Міністерство освіти і науки України
Луцький національний технічний університет
Факультет комп'ютерних та інформаційних технологій
Кафедра інженерії програмного забезпечення

КВАЛІФІКАЦІЙНА РОБОТА ЗА СТУПЕНЕМ ВИЩОЇ ОСВІТИ «БАКАЛАВР»

РЕДАКТОР АУДІОФАЙЛІВ НА БАЗІ HTML5 І JAVASCRIPT

AUDIO FILE EDITOR BASED ON HTML5 AND JAVASCRIPT

спеціальність 121 Інженерія програмного забезпечення
(шифр і назва спеціальності)

освітня програма «Інженерія програмного забезпечення»
(назва освітньої програми)

Виконав: здобувач вищої освіти
групи ІПЗ-41
Чухрай Станіслав Олександрович
_____
(підпис)
Керівник: д.т.н., професор
Матвіїв Юрій Ярославович
_____
(підпис)

Кваліфікаційну роботу
допущено до захисту
«8» червня 2023 р.
к.т.н., доцент
Гарант освітньої програми:
Ліщина Наталія Миколаївна
_____
(підпис)

Луцьк – 2023 року

ЛУЦЬКИЙ НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ

Факультет *комп'ютерних та інформаційних технологій*
Кафедра *інженерії програмного забезпечення*
Ступінь вищої освіти: *бакалавр*
Галузь знань: *12 Інформаційні технології*
Спеціальність: *121 Інженерія програмного забезпечення*
Освітня програма: *«Інженерія програмного забезпечення»*

ЗАТВЕРДЖУЮ
Завідувач кафедри
_____
«22» 02 2023 р.

ЗАВДАННЯ
НА КВАЛІФІКАЦІЙНУ РОБОТУ ЗДОБУВАЧУ ВИЩОЇ ОСВІТИ

Чухраю Станіславу Олександровичу
(прізвище, ім'я, по батькові)

1. Тема кваліфікаційної роботи: *Редактор аудіофайлів на базі HTML5 і JavaScript*

Керівник роботи: *д.т.н., професор Матвіїв Ю.Я.*
затверджені наказом закладу вищої освіти від «*23*» *грудня 2022 р. № 961-01-02*
2. Строк подання здобувачем вищої освіти кваліфікаційної роботи *«8» червня 2023 р.*
3. Вихідні дані до роботи: *технічне та програмне забезпечення ЕОМ, вимоги до розробки програмного забезпечення, ергономічні вимоги до функціонування програмного засобу, мова програмування JavaScript, HTML, CSS.*
4. Зміст розрахунково-пояснювальної записки (перелік питань, які потрібно розробити): *Аналіз сучасного стану проблеми, існуючих методів і засобів її розв'язання, аналіз і вибір засобів проектування, опис функціонального наповнення об'єкта проектування, розробка й обґрунтування системного наповнення, оцінка ергономічних та надійністних параметрів проектованої системи, функціонально-структурна схема роботи об'єкта проектування, розробка програмного продукту.*
5. Перелік графічного матеріалу: 11 рис.

6. Консультанти розділів роботи

Розділ	Прізвище, ініціали та посада консультанта	Підпис	
		завдання видав	завдання прийняв
Аналіз предметної області	*Матвіїв Ю.Я.*		
Специфікації вимог до розробленої системи	*Матвіїв Ю.Я.*		
Розробка об'єкта проєктування	*Матвіїв Ю.Я.*		
Нормоконтроль	*Матвіїв Ю.Я.*		
Гарант ОП	*Ліщина Н.М.*		
Показник запозичень тексту	2,84%		
Академічна доброчесність	*Яцук А.А.*		

7. Дата видачі завдання *«2» лютого 2023 р.*

КАЛЕНДАРНИЙ ПЛАН

№ з/п	Назва етапів кваліфікаційної роботи бакалавра	Строк виконання етапів роботи	Примітка
1	*Огляд літературних джерел по темі кваліфікаційної роботи бакалавра*	*05.02.2023 р.*	вик
2	*Аналіз проблеми розробки та впровадження об'єкту проектування*	*28.02.2023 р.*	вик
3	*Формування вимог до розроблюваного програмного продукту*	*12.03.2023 р.*	вик
4	*Обґрунтування вибору шляхів, технологій (алгоритмів) і засобів вирішення поставленого завдання*	*17.04.2023 р.*	вик
5	*Практична реалізація об'єкта проектування*	*18.05.2023 р.*	вик
6	*Тестування та налагодження об'єкта проектування*	*01.06.2023*	вик
7	*Здача чистового варіанту кваліфікаційної роботи бакалавра на кафедру*	*08.06.2023 р.*	вик

Здобувач вищої освіти _____ (підпис) *Чухрай С.О.* (прізвище, ініціали)

Керівник кваліфікаційної роботи _____ (підпис) *Матвіїв Ю.Я.* (прізвище, ініціали)

Міністерство освіти і науки України
Луцький національний технічний університет

Рецензія
на кваліфікаційну роботу

Здобувач вищої освіти: *Чухрай Станіслав Олександрович*

Тема: *«РЕДАКТОР АУДІОФАЙЛІВ НА БАЗІ HTML5 І JAVASCRIPT».*

Коротка характеристика кваліфікаційної роботи та прийнятих рішень: *Кваліфікаційна робота бакаоавра складається з трьох розділів, в яких проаналізовані проблематика та поставлені завдання за темою роботи, здійснено теоретичне дослідження та практичну реалізацію редактора аудіофайлів на базі веб-технологій, здійснено тестування розробленої системи*

Самостійні розробки і пропозицій автора: *Автор самостійно створив редактор аудіофайлів на базі веб-технологій.*

Практичне значення роботи *розроблений редактор аудіо на базі веб-технологій дозволяє здійснбвати редагування аудіофайлів у браузері комп'ютера.*

Недоліки: *Суттєвих недоліків в роботі не виявлено.*

Загальний висновок: *У кваліфікаційній роботі бакалавра наведені результати вирішення актуального завдання з розробки редактора аудіо на базі веб технологій. Було виконано всі поставлені завдання. Робота рекомендується до захисту екзаменаційній комісії та заслуговує позитивної оцінки.*

Рецензент: Ліщина Валерій Олександрович,
к.т.н., доцент, зав.кафедри комп'ютерних наук ЛНТУ.
(прізвище, ім'я, по-батькові, посада)

Рецензент кваліфікаційної роботи _____ (підпис)

«8» 06 2023 р.

АНОТАЦІЯ

Чухрай С.О. Редактор аудіофайлів на базі HTML5 і JavaScript. Рукопис.

Кваліфікаційна робота бакалавра ОП «Інженерія програмного забезпечення». Луцький національний технічний університет. Луцьк, 2023.

Кваліфікаційна робота бакалавра складається зі вступу, трьох розділів, висновків і пропозицій, списку використаних джерел та додатків.

У першому розділі здійснено аналіз сучасного стану проблеми розробки музичних редакторів на базі web-технологій і сформульовано завдання. В другому розділі визначено вимоги до розроблюваної системи, а також засоби, методи і алгоритми розробки. У третьому розділі розроблено редактор аудіо на базі HTML5 і здійснено його тестування. У висновках узагальнено інформацію, відображену у попередніх частинах. Результати розробки можуть бути застосовані в практичній діяльності для редагування аудіозаписів у веб-браузері.

Ключові слова: аудіофайл, mp3, wav, JavaScript, HTML5, редатор, ефекти.

ANNOTATION

Chukhray S.O. Audio file editor based on HTML5 and JavaScript. Manuscript.

Bachelor's qualifying thesis of the educational program "Software Engineering". Lutsk National Technical University. Lutsk, 2023.

The bachelor's qualifying thesis consists of an introduction, three sections, conclusions and proposals, a list of used sources and annexes.

In the first chapter, an analysis of the current state of the problem of developing music editors based on web technologies is carried out and the task is formulated. The second section defines the requirements for the developed system, as well as means, methods and development algorithms. In the third section, an audio editor based on HTML5 was developed and tested. The conclusions summarize the information presented in the previous sections. The results of the development can be applied in practice for editing audio recordings in a web browser.

Keywords: audio file, mp3, wav, JavaScript, HTML5, editor, effects.

ЗМІСТ

ВСТУП 6
РОЗДІЛ 1 АНАЛІЗ РЕДАКТОРІВ АУДІОФАЙЛІВ І ПОСТАНОВКА ЗАВДАНЬ НА КВАЛІФІКАЦІЙНУ РОБОТУ 8
1.1 Аналіз сучасного стану проблеми 8
1.2 Постановка завдання на кваліфікаційну роботу бакалавра 12
Висновки до розділу 1 13
РОЗДІЛ 2 СПЕЦИФІКАЦІЯ ВИМОГ ДО РОЗРОБЛЮВАНОЇ СИСТЕМИ 14
2.1 Аналіз, визначення вимог до розроблюваного програмного забезпечення та проектування програмного забезпечення 14
2.2 Вибір засобів, методів і алгоритмів вирішення поставленого завдання . 16
Висновки до розділу 2 26
РОЗДІЛ 3 РОЗРОБКА РЕДАКТОРА АУДІОФАЙЛІВ НА БАЗІ HTML5 27
3.1 Практична реалізація об’єкта проектування 27
3.2 Тестування та налагодження інформаційно-комп’ютерної системи 30
Висновки до розділу 3 33
ВИСНОВКИ 34
СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ 36
ДОДАТКИ 38

ВСТУП

Актуальність теми. У 1980-х роках у відеоігри вперше було включено 8-бітну музику і споживачі з захопленням сприйняли це [1]. З тих пір аудіо залишається невід'ємною частиною ігрового досвіду.

Аудіо набуло важливого значення і в інших технологіях. Наприклад, аудіосповіщення стали все частіше зустрічатися в сервісах обміну миттєвими повідомленнями, таких як Viber, Telegram, Facebook Messenger, Skype і багатьох інших. Ми тісно інтегрували аудіо з сучасним технологічним досвідом.

На сучасному етапі розвитку веб-технологій існує можливість використання аудіо не лише в десктопних додатках і відеоіграх, але і у веб.

Актуальною є проблема створення і редагування аудіоконтенту, для чого потрібне спеціальне програмне забезпечення.

Завдяки стрімкому розвитку процесорів зростає кількість програмного забезпечення, що спеціально розроблене для професійної обробки аудіоданих. Значний внесок у цей розвиток зробила корпорація Microsoft зі своїм програмним інтерфейсом DirectX, який спрощує роботу з графікою і звуком у реальному часі [2]. Ці два фактори здійснили невелику революцію в області обробки звуку на комп'ютерах.

Раніше відсутність необхідної елементної бази гальмувала впровадження цифрових методів обробки. На сучасному рівні розвитку обчислювальної техніки це вже не є проблемою. Цифрові методи, що реалізовані на сучасній елементній базі, знаходять усе більше застосування у різноманітних сферах обробки даних. Існує ряд десктопних програмних продуктів, що дозволяють здійснювати редагування і обробку аудіозаписів. З появою і розвитком HTML5 все більшої популярності набувають веб-додатки, які за своїми функціональними можливостями наближаються до десктопних, при цьому є кросплатформними і не потребують встановлення в операційній системі.

Зважаючи на вищесказане актуальною є розробка додатку на базі HTML5 для обробки і редагування аудіо безпосередньо у веб-браузері.

Метою роботи є розробка редактора аудіофайлів на базі HTML5 і JavaScript.

