0МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
СУМСЬКИЙ ДЕРЖАВНИЙ УНІВЕРСИТЕТ
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«До захисту допущено»
В.о. завідувача кафедри
_____ Ігор ШЕЛЕХОВ
(підпис)
01 червня 2024 р.

КВАЛІФІКАЦІЙНА РОБОТА
на здобуття освітнього ступеня бакалавр

зі спеціальності 122 – Комп'ютерних наук,
освітньо-професійної програми «Інформатика»
на тему: «Android-додаток для популяризації музичної групи»
здобувача групи ІН-03 Казьміна Данила Олеговича

Кваліфікаційна робота містить результати власних досліджень. Використання ідей, результатів і текстів інших авторів мають посилання на відповідне джерело.

_____ Данило КАЗЬМІН
(підпис)

Керівник,
старший викладач кафедри
комп'ютерних наук,
Кандидат технічних наук Артем КОРОБОВ _____
(підпис)

Суми – 2024

Сумський державний університет
Факультет електроніки та інформаційних технологій
Кафедра комп'ютерних наук

«Затверджую»
В.о. завідувача кафедри
_ _____ Ігор ШЕЛЕХОВ
(підпис)

ІНДИВІДУАЛЬНЕ ЗАВДАННЯ НА КВАЛІФІКАЦІЙНУ РОБОТУ
на здобуття освітнього ступеня бакалавра
зі спеціальності 122 - Комп'ютерних наук, освітньо-професійної програми «Інформатика»
здобувача групи ІН-03 Казьміна Данила Олеговича

1. Тема роботи: «Android-додаток для популяризації музичної групи»
   затверджую наказом по СумДУ від *«22» квітня 2024 р. № 0414-VI*
2. Термін здачі здобувачем кваліфікаційної роботи *до 4 червня 2024 року*
3. Вхідні дані до кваліфікаційної роботи _____
4. Зміст розрахунково-пояснювальної записки (перелік питань, що їх належить розробити) *1) Аналіз актуальності розробки додатку для музичного гурту, конкурентоспроможності та цільової аудиторії, постановка й формування завдань. 2) Проектування системи, розробка дизайну, розробка функціоналу. 3) Огляд та вибір засобів програмної реалізації 4) Програмна реалізація системи додатку для популяризації музичного гурту. 5) Аналіз результатів.*
5. Перелік графічного матеріалу (з точним зазначенням обов'язкових креслень) _____
6. Консультанти до проекту (роботи), із зазначенням розділів проекту, що стосується їх

Розділ	Консультант	Підпис, дата	
		Завдання видав	Завдання прийняв

7. Дата видачі завдання «__» _____ 20__ р.

Завдання прийняв на виконання _____ Керівник _____
(підпис) (підпис)

КАЛЕНДАРНИЙ ПЛАН

№ п/п	Назва етапів кваліфікаційної роботи	Термін виконання	Примітка
1	*Аналіз проблеми предметної області, конкурентоспроможності, постановка й формування завдань дослідження*	10.04.2024 - 12.04.2024	
2	*Проектування системи, розробка дизайну.*	13.04.2024 – 21.04.2024	
3	*Огляд та вибір інструментів програмної реалізації*	21.04.2024 – 24.04.2024	
4	*Розробка інформаційної системи додатку*	25.05.2024 - 04.06.2024	
5	*Аналіз отриманих результатів*	04.06.2024 – 04.06.2024	
6	*Оформлення пояснювальної записки до кваліфікаційної роботи*	30.05.2024 – 04.06.2024	

Здобувач вищої освіти _____ Керівник _____
(підпис) (підпис)

АНОТАЦІЯ

Записка: 68 стор., 18 рис., 0 табл., 2 додатки, 9 використаних джерел.

Обґрунтування актуальності теми роботи – розробка додатків для музичних гуртів є завжди актуальною з технологічної, соціальної, культурної та економічної точок зору. Це можна підтвердити якщо слідкувати за сучасними тенденціями на ринку музики, попитом серед слухачів та фанатів, та необхідністю збільшувати популярність та обсяги продажів товару та квитків гуртам. Гурту потрібен додаток, який підвищить продажі та популярність та буде зручний, зрозумілий та стабільний у функціонуванні для клієнта, це робить тему створення такого додатку вартою уваги.

Об’єкт дослідження -Android додаток для для популяризації музичного гурту, який дозволить користувачу обрати товари з каталогу, купити квитки на потрібний концерт та послухати пісні гурту без потреби завчасно завантажувати їх у пам’ять телефону.

Мета роботи – дослідження, аналіз та розробка Android додатоку для популяризації музичного гурту, який дозволить користувачу обрати товари з каталогу, купити квитки на потрібний концерт та послухати пісні гурту без потреби завчасно завантажувати їх у пам’ять телефону..

Методи дослідження - аналіз існуючих додатків у інших гуртів, інструменти розробки та проектування.

Результати - розроблено застосунок, який стабільно та ефективно надає користувачу всі необхідні функції: прослуховування пісень гурту, які відсортовані по альбомам, великий каталог товарів на будь-який смак, список концертів гурту з можливістю купівлі квитків. .

ДОДАТОК ДЛЯ ПОПУЛЯРІЗАЦІЇ МУЗИЧНОГО ГУРТУ, АДАПТИВНИЙ ДИЗАЙН, ІНФОРМАЦІЙНА СИСТЕМА, KOTLIN, XML

ЗМІСТ

Вступ

Музичні гурти мають безперечно один з найбільших впливів у формуванні культурного образу суспільства, кожен гурт намагається збільшити свою популярність та вплив на музичний жанр в якому вони діють. Найбільший вплив на популяризацію звичайно вказують продюсери, пісні, концерти та стиль гурту, проте пісні та концерти це перш за все інструмент підвищення популярності, а найголовніший інструмент заробітку грошей це продаж квитків на концерти, сувенірних товарів та іншого мерчу гурту.

Кожен гурт намагається зробити для себе свій веб-сайт для реалізації квитків та іншої супутньої продукції, проте останнім часом зростає популярність саме мобільних додатків, тому що багатьом людям зручніше мати свій акаунт у додатку, завжди мати змогу дізнатися останні новини, передивитися новинки в продукції та швидко купити квитки. Загалом на мобільних пристроях додатки більш зручніше ніж сайти, тому популярність їх зростає. Нижче я прописав переваги додатків над веб сайтами:

Зручність використання на мобільних пристроях: Мобільні додатки оптимізовані для роботи на мобільних пристроях, що робить їх більш зручними для користувачів, які використовують смартфони та планшети.

Доступ до функцій пристрою: Деякі функції мобільного пристрою, такі як камера, геолокація, сповіщення тощо, можуть бути легко інтегровані в мобільні додатки, що розширює їх можливості.

Офлайн доступ і робота в режимі офлайн: Багато мобільних додатків дозволяють користувачам використовувати їх і в режимі офлайн, що дозволяє продовжувати роботу з додатком навіть без Інтернет-з'єднання.

Зручність взаємодії з пристроями: Завдяки можливості використовувати жести, сенсорний екран та інші функції пристроїв, мобільні додатки можуть забезпечувати більш інтуїтивний та приємний досвід користувача.

Удосконалена безпека: Мобільні додатки можуть забезпечувати більш високий рівень безпеки, наприклад, за допомогою аутентифікації за відбитком пальця або обличчя, що додає додатковий захист для даних користувачів.

Мета роботи

Робота присвячена розробці функціонального додатку для музичної гурту за наступними фунціями:

- Функція магазину мерчу гурту Asking Alexandria
- Функція відображення альбомів, пісень та синглів гурту.
- Функція плеєру за допомогою якого можна слухати пісні групи.
- Можливість дізнатися останні новини пов’язані з гуртом, дізнатися розклад концертів та купити квитки.

У звіті я розгляну основні етапи розробки додатку. По-перше я проведу аналітичний огляд для визначення вимог до програми, згідно з вимогами я визначу об’єм роботи та час який треба витратити на відповідні етапи роботи, також це дасть змогу проаналізувати конкурентоспроможність мого додатку. По-друге я розгляну поетапне проектування застосунку, включаючи дизайн інтерфейсу всіх частин додатку та саму структуру програми. І у кінцевій частині звіту я розгляну програмне та інформаційне забезпечення, які включають в себе вибір технологій, середовище розробки та опис програмної реалізації.

При розробці функціонального додатку для музичного гурту я буду враховувати наступні аспекти:

- **Огляд та аналіз:** на даному етапі я визначу основні вимоги до програми, враховуючи потреби користувачів та цільову аудиторію. Також я проведу аналіз конкурентоспроможності, який допоможе мені як розробнику визначити особливості, які будуть виділяти додаток серед застосунків інших гуртів.

- **Проектування системи:** Цей аспект є ключовим та критично важливим для створення додатку. Я розроблю інтерфейс для вікон завантаження, головного меню, вікна магазину, вікна альбомів та списку пісень та багато інших вікон які будуть корисними для користувачів. Також я створю діаграму класів, яка буде служити відображенням структури додатку.

- **Програмне та інформаційне забезпечення системи:** на даному етапі я доповім про те які саме технології та інструменти розробки біли обрані мною для створення додатку. Я опишу програмну реалізацію, буду акцентувати увагу на важливі деталі, які забезпечують стабільне функціонування додатку.

Огляд та аналіз

1.1 Етапи створення музичного додатку

Взагалі мобільні додатки відіграють важливу роль у суспільстві, вони надають користувачам інтуїтивний та зручний інтерфейс. В свою чергу гурт який окрім сайту має мобільний додаток здатен охопити велику кількість шанувальників саме програм, які можна інсталювати та завжди мати під рукою стабільний, інтуїтивний та зручний продукт. Такі додатки для гурту стануть інструментом заробітку грошей через продаж квитків на концерти, сувенірних товарів та мерчу. І враховуючи вище наведене, при розробці додатку я буду керуватися наступними етапами:

- **Визначення ідеї та планування –** це перше, що треба зробити. Визначення тематики додатку це найважливіший етап, адже додатки можуть бути різні, особливо коли ми говоримо та думаємо створити додаток який буде корисний, як і користувачу так і власнику. Планування
  графіку, визначення дедлайнів, та створення покрокового плану розробки це один із перших кроків в створенні додатку.

- **Проектування –** наступний але також дуже важливий крок у створенні застосунку. Саме під час цього етапу формується розуміння та ніби у

будівництві ми закладаємо фундамент на якому буде створюватись проект. Тут вирішується зовнішній вигляд нашого майбутнього додатку, і логіка його роботи. Тож ми повинні розробити дизайн застосунку, його архітектуру та технічні особливості та характеристики.

- **Розробка** – ключовий етап створення мобільного застосунку. Тут ми втілюємо наші плани та ідеї у життя за допомогою програмного коду, середовищ розробки та багатьох інших інструментів. За допомогою всіх наших знань та бажанню вивчати щось нове для себе, наш додаток буде забезпечений необхідним функціоналом.

  Тестування – етап, який приховує у собі одне із найважливіших завдань для створення мобільного додатку, саме під час тестування ми визначаємо наскільки наш продукт готовий до публікації та зустрічі з майбутніми користувачами. На цьому етапі ми виправляємо помилки та баги, тестуємо якість роботи додатку і всіх його функцій.

- **Запуск та публікація** – майже останні етап створення застосунку. Якщо наш додаток пройшов всі етапи, і тестування показало позитивний результат, і що додаток готовий до випуску, то саме час зробити застосунок доступним для широкої аудиторії користувачів.

- **Оновлення та підтримка життєдіяльності** – кінцевий етап створення додатку, хоча я назвав цей етап кінцевим, він буде продовжуватись під час всього життя нашого застосунку. Щоб наш додаток був завжди актуальним, ми повинні виправляти помилки, додавати новий функціонал, розробляти оновлення та налагодити зворотній зв’язок та підтримку користувачів.

1.2 Аналіз вимог

Мобільні додатки це невід’ємна частина нашого життя, спираючись на наші потреби, бажання та попит користувачів вони пропонують рішення, щоб зробити наше з вами життя комфортним. Додатків тисячі, від служб доставки та таксі до державних та фінансових сервісів, від мобільних ігор та соціальних мереж до освітніх платформ та рішеннями для бізнесу. Клієнтів та користувачів завжди приваблює зручність та інтуїтивний

інтерфейс мобільних додатків. Вони обожнюють застосунки, які пропонують стабільність роботи, швидкий відгук на натискання кнопки, мінімалістичний та при цьому позбавлений інформаційного голодування інтерфейс, персоналізацію та безпеку.

Отже чітко визначимо вимоги до додатку для популяризації музичної групи:

Інтерфейс – додаток повинен мати зрозумілий та інтуїтивний інтерфейс, для будь-яких користувачів. Він повинен забезпечити легку навігацію по застосунку, також контрастність та правильне позиціонування елементів для зручної взаємодії. Весь текст буде відображений англійською мовою, адже вона є міжнародною.

Онлайн магазин – оскільки основною задачею додатку є просування та продаж продукції гурту, має бути створена модель онлайн магазину за допомогою якого користувачі зможуть обрати собі необхідні товари. Магазин буде поділений на декілька вікон такі як:

- **Shop** – головне вікно магазину у якому будуть відображатись деякі товари із каталогу продукції гурту.
- **Sale –** вікно у якому користувач може побачити товари які потрапили під розпродаж товарів, тобто товари по знижкам.
- **What`s new** – вікно у якому користувачі можуть побачити новинки і каталозі товарів музичного гурту.

Також користувач повинен мати можливість відкрити сторінку з конкретним товаром, щоб переглянути опис товару, його зовнішній вигляд, ціну та розміри якщо це одяг.

Список альбомів та пісень – користувач повинен мати можливість переглянути список усіх альбомів гурту та список пісень у кожному з них.

Список місць проведень концертів гурту – дане вікно повинно містити у собі список актуальних концертів гурту. Повинно бути зазначено дату проведення та місце, також користувач натиснувши кнопку зможе купити квитки

Музичний плеєр – користувач повинен мати змогу прослуховувати пісні гурту, на додачу користувачу не обов’язково завантажувати пісні у пам’ять телефону, додаток буде працювати як стримінговий сервіс

Вікно налаштувань – у цьому вікні користувач зможе знайти актуальні посилання на сайт гурту, на сервіси Apple Music та Spotify та вивчити політику конфіденційності та знайти контактні данні менеджерів гурту.

