

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»

Факультет комп'ютерних наук і технологій

(повне найменування інституту, назва факультету)

Кафедра програмних засобів

(повне найменування кафедри)

Пояснювальна записка

до дипломного проєкту (роботи)

магістр

(ступінь вищої освіти)

на тему ДОСЛІДЖЕННЯ ТА ПРОГРАМНА РЕАЛІЗАЦІЯ
МЕТОДІВ КОНВЕРТУВАННЯ АУДІОФОРМАТІВ
RESEARCH AND SOFTWARE IMPLEMENTATION
OF THE METHODS FOR AUDIO FORMAT CONVERSION

Виконала: студентка 2 курсу, групи КНТ-211м
Спеціальності 122 Комп'ютерні науки

(код і найменування спеціальності)

Освітня програма (спеціалізація)

Комп'ютерні науки

Галевська І.Д.

(прізвище та ініціали)

Керівник Колпакова Т.О.

(прізвище та ініціали)

Рецензент Казурова А.Є.

(прізвище та ініціали)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Національний університет «Запорізька політехніка»
(повне найменування закладу вищої освіти)

Інститут, факультет ФКНТ
Кафедра програмних засобів
Ступінь вищої освіти магістр
Спеціальність 122 Комп'ютерні науки
(код і найменування)
Освітня програма (спеціалізація) Комп'ютерні науки
(назва освітньої програми (спеціалізації))

ЗАТВЕРДЖУЮ

Завідувач кафедри ПЗ, д.т.н, проф.
С.О. Субботін
“ ” 202 року

З А В Д А Н Н Я
НА ДИПЛОМНИЙ ПРОЄКТ (РОБОТУ) СТУДЕНТКИ

Галевської Ірини Дмитрівни

(прізвище, ім'я, по батькові)

1. Тема проєкту (роботи) Дослідження та програмна реалізація методів конвертування аудіоформатів. Research and Software Implementation of the Methods for Audio Format Conversion.

керівник проєкту (роботи) Колпакова Тетяна Олексіївна, к.т.н., доцент,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом закладу вищої освіти від “4” листопада 2022 року №363

2. Строк подання студентом проєкту (роботи) 1 грудня 2022 року

3. Вихідні дані до проєкту (роботи) рекомендована література, технічне завдання.

4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити) 1. Аналіз предметної області. 2. Матеріали та методи. 3. Розробка архітектури програми. 4. Основні рішення щодо реалізації компонентів системи. 5. Експлуатація, тестування та експериментальне дослідження програми. 6. Додатки.

5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) Слайди презентації

6. Консультанти розділів проєкту (роботи)

Розділ	Прізвище, ініціали та посада Консультанта	Підпис, дата	
		завдання видав	прийняв виконане завдання
1-6 Основна частина	Колпакова Т.О., к.т.н., доцент		
Нормоконтроль	Дейнега Л.Ю., ст. викладач		

7. Дата видачі завдання “1” вересня 2022 року.

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів дипломного проєкту (роботи)	Строк виконання етапів проєкту (роботи)	Примітка
1	Постановка завдання роботи.	1 тиждень	Завдання, ТЗ
2	Аналіз предметної області.	2-3 тижні	Розділ 1
3	Розробка та удосконалення методів, моделей й алгоритмів вирішення задачі.	4-5 тижні	Розділ 2
4	Розробка архітектури програми.	6 тиждень	Розділ 3
5	Розробка програми.	7-8 тижні	Розділ 4
6	Тестування та експериментальне дослідження програмного забезпечення.	9 тиждень	Розділ 5
7	Оформлення пояснювальної записки та документів до неї.	10-13 тижні	Додатки
8	Нормоконтроль та рецензування.	14-15 тижні	
9	Захист роботи.	16 тиждень	

Студентка

(підпис) Галевська І.Д.
(прізвище та ініціали)

Керівник проєкту (роботи)

(підпис) Колпакова Т.О.
(прізвище та ініціали)

РЕФЕРАТ

Пояснювальна записка до кваліфікаційної роботи магістра: 87с., 33рис., 7 табл., 3 дод., 15 джерел.

КОНВЕРТУВАННЯ, WMA, MP3, АУДІО, WAV, КОРИСТУВАЧ, FLAC, AAC, ФАЙЛ, CSS, HTML, WINDOWS, JAVASCRIPT, SUBLIME TEXT, WINDOWS, БРАУЗЕР

Об'єкт дослідження – технології конвертування аудіоформатів.

Предмет дослідження – вебзастосунок для конвертування аудіофайлів різними методами.

Мета роботи – розробка вебзастосунку, який надає можливість конвертувати аудіоформати різними методами.

Матеріали, методи та технічні засоби: об'єктно-орієнтована скриптова мова програмування JavaScript, стандартна мова розмітки Hyper Text Markup Language, каскадні таблиці стилів Cascading Style Sheets, текстовий редактор Sublime Text, ноутбук з процесором Intel Core i5 під управлінням операційної системи Microsoft Windows 10.

Результати: створено вебзастосунок для конвертування аудіофайлів різними методами.

Висновки: розроблено вебзастосунок, який працює без інсталяції, і дозволяє користувачу конвертувати аудіофайли в обрані формати. Він дозволяє змінювати не тільки формат, а й бітрейт, частоту звуку, гучність. Після конвертації вебзастосунок обчислює ефективність стиснення у відсотках.

Галузь застосування – робота з аудіофайлами в Інтернеті.

ABSTRACT

Explanatory note to the diploma qualifying work of the master: 87 pages, 33 figures, 7 tables, 3 appendixes, 15 sources.

CONVERT, WMA, MP3, AUDIO, WAV, USER, FLAC, AAC, FILE, CSS, HTML, WINDOWS, JAVASCRIPT, SUBLIME TEXT, WINDOWS, BROWSER

The object of research is audio format conversion technologies.

The subject of research is a web application for converting audio files using various methods.

The purpose of the work is to develop a web application that provides the ability to convert audio formats using various methods.

Materials, methods and technical means: object-oriented JavaScript programming language, standard markup language Hyper Text Markup Language, Cascading Style Sheets, text editor Sublime Text, laptop with an Intel Core i5 processor running the Microsoft Windows 10 operating system.

Results: a web application for converting audio files using various methods was created.

Conclusions: a web application, which works without installation and allows user converting audio files to selected audio formats has been developed. It allows to change not only format but also bitrate, sound frequency, volume. After conversion web application calculates compression effectiveness percentage.

Field of application - working with audio files on the Internet.

ЗМІСТ

	С.
Перелік скорочень та умовних позначок	8
Вступ.....	9
1 Аналіз предметної області	10
1.1 Основні поняття	10
1.2 Огляд типів аудіоформатів.....	11
1.2.1 Нестиснені формати	12
1.2.2 Формати зі стисненням без втрат.....	13
1.2.3 Формати зі стисненням з втратами	14
1.3 Класифікація, аналіз і порівняльна характеристика існуючих засобів вирішення задачі	17
1.3.1 Audio Converter Online.....	17
1.3.2 Bear File Converter.....	18
1.3.3 AmoyShare.....	18
1.4 Аналіз методів конвертування.....	19
1.5 Висновки за розділом 1	21
2 Матеріали та методи	22
2.1 Основні алгоритми стиснення	22
2.2 Алгоритм додавання та перетворення даних	24
2.3 Методи перетворення даних	24
2.4 Висновки за розділом 2	25
3 Розробка архітектури програми	26
3.1 Опис основних вимог до програми	26
3.2 Структура програмної системи	27
3.3 Вибір технологій для розробки інтерфейсу	28
3.4 Вибір технологій та інструментарію для розробки програми.....	29
3.5 Основні розроблені алгоритми	33
3.6 Висновки за розділом 3	35
4 Керівництво програміста.....	36

4.1 Призначення та умови використання вебзастосунку	36
4.2 Звертання до вебзастосунку	36
4.3 Основні функції вебзастосунку, їх аргументи та призначення.....	36
4.4 Вхідні і вихідні дані	37
4.5 Висновки за розділом 4	38
5 Керівництво оператора	39
5.1 Призначення та використання вебзастосунку.....	39
5.2 Опис функцій.....	39
5.3 Виконання вебзастосунку	39
5.4 Висновки за розділом 5	43
Висновки	44
Перелік джерел посилання	45
ДОДАТОК А Технічне завдання	47
ДОДАТОК Б Текст програми	50
ДОДАТОК В Слайди презентації.....	78

ПЕРЕЛІК СКОРОЧЕНЬ ТА УМОВНИХ ПОЗНАК

AAC	– Advanced Audio Coding;
AIFF	– Audio Interchange File Format;
ALAC	– Apple Lossless Audio Codec;
API	– Application Programming Interface;
CSS	– Cascading Style Sheets;
FF	– Fast Forward;
FLAC	– Free Lossless Audio Codec;
HTML	– Hyper Text Markup Language;
JS	– JavaScript;
MDST	– Modified Discrete cosine Transform;
MP3	– Moving Picture (layer) Three;
MPEG	– Motion Pictures Expert Group;
W3C	– World Wide Web Consortium;
WASM	– WebAssembly;
WAV	– Waveform Audio File Format;
WMA	– Windows Media Audio.

ВСТУП

В нинішній час на ринку існують різноманітні пристрої, що керуються різними операційними системами. Користувачі нерідко зіштовхуються з тим, що їхні пристрої можуть не підтримувати той чи інший формат медіафайлу. Для вирішення таких проблем існують застосунки, за допомогою яких можна легко конвертувати аудіофайли в формат, який підтримує конкретний пристрій.

Однак, існуючі застосунки мають певні недоліки, наприклад не всі з них дають змогу обирати бажаний формат. Також більшість застосунків було створено для персональних комп'ютерів, що значно обмежує можливість зручного їх використання користувачами мобільних телефонів.

Доцільною є розробка вебзастосунку, у якому буде реалізоване конвертування аудіоформатів, доступне як з персонального комп'ютера, так і з мобільного пристрою, та який може бути корисним для заощадження обсягу пам'яті на носіях інформації.

Основною метою є розробка вебзастосунку, який допомагає конвертувати аудіоформати різними методами.

Завданням магістерської роботи є аналіз методів конвертування аудіофайлів та їхнє втілення у вигляді вебзастосунку зі зручним наочним інтерфейсом.

Готова система повинна передбачати можливість запуску як з мобільних пристроїв, так і з персональних комп'ютерів.

1 АНАЛІЗ ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Основні поняття

Аудіофайли – це файли певних форматів, які зберігають закодовані амплітуди звукових коливань, тобто слугують для збереження та передачі звукових сигналів у цифровому вигляді.

В залежності від типу аудіофайлу він може бути призначений для ефективного зберігання найліпшої якості звуку (нестиснені формати), для ефективної передачі стиснених даних (формати зі стисненням без втрат) або максимального стиснення і ефективної економії пам'яті з втратами якості (формати зі стисненням з втратами).

Конвертування – це процес перетворення аудіофайлів, результатом якого є зміна одного формату в інший, зміна первинного бітрейту для підвищення або пониження якості вихідного файлу.

Аудіокодек – це набір алгоритмів, методів для кодування та декодування цифрового потоку аудіоданих відповідно до заданого файлу. Метою аудіокодека є зменшення розміру аудіофайлу, не впливаючи на якість звуку. Якість відновлюється шляхом декодування стисненого файлу за допомогою того самого кодека перед відтворенням. Процес не тільки зменшує простір для зберігання, але також зменшує вимоги до пропускну здатності для передачі аудіосигналу.

Бітрейт або бітова швидкість – кількість бітів, які передають або обробляють за одиницю часу. Під час потокового передавання мультимедійних даних бітрейт визначає необхідну пропускну спроможність каналу зв'язку. Якість розпакованого звуку або зображення залежать від обраного під час кодування бітрейта [1].

Частота – це кількість коливань, що відбулися за секунду, чим більша частота, тим вищий звук. Також частота звукової хвилі сприймається людським вухом як тон. Одиницею вимірювання є герц.

В музиці зазвичай використовуються звуки, основна частота яких лежить від субконтроктави до п'ятої октави. Так, звуки стандартної 88-клавійної клавіатури фортепіано укладаються в діапазон від ноти ля субконтроктави (27,5 Гц) до ноти до п'ятої октави (4186,0 Гц).

Моно або монофонія – запис та відтворення звуку, означає одноканальний запис звуку.

Переваги моно:

- ідеально підходить для публічних виступів, так як дозволяє створити акцент на голосі людини, яка виступає;

Сtereo або стереофонія – відтворення звуку за допомогою двох та більше каналів, незалежних один від одного, завдяки цьому звук стає більш об'ємним.

Переваги стерео:

- відчуття об'ємного та глибинного звучання;
- можна відчутти розташування інструменту чи об'єкта відносно слухача;
- популярність формату;
- акустичні системи зі стереозвуком здатні передавати максимально живий звук під час концерту.

1.2 Огляд типів аудіоформатів

Наразі існує безліч аудіоформатів, популярних і не дуже, відкритих і ліцензованих. Вмілий програміст навіть може створити власний аудіоформат, якщо розробить обробник, який буде декодувати файли такого формату. Проте можна сформулювати певний достатньо короткий список форматів, популярних на розповсюджених платформах, які можуть відтворюватись існуючим стандартним програмним забезпеченням.

Як вже згадувалось вище, аудіофайли можна розділити на три великі групи:

- нестиснені формати, наприклад WAV, AIFF;
- формати зі стисненням без втрат, як то FLAC, Monkey's Audio, ALAC;
- формати зі стисненням з втратами, як наприклад MP3, AAC, OGG, WMA.

1.2.1 Нестиснені формати

Нестиснені формати дозволяють зберігати максимальну якість звуку, але через це мають суттєвий розмір файлу, що не завжди доречно для зберігання та передачі через мережу інтернет чи портативні пристрої.

WAV (Waveform Audio File Format) – один із перших аудіоформатів, який є необробленим аудіоформатом. WAV файл складається з двох частин: заголовку файлу і області даних. Формат використовує контейнери для зберігання аудіоданих, номерів доріжок, частоти дискретизації і швидкості передачі бітів. Файли цього формату сумісні з операційними системами Windows, Macintosh, Linux. WAV містить нестиснені аудіо без втрат, тож він може займати досить багато місця. Також WAV використовується в професійній музичній індустрії, щоб зберегти максимальну якість звуку [2].

AIFF (Audio Interchange File Format) – це формат нестиснених аудіофайлів, що застосовується для збереження звукових даних на персональних комп'ютерах і на інших електронних аудіопристроях. AIFF достатньо гнучкий, може зберігати багатоканальні звуки.

Він є одним з форматів, який використовується у професійних аудіо додатках та відео додатках, тому що на відміну від популярного формату MP3 в ньому звук не має втрат якості.

Як і будь-які нестиснені файли AIFF займають набагато більше дискового простору, аніж їх стиснені аналоги, прикладом є одна хвилина стерео звуку з частотою дискретизації 44,1 кГц і розміром вибірки 16 біт займає близько 10 МБ.

В таблиці 1.1 наведено порівняння нестиснених форматів.

Таблиця 1.1 – Порівняння нестиснених форматів

Формат	Розширення	Переваги	Недоліки	Автор
WAVE (WAV)	.wav	Підтримка великої кількості програм	Великий розмір файлу	1991 рік, Microsoft та IBM
AIFF	.aiff, .aif	Підтримка метаданих	Великий розмір файлу	1988 рік, Apple

1.2.2 Формати зі стисненням без втрат

Перевагою форматів зі стисненням без втрат є суттєво менший обсяг пам'яті, потрібної для зберігання файлу, при можливості за потреби відновити початкову якість звуку.

Відомим форматом такого типу є FLAC (Free Lossless Audio Codec), для роботи з яким існує аудіокодек, основна задача якого є стиснення аудіофайлів без втрати якості звуку. Цей формат використовується для прослуховування на звукових системах високого рівня, але його використання обмежене підтримкою пристроями і плеєрами, тому зазвичай його попередньо конвертують [3].

Monkey's Audio – це аудіоформат, що офіційно випускається для Microsoft Windows, а також для плагинів до медіапрогравачів. Даний аудіоформат не є формально вільним, оскільки його ліцензія накладає деякі обмеження на використання, незважаючи на те, що кодек поширюється безкоштовно разом із вихідним кодом.

ALAC (Apple Lossless Audio Codec) – відкритий аудіокодек для стиснення цифрової музики без втрат якості, який зазвичай використовується в продуктах Apple.

В таблиці 1.2 наведено порівняння форматів зі стисненням без втрат.