Реалізація мети вимагає вирішення наступних завдань:

– здійснити аналіз сучасного стану проблеми та існуючих аналогів;

– визначити вимоги до розроблюваного редактора аудіофайлів;

– вибрати технології, алгоритми і засоби, що дозволять здійснити розробку редактора аудіофайлів на базі HTML5 і JavaScript;

– здійснити розробку редактора аудіофайлів, що дозволить вносити зміни в існуючий аудіофайл безпосередньо в браузері, зберігати зміни;

– здійснити тестування і налагодження розробленого редактора аудіофайлів.

Об’єктом дослідження є методи і алгоритми обробки аудіофайлів.

Предметом дослідження є редактор аудіофайлій на базі веб-технологій.

Новизна роботи полягає у власній реалізації простого у використанні і інтуїтивно зрозумілого редактора аудіофайлів на базі технологій HTML5 і JavaScript.

Практичне значення розробки полягає в тому, що створене програмне забезпечення можна використовувати на будь-якому комп’ютері з сучасним браузером для редагування аудіофайлів без необхідності встановлення десктопних програмних продуктів.

РОЗДІЛ 1
АНАЛІЗ РЕДАКТОРІВ АУДІОФАЙЛІВ І ПОСТАНОВКА ЗАВДАНЬ НА КВАЛІФІКАЦІЙНУ РОБОТУ

1.1 Аналіз сучасного стану проблеми

Аудіоформат – це специфікація файлу, що використовується для зберігання цифрових аудіо даних на комп'ютері. Формат кодування звуку визначає, як аудіодані розміщуються у файлі і може бути нестисненим або стисненим для зменшення розміру файлу. Зазвичай застосовується стиснення без втрат. Дані можуть представлятися як необроблений бітовий потік в форматі кодування аудіо, але частіше за все вони включаються до контейнерного формату або формату аудіоданих з визначеним рівнем стиснення [3].

Аудіофайл – це комп'ютерний файл, що містить звукозапис і зберігає інформацію про амплітуду і частоту звуку, яку можна відтворити на комп'ютері або спеціальному пристрої для відтворення.

Існують такі типи програм для редагування аудіофайлів [4]:

– програми для запису та обробки цифрових аудіозаписів (наприклад, WaveLab, CoolEdit, Sound Forge, SAW Plus та ін.);

– секвенсори, тобто редактори синтезованої музики – MIDI (наприклад, програми MIDI Orchestrator Plus, MidiStudio, Cakewaik Pro, Cubase).

Наша розробка має відношення до першого типу програм.

Програми для обробки звуку надають можливість записувати музику, змінювати тембр, висоту, темп і застосовувати різноманітні ефекти. Після запису треку, з ним можна проводити такі процеси як мастеринг і зведення [5].

Мастеринг – це комплекс операцій, які спрямовані на покращення звучання фінального зведеного треку або його налаштування відповідно до певного стандарту. Основними процесами мастерингу є компресія і еквалізація.

Компресія – це процес зміни динаміки звуку з метою вирівнювання його гучності, роблячи гучні звуки більш тихішими.

Еквалізація – це процес обробки звукового сигналу з використанням еквалайзера. Як правило цей термін відносять до корекції амплітуди чи зміни співвідношення частот.

Зведення є етапом, на якому окремі записані треки об'єднуються в фінальний запис. У програмах обробки звуку можуть бути прості інструменти, що часто постачаються разом зі звуковими картами, а також програми, що спеціально розроблені для професійної роботи. В останніх версіях сучасних програм-секвенсорів MIDI (редакторів MIDI-файлів) є можливість запису і редагування треків, що наближає ці програми до багатоканальних звукорежисерських систем.

Розглянемо технології, що зробили можливим редагування аудіофайлів на комп’ютері.

Програмний інтерфейс DirectX, розроблений компанією Microsoft, відкрив нові можливості для створення професійних звукових програм [6]. Основна ідея полягала в тому, щоб подолати обмеження продуктивності Microsoft Windows при використанні комп’ютерної периферії. Щоб використовувати модулі ефектів, на комп’ютері необхідно встановити програмне забезпечення для обробки звуку, сумісне з DirectX. Якщо у вас є додатковий модуль ефектів, інтегрований через інтерфейс DirectX, ви зможете використовувати ці ефекти в будь-якій із перелічених програм. Ця технологія дозволяє насолоджуватися широким набором різноманітних звукових ефектів, не відмовляючись від улюблених програм для редагування звуку.

Формат VST – це рідний формат плагіна реального часу, розроблений Steinberg [7]. На сьогоднішній день існують сотні плагінів в цьому форматі, що робить його одним із найпопулярніших форматів для аудіододатків. Відмінності між плагінами DirectX і плагінами VST включають декілька параметрів, зокрема доступність на платформах ПК і Mac. Крім того, на відміну від попередніх версій DirectX, плагіни VST мають розширений інтерфейс автоматизації. Більшість сучасних програм MIDI також здатні працювати зі стандартним цифровим звуком. Ви також можете додати свій голос до треків

MIDI, зберегти результат у форматі WAV і підготувати пісню для запису на компакт-диск.

У наш час, коли навіть персональні комп'ютери мають високоякісні вбудовані пристрої введення звуку, веб-програмісти можуть почати включати аудіовхід користувача в свої програми. З прогресуючою роботою над API Web Audio [8] і MediaStream [9], прийшов час для веб-додатків використовувати всю потужність HTML5 і почати включати створені користувачами і відредаговані користувачем аудіодані виключно за допомогою нативних для браузера методів.

Підтримка аудіо офіційно з’явилася тільки в HTML5 [10] у 2014 році, хоча підтримка інших мультимедіа, таких як Java-аплети, існувала як у специфікації HTML 3.2 [11] 1997 року, так і в HTML 4.01 з 1999 року [12]. До HTML5 відтворення аудіо в Інтернеті було можливим лише за допомогою таких елементів як <object> та ActionScript (Flash).

У той же час десктопні програми для редагування аудіо, такі як Audacity, існували десятиліттями.

На відміну від настільних додатків, веб-додатки є кросплатформними, не вимагають інсталяції, мають можливість модульно інтегруватися з іншими додатками за допомогою API.

Оскільки веб-браузери починають збільшувати контроль користувача та введення таких аспектів, як аудіо та відео, існує потреба не просто в онлайн-редакторі, а в такому редакторі аудіо, що використовує лише рідні для браузера методи, оптимізовані для браузера.

На ринку існують потужні професійні програмні продукти для редагування аудіофайлів і обробки аудіоданих, які надають фахівцям широкі можливості для роботи з аудіоінформацією. Серед них можна виділити Adobe Audition, Wavelab і Sound Forge (рис. 1.1). Недоліками даних систем є недоступність широкому колу користувачів, оскільки вони є комерційними продуктами з великим набором функцій, орієнтованими на професіоналів. У той же час вони є десктопними рішеннями, що позбавлені переваг, які надають

сучасні веб-технології.

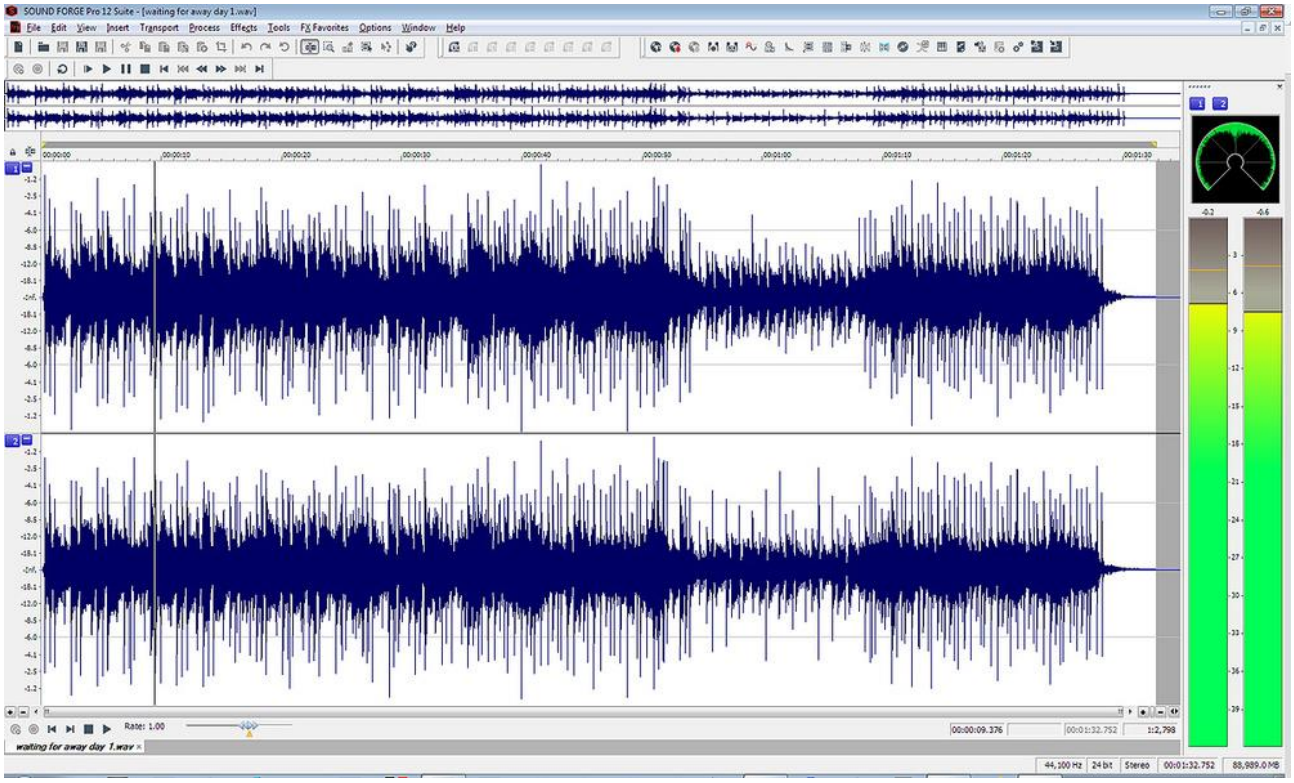

Рисунок 1.1 – Головне вікно програми Sound Forge 12 [13]

Типовий користувацький інтерфейс програми-редактора аудіофайлів складається з різних вікон, панелей та інструментів, які дозволяють користувачу здійснювати редагування, обробку і маніпуляції над звуковими файлами. Розглянемо кілька типових компонентів, які можна знайти у більшості програм-редакторів аудіофайлів:

– вікно редагування: головне вікно програми, де відображається хвильова форма аудіофайлу (осцилограма). Це вікно дозволяє відтворювати аудіо, вибирати і переміщати фрагменти файлу, виправляти дефекти, застосовувати ефекти та робити різноманітні зміни в звуковому матеріалі;

– панель інструментів: набір інструментів, доступних для використання під час редагування. Це можуть бути кнопки для відтворення, паузи, перемотування, видалення, копіювання та вставки фрагментів аудіо, а також інструменти для вимірювання гучності, виправлення нотації тощо;

– панель треків: відображає список треків аудіо, які можна редагувати

окремо. Кожен трек може містити окремий аудіофайл або частину аудіофайлу. Це дає змогу працювати з різними шарами звуку, наприклад, редагувати ритмічні частини, вокал, сольні інструменти тощо незалежно один від одного;

– панель ефектів: дозволяє додавати різні аудіо-ефекти до вибраних фрагментів або треків. Це можуть бути еквалайзери, компресори, реверберація, ехо, затримка, фільтри та багато інших. Панель ефектів зазвичай містить параметри настройки кожного ефекту.