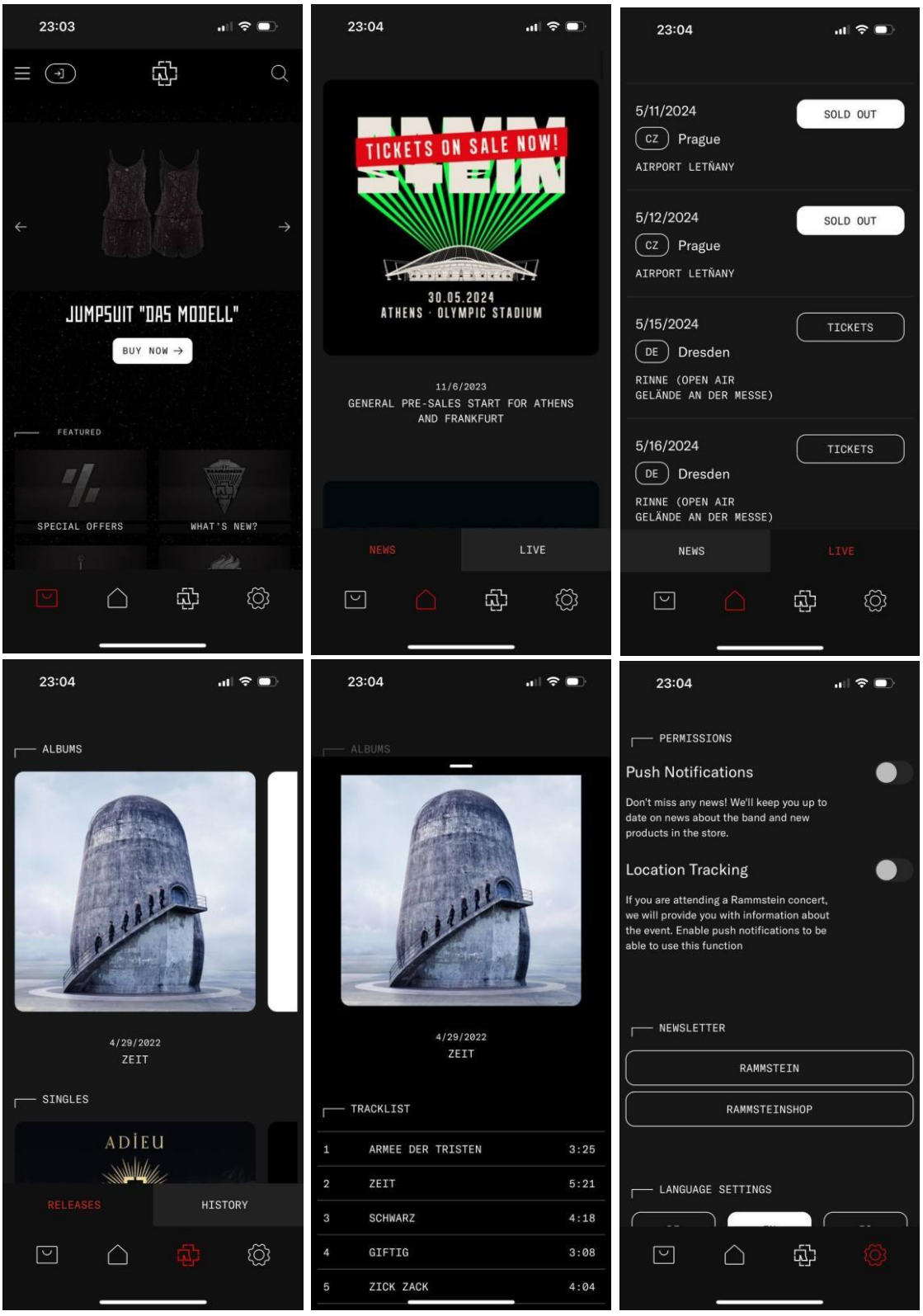
Ці пункти допоможуть зрозуміти вимоги та аспекти які потрібно дотримати та реалізувати для того щоб додаток був функціональним, безпечним, зручним та привабливим для користувачів, даруючи їм приємний досвід знайомства та взаємодію з гуртом

1.3 Аналіз конкурентоспроможності

Аналіз та розуміння конкурентоспроможності є важливим елементом при створенні та подальшого успіху будь-якого додатку чи сервісу. Даний аналіз дозволяє розробникам зрозуміти ринок, попит, визначити сильні та слабкі сторони конкурентів, а також виявити потенційні можливості для розвитку свого продукту. Нажаль матеріалу для такого аналізу виявилося небагато так як у мого майбутнього застосунку тільки один конкурент. Це мобільний додаток гурту Rammstein. Тож проаналізуємо те що є:

Rammstein – один з найпопулярніших гуртів у світі, який вказав великий вплив на розвиток музики. У них є фізичні магазини в яких вони реалізують свій товар, і також у них є свій веб-сайт. Проте в 2021 році у них з’явився свій мобільний додаток, і з часом він приніс гурту великий ріст продажів товарів та квитків на концерти. Дивлячись на досвід німецького гурту можна зрозуміти наскільки популярні мобільні застосунки. Тож розберемо зовнішній вигляд, функціонал, переваги та недоліки додатку німецького культового гурту:

Зовнішній вигляд та інтерфейс:



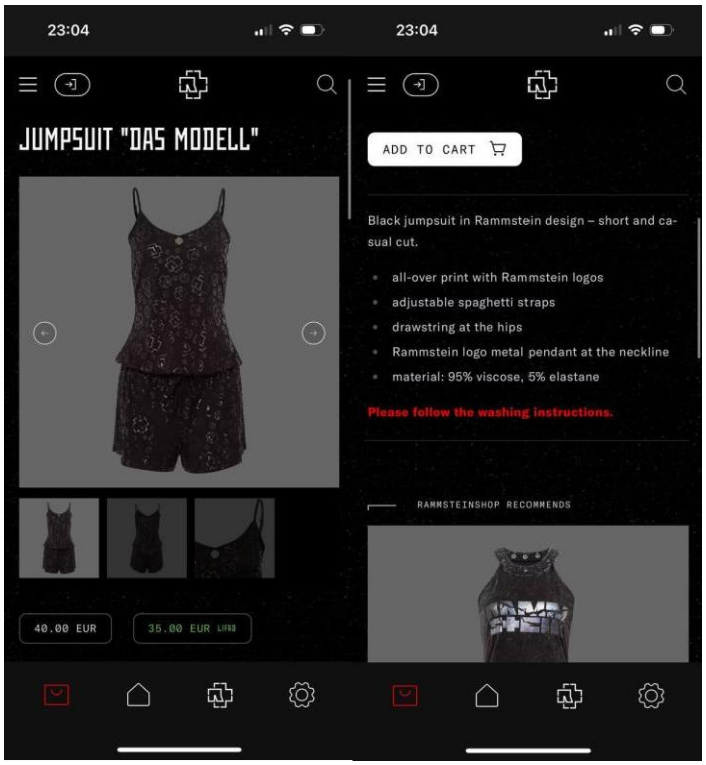


Рис 1.3.1 – Інтерфейс та функції додатку гурту Rammstein

Переваги додатку Rammstein:

- **Дизайн та інтерфейс –** інтерфейс додатку є інтуїтивно зрозумілим, не навантаженим різними елементами, дизайн створений у фірмовому та впізнаному стилі гурту, кожен товар, альбом, розклад концертів та всі вікна систематизовано так, щоб користувач міг з легкістю знайти той чи інший елемент інтерфейсу та функціоналу додатку.
- **Обширний каталог –** абсолютно усі товари які виготовляє гурт можна знайти у цьому додатку, це дає змогу користувачу замовити будь-який товар не виходячи із дому.
- **Список альбомів та пісень –** абсолютно усі альбоми гурту можна знайти у відповідній вкладці застосунку, також натискаючи на альбом можна побачити список усіх пісень даного альбому. Це дуже важлива функція, користувач має змогу вивчити творчу історії та розвиток гурту.

- **Новини та розклад концертів гурту** – користувач має змогу переглянути останні новини гурту, передивитися розклад концертів та придбати квитки.

Недоліки додатку Rammstein:

- **Відсутня можливість прослуховувати пісні –** в додатку не передбачена можливість прослуховувати музику не використовуючи сторонні сервіси, замість цього е просто посилання на Apple Music та на Spotify. На мою думку багатьом людям не вистачає такої функції, бо це дуже зручно.

Тож проаналізувавши додаток гурту Rammstein, можна зрозуміти що для того щоб привабити користувачів, додаток повинен мати зручний, інтуїтивний та красивий дизайн, всі необхідні функції які є у конкурентів та декілька функцій які у конкурентів не представлені, також важлива стабільність роботи всіх функцій застосунку.

2.Проектування системи

2.1 Методи програмної реалізації які я використав

Створення зрозумілого та інтуїтивного інтерфейсу є важливим фактором, який буде вирішувати успіх нашого додатку. Інтерфейс та функціонал повинен бути таким, щоб користувач міг легко опанувати додаток, щоб використання усіх функцій та можливостей застосунку було легким, також щоб він мав візуально приємний дизайн.

Для створення додатку я використовував наступні інструменти та середовища розробки: мова програмування Kotlin, XML, Android Studio(версія Iguana) та Figma

Мова програмування Kotlin - мовою програмування я вибрав Kotlin, оскільки ця мова стала офіційною мовою програмування для розробки мобільних додатків для Android. Він інтегрований з Android Studio та надає зручний і продуктивний досвід розробки для платформи Android, та надає високу продуктивність та зручний синтаксис. Kotlin вирізняється високою виразністю, безпекою типів, і зручністю у використанні, що робить його привабливим вибором для широкого кола завдань і проєктів. Для розробки графічного інтерфейсу буде використано Android XML Layouts.

Рис 2.1.1 – Мова програмування Kotlin

Figma – це онлайн-інструмент для дизайну та прототипування, який

використовується для створення веб-дизайну, мобільних додатків, інтерфейсів користувача, графічного дизайну та інших проектів.

Figma пропонує багато інструментів для створення векторних малюнків, інтерфейсів, роботи з текстом, масштабування та розміщення об'єктів, а також інструменти для прототипування і спільної роботи.

Інтерфейс Figma інтуїтивно зрозумілий і легкий у використанні, що робить його популярним серед дизайнерів будь-якого рівня.

Однією з ключових особливостей Figma є можливість спільної роботи над проектами. Кілька користувачів можуть одночасно працювати над одним проектом, бачити зміни в реальному часі та взаємодіяти один з одним.

Figma підтримує інтеграцію з багатьма іншими інструментами та сервісами, такими як Slack, Trello, Zeplin тощо, що спрощує спільну роботу та обмін даними між різними інструментами.

Figma дозволяє створювати інтерактивні прототипи, які допомагають визначати поведінку і функціонал додатків або веб-сайтів перед їх розробкою.

Всі дані в Figma зберігаються в хмарі, що забезпечує доступ до них з будь-якого пристрою. Платформа також забезпечує високий рівень безпеки для збережених даних.

Figma є потужним інструментом для дизайнерів та команд, які шукають ефективні способи створення та спільної роботи над проектами в онлайн-середовищі.

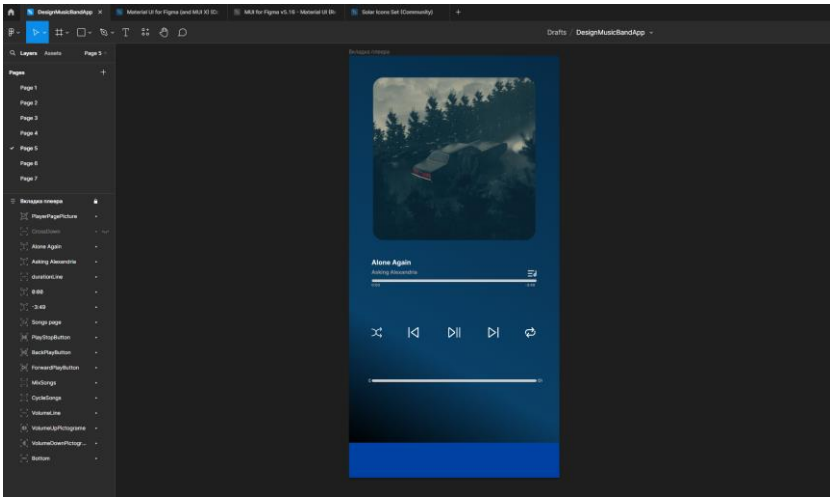


Рис 2.1.2 – Программа Figma

Android Studio - це інтегроване середовище розробки (IDE) для розробки

мобільних додатків на платформі Android. Android Studio була представлена Google у 2013 році як офіційне середовище розробки для Android. Вона стала заміною Eclipse Android Development Tools (ADT) як основне середовище розробки для платформи Android. Kotlin - це офіційна мова програмування для розробки Android-додатків в Android Studio тож Android Studio надає повну підтримку Kotlin, включаючи автоматичне перетворення коду Java в Kotlin, інструменти для міграції проектів на Kotlin та інше.

Android Studio має вбудований візуальний редактор для створення інтерфейсу користувача за допомогою панелі макетування, що дозволяє розташовувати елементи і відстежувати їх вигляд на різних пристроях. Android Studio постачається з вбудованою підтримкою Android SDK, що включає в себе компілятори, дебагери, симулятори, аналізатори профілювання та інші інструменти для розробки Android-додатків.

Android Studio підтримує розширення функціональності за допомогою плагінів. Користувачі можуть встановлювати різноманітні плагіни для розширення можливостей IDE, наприклад, для роботи з іншими мовами програмування або фреймворками.

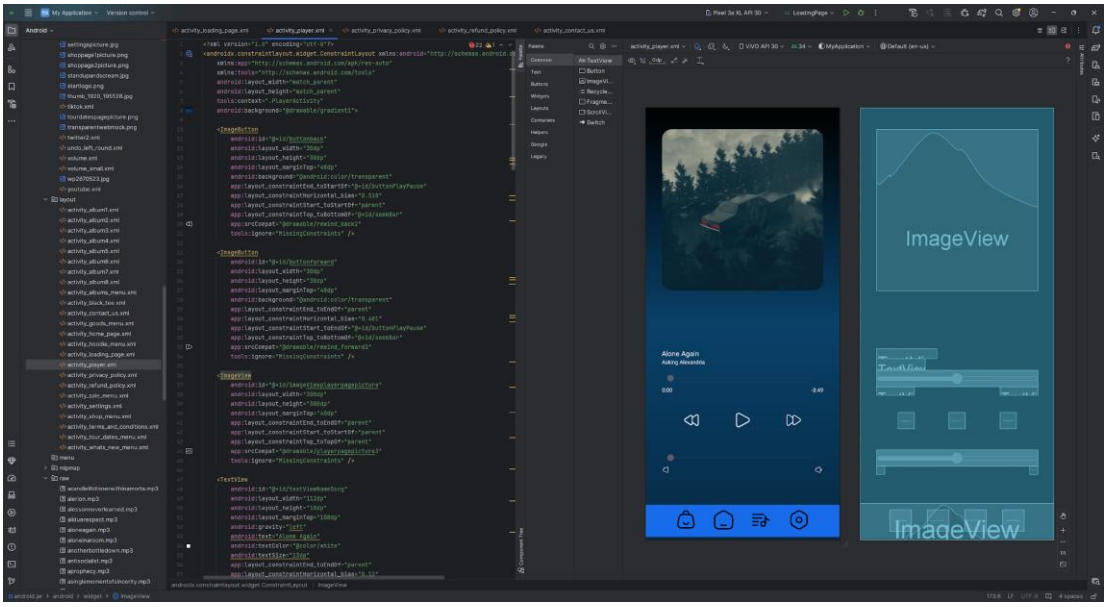


Рис 2.1.3 – Інтерфейс Android Studio

XML Розмітка Інтерфейсу

XML (Extensible Markup Language) — це мова розмітки, розроблена для зберігання та передачі даних у структурованому форматі. XML дозволяє користувачам визначати власні теги та структури, що робить його дуже гнучким та універсальним для різних застосувань.

Переваги XML:

- **Гнучкість**: XML дозволяє створювати власні теги і структури, що робить його універсальним для різних завдань.
- **Сумісність**: XML широко підтримується багатьма програмними продуктами та мовами програмування, що полегшує інтеграцію з іншими системами.
- **Масштабованість**: Можливість обробляти великі обсяги даних та складні ієрархії без втрати продуктивності.
- **Валідність та перевірка**: Використання DTD (Document Type Definition) або XML Schema для визначення структури даних дозволяє забезпечити їх валідність та цілісність.
- **Платформна незалежність**: XML є текстовим форматом, що дозволяє передавати дані між різними системами та платформами без необхідності додаткових перетворень.
- **Міжнародні стандарти**: XML є стандартом, визначеним W3C (World Wide Web Consortium), що забезпечує його стабільність та довготривалу підтримку.

