



Приватний заклад вищої освіти
Одеський технологічний університет «ШАГ»
Кафедра інформаційних технологій та фундаментальної
підготовки

Випускна кваліфікаційна робота бакалавра

**Інформаційна система управління ліцензованим аудіо контентом
(на прикладі Spotify)**

на здобуття бакалаврського рівня вищої освіти
зі спеціальності «122 Комп'ютерні науки»

Виконавці проєкту

№	П.І.Б.	Група
1	Дверницька О. В.	КН-П-211
2	Михайлюк М. О.	КН-Д-211
3	Небесна О. В.	КН-П-211
4	Рибак Д. О.	КН-П-211
5	Хіміч А. І.	КН-Д-211
6	Щербаков Д. Е.	КН-Д-211

Автор звіту

_____ Небесна Олександра Віталіївна

Науковий керівник	доц. Самойленко Д. М.
-------------------	-----------------------

АНОТАЦІЯ

УДК 004.9

Н-39

Небесна О. В. Кросплатформенна система для прослуховування музичного контенту: випускна кваліфікаційна робота на здобуття бакалаврського рівня вищої освіти зі спеціальності «122 Комп'ютерні науки». - Одеса: ОТУШ, 2025 - 17 с.

Робота присвячена дослідженню, розробці та впровадженню клієнт-серверної інформаційної системи для потокового прослуховування аудіоконтенту. Основним функціональним призначенням системи є забезпечення доступу користувачів до цифрової музики у режимі реального часу через вебінтерфейс або мобільний застосунок.

Актуальність проекту зумовлена глобальними тенденціями цифровізації медіаконтенту, стрімким зростанням аудиторії стримінгових сервісів, а також необхідністю створення адаптивних, безпечних і масштабованих мультимедійних платформ. Сучасні вимоги до зручності користування, швидкодії та персоналізації контенту обумовлюють потребу у впровадженні інноваційних технологій в архітектуру таких систем.

Об'єктом дослідження виступає багаторівнева інформаційна система для аудіострімінгу, що об'єднує різноманітні клієнтські середовища (веб- та мобільні пристрої) через єдиний програмний інтерфейс. Предмет дослідження — методи організації взаємодії між клієнтською та серверною частинами системи для забезпечення стабільної, захищеної та ефективної передачі медіаконтенту.

У якості задач дослідження було встановлене наступне:

- визначити оптимальні технології для реалізації клієнтської та серверної частин системи;
- розробити інтерфейси користувача з урахуванням принципів UI/UX-дизайну;
- реалізувати систему реєстрації, автентифікації, завантаження та прослуховування треків, управління плейлистами;
- впровадити механізми захисту персональних даних і авторизації через JWT-токени;
- провести тестування працездатності системи в різних клієнтських середовищах.

Методи дослідження включають системний аналіз, клієнт-серверне моделювання, а також аналіз користувацького досвіду (UI/UX).

Під час підготовки звіту використано 10 джерел – як друкованих, так і електронних.

ЗМІСТ

ВСТУП	3
Технічне завдання до проекту	4
РОЗДІЛ 1. Загальна характеристика системи	6
РОЗДІЛ 2. Реалізація клієнтської частини	7
РОЗДІЛ 3. Результати випробувань	11
ВИСНОВКИ	16
ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ	17

ВСТУП

У сучасному цифровому середовищі інформаційні технології стають основою більшості процесів передачі, збереження та обробки даних. Зростання популярності потокових сервісів відображає зміну підходів до споживання медіа контенту — користувачі очікують миттєвого доступу до аудіо та відео будь-де й будь-коли. За оцінками аналітичних агентств, частка аудіострімінгових сервісів у загальному медіаспоживанні продовжує стабільно зростати, стимулюючи розвиток відповідних програмних рішень. [1]

Паралельно з цим, активне використання мобільних пристроїв і вебтехнологій потребує створення систем з мультиплатформенною архітектурою, що забезпечує доступність сервісів на різних типах пристроїв — як у вебсередовищі, так і у вигляді нативних мобільних застосунків. У цьому контексті особливої актуальності набувають проекти, орієнтовані на зручний, швидкий і персоналізований доступ до мультимедійного контенту з дотриманням вимог захисту персональних даних та стабільної взаємодії з серверною частиною системи.

Дана випускна кваліфікаційна робота присвячена проектуванню та реалізації клієнт-серверної платформи для прослуховування аудіоконтенту. Особливістю даного проєкту є реалізація чотирьох взаємопов'язаних модулів: дизайн-прототипів (UI/UX), вебінтерфейсу (WebApp), мобільного застосунку (Android App) та серверної частини (Backend). Усі ці компоненти розроблялись як єдина система з чітко визначеними інтерфейсами взаємодії, що забезпечує узгоджену роботу незалежно від платформи доступу.

