функції, пов’язані з темпом, витягуються лише для цілих аудіозаписів (вони не обчислюються на короткостроковій основі). Тому їх можна додати лише до довгострокових середніх середньострокових характеристик, описаних у попередньому розділі. Для класифікаторів сегментів загальних класів аудіо (наприклад, музика чи мова) функції темпу не застосовуються.

Сегментація аудіо зосереджена на розділенні безперервного аудіосигналу на сегменти однорідного вмісту. Термін «однорідний» можна визначити багатьма різними способами, тому існує невід’ємна складність у наданні загального визначення цього поняття. Бібліотека надає алгоритмічні рішення для двох загальних підкатегорій сегментації звуку.

Перший містить алгоритми, які приймають певний тип «попередніх» знань, напр. попередньо підготовлену схему класифікації. Для цього типу сегментації бібліотека забезпечує об’єднану сегментацію фіксованого розміру – підхід до класифікації та метод на основі HMM (A hidden Markov model).

Другий тип сегментації – неконтрольований або напівконтрольований. В обох випадках попередні знання про відповідні класи аудіовмісту не використовуються. Типовими прикладами є видалення мовчання, Розпізнавання мовця та «мініатюра» звуку.

Сегментація фіксованого розміру. Цей простий спосіб сегментації аудіозапису на однорідні сегменти розбиває аудіопотік на сегменти фіксованого розміру та класифікує кожен сегмент окремо за допомогою контрольованої моделі. Послідовні сегменти, які мають спільну мітку класу, об’єднуються на етапі постобробки. Крім того, бібліотека витягує деякі базові статистичні дані, приклад таких даних наведено на рисунку 1.8.

Сегментація на основі HMM. Приховані моделі Маркова (HMM) – це стохастичні автомати, які слідують за переходами між станами на основі імовірнісних правил. Коли HMM досягає стану, він видає спостереження, яке у випадку аналізу сигналу зазвичай є безперервним вектором ознак. HMM можна використовувати для розпізнавання послідовних міток на основі відповідної

послідовності векторів звукових характеристик. Це досягається шляхом пошуку послідовності станів, яка випромінює певну послідовність спостережень з найвищою ймовірністю. Відповідь на це запитання дає методологія динамічного програмування – алгоритм Вітербі [15-17].

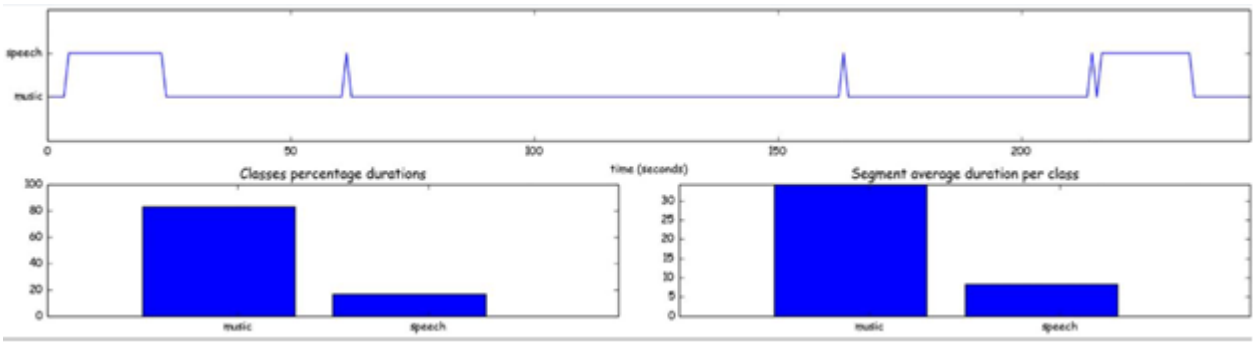

Рисунок 1.8 – Приклад сегментації фіксованого розміру [15]

Бібліотека pyAudioAnalysis надає можливість навчати та тестувати HMM для спільної сегментації-класифікації звуку. Щоб навчити модель HMM, користувач має надати анотовані дані у досить простому форматі, розділеному комами, який включає три стовпці: початкова точка сегмента, кінцева точка сегмента та мітка сегмента. Для кожного відповідного аудіозапису необхідно надати один файл анотації. Набір аудіофайлів і відповідних файлів анотацій утворює навчальний набір.

1.4 Висновки до розділу 1

Застосування web платформи для реалізації додатку аналізу аудіоданих буде виконуватися на базі Python, з використанням такого популярного фреймворка як FastAPI. Він є основою розробки вебплатформи для реалізації додатку, тому що він дуже простий у використанні, а також один з найбільш популярних і функціональних.

Також у цьому дипломі було представлено pyAudioAnalysis, бібліотеку

Python з відкритим вихідним кодом, яка реалізує широкий спектр функцій аналізу звуку та може використовуватися в кількох програмах.

За допомогою pyAudioAnalysis можна класифікувати невідомий аудіосегмент до набору попередньо визначених класів, сегментувати аудіозапис і класифікувати однорідні сегменти, видаляти зони тиші із запису мовлення, оцінювати емоції сегмента мовлення, витягувати мініатюри аудіо з музичної доріжки тощо. Також надаються високорівневі оболонки та використання командного рядка, щоб користувачі могли досягти повної функціональності.

Діапазон функцій аудіоаналізу, реалізованих у бібліотеці, охоплює більшу частину загального спектру аудіоаналізу: класифікація, регресія, сегментація, виявлення змін, кластеризація та візуалізація через зменшення розмірності. Тому pyAudioAnalysis можна використовувати як основу для більшості загальних програм аналізу звуку.

Завдяки цим інструментам не потрібно зайвий раз використовувати безліч непотрібних доповнень до Python, а також написання коду відбуватиметься у спеціально виділених вікнах функцій під цей фреймворк.

2 ПРОЄКТУВАННЯ ВЕБПЛАТФОРМИ ДЛЯ ДОДАТКУ АНАЛІЗУ АУДІОДАНИХ

2.1 Вимоги до проєкту

Для дослідження предметної області з використанням FastAPI було обрану тему реалізації додатку аналізу аудіоданих. Подібна тема дозволяє в повній мірі розглянути роботу самого фреймворка FastAPI, а також побачити всі залежності, які можуть виникнути при моделюванні проєкту.

Основною метою розробки додатку є створення інформаційної системи для візуалізації даних.

Створюючи функціональні вимоги до вебплатформи, розробленої для програми аналізу аудіоданих, треба враховувати наступні основні функціональні вимоги, які будуть важливими для такого проєкту:

- реєстрація користувачів і функція входу для авторизованого доступу до платформи (контроль доступу на основі ролей для визначення різних рівнів доступу для користувачів (таких як адміністратор, аналітики або звичайні користувачі));
- можливість для користувачів в різних форматах (безпечне зберігання та керування завантаженими аудіоданими в системі);
- функції для аналізу аудіоданих, наприклад візуалізація сигналу, частотний аналіз, спектральний аналіз або аналіз у часовій області. Інструменти для вилучення відповідних характеристик із аудіоданих, таких як темп, висота, амплітуда або спектральні характеристики (можливість запускати різні алгоритми або моделі для класифікації звуку, розпізнавання образів або інших типів аналізу);
- інструменти візуалізації для представлення результатів аналізу в зручних для користувача форматах (графіки, діаграми тощо), генерація звітів або зведень на основі проаналізованих аудіоданих;

– інтеграція із зовнішніми бібліотеками або API для обробки аудіо або моделей машинного навчання (за потреби) для розширення можливостей аналізу платформи;
– функція пошуку для отримання певних аудіофайлів або результатів аналізу на основі таких параметрів, як дата, характеристики звуку або визначені користувачем теги;
– можливість експорту результатів аналізу або звітів у різні формати (PDF, CSV тощо) (параметри спільного доступу, які дозволяють користувачам ділитися своїми висновками або звітами про аналіз з іншими користувачами);
– панель адміністрування для керування системою, включаючи керування користувачами, керування зберіганням даних та налаштування системи.

До нефункціональних вимог можливо віднести:

– простий для розуміння користувача інтерфейс;
– оптимізація.

У адміністратора сайту є можливість переглядати профілі користувачів, які зареєструвалися на сайті. Тому користувачі можуть авторизуватися повторно на цьому сайті.

Доступні функціональні можливості для різних типів користувачів можливо представити у вигляді діаграми прецедентів (рис. 2.1).

Рисунок 2.1 – Діаграма прецедентів додатку аналізу аудіоданих

Як можна побачити на рисунку, адміністратор сайту має право до усіх адміністративних можливостей сайту на відміну від користувачів, які мають основну функціональність використання додатку.

2.2 Додатки, використані при розробці проєкту

Для реалізації проєкту було використано наступні додатки: PyCharm, FastAPI.

PyCharm [14] – це інтегроване середовище розробки для мови програмування Python (див. рис. 2.2). Надає засоби для аналізу коду, графічний зневаджувач, інструмент для запуску юніт-тестів і підтримує веброзробку на Django. PyCharm розроблена чеською компанією JetBrains на основі IntelliJ IDEA. PyCharm є кращим IDE для розробки, пропонуючи ряд функцій для оптимізації кодування, налагодження та тестування вебплатформи та її компонентів.