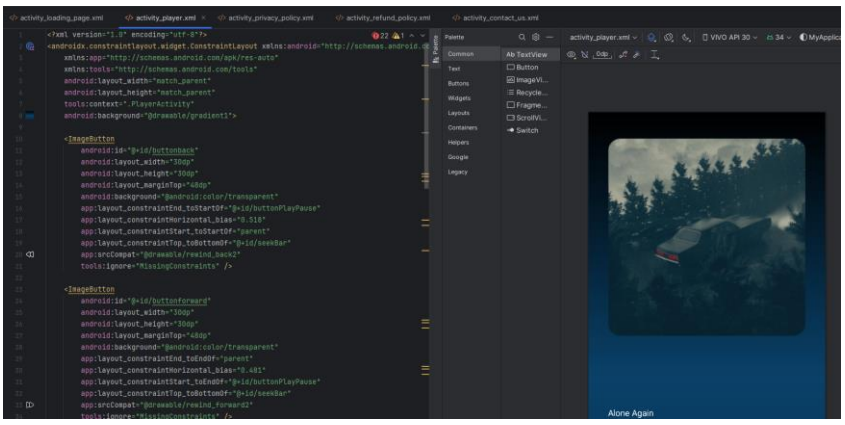


Рис 2.1.4 – XML розмітка

3. Програмна реалізація

3.1 Розробка інтерфейсу

Інтерфейс виконаний у стилі гурту, без навантаження великою кількістю елементів на екрані, це зроблено для того, щоб користувач міг легко опанувати та орієнтуватися при роботі з додатком та елементами всіх його сторінок. (рис. 3.1.1)

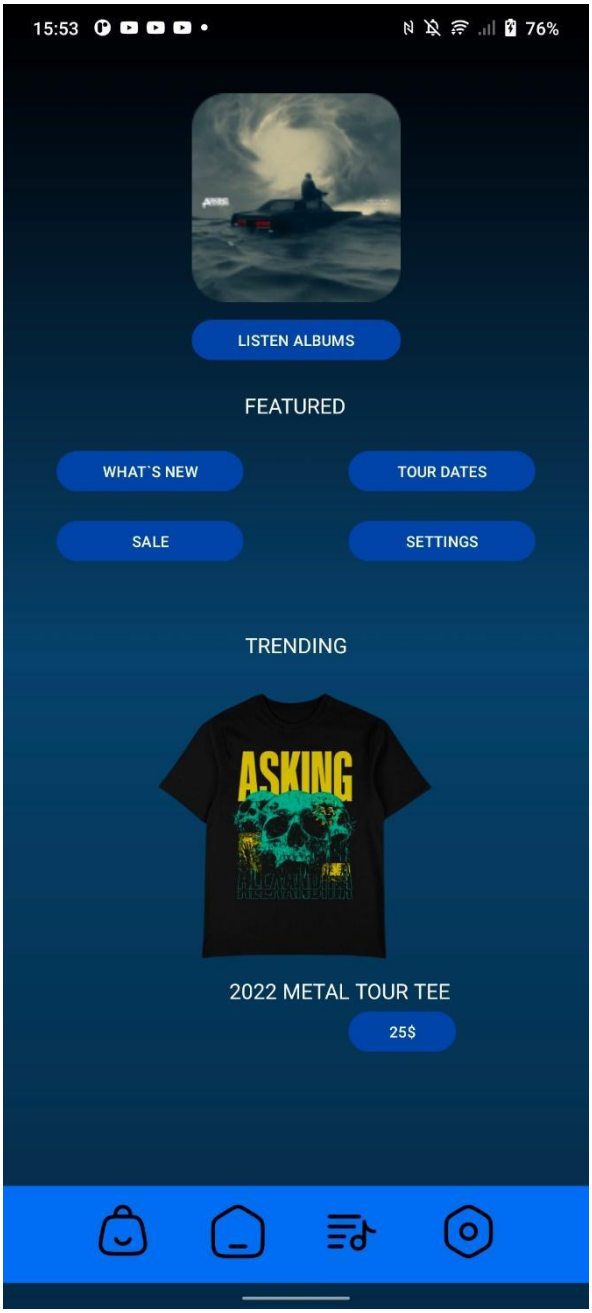


Рис 3.1.1 – Інтерфейс додатку

Кольори додатку я обрав у стилі гурту, бо це одна з деталей яка має великий вплив на впізнаваність гурту. Іконки елементів, колір кнопок виконані у більш світлому кольорі ніж фон, щоб вони виділялися та були більш помітні користувачеві.

Інтерфейс вікна завантаження програми

При запуску додатку користувача зустрічає вікно завантаження застосунку (рис. 3.2.1)

Рис 3.2.1 – Вікно завантаження програми

Вікно виконане в загальному стилі застосунку. Логотип гурту запозичений у сервісу «Pinterest» та оброблений за допомогою застосунку «Photoshop» щоб надати йому характерного стилю.

Інтерфейс головного меню

Після вікна завантаження користувач потрапить на головне меню, та головний екран застосунку. (рис. 3.3.1)

Рис 3.3.1 – Інтерфейс головного меню

На основному екрані ми можемо побачити такі елементи:

- Рекламу нового альбому гурту включаючи опис та клавішу «Meet Our New Album» для швидкого переходу до списку та прослуховування альбомів гурту.
- Рекламу кліпу гурту з кадром із відео та кнопкою «Watch Our Clip Alone Again» для перегляду данного відео на You Tube.

- Клавішу «Explore Our Shop» для переходу на вікно магазину, де користувач матиме змогу обрати та купити необхідний йому товар.
- Поле навігації по застосунку, воно знаходиться у нижній частині вікна, там розміщені кнопки для швидкої навігації між основними вікнами застосунку, а саме «Shop Now», «Main Menu», «Settings», «Albums Menu».

Інтерфейс меню магазину

Коли користувач натискає на кнопку «Shop Now» він потрапляє на сторінку магазину, де він може обрати необхідний товар (рис. 3.4.1)

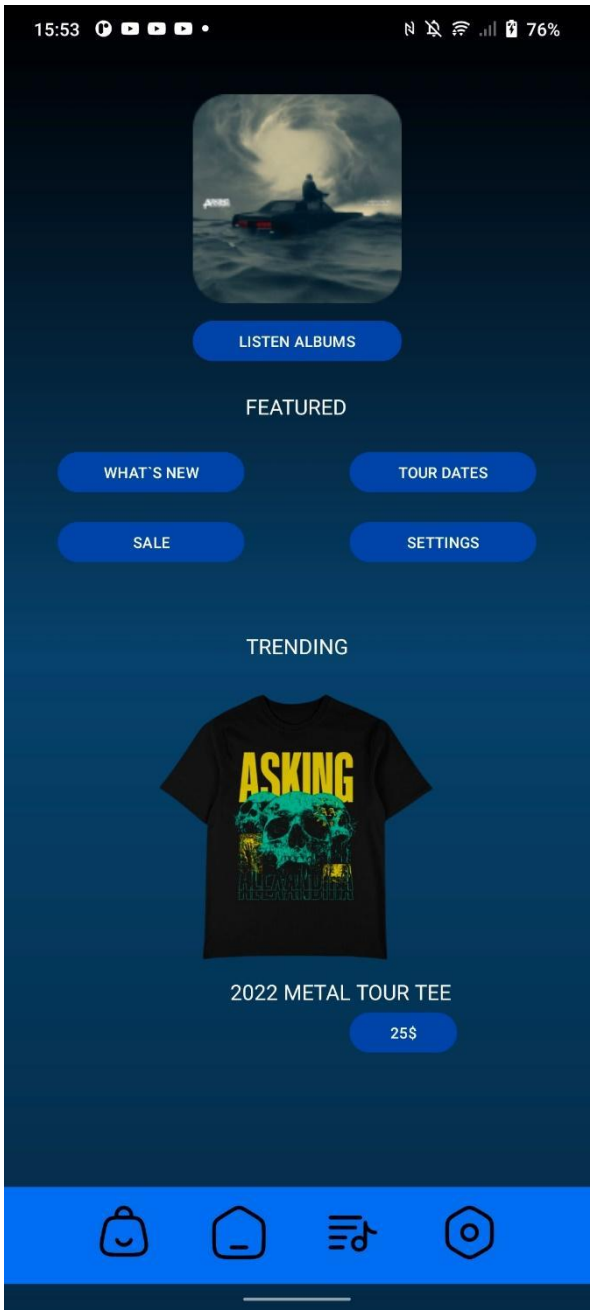


Рис 3.4.1 – Інтерфейс вікна магазину

Вікно магазину має такі елементи та функції:

- Кнопка «Listen Albums» натиснувши на неї користувач зможе потрапити на вікно «Albums Menu» де у свою чергу він зможе побачити список усіх альбомів гурту та прослухати усі пісні у кожному з них.
- Кнопка «Sale» натиснувши на неї користувач зможе переглянути які товари потрапили під розпродаж та реалізуються за нижчими цінами
- Кнопка «What`s New» натиснувши на цю кнопку користувач потрапить на вікно де показано найсвіжіші товари у каталозі
- Кнопка «Tour Dates» натиснувши на яку користувач зможе переглянути список запланованих концертів гурту та купити квитки на потрібний концерт.
- Кнопка «Settings» ця кнопка відкриває користувачу вікно налаштувань та корисної інформації. Там користувач може отримати посилання на стримінгові сервіси «Apple Music» та «Spotify», прочитати інформацію щодо конфіденційності інформації і тому подібне. Також на цьому вікні наявні посилання на офіційний сайт гурту.
- Також користувач бачить тут товар із ціною та коротким описом, натиснувши на кнопку «Price» він зможе перейти на сторінку товару де побачить більше інформації про нього, та зможе оформити покупку.
- Поле навігації по застосунку, воно знаходиться у нижній частині вікна, там розміщені кнопки для швидкої навігації між основними вікнами застосунку, а саме «Shop Now», «Main Menu», «Settings», «Albums Menu».

Інтерфейс вікна «Sale»

Натиснувши на кнопку «Sale» користувач зможе переглянути які товари потрапили під розпродаж та реалізуються за нижчими цінами (рис 3.5.1)

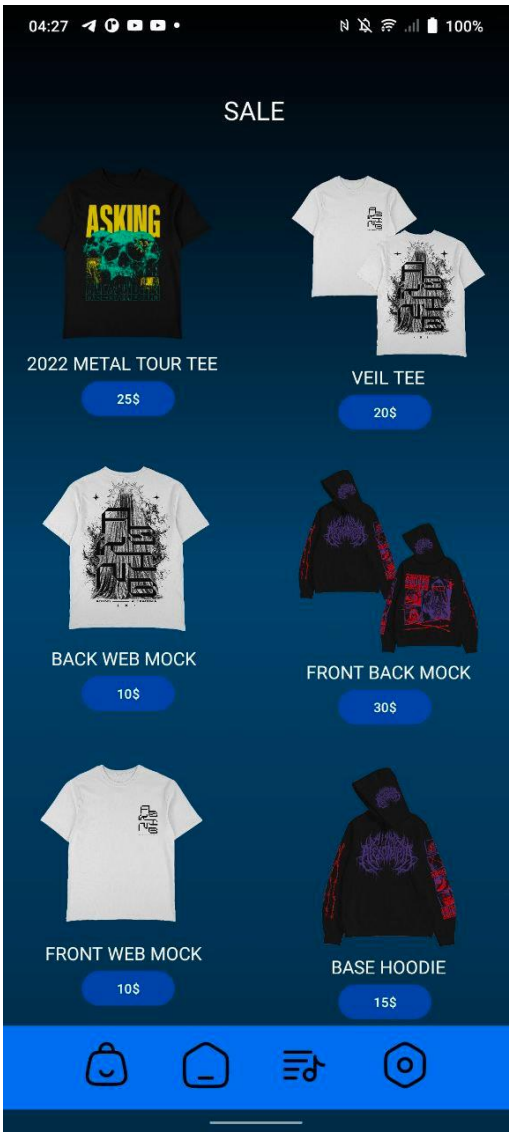


Рис 3.5.1 – Інтерфейс вікна «Sale»

Вікно «Sale» має такі елементи та функції:

- Зображення, короткі описи та ціна товарів у каталозі, які потрапили під розпродаж, він може оцінити їх зовнішній вигляд, короткий опис та ціну, натиснувши на кнопку з ціною він перейде на сторінку саме цього товару, де зможе прочитати опис, вибрати розмір та замовити товар.
- Поле навігації по застосунку, воно знаходиться у нижній частині вікна, там розміщені кнопки для швидкої навігації між основними вікнами застосунку, а саме «Shop Now», «Main Menu», «Settings»,

«Albums Menu».

Інтерфейс вікна «What`s new»

Натиснувши на кнопку «What`s new» користувач потрапить на вікно де показано найсвіжіші товари у каталозі (рис 3.6.1)

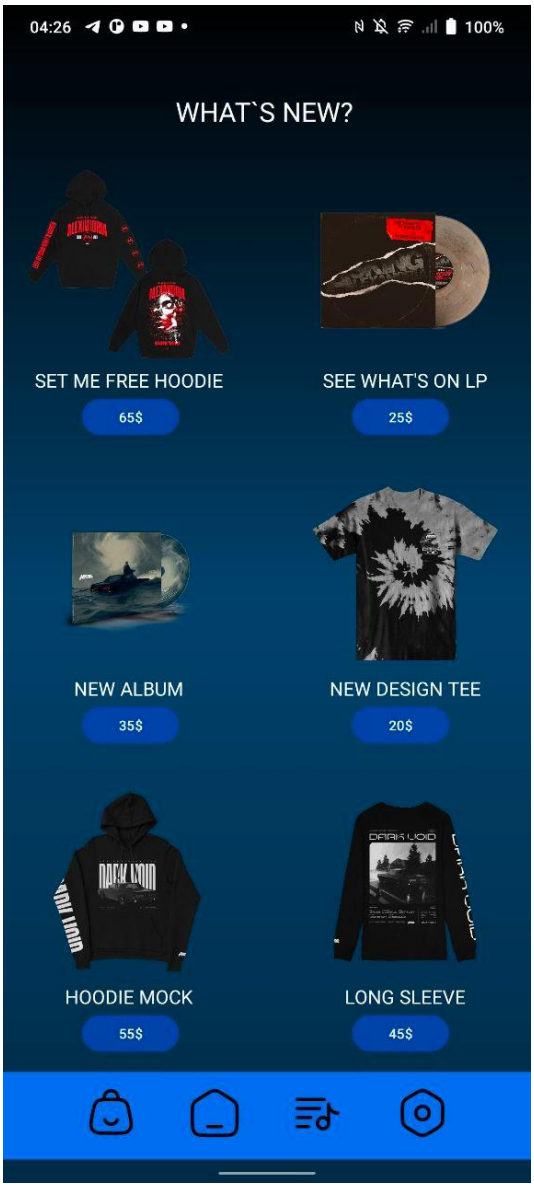


Рис 3.6.1 – Інтерфейс вікна «What`s new»

Вікно «What`s new» має такі елементи та функції:

- Зображення, короткі описи та ціна найновіших товарів у каталозі, він може оцінити їх зовнішній вигляд, короткий опис та ціну, натиснувши на кнопку з ціною він перейде на сторінку саме цього товару, де зможе прочитати опис, вибрати розмір та замовити товар.
- Поле навігації по застосунку, воно знаходиться у нижній частині вікна, там розміщені кнопки для швидкої навігації між основними

вікнами застосунку, а саме «Shop Now», «Main Menu», «Settings», «Albums Menu».

