

**НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ
імені ІГОРЯ СІКОРСЬКОГО»**

Факультет інформатики та обчислювальної техніки

Кафедра інформаційних систем та технологій

«На правах рукопису»
УДК 004.891

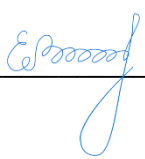
До захисту допущено:
Завідувач кафедри
_____ Олександр РОЛІК
«___» _____ 2022 р.

**Магістерська дисертація
на здобуття ступеня магістра
за освітньо-професійною програмою «Інформаційні управляючі
системи та технології»
зі спеціальності 126 «Інформаційні системи та технології»
на тему: «Інформаційна система для написання музичних творів»**

Виконала:
студентка VI курсу, групи ІС-12мп
Дмитрук Олеся Юріївна

Керівник:
доцент, к.ф.-м.н., доцент
Гавриленко Олена Валеріївна

Рецензент:
доцент кафедри ІІІ, к.т.н.,
Олійник Юрій Олександрович





Засвідчую, що у цій магістерській
дисертації немає запозичень з праць
інших авторів без відповідних
посилань.
Студентка _____

Київ – 2022 року

Національний технічний університет України
«Київський політехнічний інститут імені Ігоря Сікорського»
Факультет інформатики та обчислювальної техніки
Кафедра інформаційних систем та технологій

Рівень вищої освіти – другий (магістерський)

Спеціальність – 126 «Інформаційні системи та технології»

Освітньо-професійна програма «Інформаційні управляючі системи та технології»

ЗАТВЕРДЖУЮ

Завідувач кафедри

_____ Олександр РОЛІК

«___» _____ 2022 р.

ЗАВДАННЯ
на магістерську дисертацію студенту

Дмитрук Олесі Юріївні

1. Тема дисертації «Інформаційна система для написання музичних творів», науковий керівник дисертації Гавриленко Олена Валеріївна, к.ф.-м.н., доцент, затверджені наказом по університету від «09» 11 2022 р. № 4115-с
2. Термін подання студентом дисертації _____
3. Об'єкт дослідження – інформаційна система для генерації музичних творів на основі користувацьких референсів та параметрів.
4. Вихідні дані – результати роботи алгоритмів для генерації музики.
5. Перелік завдань, які потрібно розробити:
6. Орієнтовний перелік графічного (ілюстративного) матеріалу: діаграма класів, результати експериментів для вибору алгоритму, топологія нейронної мережі, оцінка результатів методу ланцюгів Маркова, графіки роботи генеративно-змагальної нейронної мережі, діаграма класів, діаграма послідовності, діаграма розгортання.
7. Орієнтовний перелік публікацій – 1 публікація
8. Дата видачі завдання _____

Календарний план

№ з/п	Назва етапів виконання магістерської дисертації	Термін виконання етапів магістерської дисертації	Примітка
1	Аналіз предметного середовища		
2	Розробка сценаріїв використання системи		
3	Огляд наявних математичних підходів		
4	Розробка математичного забезпечення		
5	Оцінка ефективності роботи алгоритмів		
6	Розробка програмного забезпечення		
7	Оформлення пояснювальної записки		
8	Подання роботи на захист		

Студент

Олеся ДМИТРУК

Науковий керівник

Олена ГАВРИЛЕНКО

АНОТАЦІЯ

Структура та обсяг роботи. Пояснювальна записка складається з чотирьох розділів, містить 23 рисунки, 38 таблиць, 9 додатків, 26 джерел.

Актуальність теми. Актуальність теми роботи полягає у рості потреби людей у генерації музичного супроводу для творчої та комерційної діяльності. Автоматична генерація дозволить отримувати унікальні музичні фрагменти, що дозволяють виділити себе або свій продукт, зекономити гроші на покупці ліцензій на використання готових композицій чи зверненні до професійних композиторів.

Мета дослідження. Метою створення системи є надання користувачам можливості автоматизованого створення унікальних мелодій для задоволення їхніх комерційних та особистих потреб.

Об'єкт дослідження. Інформаційна система для генерації музичних творів на основі користувацьких референсів та параметрів.

Предмет дослідження. Застосування методів ланцюгів Маркова та генеративно-змагальних нейронних мереж для генерації музичних творів.

Для реалізації мети поставлено наступні завдання:

- розробити алгоритми для генерації музичних творів на основі вхідного фрагменту;
- розробити користувацький інтерфейс.

Практичне значення отриманих результатів полягає у тому, що користувачі зможуть генерувати унікальні музичні фрагменти, котрі в подальшому можуть бути використаними для їхньої творчої та комерційної діяльності.

Публікації. Гавриленко О.В. Використання математичних моделей та засобів штучного інтелекту для автоматизації написання музичних творів / О.В. Гавриленко, О.Ю. Дмитрук, А.В. Тетерук // Вісник Національного транспортного університету. Серія «Технічні науки». Науковий журнал. – К.: НТУ, 2022. – 3 (53) (прийнято до друку) [1].

Ключові слова. ГЕНЕРАЦІЯ МУЗИКИ, МАРКІВСЬКІ ЛАНЦЮГИ, ГЕНЕРАТИВНО-ЗМАГАЛЬНІ МЕРЕЖІ

ABSTRACT

Structure and scope of work. The explanatory note consists of four sections, contains 23 figures, 38 tables, 9 appendices, and 26 sources.

Topicality. The relevance of the topic of the work lies in the growing need of people to generate musical accompaniment for creative and commercial activities. Automatic generation will allow you to receive unique musical fragments that allow you to distinguish yourself or your product, save money on buying licenses for the use of ready-made compositions or contacting professional composers.

The aim of the study. The goal of the system is to enable users to create automatically unique ringtones to meet their commercial and personal needs.

Object of study. An information system for generating musical works based on user references and parameters.

Subject of study. Application of methods of Markov chains and generative-competitive neural networks for the generation of musical works.

To achieve the goal, the following tasks have been set:

- to develop algorithms for the generation of music based on the input fragment;
- to develop a user interface.

The practical significance of the obtained results is that users will be able to generate unique musical fragments based on the references and entered parameters, which can be used for their creative and commercial activities in the future.