Детальне технічне завдання до проєкту наводиться далі. У рамках даного звіту розглядається клієнтська частина, що являє собою вебзастосунок. Звіт складається з трьох розділів, у яких описується загальна характеристика системи, деталі створення клієнтської частини та результати її розгортання та тестування. Висновки відбивають результати, що досягнуті у рамках даної частини проєкту, загальний висновок є сукупністю звітів усіх учасників проєкту.

ТЕХНІЧНЕ ЗАВДАННЯ ДО ПРОЄКТУ

Загальний опис

Програмне забезпечення (ПЗ) призначене для створення розподіленої інформаційної системи потокового відтворення музики. Основною функцією є надання користувачам доступу до аудіоконтенту в режимі онлайн, з можливістю керування відтворенням, створення і прослуховування плейлистів, альбомів, треків.

ПЗ підтримує розмежування користувачів за ролями: слухачі, виконавці (артисти) та адміністратори. Система має забезпечити швидкий доступ до контенту, персоналізацію інтерфейсу, а також безпечну взаємодію з обліковими даними користувачів.

Призначення

Система орієнтована на масового користувача, який бажає слухати музику онлайн без потреби в завантаженні файлів. Ключовими вимогами є зручність інтерфейсу, швидке реагування, підтримка мобільних і десктопних пристроїв, а також ефективна взаємодія з серверною частиною через API.

Вимоги до функціональності

Реєстрація та автентифікація:

- Автентифікація через логін та пароль.
- Підтвердження email при реєстрації.
- Авторизація та автентифікація за допомогою JWT (JSON Web Token).

Особистий кабінет користувача:

- Прослуховування треків, керування відтворенням (плей/ пауза/ гучність).
- Додавання треків у обране.
- Створення та керування плейлистами.
- Завантаження музики (доступно тільки для артистів).
- Перегляд та редагування персональних даних.

Пошук і навігація:

- Пошук за повною назвою треків.
- Добірки треків за категоріями (жанри/настрої).

Безпека:

- Захист паролів (хешування, шифрування).
- Перевірка прав доступу для кожного типу користувача.
- Обмеження дій залежно від ролі.

Архітектура

ПЗ розробляється за розподіленою клієнт-серверною архітектурою. До складу системи входять три незалежні, але взаємопов'язані компоненти:

- Backend - серверна частина, що обробляє запити, керує базою даних, виконує логіку аутентифікації і авторизації, зберігає/віддає медіафайли та забезпечує API для фронтендів.
- WebApp - вебзастосунок, що надає інтерфейс користувачам для взаємодії з контентом через браузер (доступний для слухачів та виконавців).
- MobileApp - додаток, що встановлюється на мобільних пристроях та дублює ключову функціональність WebApp.

Технології

Зберігання вихідних кодів ПЗ повинно здійснюватись за допомогою системи контролю версій Git, розміщеного на платформі GitHub. Структура файлів та каталогів повинна відображати архітектуру ПЗ, на зразок:

- Backend
- Web
- Mobile

Для кожного з модулів рекомендується (з можливістю часткової зміни) наступні технології створення:

- Backend - MySQL, Spring, Swagger, Hibernate, Azure
- Web - React, React Router, TS, HTML, CSS
- Mobile - React Native, Expo, TS

РОЗДІЛ 1

ЗАГАЛЬНА ХАРАКТЕРИСТИКА СИСТЕМИ

У відповідності до технічного завдання (ТЗ) система потокового прослуховування музики була спроектована за клієнт-серверною розподіленою архітектурою. У складі системи було умовно виділено чотири архітектурні частини:

- «Mockup»,
- «Backend»,
- «Web Frontend»,
- «Mobile Frontend».

Частина «Mockup» являє собою набір графічних інтерфейсів користувача та сценаріїв взаємодії з системою. Вона створена з використанням сервісу Figma та охоплює як мобільну, так і вебверсії. В макетах реалізовані принципи UI/UX-дизайну, зокрема логічна навігація між екранами, врахування адаптивності та ергономіки мобільних пристроїв, інтуїтивно зрозуміле керування відтворенням. Макети доступні за посиланням: <https://www.figma.com/design/yDB4Ci2Lk2QmXCWBIRGN68/lumitune>.

Частина «Backend» реалізована як централізований вузол збереження та оброблення даних, що забезпечує логіку системи. Вона включає у себе базу даних користувачів, плейлистів, треків, альбомів та програмний інтерфейс (API), що реалізує автентифікацію, авторизацію, логіку управління обліковими записами та контентом, обробку запитів із клієнтських додатків (Web і Mobile).

Частина «Backend» виконана на базі програмного забезпечення Spring та СУБД «MySQL». Додатково використані програмні пакети «Hibernate», «Azure Spring», «mp3agic». API орієнтований на REST-архітектуру. Дані передаються у форматі JSON.