Інтерфейс вікна «Music Player»

натиснувши на кнопку «Music Player» користувач має змогу скористатися функцією прослуховування музики гурту (рис 3.7.1)

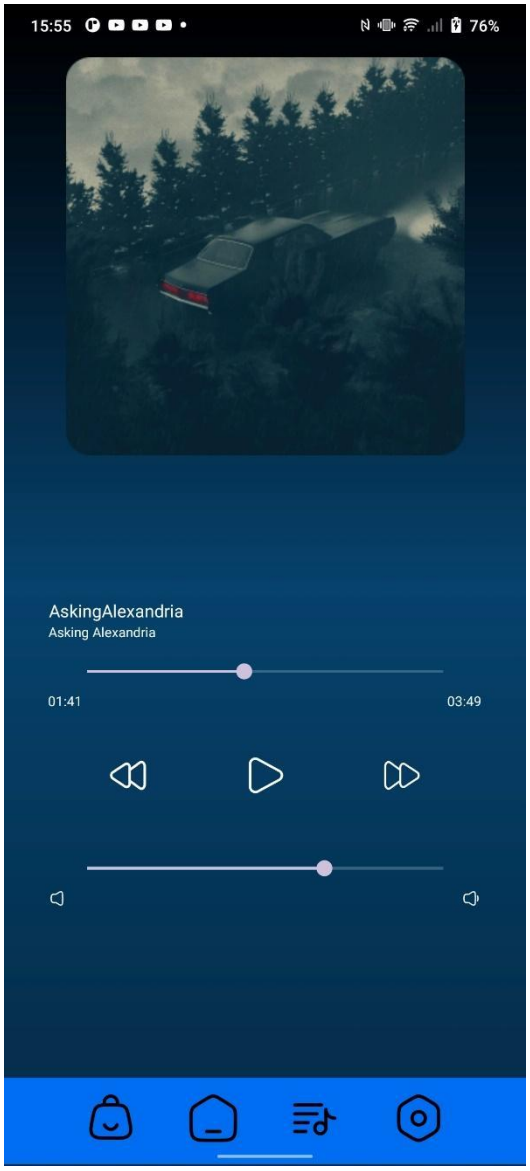


Рис 3.7.1 – Інтерфейс вікна «Music Player»

Вікно «Music Player» має такі елементи та функції:

- Логотип пісні
- «Text View» - назва пісні, що грає та автора
- Значення тривалості пісні – текстові поля, які показують значення скільки часу пісня грає, та скільки закінчилось до завершення треку. Ці значення оновлюються кожні 100мс.

- «Seek Bar» який показує скільки пісня грає та ще буде грати. При роботі повзунок шкали проходить зліва на право.

- Кнопки попередній та наступний трек, її натискання відповідно переключає пісні.
- Кнопка паузи та відтворення, це клавіша, яка зупиняє програвання треку, а при повторному натисканні продовжує відтворення пісні. Також кнопка при натисканні має вплив на шкалу відтворення пісні, та на часові індикатори, котрі зупиняються під час паузи або продовжують рух під час відтворення пісні.
- «Seek Bar» за допомогою якого можна регулювати гучність.
- Поле навігації по застосунку, воно знаходиться у нижній частині вікна, там розміщені кнопки для швидкої навігації між основними вікнами застосунку, а саме «Shop Now», «Main Menu», «Settings», «Albums Menu».

Інтерфейс вікна «Settings»

Кнопка «Settings» відкриває користувачу вікно налаштувань та корисної інформації. (рис 3.8.1)

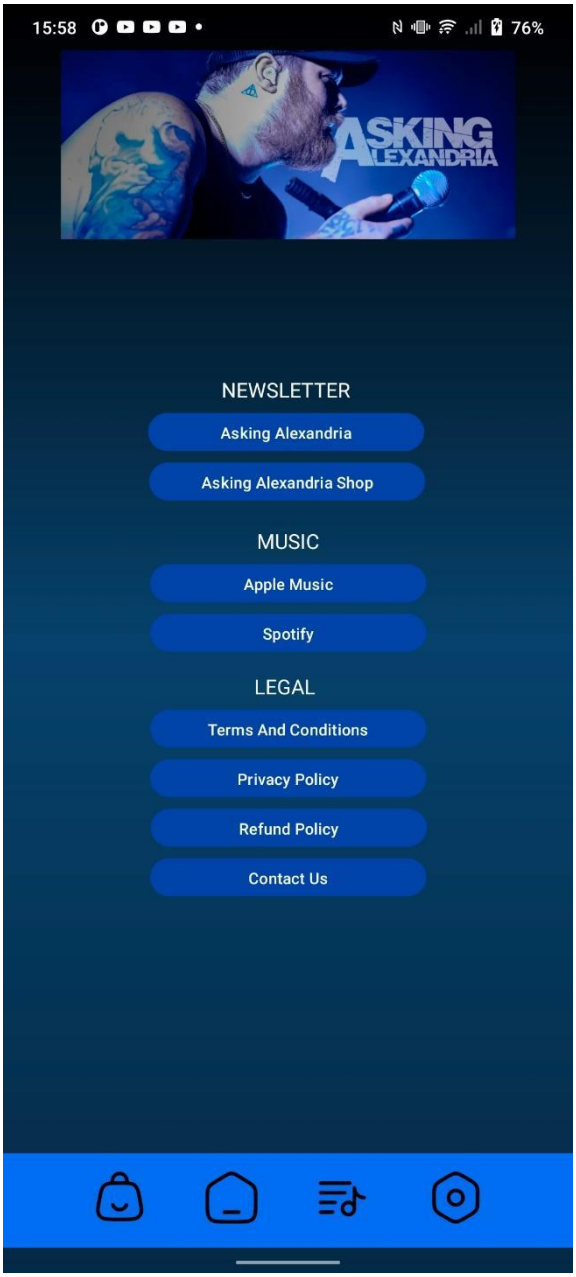


Рис 3.8.1 – Інтерфейс вікна «Settings»

Вікно «Settings» має такі елементи та функції:

- Кнопки «Asking Alexandria» та «Asking Alexandria Shop» які дозволяють користувачеві перейти на веб сайт, та на сторінку магазину веб сайту гурту відповідно
- Кнопки «Apple Music» та «Spotify» які дозволяють

користувачеві перейти на стримінгові сервіси

- Кнопки «Terms and conditions», «Privacy Policy», «Refund Policy» та «Contact Us» переходячи по яким користувач може отримати контакті дані менеджерів гурту, та прочитати політику конфіденційності та іншу корисну інформацію
- Поле навігації по застосунку, воно знаходиться у нижній частині вікна, там розміщені кнопки для швидкої навігації між основними вікнами застосунку, а саме «Shop Now», «Main Menu», «Settings», «Albums Menu».

Інтерфейс вікна «Listen Now»

Кнопка «Listen Now» відкриває користувачу вікно зі списком альбомів гурту для подальшого прослуховування пісень цих альбомів. (рис 3.9.1)

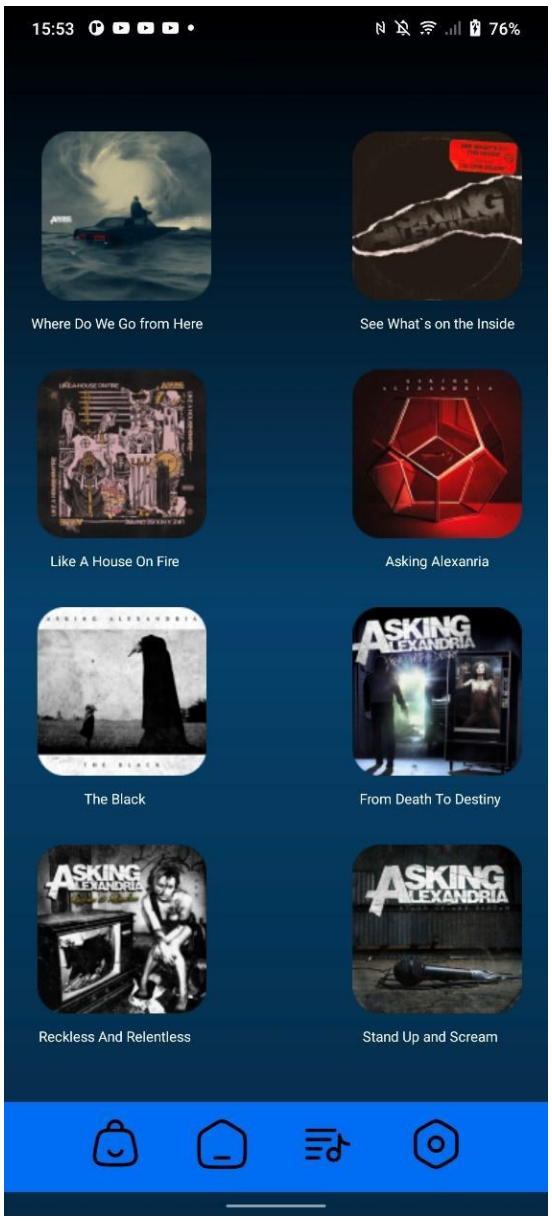


Рис 3.9.1 – Інтерфейс вікна «Albums Menu»

Вікно «Albums Menu» має такі елементи та функції:

- Вікно має 8 елементів типу «ImageButton» із зображеннями альбомів, натискаючи на обкладинки альбомів користувачу відкриється список пісень відповідного альбому вже готових до прослуховування.
- Поле навігації по застосунку, воно знаходиться у нижній частині вікна, там розміщені кнопки для швидкої навігації між основними вікнами застосунку, а саме «Shop Now», «Main Menu», «Settings»,

«Albums Menu».

Інтерфейс вікна «Buy Tickets»

Кнопка «Buy Tickets» відкриває користувачу вікно зі списком концертів під час актуального туру музичного гурту. Натиснувши кнопку користувач зможе легко придбати квитки на потрібний концерт. Елементи «TextView» та «Button» розташовані списком(рис 3.10.1)

Рис 3.10.1 – Інтерфейс вікна «Buy Tickets»

Вікно «Buy Tickets» має такі елементи та функції:

- Елементи «TextView» та «Button» розташовані списком
- Функціональні кнопки для переходу на офіційні сервіси гурту для купівлі квитків на вибраний концерт.
- Текстове наповнення з інформацією про концерт, датою та місцем проведення
- Поле навігації по застосунку, воно знаходиться у нижній частині вікна, там розміщені кнопки для швидкої навігації між основними вікнами застосунку, а саме «Shop Now», «Main Menu», «Settings», «Albums Menu».

Інтерфейс вікна «Списку пісень альбому»

Натискаючи на обкладинки альбомів у вікні «Listen Now» користувачу відкриється список пісень відповідного альбому вже готових до прослуховування (рис 3.11.1)

Рис 3.11.1 – Інтерфейс вікна «Список пісень альбому»

Вікно «Список пісень альбому» має такі елементи та функції:

- Список елементів «Button» з текстом всередині який вказує назву пісні. Також кнопки мають параметр «transparent» який робить кнопку прозорою.
- Елемент «ImageView» - логотип альбому
- Елементи «TextView» - назву альбому та напис «Track List»
- Натискаючи на пісню користувачу відкриватиметься плеєр для прослуховування пісні.
- Поле навігації по застосунку, воно знаходиться у нижній частині вікна, там розміщені кнопки для швидкої навігації між основними вікнами застосунку, а саме «Shop Now», «Main Menu», «Settings», «Albums Menu».

Інтерфейс вікна сторінки товару

Натискаючи на кнопку з ціною будь-якого товару користувачу відкриється сторінка цього товару, де він може обрати необхідний розмір, прочитати характеристики та опис товару, і замовити його(рис 3.12.1)

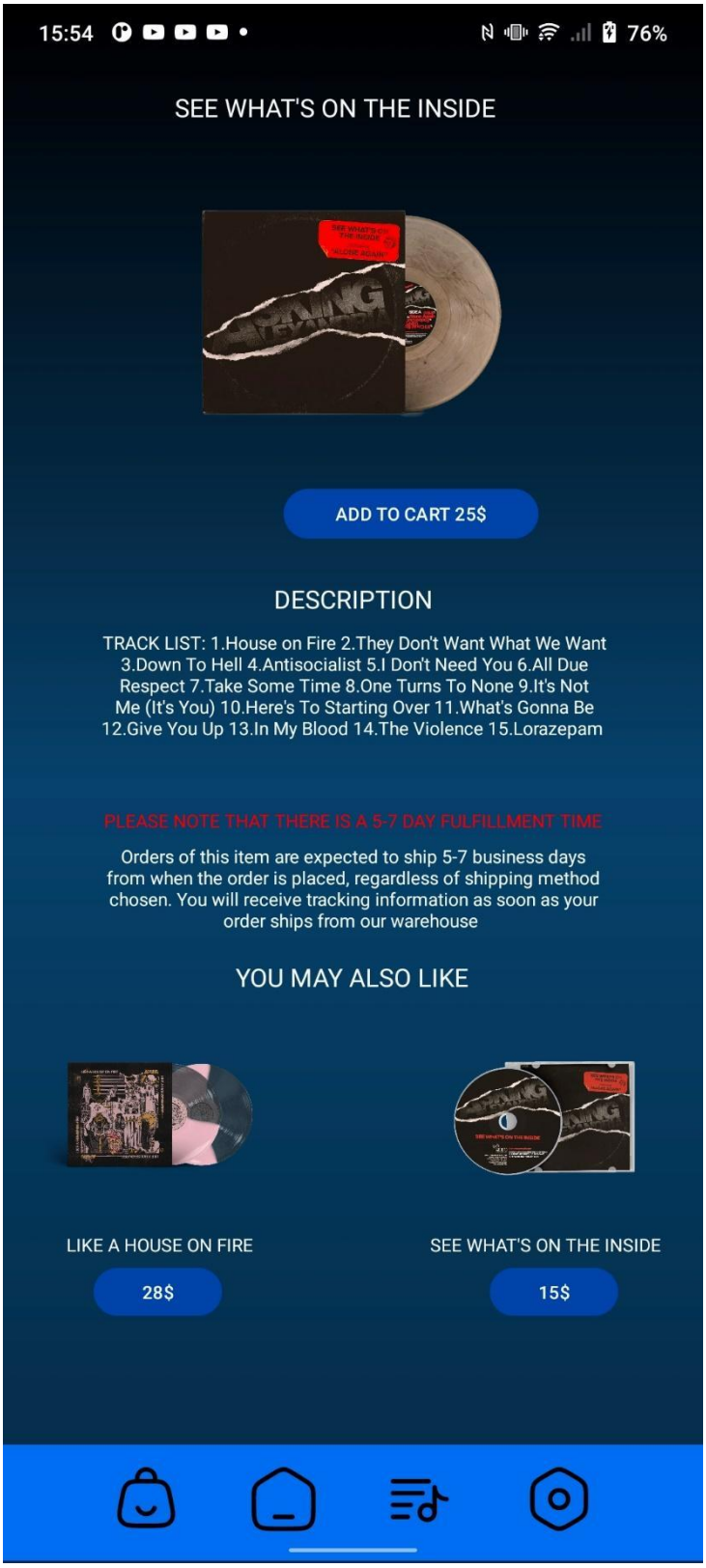


Рис 3.12.1 – Інтерфейс вікна сторінки товару

Вікно сторінки товару має такі елементи та функції:

- Зображення товару, для того щоб користувач мав змогу оцінити зовнішній вигляд товару.
- Кнопка «Add to cart» натиснувши на яку користувач зможе здійснити покупку.
- Елемент вікна з назвою «You may Also Like» з зображеннями та кнопками переходу на сторінки товарів які також можуть сподобатись покупцю.
- Текстове наповнення «Text View» з описом товару та інформацією про доставку.
- Поле навігації по застосунку, воно знаходиться у нижній частині вікна, там розміщені кнопки для швидкої навігації між основними вікнами застосунку, а саме «Shop Now», «Main Menu», «Settings», «Albums Menu».