Publications. Olena Gavrilenko, Olesia Dmytruk, Anastasia Teteruk. Mathematical models and artificial intelligence tools usage for automation of writing musical compositions. Visnyk National Transport University. Series «Technical Sciences». Scientific journal. – Kyiv: National Transport University, 2022. – Issue 3 (53).

Key words. MUSIC GENERATION, MARKOV CHAINS, GAN

ЗМІСТ

ВСТУП.....	9
1 ОПИС ПРЕДМЕТНОЇ ОБЛАСТІ	10
1.1 Аналіз предметного середовища	10
1.2 Огляд наявних аналогів	11
1.2.1 Jukebox від OpenAI.....	11
1.2.2 Magenta від Tensorflow	12
1.2.3 Soudraw	14
1.3 Огляд пов'язаних досліджень	14
1.3.1 A Comprehensive Survey on Deep Music generation: Multi-level Representations, Algorithms, Evaluations, and Future Directions	14
1.3.2 Personalized Popular Music Generation Using Imitation and Structure	15
1.3.3 Symbolic Music Generation with Transformer-GANs.....	16
1.4 Постановка задачі.....	16
1.4.1 Призначення розробки.....	16
1.4.2 Цілі та задачі розробки	17
1.5 Опис процесу діяльності.....	17
1.6 Інформаційне забезпечення.....	22
Висновки до розділу.....	23
2 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ	24
2.1 Огляд підходів до вирішення задачі генерації музики.....	24
2.1.1 Стохастичні процеси.....	24
2.1.2 Граматики.....	24
2.1.3 Оптимізаційні методи	25

2.1.4 Штучний інтелект та машинне навчання	28
2.2 Обґрунтування вибору використаних методів та їхній опис.....	35
2.2.1 Побудова ланцюга Маркова.....	36
2.2.2 Топологія генеративно-змагальної нейронної мережі	38
2.2.3 Огляд результатів роботи використаних методів	42
Висновки до розділу.....	47
3 ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ.....	48
3.1 Опис засобів розробки	48
3.2 Архітектура програмного забезпечення	51
3.2.1 Опис класів.....	51
3.2.2 Сценарій роботи	51
3.2.3 Розгортання.....	61
3.2.4 Специфікація методів	61
3.3 Інструкція користувача.....	65
3.3.1 Опис екранних форм.....	65
3.3.2 Послідовність кроків виконання.....	73
Висновки до розділу.....	74
4 РОЗРОБЛЕННЯ СТАРТАП-ПРОЄКТУ	75
4.1 Опис ідеї проєкту	75
4.2 Технологічний аудит ідеї проєкту	79
4.3 Аналіз ринкових можливостей запуску стартап-проєкту	81
4.4 Розроблення ринкової стратегії проєкту.....	89
4.5 Розроблення маркетингової програми стартап-проєкту	93
Висновки до розділу.....	97
ВИСНОВКИ.....	98

СПИСОК ЛІТЕРАТУРИ.....	99
ДОДАТОК А.....	102
ДОДАТОК Б.....	103
ДОДАТОК В.....	104
ДОДАТОК Г.....	105
ДОДАТОК Д.....	106
ДОДАТОК Е.....	107
ДОДАТОК Ж.....	108
ДОДАТОК И.....	109
ДОДАТОК К. ЛІСТИНГ ЧАСТИНИ КОДУ	102

ВСТУП

Генерація музики є однією з класичних задач застосування штучного інтелекту у мистецтві та для створення розважального контенту. Донедавна проблема автоматичного створення композицій комп'ютерними засобами полягала у виборі підходу до представлення та формалізації музичної інформації, оскільки теорія музики не є однозначною. Проте, такі розробки як поява представлення даних MIDI та програмних інструментів для аналізу музики дозволили отримувати з аудіофайлів структуровані дані, які є зручними для аналізу та перетворення.

Актуальність автоматизації процесу створення музики пов'язана з популяризацією музичних творів у соціальних мережах: накладання їх на відео, рекламу тощо. Згенерований аудіоматеріал допоможе блогерам та підприємцям уникнути проблем з авторським правом і дозволить створювати унікальний супровід контенту.

У даній роботі запропоновано варіант веб-застосунку, що надає користувачам можливість генерувати файл за референсними композиціями або обраними вибірками. Для генерації використано підходи застосування ланцюгів Маркова та нейронних мереж.

1 ОПИС ПРЕДМЕТНОЇ ОБЛАСТІ

1.1 Аналіз предметного середовища

Протягом кожного історичного етапу розвитку людської цивілізації змінювався зміст музики та розширювався спектр варіацій музичних форм. У сучасному світі композиції вийшли за рамки звичайних елементів культури та стали невід'ємною складовою комерції.

Агенція Kantar провела дослідження рекламного контенту на різних інтернет-платформах, яке показало, що музика використовується у 84% оголошень [2]. Музика не обов'язково має проявляти будь-яку особливу спорідненість з певним продуктом або послугою, щоб відігравати ефективну та корисну функцію [3]. Зазвичай на дії цільової аудиторії рекламного ролику впливає настрій музики, наприклад, доведено, що ліричні та спокійні мелодії краще пробуджують бажання придбати товар, ніж веселі композиції [4].

Розглянуті дослідження показують, що рекламодавцям важливо мати змогу створювати унікальні короткі музичні твори та контролювати стиль та настрій музики для встановлення емоційного зв'язку з клієнтом, що підвищить ефективність оголошень, розміщених в мережі Інтернет. Компанії малого бізнесу, які тільки починають свій розвиток, зазвичай не мають змоги звернутися до професійного композитора чи викупити права на використання готових композицій у своїй рекламі.

Окрім цього, музика активно використовується у контенті блогерів у соціальних мережах та на стрімінгових платформах. Задля дотримання законодавства пов'язаного з захистом авторського права, платформи дозволяють використовувати лише композиції з безкоштовної фонотеки (які не завжди підходять користувачеві) або купувати ліцензію на використання музики, середня вартість якої становить 50\$.

У зв'язку з цим, є актуальною ідея створення застосунку, де користувачі матимуть змогу генерувати свої унікальні короткотривалі музичні фрагменти. Це допоможе користувачам зекономити кошти на початку просування проєктів (блогів,

бізнесу тощо) та позбавить необхідності укладання безлічі юридичних угод з правовласниками аудіокомпозицій.