Таблиця 1.2 – Порівняння форматів зі стисненням без втрат

Формат	Розширення	Переваги	Недоліки	Автор
FLAC	.flac	Ефективне стиснення, швидке декодування	Щоб оцінити добротність звуку потрібна якісна апаратура	2001 рік, Xiph.Org Foundation, Джош Коалсон
Monkey's Audio	.ape	Збереження оригінального джерела звуку	Обмежена підтримка платформ та ліцензії	2000 рік, Matthew T. Ashland
ALAC	.m4a	Відновлення пошкоджених даних	Менш сумісний з апаратним забезпеченням	2004 рік, Apple

1.2.3 Формати зі стисненням з втратами

Формати такого типу забезпечують найефективніше стиснення файлів для більшої економії місця і швидкості пересилання мережею, однак при розпаковці такого файлу результат буде гірше оригіналу, тобто частина даних буде втрачена. Не слід однак вважати, що такі формати «псують» дані, бо хоч розпакований файл може суттєво відрізнятися від оригіналу при побітовому порівнянні, для людського вуха на практиці різниця буде ледь помітною.

Такі формати популярні в мережі інтернет, де можна пожертвувати якістю заради швидкості надсилання даних.

MP3 (Moving Picture (layer) Three) – найбільш популярний формат файлу для зберігання аудіоінформації. У даному форматі звуки кодуються частотним чином (без дискретних партій), також є підтримка стерео.

Даний формат MP3 є стисненням з втратами, тобто частина звукової інформації, яку вуха людини сприйняти не може або сприймається не всіма

людьми, знищується. Ступінь стиснення можна варіювати, зокрема в межах конкретного файлу [4].

Якість звуку MP3 файлів на фоні WAV залишається фактично рівним, на нього впливає змінний бітрейт, який починається від 128 кбіт/с до 320 кбіт/с в більшості випадків.

AAC (Advanced Audio Coding) – це стандарт для цифрової звукової компресії з втратами, виступає безкоштовним ліцензованим стандартом для Youtube, Sony Playstation 3, також поширений у техніці Apple. Він має обмеження швидкості передачі 250 кбіт/с, створюючи файл, подібний до файлу MP3. Файли AAC трохи менші, ніж MP3 тієї ж пісні. Обробляє звукові частоти вище 16 Гц [5].

OGG – формат потокового аудіо з відкритим вихідним кодом, який вільно розповсюджується, та пропонує стиснені аудіофайли невеликого розміру, але з порівняно високою якістю звуку. OGG може містити метадані аудіозаписи, наприклад, інформацію про виконавця. Розроблено відносно недавно, дає кращу якість звучання при однаковому бітрейті в порівнянні з найбільш поширеним форматом MP3.

Рівень стиснення дозволяє отримувати якість від середнього до високого, від 16 до 128 кбіт/с, що можна порівняти з якістю з форматом MP3 та іншими форматами.

WMA (Windows Media Audio) – ліцензований формат файлу, який був розроблений як аналог MP3 для зберігання та відтворення медіафайлів, і має більш високий рівень стиснення, підтримується практично всіма системами Windows. Формат відкривається за допомогою стандартного програвача Windows Media Player. WMA може бути перетворений за допомогою безлічі програм, званих конверторами аудіо. Іноді можна використовувати універсальні програми для конвертації [6].

Різний формат зберігання аудіо файлів впливає на об'єм файлу, що в свою чергу впливає на роботу програмного продукту.

В таблиці 1.3 наведено порівняння форматів зі стисненням з втратами.

Таблиця 1.3 – Порівняння форматів зі стисненням з втратами

Формат	Розширення	Переваги	Недоліки	Автор
MP3	.mp3	Наявність MDST (Modified Discrete cosine Transform) технології, що забезпечує хорошу якість при невеликому бітрейті, невеликий розмір	Обмеженість поширення	1993 рік, група інституту Фраунгофера MPEG (Motion Pictures Expert Group) на чолі з Карлгайнцом Бранденбургом
AAC	.aac, .m4a, .m4p	До 48 звукових каналів, надійне з'єднання	Не дуже поширений за межами Apple, залежний від конкретного пристрою	1997 рік, Bell Labs, Інститут інтегральних схем товариства Фраунгофера, Dolby Laboratories, Sony, Nokia, Coding Technologies[d]
WMA	.wma	Повна підтримка з боку Windows	Низька якість при низькому бітрейті	2002 рік, Microsoft
OOG	.ogg	Звукові дані кодування і декодування не матимуть зсувів	Трудність інтеграції нових кодеків, проблема синхронізації потоків	2004 рік, Xiph.Org Foundation

В ході аналізу популярних аудіоформатів було виявлено їх переваги та недоліки. Зважаючи на це було прийнято рішення провести дослідження на найбільш вдосконалених форматах та які актуальні на цей час.

1.3 Класифікація, аналіз і порівняльна характеристика існуючих засобів вирішення задачі

Нижче розглянуті аналоги, що реалізують конвертування популярних аудіофайлів, виділені їхні переваги і недоліки.

1.3.1 Audio Converter Online

Перший аналог вебзастосунку «Audio Converter Online» легкий у використанні, має простий та зрозумілий інтерфейс (рис. 1.1).

Переваги:

- добра оптимізація сайту.

Недоліки:

- встановлений ліміт загрузки розміру;
- ліміт конвертування файлів на добу.

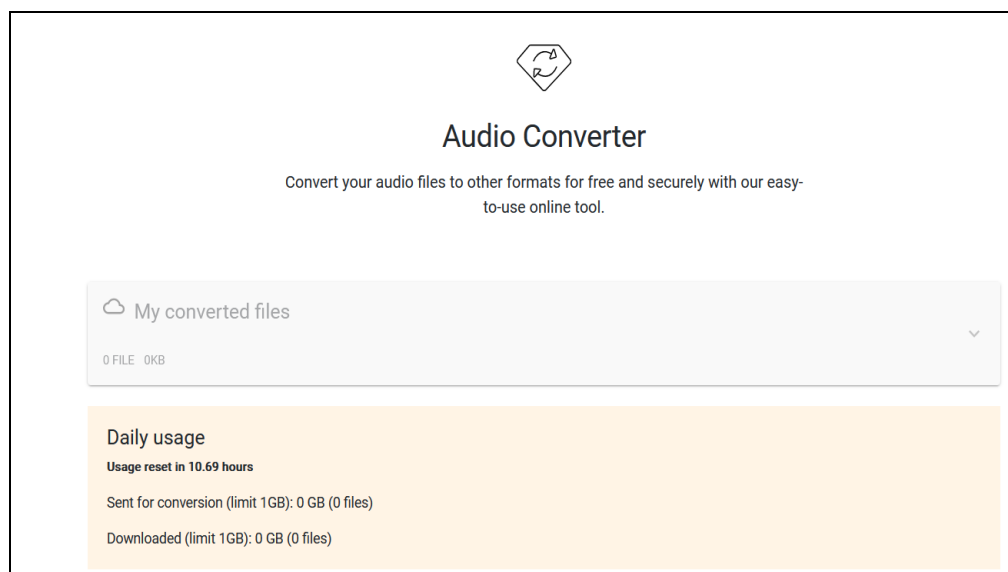


Рисунок 1.1 – Інтерфейс вебзастосунку Audio Converter Online

1.3.2 Bear File Converter

Також серед аналогів вебзастосунків для форматування аудіофайлів є «Bear File Converter» (рис.1.2).

Переваги:

- обрання мови інтерфейсу;
- зберігає аудіофайл впродовж двох годин після конвертації.

Недоліки:

- обмеження розміру файлу для завантаження;
- складний у користуванні;
- конвертування можливе тільки в один формат.

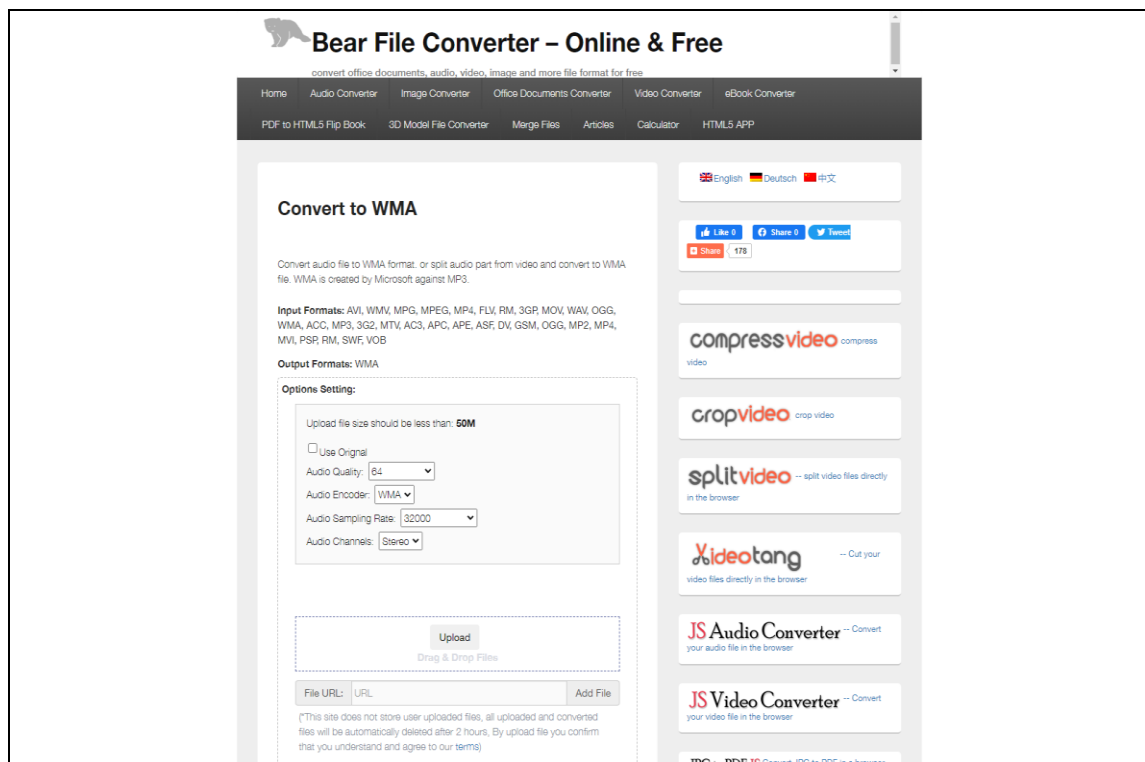


Рисунок 1.2 – Інтерфейс вебзастосунку Bear File Converter

1.3.3 AmoyShare

Інший аналог – «AmoyShare» пропонує ряд інструментів для редагування відео, конвертації аудіо до різних форматів (рис.1.3).

Переваги:

- онлайн підтримка;
- є блог-сторінка з новинами.

Недоліки:

- багато платних функцій, наприклад конвертування до найбільш популярних форматів;
- безкоштовне конвертування можливе тільки в один формат та не завжди вдається зберегти файл.

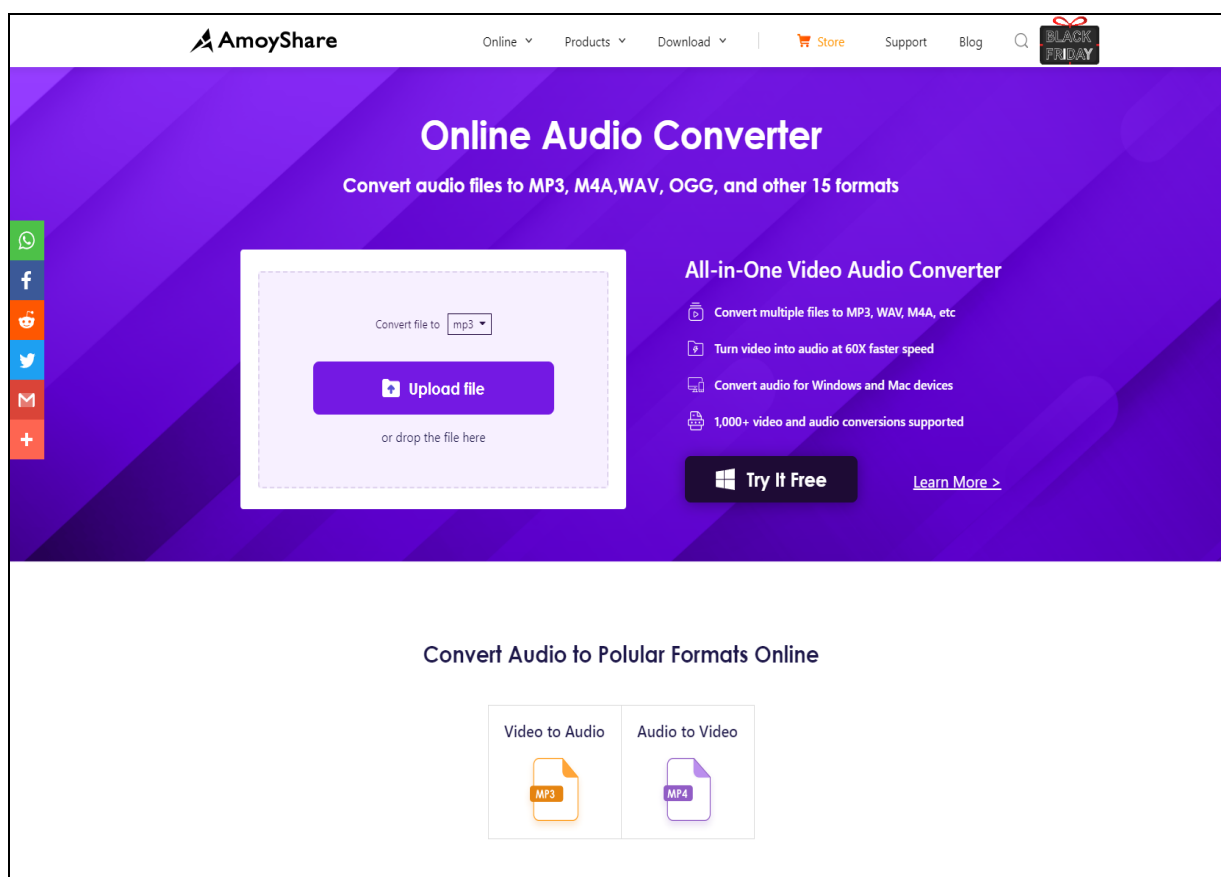


Рисунок 1.3 – Інтерфейс вебзастосунку AmoyShare

1.4 Аналіз методів конвертування

Як було описано вище, аудіофайли поділяють на три великі групи. Відповідно, з цими групами пов'язані і методи конвертування з одного формату в інший (рис. 1.4).

Метод стиснення без втрат – це метод, де вхідні дані можуть бути відновлені без погіршення якості звуку та без будь-яких втрат послідовностей символів, тобто відновлені дані ідентичні вхідним.

Метод без втрат та стиснення лише змінює формат файлу, не змінюючи при цьому початкову інформацію файлу [7].

Метод стиснення з втратами перетворюючи дані дозволяє забезпечити максимальний ступінь стиснення вихідного файлу, але при цьому частина даних не буде ідентична вхідним даним. Перевага таких методів полягає в тому, що істотно перевершують по ступеню стиснення, продовжуючи задовольняти поставленим вимогам. Методи стиснення з втратами часто використовуються для стиснення звуку або зображень, у таких випадках файл може відрізнятися від оригіналу, але практично не відрізняється для людського вуха або ока в більшості практичних застосувань.

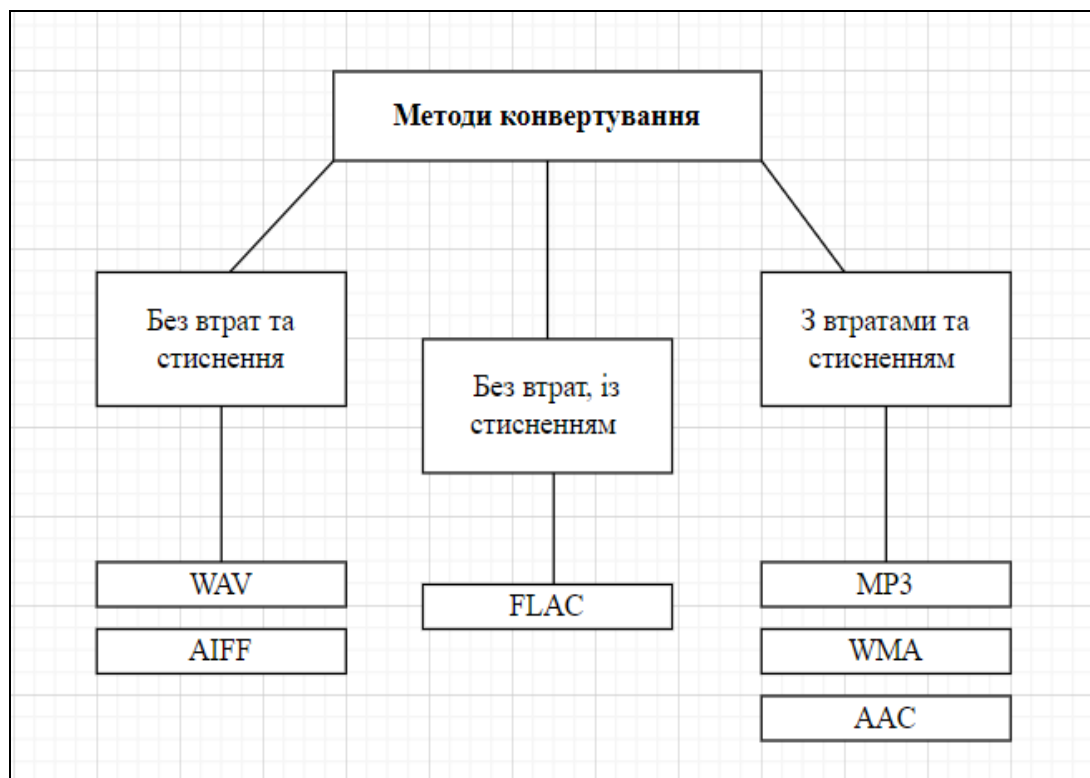


Рисунок 1.4 – Схема методів конвертування

Враховуючи різні підходи стиснення даних, для кожного типу задач та вхідних даних існують методи, що найкраще працюють з тим чи іншим

форматом. Тому для отримання найкращих результатів при стисненні завжди є важливим вибір найкращого методу стиснення даних в залежності від вимог та особливостей формату вхідних даних.