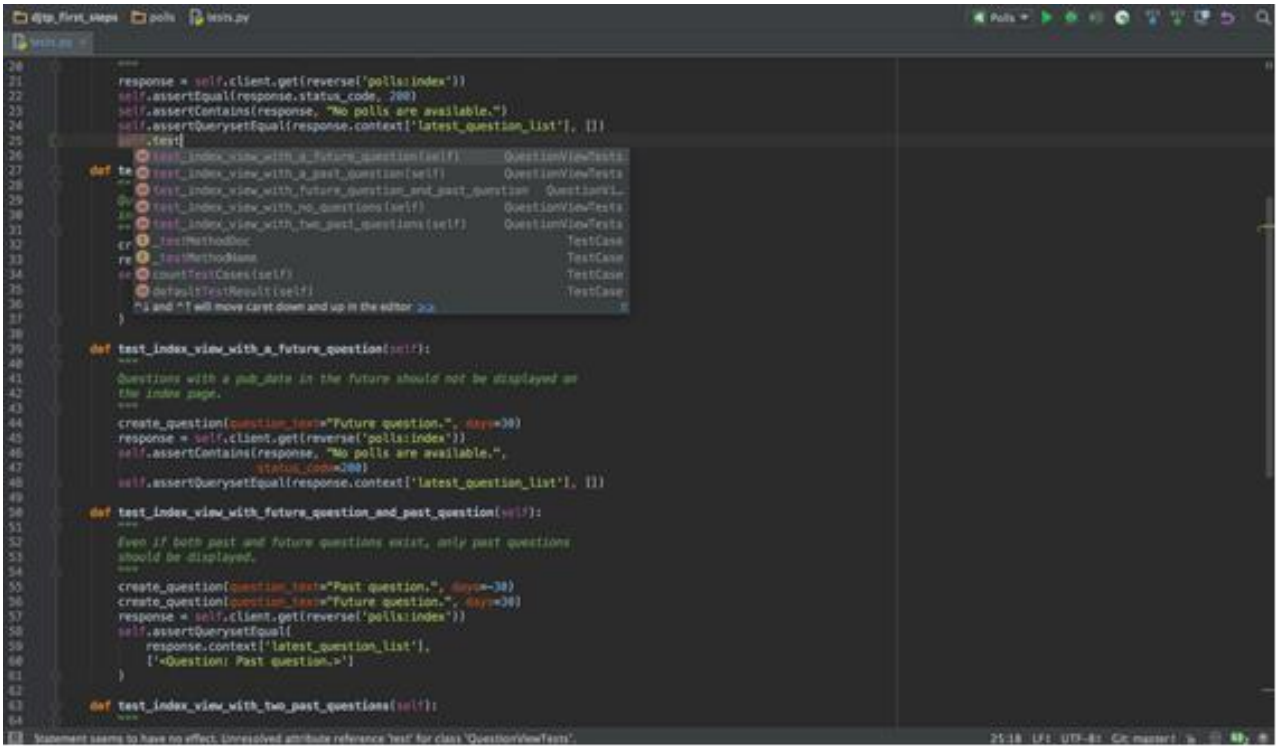

Рисунок 2.2 – Інтегроване середовище розробки PyCharm

FastAPI – це сучасна, високопродуктивна вебплатформа для створення API за допомогою Python. Він використовує підказку типу Python для перевірки даних і автоматично створює інтерактивну документацію API. FastAPI використовується для розробки серверної частини вебплатформи, що дозволяє створювати API для завантаження, аналізу та пошуку аудіоданих. справді інклюзивний досвід.

2.3 Бібліотеки та фреймворки

NumPy – розширення мови Python, що додає підтримку великих багатовимірних масивів і матриць, разом з великою бібліотекою високорівневих математичних функцій для операцій з цими масивами (рис. 2.3).

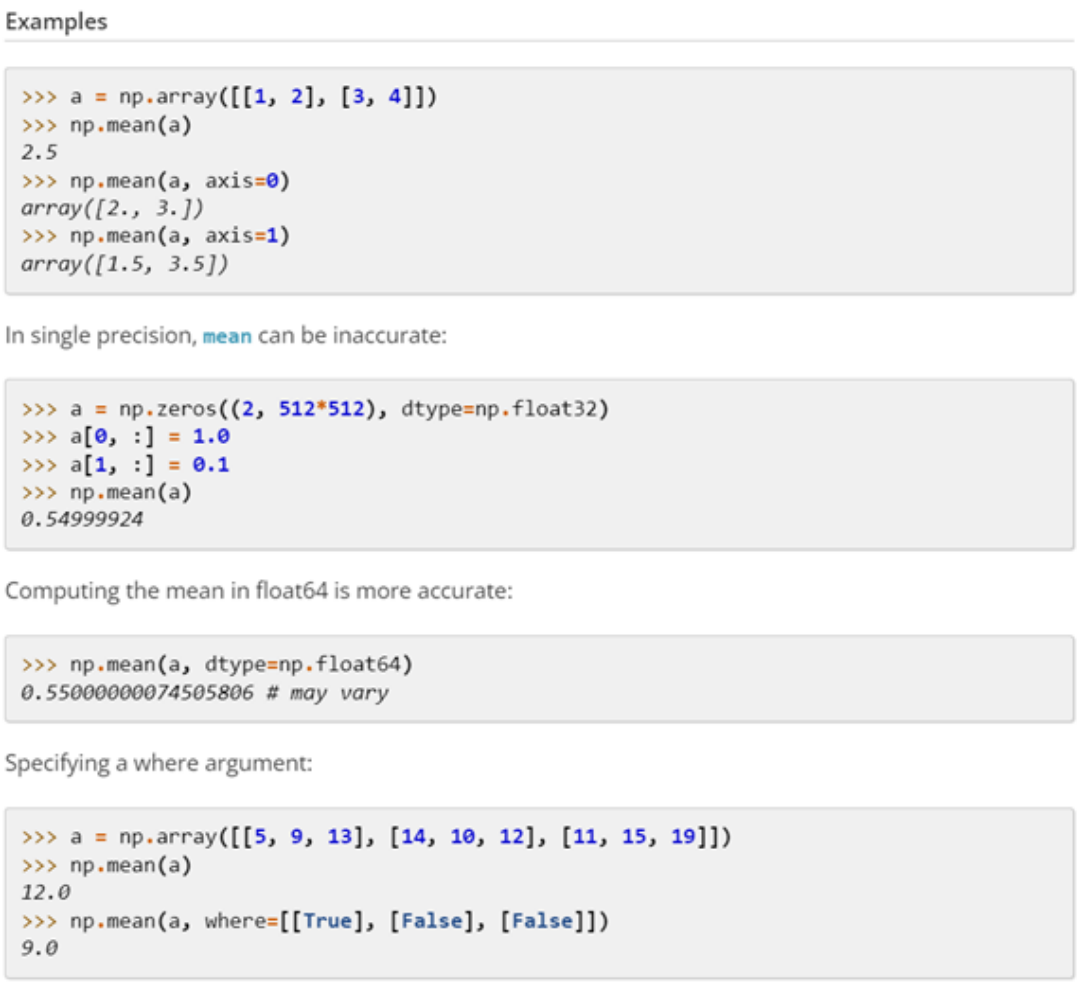

Рисунок 2.3 – Приклад роботи з розширенням NumPy

Попередник Numpy, Numeric, був спочатку створений Jim Hugunin. Numpy – відкрите програмне забезпечення і має багато розробників. NumPy можна використовувати для обробки числових даних і обробки числових даних, зокрема в контексті аналізу аудіоданих.

Matplotlib – це бібліотека для побудови графіків для Python, яка дозволяє створювати різні візуалізації, зокрема графіки, діаграми та гістограми (див. рис. 2.4). Matplotlib можна використовувати для створення візуальних представлень проаналізованих аудіоданих, сприяючи візуалізації та інтерпретації даних.

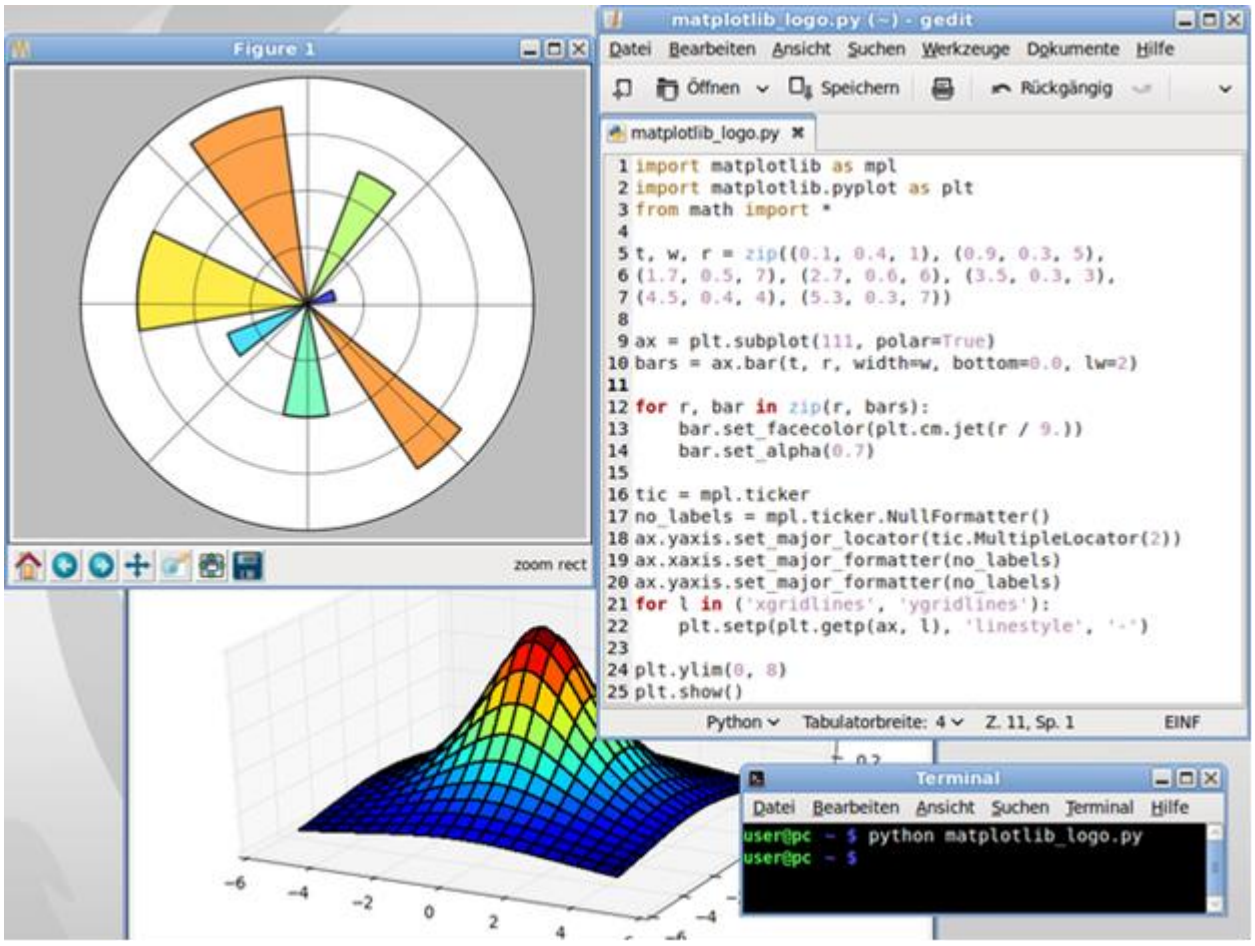

Рисунок 2.4 – Приклад роботи з бібліотекою Matplotlib

PyAudio – це набір прив’язок Python для PortAudio, який забезпечує інтерфейс для захоплення та відтворення аудіоданих за допомогою Python. PyAudio надає прив’язки Python для PortAudio v19, міжплатформенної

бібліотеки аудіо введення/виведення. За допомогою PyAudio ви можете легко використовувати Python для відтворення та запису аудіо на різноманітних платформах, таких як GNU/Linux, Microsoft Windows і Apple macOS. PyAudio можна використовувати для захоплення та обробки аудіоданих для аналізу в вебдодатку (рис. 2.5).