Клієнтська частина системи «Web Frontend» надає інтерфейс для взаємодії користувача з функціональністю системи через браузер. Вихідні коди та ресурси даної частини розміщені на репозиторії <https://github.com/nebesnayaa/lumitune-frontend>.

Клієнтська частина «Mobile Frontend» реалізована з використанням React Native, що дозволяє розгортати додаток на платформах Android. Мобільна версія надає: повноцінний музичний плеєр, всі функції з вебверсії (авторизація, медiateка). Репозиторій: <https://github.com/magnoliophyta12/lumitune-mobile>.

Задля захисту даних у клієнт-серверній взаємодії використано шифрування з використанням алгоритму AES-128. Всі запити з мобільного або вебдодатку до бекенду проходять через HTTPS із токеном авторизації (JWT). Таким чином, забезпечується:

- Захист персональних даних;
- Можливість гнучкого контролю доступу;
- Безпека в публічних Wi-Fi мережах.

РОЗДІЛ 2

РЕАЛІЗАЦІЯ КЛІЄНТСЬКОЇ ЧАСТИНИ

Клієнтська частина інформаційної системи реалізована з використанням наступних технологій:

- Засоби розроблення: мова програмування TypeScript, середовище Visual Studio Code, інструмент для збірки Vite;
- Бібліотеки: React, React Router, Axios;
- Система контролю версій: Git;
- Деплой: Vercel.

У складі клієнтської частини (вебсайту) передбачено та реалізовано наступні функції:

- Авторизація користувача, реєстрація нових користувачів;
- Керування особистими даними (кабінетом);
- Прослуховування аудіотреків;
- Створення та керування плейлистами;
- Завантаження треків та альбомів.

Початок роботи користувача із додатком супроводжується його реєстрацією як учасника інформаційної системи. При реєстрації користувач обов'язково надає відомості про електронну пошту, через яку буде здійснюватись верифікація профілю користувача. Наступний крок - створення паролю, що має відповідати мінімальним вимогам:

```
const hasLetter = /[A-Za-zА-Яа-яЁёїİіЇіЄєЎў]/.test(password);
const hasDigitOrSpecial = /[0-9_!?&#]/.test(password);
const hasMinLength = password.length >= 8;
const isValid = hasLetter && hasDigitOrSpecial && hasMinLength;
```

Також попередньо до реєстрації введений нікнейм проходить перевірку на унікальність за допомогою запиту на сервер та дата народження проходить валідацію на коректність дати та мінімальний вік:

```
const dateObj = new Date(formattedDate);
const isValidDate =
  dateObj.getFullYear() === Number(birthYear) &&
  dateObj.getMonth() + 1 === Number(birthMonth) &&
  dateObj.getDate() === Number(birthDay);

const currentYear = new Date().getFullYear();
const minYear = 1900;
const maxYear = currentYear - 10;
const isValidYear = Number(birthYear) >= minYear && Number(birthYear) <= maxYear;
```

Авторизація користувача здійснюється за допомогою введення унікального нікнейму та паролю.

При реєстрації чи авторизації здійснюється комунікація з частиною «Backend» у результаті чого авторизація підтверджується або спростовується. Пароль при передачі між зазначеними частинами шифрується за допомогою алгоритму AES-128 ECB та перетворюється у транспортне кодування Base64. Це забезпечує чутливі дані при передаванні. [2]

Засоби шифрування реалізовані та винесені в окремий файл для повторного використання у різних частинах системи:

```
export const encrypt = (plainText: string): string => {
  const encrypted = CryptoJS.AES.encrypt(plainText, key, {
    mode: CryptoJS.mode.ECB,
    padding: CryptoJS.pad.Pkcs7
  });
  return encrypted.ciphertext.toString(CryptoJS.enc.Base64);
};

export const decrypt = (cipherBase64: string): string => {
  const encryptedHex = CryptoJS.enc.Base64.parse(cipherBase64);
  const cipherParams = CryptoJS.lib.CipherParams.create({
    ciphertext: encryptedHex
  });
  const decrypted = CryptoJS.AES.decrypt(cipherParams, key, {
    mode: CryptoJS.mode.ECB,
    padding: CryptoJS.pad.Pkcs7
  });
  return decrypted.toString(CryptoJS.enc.Utf8);
};
```

Обмін даними з серверною частиною реалізовано за допомогою axios-запитів. У якості прикладу далі наведені запити на отримання та відправлення даних методами GET та POST, що широко вживається у різних частинах проекту:

```
const axiosInstance = axios.create({
  baseURL: "https://lumituneapp.azurewebsites.net/",
  headers: {
    "Content-Type": "application/json",
  },
  withCredentials: true,
});
```

```
export const getCurrentUser = async () => {
  try {
    const response = await axiosInstance.get(`/users/current`);
    return response.data;
  } catch (error) {
    console.error("Помилка при отриманні поточного користувача:", error);
    throw error;
  }
};
```

```
export const createPlaylist = async(name: string) => {
  try {
    const response = await axiosInstance.post("/playlists/", name);
    return response.data;
  } catch (error) {
    console.error("Помилка при створенні плейлиста:", error);
    return null;
  }
}
```