В найпростішому випадку типова програма для редагування аудіофайлів повинна відкривати вихідний аудіофайл у хвильовій формі з часовою шкалою, забезпечуючи можливість виділення фрагмента аудіозапису, а також виконувати базові операції копіювання, вставлення, видалення фрагменту в різних положеннях часової шкали, обрізання аудіозапису до виділеного фрагмента. Після внесення змін повинна бути можливість зберігання зміненого аудіозапису у файл. Серед додаткових можливостей редактор аудіо дозволяє застосовувати спеціальні ефекти як до виділених фрагментів так і усієї аудіодоріжки.

Реалізація редактора аудіо з типовим інтерфейсом користувача, а також набором базових функцій і ефектів на базі HTML5 і JavaScript надасть можливість використовувати його широкому колу користувачів на будь-якому комп’ютері з сучасним веб браузером.

1.2 Постановка завдання на кваліфікаційну роботу бакалавра

В результаті аналізу сучасного стану проблеми було сформовано такі завдання на кваліфікаційну роботу:

– визначити вимоги до розроблюваного програмного засобу – редактора аудіофайлів на базі HTML5 і JavaScript;

– вибрати технології, алгоритми і засоби, що дозволять здійснити розробку редактора аудіофайлів на базі HTML5 і JavaScript;

– здійснити розробку редактора аудіофайлів, що дозволить вносити зміни в існуючий аудіофайл безпосередньо в браузері, зберігати зміни;

– забезпечити відображення аудіоданих у хвильовій формі на часовій шкалі, можливість виділяти фрагмент аудіо на часовій шкалі, а також виконувати базові операції копіювання, вставлення, видалення фрагменту в різних положеннях часової шкали, обрізання аудіозапису;

– реалізувати можливість застосовувати ефекти до виділених фрагментів аудіо;

– здійснити тестування і налагодження розробленого редактора аудіофайлів.

Висновки до розділу 1

Було здійснено аналіз сучасного стану питання редагування аудіофайлів, розглянуто базові можливості типового редактора аудіо, а також можливість його реалізації на базі HTML5 і JavaScript. Сформовано завдання на кваліфікаційну роботу, що передбачають створення зазначеного аудіоредактора, а також його тестування і налагодження.

РОЗДІЛ 2
СПЕЦИФІКАЦІЯ ВИМОГ ДО РОЗРОБЛЮВАНОЇ СИСТЕМИ

2.1 Аналіз, визначення вимог до розроблюваного програмного забезпечення та проектування програмного забезпечення

У розділі 1 розглянуто функціональні особливості типового редактора аудіо. Виходячи з цього сформулюємо наступні вимоги до розроблюваного редактора аудіо з врахуванням особливостей веб-додатків.

Розроблюваний аудіоредактор повинен відповідати наступним функціональним вимогам:

– можливість додавати новий аудіофайл в додаток для подальшого редагування;

– можливість бачити відкриту аудіодоріжку на шкалі часу в хвильовій формі;

– для аудіозаписів, які мають 2 звукові доріжки (лівий і правий канал), можливість переглядати їх у хвильовій формі на шкалі часу, відображаючи їх одна над одною;

– можливість встановлювати і переміщувати вказівник на шкалі часу;

– можливість виділяти фрагмент аудіодоріжки, або всю аудіодоріжку і знімати виділення;

– можливість розпочинати нове відтворення раніше доданого аудіофайлу або виділеного фрагмента;

– можливість зупиняти поточне відтворення аудіофайлу;

– можливість копіювати вміст виділеного фрагмента, вирізати, вставляти скопійований раніше фрагмент на місці вказівника;

– можливість обрізати звукову доріжку;

– можливість застосовувати звукові ефекти до виділеного фрагмента аудіодоріжки;

– можливість зберегти відредаговану звукову доріжку в формат файлу, придатний для відтворення на комп’ютері.

Вимоги до користувацького інтерфейсу розроблюваного редактора аудіо:

– інтерфейс користувача повинен бути простим, інтуїтивно зрозумілим, візуально сповіщати користувачів, коли в програму завантажено поточний запис;

– інтерфейс користувача повинен візуально сповіщати користувачів, коли він має постійне відтворення;

– інтерфейс користувача повинен забезпечувати спосіб для швидкого запуску нового відтворення та зупинки поточного відтворення;

– інтерфейс користувача повинен забезпечувати спосіб для швидкого збереження звукової доріжки;

– інтерфейс користувача повинен забезпечувати спосіб швидкого виконання інших дій, пов’язаних з редагуванням звукової доріжки;

– інтерфейс користувача повинен бути комфортним для перегляду, зокрема шрифти повинні бути достатнього розміру, кольори забезпечувати достатню контрасність елементів інтерфейсу, при цьому не будучи невиправдано насиченими.

Системні вимоги і вимоги до продуктивності:

– розроблений програмний продукт повинен бути сумісний з новими десктопними версіями найпопулярніших веб-браузерів, зокрема Google Chrome, Mozilla Firefox і Opera незалежно від операційної системи (Windows, Mac OS, Linux);

– функціональність розроблюваного редактора не повинна залежати від плагінів браузера, зокрема користувачі повинні мати змогу відтворювати аудіо, зберігати аудіо без допомоги плагінів браузера;

– взаємодія користувача з елементами інтерфейсу системи при роботі на комп’ютері, параметри якого відповідають мінімальним системним вимогам встановленої на нього операційної системи, не повинна супроводжуватися

втратами чутливості сторінки до дій користувача, заморожуванням елементів інтерфейсу, неплавною прокруткою елементів інтерфейсу на сторінці;

– не допускається надмірне використання мережевих ресурсів;

– можливість працювати локально після повного завантаження програми у браузер, користувачі повинні мати змогу редагувати аудіо навіть при втраті доступу до Інтернету, якщо вкладка браузера з розробленим редактором є повністю завантаженою і залишається відкритою.

2.2 Вибір засобів, методів і алгоритмів вирішення поставленого завдання

Оскільки редактор аудіофайлів буде розроблено на базі веб-технологій, таких як HTML5 і JavaScript, то це дозволяє зупинитися на інструментах розробки, що використовуються для веб-розробки.

Серед популярних редакторів коду можна виділити Sublime Text, Notepad++ і Visual Studio Code [14].

Sublime Text – легкий та швидкий, з мінімалістичним інтерфейсом. Налаштовується за допомогою великого вибору плагінів. Підтримує велику кількість мов програмування. Має функції, такі як роздільне редагування, можливість вибору декількох рядків одночасно та потужну функцію пошуку та заміни. Підходить як для початківців, так і для розробників з досвідом. Доступний для Windows, macOS та Linux.

Notepad++ – безкоштовний редактор з відкритим вихідним кодом. Має простий та зручний інтерфейс, підтримує різноманітні мови програмування та надає підсвічування синтаксису. Має можливості, такі як вкладений інтерфейс, автодоповнення та запис/відтворення макросів. Обмежені можливості розширення порівняно з Sublime Text і Visual Studio Code. Головним чином призначений для користувачів Windows.

Visual Studio Code – безкоштовний редактор коду з відкритим вихідним кодом, розроблений компанією Microsoft. Він має багатий набір функцій для

написання коду та розробки програм. Підтримує велику кількість мов програмування та надає широку інтеграцію з ними. Має налаштовуваний інтерфейс користувача з вбудованим терміналом та інтеграцією з Git.

Visual Studio Code також має велику кількість розширень для додаткової функціональності, підходить для початківців, а також для просунутих розробників. Доступні версії для Windows, macOS і Linux.

Таким чином, Sublime Text відомий своєю швидкістю та гнучкістю, тоді як Notepad++ є легким і простим у користуванні редактором, орієнтованим насамперед на користувачів Windows. З іншого боку, Visual Studio Code – це багатофункціональний редактор із широкою підтримкою мов програмування, параметрами налаштування та великою екосистемою розширень, що робить його популярним вибором серед розробників на різних платформах.

Для розробки нашої системи було використано Visual Studio Code з розширенням Live Server (рис.2.1).

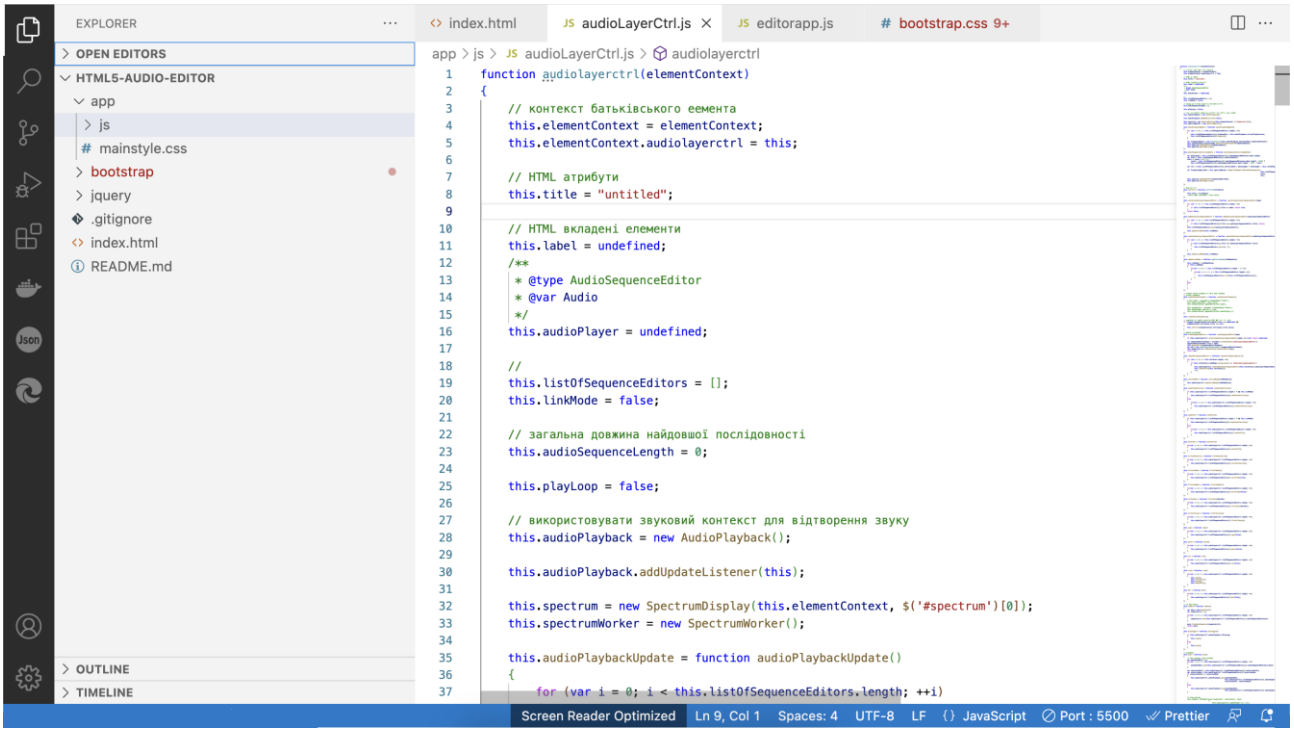

Рисунок 2.1 – Редактор коду Visual Studio Code

Розроблювана програма для редагування аудіофайлів орієнтована в першу

чергу на роботу в Google Chrome, який на сьогодні є одним з найпопулярніших веб браузерів, тестування здійснювалося за допомогою інструментів розробника Chrome. Незважаючи на це можливе використання програми в нових версіях браузерів Mozilla Firefx, Opera та ін.