Таких вікон буде декілька так як каталог магазину наповнений певною кількістю товарі

3.2 Опис програмної реалізації

Опис роботи програми:

Коли користувач відкриває додаток, його зустрічає вікно завантаження Loading Page Activity, і через невеликий проміжок часу переходить до головного меню Main Menu Activity. В Main Menu Activity користувач має змогу натиснувши відповідні кнопки перейти на наступні вікна:

- Albums Menu Activity(список альбомів гурту) для цього треба натиснути кнопку Meet Our New Album або натиснути відповідну кнопку в навігаційному меню знизу. На цій сторінці користувач може натиснути на зображення будь-якого альбому, в цьому випадку використовується метод «ImageButton», таким чином він потрапить на вікно де побачить список пісень альбому, натиснувши на будь-яку пісню відкриється активність «PlayerActivity» і почне грати вибрана пісня. На сторінці «PlayerActivity» користувачу доступні наступні функції: функція паузи та відтворення треку, функція перемикання треків на наступний чи на попередній, функція переміщення по треку за допомогою елемента «SeekBar», функція зміни гучності яка реалізована також за допомогою «SeekBar» та функція автоматичного переходу на наступну пісню після закінчення попередньої. Користувачу не потрібно завчасно завантажувати у пам’ять телефону пісні гурту, бо вони вже вбудовані в додаток, і він зможе слухати пісні, як на стрімінговому сервісі.
- Shop Menu Activity(сторінка магазину гурту) для цього треба натиснути кнопку Explore Our Shop або натиснути відповідну кнопку в навігаційному меню знизу.
- Також є можливість переглянути кліп гурту на сервісі You Tube, для цього треба натиснути кнопку Watch Our Clip Alone Again, і відкриється сторінка гурту та кліп на відповідному сервісі.
- Також на вікні головного меню, як і на будь-якому вікні наявне поле навігації по застосунку, воно знаходиться у нижній частині вікна, там розміщені кнопки для швидкої навігації між основними вікнами застосунку, а саме «Shop Now», «Main Menu», «Settings», «Albums

Menu».

На сторінці магазину «Shop Activity» користувач має змогу вивчити каталог магазину та обрати необхідні товари. Для цього на вікні є необхідні кнопки, а саме:

- Кнопка «Sale» натиснувши на неї користувач зможе переглянути які товари потрапили під розпродаж та реалізуються за нижчими цінами. Кнопка здійснює перехід на вікно Sale Menu Activity
- Кнопка «What`s New» натиснувши на цю кнопку користувач потрапить на вікно де показано найсвіжіші товари у каталозі. Кнопка здійснює перехід на вікно What`s New Menu Activity
- Також користувач бачить тут товар із ціною та коротким описом, натиснувши на кнопку «Price» він зможе перейти на сторінку товару де побачить більше інформації про нього, та зможе оформити покупку. На сторінці будь якого товару, користувач зможе прочитати опис, побачити ціну, прочитати умови доставки, вибрати розмір якщо це одяг, побачити зображення товару. Також йому пропонуються товари які зможуть йому сподобатись. Таких вікон багато так як є певна кількість товарів у каталозі, функціонал один і той самий, користувач натискає на кнопку з ціною на будь якому товарі, і переходить на сторінку товару.
- За допомогою кнопки «Tour Dates» користувач може переглянути список запланованих концертів гурту та купити квитки на потрібний концерт.
- За допомогою кнопки «Settings» користувач відкриє вікно налаштувань та корисної інформації. Там він може отримати посилання на стримінгові сервіси «Apple Music» та «Spotify», прочитати інформацію щодо конфіденційності інформації і тому подібне. Також на цьому вікні наявні посилання на офіційний сайт гурту.
- Також на вікні головного меню, як і на будь-якому вікні наявне поле навігації по застосунку, воно знаходиться у нижній частині вікна, там розміщені кнопки для швидкої навігації між основними вікнами

застосунку, а саме «Shop Now», «Main Menu», «Settings», «Albums Menu».

Основні компоненти та функції додатку:

LoadingPage:

- Використання методу «CoroutineScope» задля того щоб екран завантаження відображався протягом певного часу, а саме 1 секунду (delay(1000))

AlbumActivity:

- Використання Private val buttons = listOf (R.id.buttonSong to R.raw.badblood**)** для створення списку кнопок та відповідних їм аудіофайлів з папки «RAW»
- Використання Private fun setupNavigationButtons(){} – метод для налаштування кнопок навігації
- Використання Private fun setupSongButtons(){} – метод для налаштування кнопок пісень, у якому є метод playAudio(index) для того щоб при натисканні на кнопку грав потрібний аудіофайл.
- Використання Private fun PlayAudio – метод для відтворення аудіофайлів.
- Val audioUris = buttons.map{Uri.parse()} – створення списку URI усіх аудіофайлів
- Val intent = intent(this, PlayerActivity::class.java).apply{ - створення інтенту для переходу на активність PlayerActivity.
- putParceLableArrayListExtra(“audioUris”, ArrayList(audioUris)) – передача списку URI
- putExtra(“currentIndex”, index) – передача індексу поточного аудіофайлу
- startActivity(intent) – запуск PlayerActivity

PlayerActivity:

- audioUris = intent.getParcelableArrayListExtra("audioUris")!! currentIndex = intent.getIntExtra("currentIndex", 0) - Отримання списку

аудіо URI та поточного індексу.

- seekBar.setOnSeekBarChangeListener(object:SeekBar.OnSeekBarChangeListener {override fun onProgressChanged(seekBar: SeekBar?, progress: Int, fromUser: Boolean) { if (fromUser) {mediaPlayer.seekTo(progress * 1000)
  }
  } - Налаштування повзунка для перемотування поточної пісні.
- playAudio() - Відтворення аудіофайлу
- playButton.setOnClickListener {
  if (isPlaying) {
  mediaPlayer.pause()
  } else {
  mediaPlayer.start()updateSeekBar()
  }
  isPlaying = !isPlaying
  } - Налаштування кнопки відтворення/паузи.
- nextButton.setOnClickListener {
  if (currentIndex < audioUris.size - 1) {
  currentIndex++
  playAudio()
  }
  } - Налаштування кнопки наступної пісні.
- prevButton.setOnClickListener {
  if (currentIndex > 0) {
  currentIndex--
  playAudio()
  }
  } - Налаштування кнопки попередньої пісні.

- private fun setupVolumeControl() {val audioManager = getSystemService(AUDIO_SERVICE) as AudioManager val maxVolume = audioManager.getStreamMaxVolume(AudioManager.STREAM_MUSIC) val currentVolume = audioManager.getStreamVolume(AudioManager.STREAM_MUSIC) - Метод для налаштування керування гучністю.

3.3 Тестування застосунку

Для тестування додатку я обрав мануальний метод тестування. Цей спосіб дозволяє ретельно перевірити функціонування усіх можливостей додатку та роботу усіх елементів користувацького інтерфейсу. Тестування я виконав на своєму смартфоні.

Пристрій для тестування: Vivo Y53s (Android 11).

Підготовка

- **Встановлення застосунку**: Я встановив APK-файл додатку на свій смартфон.
- **Налаштування середовища**: Я переконався, що на пристрої встановлено необхідна версія Android та наявний стабільний доступ до Інтернету.

Тестування основних функцій:

Навігація між вікнами та функціонал застосунку:

Мета: Перевірити коректність роботи навігації між вікнами та функціонал.

Процедура: Перейшов між різними вікнами застосунку, перевірив роботу функцій додатку.

Результат: Навігація та функціонал працює коректно (рис 3.3.1).

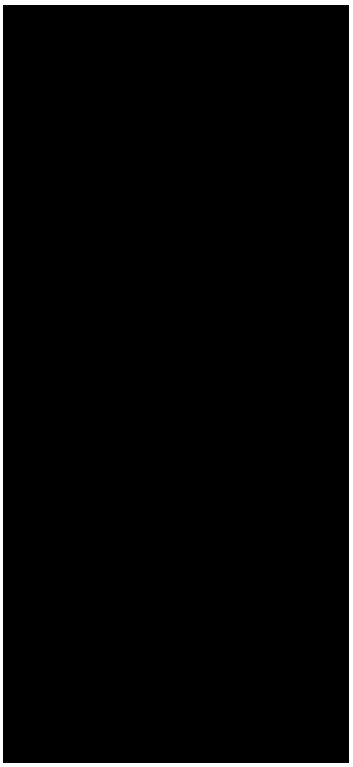


Рис 3.3.1 – Файл у форматі Gif, який відображає тестування функціоналу та навігації у застосунку.

ВИСНОВОК

У процесі створення додатку для популяризації музичного гурту я пройшов кожен крок створення застосунку, вивчав нові для себе методи розробки, проведена мною робота призвела до створення функціонального продукту. Вибір технологій розробки, а саме: мови програмування, середовища розробки та набору інструментів для створення дизайну спирався на сучасні вимоги до створення зручного у використанні та функціонального додатку .

Як середовище для розробки дизайну я обрав Figma, щоб надалі спираючись на створений дизайн макетів спроектувати інтерфейс додатку використовуючи XML-розмітку. Мова програмування Kotlin була обрана мною тому-що вона є офіційною мовою для створення Android додатків та як на мене вона є інтуїтивно зрозумілою. Середовище розробки Android Studio має набір всіх необхідних інструментів для створення додатку за допомогою Kotlin, саме тому я і обрав це середовище розробки.

Програмне забезпечення було виконане спираючись на вимоги та бажання майбутніх користувачів та гурту для якого цей додаток був створений, ці вимоги я виокремив для себе під час аналізу конкурентоспроможності. Застосунок забезпечує усі необхідні функції для купівлі товарів гурту, купівлі

квитків та прослуховування пісень гурту.

Результатом цієї роботи стало створення додатку, що відповідає усім сучасним вимогам і забезпечує приємний досвід для клієнта. У процесі розробки я отримав практичні навички роботи з багатьма технологіями.

СПИСОК ВИКОРИСТАНОЇ ЛІТЕРАТУРИ

1. Програмування з нуля на Kotlin українською (146) Програмування з нуля на Kotlin українською - YouTube
2. Основи мови програмування на Kotlin Основи Kotlin | krypton.com.ua
3. Основи Android Studio Основи Android Studio. Уроки для початківців. W3Schools українською (w3schoolsua.github.io)
4. Android з 0 до junior ANDROID УКРАЇНСЬКОЮ. Старт курсу, урок перший, Android Studio (youtube.com)
5. Figma з нуля. Вчимось працювати у фігма Figma з нуля. Вчимось працювати у фігма - UXPUB ·· Дизайн-спільнота
6. Figma українською | Що таке Фігма і як вона працює? Figma українською | Що таке Фігма і як вона працює? (youtube.com)
7. Робота зі звуком Kotlin #7 – Работа со звуком (MediaPlayer) – курси українською (itproger.com)
8. Створення градієнту та інших стилів Background LinearGradient | Android Developers
9. XML для початківців Підручник XML для початківців (guru99.com)

Додатки

Додаток 1 – Код модулів додатку

Album1Activity

```
package com.example.myapplication

import android.content.ContentResolver
import android.content.Intent
import android.database.Cursor
import android.media.MediaPlayer
import android.net.Uri
import android.os.Bundle
import android.provider.MediaStore
import android.view.View
import android.widget.Button
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity

class Album1Activity : AppCompatActivity() {

    private val buttons = listOf(
        R.id.buttonSong1 to R.raw.badblood,
        R.id.buttonSong2 to R.raw.thingscouldbedifferent,
        R.id.buttonSong3 to R.raw.letgo,
        R.id.buttonSong4 to R.raw.psycho,
        R.id.buttonSong5 to R.raw.darkvoid,
        R.id.buttonSong6 to R.raw.nothingleft,
        R.id.buttonSong7 to R.raw.feel,
        R.id.buttonSong8 to R.raw.letthedeadtakeme,
        R.id.buttonSong9 to R.raw.killitwithfire,
        R.id.buttonSong10 to R.raw.holdingontosomethingmore,
        R.id.buttonSong11 to R.raw.wheredowegofromhere
    )

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_album1)


        setupNavigationButtons()

        setupSongButtons()
    }

    private fun setupNavigationButtons() {

        findViewById<View>(R.id.imageButtonHomePage)?.setOnClickListener {
            startActivity(Intent(this, HomePageActivity::class.java))
        }

        findViewById<View>(R.id.imageButtonShop)?.setOnClickListener {
            startActivity(Intent(this, ShopMenuActivity::class.java))
        }

        findViewById<View>(R.id.imageButtonAlbumsRecycler)?.setOnClickListener
{
            startActivity(Intent(this, AlbumsMenuActivity::class.java))
        }

        findViewById<View>(R.id.imageButtonSettings)?.setOnClickListener {
            startActivity(Intent(this, SettingsActivity::class.java))
        }
    }

    private fun setupSongButtons() {
```

```
        buttons.forEachIndexed { index, (buttonId, audioId) ->
            findViewById<Button>(buttonId).setOnClickListener {
                playAudio(index)
            }
        }
    }

    private fun playAudio(index: Int) {

        val audioUris = buttons.map {
Uri.parse("android.resource://${packageName}/${it.second}") }

        val intent = Intent(this, PlayerActivity::class.java).apply {
            putParcelableArrayListExtra("audioUris", ArrayList(audioUris))
            putExtra("currentIndex", index)
        }
        startActivity(intent)
    }
}
```

Album2Activity

```
package com.example.myapplication

import android.content.ContentResolver
import android.content.Intent
import android.database.Cursor
import android.media.MediaPlayer
import android.net.Uri
import android.os.Bundle
import android.provider.MediaStore
import android.view.View
import android.widget.Button
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity

class Album2Activity : AppCompatActivity() {

    private val buttons = listOf(
        R.id.buttonSong1 to R.raw.intro,
        R.id.buttonSong2 to R.raw.aloneagain,
        R.id.buttonSong3 to R.raw.thegrey,
        R.id.buttonSong4 to R.raw.fame,
        R.id.buttonSong5 to R.raw.seewhatontheinside,
        R.id.buttonSong6 to R.raw.ifcoulderaseit,
        R.id.buttonSong7 to R.raw.miserylovescompany,
        R.id.buttonSong8 to R.raw.findmyself,
        R.id.buttonSong9 to R.raw.fadedout,
        R.id.buttonSong10 to R.raw.nevergonnalearn,
        R.id.buttonSong11 to R.raw.youmadeitthisfar
    )

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_album2)


        setupNavigationButtons()

        setupSongButtons()
    }

    private fun setupNavigationButtons() {

        findViewById<View>(R.id.imageButtonHomePage)?.setOnClickListener {
            startActivity(Intent(this, HomePageActivity::class.java))
        }
```

```
        findViewById<View>(R.id.imageButtonShop)?.setOnClickListener {
            startActivity(Intent(this, ShopMenuActivity::class.java))
        }

        findViewById<View>(R.id.imageButtonAlbumsRecycler)?.setOnClickListener
{
            startActivity(Intent(this, AlbumsMenuActivity::class.java))
        }

        findViewById<View>(R.id.imageButtonSettings)?.setOnClickListener {
            startActivity(Intent(this, SettingsActivity::class.java))
        }
    }

    private fun setupSongButtons() {

        buttons.forEachIndexed { index, (buttonId, audioId) ->
            findViewById<Button>(buttonId).setOnClickListener {
                playAudio(index)
            }
        }
    }

    private fun playAudio(index: Int) {

        val audioUris = buttons.map {
Uri.parse("android.resource://${packageName}/${it.second}") }

        val intent = Intent(this, PlayerActivity::class.java).apply {
            putParcelableArrayListExtra("audioUris", ArrayList(audioUris))
            putExtra("currentIndex", index)
        }
        startActivity(intent)
    }
}
```