Застосунок, до якого користувачі звертаються за потреби отримати аудіосупровід, може також бути використаним для просування маловідомих авторів музичних творів. На відомому сервісі розповсюдження музики SoundCloud автори доволі часто розміщують свої треки з поміткою “CREATIVE COMMONS” у назві. Дана помітка означає, що люди можуть вільно використовувати трек у своєму контенті. Автори використовують такий прийом задля популяризації своєї творчості. Тому застосунок для генерації мелодії доцільно розширити функціоналом, що надає можливість завантажити композицію з треків маловідомих авторів або ж покласти один з таких треків в основу генерації мелодії. Таким чином, ресурс задовільнить як потребу авторів контенту в аудіосупроводі, так і потребу авторів музики в популяризації своєї діяльності.

1.2 Огляд наявних аналогів

1.2.1 Jukebox від OpenAI

Модель автокодувальника Jukebox стискає аудіо до дискретного простору, використовуючи підхід на основі квантування під назвою VQ-VAE [5]. Ієрархічні VQ-VAE можуть генерувати короткі інструментальні п'єси з кількох наборів інструментів, однак вони страждають від колапсу ієрархії через використання послідовних кодерів у поєднанні з авторегресійними декодерами. Спрощений варіант під назвою VQ-VAE-2 дозволяє уникнути цих проблем.

У VQ-VAE використано три рівні, які стискають необроблений аудіо 44 кГц у 8х, 32х і 128х відповідно, з розміром кодової книги 2048 для кожного рівня. Наступним етапом є навчання моделей, метою яких є вивчення розподілу музичних кодів, закодованих VQ-VAE, і створення музики в цьому стисненому дискретному просторі. Подібно до VQ-VAE, моделі мають три рівні пріоритетів: пріоритет верхнього рівня, який генерує найбільш стислі коди, і два пріоритети підвищення

дискретизації, які генерують менш стиснуті коди. Для тренування моделі було підготовлено новий набір даних із 1,2 мільйона пісень (600 000 із яких англійською мовою) у поєднанні з відповідними текстами пісень і метаданими з LyricWiki. Трансформатор верхнього рівня навчається завданням передбачати стислі звукові маркери. Щоб зіставити аудіофрагменти з відповідним текстом використана евристика, яка вирівнює символи тексту, щоб лінійно охопити тривалість кожної пісні, і передати вікно фіксованого розміру символів, зосереджених навколо поточного сегмента під час навчання.

Перевагою розробки є можливість виділення звукових маркерів та можливість задати в моделі додаткову інформацію, наприклад, про виконавця та жанр. Це дозволяє зменшити ентропію звукового прогнозу, тому модель може досягти кращої якості у будь-якому конкретному стилі. Окрім цього, під час генерації можливо керувати моделлю для генерації у стилі за вибором користувача.

Недоліком, який виділяють OpenAI, є погана робота з розподілення тексту відносно музичного супроводу для швидких жанрів, таких як Нір-Нор.

Слід зазначити, що дане рішення доступне лише в форматі коду, тому є інструментом для розробників, проте не готовим програмним застосунком для кінцевого користувача.

1.2.2 Magenta від Tensorflow

Рішення Magenta представляє архітектуру Perceiver AR [6]. Модель Perceiver відображає вхідні дані в латентний простір фіксованого розміру, де відбувається вся подальша обробка. Це дає змогу масштабувати введення понад 100 тисяч токенів. Perceiver AR будується на початковій архітектурі Perceiver, додаючи причинне маскування.

Perceiver AR спочатку відображає вхідні дані (рис. 1.2) у латентний масив фіксованого розміру за допомогою однієї cross-attention операції. Потім ці латенти взаємодіють у глибокому стеку self-attention шарів, щоб отримати оцінки для кожної цілі. Останні вхідні дані відповідають запитам, а кожен латент відповідає цільовій

позиції. У музичному домені використано до 65 тис. токенів, що відповідає декільком хвилинам тривалості у символічному домені та одній хвилині в необробленому аудіо домені.

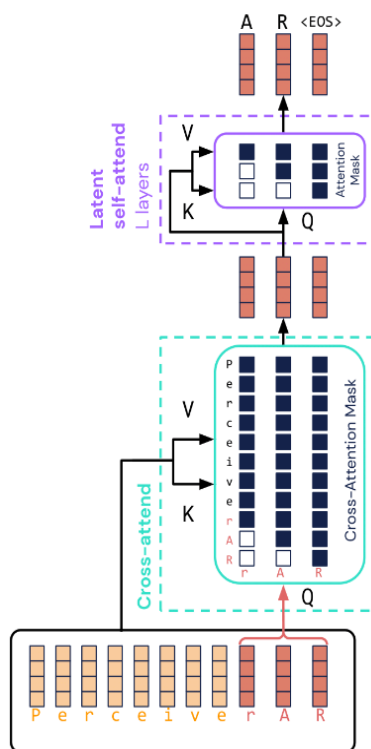


Рисунок 1.1 – Модель Perceiver AR

Перевагою рішення серед інших моделей-трансформаторів є те, що воно вирішує проблему масштабування вхідних даних. На класичні авторегресійні трансформатори негативно впливає некоректний розмір вхідних файлів, оскільки він унеможлиблює моделювання тривалих фрагментів.

Як і Jukebox, рішення Magenta представлено у вигляді програмного коду. На офіційному веб-ресурсі можна переглянути демонстраційні варіанти можливостей практичного застосування розробки, проте вони не є готовими програмними продуктами.

1.2.3 Soundraw

На відміну від Jukebox та Magenta, Soundraw є готовим програмним рішенням, що дозволяє генерувати музичні композиції та надає ліцензії на їхнє використання [7]. Веб-застосунок дозволяє обрати жанр, настрій тривалість, темп та обрати інструментал. Готовий фрагмент можна відредагувати, змінивши ритм, тривалість та структуру.

Основним недоліком розробки є відсутність масштабованості. Даний веб-застосунок підтримується тільки браузером Google Chrome.

1.3 Огляд пов'язаних досліджень

Проблема автоматичної генерації музики активно досліджується впродовж останніх десятиліть. Існує низка наукових праць, у яких автори пропонують варіації граматики для представлення музики, проектують різноманітні моделі нейронних мереж та застосовують еволюційне моделювання.