HTML нерозривно зв'язаний з CSS та JavaScript [10, 15-16]. Без використання цих технологій неможливо уявити сучасний веб-додаток. На сьогоднішній день актуальними є HTML5, CSS3, а також JavaScript специфікації ES6.

Для полегшення розробки використано бібліотеки Bootstrap [17] і jQuery [18].

Bootstrap – це найпопулярніший інструментарій або фреймворк для створення веб-інтерфейсів. Він надає набір готових до використання стилів, шаблонів і компонентів, які допоможуть швидко створювати сучасні, адаптивні та чудові веб-сайти та програми.

Bootstrap пропонує можливість адаптуватися до різних розмірів екрана, дозволяючи веб-сайтам правильно відображатися на пристроях із різною шириною екрана, таких як комп'ютери, планшети та смартфони. Bootstrap також має велику кількість готових компонентів, таких як форми, кнопки, навігаційні панелі, каруселі тощо. Ці компоненти дозволяють швидко додавати функціональність до веб-сайтів без необхідності переписувати з нуля код.

Також Bootstrap надає широкий вибір стилів тексту, заголовків, списків, таблиць та інших типографічних елементів. Це забезпечує єдність і читабельність вмісту сайту.

Налаштування теми: Bootstrap дозволяє налаштовувати зовнішній вигляд веб-додатків, вибираючи попередньо визначені стилі, налаштовувати такі змінні, як кольори, шрифти, відступи тощо.

jQuery – це популярна бібліотека JavaScript, яка спрощує взаємодію з документами HTML, анімацію, обробку подій, AJAX тощо.

jQuery забезпечує простий синтаксис, який спрощує розробку веб-додатків. Він дозволяє виконувати багато рутинних завдань із динамічними

ефектами та маніпулюванням DOM за допомогою кількох простих рядків коду.

jQuery надає прості та потужні методи для вибору, маніпулювання та модифікації елементів DOM, дозволяє змінювати стилі, додавати, видаляти чи змінювати елементи, обробляти події та багато іншого.

jQuery має велику кількість плагінів, які розширюють його функціональність. Ці плагіни додають нові функції, такі як каруселі, перевірка форм, ефекти прокручування, діаграми тощо.

Для розробки інтерфейсу застосовано онлайн-сервіс Figma (рис. 2.2) [19].

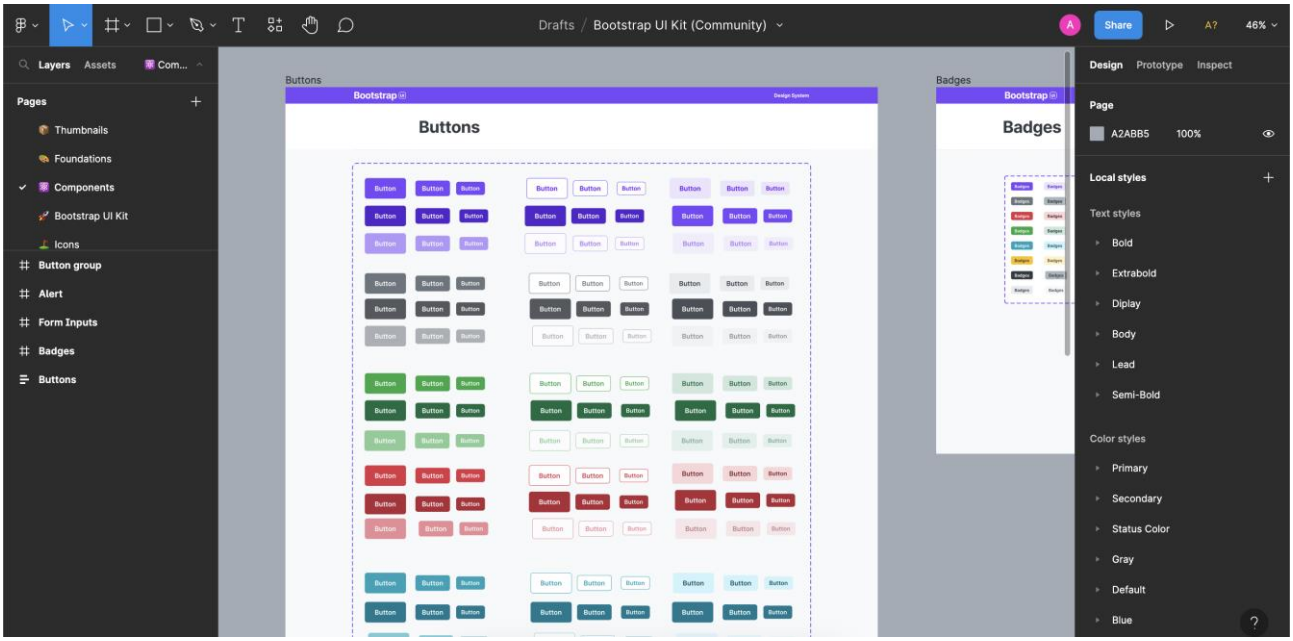

Рисунок 2.2 – Веб-версія додатку Figma

Figma – це веб-додаток, який призначенний для дизайну і прототипування інтерфейсів. Це популярний інструмент, який використовується в дизайні UI/UX для створення графічних макетів, роботи з векторною графікою, спільної роботи в команді та створення інтерактивних прототипів.

Figma працює в онлайн-середовищі. Це означає, що ви можете отримувати доступ до своїх проектів через браузер зз будь-якого пристрою з Інтернет-з’єднанням. Ви можете редагувати ці проекти, спілкуватися з командою та ділитися макетами без необхідності встановлення спеціального програмного забезпечення. Figma надає можливість одночасно спільно

працювати над проектом кільком користувачам. Також Fifma має вбудований редактор векторної графіки, що дозволяє створювати, редагувати та маніпулювати формами, шляхами, стилями та іншими елементами дизайну. Можна створювати складні макети та ілюстрації безпосередньо в Figma. Figma дозволяє створювати інтерактивні прототипи, що демонструють поведінку веб-сторінок та додатків.

Вибір методів вирішення поставленого завдання

До появи елемента <audio>, як складової HTML5, потрібен був Flash або інший плагін для відтворення аудіо в інтернеті. Хоча для аудіо в Інтернеті більше не потрібен плагін, проте <audio>-тег накладає значні обмеження на реалізацію складних комп'ютерних ігор та інтерактивних програм.

Web Audio API – це високорівневий API JavaScript для обробки та синтезу звуку у веб-додатках. Мета цього API полягає в тому, щоб включити можливості, наявні в сучасних ігрових звукових механізмах, і деякі завдання мікшування, обробки та фільтрації, які є в сучасних настільних програмах створення аудіо. Далі наведено короткий вступ до використання цього потужного API [20].

AudioContext призначений для керування та відтворення всіх звуків. Щоб створити звук за допомогою Web Audio API, потрібно створити одне або кілька джерел звуку та підключити їх до місця призначення звуку, наданого примірником AudioContext. Це з'єднання не обов'язково має бути прямим і може проходити через будь-яку кількість проміжних AudioNodes , які діють як модулі обробки аудіосигналу. Ця маршрутизація описана більш детально в специфікації Web Audio .

Один екземпляр AudioContext може підтримувати кілька звукових входів і складні звукові графіки, тому нам знадобиться лише один із них для кожної створеної аудіопрограми. У лістингу 2.1 показано фрагмент коду для створення AudioContext.

Лістинг 2.1 – Приклад створення AudioContext

```
var cntxt1;
     window.addEventListener('load', init1, false);
```

Продовження лістингу 2.1

```
     function init1() {
         try {
             cntxt1 = new AudioContext();
         }
         catch(er) {
             console.log('Веб Audio API не підтримується');
         } }
```

кінець лістингу 2.1

Багато цікавих функцій Web Audio API, наприклад створення AudioNodes і декодування даних аудіофайлів, є методами AudioContext.

Web Audio API використовує AudioBuffer для коротких і середніх звуків. Основний підхід полягає у використанні XMLHttpRequest для отримання звукових файлів.

API підтримує завантаження даних аудіофайлів у різних форматах, таких як WAV, OGG, MP3, AAC та інших . Підтримка браузером різних аудіоформатів може відрізнятися.

Приклад завантаження зразка звуку показано у лістингу 2.2.

Лістинг 2.2 – Завантаження зразка звуку

```
var someBuffer1 = null;
var cntxt2 = new AudioContext();

function loadSound(url) {
    var rqst1 = new XMLHttpRequest();
    rqst1.open('GET', url, true);
    rqst1.responseType = 'arraybuffer';

    // Асинхронно декодувати
```

```
    rqst1.onload = function() {
         cntxt2.decodeAudioData(rqst1.response, function(buff) {
```

Продовження лістингу 2.2

```
              someBuffer1 = buff;
         }, onError);
    }
    rqst1.send();
}
```

кінець лістингу 2.2

ResponseType встановлено як «arraybuffer» для запиту значення, оскільки дані аудіофайлу є двійковими.

Після отримання недекодованих даних аудіофайлу їх можна зберігати для подальшого декодування чи відразу декодувати за допомогою методу AudioContext decodeAudioData(). Цей метод бере дані ArrayBuffer аудіофайлу, що зберігаються в, rqst1.response і декодує їх асинхронно – не блокуючи основний потік JavaScript.

Після того як decodeAudioData() завершиться, він викликає функцію зворотного виклику, що надає декодовані аудіодані PCM як AudioBuffer.

Після завантаження AudioBuffers ми можемо відтворювати один або декілька звуків. Припустімо, що ми щойно завантажили файл AudioBuffer із певним звуком, і завантаження завершилося. Тоді ми можемо відтворити (рис. 2.3) цей буфер за допомогою наступного коду, представленого в лістингу 2.3.

Лістинг 2.3 – Відтворення AudioBuffer

```
var cntx3 = new AudioContext();
function play(buff3) {
    var src3 = cntx3.createBufferSource(); // створює звуковий ресурс
    src3.buffer = buff3;                   // вказує джерелу, який
```

```
    // звук відтворювати
    // з’єднати джерело з призначенням контексту (колонки комп’ютера)
```

Продовження лістингу 2.3

```
    src3.connect(cntx3.destination);
    src3.noteOn(0);                              // відтворити зараз
}
```

кінець лістингу 2.3

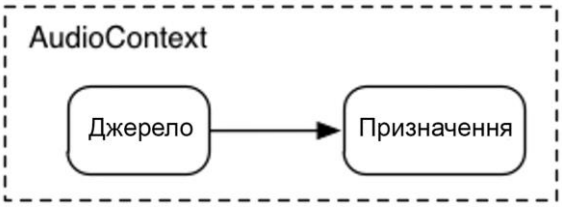

Рисунок 2.3 – Простий звуковий графік

Цю функцію play() можна викликати щоразу, коли хтось натискає клавішу чи клацає щось мишею.

Функція noteOn(time) дозволяє легко запланувати точне відтворення критичних за часом програм. Щоб це планування працювало належним чином, потрібно впевнитись, що звукові буфери попередньо завантажено.

Однією з найпростіших операцій, які можна виконати зі звуком, є зміна його гучності. Використовуючи API веб-аудіо, можна направляти наше джерело до місця призначення через AudioGainNode (вузол підсилення) і таким чином маніпулювати гучністю (рис. 2.4).