1.5 Висновки за розділом 1

В першому розділі було розглянуто поняття аудіоформату, виконано порівняння найбільш популярних форматів. Були розглянуті деякі застосунки, які виконують перетворення форматів аудіо, розглянуто їх переваги та недоліки, а також було доведено актуальність цієї теми.

Аналіз та програмна реалізація методів конвертації аудіофайлів у вигляді вебзастосунку, який дозволить користувачеві конвертувати аудіофайли різних форматів є актуальною задачею. Система повинна складатися з клієнтської частини, мати перелік доступних форматів для конвертування файлів та встановлення бітрейту, гучність та частота звуку. Також система повинна мати зручний інтерфейс, завдяки якому користувачеві буде зрозуміло користуватись застосунком.

Система повинна надавати користувачеві наступні функції:

- обрання необхідного формату для конвертування аудіофайлу;
- збереження аудіофайлу у певний формат;
- переформатування файлу у інший обраний формат користувачем;
- обрання типу відтворення;
- обрання частоти звуку;
- обрання бітрейту;
- перегляд результату конвертування у відсотках;
- завантаження до локального сховища даних.

2 МАТЕРІАЛИ ТА МЕТОДИ

2.1 Основні алгоритми стиснення

Існує декілька підходів до проблеми стиснення інформації. Одні мають досить складну теоретично-математичну основу, складні в реалізації, однак забезпечують максимальний ефект від стиснення.

Будь-який метод, підхід чи алгоритм, який реалізовує стиснення даних призначений для усунення надлишковості інформації, призначений для зменшення об'єму пам'яті, що потрібно для зберігання інформації часто стиснення інформації називають: компресією інформації.

Метою стиснення є отримання компактного файлу. Основними характеристиками процесів стиснення та результатом роботи є наступне:

- ступінь стиснення;
- швидкість стиснення;
- якість стиснення;

Усі алгоритми стиснення діляться на два класи: статистичні, словникові (табл. 2.1). В основі статистичного класу лежить якісне стиснення, так як перед кодуванням спочатку збирає інформацію про файл, а потім виконує розподіл ймовірностей. Виходячи з цих переваг розглянемо алгоритм Хаффмана.

У алгоритмі Хаффмана є два підходи:

- статичне кодування. У вихідний потік записується кодова таблиця яка потім використовується для декомпресії;
- динамічне кодування. Змінює схему кодування, в залежності від змін вхідного потоку.

Арифметичне кодування також поділяється на два підходи:

- статичне кодування. Досягається теоретична границя ступеня стиснення ентропії вхідного потоку;
- динамічне кодування. Двійковий дріб з результатом стиснення є послідовністю двійкових цифр із цього дробу.

Таблиця 2.1 – Приклад алгоритмів стиснення

Алгоритми стиснення			
Статистичні алгоритми		Словникові алгоритми	
Кодування Хаффмана	Арифметичне кодування	На основі алгоритму LZ77	На основі алгоритму LZ78
Статичне кодування	Статичне стиснення	Класичні однорівневі	LZW-алгоритм
Динамічне Стиснення	Динамічне Стиснення	Двоступеневі	MW-алгоритм
-	-	LZHUF(LZ77-Huffman)	AP-алгоритм
-	-	LZARI(LZ77-арифмет)	Y-алгоритм

На прикладі формату MP3, який часто використовується, розглянемо роботу відомого алгоритму Хаффмана. У форматі MP3 закладений алгоритм Хаффмана. Цей процес дозволяє поліпшити ступінь стиснення відносно однорідних сигналів. Цей вид кодування зазвичай дозволяє стиснути аудіодані 10:1 або вище. На схемі нижче зображено процес обробки файлу. Якщо файл має однакові значення вхідного потоку даних, то буде виконуватися стиснення алгоритмом Хаффмана (рис. 2.1).

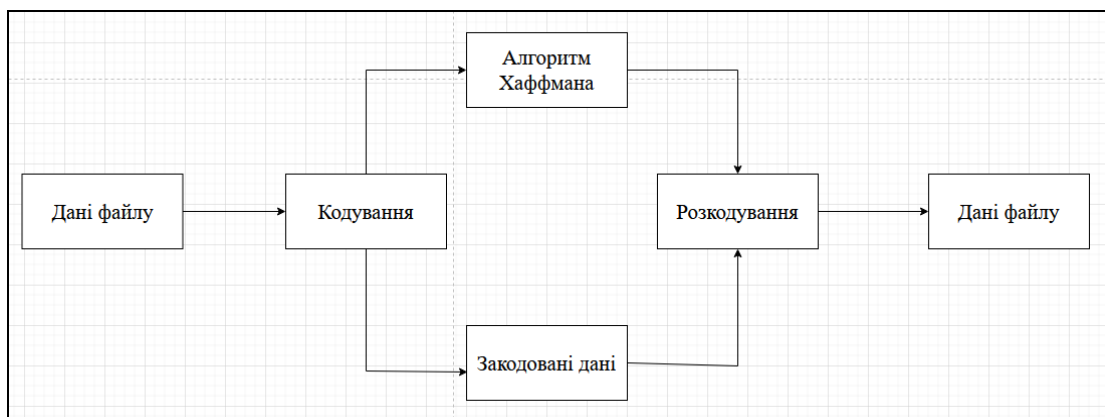


Рисунок 2.1 – Схема алгоритму конвертування

2.2 Алгоритм додавання та перетворення даних

Для роботи системи було обрано алгоритм HTML5 FileReader API. Це алгоритм додавання файлів до вебзастосунку без фактичного їх завантаження на сервер. FileReader дозволяє зчитувати вміст файлу, який був вибраний користувачем на його пристрої за допомогою діалогового вікна, безпосередньо в браузері за допомогою мови JavaScript без завантаження його на сервер.

У таблиці нижче представлені браузери, які підтримують цю технологію (табл. 2.2).

Таблиця 2.2 – Список браузерів, які підтримують технологію FileReader

Назва браузера	Версія
Firefox	106
Chrome	107
Opera	75
Safari	14.1

HTML5 FileReader API включає в себе наступні переваги:

- незалежність від плагінів;
- є можливість відображати інформацію про файл;
- можна обрати одразу декілька файлів через стандартне поле вибору файлу;
- є можливість виконання інтерфейсу drag and drop, тобто претягування файлу за допомогою «миші»;
- дозволяє контролювати процес завантаження.

2.3 Методи перетворення даних

Основна відмінність методів стиснення з втратами та стиснення без втрат полягає в тому, що при стисненні з втратами створюється близьке

співвідношення даних, а без втрат створюються точні вихідні дані, які представлені у порівняльній характеристиці (табл. 2.3).

Таблиця 2.3 – Порівняльна характеристика

Характеристики	Стиснення з втратами	Стиснення без втрат
Процес	Видаляє частину даних, що не можна виявити	Відтворює точні дані
Зменшення розміру	Значно	Незначно
Якість даних	Погіршується	Не погіршується
Використання	Зображення, аудіо та відео	Текст або програма, зображення та звук

2.4 Висновки за розділом 2

В ході виконання наступного етапу була представлена схема формування та обробки інформації в автоматизованій інформаційній системі. Також представлений алгоритм перетворення даних та порівняльна характеристика двох методів для конвертування аудіофайлів, а саме: стиснення з втратами та стиснення без втрат.

3 РОЗРОБКА АРХІТЕКТУРИ ПРОГРАМИ

3.1 Опис основних вимог до програми

Після проведеного аналізу щодо реалізації методів конвертування, було висунуто наступні вимоги – це мати зручний та зрозумілий інтерфейс, бути кросбраузерною. Вимоги до функціонування вебзастосунку були сформовані виходячи з аналізу існуючих методів, а також для позбавлення недоліків, які були виявлені у аналогах.

До основних функцій програми належать:

- обрання необхідного формату для конвертування аудіофайлу;
- збереження аудіофайлу у певний формат;
- переформатування файлу у інший обраний формат користувачем;
- обрання типу відтворення;
- обрання частоти звуку;
- обрання бітрейту;
- перегляд результату конвертування у відсотках;
- завантаження до локального сховища даних.

В якості мови розробки було обрано мову програмування JavaScript, тому що:

- для її використання не потрібен компілятор, вона інтерпретується «на льоту» в браузері;
- за її допомогою можна виконати валідацію вхідних даних;
- використання JavaScript робить сайт більш інтерактивним, при цьому дозволяючи йому завантажуватись швидше і легше, ніж сайти, написані на інших мовах програмування.

3.2 Структура програмної системи

Структура вебзастосунку складається з наступних файлів:

- папка першого рівня «fileless.ua», у якій зберігаються всі дані керуванням та користуванням вебзастосунком;
- папка другого рівня «assets», у якій зберігаються папки стилів, шрифтів та скриптів;
- головний фал «index.html», який містить гіпертекстову розмітку стилів;
- файл «style.css» для зберігання таблиці стилів;
- папка «fonts», що містить шрифти для вебзастосунку;
- папка «img», яка містить зображення, які використовуються в інтерфейсі;
- файл «script.js» у якому міститься скриптовий алгоритм конвертування аудіофайлів.

Нижче представлена ієрархія файлів вебзастосунку (рис. 3.1).

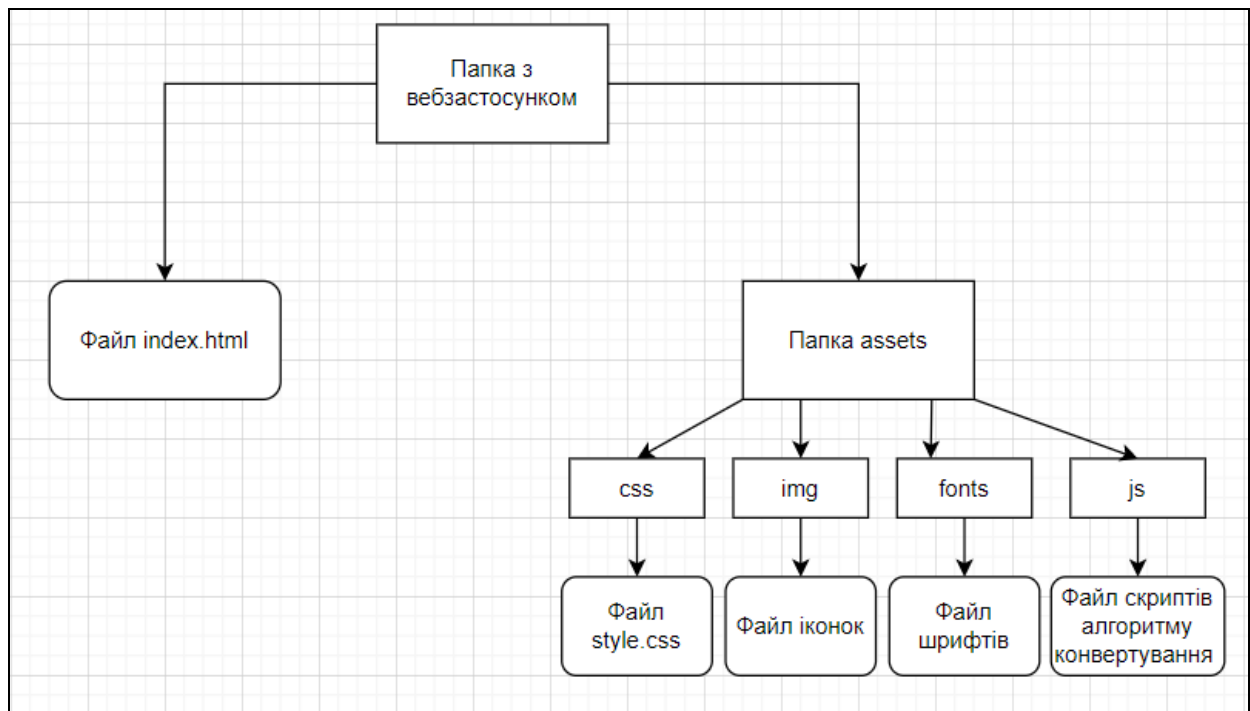


Рисунок 3.1 – Ієрархія файлів вебзастосунку

3.3 Вибір технологій для розробки інтерфейсу

Для реалізації інтерфейсу було прийнято рішення використовувати Figma.

Figma – графічний онлайн-сервіс для розробки інтерфейсів та прототипування з можливістю організації спільної роботи. Працює у двох форматах: у браузері та як клієнтський додаток на десктопі користувача. Також зберігає онлайн версії файлів, в яких працював користувач [8].

Також у Figma відкритий вихідний код. Програма може працювати в розрахованому на багато користувачів режимі на будь-якому пристрої, та у будь-якому браузері.

Для написання коду було обрано текстовий редактор Sublime Text, розроблений Джоном Спіннером у 2011 році на мовах програмування C++ та Python. Також редактор підтримує велику кількість мов програмування і має можливість підсвічування синтаксису для C, C++, C#, CSS, Dylan, Erlang, HTML, Groovy, Haskell, Java, JavaScript, LaTeX, Lisp, Lua, Markdown, Perl, Python, R, Ruby та інші.

Для створення вебсторінок та вебдодатків вебсерверу було обрано стандартну мову гіпертекстової розмітки HTML (HyperText Markup Language).

HTML – це мова опису структури сторінок документів, яка дозволяє звичайний текст формувати в абзаци, заголовки, списки та інші структури, створювати посилання на інші сторінки, в якій є інструкції з форматування, які називаються тегами, містять конкретну інформацію [9].

Для стилізації сайту було використано каскадні таблиці стилів CSS.

CSS (Cascading Style Sheets) – це мова, яка визначає стиль HTML-документів та включає пара-метри, пов’язані із шрифтами, кольорами, полями, лініями, висотою, шириною, фоновими зображеннями тощо. Також CSS запропонував Хокон Біум Лі, де потім було схвалено консиліумом W3C (World Wide Web Consortium) [10].

3.4 Вибір технологій та інструментарію для розробки програми

3.4.1 Мова програмування JavaScript

Для інтерактивності сайту було обрано мову програмування JavaScript (JS).

JavaScript – це об'єктно-орієнтована мова програмування, що дозволяє реалізувати ряд рішень в вебдокументах, також підтримується додатками, що працюють з дизайном сайту. Вона допомагає зробити сторінки сайту більш інтерактивними, обробляє дії користувачів сайту. Поряд з HTML і CSS, JavaScript – третій важливий блок, на основі якого будується більшість стандартних вебінтерфейсів [11].

Ядро JavaScript включає цілий ряд функцій, які дають наступні можливості:

- зберігати дані в змінних;
- активувати частину коду у відповідності з певними сценаріями, які здійснюються на сторінці сайту;
- створювати динамічний контент;
- є можливість використовувати велику кількість бібліотек;
- управляти мультимедійними можливостями, тобто працювати з відео, анімувати зображення [12].

3.4.2 Бібліотека JQuery

Для більш простої взаємодії користувача з застосунком було використано бібліотеку JQuery.

JQuery – набір функцій JavaScript, що фокусується на взаємодії JavaScript і HTML [13].

Також JQuery має ряд особливостей, які значно впливають на вибір цієї бібліотеки, а саме:

- кросбраузерність – модулі бібліотеки однаково добре працюють в будь-якому браузері;
- швидкість введення коду, якщо в JavaScript найпростіші команди використовують кілька рядків коду, то в JQuery можна заощадити;
- зрозумілість і зручність читання команд;
- доступне з'єднання різних плагінів можливість завдання їх стилів;
- вдало і просто поєднує між собою Javascript, HTML і файли CSS-стилів.

3.4.3 HTML FileReader API

Як згадувалось у пункті 2 було використано HTML FileReader API.

Він дає можливість вебсховищу підтримуватися у HTML5 на хорошому рівні. Але це не єдиний спосіб отримання інформації на веб-сторінках. У цю область потроху вводиться стандарт File API, та має добрий рівень підтримки на всіх сучасних браузерах.

Вебзастосунок надсилає запит у браузері через Інтернет до API. API звертається до сервера та віддає інформацію за певним запитом (рис 3.2).