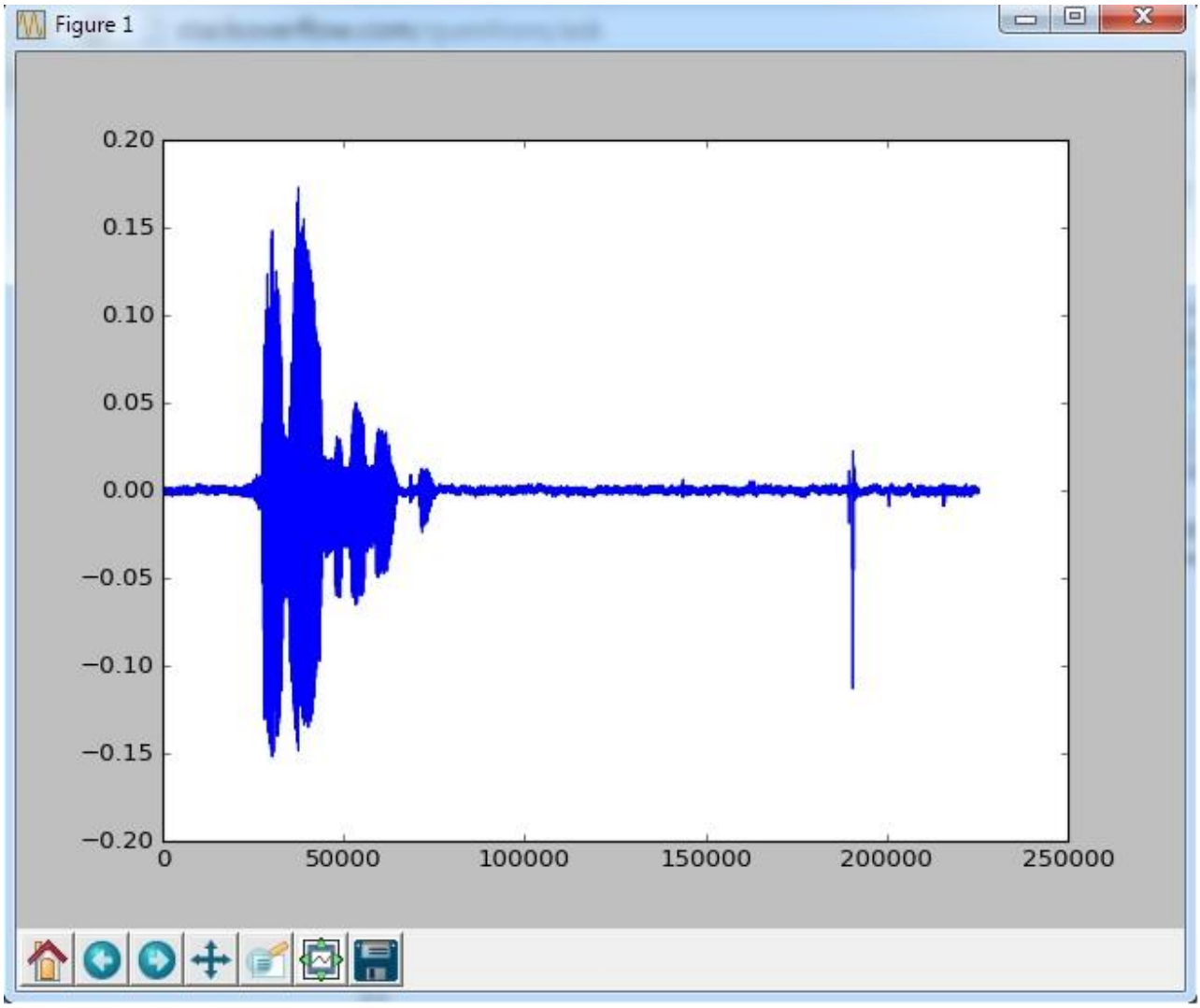

Рисунок 2.5 – Приклад роботи з PyAudio

2.4 Функціональне проєктування

Проєктування нових видів і зразків машин, пристроїв, апаратів представляє складний і тривалий процес, що включає розробку початкових

даних, креслень, технічної документації, необхідних для виготовлення дослідних зразків і наступного виробництва і експлуатації об’єктів проєктування.

В процесі проєктування виникає необхідність створення опису, необхідного для побудови ще неіснуючого об’єкту. Тому можна скористатися методологією IDEF0, суттю якої є побудова ієрархічної системи діаграм. Тобто кожна діаграма поодинці буде описом частини системи.

Перед побудовою моделі необхідно визначитися, яка модель системи буде побудована. А також визначення позиції, з точки зору якої будується модель. Точку зору найкраще уявляти собі як позицію людини або об’єкта, в яке треба встати, щоб побачити систему в дії.

Нижче надано діаграму методології IDEF0 першого рівня (рис. 2.6).

Рисунок 2.6 – Контекстна діаграма IDEF0

Після побудови контекстна діаграма деталізується за допомогою діаграми декомпозиції першого рівня. На цій діаграмі відображаються функції системи, які повинні бути реалізовані в рамках основної функції. Діаграма декомпозиції першого рівня розкладається на різні підфункції, кожна з яких представляє більш детальний аспект системи.

Це можуть бути такі функції, як «Збір аудіоданих», «Попередня обробка

даних», «Вилучення функцій», «Аналіз даних», «Візуалізація», «Інтерфейс користувача», «Звітування», «Керування системою» тощо.

Стрілки ілюструють входи, виходи та елементи керування між цими підфункціями, показуючи, як вони взаємодіють і залежать одна від одної.

Кожну підфункцію з Рівня першого можна далі розкласти на більш детальні функції. Функцію «Аналіз даних» можна розділити на «Спектральний аналіз», «Аналіз у часовій області», «Розпізнавання образів» тощо. Діаграми рівня другого зосереджені на визначенні детальних операцій і підпроцесів у кожній підфункції.

Система працює за класичною схемою клієнт-серверного додатку. Користувацька частина системи отримує інформацію та дані з запитів відправлених на сервер, який в свою чергу оброблює їх та за необхідністю звертається до даних що зберігаються у БД, як показано на рисунку 2.7.

Рисунок 2.7 – Діаграма розгортання додатку аналізу аудіоданих

Для більшого розуміння взаємодії компонентів системи необхідно розглянути окремі функції системи та роботу частин додатку всередині них. Нище, на рисунку 2.8 наведено функцію аторизації.

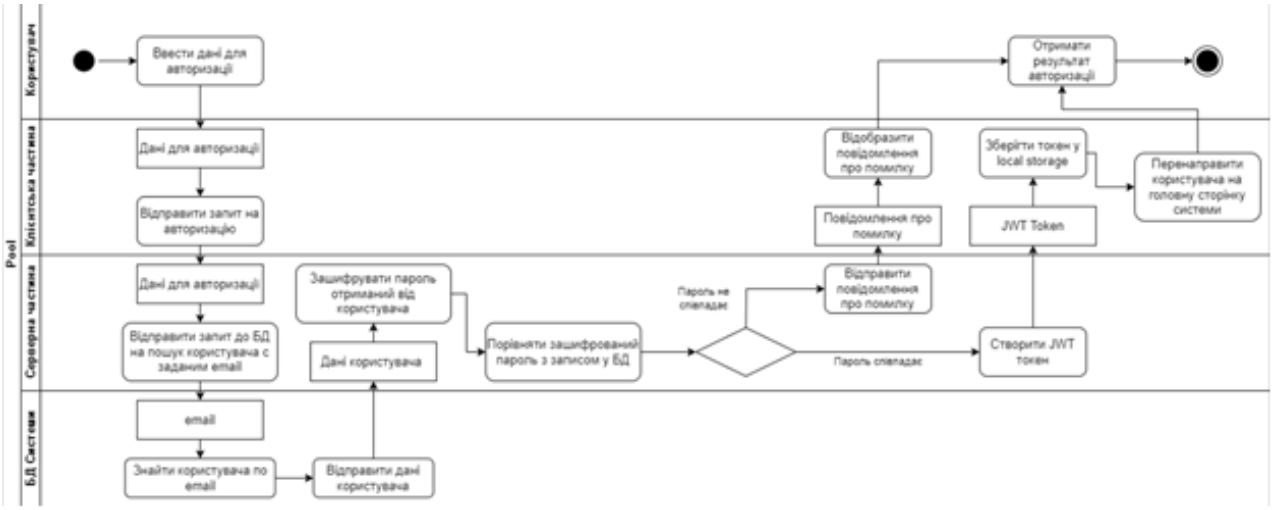

Рисунок 2.8 – Діаграма діяльності для функції авторизації

Як можна побачити з діаграми, сервер виконує роль контролера аутентифікації та доступу до ресурсів системи. А клієнтська частина надає користувачу інтерфейс взаємодії з сервером та відображає наявні дані.

2.5 Концептуальне проєктування

Концептуальні проєкти можна представити у вигляді ER-діаграм. Діаграми «сутність-зв’язок» допомагають зрозуміти зв’язки між різними елементами системи, такими як продукти, клієнти та номери замовлень. Перш ніж почати будувати базу даних, ER-діаграма дозволяє спочатку структурувати її концепції, від загальної структури до конкретних способів взаємодії компонентів один з одним. Тому при проєктуванні бази даних для розробки програмного забезпечення ER-діаграма допоможе візуалізувати структуру необхідної бази даних та виявити й усунути її недоліки на початковому етапі роботи.

Нижче буде наведено реалізовану концептуальну ER-діаграму (рис. 2.9).