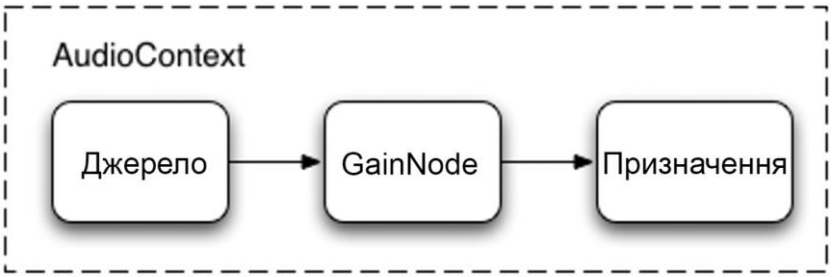

Рисунок 2.4 – Аудіограф з вузлом посилення

Це налаштування підключення може бути досягнуто як показано в лістингу 2.4.

Лістинг 2.4 – Підключення gainNode

```
// Створити вузол підсилення.
var gainNd = context.createGainNode();
// Підключити джерело до вузла підсилення.
source.connect(gainNd);

// Підключити вузол підсилення до призначення.
gainNd.connect(context.destination);
```

кінець лістингу 2.4

Після цього можна програмно змінити гучність, маніпулюючи gainNd.gain.value наступним чином як показано в лістингу 2.5.

Лістинг 2.5 – Зміна гучності аудіо

```
// Зменшити гучність
gainNd.gain.value = 0.25;
```

кінець лістингу 2.5

Web Audio API дозволяє передавати звук з одного аудіовузла в інший, створюючи потенційно складний ланцюжок процесорів для додавання складних ефектів до звукових форм.

Один із способів зробити це – розмістити BiquadFilterNode (рис. 2.5) між джерелом звуку та призначенням. Цей тип аудіовузла може виконувати різноманітні фільтри низького порядку, які можна використовувати для створення графічних еквалайзерів і навіть більш складних ефектів, здебільшого пов’язаних із вибором частин частотного спектру звуку, які потрібно підкреслити, а які – приглушити.

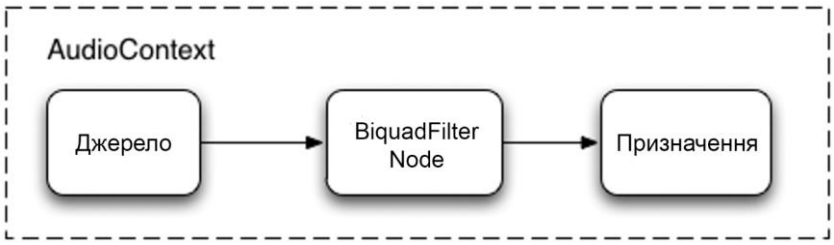

Рисунок 2.5 – Аудіограф із вузлом біквадратного фільтра

Серед підтримуваних типів фільтрів: фільтр низьких частот, фільтр високих частот, смуговий фільтр, піковий фільтр та ін.

Всі фільтри включають параметри для визначення певної величини посилення, частоти застосування фільтра та коефіцієнта якості. Фільтр низьких частот зберігає низький діапазон частот, але відкидає високі частоти. Точка розриву визначається значенням частоти. Коефіцієнт посилення впливає лише на певні фільтри, такі як фільтр низьких частот і піковий фільтр, а не на цей фільтр низьких частот.

Приклад встановлення простого фільтра низьких частот для витягування лише основи із звукового зразка, показано у лістингу 2.6.

Лістинг 2.6 – Застосування фільтра низьких частот

```
// Створити фільр
var filter1 = context.createBiquadFilter();
// Створити аудіограф
source.connect(filter1);
// Створити і визначити параметри для низькорівневого фільтра
filter1.connect(context.destination);
// Низькорівневий фільтр
filter1.type = 0;
filter1.frequency.value = 420;
// відтворити звук
source.noteOn(0);
```

кінець лістингу 2.6

Висновки до розділу 2

У цьому розділі визначено вимоги до розроблюваного редактора аудіо, зокрема функціональні вимоги, вимоги до інтерфейсу користувача, системні вимоги. Також проаналізовано і вибрано засоби і технології для розробки. Для розробки редактора аудіо було використано Web Audio API.

РОЗДІЛ 3
РОЗРОБКА РЕДАКТОРА АУДІОФАЙЛІВ НА БАЗІ HTML5

3.1 Практична реалізація об’єкта проектування

Для роботи програми файли повинні знаходитись на сервері. Доступ до програми здійснюється через браузер шляхом введення URL-адреси.

Головним файлом, точкою входу в програму, є файл index.html, що містить HTML-розмітку елементів інтерфейсу програми.

Графічний інтерфейс користувача розроблено з використанням бібліотеки Bootstrap. Для цього на початку сторінки index.html підключені файли стилів CSS, а також файли скриптів JavaScript, що забезпечують коректне функціонування всіх графічних елементів.

Додаткові стилі CSS розміщено у файлі app/mainstyle.css у папці з проектом.

Крім того використовується бібліотека JQuery, що спрощує написання коду на JavaScript.

Файли JavaScript, що забезпечують можливість роботи з аудіофайлами і їх редагування розміщено в папці app/js.

Для реалізації аудіоредактора використано Web Audio API замість звичайного аудіотега HTML5 через незручності, пов'язані з останнім. Розроблюваний редактор аудіо безпосередньо використовує Web Audio API для відтворення буферизованих аудіоданих.

Перетворюючи AudioBuffer, що містить аудіодані, на AudioNode та підключаючи його до місця призначення динаміків користувача за замовчуванням AudioContext, можна легко відтворювати обрізаний аудіозапис, вказавши час і тривалість початку під час запиту AudioNode на відтворення (додаток А).

Функціональна версія редактора аудіо показана на рисунку 3.1, а HTML-розмітка в додатку Б.

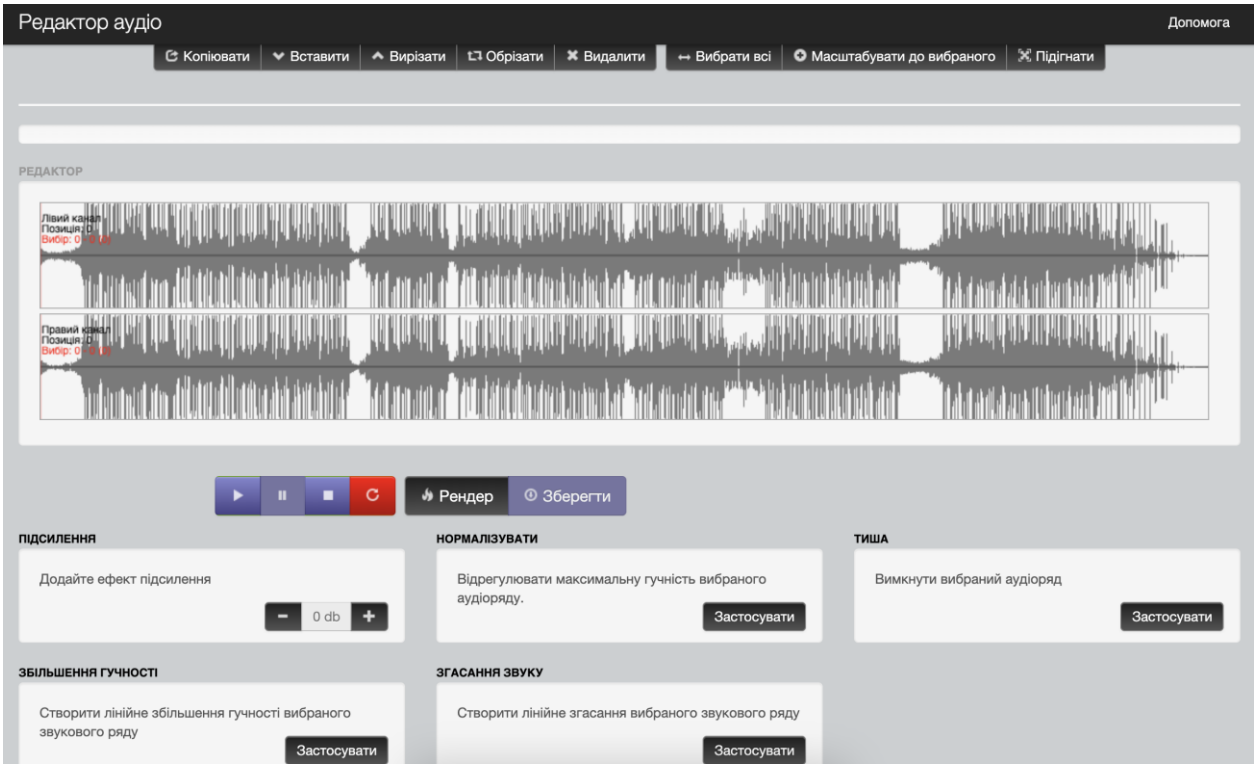

Рисунок 3.1 – Графічний інтерфейс користувача розробленого редактора аудіо

Щоб розпочати відтворення, користувачі можуть натиснути кнопку «Відтворити». Щоб зупинити поточне відтворення, користувачі можуть натиснути кнопку «Зупинити». Кнопка «Пауза» призупиняє відтворення зі збереженням вказівника на часовій шкалі.

Програма працює наступним чином. Для початку роботи потрібно перетягнути аудіофайл в область редактора, з текстом «Перетягніть аудіофайл для редагування» (рис. 3.2), після чого відобразиться хвильовий вигляд аудіозапису і з’являться можливості для відтворення і редагування.

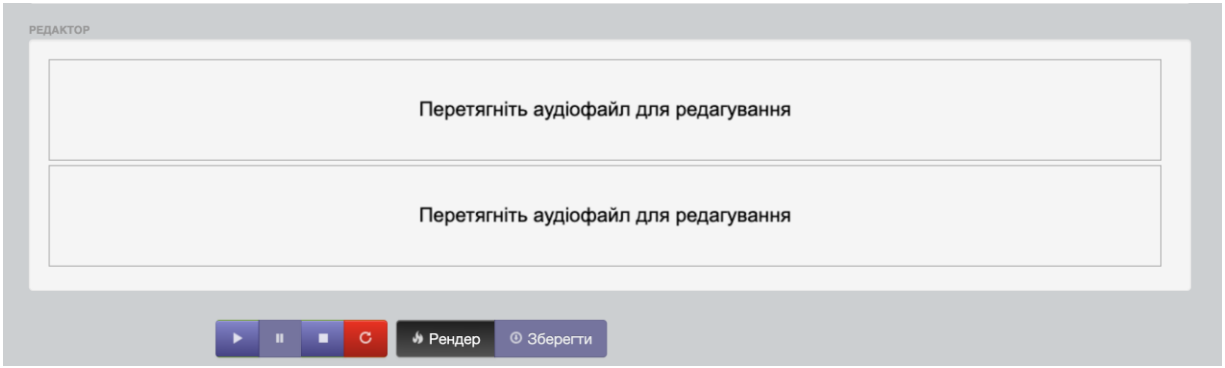

Рисунок 3.2 – Область для редагування перед завантаженням файлу

Масштабування дозволяє користувачам збільшувати та зменшувати співвідношення часу до пікселів за допомогою кнопок «Масштабувати до вибраного» і «Підігнати». У першому випадку вибрана область розширюється до розміру редактора, в іншому стискається, щоб повністю відобразити аудіозапис в редакторі.