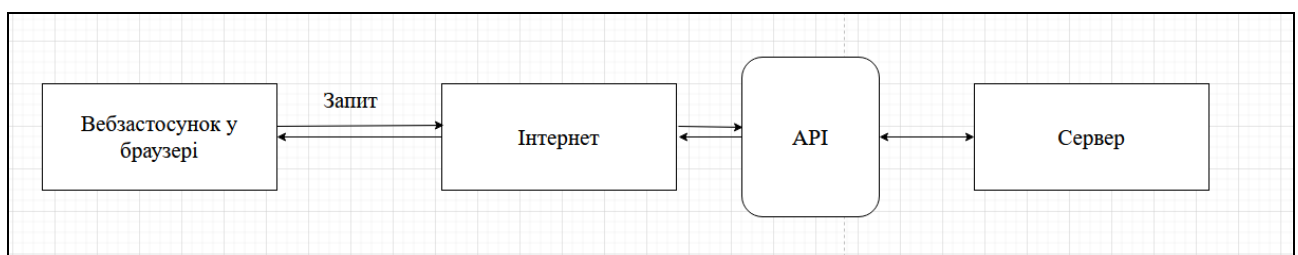


Рисунок 3.2 – Алгоритм роботи HTML FileReader API

3.4.4 Мультимедійна бібліотека FFmpeg

Для обробки аудіо була скомпільована та використана мультимедійна бібліотека FFmpeg.

FFmpeg – програма, назва якої походить від експертної групи MPEG та FF (Fast Forward). Це колекція безкоштовних програмних засобів для мультимедіа. За допомогою нього можна конвертувати між різними мультимедійними форматами (аудіо та відео), записувати, виправляти пошкоджені відео тощо [14].

На схемі представлено основний алгоритм роботи бібліотеки з даними файлу (рис. 3.3).

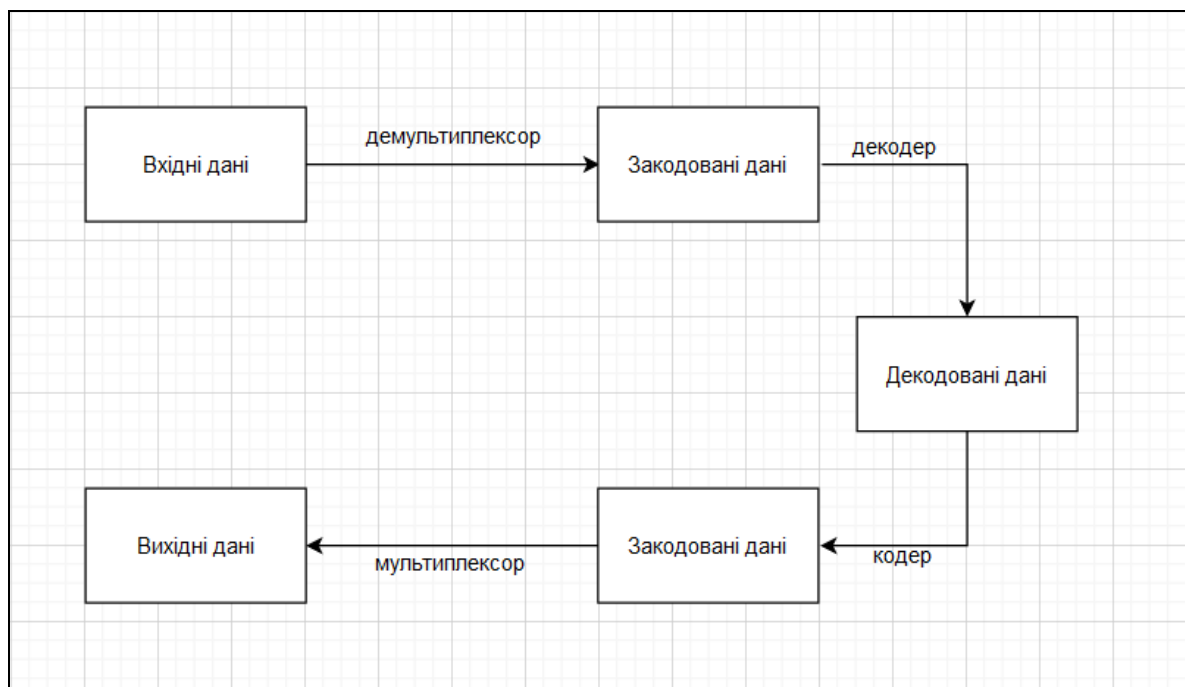


Рисунок 3.3 – Алгоритм роботи бібліотеки з даними файлу

Детальний опис алгоритму роботи бібліотеки FFmpeg:

- вхідними даними виступають аудіофайли;
- демультиплексор обробляє формат аудіофайлу, виявляє тип даних;
- в ролі закодованих даних виступає результат виявленого типу даних;
- декодер розкодує інформацію для обробки в інший формат;
- в ролі декодованих даних можуть виступати бітрейт, частота, канал, гучність, тип даних;
- кодер виконує кодування за новими параметрами;
- закодовані дані містять нову закодовану інформацію;

- мультиплексор перетворює встановлені параметри у новий тип форматує;
- вихідними даними слугує перетворений аудіофайл.

Також FFmpeg дозволяє використовувати різні кодеки, що в свою чергу кодують та декодують аудіо та відео файли. Кодеки не призначені для самостійного відкриття та відтворення файлів, їх завдання – допомогти відкрити файли.

Для зборки бібліотеки FFmpeg був використаний компілятор emscripten. Це інструмент, створений для компілювання програм, які використовують стандартні бібліотеки C і C++ (рис. 3.4).

Основною метою алгоритму, полягає у створенні інструментів, які запускають код, незалежно від мови програмування якою він написаний, містить багато мультимедійних бібліотек, API (Application Programming Interface) для інтеграції з веб-API і кодом JavaScript.

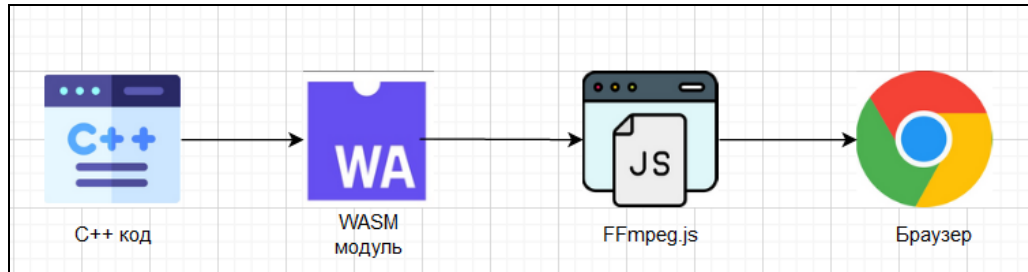


Рисунок 3.4 – Схема процесу компіляції бібліотеки

Детальний опис процесу роботи компілятора emscripten:

- вхідними даними виступають бібліотеки написані на коді C++;
- WASM (WebAssembly) модуль є проміжним кодом для виконання в браузері застосунків, скомпільованих з різних мов програмування [15];
- FFmpeg.js – скомпільований набір бібліотек;
- браузер виступає середовищем використання бібліотеки.

3.4.5 Використання бібліотеки libavcodec

Libavcodec – бібліотека кодеків із відкритим кодом для кодування та декодування відео та аудіоданих, є частиною бібліотеки FFmpeg.

Libavcodec є невід’ємною частиною багатьох мультимедійних програм. Популярні медіаплеєри Kang Multimedia Player, xine та VideoLAN Client використовують його як основний вбудований механізм декодування, який дозволяє відтворювати аудіо та відеофайли. Libavcodec також використовується в програмах для редагування та перекодування відео, таких як Avidemux, MEncoder або Kdenlive як для декодування, так і для кодування.

Libavcodec містить реалізацію декодера та кодера власних форматів. Таким чином, значна частина зворотного проектування є частиною розробки libavcodec. Наявність таких кодеків, доступних у стандартній структурі libavcodec, дає ряд переваг порівняно з використанням оригінальних кодеків, зокрема підвищену портативність та продуктивність.

3.5 Основні розроблені алгоритми

В процесі розробки вебзастосунку були реалізовані алгоритми функціонування для користувача.

Було розроблено алгоритм, в основі якого лежить діаграма прецедентів, на якій зображено основні дії користувача. Користувач завантажує аудіофайл з певним форматом у систему, після чого йде процес конвертації. Якщо він завантажить помилковий формат, буде видавати помилку (рис 3.5).

Також був розроблений алгоритм додавання файлу до системи та його обробка (рис. 3.6). Користувач здійснює перетягування файлу у поле «Завантажити файл». Далі він виконує бажані налаштування, після чого натискає на кнопку «Конвертувати». Після обробки запиту створюється посилання на оброблений файл до атрибутів кнопки натиснувши на «Завантажити».

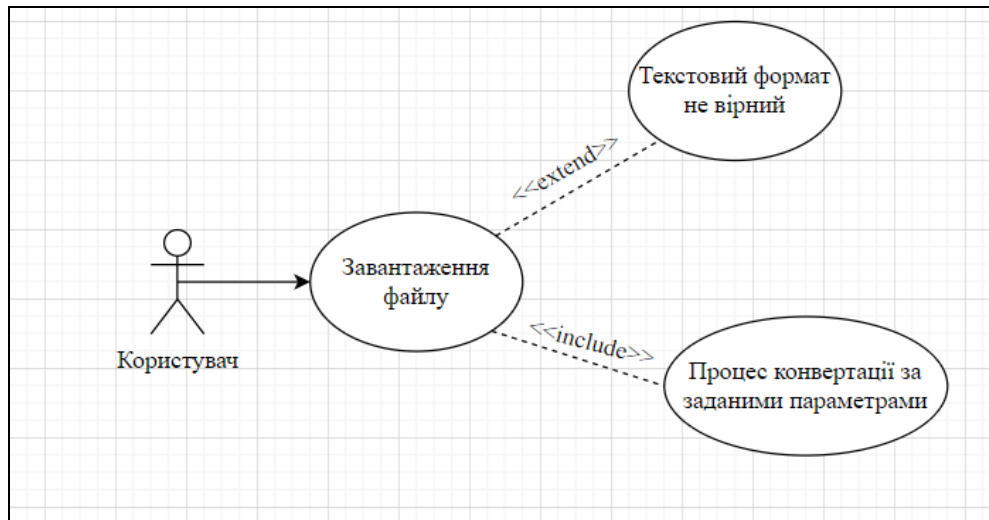


Рисунок 3.5 – Інтерфейс взаємодії користувача з системою

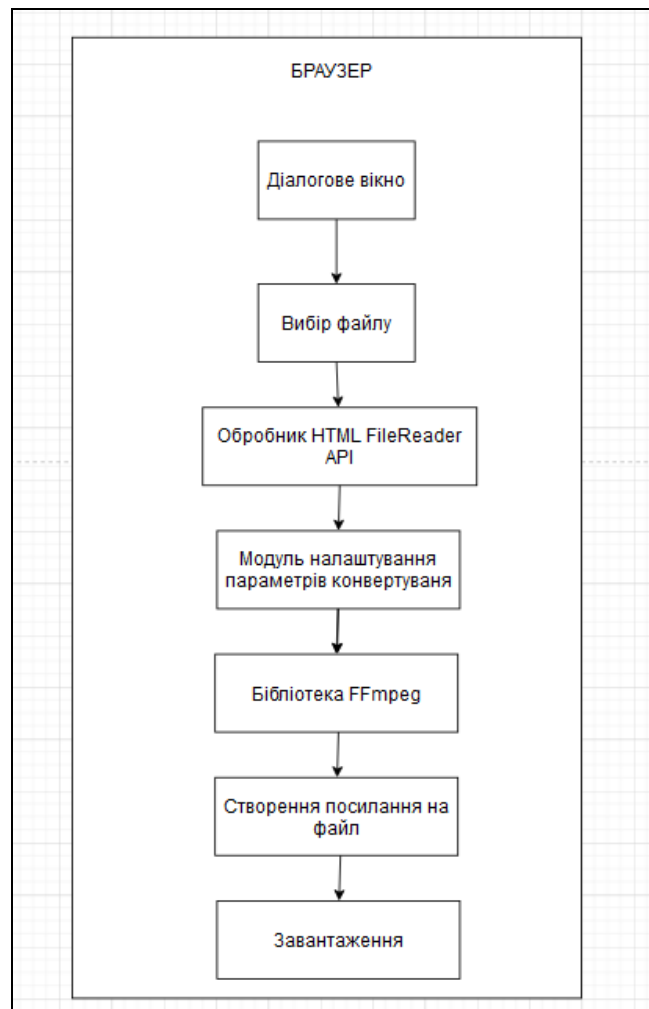


Рисунок 3.6 – Алгоритм додавання файлу та його обробка

3.6 Висновки за розділом 3

В ході виконання роботи було сформовано основні вимоги щодо реалізації вебзастосунку, розроблене функціонування та поставлені задачі щодо функціонування програми.

Обрано мову програмування та інструментарій для розробки програмного продукту.

Також обрана бібліотека для злагодженого функціонування системи.

4 КЕРІВНИЦТВО ПРОГРАМІСТА

4.1 Призначення та умови використання вебзастосунку

Розроблений вебзастосунок призначений для конвертування аудіоформатів, без встановлення додаткових програм або скриптів, призначений для використання як на мобільних пристроях так і на персональних комп'ютерах.

Для повноцінного використання вебзастосунку необхідно мати комп'ютер з наступними мінімальними вимогами:

- частота процесора 2.0 ГГц;
- оперативна пам'ять не менше 1 Гб;
- браузер Mozilla, Opera, Google Chrome, Microsoft Edge.

4.2 Звертання до вебзастосунку

Після звернення до вебзастосунку буде відображене головна сторінка з основними компонентами, меню та інша секція для подальшої роботи з перетворенням аудіофайлів.

Робота програми організована у вигляді застосунку, з використанням графічних елементів: зображення, кнопок, тексту, лейблів.

Дані компоненти реалізовані візуальними елементами за допомогою CSS, а функціонал скриптовою мовою JS.

4.3 Основні функції вебзастосунку, їх аргументи та призначення

Вебзастосунок складається з функцій та аргументів, які необхідні для коректного функціонування.

За допомогою бібліотеки FFmpeg було проаналізовано та розроблено функції для перетворення аудіофайлів (табл 4.1).

Таблиця 4.1 – Функції, їх аргументи та призначення

Функція	Аргументи	Призначення
getBitrate()	Немає	Отримання даних про бітрейт, який обрав користувач
getOutFormat()	Немає	Отримання даних про формат, який обрав користувач
readInputFile()	file	Зчитування доданого файлу
handleDropSelect()	event	Перетягування файлу у діалогове вікно
timeToSeconds()	time	Перетворення хвилин на секунди для конвертування

Для більш детальної інформації про результат конвертування було додано відображення до та після стиснення файлів, розроблено відсоткове співвідношення файлів (рис. 4.1).

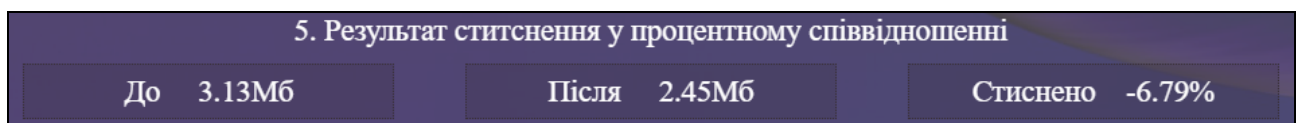


Рисунок 4.1 – Результат стиснення

4.4 Вхідні і вихідні дані

В ролі вхідних даних виступає завантажений користувачем аудіофайл у форматі mp3, wav, flac або іншому аудіоформаті з переліку підтримуваних.

Вихідними даними є файл, перетворений в інший формат, обраний користувачем.

4.5 Висновки за розділом 4

В ході виконання роботи було проаналізовано та створено основні функції для роботи з аудіофайлами, та створено вебзастосунок, для керування яким не потрібна допомога адміністратора

Застосунок обчислює ефективність результату перетворення файлів у відсотках.

5 КЕРІВНИЦТВО ОПЕРАТОРА

5.1 Призначення та використання вебзастосунку

Вебзастосунок призначений для конвертування аудіоформатів онлайн різними методами, має єдину форму інтерфейсу, в якій міститься блок для детального налаштування аудіофайлів.

Вебзастосунок працює у таких браузерах як: Mozilla, Opera, Google Chrome, Microsoft Edge.

Для отримання доступу до вебзастосунку треба мати стабільне підключення до мережі Інтернет.

5.2 Опис функцій

Інтерфейс вебзастосунку складається з трьох секцій, головного меню та безпосередньо самого функціоналу конвертування.

Перша секція містить логотип та меню системи.