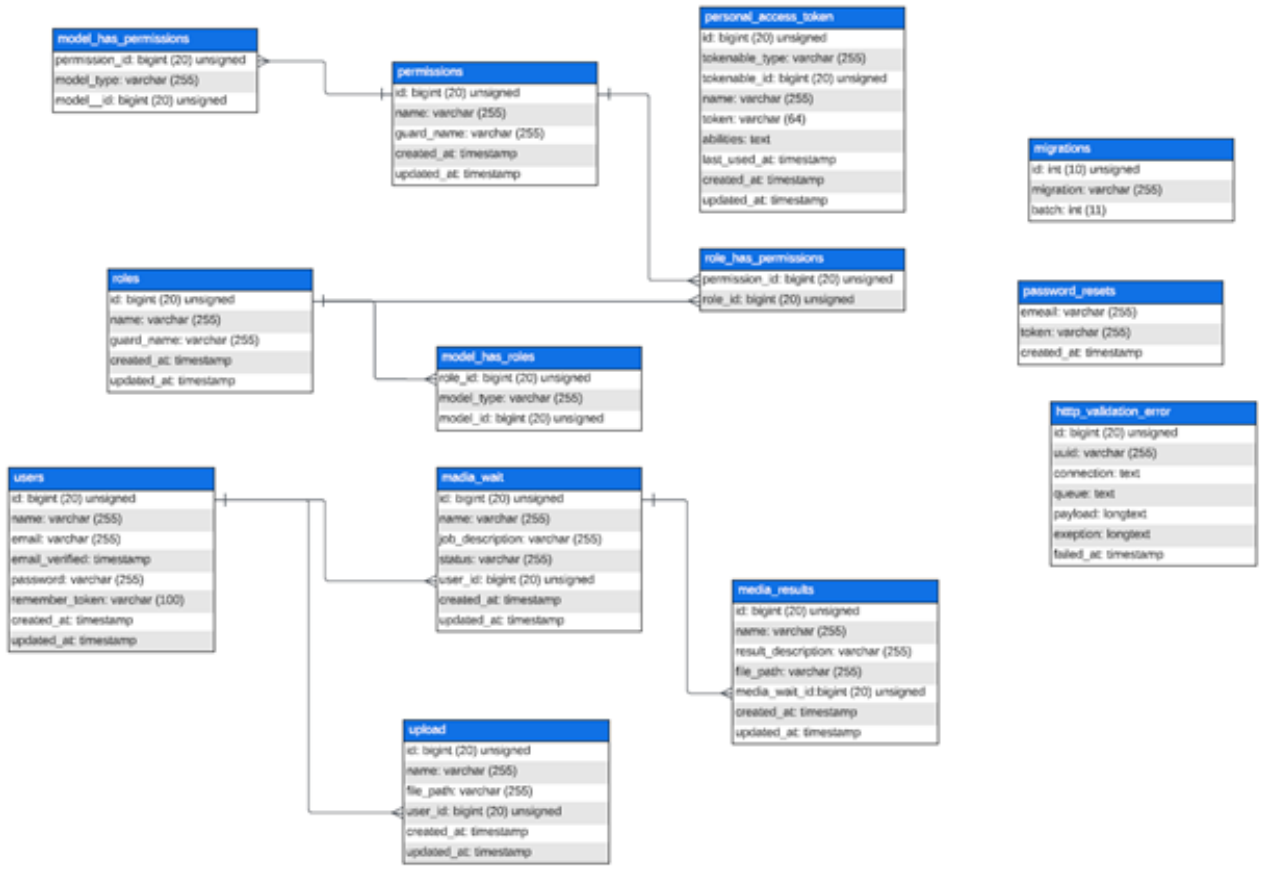

Рисунок 2.9 – ER-діаграма

2.6 Логічне проєктування

Основна думка логічного проєктування полягає у створення логічної моделі даних та перетворенні концептуальної моделі даних у логічну модель даної системи з урахуванням обраного типу СКБД.

Таблиця 2.1 upload зберігає аудіофайли, які мають буди завантажені користувачем чи адміністратором на сайти для аналізу цього аудіофайлу.

Перший атрибут id – первинний ключ, на кожній таблиці він буде однаковий, тому його характеристики будуть перелічені лише один раз.

Атрибут name зберігає назву аудіофайла та є символьним типом даних з обмеженням на кількість символів 255.

Атрибут file_path зберігає шлях звідки буде здійснюватись завантеження аудіофайлу користувачем чи адміністратором на сайт.

Атрибут created_at – час, коли було створено цей тип місць, та updated_at – час, коли тип місць був оновлений при редагуванні місць.

Таблиця 2.1 – Структура таблиці «upload»

Назва	**Тип даних**	**Властивості**	**Зв'язок**
id	bigint(20)	Auto_increment	-
name	Varchar(255)	-	-
file_path	Varchar(255)	-	-
user_id	bigint(20)	-	Від Users (id)
created_at	TimeStamps	nullable	-
updated_at	TimeStamps	nullable	-

Таблиця 2.2 media_wait зберігає інформацію, яка має заповнювати місце під час аналізу аудіо, тому шо це не миттєвий процес і програма має деякий час виконання.

Таблиця 2.2 – Структура таблиці «media_wait»

Назва	**Тип даних**	**Властивості**	**Зв'язок**
id	bigint(20)	Auto_increment	-
name	Varchar(255)	-	-
job_describtion	Varchar(255)	-	-
status	Varchar(255)	-	-
user_id	bigint(20)	-	Від users (id)
created_at	TimeStamps	nullable	-
updated_at	TimeStamps	nullable	-

Атрибут name зберігає назву самого процесу під час аналізу аудіо та є символьним типом даних з обмеженням на кількість символів 255.

Атрибут job_describtion зберігає опис роботи з аналізу, це може бути будь-який текст, який адміністратор додасть під час процесу виконання програми.

Атрибут status зберігає статус, який має процес програми. Це може бути як текст, так і відсоток завершення виконання аналізу аудіофайлу програмою.

Атрибут user_id зберігає первинний ключ користувачів, які на даний момент користуються системою та чекають виконання процесу аналізу аудіофайлу програми.

Атрибут created_at – час, коли було створено цей тип місць, та updated_at – час, коли тип місць був оновлений при редагуванні місць.

Таблиця 2.3 media_results займає вагому частину логічного проєктування. Вона зберігає всі вихідні результати процесу аналізу аудіоданих програми у вигляді діаграм.

Атрибут name зберігає назву типу результати процесу аналізу аудіоданих програми та є символьним типом даних з обмеженням на кількість символів 255.

Атрибут result_description зберігає опис результатів процесу аналізу аудіоданих програми та є символьним типом даних з обмеженням на кількість символів 255.

Атрибут file_path зберігає путь для завантаження результатів процесу аналізу аудіоданих програми на комп’ютер чи пристрій користувача та є символьним типом даних з обмеженням на кількість символів 255.

Атрибут created_at – час, коли було створено цей тип місць, та updated_at – час, коли тип місць був оновлений при редагуванні місць.

Таблиця 2.3 – Структура таблиці «media_results»

Назва	**Тип даних**	**Властивості**	**Зв’язок**
id	bigint(20)	Auto_increment	-
name	Varchar(255)	-	-
result_description	Varchar(255)	-	-
file_path	Varchar(255)	-	-
media_wait_id	bigint(20)	-	Від media_wait (id)
created_at	TimeStamps	nullable	-
updated_at	TimeStamps	nullable	-

Таблиця 2.4 users зберігає самих користувачів, які повинні при вході на сайт зареєструватися чи авторизуватися, якщо вони вже були зареєстровані.

Атрибут name зберігає назву роботи з аналізу та є символьним типом даних з обмеженням на кількість символів 255.

Атрибут email зберігає email користувачів, який вони вводять при реєстрації на сайті. Завдяки саме цьому атрибуту можна привласнювати результати аналізу аудіо, тобто діаграми аналізу будуть відіслані саме по цьому email користувачам. Атрибут email є символьним типом даних з обмеженням на кількість символів 255.

Атрибут email_verified_at – час, коли був сотворений цей email. Тип даних created_at та updated_at є TimeStamps.

Атрибут password зберігає пароль користувачів, який вони вводять при реєстрації на сайті. Завдяки саме цьому атрибуту користувачі авторизуються на сайті системи. Атрибут password є символьним типом даних з обмеженням на кількість символів 255.

Атрибут remember_token зберігає пароль користувачів у зашифрованому виді, який вони вводять при реєстрації на сайті. Атрибут remember_token є символьним типом даних з обмеженням на кількість символів 100.

Таблиця 2.4 – Структура таблиці «users»

Назва	**Тип даних**	**Властивості**	**Зв’язок**
id	Bigint(20)	Auto_increment	-
name	Varchar(255)	-	-
email	Varchar(255)	-	-
email_verified_at	TimeStamps	nullable	-
password	Varchar(255)	-	-
remember_token	Varchar(100)	-	-
created_at	TimeStamps	nullable	-
updated_at	TimeStamps	nullable	-

2.7 Висновки до розділу 2

У другій частині розглядалися вимоги проєкту та представлено доступні функції для різних типів користувачів у вигляді прецедентної діаграми. Проєкт реалізовано в середовищі Pycharm з використанням серверної платформи docker і найбільшої бібліотеки для математичних обчислювань matplotlib. Також використовувалися такі фреймворки, як fastapi, pyaudio numpy, тощо.

Завдяки функціональному дизайну, використовуючи такі методи, як IDEF0, а також діаграми декомпозиції, розгортання та діяльності, модель вебплатформи для реалізації додатку аналізу аудіоданих розроблена інтуїтивно.

Концептуальний дизайн допомагає зрозуміти зв’язок між різними елементами системи на основі побудованої ER-діаграми. Нарешті, логічне проєктування допомагає більш детально описати концептуально розроблену модель даних у логічну модель даної системи за допомогою побудованих таблиць і систем.

3 РОЗРОБКА ВЕБПЛАТФОРМИ ДЛЯ РЕАЛІЗАЦІЇ ДОДАТКУ

3.1 Початкові налаштування проєкту

З початку роботи над проєктом необхідно визначитися, які необхідно завантажити пакети для розробки, та як потрібно налаштувати зв'язок між серверною та клієнтською частиною додатку.

Під час роботи з FastAPI, сучасною, швидкою (високопродуктивною) вебплатформою для створення API з Python 3.7+ на основі стандартних підказок типу Python, є певні початкові параметри та конфігурації, які потрібно розглянути. Наприклад, встановлення FastAPI та Uvicorn. FastAPI можна встановити за допомогою pip. Uvicorn рекомендовано як сервер ASGI. Та створення каталог проєкту у чіткій структурі каталогів, як показано на рисунку 3.1.

```
mkdir my_fastapi_project
cd my_fastapi_project
```

Рисунок 3.1 – Створення початкового каталогу для FastAPI

Наступним кроком є налаштування віртуального середовища та активація віртуального (див. рис. 3.2) середовища на основі моєї операційної системи для ізоляції залежностей проєкту.

```
Windows: "venv\Scripts\activate"
```

Рисунок 3.2 – Налаштувати віртуального середовища

Щоб налаштувати базове середовище з Matplotlib, PyAudio та NumPy, треба виконати наступні дії, як показано на рисунку 3.3.

```
# Create a virtual environment
python -m venv venv
# Activate the virtual environment
# On Windows
venv\Scripts\activate
# On macOS/Linux
 source venv/bin/activate
```

Рисунок 3.3 – Налаштувати базового середовища

Далі встановлення залежності у базовому середовище з Matplotlib, PyAudio та NumPy: «pip install matplotlib pyaudio numpy».