Збільшення масштабу призводить до меншого співвідношення часу і пікселів, створюючи збільшений вигляд меншої частини аудіо доріжки. І навпаки, зменшення призводить до більшого співвідношення часу до пікселів, створюючи зменшений огляд більшої ділянки доріжки. Скидання масштабування за допомогою кнопки «Підігнати» повертає значення масштабу до базового масштабу 100%, що відповідає найбільш зменшеному перегляду, де вся звукова доріжка точно поміщається в ширину інтерфейсу користувача.

Розроблюваний редактор аудіо також має підтримку двоканальних звукових доріжок. Розроблюваний редактор аудіо належним чином завантажує та відображає два канали у дисплеї осцилограми. Завдяки підтримці двоканального звуку користувачі можуть редагувати файли об'ємного звуку.

Для застосування аудіоефекту потрібно вибрати фрагмент аудіо на осцилограмі і натиснути на кнопку «Застосувати» одного з доступних ефектів (рис. 3.3). Можливе кількаразове послідовне накладання ефектів до виділеного фрагменту аудіо.

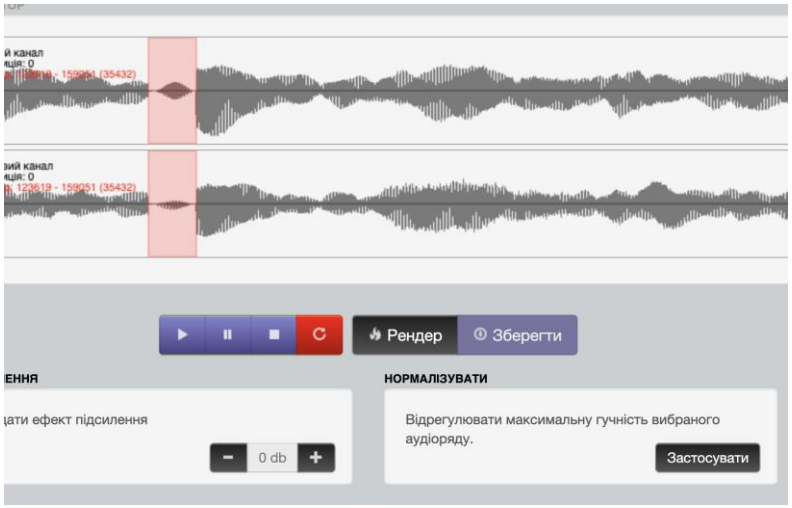

Рисунок 3.3 – Застосування аудіоефектів

Розроблюваний редактор аудіо використовує API Web Audio та MediaRecorder [8] для збереження аудіо. Розроблюваний редактор аудіо підтримує збереження у аудіоформаті WAV. Розроблюваний редактор аудіо створює формат WAV, вручну кодуючи відредаговану звукову доріжку за допомогою 32-бітного формату IEEE-float.

Для збереження файлу потрібно натиснути кнопку «Рендер» і «Зберегти», в результаті чого буде завантажено змінений аудіофайл, який можна відтворювати в будь-якому популярному аудіоплеєрі.

3.2 Тестування та налагодження інформаційно-комп'ютерної системи

Метою тестування розробленого веб-додатку є перевірка та оцінкиа функціональності, надійності, безпеки та продуктивності веб-додатку з метою забезпечення його якості [21].

Основні аспекти, які підлягають тестуванню:

– функціональність: перевірка, чи виконує веб-додаток свої функції згідно вимог та специфікацій, описаних в п. 2.1. Це включає тестування функцій, роботу з формами, навігацію та інші інтерактивні елементи;

– користувацький інтерфейс: перевірка зручності та коректності відображення додатка на різних пристроях та браузерах. Це включає перевірку забезпечення однакового досвіду користувача на різних платформах та перевірку відповідності дизайну інтерфейсу;

– надійність: виконання тестів для виявлення помилок, відмов та несправностей у додатку, включає тестування стабільності, обробку помилок та виявлення потенційних проблем;

– продуктивність: вимірювання та перевірка продуктивності веб-додатка.

Варто зазначити, що розроблений редактор аудіофайлів з самого початку орієнтований на роботу на екранах комп'ютерів, тому тестування не здійснювалося на мобільних пристроях. Також особливість розробки не

передбачає тестування безпеки, оскільки розроблений редактор не несе в собі загрози цілісності чи конфіденційності інформації, а механізми захисту забезпечуються на рівні операційної системи і веб-браузера.

В результаті тестування функціональності було встановлено працездатність всіх раніше заявлених функцій програми.

Інтерфейс користувача простий і зручний, елементи інтерфейсу коректно відображаються на екранах різних розмірів і роздільної здатності (рис. 3.4). При цьому не беруться до уваги екрани мобільних пристроїв, для яких цей додаток не розроблявся.

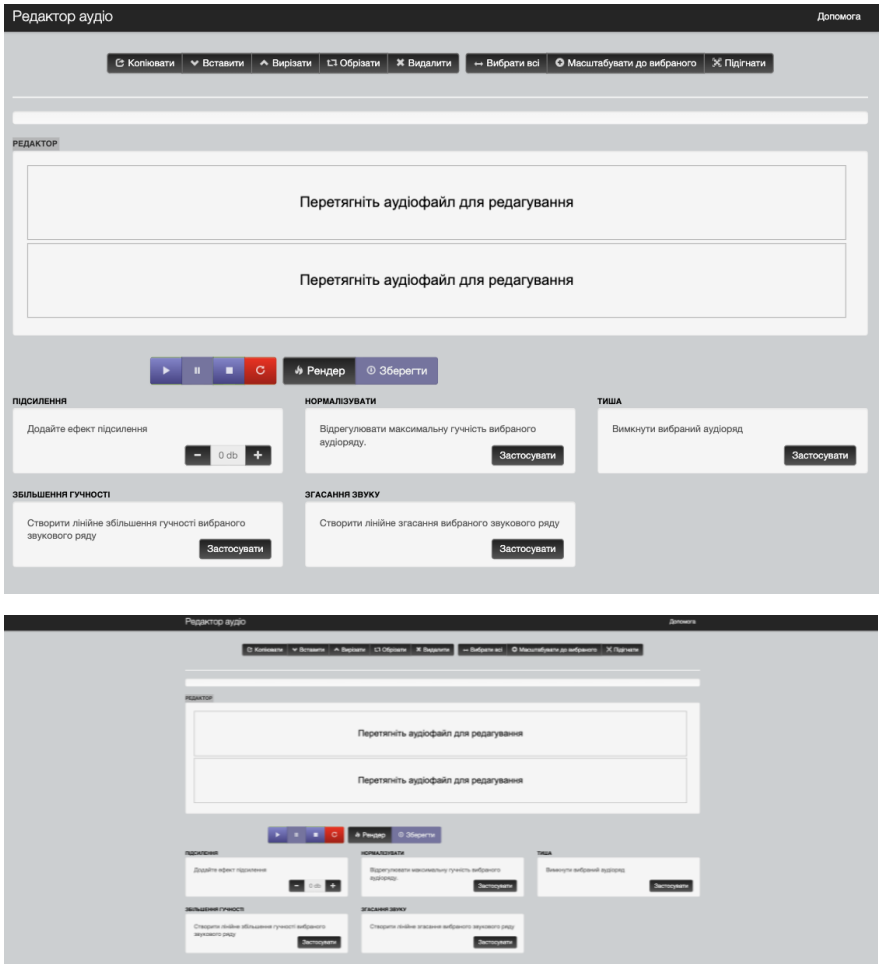

Рисунок 3.4 – Тестування редактора аудіо на екранах різних розмірів

Тестування проводилось для різних браузерів, а саме Chrome, Mozilla Firefox, Opera, Microsoft Edge найновіших версій. В результаті не було виявлено проблем, пов’язаних з несумісністю з браузером певного типу.

Тестування надійності полягало в перевірці працездатності редактора при роботі з файлами різних форматів: WAV, mp3, різного розміру, тривалості запису, з однією і двома звуковими доріжками. Зниження швидкодії і продуктивності спостерігалось при роботі з дуже великими файлами. Максимальний розмір файлу визначається особливостями конкретної системи і може змінюватися залежно від апаратного забезпечення і браузера. В цілому програма працює швидко і стабільно для невеликих файлів з тривалістю аудіозапису кілька хвилин.

Для виявлення помилок, а також показників продуктивності і швидкості завантаження використовувались інструменти розробника Google (рис. 3.5).

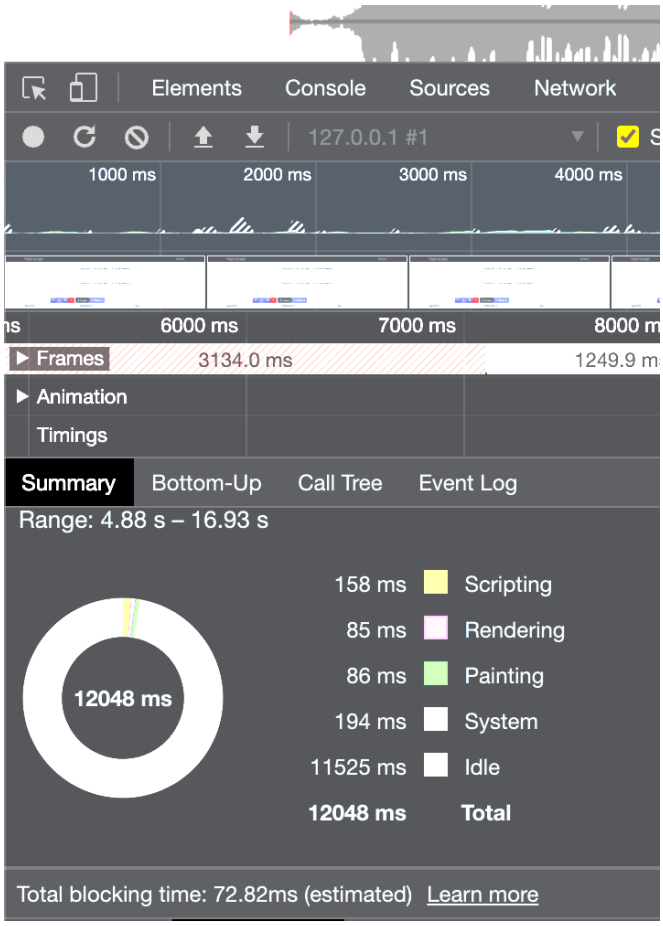

Рисунок 3.5 – Тестування продуктивності додатка

Синтаксичні помилки, що виявлялися у консолі інструментів розробника в процесі написання коду, були успішно усунуті в фінальній версії розробки.

Висновки до розділу 3

Було розроблено редактор аудіофайлів, що дозволяє відкривати, відтворювати і редагувати аудіофайли в браузері, а також зберігати одержаний результат у вигляді нового аудіофайла. Редагування передбачає виділення фрагмента аудіозапису з можливістю його копіювання, вирізання, вставлення з буфера обміну, застосування ефектів до вибраного фрагмента. Додаток підтримує роботу з двоканальними аудіозаписами. Проведено тестування і налагодження додатка. Підтверджено його високу швидкодію і стабільну роботу в нових версіях браузерів Chrome, Mozilla Firefox, Opera і Edge. Додаток може повноцінно працювати після зникнення Інтернет-з’єднання за умови повного попереднього завантаження сторінки у браузері, всі функції реалізовано на стороні клієнта на базі технологій HTML5, CSS3 та JavaScript.