У другій секції розміщено функціонал системи, а саме:

- блок завантажень файлу;
- блок обрання методів для конвертування файлів;
- блок налаштування форматів, де можна обрати бітрейт, гучність або частоту аудіо;
- блок керування процесом конвертування;
- блок результату стиснення, вираженому у відсотках.

5.3 Виконання вебзастосунку

Для входу на сторінку даного вебзастосунку, треба перейти за посиланням «<http://FileLess>». Перед початком конвертування файлів переконатися у робото спроможності Інтернету. Після запуску вебзастосунку FileLess відкриється вікно завантаження (рис. 5.1).



Рисунок 5.1 – Вікно завантаження сторінки

Після звертання за вказаною адресою відкриється головна сторінка (рис. 5.2), на якій розміщено меню, яке складається з наступних пунктів: «Головна», «Конвертування файлу» та «Про нас».

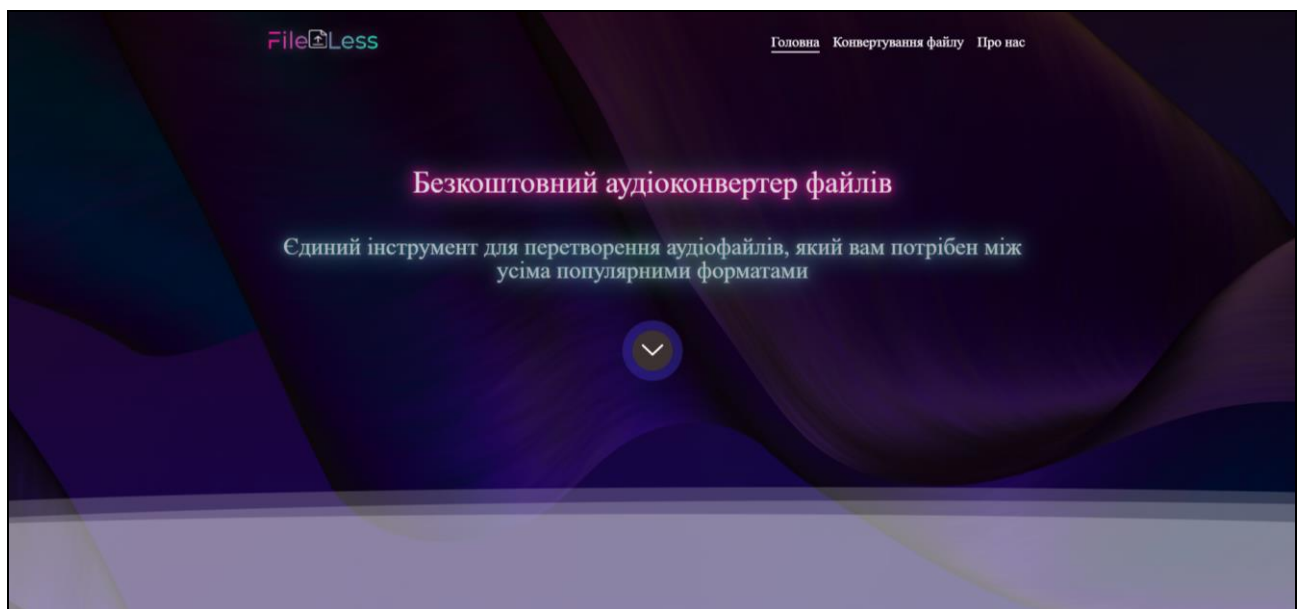


Рисунок 5.2 – Головна сторінка

В секції «Конвертування файлу» (рис. 5.3) користувач може завантажити файли за допомогою діалогового вікна або перетягнути файл у поле «Перетягніть файл».

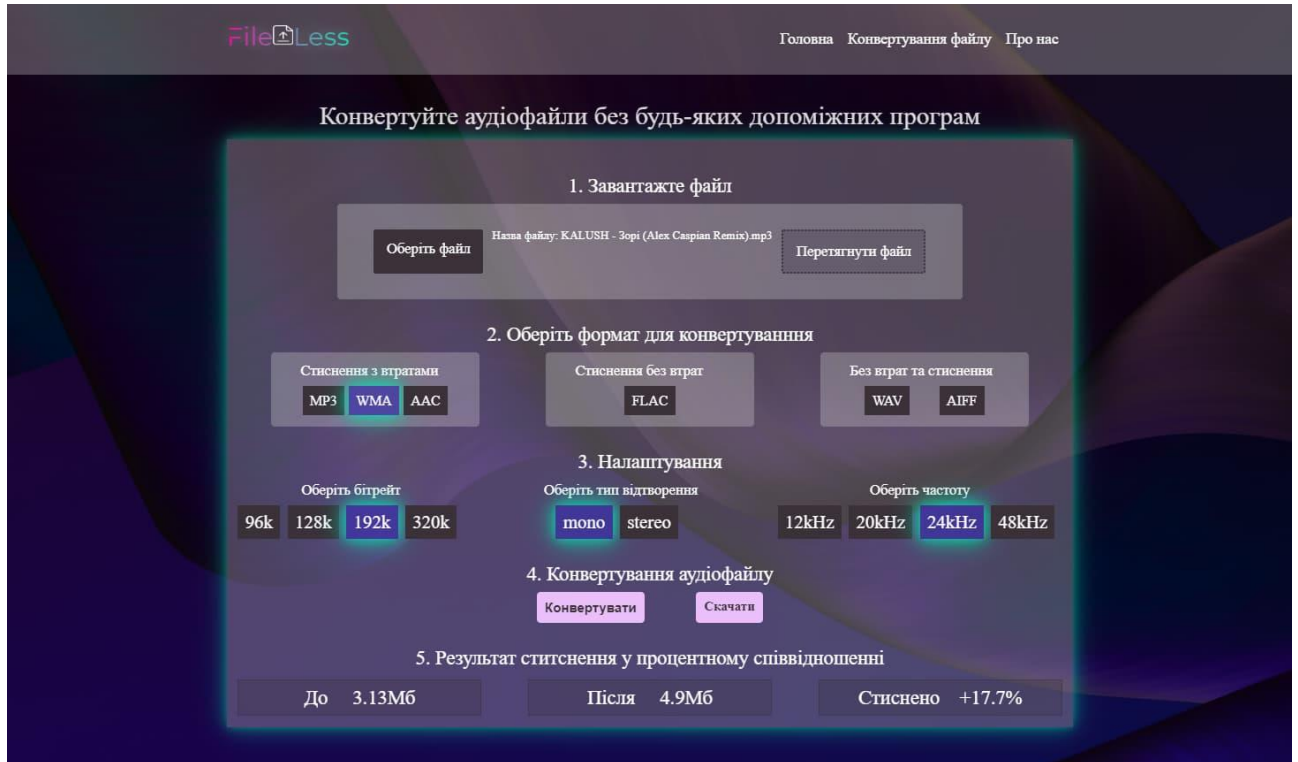


Рисунок 5.3 – Функціонал вебзастосунку

Якщо користувач завантажує файл невірної формату, то відобразиться помилка про невірний формат (рис. 5.4).

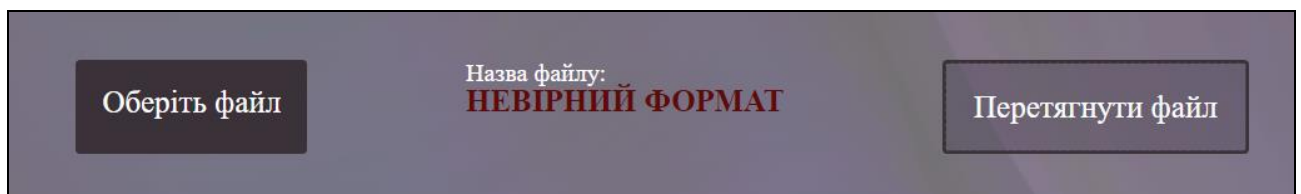


Рисунок 5.4 – Помилка про невірний формат

На другому етапі користувач може обрати необхідний йому формат для конвертування натиснувши безпосередньо на сам формат (рис. 5.5) та скористатися додатковими функціями налаштування аудіофайлу (рис. 5.6).

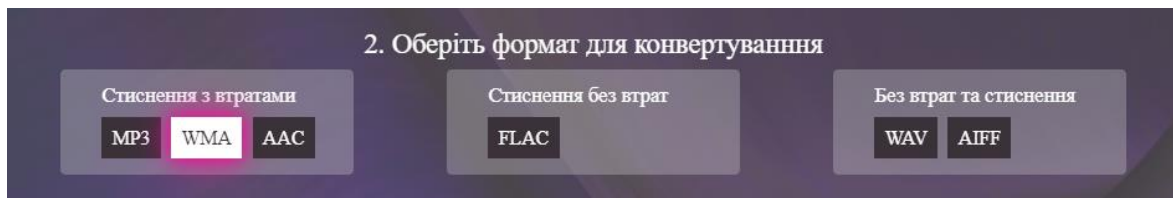


Рисунок 5.5 – Обрання формату

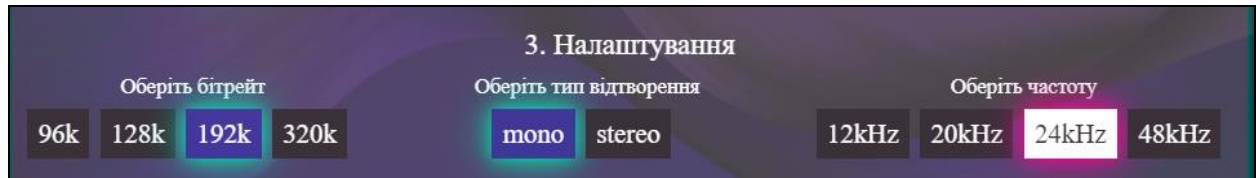


Рисунок 5.6 – Налаштування додатковими функціями

Після цих маніпуляцій користувач переходить до конвертування, натиснувши на кнопку «Конвертувати» (рис. 5.7). В залежності від обраних параметрів процес конвертування займає певний час. Після успішного завершення процесу користувач може скачати файл у новому форматі, натиснувши на кнопку «Скачати». Після конвертації користувач може наочно побачити ефективність конвертування зі стисненням у відсотковому співвідношенні.

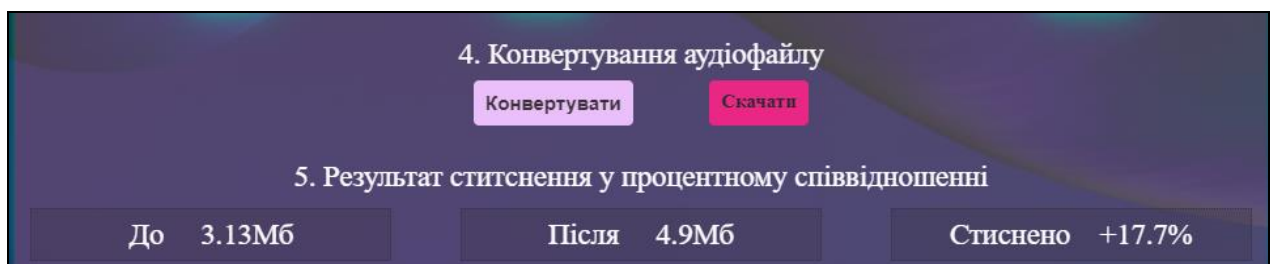


Рисунок 5.7 – Процес конвертування

Також, щоб переконатися у вірності перетворення файлу користувач може відкрити його властивості (рис. 5.8).

Файл	
Ім'я	KALUSH - Зорі (Alex Cas...
Тип елемента	Файл WMA
Шлях до папки	C:\Користувачі\Irena\За...
Дата створення	08.12.2022 17:34
Дата змінення	08.12.2022 17:34
Розмір	4.90 МБ
Аудіо	
Бітова швидкість передава...	192 кбіт/с

Рисунок 5.8 – Інформація про перетворення

Користувач може продивитись інформацію про сайт, натиснувши кнопку меню «Про нас» (рис. 5.9).

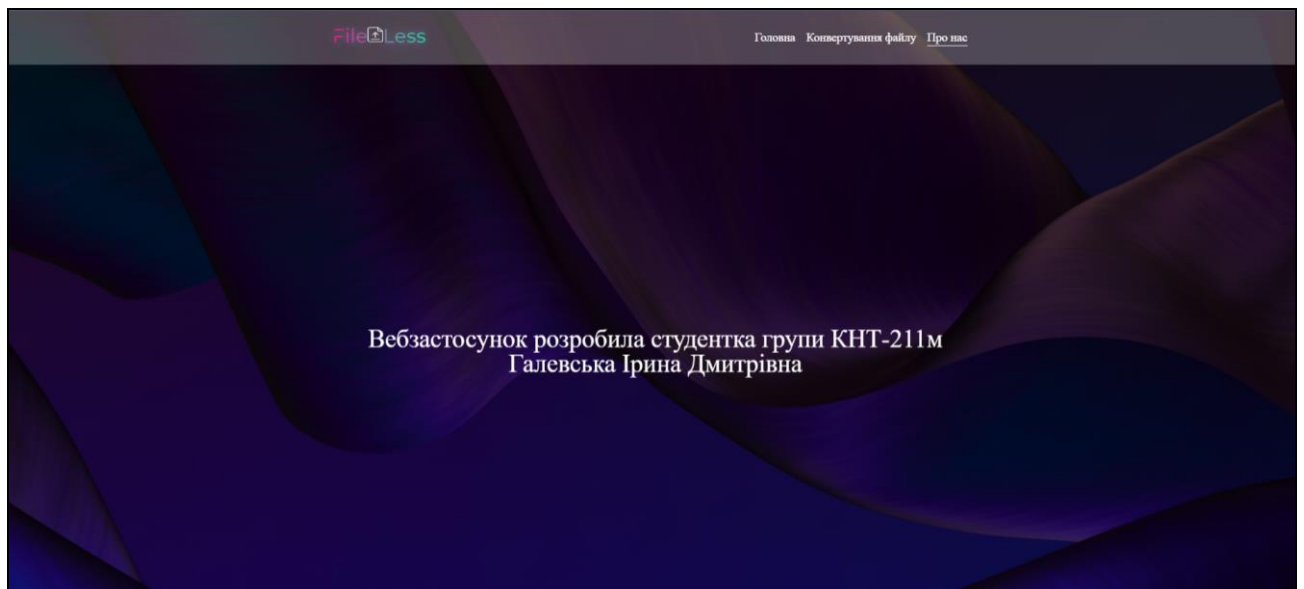


Рисунок 5.9 – Про нас

5.4 Висновки за розділом 5

В даному розділі було описано розроблені функції системи, наглядно продемонстровано функціонування вебзастосунку із використанням інтерфейсу. Також показано результат перетворення даних.

ВИСНОВКИ

Під час виконання випускної кваліфікаційної роботи магістра було розроблено вебзастосунок для конвертування аудіоформатів.

В ході виконання магістерської роботи було проаналізовано предметну область, розглянуто можливості бібліотеки FFmpeg для роботи з форматами файлів, проведено аналіз існуючих програм, що виконують аналогічну роботу, за допомогою мови програмування JavaScript створено застосунок конвертування аудіофайлів.

Вебзастосунок має наступні функції:

- обрання необхідного формату для конвертування аудіофайлу;
- збереження аудіофайлу у певний формат;
- переформатування файлу у інший обраний формат користувачем;
- обрання типу відтворення;
- обрання частоти звуку;
- обрання бітрейту;
- перегляд результату конвертування у відсотках;
- завантаження до локального сховища даних.

Клієнтська частина вебзастосунку була розроблена за допомогою мови гіпертекстової розмітки HTML, каскадних таблиць стилів CSS, мовою JavaScript для набуття інтерактивності, бібліотекою FFmpeg.

Середовищем функціонування можуть бути браузері: Mozilla, Opera, Google Chrome, Microsoft Edge.

Переваги розробленого продукту наступні:

- не потребує додаткових бібліотек;
- кросплатформеність, надійність;
- стильний, сучасний, зручний інтерфейс у використанні який не потребує багато часу на опанування принципів роботи з ним.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Ronald, C. Smart Bitrate Control [Текст] / C. Ronald. – Wasington: VSD, 2012. – 395 с.
2. Формат WAV [Електрон. ресурс]. – Режим доступу: <https://ua.phhsnews.com/articles/howto/what-are-wav-and-wave-files-and-how-do-i-open-them.html>.
3. Madhavi, G. FLAC and Numerical Modeling in Geomechanics [Текст] / G. Madhavi – New-York: O'Reilly Media, 1999. – 528 с.
4. Gertler, N. Complete Idiot's Guide to MP3: Music on the Internet [Текст] / N. Gertler. – New-York: Apress, Inc, 2000. – 294 с.
5. Формат AAC [Електрон. ресурс]. – Режим доступу: <https://quauty.pro/uk/blog-cat-ua/shho-take-kodek-sbc-aac-ta-ldac/>.
6. Формат WMA [Електрон. ресурс]. – Режим доступу: <https://uk.begin-it.com/34-what-is-a-wma-file-and-how-do-i-open-one>.
7. Salomon, D. A Guide to Data Compression Methods [Текст] / D. Salomon. – New-York: O'Reilly Media, 2002. – 307 с.
8. Figma [Електрон. ресурс]. – Режим доступу: <https://aboutmarketing.info/internet-marketynh/instrumenty/shcho-take-figma-ta-yak-yiyi-vykorystovuvaty-digital-fakhivtsy/>.
9. Robbins, J. Learning Web Design: A Beginner's Guide to HTML, CSS, JavaScript, and Web Graphics 5th Edition [Текст] / J. Robbins. – Boston: O'Reilly Media, 2018. – 808 с.
10. CSS [Електрон. ресурс]. – Режим доступу: https://css.in.ua/article/shcho-take-html_10.
11. JavaScript [Електрон. ресурс]. – Режим доступу: <https://astwellsoft.com/uk/blog/tehnology/javascript.html>.
12. Робсон, Е. Head First. Програмування на JavaScript [Текст] / Е. Робсон, Е. Фрімен. – Харків: Фабула, 2022. – 672 с.
13. Фрімен, А. JQuery для професіоналів [Текст] / А. Фрімен. – Харків:

«Книжкова Лавка», 2022. – 672 с.