Album3Activity

```
package com.example.myapplication

import android.content.ContentResolver
import android.content.Intent
import android.database.Cursor
import android.media.MediaPlayer
import android.net.Uri
import android.os.Bundle
import android.provider.MediaStore
import android.view.View
import android.widget.Button
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity

class Album3Activity : AppCompatActivity() {
    private val buttons = listOf(
        R.id.buttonSong1 to R.raw.houseonfire,
        R.id.buttonSong2 to R.raw.theydontwantwhatwewant,
        R.id.buttonSong3 to R.raw.downtohell,
        R.id.buttonSong4 to R.raw.antisocialist,
        R.id.buttonSong5 to R.raw.idontneedyou,
        R.id.buttonSong6 to R.raw.allduerespect,
        R.id.buttonSong7 to R.raw.takesometime,
        R.id.buttonSong8 to R.raw.oneturnstonone,
        R.id.buttonSong9 to R.raw.itsnotme,
        R.id.buttonSong10 to R.raw.herestostartingover,
        R.id.buttonSong11 to R.raw.whatsgonnabe,
        R.id.buttonSong12 to R.raw.giveyouup,
        R.id.buttonSong13 to R.raw.inmyblood,
        R.id.buttonSong14 to R.raw.theviolence,
```

```
    )

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_album3)

        setupNavigationButtons()

        setupSongButtons()
    }

    private fun setupNavigationButtons() {
        findViewById<View>(R.id.imageButtonHomePage)?.setOnClickListener {
            startActivity(Intent(this, HomePageActivity::class.java))
        }

        findViewById<View>(R.id.imageButtonShop)?.setOnClickListener {
            startActivity(Intent(this, ShopMenuActivity::class.java))
        }

        findViewById<View>(R.id.imageButtonAlbumsRecycler)?.setOnClickListener
{
            startActivity(Intent(this, AlbumsMenuActivity::class.java))
        }

        findViewById<View>(R.id.imageButtonSettings)?.setOnClickListener {
            startActivity(Intent(this, SettingsActivity::class.java))
        }
    }

    private fun setupSongButtons() {
        buttons.forEachIndexed { index, (buttonId, audioId) ->
            findViewById<Button>(buttonId).setOnClickListener {
                playAudio(index)
            }
        }
    }

    private fun playAudio(index: Int) {
        val audioUris = buttons.map {
Uri.parse("android.resource://${packageName}/${it.second}") }
        val intent = Intent(this, PlayerActivity::class.java).apply {
            putParcelableArrayListExtra("audioUris", ArrayList(audioUris))
            putExtra("currentIndex", index)
        }
        startActivity(intent)
    }
}
```

Album4Activity

```
package com.example.myapplication

import android.content.ContentResolver
import android.content.Intent
import android.database.Cursor
import android.media.MediaPlayer
import android.net.Uri
import android.os.Bundle
import android.provider.MediaStore
import android.view.View
import android.widget.Button
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity

class Album4Activity : AppCompatActivity() {
    private val buttons = listOf(
        R.id.buttonSong1 to R.raw.aloneinaroom,
```

```
        R.id.buttonSong2 to R.raw.intothefire,
        R.id.buttonSong3 to R.raw.hopelesslyhopeful,
        R.id.buttonSong4 to R.raw.whereditdgo,
        R.id.buttonSong5 to R.raw.riseup,
        R.id.buttonSong6 to R.raw.whenthelightscomeon,
        R.id.buttonSong7 to R.raw.underdenver,
        R.id.buttonSong8 to R.raw.vultures,
        R.id.buttonSong9 to R.raw.eve,
        R.id.buttonSong10 to R.raw.iamone,
        R.id.buttonSong11 to R.raw.empire,
        R.id.buttonSong12 to R.raw.room138,
    )

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_album4)

        setupNavigationButtons()
        setupSongButtons()
    }

    private fun setupNavigationButtons() {
        findViewById<View>(R.id.imageButtonHomePage)?.setOnClickListener {
            startActivity(Intent(this, HomePageActivity::class.java))
        }

        findViewById<View>(R.id.imageButtonShop)?.setOnClickListener {
            startActivity(Intent(this, ShopMenuActivity::class.java))
        }

        findViewById<View>(R.id.imageButtonAlbumsRecycler)?.setOnClickListener
{
            startActivity(Intent(this, AlbumsMenuActivity::class.java))
        }

        findViewById<View>(R.id.imageButtonSettings)?.setOnClickListener {
            startActivity(Intent(this, SettingsActivity::class.java))
        }
    }

    private fun setupSongButtons() {
        buttons.forEachIndexed { index, (buttonId, audioId) ->
            findViewById<Button>(buttonId).setOnClickListener {
                playAudio(index)
            }
        }
    }

    private fun playAudio(index: Int) {
        val audioUris = buttons.map {
Uri.parse("android.resource://${packageName}/${it.second}") }
        val intent = Intent(this, PlayerActivity::class.java).apply {
            putParcelableArrayListExtra("audioUris", ArrayList(audioUris))
            putExtra("currentIndex", index)
        }
        startActivity(intent)
    }
}
```

Album5Activity

```
package com.example.myapplication

import android.content.ContentResolver
import android.content.Intent
import android.database.Cursor
import android.media.MediaPlayer
import android.net.Uri
```

```
import android.os.Bundle
import android.provider.MediaStore
import android.view.View
import android.widget.Button
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity

class Album5Activity : AppCompatActivity() {
    private val buttons = listOf(
        R.id.buttonSong1 to R.raw.welcome,
        R.id.buttonSong2 to R.raw.dearinsanity,
        R.id.buttonSong3 to R.raw.closure,
        R.id.buttonSong4 to R.raw.alessonneverlearned,
        R.id.buttonSong5 to R.raw.tothestage,
        R.id.buttonSong6 to R.raw.dedication,
        R.id.buttonSong7 to R.raw.someonesomewhere,
        R.id.buttonSong8 to R.raw.breathless,
        R.id.buttonSong9 to R.raw.thematch,
        R.id.buttonSong10 to R.raw.anotherbottledown,
        R.id.buttonSong11 to R.raw.recklessrelentless,
        R.id.buttonSong12 to R.raw.morteetdabo,
    )

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_album5)

        setupNavigationButtons()
        setupSongButtons()
    }

    private fun setupNavigationButtons() {
        findViewById<View>(R.id.imageButtonHomePage)?.setOnClickListener {
            startActivity(Intent(this, HomePageActivity::class.java))
        }

        findViewById<View>(R.id.imageButtonShop)?.setOnClickListener {
            startActivity(Intent(this, ShopMenuActivity::class.java))
        }

        findViewById<View>(R.id.imageButtonAlbumsRecycler)?.setOnClickListener
{
            startActivity(Intent(this, AlbumsMenuActivity::class.java))
        }

        findViewById<View>(R.id.imageButtonSettings)?.setOnClickListener {
            startActivity(Intent(this, SettingsActivity::class.java))
        }
    }

    private fun setupSongButtons() {
        buttons.forEachIndexed { index, (buttonId, audioId) ->
            findViewById<Button>(buttonId).setOnClickListener {
                playAudio(index)
            }
        }
    }

    private fun playAudio(index: Int) {
        val audioUris = buttons.map {
Uri.parse("android.resource://${packageName}/${it.second}") }
        val intent = Intent(this, PlayerActivity::class.java).apply {
            putParcelableArrayListExtra("audioUris", ArrayList(audioUris))
            putExtra("currentIndex", index)
        }
        startActivity(intent)
    }
}
```

Album6Activity

```
package com.example.myapplication

import android.content.ContentResolver
import android.content.Intent
import android.database.Cursor
import android.media.MediaPlayer
import android.net.Uri
import android.os.Bundle
import android.provider.MediaStore
import android.view.View
import android.widget.Button
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity

class Album6Activity : AppCompatActivity() {
    private val buttons = listOf(
        R.id.buttonSong1 to R.raw.dontprayforme,
        R.id.buttonSong2 to R.raw.killingyou,
        R.id.buttonSong3 to R.raw.thedeathofme,
        R.id.buttonSong4 to R.raw.runfree,
        R.id.buttonSong5 to R.raw.breakdownthewalls,
        R.id.buttonSong6 to R.raw.poison,
        R.id.buttonSong7 to R.raw.believe,
        R.id.buttonSong8 to R.raw.creature,
        R.id.buttonSong9 to R.raw.whitelinefever,
        R.id.buttonSong10 to R.raw.movingon,
        R.id.buttonSong11 to R.raw.theroad,
        R.id.buttonSong12 to R.raw.untiltheend,
    )

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_album6)

        setupNavigationButtons()
        setupSongButtons()
    }

    private fun setupNavigationButtons() {
        findViewById<View>(R.id.imageButtonHomePage)?.setOnClickListener {
            startActivity(Intent(this, HomePageActivity::class.java))
        }

        findViewById<View>(R.id.imageButtonShop)?.setOnClickListener {
            startActivity(Intent(this, ShopMenuActivity::class.java))
        }

        findViewById<View>(R.id.imageButtonAlbumsRecycler)?.setOnClickListener
{
            startActivity(Intent(this, AlbumsMenuActivity::class.java))
        }

        findViewById<View>(R.id.imageButtonSettings)?.setOnClickListener {
            startActivity(Intent(this, SettingsActivity::class.java))
        }
    }

    private fun setupSongButtons() {
        buttons.forEachIndexed { index, (buttonId, audioId) ->
            findViewById<Button>(buttonId).setOnClickListener {
                playAudio(index)
            }
        }
    }
```

```
    private fun playAudio(index: Int) {
        val audioUris = buttons.map {
Uri.parse("android.resource://${packageName}/${it.second}") }
        val intent = Intent(this, PlayerActivity::class.java).apply {
            putParcelableArrayListExtra("audioUris", ArrayList(audioUris))
            putExtra("currentIndex", index)
        }
        startActivity(intent)
    }
}
```

Album7Activity

```
package com.example.myapplication

import android.content.ContentResolver
import android.content.Intent
import android.database.Cursor
import android.media.MediaPlayer
import android.net.Uri
import android.os.Bundle
import android.provider.MediaStore
import android.view.View
import android.widget.Button
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity

class Album7Activity : AppCompatActivity() {
    private val buttons = listOf(
        R.id.buttonSong1 to R.raw.welcome,
        R.id.buttonSong2 to R.raw.dearinsanity,
        R.id.buttonSong3 to R.raw.closure,
        R.id.buttonSong4 to R.raw.alessonneverlearned,
        R.id.buttonSong5 to R.raw.tothestage,
        R.id.buttonSong6 to R.raw.dedication,
        R.id.buttonSong7 to R.raw.someonesomewhere,
        R.id.buttonSong8 to R.raw.breathless,
        R.id.buttonSong9 to R.raw.thematch,
        R.id.buttonSong10 to R.raw.anotherbottledown,
        R.id.buttonSong11 to R.raw.recklessrelentless,
        R.id.buttonSong12 to R.raw.morteetdabo,
    )

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_album7)

        setupNavigationButtons()
        setupSongButtons()
    }

    private fun setupNavigationButtons() {
        findViewById<View>(R.id.imageButtonHomePage)?.setOnClickListener {
            startActivity(Intent(this, HomePageActivity::class.java))
        }

        findViewById<View>(R.id.imageButtonShop)?.setOnClickListener {
            startActivity(Intent(this, ShopMenuActivity::class.java))
        }

        findViewById<View>(R.id.imageButtonAlbumsRecycler)?.setOnClickListener
{
            startActivity(Intent(this, AlbumsMenuActivity::class.java))
        }

        findViewById<View>(R.id.imageButtonSettings)?.setOnClickListener {
            startActivity(Intent(this, SettingsActivity::class.java))
        }
```

```
    }

    private fun setupSongButtons() {
        buttons.forEachIndexed { index, (buttonId, audioId) ->
            findViewById<Button>(buttonId).setOnClickListener {
                playAudio(index)
            }
        }
    }

    private fun playAudio(index: Int) {
        val audioUris = buttons.map {
Uri.parse("android.resource://${packageName}/${it.second}") }
        val intent = Intent(this, PlayerActivity::class.java).apply {
            putParcelableArrayListExtra("audioUris", ArrayList(audioUris))
            putExtra("currentIndex", index)
        }
        startActivity(intent)
    }
}
```

Album8Activity

```
package com.example.myapplication

import android.content.ContentResolver
import android.content.Intent
import android.database.Cursor
import android.media.MediaPlayer
import android.net.Uri
import android.os.Bundle
import android.provider.MediaStore
import android.view.View
import android.widget.Button
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity

class Album8Activity : AppCompatActivity() {
    private val buttons = listOf(
        R.id.buttonSong1 to R.raw.alerion,
        R.id.buttonSong2 to R.raw.acandlelitdinnerwithinamorta,
        R.id.buttonSong3 to R.raw.nobodydontdancenomore,
        R.id.buttonSong4 to R.raw.heytheremrbrooks,
        R.id.buttonSong5 to R.raw.hiatus,
        R.id.buttonSong6 to R.raw.ifyoucantridetwohorsesatonce,
        R.id.buttonSong7 to R.raw.asinglemomentofsincerity,
        R.id.buttonSong8 to R.raw.nottheamericanaverage,
        R.id.buttonSong9 to R.raw.iusedtohaveabestfriend,
        R.id.buttonSong10 to R.raw.aprophecy,
        R.id.buttonSong11 to R.raw.iwasoncepossiblymaybeperhapsacowboyking,
        R.id.buttonSong12 to R.raw.wheneverydaystheweekend,
    )

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_album8)

        setupNavigationButtons()
        setupSongButtons()
    }

    private fun setupNavigationButtons() {
        findViewById<View>(R.id.imageButtonHomePage)?.setOnClickListener {
            startActivity(Intent(this, HomePageActivity::class.java))
        }

        findViewById<View>(R.id.imageButtonShop)?.setOnClickListener {
            startActivity(Intent(this, ShopMenuActivity::class.java))
```

```
        }

        findViewById<View>(R.id.imageButtonAlbumsRecycler)?.setOnClickListener
{
            startActivity(Intent(this, AlbumsMenuActivity::class.java))
        }

        findViewById<View>(R.id.imageButtonSettings)?.setOnClickListener {
            startActivity(Intent(this, SettingsActivity::class.java))
        }
    }

    private fun setupSongButtons() {
        buttons.forEachIndexed { index, (buttonId, audioId) ->
            findViewById<Button>(buttonId).setOnClickListener {
                playAudio(index)
            }
        }
    }

    private fun playAudio(index: Int) {
        val audioUris = buttons.map {
Uri.parse("android.resource://${packageName}/${it.second}") }
        val intent = Intent(this, PlayerActivity::class.java).apply {
            putParcelableArrayListExtra("audioUris", ArrayList(audioUris))
            putExtra("currentIndex", index)
        }
        startActivity(intent)
    }
}
```