Щоб переконатися, що встановлення бібліотек було здійснено перевіримо, як показано на рисунку 3.4.

```
import matplotlib.pyplot as plt
import numpy as np
# Generate some sample data
x = np.linspace(0, 10, 100)
y = np.sin(x)
# Plot the data
plt.plot(x, y)
plt.title('Matplotlib Example')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')
# Show the plot
plt.show()
python plot_example.py
```

Рисунок 3.4 – Перевірка встановлення бібліотек

Якщо перевірка працює, з’являється графік, що показує синусоїду. Далі слід перевірити роботу PyAudio (див. рис. 3.5).

```
import pyaudio
import numpy as np
# Set up PyAudio
p = pyaudio.PyAudio()
# Parameters
FORMAT = pyaudio.paInt16
CHANNELS = 1
RATE = 44100
CHUNK = 1024
# Open a stream
stream = p.open(format=FORMAT,
                channels=CHANNELS,
                rate=RATE,
                input=True,
                frames_per_buffer=CHUNK)
print("Recording...")
# Record for 5 seconds
for i in range(int(RATE / CHUNK * 5)):
    data = np.frombuffer(stream.read(CHUNK), dtype=np.int16)
    # Process or analyze the audio data as needed
# Close the stream
stream.stop_stream()
stream.close()
p.terminate()
print("Recording finished.")
python audio_example.py
```

Рисунок 3.5 – Перевірка роботи бібліотеки PyAudio

Треба переконатися, що мікрофон підключено, і можна побачити «Запис...» на консолі. Сценарій буде записувати звук протягом 5 секунд.

Надалі перевірка роботи з NumPy (див. рис. 3.6).

```
import numpy as np

# Create a NumPy array
arr = np.array([1, 2, 3, 4, 5])
# Perform some operations
arr_squared = arr**2
arr_sum = np.sum(arr)
# Print the results
print("Original Array:", arr)
print("Squared Array:", arr_squared)
print("Sum of Array:", arr_sum)

python numpy_example.py
```

Рисунок 3.6 – Перевірка роботи бібліотеки NumPy

Можна побачити вихідний масив, його квадратичні значення та суму.

Ці кроки мають допомогти при налаштувані базового середовища для роботи з Matplotlib, PyAudio та NumPy. Може знадобитися коригування залежно від подальших конкретних потреб і конфігурації системи.

3.2 Створення БД

Працюючи з FastAPI та Docker, можна створити контейнерне середовище для своєї програми FastAPI. Далі будуть зображені початкові налаштування проєкту для FastAPI за допомогою Docker (див. рис. 3.7).

Далі потрібно створити файл вимог .txt та образ Docker. Для цього потрібно відкрити термінал і перейти до каталогу проєкту.

Наступним кроком створити образ Docker за допомогою такої команди: «docker build -t my_fastapi_app.».

Прапорець -t призначає назву зображенню (треба замінити my_fastapi_app на бажане ім’я).

```
# Dockerfile
# Use an official Python runtime as a parent image
FROM python:3.8-slim
# Set the working directory in the container
WORKDIR /app
# Copy the local code to the container at /app
COPY . /app
# Install any needed packages specified in requirements.txt
RUN pip install --no-cache-dir -r requirements.txt
# Make port 80 available to the world outside this container
EXPOSE 80
# Define environment variable
ENV NAME World
# Run app.py when the container launches
CMD ["uvicorn", "main:app", "--host", "0.0.0.0", "--port", "80"]
```

Рисунок 3.7 – Початкові налаштування для FastAPI

Після створення образу запускаємо контейнер на основі наступного коду: «docker run -p 8000:80 my_fastapi_app».

Це зіставляє порт 8000 на хост-машині з портом 80 всередині контейнера. За потреби можна відрегулювати порти.

Майже останнім кроком потрібно відкрити браузер і перейти на сторінку http://localhost:8000/docs, щоб отримати доступ до інтерактивної документації FastAPI, як показано на рисунку 3.8.

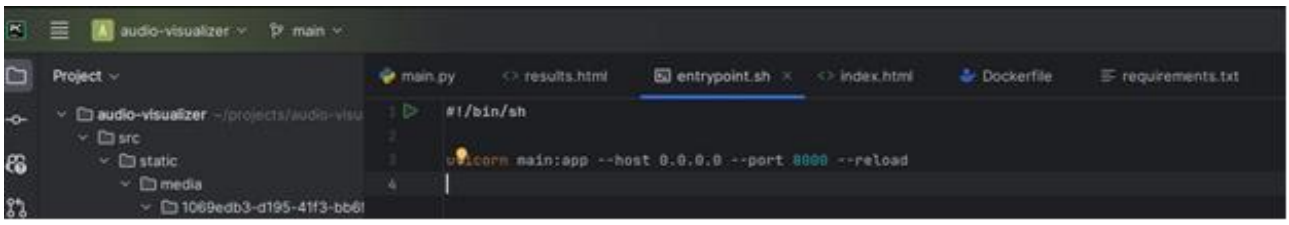

Рисунок 3.8 – Працюючий порт, щоб перейти на сторінку в браузері

Дотримуючись цих початкових налаштувань, можна створити середовище Dockerized для своєї програми FastAPI, що полегшує керування залежностями та

розгортання в різних середовищах (див. рис. 3.9). Налаштування цих параметрів було відповідно до конкретних вимог цього проєкту у FastAPI.

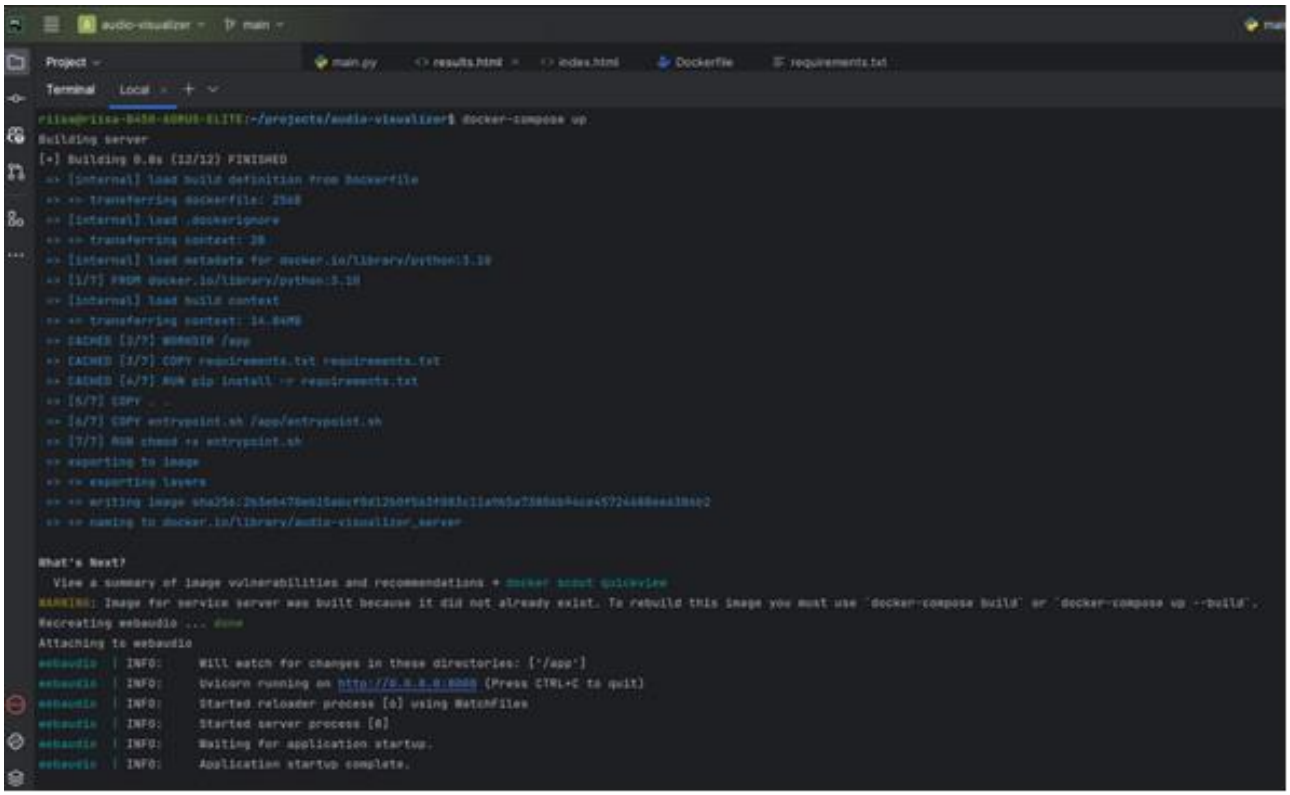

Рисунок 3.9 – Завершення встановлення середовища Docker

3.3 Розробка дизайну проєкту

Перед початком роботи над макетами дизайну необхідно визначити, які сторінки буде містити клієнтська частина додатку. Тому на рисунку 3.10 показана карта сторінок додатку аналізу аудіоданих.

Наступним кроком є створення макетів сторінок з використанням шаблонів на HTML. Першим кроком було створено головну сторінку web платформи для додатку, на якій має бути кнопка з реєстрацією та авторизацією, кнопка з переходом на сторінку особистого кабінету, кнопка за переходом на сторінку до головної сторінці, на які саме і треба завантажувати файли, як показано на рисунку 3.11.

Рисунок 3.10 – Карта сторінок web платформи для додатку

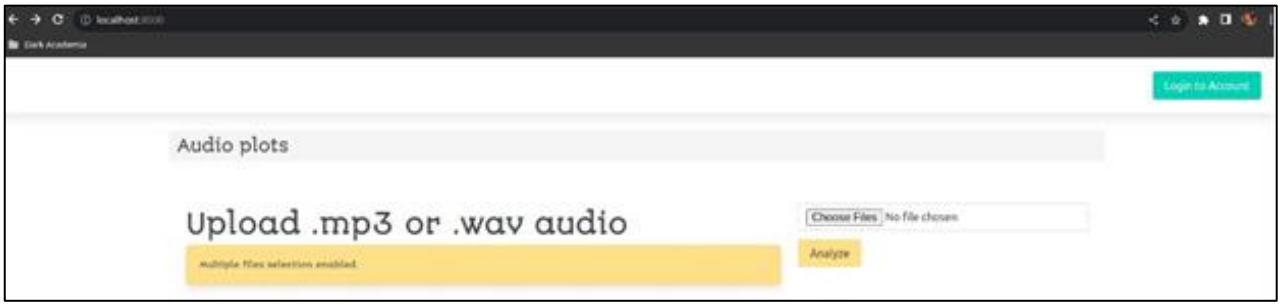

Рисунок 3.11 – Початкова сторінка web платформи для додакту

Сторінка реєстрації/авторизації, де користувач і адміністратор можуть потрапити на сайт, як показано на рисунку 3.12.