1.3.1 A Comprehensive Survey on Deep Music generation: Multi-level Representations, Algorithms, Evaluations, and Future Directions

У роботі «A Comprehensive Survey on Deep Music generation: Multi-level Representations, Algorithms, Evaluations, and Future Directions» (Shulei Ji, Jing Luo, and Xinyu Yang; School of Computer Science and Technology, Xi'an Jiaotong University, China, 2020) автори розглядають історію проблематики автоматизованого написання музики та різні способи представлення даних [8]. Пропонується розглядати процес генерації мелодії за наступними рівнями:

- верхній рівень: генерація партитури;
- середній рівень: відтворення музики;
- нижній рівень: генерація аудіо.

Задачі верхнього рівня включають в себе виділення акордів вхідного твору, музичну гармонізацію та накладання додаткового умовного акомпонементу на мелодію. На середньому рівні задача алгоритмів та моделей полягає у застосуванні параметрів відтворення до виділених на верхньому рівні частин мелодії. Нижній рівень моделей вирішує найважчі завдання, такі як синтез аудіо та вокальної партії. Автори дослідження наводять приклади моделей для вирішення задач кожного з рівнів. Переважно розглядаються генеративні нейронні мережі.

1.3.2 Personalized Popular Music Generation Using Imitation and Structure

Праця «Personalized Popular Music Generation Using Imitation and Structure» (Shuqi Dai, Carnegie Mellon University; Xichu Ma, National University of Singapore; Ye Wang, National University of Singapore; Roger B. Dannenberg, Carnegie Mellon University, 2021) спрямована на вирішення задачі автоматичного створення персоналізованої музики, імітуючи одну або кілька пісень, вибраних користувачем [9]. Описана система приймає одну початкову пісню як вхід. Після деяких кроків попередньої обробки кожна виділена з вхідного матеріалу частина передається у відповідний модуль генерації. Наявні наступні модулі генерації:

- структурний;
- виділення прогресії акордів (партетури);
- модуль мелодії;
- накладання басу.

Результати з структурного модуля та модуля виділення прогресії акордів подаються в інші модулі як вхідні дані. Темп відповідає темпу початкової пісні. Мелодія формується відповідно до розрахованих частот даних структури та прогресії акордів. Для накладання стилізованого басу використовується метод ланцюгів Маркова. Виходом є поєднання стилістичної мелодії, акордового супроводу і басових MIDI-треків, синтезоване для отримання аудіо.

1.3.3 Symbolic Music Generation with Transformer-GANs

Дослідження “Symbolic Music Generation with Transformer-GANs” (Aashiq Muhamed; Liang Li; Xingjian Shi; Suri Yaddanapudi; Wayne Chi; Dylan Jackson; Rahul Suresh; Zachary C. Lipton; Alexander J. Smola; Amazon Web Services, Carnegie Mellon University, 2021) спрямоване на покращення підходу використання генеративно-змагальних мереж-трансформерів (Transformer-GAN) для створення довготривалих музичних сутностей [10]. Основні результати роботи включають в себе:

- новий підхід Transformer-GAN для створення довгих музичних послідовностей із понад 1000 маркерів із використанням попередньо навченої мережі SpanBERT у якості дискримінатора;
- детальне дослідження впливу попереднього навчання, функцій втрат, регуляризації та кількості заморожених шарів у дискримінаторі на якість музики;
- ряд прийомів для тренування генеративно-змагальної мережі;
- показник на основі класифікатора для оцінки генеративних моделей.

Проаналізовані публікації демонструють, що проблема розвитку алгоритмів генерації музики залишається актуальною.

1.4 Постановка задачі

1.4.1 Призначення розробки

Інформаційна система для написання музичних творів призначена для генерації музичних композицій на основі референсних фрагментів та користувацьких параметрів.

1.4.2 Цілі та задачі розробки

Метою створення системи є надання користувачам можливості автоматизованого створення унікальних мелодій для задоволення їхніх комерційних та особистих потреб.

Для досягнення мети необхідно реалізувати наступні задачі:

1. Розробити алгоритми для генерації музичних творів на основі вхідного фрагменту.
2. Розробити алгоритм для накладання користувацьких параметрів на згенерований фрагмент.
3. Дослідити ефективність алгоритмів.
4. Розробити користувацький інтерфейс.

1.5 Опис процесу діяльності

Розроблюваний веб-застосунок є програмним продуктом, призначеним для генерації музичних фрагментів на основі референсних композицій користувача або вбудованої вибірки з урахуванням його побажань відносно інструменталу та нотного інтервалу вихідної композиції.

Діаграму варіантів використання наведено в додатку А. Передбачені у системі варіанти використання наведено у таблицях 1.1 - 1.3.

Таблиця 1.1 – Варіант використання UC01

Назва	Завантаження референсної композиції
Опис	Користувач завантажує мелодію, на яку повинна бути подібною згенерована композиція
Учасники	Користувач
Передумова	Користувач знаходиться на головній сторінці веб-застосунку

Продовження таблиці 1.1

Постумова	Базовий сценарій: 1. Аудіофайл успішно збережено в системі для подальшої обробки. 2. Система скерувала користувача на сторінку “Вибір інструменталу композиції”. Альтернативний сценарій: 1. Система повернула помилку “Некоректний формат аудіофайлу. Будь ласка, завантаж файл у форматі .mid/.midi”.
Сценарій	1. Користувач натискає кнопку “Завантажити композицію” на сторінці “Головна”. 2. Користувач вибирає з файлової системи ПК файл у форматі .mid/.midi. 3. Користувач натискає кнопку “Завантажити”.

Таблиця 1.2 – Варіант використання UC02

Назва	Вибір референсної композиції зі списку
Опис	Користувач обирає з наявного списку мелодію, на яку повинна бути подібною згенерована композиція
Учасники	Користувач
Передумова	Користувач знаходиться на головній сторінці веб-застосунку
Постумова	1. Інформацію про обраний аудіофайл успішно збережено в системі для подальшої обробки. 2. Система скерувала користувача на сторінку “Вибір інструменталу композиції”.