Основну частину взаємодії користувача із додатком реалізовано за допомогою роботи з аудіоконтентом. Для цього розроблено компоненти для відображення треків, альбомів та плейлистів, які легко імплементуються на основні сторінки (головну, сторінку плейлиста тощо). Приклад виведення треків у вигляді карток наведено нижче:

[illegible]

```

        <span className={` ${styles.scrollText}`}>
          {song.name}
        </span>
      )}
    </div>
  </div>
  <p className={styles.authorName}>{artists[song.artistId] || "Author"}</p>
</div>
)}}

```

З метою покращення взаємодії користувача з інтерфейсом сайту, було реалізовано кастомний React-хук `useDragScroll`, який забезпечує інтуїтивну горизонтальну прокрутку елементів за допомогою натискання та перетягування миші. Його використано, зокрема, в компонентах, де відображаються плейлисти, альбоми у вигляді каруселі.

Основна ідея полягає в додаванні до DOM-елементу обробників подій `mousedown`, `mousemove`, `mouseup`, які імітують поведінку "свайпу". Користувач отримує можливість прокручувати вміст, натискаючи та тягнучи вліво або вправо, як у звичних мобільних або desktop застосунках, що позитивно впливає на UX. [3]

Як вже зазначалось вище, з повним кодом проєкту можна ознайомитись у репозиторії <https://github.com/nebesnayaa/lumitune-frontend>

РОЗДІЛ 3

РЕЗУЛЬТАТИ ВИПРОБУВАНЬ

Результатом розробки клієнтської частини стало створення повнофункціонального вебзастосунку, який забезпечує доступ до аудіострімінгової платформи через браузер.

Головне тестування інтерфейсів і функціональності проводилось у браузерах Google Chrome, Opera та Firefox на ОС Windows 11.

Початок роботи нового користувача з додатком починається з реєстрації користувача у системі. Реєстрація стратегічно поділена на три кроки - введення електронної пошти, створення паролю та заповнення особистих даних - для запобігання перевантаження користувача всіма полями одразу. Після реєстрації на електронну пошту надсилається лист з посиланням, перейшовши за яким система верифікує пошту користувача.

Як тільки користувач завершив етап реєстрації, він може увійти до системи за допомогою нікнейму та паролю. Зовнішній вигляд екранів реєстрації та входу наведено на рис. 1-2.

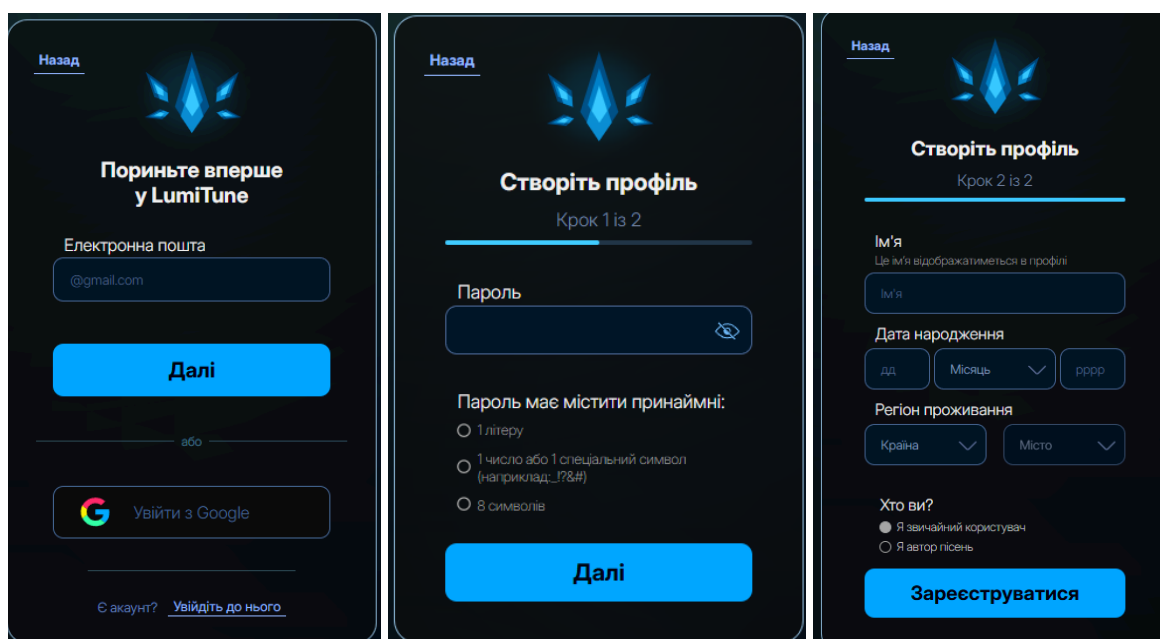
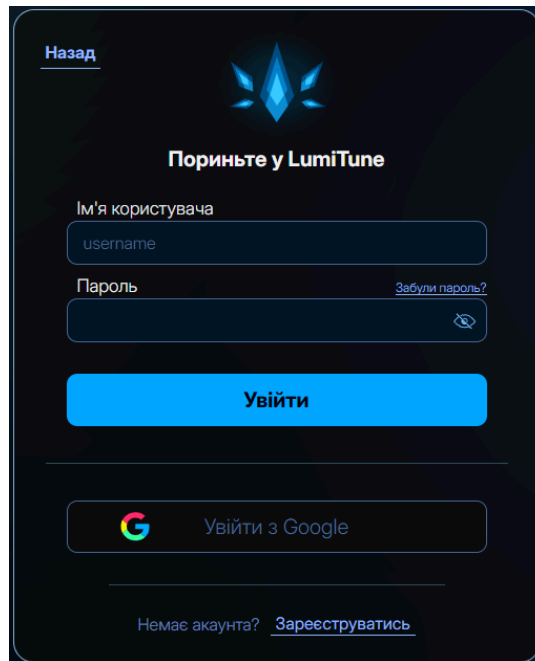