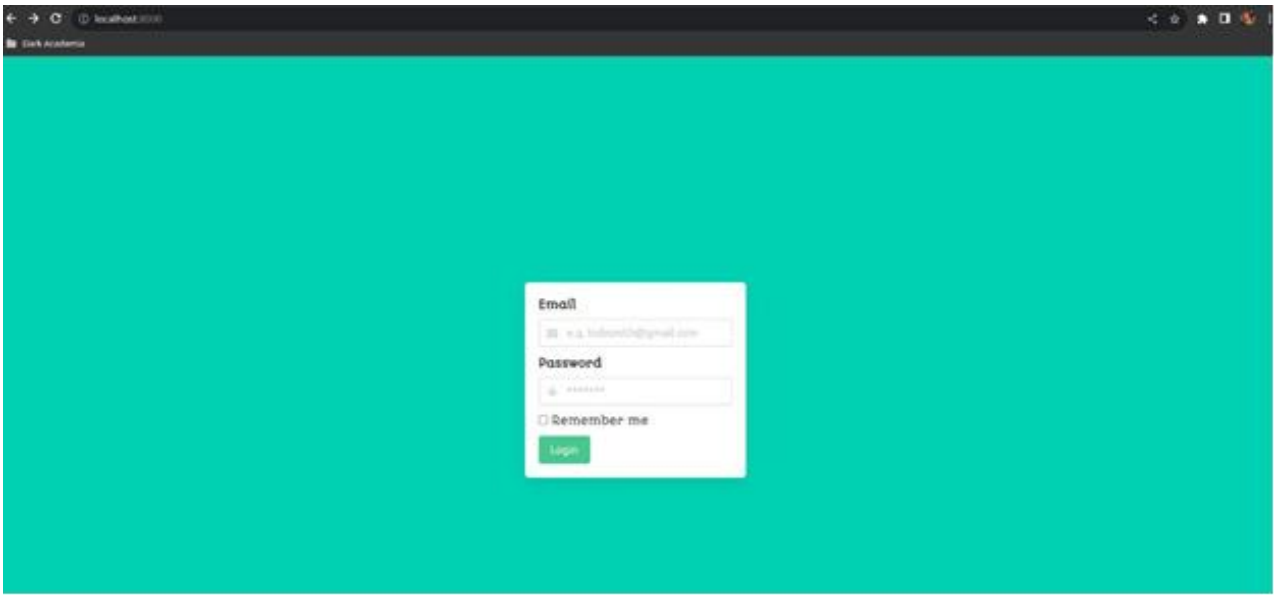

Рисунок 3.12 – Сторінка реєстрації/авторизації

На рисунку 3.13. показана головна сторінка, де користувач може заповнити форму завантаження файлів.

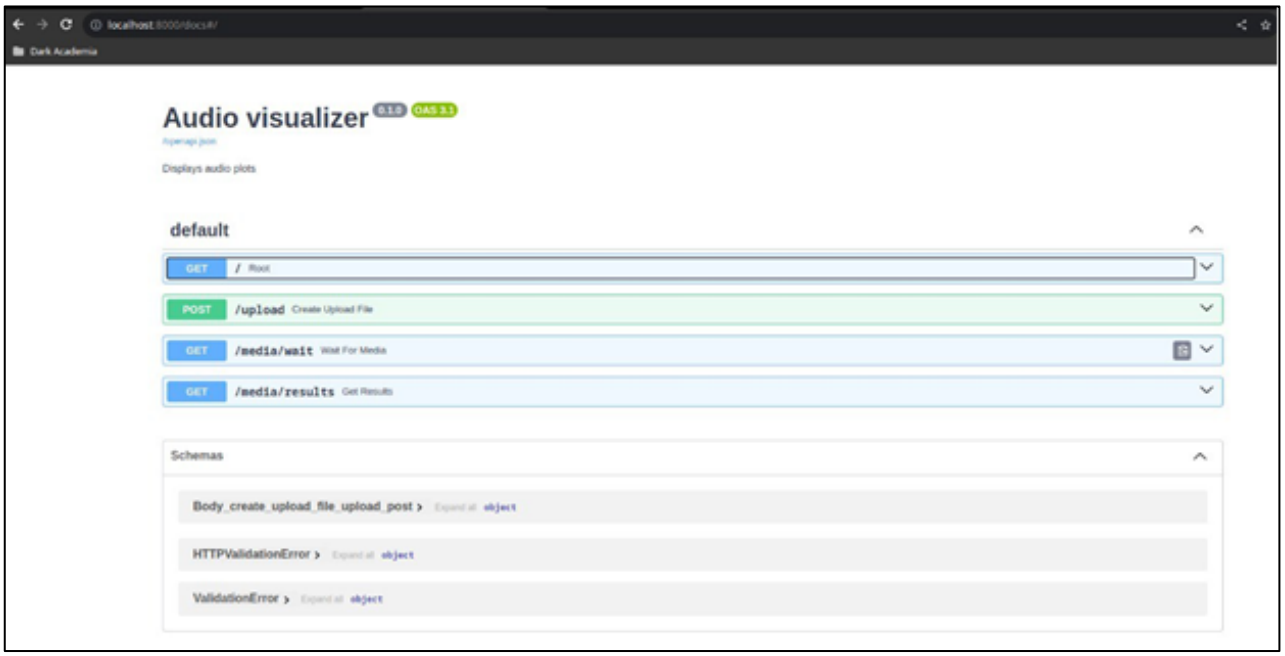

Рисунок 3.13 – Головна сторінка для завантаження фалів

На рисунку 3.14. показана головна сторінка для завантаження фалів у розгорнутому виді, де користувач може додавати завантаженні аудіофайли для аналізу.

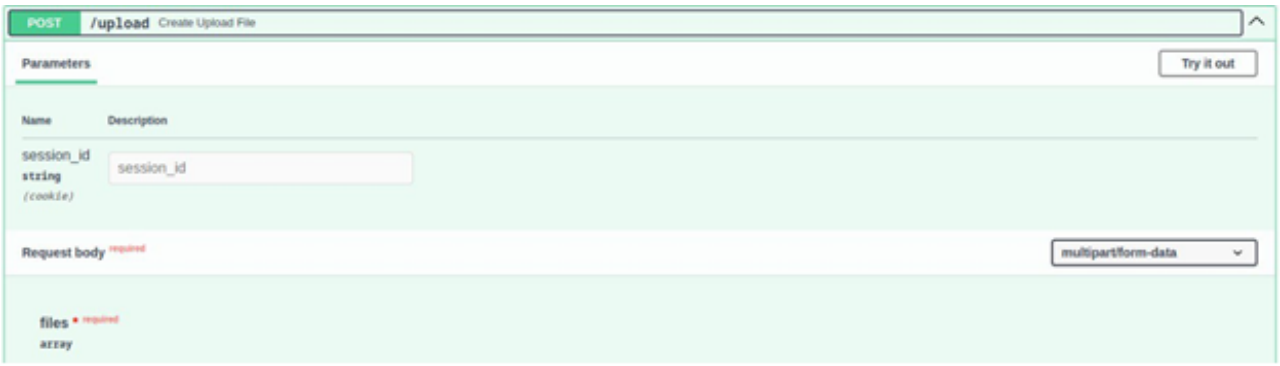

Рисунок 3.14 – Головна сторінка для завантаження фалів розгорнутого виду

На рисунку 3.15. показана головна сторінка для завантаження фалу у розгорнутому виді, де користувач має почекати поки файли обробляться, також там відображаються статус виконання роботи та деякий текст виконання.

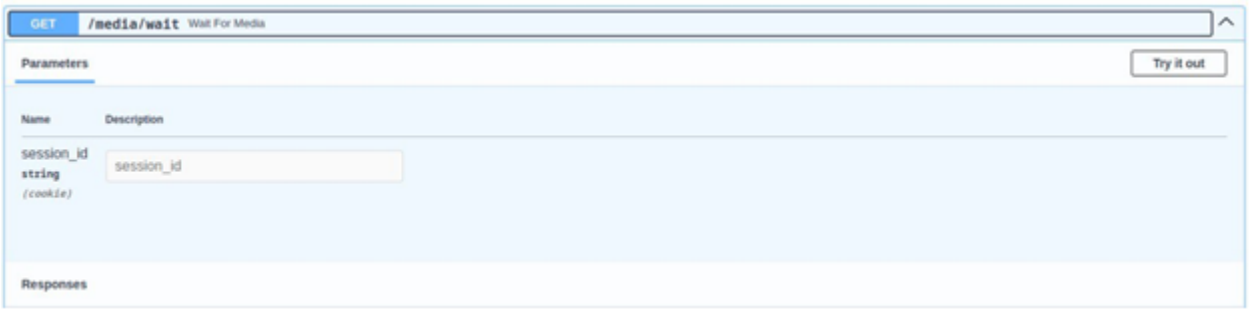

Рисунок 3.15 – Головна сторінка розгорнутого виду обробки аудіфайлів

Як тільки сервер створить графіки, сюди приходять нотифікації та пересилає на сторінку результату (рис. 3.16).

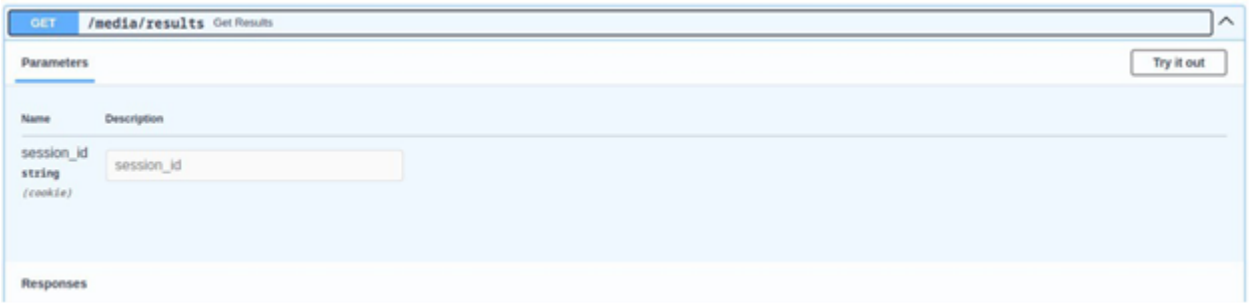

Рисунок 3.16 – Головна сторінка розгорнутого виду з нотифікаціями

На рисунку 3.17. показана головна сторінка для завантаження фалу у розгорнутому виді, де адміністратор має розширенні права керування web платформою.