Продовження таблиці 1.2

Сценарій	1. Користувач натискає кнопку “Обрати зі списку” на сторінці “Головна”. 2. Система скеровує користувача на сторінку “Список композицій”. 3. Користувач прослуховує мелодії на сторінці “Список композицій”. 4. Користувач натискає кнопку “Обрати”, розташовану біля елемента плеєра композиції, яку він бажає використати у якості референсу.
----------	---

Таблиця 1.3 – Варіант використання UC03

Назва	Вибір інструменталу вихідної мелодії
Опис	Користувач задає інструменти, що звучатимуть в мелодії, яку він має отримати на виході
Учасники	Користувач
Передумова	Користувач знаходиться на сторінці “Вибір інструменталу композиції”.
Постумова	1. Система зберегла задані параметри інструменталу. Для генерації за референсним аудіофайлом: 2. Система скерувала користувача на сторінку “Результат генерації”. Для генерації за референсною вибіркою: 2. Система скерувала користувача на сторінку “Вибір нотного інтервалу”.

Продовження таблиці 1.3

Сценарій	1. Користувач заповнює чек-бокс для вибору інструменталу композиції 2. Користувач натискає кнопку “Обрати”
----------	---

Таблиця 1.4 – Варіант використання UC04

Назва	Перегляд результатів генерації, проведеної на основі референсної композиції
Опис	Користувач ознайомлюється з результатами генерації
Учасники	Користувач
Передумова	Користувач знаходиться на сторінці “Результат генерації”.
Постумова	-
Сценарій	Базовий сценарій: 1. Користувач ознайомлюється зі згенерованою мелодією, використовуючи кнопки елемпету плеєра. 2. Користувач ознайомлюється з гістограмами частот входження акордів вхідної композиції (або композицій вибірки) та результуючого аудіофрагменту. Додатковий сценарій: 3. Користувач завантажує згенеровану мелодію у форматі .wav

Таблиця 1.5 – Варіант використання UC05

Назва	Вибір референсної вибірки зі списку
Опис	Користувач обирає з наявного списку мелодію, на яку повинна бути подібною згенерована композиція
Учасники	Користувач
Передумова	Користувач знаходиться на будь-якій сторінці веб-застосунку
Постумова	1. Інформацію про обрану вибірку успішно збережено в системі для подальшої обробки. 2. Система скерувала користувача на сторінку “Вибір інструменталу композиції”.
Сценарій	1. Користувач натискає кнопку “Вибірки” в навігаційному меню. 2. Система скеровує користувача на сторінку “Список вибірок”. 3. Користувач обирає вибірку зі списку.

Таблиця 1.6 – Варіант використання UC06

Назва	Задання нотного інтервалу
Опис	Користувач задає нотні інтервали для обраних інструментів
Учасники	Користувач
Передумова	Користувач знаходиться на сторінці “Вибір нотного інтервалу”.

Продовження таблиці 1.6

Постумова	1. Задані інтервали успішно збережено в системі. 2. Система скерувала користувача на сторінку “Результат генерації”.
Сценарій	1. Користувач задає нотні інтервали для відповідних інструментів 3. Користувач натискає кнопку “Задати”

Таблиця 1.7 – Варіант використання UC07

Назва	Завантаження мелодії зі списку композицій
Опис	Користувач завантажує мелодію зі списку наявних композицій.
Учасники	Користувач
Передумова	Користувач знаходиться на сторінці “Список композицій”.
Постумова	Композицію успішно завантажено на ПК користувача.
Сценарій	1. Користувач прослуховує мелодії на сторінці “Список композицій”. 2. Користувач завантажує аудіофайл у форматі .wav .

1.6 Інформаційне забезпечення

Користувач взаємодіє з системою через графічний інтерфейс та завантаження аудіофайлів у форматі .mid/.midi. Вибір формату аудіо обумовлений тим, що при збереженні аудіо даних в MIDI, інформація про параметри гучності, конкретні ноти, тональність, ритм, звукові та шумові ефекти, тощо, кодується в цифровій формі і

зберігається у вигляді команд. Таким чином, даний формат є зручним для виділення інформації комп'ютерними засобами та подальшої обробки обробки.

Таким чином, на вхід системи користувач подає:

- аудіофайл у форматі .mid/.midi (для випадку генерації за завантаженим референсом);
- мітку обраного аудіофайлу (для випадку генерації за референсом зі списку);
- мітку обраної вибірки (для випадку генерації на основі вбудованої вибірки);
- список музичних інструментів, що повинні бути використаними при генерації мелодії;
- числове значення нотних інтервалів (для випадку генерації на основі вбудованої вибірки).

На виході система повертає користувачеві:

- згенерований аудіофайл у форматі .wav;
- порівняльний аналіз вхідної та вихідної композиції у вигляді графіків;
- зображення нот вихідної композиції.

Висновки до розділу

У даному розділі було проаналізовано предметну область та виділено можливі сфери застосування розроблюваного застосунку. Проведено огляд наукових досліджень пов'язаних з вирішенням задачі автоматизованої генерації музичних композицій. Наведено наявні аналоги програмних рішень. Окреслено постановку задачі. Виділено основні сценарії взаємодії користувача з розроблюваною системою та визначено формат вхідних і вихідних даних.

2 МАТЕМАТИЧНЕ ЗАБЕЗПЕЧЕННЯ

2.1 Огляд підходів до вирішення задачі генерації музики

2.1.1 Стохастичні процеси

Для вирішення задачі створення мелодій комп'ютерними засобами часто використовуються стохастичні процеси. Найбільшого поширення набуло використання ланцюгів Маркова. Ланцюг Маркова - це модель, яка описує послідовність можливих подій. Ця послідовність повинна задовольняти припущення Маркова - ймовірність наступного стану залежить від попереднього стану, а не від усіх попередніх станів у послідовності.

Розглянемо послідовність випадкових величин $X_1, X_2, \dots, X_n$. Такий набір випадкових величин $\{X_i\}$ утворює стохастичний процес. Індекс i в X_i часто називають часом. Таким чином, X_i - стан ланцюга в фіксований час i . Ланцюг Маркова - це випадковий процес, такий, що для всіх i виконується умова $[X_i \vee X_1, \dots, X_{i-1}] \sim [X_i \vee X_{i-1}]$ [11].

Музичний твір можна розглянути як послідовність випадкових величин - акордів. За допомогою аналізу входження акордів до вхідної композиції можна побудувати розподіл ймовірностей для акордів, які слідує за певним акордом. Після цього, можна виконувати побудову мелодії обираючи випадковим чином наступний акорд з урахуванням ймовірності його слідування за поточним акордом. Дані дії можна повторювати встановлену кількість разів. Отриманий ланцюг акордів є музичною композицією.