Рисунок 1 - Покрокова реєстрація користувача



The image shows a login interface for 'LumiTune'. At the top left is a 'Назад' (Back) link. In the center is a blue diamond-shaped logo. Below the logo is the title 'Пориньте у LumiTune'. There are two input fields: 'Ім'я користувача' (Username) with the placeholder 'username' and 'Пароль' (Password). To the right of the password field is a 'Забули пароль?' (Forgot password?) link and an eye icon for toggling visibility. Below the fields is a large blue 'Увійти' (Login) button. A horizontal line separates this from a 'Увійти з Google' (Login with Google) button, which features the Google 'G' logo. At the bottom, there is a link 'Немає акаунта? Зареєструватись' (Don't have an account? Register).

Рисунок 2 - Вхід у систему

Після входу до системи користувач отримує доступ до основного контенту. Насамперед, це головна сторінка з музичними підбірками за жанрами/настроєм та рекомендаціями. Перейшовши за настроєм або жанром користувач потрапляє на сторінку з підбіркою треків. На рисунку 3 наведено екрани головної сторінки та підбірки треків.

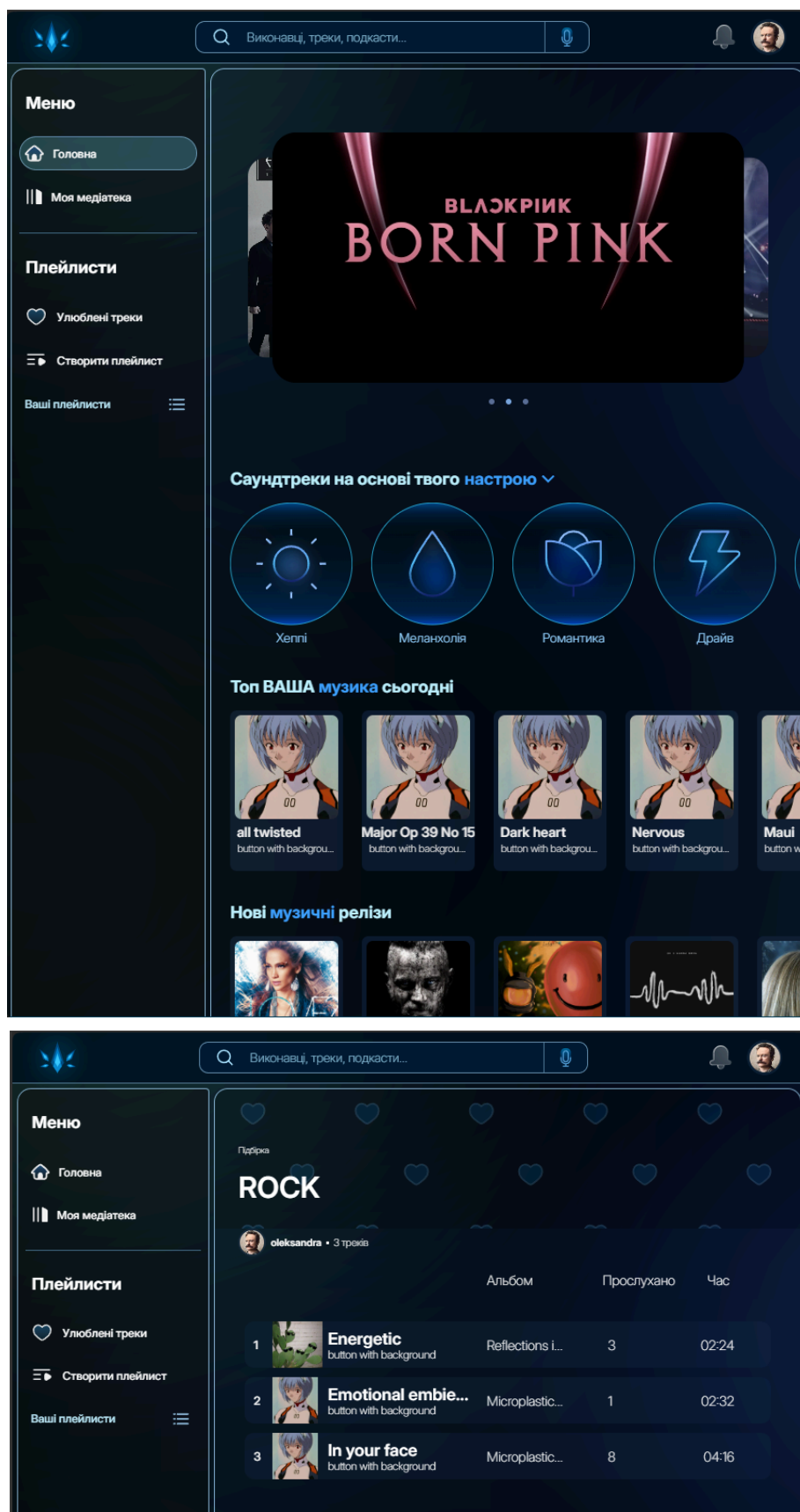


Рисунок 3 - Головна сторінка, підбірка треків за жанром

Натиснувши на трек, запускається аудіоплеєр, розміщений в нижній частині екрану. Зліва розміщено інформацію про трек (обкладинка альбому, назва треку, виконавець) та кнопка Вподобання. По центру відображено елементи керування (пауза/плей) та строка програвання треку з таймером. Справа розміщено слайдер для керування гучністю треку. Переглянути плеєр можна на рис. 4.

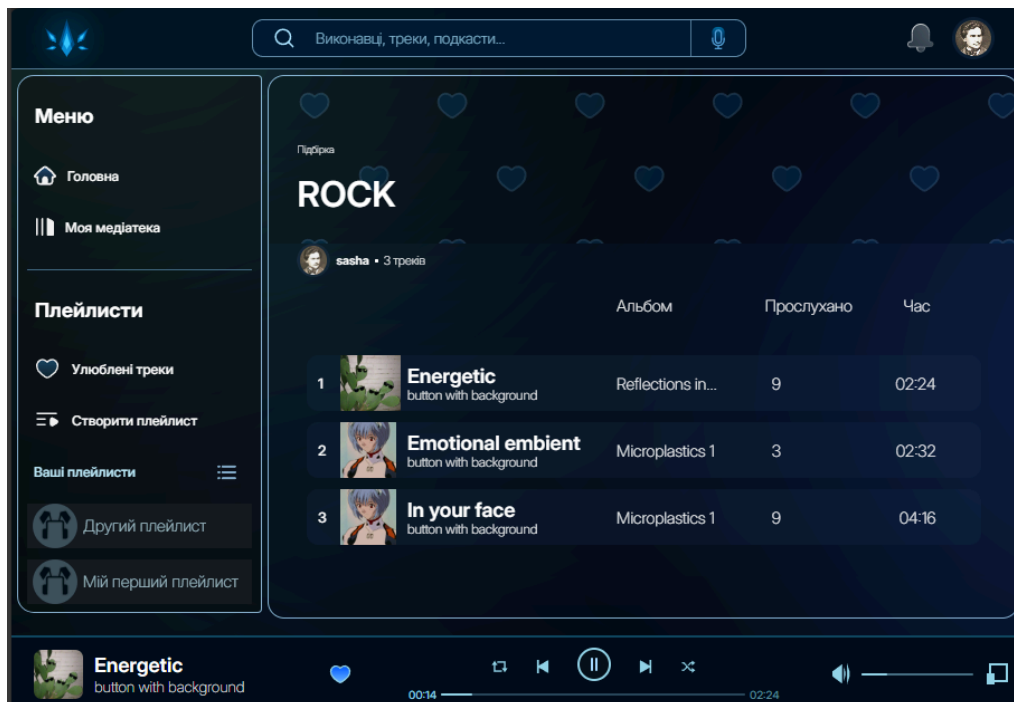


Рисунок 4 - Аудіоплеєр

Кабінет користувача надає персоналізований інтерфейс в залежності від того, чи є користувач виконавцем.

Звичайний користувач має змогу переглянути свої статистичні дані, такі як кількість підписок/підписників, редагувати свій нікнейм та завантажити нове фото профілю.