14. FFmpeg [Electronic resource]. – Access mode: <https://ffmpeg.org/>.
15. Shashank, J. WebAssembly for Cloud: A Basic Guide for Wasm-Based Cloud Apps [Текст] / J. Shashank. – New-York: Apress, Inc, 2021. – 180 с.

ДОДАТОК А
Технічне завдання

A.1 Вступ

A.1.1 Назва розроблюваного програмного забезпечення

Назва розроблюваного програмного забезпечення «FileLess».

A.1.2 Призначення та область застосування

Вебзастосунок призначений для конвертування аудіоформатів, який може надати користувачеві бажаний йому результат за встановленими параметрами у мережі Інтернет.

A.2 Підстави для розробки

Підставою для розробки є завдання на дипломну кваліфікаційну роботу магістра на тему: «Дослідження та програмна реалізація методів конвертування аудіоформатів», затверджену наказом закладу вищої освіти від 4 листопада 2022 року №363.

A.3 Призначення розробки

Розробка призначена для заощадження обсягу пам'яті на носіях інформації.

A.4 Вимоги до програмного виробу

A.4.1 Вимоги до функціональних характеристик

Алгоритм має виконувати такі функції:

- обрання необхідного формату для конвертування аудіофайлу;
- збереження аудіофайлу у певний формат;
- переформатування файлу у іншій обраний формат користувачем;

- обрання типу відтворення;
- обрання частоти звуку;
- обрання бітртейту;
- перегляд результату конвертування у відсотках;
- завантаження до локального сховища даних.

Інтерфейс повинен бути зручним у використанні, який не потребує багато часу на опанування принципів роботи з ним.

A.4.2 Вимоги до надійності

У випадку некоректної роботи вебзастосунку після перезавантаження сторінки система повинна відновити до первинних функцій застосунку.

A.4.3 Умови експлуатації

Для отримання доступу до вебзастосунку треба мати стабільне підключення до мережі Інтернет.

A.4.4 Вимоги до складу та параметрів технічних засобів

Для повноцінного використання вебзастосунку необхідно мати комп'ютер з наступними мінімальними вимогами:

- частота процесора 2.0 ГГц;
- оперативна пам'ять не менше 1 Гб;
- браузер Mozilla, Opera, Google Chrome, Microsoft Edge.

A.4.5 Вимоги до зберігання

Доступ до вебзастосунку надається через сервер, доступ до якого має адміністратор.

ДОДАТОК Б
Текст програми

```

jQuery(document).ready(function($){

// menu fixed
var header = $('#header');
$(window).scroll(function() {
    var scroll = $(window).scrollTop();

    if (scroll >= 20) {
        header.removeClass('clearHeader').addClass('darkHeader');
    } else {
        header.removeClass('darkHeader').addClass('clearHeader');
    }
});

// 1
// Обрати
// перетягування
var drop = document.getElementById('drop__you__file');
drop.addEventListener('dragover', handleDragOver);
drop.addEventListener('drop', handleDropSelect);

// Назва файлу при обранні
function handleFileSelect(event) {
    var files = event.target.files; // FileList object

    // files is a FileList of File objects. display first file name
    file = files[0];
    if (file) {
        $('#file_name').text("Назва файлу: " + file.name);
        readInputFile(file);
    }
}

// Назва файлу при перетягуванні

```

```

function handleDropSelect(event) {
    event.stopPropagation();
    event.preventDefault();

    var files = event.dataTransfer.files; // FileList object

    // files is a FileList of File objects. display first file name
    file = files[0];
    if (file) {
        $("#file_name").text("Назва файлу: " + file.name);
        readInputFile(file);
    }
}

// 2
// Формат
//отримання значення
function getOutFormat() {
    return $('input[name=format]:checked').val();
}

// 3
// Бітрейт
function getBitrate() {
    return $('input[name=bitr]:checked').val();
}

// input file params
var Name;
var buffer;

function timeToSeconds(time) {
    var parts = time.split(":");
    return parseFloat(parts[0]) * 60 * 60 + parseFloat(parts[1]) * 60 + parseFloat(parts[2]) +
    parseFloat("0." + parts[3]);
}

```

```

// create ffmpeg worker
var ffmpegWorker = getFFMPEGWorker();
var ffmpegRunning = false;

$('#convert').click(function() {
    // terminate existing worker
    if (ffmpegRunning) {
        ffmpegWorker.terminate();
        ffmpegWorker = getFFMPEGWorker();
    }
    ffmpegRunning = true;

    // display converting animation
    $('#converting').show();
    $('#error').hide();

    // hide download div
    $('#download').hide();

    // change download file name
    var NameExt = Name.substr(Name.lastIndexOf('.') + 1);
    var outName = Name.substr(0, Name.lastIndexOf('.')) + "." + getOutFormat();
    $('#downloadLink').attr("download", outName);
    $('#downloadLink').text(outName);

    var arg = [];
    arg.push('-i');
    arg.push(Name);
    switch (getOutFormat()) {
        case "mp3":
            arg.push("-acodec");
            arg.push("liblame");
            arg.push("out.mp3");
            break;
        case "ogg":

```

```

        arg.push("-acodec");
        arg.push("libvorbis");
        arg.push("out.ogg");
        break;
    case "aac":
        arg.push("-acodec");
        arg.push("libfdk_aac");
        arg.push("out.mp4");
        break;
    case "wma":
        arg.push("-acodec");
        arg.push("wmav1");
        arg.push("out.asf");
        break;
    case "flac":
        arg.push("out.flac");
        break;
}

ffmpegWorker.postMessage({
    type: "command",
    arg: arg,
    files: [
        {
            "name": Name,
            "buffer": buffer
        }
    ]
});
});

// disable conversion at start
$('#convert').attr('disabled', 'true');
```

```

function readInputFile(file) {
    // disable conversion for the time of file loading
    $('#convert').attr('disabled', 'true');

    // load file content
    var rDownload = new FileReader();
    rDownload.onload = function(e) {
        $('#convert').removeAttr('disabled');
        Name = file.name;
        buffer = e.target.result;
    }
    rDownload.readAsArrayBuffer(file);
}

// // reset file selector at start
// function resetInputFile() {
//     $('#inFile').wrap('<form>').closest('form').get(0).reset();
//     $('#inFile').unwrap();
// }
// resetInputFile();

// setup input file listeners
document.getElementById('upload__fi').addEventListener('change', handleFileSelect, false);
$('#upload__fi').on('change', function(){
    console.log(this.files[0].size / 1024 / 1024 + 'Mb');
});

function handleDragOver(event) {
    event.stopPropagation();
    event.preventDefault();
    event.dataTransfer.dropEffect = 'copy'; // explicitly show this is a copy
}

```

```

// create ffmpeg worker
function getFFMPEGWorker() {
    // regexps for extracting time from ffmpeg logs
    var durationReg = /Duration: (.?)/, /
    var timeReg = /time=(.?) /;
    var duration;

    var ffmpegWorker = new Worker('./assets/js/worker.js');

    ffmpegWorker.addEventListener('message', function(event) {
        var message = event.data;
        if (message.type === "ready" && window.File && window.FileList &&
window.FileReader) {
            // script loaded, hide loader
            $('#loading').hide();
        } else if (message.type == "stdout") {
            console.log(message.data);
        } else if (message.type == "stderr") {
            console.log(message.data);

            // try to extract duration
            if (durationReg.exec(message.data)) {
                duration = timeToSeconds(durationReg.exec(message.data)[1]);
            }
            // try to extract time
            if (timeReg.exec(message.data)) {
                var time = timeToSeconds(timeReg.exec(message.data)[1]);
                if (duration) {
                    $('#progress').text("Progress: " + Math.floor(time / duration * 100) + "%");
                    $('#progress').show();
                }
            }
        } else if (message.type == "done") {
            var code = message.data.code;
            var outN = Object.keys(message.data.outputFiles);

```

```

    if (code == 0 && outN.length) {
        var outName = outN[0];
        var outbuffer = message.data.outputFiles[outName];
        var mbres = outbuffer.byteLength;
        var totalSizeMB = mbres / Math.pow(1024,2)
        var src = window.URL.createObjectURL(new Blob([outbuffer]));
        document.getElementById('mb').innerHTML = totalSizeMB.toFixed(2) + ' Mb';
        $("#downloadLink").attr('href', src);
        $("#download").show();
    } else {
        $("#error").show();
    }
    $("#converting").hide();
    $("#progress").hide();
}
}, false);
return ffmpegWorker;
}

```

```

$('.convert__format1 input[type=radio]').click(function(){
    let val = $(this).val();
    console.log(val);
});

```

```

$('.format__bitrate input[type=radio]').click(function(){
    let val = $(this).val();
    console.log(val);
});

```

```

})<!DOCTYPE html>

```

```

<html lang="en">

```

```

<head>

```

```

    <meta charset="UTF-8">

```

```

    <meta name="viewport" content="width=device-width, initial-scale=1.0">

```

```

    <link rel="stylesheet" href="/assets/css/style.css">

```

```

<link
                                                                    rel="stylesheet"
href="https://cdnjs.cloudflare.com/ajax/libs/fullPage.js/2.9.7/jquery.fullpage.min.css">
<title>Document</title>
</head>
<body>
  <!-- preloader -->
  <div class="preloader">
    <ul>
      <li>
        F
      </li>
      <li>
        i
      </li>
      <li>
        l
      </li>
      <li>
        e
      </li>
      <li>
        
      </li>
      <li>
        L
      </li>
      <li>
        e
      </li>
      <li>
        s
      </li>
      <li>
        s
      </li>
    </ul>
  </div>

```

```

        </ul>
    </div>
    <header id="header">
        <div class="container">
            <div class="general">
                <div class="div">
                    <div class="logo-right">
                        <p>File<img alt="logo">Less</p>
                        </div>
                    </div>
                <div class="menu">
                    <ul id="menu">
                        <li data-menuanchor="hero"><a href="#hero">Головна</a></li>
                        <li data-menuanchor="forbtn"><a href="#forbtn">Конвертування файлу</a></li>
                        <li data-menuanchor="aboutus"><a href="#aboutus">Про нас</a></li>
                    </ul>
                </div>
            </div>
        </div>
    </header>
    <!-- prev -->
    <main id="fullpage">
        <section id="hero" class="section" data-anchor="hero">
            <div class="container">
                <p class="text__main">Безкоштовний аудіоконвертер файлів</p>
                <p class="text__txt">Єдиний інструмент для перетворення аудіофайлів, який вам потрібен між усіма популярними форматами</p>
                <!-- button next-->
                <div class="arrow__next">
                    <div class="arrow__img">

```

```

        <a href="#forbtn"></a>

        </div>
        <div class="wave animated"></div>

    </div>

</div>

    </div>

    <!-- wave bottom -->
        <div class="wave__3">
<div class="wave__bottom"></div>
<div class="wave__bottom"></div>
<div class="wave__bottom"></div>
</div>
</section>
<!-- 2 section -->
<section id="forbtn" class="section" data-anchor="forbtn">
    <div class="container">
        <p class="maintxtconvert">Конвертуйте аудіофайли без будь-яких
допоміжних програм</p>
        <div class="color__bg">
            <!-- upload file -->
            <div class="upload">
                <p class="upload__txt">1. Завантажте файл</p>
                <div class="upload__bg">
                    <!-- button -->
                    <input type="file" class="upload__file" id="upload__fi" accept=".mp3">
                    <label for="upload__fi">Оберіть файл</label>
                    <p id="file_name">Назва файлу: <span>НЕВІРНИЙ
ФОРМАТ</span></p>
                    <p class="drop__file" id="drop__you__file">Перетягнути файл</p>
                </div>
            <!-- <p id="file_name"></p> -->
            </div>
            <!-- format convert -->
            <div class="convert">

```

<p class="convert__txt">2. Оберіть формат для конвертування</p>

<div class="convert__format">

<!-- 1 -->

<div class="convert__format1">

<p>Стиснення з втратами</p>

<div class="select">

<label for="mp3">MP3</label>

<input type="radio" name="format" id="mp3" value="mp3"

checked>

<label for="wma">WMA</label>

<input type="radio" name="format" id="wma" value="wma">

<label for="aac">AAC</label>

<input type="radio" name="format" id="aac" value="aac">

</div>

</div>

<!-- 2 -->

<div class="convert__format1">

<p>Стиснення без втрат</p>

<div class="select">

<label for="flac">FLAC</label>

<input type="radio" name="format" id="flac" value="flac">

</div>

</div>

<!-- 3 -->

<div class="convert__format1">

<p>Без втрат та стиснення</p>

<div class="select">

<label for="wav">WAV</label>

<input type="radio" name="format" id="wav" value="wav">

<label for="aiff">AIFF</label>

<input type="radio" name="format" id="aiff" value="aiff">

</div>

</div>

```

        </div>
    </div>
    <!-- settings -->
    <div class="settings">
        <p class="settings__txt">3. Налаштування</p>
    <!-- choose a bitrate -->
        <div class="bitrate__all">
    <div class="bitrate">
        <p class="bitrate__txt">Оберіть бітрейт</p>
        <div class="format__bitrate">
            <label for="k96">96k</label>
            <input name="bitr" type="radio" id="k96" value="96k">
            <label for="k128">128k</label>
            <input name="bitr" type="radio" id="k128" value="128k">
            <label for="k192">192k</label>
            <input name="bitr" type="radio" id="k192" value="192k">
            <label for="k320">320k</label>
            <input name="bitr" type="radio" id="k320" value="320k">
        </div>
    </div>
        <div class="bitrate">
            <p class="bitrate__txt">Оберіть канал</p>
            <div class="format__bitrate">
                <label for="one">Моно</label>
                <input name="ch" type="radio" id="one" value="1">
                <label for="two">Стерео</label>
                <input name="ch" type="radio" id="two" value="2">
            </div>
        </div>
    </div>
        <div class="bitrate">
            <p class="bitrate__txt">Оберіть частоту</p>
            <div class="format__bitrate">
                <label for="k12">12kHz</label>
                <input name="hertz" type="radio" id="k12" value="12000">

```

```

<label for="k20">20kHz</label>
<input name="hertz" type="radio" id="k20" value="20000">
<label for="k24">24kHz</label>
<input name="hertz" type="radio" id="k24" value="24000">
<label for="k48">48kHz</label>
<input name="hertz" type="radio" id="k48" value="48000">
</div>
</div>
</div>
<div class="convert__to__file">
  <p class="convert__toaudio__txt">4. Конвертування аудіофайлу</p>
  <div class="btn__flex">
    <button id="convert">Конвертувати</button>
    <!-- <button id="result__btn">Результат у відсотках</button> -->
     <b
id="progress" class="quick-hidden"></b>
    <a href="" id="download" download>Скачати</a>
  </div>
</div>
<!-- result of compaction in percentage ratio -->
<div class="result">
  <p class="result__txt">5. Результат стиснення у процентному
співвідношенні</p>
  <div class="result__ratio">
    <p class="result__interest">До <span id="before_result"></span>
Мб</p>
    <p class="result__interest">Після <span id="after_result">
</span>Мб</p>
    <p class="result__interest">Стиснено <span class="plus"> + </span>
<span class="minus">-</span><span id="percent_result"></span>%</p>
  </div>
</div>
</div>

```

```

        </div>
    </section>
    <!-- about us -->
    <section id="aboutus" class="section" data-anchor="aboutus">
        <div class="container">
            <p class="about">Вебзастосунок розробила студентка групи КНТ-211м
Галевська Ірина Дмитрівна</p>
        </div>
    </section>
    <footer class="section fp-auto-height">
        <p class="footer__txt">Розробка вебзастосунку 2022р.</p>
    </footer>
</main>

<script src="https://cdnjs.cloudflare.com/ajax/libs/jquery/3.6.1/jquery.min.js">
</script>
<script src="https://cdnjs.cloudflare.com/ajax/libs/fullPage.js/2.9.7/jquery.fullpage.min.js" >
</script>
<script src="assets/js/main.js" >

</script>
</body>
</html>

/* http://meyerweb.com/eric/tools/css/reset/
   v2.0 | 20110126
   License: none (public domain)
*/