2.1.2 Граматики

Формальна граматика - це кінцевий набір правил, які описують, як генерувати правильні речення формальною мовою. Якщо речення не може бути створене за допомогою граматичних правил, воно не є реченням з описаної формальної мови [12].

Граматика формально визначається як кортеж (V_N, V_T, P, S) , де:

- V_N – це скінченний, непорожній набір символів, які називаються нетермінальними змінними, кожна з яких потім може бути замінена іншим нетермінальним або термінальним символом.
- V_T – скінченна множина термінальних символів, які є незмінними; неможливо переписати термінальний символ, це припиняє будь-яке похідне. Кожен термінальний символ є фактичним словом із зазначеної мови.
- P – набір правил продукування. Кожне правило продукування містить нетермінальний символ, який описується правилом.
- S – початковий символ; це особливий символ у V_N , який іноді називають символом речення.

Починаючи з S , після кінцевої кількості кроків згідно з правилами продукування, можна отримати рядок, який містить лише символи терміналів. Рядок називається мовою, створеною граматикою, якщо він містить лише термінальні символи, які походять від початкового символу S .

Граматика можна використовувати для створення музичних творів. За основу формальної мови граматика береться певний набір музичних знань про гармонійне звучання, правила переходу, темп тощо. Для кращого результату вхідний набір аргументів повинен бути підготованим людиною, оскільки людина може сформулювати правила продукування беручи в якості символів не лише окремі ноти/акорди, але і їхні комбінації. Проте, можна виділяти набори правил для певних стилів композицій шляхом застосування комп'ютерних засобів для аналізу музики. У такому випадку граматика представляють у вигляді матриці суміжності, де 1 позначає те, що перехід від одного акорду в інший можливий, 0 - неможливий.

2.1.3 Оптимізаційні методи

2.1.3.1 Локальний пошук

Оптимізаційні методи застосовуються для генерації мелодії певного стилю. На вхід подається певна довільна композиція, стиль якої необхідно змінити. Ця

модифікація стилю розглядається як проблема оптимізації, коли музику потрібно оптимізувати щодо різних цілей.

Перед реалізацією оптимізації необхідно створити масив даних прикладів музики цільового стилю, а також контрприкладів. На основі виділених ознак масивів будується функція оцінки.

Просте монофонічне символічне музичне представлення можна розглядати як серію подій I ноти N , де кожна подія ноти n_i є кортежем висоти p_i як номера ноти MIDI та тривалості d_i у чвертях [13]:

$$N = (n_1, n_2, \dots, n_I), n_i = (p_i, d_i). \quad (2.1)$$

У випадку паузи, попередньо визначений символ паузи займає місце номера ноти. Разом із нотними подіями необхідна анотація акорду, яка також представлена як серія J акордових подій C , де кожна акордова подія c_j є кортежем акордового символу s_j і тривалості d_j :

$$C = (c_1, c_2, \dots, c_J), c_i = (s_i, d_i). \quad (2.2)$$

Таку існуючу N з відповідним C слід модифікувати таким чином, щоб вона наблизилась до певного стилю. Це завдання розглядається як локальний пошук, що виконує багатоцільову оптимізацію для пошуку локально-оптимального N . Основна ідея полягає в тому, щоб почати з вхідного фрагменту музичного твору і спробувати оцінити його сусідів. Сусіди утворюються випадковим виконанням однієї з таких змін:

- зміна висоти p_i окремої події ноти таким чином, що максимальний інтервал зміни становить велику третину вгору або вниз;
- зміна тривалості двох нот d_{i1} і d_{i2} так, щоб загальна тривалість послідовності нот залишалася незмінною;
- нота n_i ділиться на кілька, початкова тривалість d_i ділиться у співвідношеннях $(1/2, 1/2)$, $(2/3, 1/3)$ або $(1/3, 1/3, 1/3)$;
- Дві послідовні ноти n_i та n_{i+1} об'єднуються в одну ноту тривалістю $d_i + d_{i+1}$.

Якщо сусід краще вхідного фрагменту, сусід зберігається, і процес ітеративно продовжується починаючи з обраного сусіда. Для оцінки сусідів можна застосовувати

критерій оптимізації Парето: стан вважається кращим лише тоді, коли всі цілі залишаються незмінними або набувають кращого значення. Процес завершується, коли знайдено оптимального сусіда - даний сусід є мелодією певного стилю.

2.1.3.2 Генетичні алгоритми

Генетичні алгоритми (ГА) підходять для створення музики, оскільки їхні оператори дозволяють безперервно вдосконалювати згенеровану музику, як це відбувається під час реального написання творів композиторами [14]. Проте, генетичні алгоритми необхідно комбінувати з іншими підходами, оскільки для оцінки популяції потрібні або набори правил, які дозволяють визначити відповідність мелодії бажаному результату, або нейронна мережа, яка оцінює подібність мелодії на задану композицію/вибірку.

Генетичний алгоритм для даної задачі працює наступним чином:

1. Спочатку визначається конфігурація операторів і параметрів ГА (швидкість мутацій, швидкість кросинговеру, кількість поколінь, розмір популяції, тип селекції та тип кросинговеру).
2. ГА починається з випадкового генерування хромосом початкової популяції. Кожна хромосома складається з 40 випадкових генів (нот або акордів).
3. Ранжування популяції. Обчислюється значення придатності для кожної хромосоми в популяції відповідно до відсотку відповідності заданим правилам або ж оцінці нейронної мережі (залежить від обраного допоміжного методу). Хромосоми впорядковуються відповідно до значення придатності. Зберігається певний заданий відсоток хромосом.
4. Створення наступної популяції:
 - a. Відбір батьків відповідно до значення придатності - оператор відбору повинен бути таким, щоб більше значення придатності надавало хромосомі більшу ймовірність бути відбраною. Можна чередувати декілька різних операторів відбору.
 - b. Застосування кросоверу для генерації нащадка.

- с. Мутація: обрати хромосому (ноту) оператором мутації та замінити її іншою хромосомою (ноту). Можна застосовувати для декількох хромосом.
- d. Додавання нащадку до популяції.