Виконавцю доступні такі ж функціональності, але він додатково ще бачить кількість своїх музичних альбомів та має опис профілю, який можна відредагувати разом з іншими даними.

Приклад взаємодії з особистим профілем наведено на рис. 5.

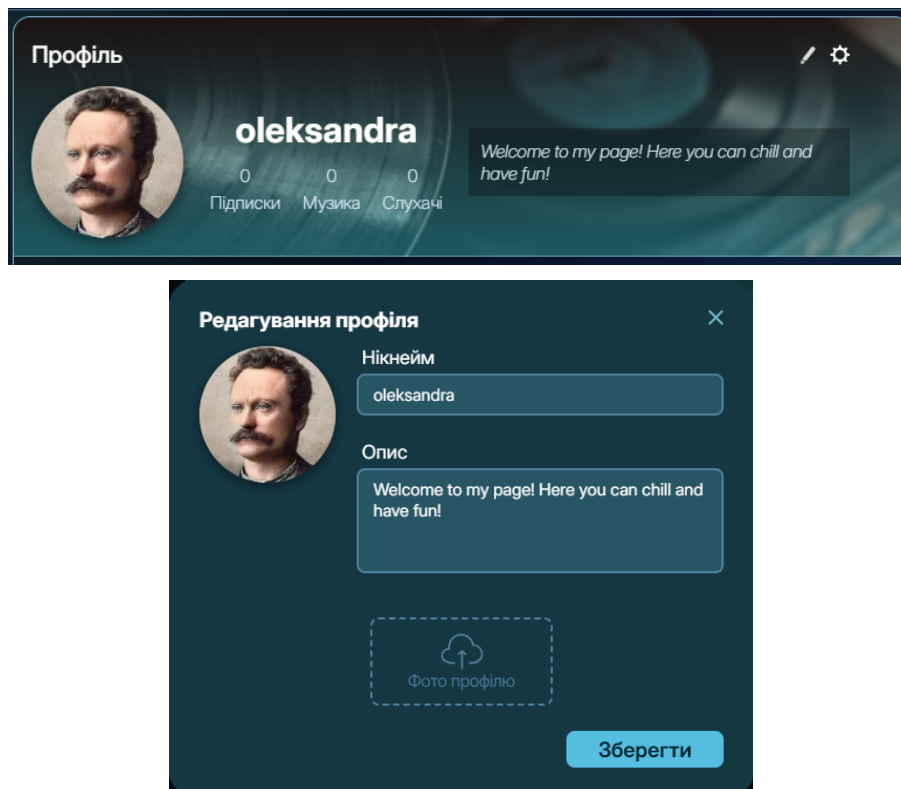


Рисунок 5 - Особистий профіль користувача, редагування особистих даних

Навігація на сайті здійснюється завдяки бічному меню, переглянути яке можна на рис. 6. Тут розміщені списком наявні плейлисти користувача, реалізовано можливість створити новий плейлист та переглянути улюблені треки користувача.

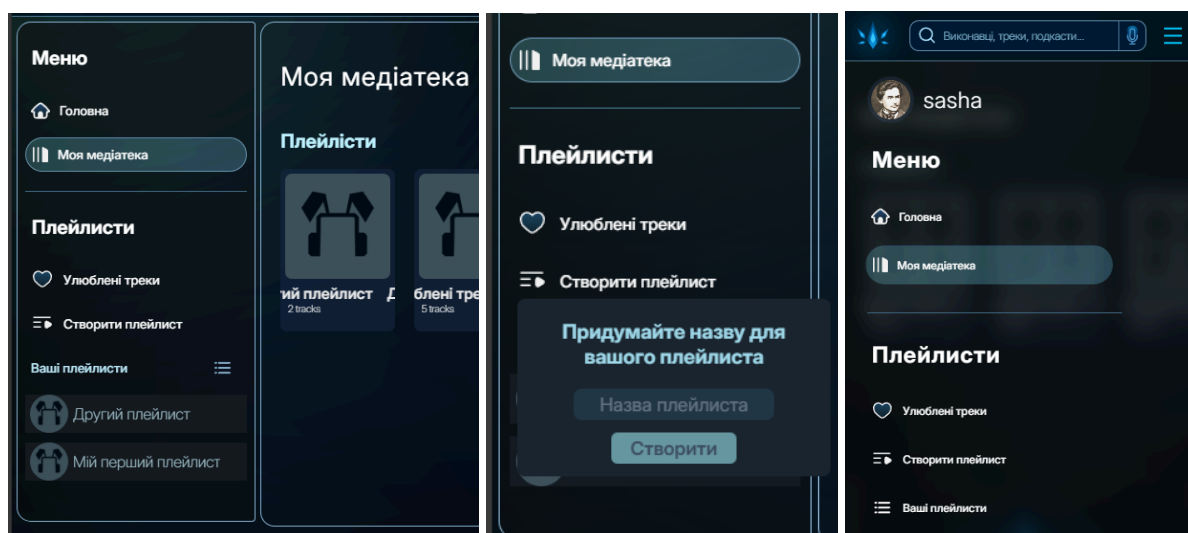


Рисунок 6 - Бічне меню, створення нового плейлиста, адаптивне меню для мобільних пристроїв

У підсумку, всі функції клієнтської частини, що були заявлені технічним завданням було реалізовано та випробувано.