AlbumsMenuActivity

```
package com.example.myapplication

import android.annotation.SuppressLint
import android.content.Intent
import android.os.Bundle
import android.view.View
import android.widget.Button
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity
import androidx.core.view.ViewCompat
import androidx.core.view.WindowInsetsCompat

class AlbumsMenuActivity : AppCompatActivity() {
    @SuppressLint("WrongViewCast")
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_albums_menu)
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) { v,
insets ->
            val systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top, systemBars.right,
systemBars.bottom)
            insets
        }
        val buttonAlbum1: View? = findViewById(R.id.imageButtonAlbum1)
        if (buttonAlbum1 != null) {
            buttonAlbum1.setOnClickListener {
                val intent = Intent(this, Album1Activity::class.java)
                startActivity(intent)
            }
        }
        val buttonAlbum2: View? = findViewById(R.id.imageButtonAlbum2)
        if (buttonAlbum2 != null) {
            buttonAlbum2.setOnClickListener {
```

```
                val intent = Intent(this, Album2Activity::class.java)
                startActivity(intent)
            }
        }
        val buttonAlbum3: View? = findViewById(R.id.imageButtonAlbum3)
        if (buttonAlbum3 != null) {
            buttonAlbum3.setOnClickListener {
                val intent = Intent(this, Album3Activity::class.java)
                startActivity(intent)
            }
        }
        val buttonAlbum4: View? = findViewById(R.id.imageButtonAlbum4)
        if (buttonAlbum4 != null) {
            buttonAlbum4.setOnClickListener {
                val intent = Intent(this, Album4Activity::class.java)
                startActivity(intent)
            }
        }
        val buttonAlbum5: View? = findViewById(R.id.imageButtonAlbum5)
        if (buttonAlbum5 != null) {
            buttonAlbum5.setOnClickListener {
                val intent = Intent(this, Album5Activity::class.java)
                startActivity(intent)
            }
        }
        val buttonAlbum6: View? = findViewById(R.id.imageButtonAlbum6)
        if (buttonAlbum6 != null) {
            buttonAlbum6.setOnClickListener {
                val intent = Intent(this, Album6Activity::class.java)
                startActivity(intent)
            }
        }
        val buttonAlbum7: View? = findViewById(R.id.imageButtonAlbum7)
        if (buttonAlbum7 != null) {
            buttonAlbum7.setOnClickListener {
                val intent = Intent(this, Album7Activity::class.java)
                startActivity(intent)
            }
        }
        val buttonAlbum8: View? = findViewById(R.id.imageButtonAlbum8)
        if (buttonAlbum8 != null) {
            buttonAlbum8.setOnClickListener {
                val intent = Intent(this, Album8Activity::class.java)
                startActivity(intent)
            }
        }

        val ButtonHomePage: View? = findViewById(R.id.imageButtonHomePage)
        if (ButtonHomePage != null) {
            ButtonHomePage.setOnClickListener {
                val intent = Intent(this, HomePageActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonShop: View? = findViewById(R.id.imageButtonShop)
        if (ButtonShop != null) {
            ButtonShop.setOnClickListener {
                val intent = Intent(this, ShopMenuActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonAlbumsRecycler: View? =
findViewById(R.id.imageButtonAlbumsRecycler)
        if (ButtonAlbumsRecycler != null) {
            ButtonAlbumsRecycler.setOnClickListener {
                val intent = Intent(this, AlbumsMenuActivity::class.java)
                startActivity(intent)
```

```
            }
        }

        val ButtonSettings: View? = findViewById(R.id.imageButtonSettings)
        if (ButtonSettings != null) {
            ButtonSettings.setOnClickListener {
                val intent = Intent(this, SettingsActivity::class.java)
                startActivity(intent)
            }
        }
    }
}
```

HomePageActivity

```
package com.example.myapplication

import android.content.Intent
import android.net.Uri
import android.os.Bundle
import android.view.View
import android.widget.Button
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity
import androidx.core.view.ViewCompat
import androidx.core.view.WindowInsetsCompat

class HomePageActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_home_page)

        val buttonShop: Button = findViewById(R.id.buttonShop)
        buttonShop.setOnClickListener {
            val intent = Intent(this, ShopMenuActivity::class.java)
            startActivity(intent)
        }
        val buttonAloneAgainClip: Button =
findViewById(R.id.buttonAloneAgainClip)
        buttonAloneAgainClip.setOnClickListener {
            val browserIntent = Intent(Intent.ACTION_VIEW,
Uri.parse("https://www.youtube.com/watch?v=k_INM6EUjBI&ab_channel=AskingAlexand
ria"))
            startActivity(browserIntent)
        }
        val buttonNewAlbums: Button = findViewById(R.id.buttonNewAlbums)
        buttonNewAlbums.setOnClickListener {
            val intent = Intent(this, AlbumsMenuActivity::class.java)
            startActivity(intent)
        }

        val ButtonHomePage: View? = findViewById(R.id.imageButtonHomePage)
        if (ButtonHomePage != null) {
            ButtonHomePage.setOnClickListener {
                val intent = Intent(this, HomePageActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonShop: View? = findViewById(R.id.imageButtonShop)
        if (ButtonShop != null) {
            ButtonShop.setOnClickListener {
                val intent = Intent(this, ShopMenuActivity::class.java)
                startActivity(intent)
            }
        }
```

```
        val ButtonAlbumsRecycler: View? =
findViewById(R.id.imageButtonAlbumsRecycler)
        if (ButtonAlbumsRecycler != null) {
            ButtonAlbumsRecycler.setOnClickListener {
                val intent = Intent(this, AlbumsMenuActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonSettings: View? = findViewById(R.id.imageButtonSettings)
        if (ButtonSettings != null) {
            ButtonSettings.setOnClickListener {
                val intent = Intent(this, SettingsActivity::class.java)
                startActivity(intent)
            }
        }

    }
}
```

LoadingPageActivity

```
package com.example.myapplication

import android.content.Intent
import android.os.Bundle
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity
import androidx.core.view.ViewCompat
import androidx.core.view.WindowInsetsCompat
import kotlinx.coroutines.CoroutineScope
import kotlinx.coroutines.Dispatchers
import kotlinx.coroutines.delay
import kotlinx.coroutines.launch

class LoadingPage : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_loading_page)
        val handler = CoroutineScope(Dispatchers.Main).launch {
            delay(1000)
            val intent = Intent(this@LoadingPage, HomePageActivity
::class.java)
            startActivity(intent)
            finish()
        }
    }
}
```

ShopMenuActivity

```
package com.example.myapplication

import android.annotation.SuppressLint
import android.content.Intent
import android.os.Bundle
import android.view.View
import android.widget.Button
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity
import androidx.core.view.ViewCompat
import androidx.core.view.WindowInsetsCompat

class ShopMenuActivity : AppCompatActivity() {
    @SuppressLint("MissingInflatedId")
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_shop_menu)
```

```
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) { v,
insets ->
            val systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top, systemBars.right,
systemBars.bottom)
            insets
        }
        val buttonWhatsNew: Button = findViewById(R.id.buttonWhatsNew)
        buttonWhatsNew.setOnClickListener {
            val intent = Intent(this, WhatsNewMenuActivity::class.java)
            startActivity(intent)
        }
        val buttonSale: Button = findViewById(R.id.buttonSale)
        buttonSale.setOnClickListener {
            val intent = Intent(this, SaleMenuActivity::class.java)
            startActivity(intent)
        }
        val buttonSettings: Button = findViewById(R.id.buttonSettings)
        buttonSettings.setOnClickListener {
            val intent = Intent(this, SettingsActivity::class.java)
            startActivity(intent)
        }
        val buttonTourDates: Button = findViewById(R.id.buttonTourDates)
        buttonTourDates.setOnClickListener {
            val intent = Intent(this, TourDatesMenuActivity::class.java)
            startActivity(intent)
        }
        val buttonBlackTeePrice: Button =
findViewById(R.id.buttonPriceMetalTee)
        buttonBlackTeePrice.setOnClickListener {
            val intent = Intent(this, BlackTeeActivity::class.java)
            startActivity(intent)
        }

        val buttonAlbums: Button = findViewById(R.id.buttonAlbums)
        buttonAlbums.setOnClickListener {
            val intent = Intent(this, AlbumsMenuActivity::class.java)
            startActivity(intent)
        }

        val ButtonHomePage: View? = findViewById(R.id.imageButtonHomePage)
        if (ButtonHomePage != null) {
            ButtonHomePage.setOnClickListener {
                val intent = Intent(this, HomePageActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonShop: View? = findViewById(R.id.imageButtonShop)
        if (ButtonShop != null) {
            ButtonShop.setOnClickListener {
                val intent = Intent(this, ShopMenuActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonAlbumsRecycler: View? =
findViewById(R.id.imageButtonAlbumsRecycler)
        if (ButtonAlbumsRecycler != null) {
            ButtonAlbumsRecycler.setOnClickListener {
                val intent = Intent(this, AlbumsMenuActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonSettings: View? = findViewById(R.id.imageButtonSettings)
        if (ButtonSettings != null) {
            ButtonSettings.setOnClickListener {
```

```
                val intent = Intent(this, SettingsActivity::class.java)
                startActivity(intent)
            }
        }

    }
}
```

WhatsNewActivity

```
package com.example.myapplication

import android.annotation.SuppressLint
import android.content.Intent
import android.os.Bundle
import android.view.View
import android.widget.Button
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity
import androidx.core.view.ViewCompat
import androidx.core.view.WindowInsetsCompat

class WhatsNewMenuActivity : AppCompatActivity() {
    @SuppressLint("MissingInflatedId")
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_whats_new_menu)
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) { v,
insets ->
            val systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top, systemBars.right,
systemBars.bottom)
            insets
        }
        val buttonPriceVeil: Button = findViewById(R.id.buttonCDLP)
        buttonPriceVeil.setOnClickListener {
            val intent = Intent(this, GoodsMenuActivity::class.java)
            startActivity(intent)
        }
        val buttonHoodiePrice: Button = findViewById(R.id.buttonMetalTourTee)
        buttonHoodiePrice.setOnClickListener {
            val intent = Intent(this, HoodieMenuActivity::class.java)
            startActivity(intent)
        }
        val ButtonHomePage: View? = findViewById(R.id.imageButtonHomePage)
        if (ButtonHomePage != null) {
            ButtonHomePage.setOnClickListener {
                val intent = Intent(this, HomePageActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonShop: View? = findViewById(R.id.imageButtonShop)
        if (ButtonShop != null) {
            ButtonShop.setOnClickListener {
                val intent = Intent(this, ShopMenuActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonAlbumsRecycler: View? =
findViewById(R.id.imageButtonAlbumsRecycler)
        if (ButtonAlbumsRecycler != null) {
            ButtonAlbumsRecycler.setOnClickListener {
                val intent = Intent(this, AlbumsMenuActivity::class.java)
                startActivity(intent)
            }
```

```
        }

        val ButtonSettings: View? = findViewById(R.id.imageButtonSettings)
        if (ButtonSettings != null) {
            ButtonSettings.setOnClickListener {
                val intent = Intent(this, SettingsActivity::class.java)
                startActivity(intent)
            }
        }

        val buttonCDprice: Button = findViewById(R.id.buttonCDprice)
        buttonCDprice.setOnClickListener {
            val intent = Intent(this, NewAlbumCDActivity::class.java)
            startActivity(intent)
        }

        val buttonNewDesignTeePrice: Button =
findViewById(R.id.buttonNewDesignTeePrice)
        buttonNewDesignTeePrice.setOnClickListener {
            val intent = Intent(this, NewDesignTeeActivity::class.java)
            startActivity(intent)
        }

        val buttonHoodieMock: Button = findViewById(R.id.buttonHoodieMock)
        buttonHoodieMock.setOnClickListener {
            val intent = Intent(this, HoodieMockActivity::class.java)
            startActivity(intent)
        }

        val buttonLongSleeve: Button = findViewById(R.id.buttonLongSleeve)
        buttonLongSleeve.setOnClickListener {
            val intent = Intent(this, LongSleeveActivity::class.java)
            startActivity(intent)
        }
    }
}
```

SaleMenuActivity

```
package com.example.myapplication

import android.annotation.SuppressLint
import android.content.Intent
import android.os.Bundle
import android.view.View
import android.widget.Button
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity
import androidx.core.view.ViewCompat
import androidx.core.view.WindowInsetsCompat

class SaleMenuActivity : AppCompatActivity() {
    @SuppressLint("MissingInflatedId")
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_sale_menu)
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) { v,
insets ->
            val systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top, systemBars.right,
systemBars.bottom)
            insets
        }
        val buttonVeilTeePrice: Button = findViewById(R.id.buttonVeilTeePrice)
        buttonVeilTeePrice.setOnClickListener {
            val intent = Intent(this, VeilTeeActivity::class.java)
            startActivity(intent)
```

```
        }
        val buttonPriceMetalTee: Button = findViewById(R.id.buttonTeePrice)
        buttonPriceMetalTee.setOnClickListener {
            val intent = Intent(this, BlackTeeActivity::class.java)
            startActivity(intent)
        }

        val buttonBACKWEBMOCK: Button = findViewById(R.id.buttonBACKWEBMOCK)
        buttonBACKWEBMOCK.setOnClickListener {
            val intent = Intent(this, back_web_mock_activity::class.java)
            startActivity(intent)
        }

        val buttonfrontbackmock: Button =
findViewById(R.id.buttonfrontbackmock)
        buttonfrontbackmock.setOnClickListener {
            val intent = Intent(this, FrontBackMockActivity::class.java)
            startActivity(intent)
        }

        val buttonfrontwebmock: Button = findViewById(R.id.buttonfrontwebmock)
        buttonfrontwebmock.setOnClickListener {
            val intent = Intent(this, FrontWebMockActivity::class.java)
            startActivity(intent)
        }

        val buttonaskingalexandriahoodie: Button =
findViewById(R.id.buttonaskingalexandriahoodie)
        buttonaskingalexandriahoodie.setOnClickListener {
            val intent = Intent(this, BaseHoodieActivity::class.java)
            startActivity(intent)
        }

        val ButtonHomePage: View? = findViewById(R.id.imageButtonHomePage)
        if (ButtonHomePage != null) {
            ButtonHomePage.setOnClickListener {
                val intent = Intent(this, HomePageActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonShop: View? = findViewById(R.id.imageButtonShop)
        if (ButtonShop != null) {
            ButtonShop.setOnClickListener {
                val intent = Intent(this, ShopMenuActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonAlbumsRecycler: View? =
findViewById(R.id.imageButtonAlbumsRecycler)
        if (ButtonAlbumsRecycler != null) {
            ButtonAlbumsRecycler.setOnClickListener {
                val intent = Intent(this, AlbumsMenuActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonSettings: View? = findViewById(R.id.imageButtonSettings)
        if (ButtonSettings != null) {
            ButtonSettings.setOnClickListener {
                val intent = Intent(this, SettingsActivity::class.java)
                startActivity(intent)
            }
        }
    }
}
```