ВИСНОВКИ

За результатами проведеного дослідження варто зробити відповідні висновки:

– здійснено аналіз сучасного стану проблеми та огляд існуючих аналогів редакторів аудіозаписів, зокрема WaveLab, CoolEdit, Sound Forge та SAW Plus;

– визначено вимоги до розроблюваного редактора аудіофайлів на базі HTML5 і JavaScript, приділена увага інтуїтивно зрозумілому і простому інтерфейсу користувача, а також системним вимогам;

– вибрано технології, алгоритми і засоби, що дозволять здійснити розробку редактора аудіофайлів на базі HTML5 і JavaScript. Зокрема для розробки використано Visual Studio Code з розширенням LiveServer, для розробки дизайну інтерфейсу користувача – Figma, в проекті використано бібліотеки Bootstrap, jQuery, а функціональні можливості редактора забезпечуються за допомогою Web Audio API;

– розроблений редактор аудіофайлів, дозволяє вносити зміни в існуючий аудіофайл безпосередньо в браузері, зберігати зміни, забезпечує відображення аудіоданих у хвильовій формі (осцилограма) на часовій шкалі, можливість виділяти фрагмент аудіо на часовій шкалі, а також виконувати базові операції копіювання, вставлення, видалення фрагменту в різних положеннях часової шкали, обрізання аудіозапису;

– існує можливість застосовувати ефекти до виділених фрагментів аудіо;

– в результаті тестування і налагодження розробленого редактора аудіофайлів було виявлено і усунуто недоліки.

Отже, всі поставлені завдання було успішно виконано. Розроблений додаток повністю функціональний, проте не виключає додавання нових функцій і розширених можливостей в наступних версіях.

СПИСОК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Daniel Scarratt. The evolution of audio in videogames. URL: https://www.acmi.net.au/stories-and-ideas/evolution-audio-videogames/ (date of access: 10.03.2023).

2. A Brief History of Windows Audio APIs. URL: http://shanekirk.com/2015/10/a-brief-history-of-windows-audio-apis/ (date of access: 10.03.2023).

3. Audio file format. URL: https://en.wikipedia.org/wiki/Audio_file_format (date of access: 12.03.2023).

4. 13 Best Audio Editing Software [2023 Audio Editor Review] URL: https://www.softwaretestinghelp.com/best-audio-editing-software/ (date of access: 12.03.2023).

5. What is EQ and Compression: The Basics of Home Recording (Part I) URL: https://www.sageaudio.com/blog/pre-mastering/what-is-eq-and-compression-the-basics-of-home-recording-part-i (date of access: 12.03.2023).

6. The History of DirectX. URL: https://www.codingunit.com/the-history-of-directx (date of access: 18.03.2023).

7. Audio Plugin Formats. URL: https://frozenplain.com/audio-plugin-formats/ (date of access: 18.03.2023).

8. Paul Adenot and Chris Wilson. Web Audio API. W3C Audio Working Group. 2015. URL: https://www.w3.org/TR/webaudio/. Working Draft (date of access: 19.03.2023).

9. Daniel C. Burnett et al. Media Capture and Streams. W3C Devices and Sensors Working Group, W3C Web Real-Time Communications Working Group. URL: https://www.w3.org/TR/2016/CR-mediacapture-streams-20160519/ (date of access: 05.04.2023).

10. Ian Hickson et al. HTML5. A vocabulary and associated APIs for HTML and XHTML. W3C HTML Working Group and WHATWG. URL: https://www.w3.org/TR/2014/REC-html5-20141028/ (date of access: 17.04.2023).

11. Dave Raggett. HTML 3.2 Reference Specification. W3C. URL: https: //www.w3.org/TR/REC-html32 (date of access: 20.04.2023).

12. Dave Raggett, Arnaud Le Hors, and Ian Jacobs. HTML 4.01 Specification. W3C HTML Working Group. URL: https://www.w3.org/TR/html401/ (date of access: 24.04.2023).

13. Magix Sound Forge 12. URL: https://www.soundonsound.com/reviews/magix-sound-forge-12 (date of access: 11.05.2023).

14. Ishika Kesarwani. Top 10 Code Editors — A Must-Have List for Developers. URL: https://www.altogic.com/blog/top-10-code-editors (date of access: 11.05.2023).

15. Descriptions of all CSS specifications. URL: https://www.w3.org/Style/CSS/specs.en.html (date of access: 13.05.2023).

16. JavaScript web APIs. URL: https://www.w3.org/standards/webdesign/script (date of access: 16.05.2023).

17. Alexandre Ouellette. What is Bootstrap: A Beginner's Guide. URL: https://careerfoundry.com/en/blog/web-development/what-is-bootstrap-a-beginners-guide/ (date of access: 16.05.2023).

18. What Is jQuery? A Look At the Web’s Most-Used JavaScript Library. URL: https://kinsta.com/knowledgebase/what-is-jquery/ (date of access: 10.03.2023).

19. What is Figma? URL: https://help.figma.com/hc/en-us/articles/14563969806359-What-is-Figma- (date of access: 19.05.2023).

20. Boris Smus. Getting started with Web Audio API. URL: https://web.dev/webaudio-intro/ (date of access: 23.05.2023).

21. What is software testing? URL: https://www.ibm.com/topics/software-testing (date of access: 27.05.2023).

ДОДАТКИ

Додаток А

HTML-розмітка редактора аудіо

```
<!DOCTYPE html>
<html lang="uk">
 <head>
  <meta charset="utf-8">
  <title>Редактор аудіо</title>
  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <link href="bootstrap/css/bootstrap.css" rel="stylesheet">
  <!— інші ресурси … -->
 </head>
 <body onload="onDocumentLoaded()">
  <div class="navbar navbar-fixed-top">
   <div class="navbar-inner">
    <div class="container">
     <a class="brand" href="#">Редактор аудіо</a>
     <ul class="nav pull-right">
      <li><a data-toggle="modal" href="#helpAsModal">Допомога</a></li>
     </ul>
    </div>
   </div>
  </div>
  <div class="container">
   <div class="row">
    <div class="span12">
     <center>
      <div class="btn-toolbar">
       <div class="btn-group">
        <a class="btn btn-primary" onclick="$('#audiolayerctrl')[0].copy();"><i class="icon-share icon-white"></i> Копіювати</a>
        <a class="btn btn-primary" onclick="$('#audiolayerctrl')[0].paste();"><i class="icon-chevron-down icon-white"></i>
Вставити</a>
        <a class="btn btn-primary" onclick="$('#audiolayerctrl')[0].cut();"><i class="icon-chevron-up icon-white"></i> Вирізати</a>
        <a class="btn btn-primary" onclick="$('#audiolayerctrl')[0].crop();"><i class="icon-retweet icon-white"></i> Обрізати</a>
        <a class="btn btn-primary" onclick="$('#audiolayerctrl')[0].del();"><i class="icon-remove icon-white"></i> Видалити</a>
       </div>
       <div class="btn-group">
        <a class="btn btn-primary" onclick="$('#audiolayerctrl')[0].selectAll();"><i class="icon-resize-horizontal icon-white"></i>
Вибрати все</a>
        <a class="btn btn-primary" onclick="$('#audiolayerctrl')[0].zoomIntoSelection();"><i class="icon-plus-sign icon-white"></i>
Масштабувати до вибраного</a>
        <a class="btn btn-primary" onclick="$('#audiolayerctrl')[0].zoomToFit();"><i class="icon-fullscreen icon-white"></i>
Підігнати</a>
       </div>
      </div>
     </center>
    </div>
   </div>
   <hr />
   <div class="row">
    <div class="span12">
     <div class="progress progress-striped active">
      <div id="app-progress" class="bar" style="width: 0%;"></div>
     </div>
    </div>
```

```
    </div>
    <div class="row">
      <div class="span12">
        <h6>Редактор</h6>
        <div class="well">
          <audiolayerctrl id="audiolayerctrl" title="demo.mp3" />
        </div>
      </div>
    </div>
    <div class="row">
      <div class="span8">
        <center>
          <div class="btn-toolbar">
            <div class="btn-group">
              <a id="btn_play" class="btn btn-success btn-large"
onclick="$('#btn_pause').removeClass('active');document.getElementById('audiolayerctrl').play();" rel="tooltip" title="first tooltip"><i
class="icon-play icon-white"></i></a>
              <a id="btn_pause" class="btn btn-success btn-large" data-toggle="button"
onclick="document.getElementById('audiolayerctrl').stop(true)"><i class="icon-pause icon-white"></i></a>
              <a id="btn_stop" class="btn btn-success btn-large"
onclick="$('#btn_pause').removeClass('active');document.getElementById('audiolayerctrl').stop();"><i class="icon-stop icon-
white"></i></a>
              <a id="btn_loop" class="btn btn-warning btn-large" data-toggle="button"
onclick="document.getElementById('audiolayerctrl').toggleLoop();"><i class="icon-repeat icon-white"></i></a>
            </div>
            <div class="btn-group" style="display: none;">
              <a id="btn_spect" class="btn btn-primary btn-large" onclick="$('#spectrumModal').modal('show');">Спектр</a>
            </div>
            <div class="btn-group">
              <a class="btn btn-large btn-primary" onclick="document.getElementById('audiolayerctrl').save($('#savelink')[0]);"><i
class="icon-fire icon-white"></i> Рендер</a>
              <a class="btn btn-large btn-success disabled" id="savelink" download="export.wav"><i class="icon-download icon-white"></i>
Зберегти</a>
            </div>
          </div>
        </center>
      </div>
    </div>
    <div class="row">
      <div class="span4">
        <h6 style="color: black;">Підсилення</h6>
        <div class="well" style="height: 50px; position:relative">
          <p>Додати ефект підсилення</p>

          <div class="pull-right" style=" position:absolute; bottom: 10px; right: 15px;" >
            <div class="btn-group" >
              <button class="btn btn-primary" onclick="decrease_db()"><i class="icon-minus icon-white"></i></button>
              <button id="gain-db" class="btn disabled" onclick="gain_btn_clicked()">0 db</button>
              <button class="btn btn-primary" onclick="increase_db()"><i class="icon-plus icon-white"></i></button>
            </div>
          </div>
        </div>
      </div>
      <div class="span4">
        <h6 style="color: black;">Нормалізувати</h6>
        <div class="well" style="height: 50px; position:relative">
          <p>Відрегулювати максимальну гучність вибраного аудіоряду.</p>
```

```
          <a style="position:absolute; bottom: 10px; right: 15px"  class="btn btn-primary"
onclick="$('#audiolayerctrl')[0].filterNormalize();">Застосувати</a>
          <br>
         </div>
        </div>
        <div class="span4">
         <h6 style="color: black;">Тиша</h6>
         <div class="well" style="height: 50px; position:relative">
          <p>Вимкнути вибраний аудіоряд</p>
          <a style="position:absolute; bottom: 10px; right: 15px"  class="btn btn-primary pull-right"
onclick="$('#audiolayerctrl')[0].filterSilence();">Застосувати</a>
          <br>
         </div>
        </div>
        <div class="span4">
         <h6 style="color: black;">Збільшення гучності</h6>
         <div class="well" style="height: 50px; position:relative">
          <p>Створити лінійне збільшення гучності вибраного звукового ряду</p>
          <a style="position:absolute; bottom: 10px; right: 15px"  class="btn btn-primary pull-right"
onclick="$('#audiolayerctrl')[0].filterFadeIn();">Застосувати</a>
          <br>
         </div>
        </div>
        <div class="span4">
         <h6 style="color: black;">Згасання звуку</h6>
         <div class="well" style="height: 50px; position:relative">
          <p>Створити лінійне згасання вибраного звукового ряду</p>
          <a style="position:absolute; bottom: 10px; right: 15px" class="btn btn-primary pull-right"
onclick="$('#audiolayerctrl')[0].filterFadeOut();">Застосувати</a>
          <br>
         </div>
        </div>
       </div>
       <br>
       <div class="row" style="display: none;">
        <div class="span8">
        </div>
       </div>
     <footer class="footer"></footer>
     <div class="modal hide" id="spectrumModal">
       <div class="modal-header">
        <button type="button" class="close" data-dismiss="modal">×</button>
        <h3>Спектр</h3>
       </div>
       <div class="modal-body">
        <div class="span4">
         <h6>Спектр</h6>
         <div class="well" style="height: 500px; position:relative">
          <div id="spectrum"></div>
         </div>
        </div>
       </div>
       <div class="modal-footer">
        <a href="#" class="btn" data-dismiss="modal">Закрити</a>
       </div>
     </div>
     <div class="modal hide" id="helpAsModal">
       <div class="modal-header">
```

```
      <button type="button" class="close" data-dismiss="modal">×</button>
      <h3>Допомога</h3>
    </div>
    <div class="modal-body">
      <p>Програма для редагування аудіофайлів</p>
    </div>
    <div class="modal-footer">
      <a href="#" class="btn" data-dismiss="modal">Закрити</a>
    </div>
  </div>

  <div class="modal hide" id="welcomeAsModal">
    <div class="modal-header">
      <button type="button" class="close" data-dismiss="modal">×</button>
      <h3>Привіт!</h3>
    </div>
    <div class="modal-body">
      <p>Це аудіоредактор на базі html5 і JavaScript.<br>
        Перетягніть музичний файл і почніть його редашувати</p>
    </div>
    <div class="modal-footer">
      <a href="#" class="btn" data-dismiss="modal">ОК</a>
    </div>
  </div>
  <script src="jquery/js/jquery-1.7.2.js"></script>
  <!—інші файли скриптів … -->
  <script type="text/javascript" src="app/js/editorapp.js"></script>
 </body>
</html>
```