ВИСНОВКИ

Проведено огляд сучасних інформаційних систем, орієнтованих на онлайн-стрімінг мультимедійного контенту. Встановлено, що значну популярність мають сервіси для прослуховування музики, які надають зручний доступ до аудіофайлів у режимі реального часу та функціональну інтеграцію з акаунтами користувачів. У якості прикладу для реалізації обрано систему потокового відтворення музики — аналог популярного сервісу Spotify.

Визначено набір технологій та програмних засобів для реалізації системи: REST API для обміну даними між клієнтом і сервером, механізми авторизації та автентифікації на основі JWT, засоби керування станом інтерфейсу та взаємодії з медіа-контентом у вебсередовищі.

Спроектовано, реалізовано та протестовано розподілену інформаційну систему, що складається з чотирьох основних компонентів, розміщених за адресами:

- «Mockup»
<https://www.figma.com/design/yDB4Cl2Lk2QmXCWBIRGN68/lumitune>,
- «Backend» <https://github.com/Sheogorattio/lumitunespring>,
- «Frontend Web» <https://github.com/nebesnayaa/lumitune-frontend>,
- «Frontend Mobile» <https://github.com/magnoliophyta12/lumitune-mobile>.

Особливу увагу приділено захисту персональних даних користувачів: реалізовано систему реєстрації з підтвердженням електронної пошти, впроваджено шифрування JWT-токенів, що дозволяє забезпечити безпечну взаємодію клієнта з сервером. Архітектура системи дозволяє у майбутньому легко інтегрувати двофакторну аутентифікацію.

Проведено випробування системи на різних пристроях, включаючи десктопні браузерери та мобільні екрани. Тестування підтвердило коректність роботи програвача, підтримку перемикання треків, стабільну роботу при зміні стану програвача та відповідність інтерфейсу сучасним вимогам до UX/UI. Система показала високу стабільність і функціональну відповідність технічному завданню.

ПЕРЕЛІК ВИКОРИСТАНИХ ДЖЕРЕЛ

1. Deloitte Insights. Consumer audio market trends predict more global consumers in 2024. – [Електронний ресурс]. – URL: <https://www.deloitte.com/.../consumer-audio-market-trends-predict-more-global-consumers-in-2024.html>. – Дата звернення: 13.04.2025.
2. Stack Overflow. CryptoJS javascript AES-128, ECB encrypt / decrypt. – [Електронний ресурс]. – URL: <https://stackoverflow.com/questions/72570359/cryptojs-javascript-aes-128-ecb-encrypt-decrypt>. – Дата звернення: 20.05.2025.
3. MDN Web Docs. MouseEvent – Web APIs. – [Електронний ресурс]. – URL: <https://developer.mozilla.org/en-US/docs/Web/API/MouseEvent>. – Дата звернення: 28.05.2025.
4. Барановська Т. П., Ковальчук О. І. Розробка веб-застосунків з використанням REST API. // Інформаційні технології та комп'ютерна інженерія. – 2021. – № 3(58). – С. 45–52.
5. Тарасов С. А., Беспалов Д. А. Побудова мультимедійних інформаційних систем на основі потокового передавання даних. // Системи обробки інформації. – 2020. – Вип. 2(161). – С. 56–61.
6. G. Banks, H. Thomas, “Learn React with TypeScript”. — 2-е вид., Packt Publishing, 2023. – 474 с.
7. Мартін Краузе. The Complete Developer: Master the Full Stack with TypeScript, React, Next.js, MongoDB, and Docker. — No Starch Press, 2024. — 336 с. — ISBN 978-1718503281.
8. Meta et al. React Documentation. Версія 19.1. – [Електронний ресурс]. – URL: <https://react.dev/>. – Дата звернення: 14.05.2025.
9. Shopify et al. React Router Official Documentation (version 7). – [Електронний ресурс]. – URL: <https://reactrouter.com/>. – Дата звернення: 15.05.2025.
10. Чеботаренко А. В. Назва дипломної роботи: Мобільний додаток для передачі та синхронного відтворення аудіоінформації. – 2021 р. – [Електронний ресурс]. – URL: <https://ela.kpi.ua/server/api/core/bitstreams/e2e5b8c0-8768-44b6-9b1e-ef66992e419b/content> – Дата звернення: 21.05.2025.