2.1.4 Штучний інтелект та машинне навчання

Застосування штучного інтелекту та машинного навчання є найбільш популярним підходом до генерації музичних композицій. Для цього використовуються моделі-трансформери, рекурентні нейронні мережі (Recurrent neural network, RNN), нейронні мережі довгої короткочасної пам'яті (Long short-term memory, LSTM), згорткові нейронні мережі (Convolutional neural network, CNN), генеративно-змагальні мережі (Generative adversarial networks, GAN).

2.1.4.1 Моделі-трансформери

Нехай $V = \{v_i, i \in \mathfrak{I}\}$ - словник, індексований набором $\mathfrak{I}$, і послідовність $s = (i_t \in \mathfrak{I}, t \in [0, N])$ довжини N , де t - індекс у послідовності. Вихід трансформатора є категоріальним розподілом ймовірностей за $\mathfrak{I}$, залежним від s , з якого можна вивести N -й елемент $S(N)$ [15].

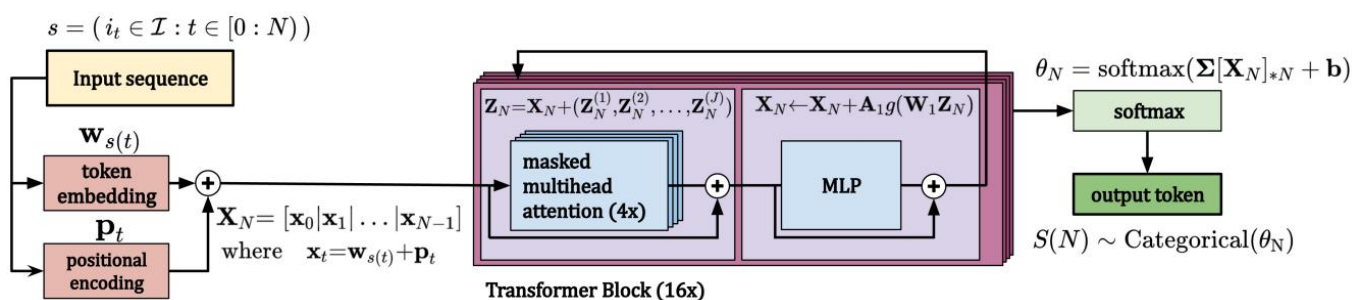


Рисунок 2.1 – Архітектура моделі-трансформера, що обробляє послідовність s довжини N для виведення наступного елемента послідовності $S(N)$

Кожен елемент s представлений сумою вектора d_e -вимірному дійсного простору та вектора, що кодує його позицію в послідовності. Тобто, елемент t з s кодується як $w_{s(t)} \in \mathbb{R}^{d_e}$, а його позиція кодується як $p_t \in \mathbb{R}^{d_e}$. Нехай $X_N = [x_0, x_1, \dots, x_{N-1}]$, де $x_t = w_{s(t)} + p_t$. Даний вектор обробляється першим шаром з кількістю вузлів уваги (attention) рівній J , кожна операція є підпростором $\mathbb{R}^{d_e}$ розмірності $d_h = d_e/J$. Вузол уваги j продукує запити типу ключ-значення: $Q_N^{(j)} = W_j^Q X_N$, $K_N^{(j)} = W_j^K X_N$ та $V_N^{(j)} = W_j^V X_N$, де $W_j^Q, W_j^K, W_j^V \in \mathbb{R}^{d_h \times d_e}$. Після цього j -ий вузол уваги формує масштабовану внутрішню функцію $S_N^{(j)} = (d_h)^{-1/2} [Q_N^{(j)}]^T K_N^{(j)}$ та приводить кожен стовпець $S_N^{(j)}$ до форми $\hat{S}_N^{(j)} \in \mathbb{R}^{N \times N}$, після чого формує вихід $Z_N^{(j)} = V_N^{(j)} \hat{S}_N^{(j)} \in \mathbb{R}^{d_h \times N}$. Виходи всіх вузлів уваги J об'єднуються $Z_N = X_N + (Z_N^{(1)}, Z_N^{(2)}, \dots, Z_N^{(J)}) \in \mathbb{R}^{d_e \times N}$ та проходять нелінійне перетворення $Z'_N = g(W_1 Z_N)$, де $W_1 \in \mathbb{R}^{4d_e \times d_e}$. Кінцевим виходом першого шару є $X_N \leftarrow X_N + A_1 Z'_N$, де $A_1 \in \mathbb{R}^{d_e \times 4d_e}$. Наступний шар уваги виконує ті самі операції над оновленим значенням X_N зі зміненими параметрами (на параметри накладаються ваги при переході з одного шару в інший). Після L шарів уваги фінальний шар приймає на вхід останній стовпець X_N та обчислює параметри шляхом розподілу $\theta_N = \text{softmax}(\Sigma[X_n]_{*N} + b)$ над $\mathfrak{Z}$, де $\Sigma \in \mathbb{R}^{|\mathfrak{Z}| \times d_e}$ і $b \in \mathbb{R}^{|\mathfrak{Z}|}$. Стандартною функцією втрат для моделі-трансформера є крос-ентропія.

Для генерації музики спочатку виконується обробка вхідної вибірки. Музичні твори аналізуються, після чого формується словник з токенів, які репрезентують метр, висоту, тональність і т.д. Експериментальним методом встановлюється, які параметри найбільше впливають на вихідний результат, після чого застосовується описаний вище підхід.

2.1.4.2 Рекурентні нейронні мережі (RNN)

RNN належать до сімейства нейронних мереж зі зворотним зв'язком. Вони відрізняються від інших мереж сімейства здатністю зберігати інформацію про кроки

в часі [16]. RNN розглядають послідовну інформацію як вхідну та надають інформацію в послідовній формі замість того, щоб приймати стабільний вхід і забезпечувати стабільний вихід.

Вихідні дані, надані шарами всередині мереж, входять в шари повторно у якості вхідних даних для обчислення значення шару. Завдяки цьому процесу мережа навчається на основі поточних і попередніх даних разом. Базова RNN складається з багатьох копій нейронної мережі, з'єднаних і працюючих у ланцюжку. Таким чином, RNN може працювати з послідовними вхідними та вихідними даними.