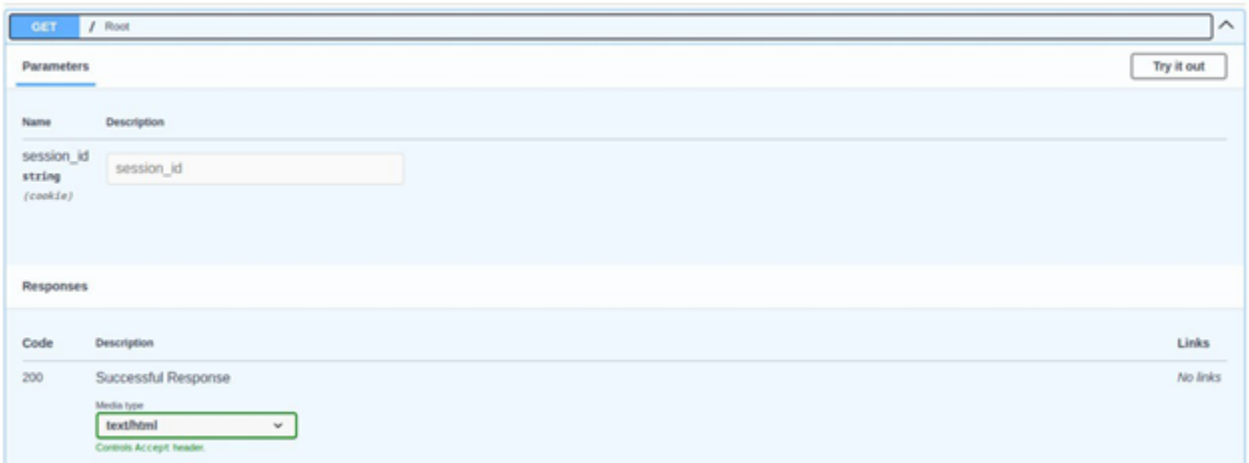

Рисунок 3.17 – Головна сторінка з розширеними правами адміністратора

На рисунку 3.18 можна побачити, що користувач може завантажувати та вибирати зі свого пристрою декілька аудіофайлів, які будуть проаналізовані додатком.

Рисунок 3.18 – Частинка сторінки прикладу завантаження декількох аудіофайлів

Як видно на рисунку 3.19, так виглядає остаточний процес роботи програми для аналізу аудіофайлів, користувач може побачити наявну сторінку результатів, на якій відображається сама діаграма.

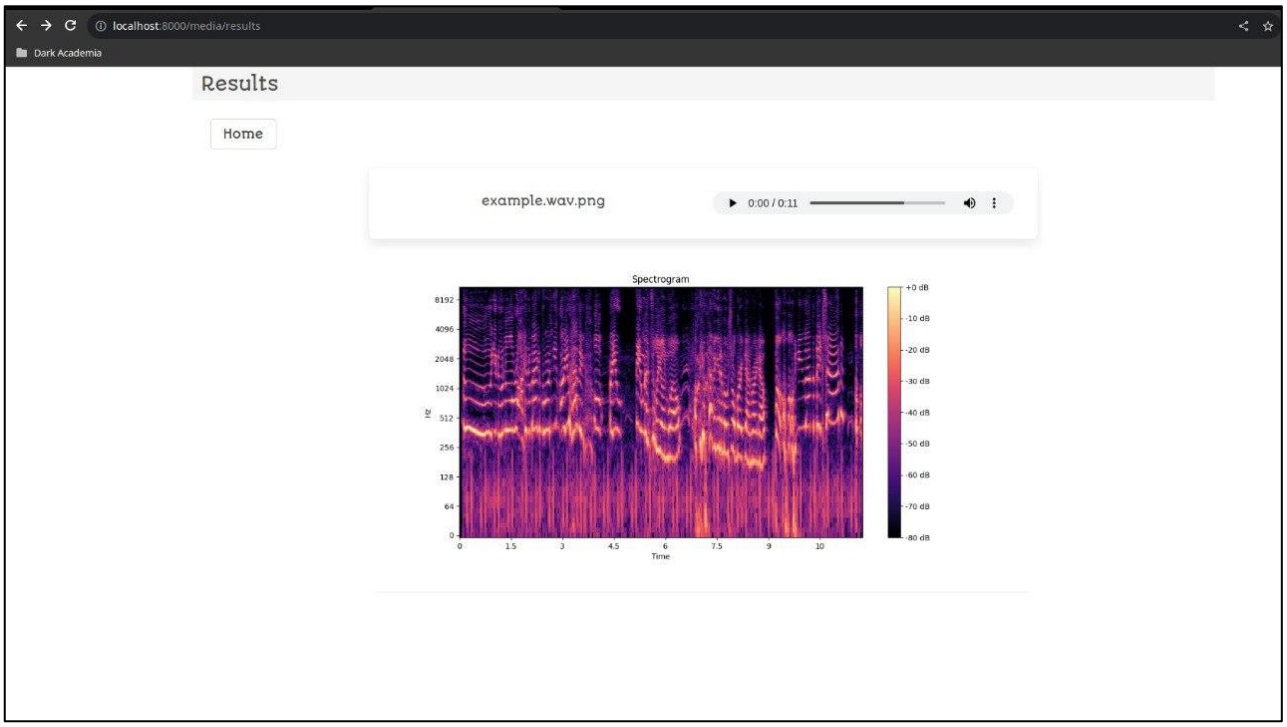

Рисунок 3.19 – Сторінка результатів аналізу аудіфайлу, діаграма

3.4 Middleware для аутентифікації

Проміжне програмне забезпечення – це фреймворк, який підключається до обробки запитів/відповідей Django. Це легка низькорівнева система «плагінів» для глобальної зміни введення або виведення Django.

Кожен компонент проміжного програмного забезпечення відповідає за виконання певної функції. Наприклад, Django містить компонент проміжного програмного забезпечення AuthenticationMiddleware, який пов’язує користувачів із запитами за допомогою сеансів.

У цьому документі пояснюється, як працює проміжне програмне забезпечення, як активувати проміжне програмне забезпечення та як написати власне проміжне програмне забезпечення. Django поставляється з деяким вбудованим проміжним програмним забезпеченням, яке можна використовувати одразу після виходу з коробки. Вони задокументовані у вбудованому довіднику проміжного програмного забезпечення (рис. 3.20).

```
def simple_middleware(get_response):
    # One-time configuration and initialization.
    def middleware(request):
        # Code to be executed for each request before
        # the view (and later middleware) are called.
        response = get_response(request)
        # Code to be executed for each request/response after
        # the view is called.
        return response
    return middleware
```

Рисунок 3.20 – Шаблон для написання простого middleware в Django

Або його можна записати як клас, екземпляри якого можна викликати. За цей функціонал відповідає наступний код на рисунку 3.21.

```
class SimpleMiddleware:
    def __init__(self, get_response):
        self.get_response = get_response
        # One-time configuration and initialization.
    def __call__(self, request):
        # Code to be executed for each request before
        # the view (and later middleware) are called.
        response = self.get_response(request)
        # Code to be executed for each request/response after
        # the view is called.
        return response
```

Рисунок 3.21 – Middleware для Django у формі класу

Останнє – використовуючи асинхронне проміжне програмне забезпечення (див. рис. 3.22) на основі класів, ви повинні переконатися, що екземпляри правильно позначені як функції співпрограми.

```
from asgiref.sync import iscoroutinefunction, markcoroutinefunction
class AsyncMiddleware:
    async_capable = True
    sync_capable = False
    def __init__(self, get_response):
        self.get_response = get_response
        if iscoroutinefunction(self.get_response):
            markcoroutinefunction(self)
    async def __call__(self, request):
        response = await self.get_response(request)
        # Some logic ...
        return response
```

Рисунок 3.22 – Middleware для Django для асинхронного використання

3.5 Аудіо-візуалізація

Опублікований код на рисунку 3.23 виглядає як один з фрагментів XML, який представляє параметри конфігурації для модуля Python в інтегрованому середовищі розробки (IDE). Зокрема, це пов’язано з IDE JetBrains, яка використовує формат файлу проєкту JetBrains.

```
<?xml version="1.0" encoding="UTF-8"?>

<module type="PYTHON_MODULE" version="4">
  <component name="NewModuleRootManager">
    <content url="file://$MODULE_DIR$" />
    <orderEntry type="inheritedJdk" />
    <orderEntry type="sourceFolder" forTests="false" />

  </component>
  <component name="SonarLintModuleSettings">
    <option name="uniqueId" value="7819108f-6320-46ae-a7d0-4134d70a582b" />

  </component>
</module>.
```

Рисунок 3.23 – Результат XML запиту JetBrains

Цей фрагмент XML є частиною конфігурації проєкту для модуля Python і містить інформацію про кореневий каталог модуля, успадкування JDK і налаштування для SonarLint. У лістингу програми додатку А можна побачити увесь код процесу аудіо-візуалізації.

3.6 Темплейти сайту

Додаток у дипломній роботі зроблений на мові Python, а темплейти сторінок написані на типовому html. Увесь код виглядає як шаблон HTML для

вебсторінки, тому на рисунку 3.24 можна побачити приклад сторінки сайту.

```
<html>
<head>
  <script src="https://code.jquery.com/jquery-3.7.1.min.js" integrity="sha256-
/JqT3SQfawRcv/BIHPThkBvs0OEvtFFmqPF/lYI/Cxo=" crossorigin="anonymous"></script>
  <script src="/static/upload.js" type="text/javascript"></script>
  <link rel="stylesheet"
href="https://cdn.jsdelivr.net/npm/bulma@0.9.4/css/bulma.min.css">
  <link rel="preconnect" href="https://fonts.googleapis.com">
  <link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
  <link href="https://fonts.googleapis.com/css2?family=Autour+One&display=swap"
rel="stylesheet">
  <link href="/static/index.css" rel="stylesheet">
  <title>Backend</title>
</head>
<body>
{% block body %}
{% endblock %}
</body>
</html>
```

Рисунок 3.24 – Шаблон сторінки сайту index.html

На рисунку 3.25 показаний ще один приклад коду сторінки з завантаженням аудіофайлів користувачем.

```
{% extends "index.html" %}
{% block body %}
<header class="container is-fullhd has-background-light px-3 py-1">
  <h1 class="is-size-4">Audio plots</h1>
</header>
<div class="container is-fullhd p-5">
  <div class="field mt-5 columns">
    <div class="column is-vcentered is-two-thirds">
      <div class="is-size-2 mb-0 block">Upload .mp3 or .wav audio</div>
```

```
                <span class="is-size-7 has-background-warning box">Multiple files selection enabled</span>
            </div>
            <div class="column is-vcentered">
              <div class="control">
                <form method="POST" action="/upload" class="file-form">
                  <input class="input mb-3" type="file" id="files" name="files"
                       placeholder="Upload .mp3 or .wav audio" accept=".mp3,.wav" multiple>
                  <button class="button is-warning ml-auto" type="submit">Analyze</button>
                </form>
              </div>
            </div>
          </div>
        </div>
        {% endblock %}
```

Рисунок 3.25 – Шаблон сторінки сайту results.html

Потрібно не забувати, що саме пітон займається рендерингом сторінок і проганяє це все у вигляді сторінок у браузері, тому завдяки пітону, якщо знадобляться якісь шаблони, їх можна легко додати в розроблений проєкт.