Додаток Б

Фрагмент коду аудіоредактора

```
function AudioPlayback()
{
    /**
     * Внутрішня подія оновлення, щоб заповнити буфер аудіоданими
     */
    this.onAudioUpdate = function onAudioUpdate(evt)
    {
        var audioPlayback = this.eventHost;
        var bufferSize = audioPlayback.audioBufferSize;
        var elapsedTime = bufferSize / audioPlayback.sampleRate;

        // повернути якщо плейбек зупинено
        if (audioPlayback.isPlaying === false) return;

        // посилання на масив аудіоданих і аудіобуфер
        var audioData = audioPlayback.audioDataRef;
        var leftBuffer = evt.outputBuffer.getChannelData(0);
        var rightBuffer = evt.outputBuffer.getChannelData(1);

        if (audioData.length == 1) // моно
        {
            audioPlayback.copyChannelDataToBuffer(leftBuffer, audioData[0], audioPlayback.currentPlayPosition, bufferSize,
audioPlayback.playStart, audioPlayback.playEnd, audioPlayback.isLooped);
            audioPlayback.currentPlayPosition = audioPlayback.copyChannelDataToBuffer(rightBuffer, audioData[0],
audioPlayback.currentPlayPosition, bufferSize, audioPlayback.playStart, audioPlayback.playEnd, audioPlayback.isLooped);
        }
        else if (audioData.length == 2) // стерео
        {
            audioPlayback.copyChannelDataToBuffer(leftBuffer, audioData[0], audioPlayback.currentPlayPosition, bufferSize,
audioPlayback.playStart, audioPlayback.playEnd, audioPlayback.isLooped);
            audioPlayback.currentPlayPosition = audioPlayback.copyChannelDataToBuffer(rightBuffer, audioData[1],
audioPlayback.currentPlayPosition, bufferSize, audioPlayback.playStart, audioPlayback.playEnd, audioPlayback.isLooped);
        }
        // плейбек завершено
        if (audioPlayback.currentPlayPosition === undefined)
        {
            // зупинити відтворення, від'єднати від буфера
            audioPlayback.stop();
        }
        else
        {
            // Оновити обробник сповіщень
            audioPlayback.lastPlaybackUpdate -= elapsedTime;
            if (audioPlayback.lastPlaybackUpdate < 0.0)
            {
                audioPlayback.lastPlaybackUpdate = audioPlayback.playbackUpdateInterval;
                audioPlayback.notifyUpdateListener();
            }
        }
    };

    /**
     * Копіює аудіо дані в буфер каналу і встановоює нову позицію для відтворення. Якщо циклічне відтворення увімкнено, то
     * позиція встановоюється автоматично.
```

```
    * @param bufferReference Посмилання на буфер каналу
    * @param dataReference Посилання на аудіодані
    * @param position Поточна позиція плейбеку
    * @param len Довжина фрагмента
    * @param startPosition Початкова позиція для циклу
    * @param endPosition Кінцека позиція для циклу
    * @param isLooped Вклбчити киклічне відтворення.
    */
   this.copyChannelDataToBuffer = function copyChannelDataToBuffer(bufferReference, dataReference, position, len, startPosition,
endPosition, isLooped)
   {
      /* Щоб увімкнути цикл, нам потрібно розділити аудіо, коли досягнуто кінця аудіоданих,
        щоб почати з першої позиції. Тому нам необхідно розділити на два діапазони
       */
      var firstSplitStart = position;
      var firstSplitEnd = (position + len > dataReference.length) ?
                              dataReference.length : (position + len > endPosition) ?
                                    endPosition : (position + len);

      var firstSplitLen = firstSplitEnd - firstSplitStart;

      var secondSplitStart = (firstSplitLen < bufferReference.length) ?
                              (isLooped) ? startPosition : 0 : undefined;

      var secondSplitEnd = (secondSplitStart !== undefined) ? bufferReference.length - firstSplitLen + secondSplitStart : undefined;

      var secondSplitOffset = bufferReference.length - (firstSplitEnd - firstSplitStart);

      if (secondSplitStart === undefined)
      {
         this.copyIntoBuffer(bufferReference, 0, dataReference, firstSplitStart, firstSplitEnd);
         return firstSplitEnd;
      }
      else
      {
         this.copyIntoBuffer(bufferReference, 0, dataReference, firstSplitStart, firstSplitEnd);

         if (isLooped)
         {
            this.copyIntoBuffer(bufferReference, firstSplitLen, dataReference, secondSplitStart, secondSplitEnd);

            return secondSplitEnd;
         }
         else
         {
            return undefined;
         }
      }
   };


   /**
    * копіює дані з масиву в буфер за допомогою методів швидкого копіювання
    */
   this.copyIntoBuffer = function copyIntoBuffer(bufferReference, bufferOffset, dataReference, dataOffset, end)
   {
      bufferReference.set(dataReference.slice(dataOffset, end), bufferOffset);
   };
```

```
    this.play = function play(audioDataRef, sampleRate, isLooped, start, end)
    {
        this.isPaused = false;
        // перевіряємо, чи вже відтворюється, чи дані не надано
        if (this.isPlaying || audioDataRef === undefined || audioDataRef.length < 1 ||
            sampleRate === undefined || sampleRate <= 0) return;


        // оновити змінні відтворення
        this.audioDataRef = audioDataRef;
        this.sampleRate = sampleRate;
        this.isLooped = (isLooped === undefined) ? false : isLooped;
        this.playStart = (start === undefined || start < 0 || start >= audioDataRef[0].length) ? 0 : start;
        this.playEnd = (end === undefined || end - this.audioBufferSize < start || end >= audioDataRef[0].length) ? audioDataRef[0].length :
end;
        this.currentPlayPosition = this.playStart;
        this.isPlaying = true;

        // під'єднатися до вузла, відтворювати
        this.javaScriptNode.connect(this.analyserNode);

        // повідомити обробник оновлень
        this.notifyUpdateListener();
    };
    /**
     * Зупиняє відтворення та встановлює для всіх посилань значення undefined (відновлення неможливе)
     */
    this.stop = function stop()
    {
        this.isPaused = false;
        // аудіо не відтворюється, нема чого зупиняти
        if (this.isPlaying === false) return;
        // від'єднати вузол, зупнинити відтворення
        this.javaScriptNode.disconnect(this.analyserNode);

        // встановити всю інформацію про відтворення за замовчуванням
        this.playStart = 0;
        this.playEnd = 0;
        this.isLooped = false;
        this.currentPlayPosition = 0;
        this.isPlaying = false;
        this.lastPlaybackUpdate = 0;

        // видалити посилання на аудіодані
        this.audioDataRef = undefined;
        this.sampleRate = 0;

        // інформувати обробник оновлення
        this.notifyUpdateListener();
    };

    /**
     * Призупинити відтвонення
     */
    this.pause = function pause()
    {
```

```
    // не відтворюється аудіо, немає нічого для паузи
    if (this.isPlaying === false) {
        if (this.isPaused) {
            this.isPaused = false;
           // console.log("Відновити", this.eventHost, this.resume());
            //  this.audiolayerctrl.stopFromAudioContext();
              this.resume();
              return;
        }

        return;
    }
    this.isPlaying = false;
  //  this.lastPlaybackUpdate = 0;

    // diconnect the node, stop!
    this.javaScriptNode.disconnect(this.analyserNode);
    this.isPaused = true;
    // inform updatelistener
    this.notifyUpdateListener();
};

/**
 * RВідновити відтворення з останньої позиції
 */
this.resume = function resume()
{
    // перевірити чи вже відтворюється або є дані
    if (this.isPlaying || this.audioDataRef === undefined || this.audioDataRef.length < 1) return;
    this.isPlaying = true;

    // приєлнатись до вузла, відтворбвати аудіо
    this.javaScriptNode.connect(this.analyserNode);

    // проінформувати обробник оновлень
    this.notifyUpdateListener();
};

/**
 * Додати обробник оновлень, який буде проінформований про зміни в плейбеці
 */
this.addUpdateListener = function addUpdateListener(updateCallback)
{
    this.updateListener.push(updateCallback);
};


/**
 * інформує обробник оновлень
 */
this.notifyUpdateListener = function notifyUpdateListener()
{
    for(var i = 0; i < this.updateListener.length; ++i)
    {
        this.updateListener[i].audioPlaybackUpdate();
    }
};
```

```
    // Створення нового аудіоконтексту
    this.audioBufferSize = 1024;
    this.sampleRate = 0;
    window.AudioContext = window.AudioContext || window.webkitAudioContext;
    this.audioContext = new AudioContext();

    // JavaScriptNode використовується для модифікації буфера виводу
    this.javaScriptNode = this.audioContext.createScriptProcessor(this.audioBufferSize, 1, 2);
    this.javaScriptNode.onaudioprocess = this.onAudioUpdate;
    this.javaScriptNode.eventHost = this;

    this.analyserNode = this.audioContext.createAnalyser();
    this.analyserNode.minDecibels = -100;
    this.analyserNode.maxDecibels = 0;
    this.analyserNode.smoothingTimeConstant = 0.0;
    this.analyserNode.connect(this.audioContext.destination);
    this.audioDataRef = undefined;

    // Інформація про плейбек
    this.playStart = 0;
    this.playEnd = 0;
    this.isLooped = false;
    this.currentPlayPosition = 0;
    this.isPlaying = false;

    // Інформація функції зворотнього виклику
    this.updateListener = [];
    this.playbackUpdateInterval = 0.0; // в секундах
    this.lastPlaybackUpdate = 0;

}
```