```

html, body, div, span, applet, object, iframe,
 h1, h2, h3, h4, h5, h6, p, blockquote, pre,
 a, abbr, acronym, address, big, cite, code,
 del, dfn, em, img, ins, kbd, q, s, samp,
 small, strike, strong, sub, sup, tt, var,
 b, u, i, center,
 dl, dt, dd, ol, ul, li,
 fieldset, form, label, legend,

```
table, caption, tbody, tfoot, thead, tr, th, td,
article, aside, canvas, details, embed,
figure, figcaption, footer, header, hgroup,
menu, nav, output, ruby, section, summary,
time, mark, audio, video {
```

```
    margin: 0;
    padding: 0;
    border: 0;
    font-size: 100%;
    font: inherit;
    vertical-align: baseline;
}
```

```
html {
    scroll-behavior: smooth;
}
```

```
/* HTML5 display-role reset for older browsers */
```

```
article, aside, details, figcaption, figure,
footer, header, hgroup, menu, nav, section {
    display: block;
}
```

```
body {
    line-height: 1;
    overflow: hidden;
}
```

```
ol, ul {
    list-style: none;
}
```

```
blockquote, q {
    quotes: none;
}
```

```
blockquote:before, blockquote:after,
q:before, q:after {
    content: "";
    content: none;
}
```

```

}
table {
  border-collapse: collapse;
  border-spacing: 0;
}
@font-face {
  font-family: "MontserratAlt1-Medium";
  src: local("MontserratAlt1-Medium"),
  url("../fonts/MontserratAlt1-Medium.ttf") format("truetype")
}
/*@import url('https://fonts.googleapis.com/css2?family=Oswald&display=swap');*/
@import url('https://fonts.googleapis.com/css2?family=Brygada+1918&display=swap');
.container{
  max-width: 1150px;
  position: relative;
  width: 100%;
  padding: 0 10px;
  margin: 0 auto;
  display: block;
font-family: 'Brygada 1918', serif;
}
section{
  height: calc(100vh - 135px);
  padding-top: 135px;
  position: relative;
  overflow: hidden;
  background-blend-mode: multiply;
  background: url('../img/f2.jpg'), linear-gradient(180deg, rgba(59,50,50), rgba(59,46,179,1) 100%);
}
header{
  position: fixed;
  width: 100%;
  top: 0;
  z-index: 99999;
}

```

```

.general{
  display: flex;
  justify-content: space-between;
  align-items: center;
  /*padding-top: 30px;*/
  padding: 30px 0;
}

.logo-right{
  display: flex;
}

.logo-right p{
  font-family: "MontserratAlt1-Medium";
  font-size: 35px;
  background:linear-gradient(90deg, rgba(255, 0, 153, 1), rgba(0, 228, 173, 1) 100%);
  -webkit-background-clip: text;
  -webkit-text-fill-color: transparent;
  font-weight: 500;
  text-shadow: 0 0 10px rgba(216, 206, 255, 0.1), 0 0 20px rgba(216, 206, 255, 0.1),0 0 30px
  rgba(216, 206, 255, 0.1),0 0 40px rgba(216, 206, 255, 0.1);
}

.logo-right img{
  height: 28px;
  width: 27px;
  /* -webkit-box-shadow: 0px 0px 17px 0px rgba(255, 0, 153, 0.5);
  -moz-box-shadow: 0px 0px 17px 0px rgba(255, 0, 153, 0.5);
  box-shadow: 0px 0px 17px 0px rgba(255, 0, 153, 0.5);*/
}

/*menu*/
.menu ul{
  display: flex;
}

.menu a{
  text-decoration: none;
  color: #fff;
  font-size: 21px;

```

```

}
.menu li{
    margin-right: 20px;
    cursor: pointer;
    position: relative;
    text-decoration: none;
}
.menu li::before {
    content: "";
    position: absolute;
    left: 0; bottom: -5px;
    width: 0;
    height: 1.5px;
    background-color: #fff;
    transition: .5s;
}
.menu li:hover:before {
    width: 100%;
}
/*prev*/
.text__main{
    color: #fff;
    font-size: 45px;
    text-align: center;
    padding-top: 100px;
    opacity: 85%;
    text-shadow: 0 0 10px rgba(255, 0, 153, 1),0 0 20px rgba(255, 0, 153, 1),0 0 30px rgba(255, 0,
153, 1),0 0 40px rgba(0, 228, 173, 1);

}
.text__txt{
    color: #fff;
    font-size: 35px;
    text-align: center;
    padding-top: 55px;

```

```

    opacity: 0.7;
    text-shadow: 0 0 10px rgba(0, 228, 173, 1),0 0 20px rgba(255, 0, 153, 1),0 0 30px rgba(0, 228,
173, 1),0 0 40px rgba(0, 228, 173, 1);
}
/*button next*/
.arrow__next{
    width: 100px;
    height: 100px;
    margin: 0 auto;
    position: absolute;
    left: 0;
    cursor: pointer;
    right: 0;
    bottom: -152px;
    transition: 0.5s;
}
.arrow__next:hover {
    -webkit-transform: scale(1.2);
    -ms-transform: scale(1.2);
    transform: scale(1.2);
    transition: 0.5s;
}
.wave{
    width: 100px;
    height: 100px;
    background: rgb(59,46,179,1);
    opacity: 0.5;
    border-radius: 50%;
    animation-name: anim;
    position: absolute;
}
.arrow__img{
    width: 60px;
    height: 60px;
    background:rgb(59,50,50);

```

```
border-radius: 50%;
position: absolute;
top: 20px;
left: 20px;
z-index: 1;
display: table;
}
```

```
@keyframes myanim {
  0% {
    opacity: 0;
    -webkit-transform: scale3d(.4, .4, .4);
    transform: scale3d(.4, .4, .4);
  }
  50% {
    opacity: 0.5;
  }

  100% {
    opacity: 0;
  }
}

.animated {
  animation: myanim 2s infinite;
}

.arrow__img{
  display: flex;
  justify-content: center;
  align-items: center;
}

/*vawe bottom*/
.wave__3{
  width: 100%;
}

.wave__bottom {
```

```

background: rgb(255 255 255 / 25%);
border-radius: 1000% 1000% 0 0;
position: absolute;
width: 200%;
height: 12em;
animation: wave 10s -3s linear infinite;
transform: translate3d(0, 0, 0);
opacity: 0.8;
bottom: 0;
left: 0;
}
.wave__bottom:nth-of-type(2) {
  bottom: -1.25em;
  animation: wave 18s linear reverse infinite;
  opacity: 0.8;
}
.wave__bottom:nth-of-type(3) {
  bottom: -2.5em;
  animation: wave 20s -1s reverse infinite;
  opacity: 0.9;
}
@keyframes wave {
  2% {
    transform: translateX(1);
  }
  25% {
    transform: translateX(-25%);
  }
  50% {
    transform: translateX(-50%);
  }
  75% {
    transform: translateX(-25%);
  }
  100% {

```

```

        transform: translateX(1);
    }
}
/*2 section*/
.maintxtconvert{
    color: #fff;
    font-size: 35px;
    text-align: center;
    opacity: 85%;
}
/*upload file*/
.upload{
    padding-top: 35px;
}
.upload__txt{
    color: #fff;
    font-size: 27px;
    text-align: center;
}
.upload__bg{
    background-color: rgba(217, 217, 217, 0.3);
    max-width: fit-content;
    border-radius: 5px;
    display: block;
    margin: 0 auto;
    margin-top: 10px;
    padding: 35px 50px 35px 50px;
    width: 100%;
    justify-content: space-between;
    display: flex;
    max-width: 750px;
}
/*button*/
.upload__file{
    display: none;

```

```

}
.upload__bg label{
  background-color: rgb(58, 49, 57);
  color: #fff;
  padding: 17px;
  cursor: pointer;
  border-radius: 3px;
  font-size: 20px;
  transition: 0.5s;
  display: block;
}
.upload__bg label:hover{
  transition: 0.5s;
  background-color: rgba(59, 49, 172, 0.7);
  box-shadow: 0 0 10px rgb(58, 49, 57),0 0 20px rgb(58, 49, 57),0 0 30px rgb(58, 49, 57),0 0 40px
rgb(58, 49, 57);
}
#file_name{
  color: #fff;
  /* width: 40px;*/
  box-sizing: border-box;
  /*padding: 70px;*/

}
.drop__file{
  background-color: rgba(58, 49, 69, 0.3);
  color: #fff;
  padding: 17px;
  border-radius: 3px;
  font-size: 20px;
  transition: 0.5s;
  display: block;
  border: 3px dotted rgb(58, 49, 57);
}
.drop__file:hover{

```

```

    background-color: rgba(59, 49, 172, 0.3);
}
/*format convert*/
.convert{
    padding-top: 35px;
}
.convert__txt{
    color: #fff;
    font-size: 27px;
    text-align: center;
}
.convert__format{
    justify-content: space-between;
    display: flex;
}
.convert__format p{
    color: #fff;
    font-size: 20px;
}
.convert__format label{
    background-color:rgb(58, 49, 57);
    color: #fff;
    display: block;
    max-width: fit-content;
    padding: 10px;
    text-decoration: none;
    cursor: pointer;
    transition: 0.5s;
    margin-right: 10px;
}
.convert__format input{
    display: none;
}
.convert__format label:hover{
    background-color: #fff;

```

```

color: rgb(58, 49, 57);
transition: 0.5s;
box-shadow: 0 0 10px rgba(255, 0, 153, 0.5),0 0 20px rgba(255, 0, 153, 0.5),0 0 30px rgba(255, 0,
153, 0.5),0 0 40px rgba(255, 0, 153, 0.5);
}
.convert__format .active{
  box-shadow: 0 0 10px rgba(0, 228, 173, 0.5),0 0 20px rgba(0, 228, 173, 0.5),0 0 30px rgba(0, 228,
173, 0.5),0 0 40px rgba(0, 228, 173, 0.5);
  background-color: rgba(59, 49, 172, 0.7);
}
.convert__format1{
  width: 100%;
  background-color: rgba(217, 217, 217, 0.3);
  max-width: fit-content;
  border-radius: 5px;
  display: block;
  margin: 0 auto;
  margin-top: 10px;
  padding: 16px 40px 16px 40px;
  width: 100%;
  justify-content: space-between;
  max-width: 203px;
}
.select{
  display: flex;
  padding-top: 10px;
}
/*settings*/
.settings{
  padding-top: 35px;
}
.settings__txt{
  color: #fff;
  font-size: 27px;
  text-align: center;

```

```

}
.bitrate__all{
  justify-content: space-between;
  display: flex;
  margin-top: 15px;
}
/*choose a bitrate*/
.bitrate__txt{
  color: #fff;
  font-size: 20px;
  text-align: center;
}
.format__bitrate label{
  background-color:rgb(58, 49, 57);
  color: #fff;
  display: block;
  max-width: fit-content;
  padding: 10px;
  text-decoration: none;
  cursor: pointer;
  transition: 0.5s;
  margin-right: 10px;
}

/* Define animation */
@keyframes grow {
  0%,
  40%,
  100% {
    transform: scale(0.7);
  }
  20% {
    transform: scale(1);
  }
}

```

```
/* Define li properties */
.preloader ul li:nth-child(1) {
  animation-delay: -1.4s;
}
.preloader ul li:nth-child(2) {
  animation-delay: -1.2s;
}
.preloader ul li:nth-child(3) {
  animation-delay: -1s;
}
.preloader ul li:nth-child(4) {
  animation-delay: -0.8s;
}
.preloader ul li:nth-child(5) {
  animation-delay: -0.6s;
}
.preloader ul li:nth-child(6) {
  animation-delay: -0.4s;
}
.preloader ul li:nth-child(7) {
  animation-delay: -0.2s;
}
#file_name span{
  display: none;
}
#file_name .error{
  color: rgb(96, 10, 10);
  font-size: 20px;
  display: block;
  font-weight: 700;
  text-align: center;
}
```

ДОДАТОК В
Слайди презентації

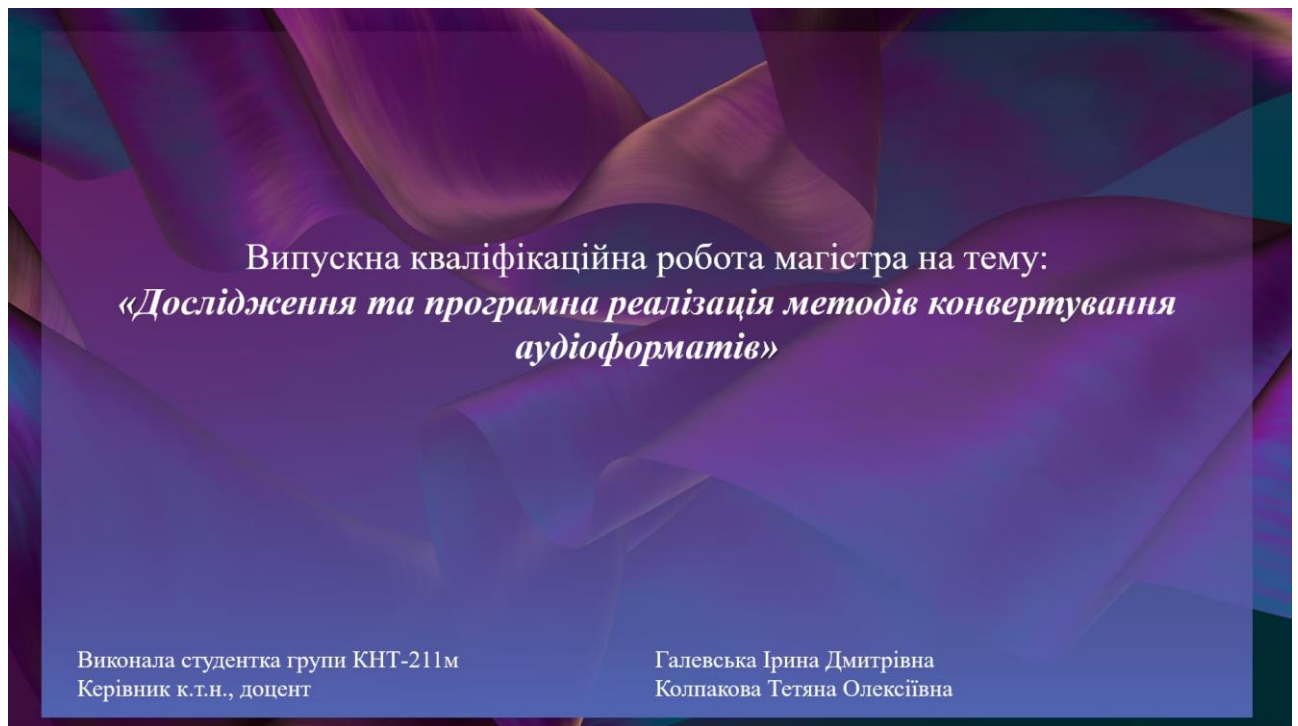


Рисунок В.1 – Тема роботи

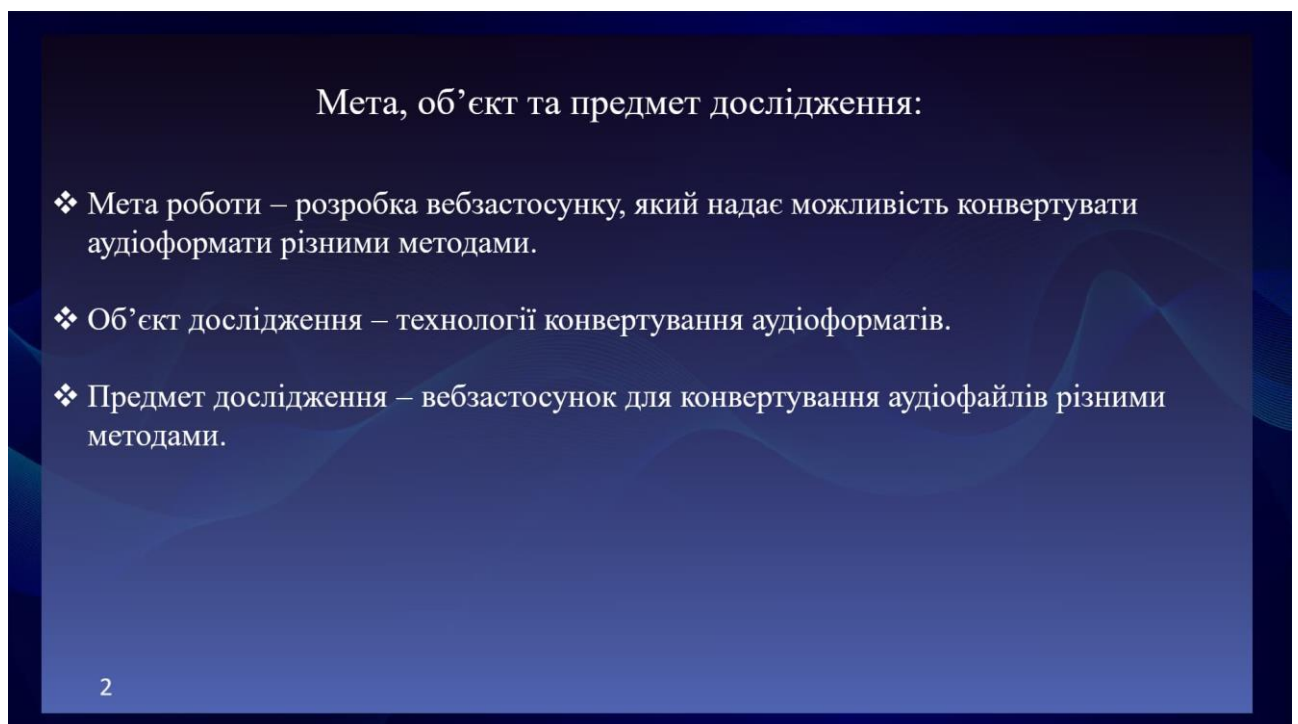


Рисунок В.2 – Мета об'єкт та предмет дослідження

Аналіз предметної області

Аудіофайли поділяють на три великі групи. Відповідно, з цими групами пов'язані і методи конвертування з одного формату в інший.