ВИСНОВКИ

В процесі роботи над темою кваліфікаційної роботи були розширені і поглибленні знання з багатьох видів проєктування: процесу визначення архітектури, компонентів, інтерфейсів та інших характеристик системи.

Рівним чином, були отриманні практичні навички роботи з фреймворками, бібліотеками, різними інструментами роботи над проєктом, а також створення додатку для аналізу аудіоданих та вебдодатку для візуалізації даних. Ці навички не тільки дають людям змогу розробляти візуально привабливі платформи, але й забезпечує повну інтеграцію алгоритмів обробки звуку та інструментів візуалізації даних.

На прикладі розв’язання окремих задач, при створенні вебдодатку показана ефективність застосування фреймворку Fastapi, Matplotlib та pyaudio та розв’язування прикладних задач. Можливість використовувати ці інструменти дозволяє створювати динамічні користувальницькі інтерфейси, які покращують загальний досвід користувача, забезпечуючи ефективну взаємодію зі складними наборами аудіоданих.

Ще одна важлива навичка, набута під час вивчення цієї дисертації, – це знання обробки звукових сигналів. Розуміння тонкощів аудіоданих, від захоплення та попередньої обробки до виділення й аналізу функцій, є основоположним для отримання точної та значущої інформації. Алгоритми обробки сигналів, такі як перетворення Фур’є та вейвлет-аналіз, стають потужними інструментами, якщо їх застосовувати в контексті вебплатформи, надаючи користувачам інтерактивний зворотний зв’язок щодо їхніх аудіоданих у реальному часі.

Застосування методів машинного навчання також є життєво важливим навиком у сфері аналізу аудіоданих. Навчальні моделі для розпізнавання шаблонів і аномалій у наборах аудіоданих дають змогу вебплатформі пропонувати розширені аналітичні можливості. Незалежно від того, чи йдеться

про виявлення певних аудіоподій, класифікацію звуків або витяг значущої інформації з великих наборів даних, машинне навчання стає наріжним каменем для вдосконалення функціональності платформи.

Ефективне використання цієї вебплатформи може бути у таких галузях як: виробництво музики, охорона здоров’я, безпека та моніторинг навколишнього середовища, отримають значну користь від інтеграції інструментів аналізу аудіоданих. Наприклад, у музичному виробництві платформа може допомогти у звуковій інженерії та створенні музики, а в охороні здоров’я її можна використовувати для діагностики та моніторингу певних захворювань на основі аудіобіомаркерів.

Підсумовуючи, кульмінація навичок, набутих протягом цього проєкту, дозволяє людям у галузі аналізу аудіоданих бездоганно інтегрувати вебтехнології з передовими методами обробки аудіо. Ці навички відкривають можливості для створення універсальних і потужних програм.

ПЕРЕЛІК ПОСИЛАНЬ

1. Oppenheim A., Schafer W. Discrete-Time Signal Processing. New Jersy : Prentice-Hall, 2010. 893 p. URL: https://research.iaun.ac.ir/pd/naghsh/pdfs/UploadFile_2230.pdf (дата звернення: 28.07.2023).

2. Lyons R. Understanding Digital Signal Processing. Berkeley, Michigan : Pearson Education, 2010. 858 p. URL: https://www.iro.umontreal.ca/~mignotte/IFT3205/Documents/UnderstandingDigitalSignalProcessing.pdf (дата звернення: 28.07.2023).

3. Aurélien G. Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow. Sebastopol, California : O'Reilly Media, 2019. 856 p. URL: https://www.oreilly.com/library/view/hands-on-machine-learning/9781492032632/ (дата звернення: 28.07.2023).

4. Patterson M. The Essential Guide to Digital Signal Processing. New Jersey : Prentice Hall, 1999. 208 p. URL: https://dokumen.pub/the-essential-guide-to-digital-signal-processing-0133804429-9780133804423.html (дата звернення: 28.07.2023).

5. Data-Driven Engineering, Audio Data Analysis. URL: https://apmonitor.com/dde/index.php/Main/AudioAnalysis (дата звернення: 09.08.2023).

6. Usage of server-side programming languages broken down by ranking, technologies overview, Web Technology Surveys. URL: https://w3techs.com/technologies/cross/_programming_language/ranking (дата звернення: 09.08.2023).

7. Python for character recognition – tesseract, audio data analysis using python. URL: https://www.topcoder.com/thrive/articles/audio-data-analysis-using-python (дата звернення: 09.08.2023).

8. Exploring Audio Data with Python, An Introduction to Working with Audio Files, Working with Audio Files in Python. URL: https://medium.com/@bhagat_16083/exploring-audio-data-with-python-an-introduction-to-working-with-

audio-files-a259f9f5027f (дата звернення: 09.08.2023).

9. NEUROTECH AFRICA, Audio analysis with librosa. URL: https://blog.neurotech.africa/audio-analysis-with-librosa/ (дата звернення: 18.08.2023).

10. W3techs, Usage statistics of server-side programming languages for websites. URL: https://w3techs.com/technologies/overview/programming_language (дата звернення: 18.08.2023).

11. Starlette Io, The little ASGI framework that shines. URL: https://www.starlette.io (дата звернення: 18.08.2023).

12. Pydantic, Most widely used data validation library for Python. URL: https://docs.pydantic.dev/latest/ (дата звернення: 01.09.2023).

13. PyCharm, PyCharm IDE 2022 Fundamentals. URL: https://www.jetbrains.com/ru-ru/pycharm/download/?section=windows (дата звернення: 01.09.2023).

14. PyAudioAnalysis, An Open-Source Python Library for Audio Signal Analysis. URL: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4676707/ (дата звернення: 01.09.2023).

15. Giannakopoulos T., Pikrakis A. Introduction to Audio Analysis: A MATLAB Approach. Academic Press, 2014. 802 p.

16. Theodoridis S., Koutroumbas K. Pattern Recognition, Fourth Edition. Academic Press, 2008. 551 p.

17. Plumpe M., Acero A., Hon H., Huang X. HMM-based smoothing for concatenative speech synthesis Proc. Academic Press, 1998. 273 p.

18. GitHub, FastAPI GitHub. URL: https://github.com/tiangolo/fastapi (дата звернення: 15.07.2023).

ДОДАТОК А

Лістинг програми основної частини додатку

```
import os
import shutil
from typing import Annotated
from uuid import uuid4

from fastapi import FastAPI, Request, UploadFile, Cookie, HTTPException
from fastapi.responses import HTMLResponse, RedirectResponse
from fastapi.staticfiles import StaticFiles
from fastapi.templating import Jinja2Templates
import librosa
import librosa.display
import matplotlib.pyplot as plt
import numpy as np

app = FastAPI(
    title="Audio visualizer",
    description="Displays audio plots",
)

app.mount("/static", StaticFiles(directory="static"), name="static")
templates = Jinja2Templates(directory="templates")



@app.get("/", response_class=HTMLResponse)
async def root(
        request: Request,
```

```
        session_id: Annotated[str, Cookie()] = None,
    ):
        response = templates.TemplateResponse("upload_form.html", {"request": request})
        if not session_id:
            response.set_cookie(key="session_id", value=uuid4())
        return response


    @app.post("/upload")
    def create_upload_file(
        request: Request,
        files: list[UploadFile],
        session_id: Annotated[str, Cookie()] = None,
    ):
        if not session_id:
            raise HTTPException(status_code=404, detail="No session_id cookie")

        file_upload_path = os.path.join("/app/static/media", session_id)
        if not os.path.exists(file_upload_path):
            os.makedirs(file_upload_path)
        else:
            shutil.rmtree(file_upload_path)
            os.makedirs(file_upload_path)

        for file in files:
            with open(os.path.join(file_upload_path, file.filename), "wb") as f:
                f.write(file.file.read())

        return {"redirect_url": "/media/wait"}
```

```
@app.get("/media/wait", response_class=RedirectResponse)
def wait_for_media(
        request: Request,
        session_id: Annotated[str, Cookie()] = None,
):
    base_path = os.path.join("/app/static/media", session_id)
    result_path = os.path.join(base_path, "result/")
    if not os.path.exists(result_path):
        os.mkdir(result_path)

    for file in os.scandir(base_path):
        plt.clf()

        if not file.is_file():
            continue

        y, sr = librosa.load(file.path)
        wave = librosa.amplitude_to_db(librosa.stft(y), ref=np.max)

        plt.figure(figsize=(12, 6))
        librosa.display.specshow(wave, sr=sr, x_axis='time', y_axis='log')
        plt.colorbar(format='%+2.0f dB')
        plt.title('Spectrogram')

        plt.savefig(os.path.join(result_path, file.name + '.png'), format='png')

    return RedirectResponse(url="/media/results")



@app.get("/media/results")
```

```
    def get_results(
        request: Request,
        session_id: Annotated[str, Cookie()] = None,
    ):
        if not session_id:
            raise HTTPException(status_code=404, detail="No session_id cookie")

        result_images_path = os.path.join("/app/static/media", session_id, 'result/')
        if not os.path.exists(result_images_path):
            raise HTTPException(status_code=404, detail="No media found")

        plot_files = [
            {
                "name": filename,
                "original": os.path.join("/static/media", session_id, filename).replace('.png', ''),
                "url": os.path.join(result_images_path, filename).replace('/app', '')
            }
            for filename in os.listdir(result_images_path)
        ]
        return templates.TemplateResponse("results.html", {"request": request, "zip_files": plot_files})
```

ДОДАТОК Б

Лістинг програми відправки файлів на сервер

```
$(document).ready(function() {
    $('.file-form').on('submit', function(e){
        e.preventDefault();
        jQuery.ajax({
        type: 'POST',
        url:"/upload",
        data: new FormData($(".file-form")[0]),
        processData: false,
        contentType: false,
        success: function(response) {
            window.location.replace(response.redirect_url);
         }
    });
    })
});
```