SettingsMenuActivity

```
package com.example.myapplication

import android.content.Intent
import android.net.Uri
import android.os.Bundle
import android.view.View
import android.widget.Button
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity
import androidx.core.view.ViewCompat
import androidx.core.view.WindowInsetsCompat

class SettingsActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_settings)
        ViewCompat.setOnApplyWindowInsetsListener(findViewById(R.id.main)) { v,
insets ->
            val systemBars =
insets.getInsets(WindowInsetsCompat.Type.systemBars())
            v.setPadding(systemBars.left, systemBars.top, systemBars.right,
systemBars.bottom)
            insets
        }

        val ButtonHomePage: View? = findViewById(R.id.imageButtonHomePage)
        if (ButtonHomePage != null) {
            ButtonHomePage.setOnClickListener {
                val intent = Intent(this, ShopMenuActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonShop: View? = findViewById(R.id.imageButtonShop)
        if (ButtonShop != null) {
            ButtonShop.setOnClickListener {
                val intent = Intent(this, ShopMenuActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonAlbumsRecycler: View? =
findViewById(R.id.imageButtonAlbumsRecycler)
        if (ButtonAlbumsRecycler != null) {
            ButtonAlbumsRecycler.setOnClickListener {
                val intent = Intent(this, AlbumsMenuActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonSettings: View? = findViewById(R.id.imageButtonSettings)
        if (ButtonSettings != null) {
            ButtonSettings.setOnClickListener {
                val intent = Intent(this, SettingsActivity::class.java)
                startActivity(intent)
            }
        }

        val buttonTermsAndConditions: Button =
findViewById(R.id.buttonTermsAndConditions)
        buttonTermsAndConditions.setOnClickListener {
            val intent = Intent(this, TermsAndConditions::class.java)
            startActivity(intent)
        }
        val buttonPrivacyPolicy: Button =
findViewById(R.id.buttonPrivacyPolicy)
        buttonPrivacyPolicy.setOnClickListener {
```

```
            val intent = Intent(this, PrivacyPolicy::class.java)
            startActivity(intent)
        }
        val buttonRefundPolicy: Button = findViewById(R.id.buttonRefundPolicy)
        buttonRefundPolicy.setOnClickListener {
            val intent = Intent(this, RefundPolicy::class.java)
            startActivity(intent)
        }
        val buttonContactUs: Button = findViewById(R.id.buttonContactUs)
        buttonContactUs.setOnClickListener {
            val intent = Intent(this, ContactUS::class.java)
            startActivity(intent)
        }
        val buttonAskingAlexandria: Button =
findViewById(R.id.buttonSiteAsking)
        buttonAskingAlexandria.setOnClickListener {
            val browserIntent = Intent(Intent.ACTION_VIEW,
Uri.parse("https://store.askingalexandria.com/"))
            startActivity(browserIntent)
        }
        val buttonAskingAlexandriaShop: Button =
findViewById(R.id.buttonShopAsking)
        buttonAskingAlexandriaShop.setOnClickListener {
            val browserIntent = Intent(Intent.ACTION_VIEW,
Uri.parse("https://store.askingalexandria.com/collections/all-products"))
            startActivity(browserIntent)
        }
        val buttonAppleMusic: Button = findViewById(R.id.buttonAppleMusic)
        buttonAppleMusic.setOnClickListener {
            val browserIntent = Intent(Intent.ACTION_VIEW,
Uri.parse("https://music.apple.com/ru/artist/asking-alexandria/329938300"))
            startActivity(browserIntent)
        }
        val buttonSpotify: Button = findViewById(R.id.buttonSpotify)
        buttonSpotify.setOnClickListener {
            val browserIntent = Intent(Intent.ACTION_VIEW,
Uri.parse("https://open.spotify.com/artist/1caBfBEapzw8z2Qz9q0OaQ"))
            startActivity(browserIntent)
        }


    }
}
```

TourDatesMenuActivity

```
package com.example.myapplication

import android.content.Intent
import android.net.Uri
import android.os.Bundle
import android.view.View
import android.widget.Button
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity
import androidx.core.view.ViewCompat
import androidx.core.view.WindowInsetsCompat

class TourDatesMenuActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_tour_dates_menu)

        val ButtonHomePage: View? = findViewById(R.id.imageButtonHomePage)
        if (ButtonHomePage != null) {
            ButtonHomePage.setOnClickListener {
                val intent = Intent(this, HomePageActivity::class.java)
                startActivity(intent)
```

```
            }
        }

        val ButtonShop: View? = findViewById(R.id.imageButtonShop)
        if (ButtonShop != null) {
            ButtonShop.setOnClickListener {
                val intent = Intent(this, ShopMenuActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonAlbumsRecycler: View? =
findViewById(R.id.imageButtonAlbumsRecycler)
        if (ButtonAlbumsRecycler != null) {
            ButtonAlbumsRecycler.setOnClickListener {
                val intent = Intent(this, AlbumsMenuActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonSettings: View? = findViewById(R.id.imageButtonSettings)
        if (ButtonSettings != null) {
            ButtonSettings.setOnClickListener {
                val intent = Intent(this, SettingsActivity::class.java)
                startActivity(intent)
            }
        }

        val buttonDate1: Button = findViewById(R.id.buttonDate1)
        buttonDate1.setOnClickListener {
            val browserIntent = Intent(Intent.ACTION_VIEW,
Uri.parse("https://www.last.fm/ru/event/4793364+Asking+Alexandria+at+Maxx+Watts
+on+20+March+2024"))
            startActivity(browserIntent)
        }

        val buttonDate2: Button = findViewById(R.id.buttonDate2)
        buttonDate2.setOnClickListener {
            val browserIntent = Intent(Intent.ACTION_VIEW,
Uri.parse("https://www.last.fm/ru/festival/4774114+Knotfest+Australia+2024"))
            startActivity(browserIntent)
        }

        val buttonDate3: Button = findViewById(R.id.buttonDate3)
        buttonDate3.setOnClickListener {
            val browserIntent = Intent(Intent.ACTION_VIEW,
Uri.parse("https://www.last.fm/ru/event/4774348+Knotfest+Australia+2024"))
            startActivity(browserIntent)
        }

        val buttonDate4: Button = findViewById(R.id.buttonDate4)
        buttonDate4.setOnClickListener {
            val browserIntent = Intent(Intent.ACTION_VIEW,
Uri.parse("https://www.last.fm/ru/festival/4774342+Knotfest+Australia+2024"))
            startActivity(browserIntent)
        }

        val buttonDate5: Button = findViewById(R.id.buttonDate5)
        buttonDate5.setOnClickListener {
            val browserIntent = Intent(Intent.ACTION_VIEW,
Uri.parse("https://www.last.fm/ru/event/4794343+Wage+War+at+The+Triffid+on+26+M
arch+2024"))
            startActivity(browserIntent)
        }

        val buttonDate6: Button = findViewById(R.id.buttonDate6)
        buttonDate6.setOnClickListener {
            val browserIntent = Intent(Intent.ACTION_VIEW,
Uri.parse("https://www.last.fm/ru/event/4812579+Asking+Alexandria:+All+My+Frien
```

```
ds+North+American+Tour+2024"))
            startActivity(browserIntent)
        }

        val buttonDate7: Button = findViewById(R.id.buttonDate7)
        buttonDate7.setOnClickListener {
            val browserIntent = Intent(Intent.ACTION_VIEW,
Uri.parse("https://www.last.fm/ru/event/4771097+Hog+Fest+starring+Staind+2024")
)
            startActivity(browserIntent)
        }

        val buttonDate8: Button = findViewById(R.id.buttonDate8)
        buttonDate8.setOnClickListener {
            val browserIntent = Intent(Intent.ACTION_VIEW,
Uri.parse("https://www.last.fm/ru/event/4799255+All+My+Friends+Tour"))
            startActivity(browserIntent)
        }

        val buttonDate9: Button = findViewById(R.id.buttonDate9)
        buttonDate9.setOnClickListener {
            val browserIntent = Intent(Intent.ACTION_VIEW,
Uri.parse("https://www.last.fm/ru/event/4812611+All+My+Friends+Tour"))
            startActivity(browserIntent)
        }

        val buttonDate10: Button = findViewById(R.id.buttonDate10)
        buttonDate10.setOnClickListener {
            val browserIntent = Intent(Intent.ACTION_VIEW,
Uri.parse("https://www.last.fm/ru/festival/4768121+Slam+Dunk+Festival+2024+-
+South"))
            startActivity(browserIntent)
        }

        val buttonDate11: Button = findViewById(R.id.buttonDate11)
        buttonDate11.setOnClickListener {
            val browserIntent = Intent(Intent.ACTION_VIEW,
Uri.parse("https://www.last.fm/ru/festival/4768124+Slam+Dunk+Festival+2024+-
+North"))
            startActivity(browserIntent)
        }

    }
}
```

PlayerMenuActivity

```
package com.example.myapplication

import android.content.Intent
import android.media.AudioManager
import android.media.MediaPlayer
import android.net.Uri
import android.os.Bundle
import android.os.Handler
import android.view.View
import android.widget.ImageButton
import android.widget.SeekBar
import android.widget.TextView
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity

class PlayerActivity : AppCompatActivity() {
    private lateinit var mediaPlayer: MediaPlayer
    private lateinit var durationPlayed: TextView
    private lateinit var durationTotal: TextView
    private lateinit var seekBar: SeekBar
    private lateinit var volumeSeekBar: SeekBar
    private val handler = Handler()
```

```
    private lateinit var audioUris: List<Uri>
    private var currentIndex = 0

    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_player)


        durationPlayed = findViewById(R.id.textViewTimeStart)
        durationTotal = findViewById(R.id.textViewTimeEnd)
        seekBar = findViewById(R.id.seekBar)
        volumeSeekBar = findViewById(R.id.volumeSeekBar)
        val playButton: ImageButton = findViewById(R.id.buttonPlayPause)
        val nextButton: ImageButton = findViewById(R.id.buttonforward)
        val prevButton: ImageButton = findViewById(R.id.buttonback)

        var isPlaying = true


        audioUris = intent.getParcelableArrayListExtra("audioUris")!!
        currentIndex = intent.getIntExtra("currentIndex", 0)


        val songName: TextView = findViewById(R.id.textViewNameSong)
        songName.text = "AskingAlexandria"




        seekBar.setOnSeekBarChangeListener(object :
SeekBar.OnSeekBarChangeListener {
            override fun onProgressChanged(seekBar: SeekBar?, progress: Int,
fromUser: Boolean) {
                if (fromUser) {
                    mediaPlayer.seekTo(progress * 1000)
                }
            }

            override fun onStartTrackingTouch(seekBar: SeekBar?) {}
            override fun onStopTrackingTouch(seekBar: SeekBar?) {}
        })


        playAudio()


        playButton.setOnClickListener {
            if (isPlaying) {
                mediaPlayer.pause()
            } else {
                mediaPlayer.start()
                updateSeekBar()
            }
            isPlaying = !isPlaying
        }


        nextButton.setOnClickListener {
            if (currentIndex < audioUris.size - 1) {
                currentIndex++
                playAudio()
            }
        }


        prevButton.setOnClickListener {
            if (currentIndex > 0) {
```

```
                currentIndex--
                playAudio()
            }
        }


        setupVolumeControl()


        updateSeekBar()

        val ButtonHomePage: View? = findViewById(R.id.imageButtonHomePage)
        if (ButtonHomePage != null) {
            ButtonHomePage.setOnClickListener {
                val intent = Intent(this, HomePageActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonShop: View? = findViewById(R.id.imageButtonShop)
        if (ButtonShop != null) {
            ButtonShop.setOnClickListener {
                val intent = Intent(this, ShopMenuActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonAlbumsRecycler: View? =
findViewById(R.id.imageButtonAlbumsRecycler)
        if (ButtonAlbumsRecycler != null) {
            ButtonAlbumsRecycler.setOnClickListener {
                val intent = Intent(this, AlbumsMenuActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonSettings: View? = findViewById(R.id.imageButtonSettings)
        if (ButtonSettings != null) {
            ButtonSettings.setOnClickListener {
                val intent = Intent(this, SettingsActivity::class.java)
                startActivity(intent)
            }
        }
    }



    private fun playAudio() {

        if (this::mediaPlayer.isInitialized) {
            mediaPlayer.release()
        }

        mediaPlayer = MediaPlayer.create(this, audioUris[currentIndex])
        mediaPlayer.start()

        seekBar.max = mediaPlayer.duration / 1000
        durationTotal.text = formatTime(mediaPlayer.duration / 1000)

        mediaPlayer.setOnCompletionListener {
            if (currentIndex < audioUris.size - 1) {
                currentIndex++
                playAudio()
            }
        }

        updateSeekBar()
    }
```

```
    private fun setupVolumeControl() {

        val audioManager = getSystemService(AUDIO_SERVICE) as AudioManager
        val maxVolume =
audioManager.getStreamMaxVolume(AudioManager.STREAM_MUSIC)
        val currentVolume =
audioManager.getStreamVolume(AudioManager.STREAM_MUSIC)


        volumeSeekBar.max = maxVolume
        volumeSeekBar.progress = currentVolume


        volumeSeekBar.setOnSeekBarChangeListener(object :
SeekBar.OnSeekBarChangeListener {
            override fun onProgressChanged(seekBar: SeekBar?, progress: Int,
fromUser: Boolean) {
                if (fromUser) {
                    audioManager.setStreamVolume(AudioManager.STREAM_MUSIC,
progress, 0)
                }
            }

            override fun onStartTrackingTouch(seekBar: SeekBar?) {}
            override fun onStopTrackingTouch(seekBar: SeekBar?) {}
        })
    }

    private fun updateSeekBar() {

        handler.postDelayed({
            if (mediaPlayer.isPlaying) {
                val currentPosition = mediaPlayer.currentPosition / 1000
                seekBar.progress = currentPosition
                durationPlayed.text = formatTime(currentPosition)
                updateSeekBar()
            }
        }, 100)
    }

    private fun formatTime(seconds: Int): String {

        val minutes = (seconds / 60) % 60
        val secondsRemaining = seconds % 60
        return String.format("%02d:%02d", minutes, secondsRemaining)
    }

    override fun onDestroy() {
        super.onDestroy()

        mediaPlayer.release()
    }
}
```

GoodsMenuActivity

```
package com.example.myapplication

import android.content.Intent
import android.os.Bundle
import android.view.View
import androidx.activity.enableEdgeToEdge
import androidx.appcompat.app.AppCompatActivity
import androidx.core.view.ViewCompat
import androidx.core.view.WindowInsetsCompat

class GoodsMenuActivity : AppCompatActivity() {
    override fun onCreate(savedInstanceState: Bundle?) {
        super.onCreate(savedInstanceState)
        enableEdgeToEdge()
        setContentView(R.layout.activity_goods_menu)

        val ButtonHomePage: View? = findViewById(R.id.imageButtonHomePage)
        if (ButtonHomePage != null) {
            ButtonHomePage.setOnClickListener {
                val intent = Intent(this, HomePageActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonShop: View? = findViewById(R.id.imageButtonShop)
        if (ButtonShop != null) {
            ButtonShop.setOnClickListener {
                val intent = Intent(this, ShopMenuActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonAlbumsRecycler: View? =
findViewById(R.id.imageButtonAlbumsRecycler)
        if (ButtonAlbumsRecycler != null) {
            ButtonAlbumsRecycler.setOnClickListener {
                val intent = Intent(this, AlbumsMenuActivity::class.java)
                startActivity(intent)
            }
        }

        val ButtonSettings: View? = findViewById(R.id.imageButtonSettings)
        if (ButtonSettings != null) {
            ButtonSettings.setOnClickListener {
                val intent = Intent(this, SettingsActivity::class.java)
                startActivity(intent)
            }
        }
    }
}
```