Нестиснені формати

Формат	Розширення	Переваги	Недоліки	Автор
WAVE (WAV)	.wav	Підтримка великої кількості програм	Великий розмір файлу	1991 рік, Microsoft та IBM
AIFF	.aiff, .aif	Підтримка метаданих	Великий розмір файлу	1988 рік, Apple

Стиснення без втрат

Формат	Розширення	Переваги	Недоліки	Автор
FLAC	.flac	Ефективне стиснення, швидке декодування	Щоб оцінити добротність звуку потрібна якісна апаратура	2001 рік, Xiph.Org Foundation, Джим Коалсон
Monkey's Audio	.ape	Збереження оригінального джерела звуку	Обмежена підтримка платформ та ліцензії	2000 рік, Matthew T. Ashland
ALAC	.m4a	Відновлення пошкоджених даних	Менш сумісний з апаратним забезпеченням	2004 рік, Apple

Стиснення з втратами

Формат	Розширення	Переваги	Недоліки	Автор
MP3	.mp3	Найширше MDST (Modified Discrete cosine Transform) технології, що забезпечує хорошу якість при невеликому бітрейті, невеликий розмір	Обмеженість поширення	1993 рік, група інституту Франкофера MPEG (Motion Pictures Expert Group) на чолі з Карл-Гардом Брандсбургеном
AAC	.aac, .m4a, .m4p	До 48 звукових каналів, видалення з'єднання	Не дуже поширений за межами Apple, залежить від конкретного пристрою	1997 рік, Bell Labs, Інститут інтегральних схем товариства Франкофера, Dolby Laboratories, Sony, Nokia, Coding Technologies[4]
WMA	.wma	Повна підтримка з боку Windows	Низька якість при низькому бітрейті	2002 рік, Microsoft
OOG	.ogg	Звукові дані, кодування і декодування не матимуть кування	Трудність інтеграції нових кодків, проблема синхронізації потоків	2004 рік, Xiph.Org Foundation

3

Рисунок В.3 – Аналіз предметної області

Порівняльна характеристика існуючих засобів

Audio Converter Online	Bear File Converter	AmoyShare
		
Переваги: ❖ добра оптимізація сайту.	Переваги: ❖ обрання мови інтерфейсу; ❖ зберігає аудіофайл впродовж двох годин після конвертації.	Переваги: ❖ онлайн підтримка; ❖ є блог-сторінка з новинами.
Недоліки: ❖ встановлений ліміт завантаження розміру; ❖ ліміт конвертування файлів на добу.	Недоліки: ❖ обмеження розміру файлу для завантаження; ❖ складний у користуванні; ❖ конвертування можливе тільки в один формат.	Недоліки: ❖ багато платних функцій, наприклад конвертування до найбільш популярних форматів; ❖ безкоштовне конвертування можливе тільки в один формат та не завжди вдається зберегти файл.

4

Рисунок В.4 – Порівняльна характеристика існуючих засобів



Рисунок В.5 – Схема методів конвертування

Методи перетворення даних

Основна відмінність методів стиснення з втратами та стиснення без втрат полягає в тому, що при стисненні з втратами створюється близьке співвідношення даних, а без втрат створюються точні вихідні дані, які представлені у порівняльній характеристиці.

Характеристики	Стиснення з втратами	Стиснення без втрат
Процес	Видаляє частину даних, що не можна виявити	Відтворює точні дані
Зменшення розміру	Значно	Незначно
Якість даних	Погіршується	Не погіршується
Використання	Зображення, аудіо та відео	Текст або програма, зображення та звук

6

Рисунок В.6 – Методи перетворення даних

Таблиця прикладів алгоритмів стиснення

Статистичні алгоритми		Словникові алгоритми	
Кодування Хаффмана	Арифметичне кодування	На основі алгоритму LZ77	На основі алгоритму LZ78
Статичне стиснення	Статичне стиснення	Класичні однорівневі	LZW-алгоритм
Динамічне стиснення	Динамічне стиснення	Двоступеневі	MW-алгоритм
-	-	LZHUF(LZ77-Huffman)	AP-алгоритм
-	-	LZARI(LZ77-арифмет)	Y-алгоритм

7

Рисунок В.7 – Таблиця прикладів алгоритмів стиснення

Алгоритм конвертування

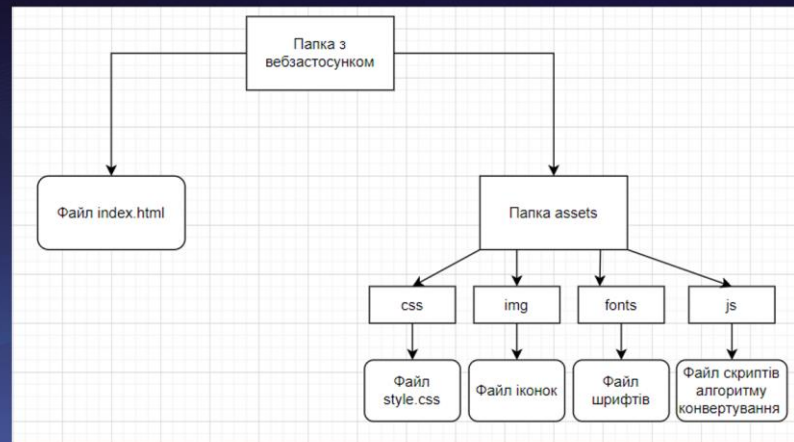
У форматі MP3 закладений алгоритм Хаффмана. Цей процес дозволяє поліпшити ступінь стиснення відносно однорідних сигналів. Цей вид кодування зазвичай дозволяє стиснути аудіодані 10:1 або вище.



8

Рисунок В.8 – Алгоритм конвертування

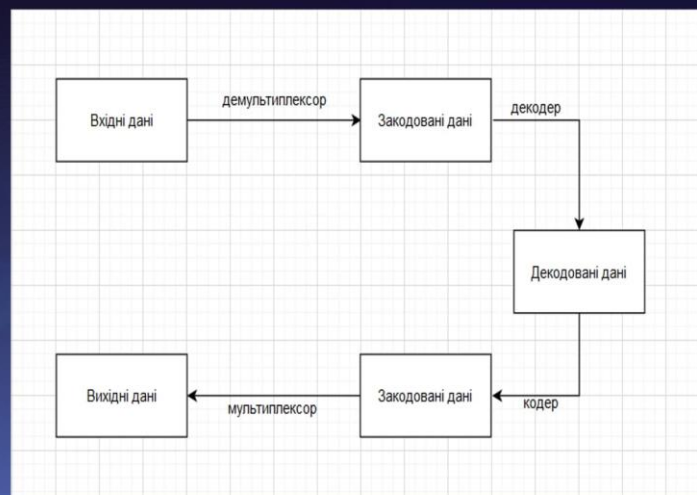
Схема структури вебзастосунку



9

Рисунок В.9 – Схема структури вебзастосунку

Функціонування бібліотеки FFmpeg

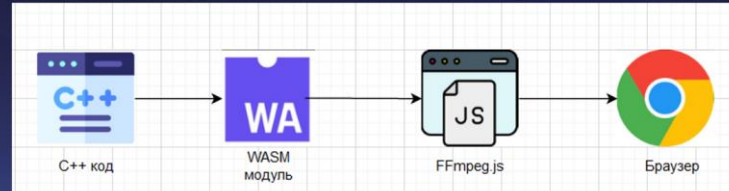


10

Рисунок В.10 – Функціонування бібліотеки FFmpeg

Схема процесу компіляції бібліотеки

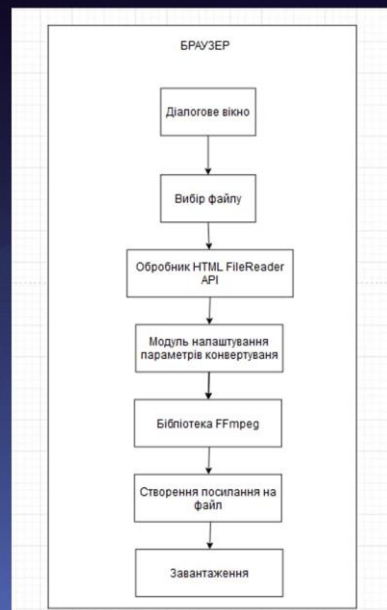
Для зборки бібліотеки *FFmpeg* був використаний компілятор *emscripten*. Це інструмент, створений для компілювання програм, які використовують стандартні бібліотеки C і C++.



11

Рисунок В.11 – Схема процесу компіляції бібліотеки

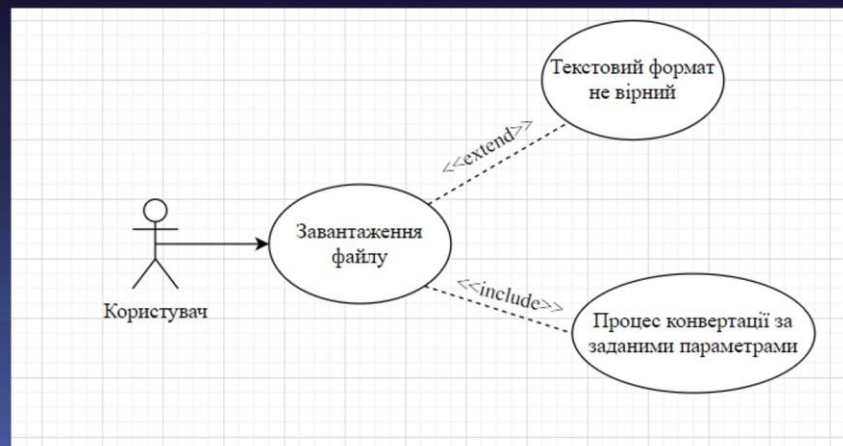
Алгоритм додавання файлу до системи та його обробка



12

Рисунок В.12 – Алгоритм додавання файлу

Алгоритм обробки помилки, в основі якого лежить діаграма прецедентів



13

Рисунок В.13 – Діаграма прецедентів

Функціонування вебзастосунку

The screenshot shows the FileLess web application interface. The header includes the logo 'FileLess' and navigation links: 'Головна', 'Конвертування файлу', and 'Про нас'. The main heading is 'Конвертуйте аудіофайли без будь-яких допоміжних програм'. The interface is divided into five numbered steps:

- 1. Завантажте файл**: Includes buttons 'Оберіть файл' and 'Перезавантажити файл'. Below, it shows the selected file: 'Назва файлу: KAUJH - Зорі (Alex Seregin Remix).mp3'.
- 2. Оберіть формат для конвертування**: Offers three categories: 'Стиснення з втратами' (MP3, WMA, AAC), 'Стиснення без втрат' (FLAC), and 'Без втрат та стиснення' (WAV, AIFF). The 'WMA' option is highlighted.
- 3. Налаштування**: Includes 'Оберіть бітрейт' (96k, 128k, 192k, 320k), 'Оберіть тип відтворення' (mono, stereo), and 'Оберіть частоту' (12kHz, 20kHz, 24kHz, 48kHz). The '192k' and '24kHz' options are highlighted.
- 4. Конвертування аудіофайлу**: Includes buttons 'Конвертувати' and 'Скачати'.
- 5. Результат стиснення у процентному співвідношенні**: Shows a comparison table:

До	Після	Стиснено
3.13Мб	4.9Мб	+17.7%

14

Рисунок В.14 – Функціонування вебзастосунку



Рисунок В.15 – Функціонування вебзастосунку

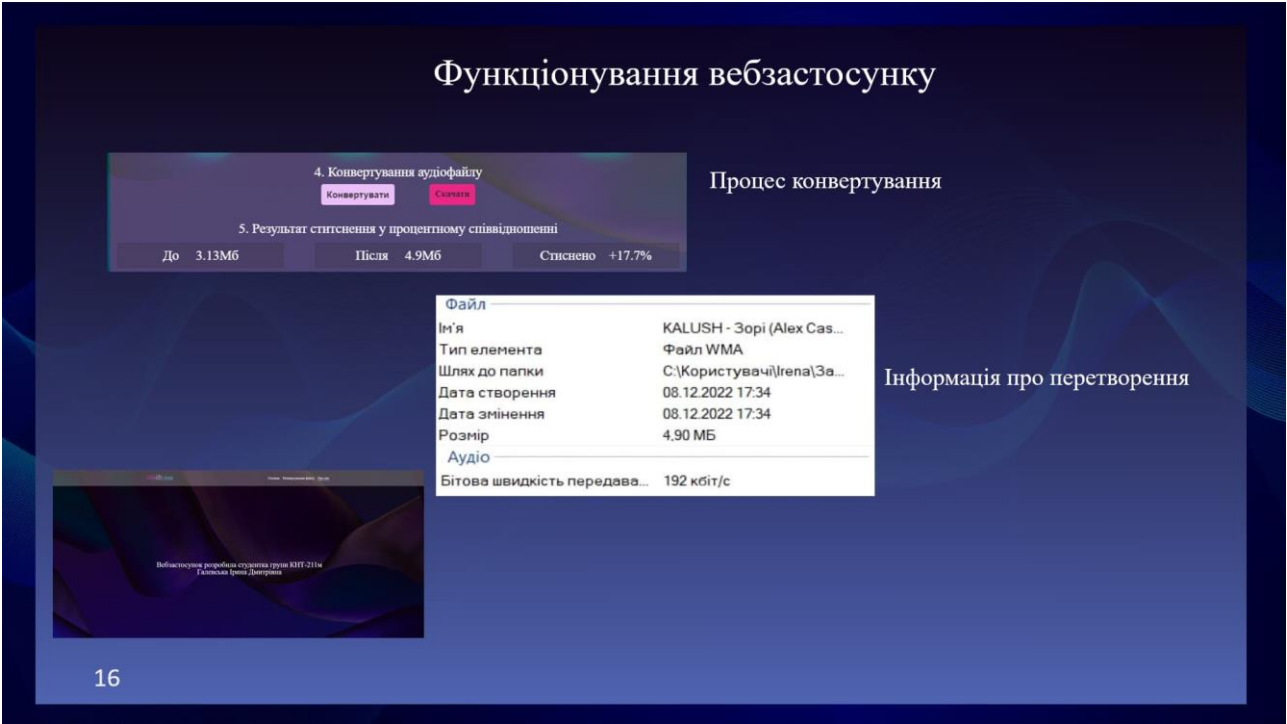


Рисунок В.16 – Функціонування вебзастосунку

Висновки

Під час виконання випускної кваліфікаційної роботи магістра було розроблено вебзастосунок для конвертування аудіоформатів.

В ході виконання магістерської роботи було проаналізовано предметну область, розглянуто можливості бібліотеки FFmpeg для роботи з форматами файлів, проведено аналіз існуючих програм, що виконують аналогічну роботу, за допомогою мови програмування JavaScript створено застосунок конвертування аудіофайлів.

Вебзастосунок має наступні функції:

- ❖ обрання необхідного формату для конвертування аудіофайлу;
- ❖ збереження аудіофайлу у певний формат;
- ❖ переформатування файлу у інший обраний формат користувачем;
- ❖ обрання типу відтворення;
- ❖ обрання частоти звуку;
- ❖ обрання бітрейту;
- ❖ перегляд результату конвертування у відсотках;
- ❖ завантаження до локального сховища даних.

Переваги розробленого продукту наступні:

- ❖ не потребує додаткових бібліотек;
- ❖ кросплатформеність, надійність;
- ❖ стильний, сучасний, зручний інтерфейс у використанні який не потребує багато часу на опанування принципів роботи з ним.

Рисунок В.17 – Висновки