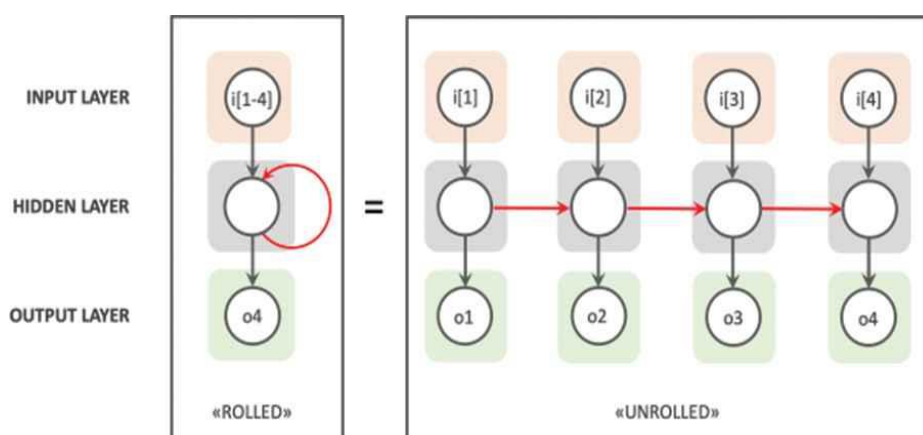


Рисунок 2.2 – Архітектура базової моделі RNN

Аудіодані також є послідовними даними, які можна розглядати як сигнал, що має модуляцію з часом, подібно до даних часового ряду, де точки даних збираються в послідовності з часовими значеннями. Комп'ютер зберігає аудіодані як дані про зміни числових значень із часом. Моделі RNN добре відомі своєю ефективністю в області послідовного моделювання, тому широко застосовуються для вирішені задач генерації та зміни стилю музики.

2.1.4.3 Нейронні мережі довгої короткочасної пам'яті (LSTM)

LSTM – це рекурентні нейронні мережі, що містять в повторюваному прихованому шарі блоки пам'яті – спеціальні одиниці, що представляють осередки пам'яті з самоз'єднаннями, які зберігають тимчасовий стан мережі на додаток до

спеціальних мультиплікативних одиниць, які називаються воротами для керування потоком інформації [17]. Кожен блок пам'яті в оригінальній архітектурі містив вхідний (input gate) і вихідний вентиль (output gate). Вхідний вентиль керує потоком активацій введення в комірку пам'яті. Вихідний шлюз керує вихідним потоком активації осередку в решту мережі. Пізніше до блоку пам'яті було додано шлюз забуття (forget gate). Шлюз забуття масштабує внутрішній стан комірки перед тим, як додати його як вхід до комірки через самоповторююче з'єднання осередку, таким чином адаптивно забуваючи або скидаючи пам'ять комірки. Мережа LSTM обчислює відображення вхідної послідовності $x=(x_1, \dots, x_T)$ до вихідної послідовності $y=(y_1, \dots, y_T)$ шляхом розрахунку активації мережевого блоку з використанням наступних рівнянь ітераційно з $t=1$ до T :

$$i_t = \sigma(W_{ix}x_t + W_{iz}m_{t-1} + W_{ic}c_{t-1} + b_i), \quad (2.3)$$

$$f_t = \sigma(W_{fx}x_t + W_{fz}m_{t-1} + W_{fc}c_{t-1} + b_f), \quad (2.4)$$

$$c_t = f_t \odot c_{t-1} + i_t \odot g(W_{cx}x_t + W_{cz}m_{t-1} + b_c), \quad (2.5)$$

$$o_t = \sigma(W_{ox}x_t + W_{oz}m_{t-1} + W_{oc}c_{t-1} + b_o), \quad (2.6)$$

$$m_t = o_t \odot h(c_t), \quad (2.7)$$

$$y_t = \phi(W_{ym}m_t + b_y), \quad (2.8)$$

де W позначають вагові матриці (наприклад W_{ix} - це матриця ваг від вхідного отвору до входу); W_{ic} , W_{fc} , W_{oc} - це діагональні вагові матриці для з'єднань «вічка», b позначають вектори зміщення (b_i - вхідний вентиль вектор зміщення), σ — це логістична сигмоїдна функція; i , f , o та c — відповідно вхідні ворота, ворота забуття, вихідні ворота та вектори активації осередку, усі з яких мають той самий розмір, що й вектор активації вихідної клітини m ; m — поелементний добуток векторів; g і h — функції активації входу та виходу клітинки; ϕ — функція активації виходу мережі.

LSTM є вдосконаленням RNN мереж, тому також використовуються для генерації музики та показують кращі результати.

2.1.4.4 Згорткові нейронні мережі (CNN)

На відміну від LSTM, CNN, що використовує фільтри спільної ваги, не підходить для виявлення та обробки часових особливостей музичних творів [18]. Різні області музичної послідовності мають різні локальні статистичні характеристики. Таким чином, припущення просторової стаціонарності згортки не може виконуватися. Цей недолік вирішується застосування згорткової нейронної мережі з локальним з'єднанням. Оскільки кожен часовий крок вхідної послідовності повинен мати різні характеристики, різні частини певної послідовності не повинні мати одне ядро.

CNN є ядром авторегресійної моделі генерації, а локальна згортка є ядром CNN. Рівень локальної згортки працює подібно до простого шару згортки, за винятком того, що ваги не використовуються спільно. Тобто інший набір фільтрів застосовується до кожного окремого фрагмента входу. Загалом, для завдання генерації послідовності простий згортковий рівень розширює своє ядро за допомогою вимірювання каналу, тоді як рівень локального перетворення розширює своє ядро за допомогою вимірювання каналу та кроку часу.

Нехай вхід і вихід рівня CNN – це F_{in} з N часовим кроком і F_{out} з M часовим кроком відповідно:

$$F_{in} = \{x_1, x_2, \dots, x_N\}, F_{out} = \{y_1, y_2, \dots, y_M\}. \quad (2.9)$$

Позначимо примітивну згортку як f з розміром ядра CNN K , заповненням P та кроком S , так що $i \in [1, N]$, $j \in [1, M - K]$, $j = (i - K + 2P) / S + 1$. Примітивне представлення CNN можна описати так:

$$y_i = f(x_j, x_{j+1}, \dots, x_{j+K-1}). \quad (2.10)$$

Для операції локального перетворення використовується локальна згорткова функція f_i з розміром ядра M :

$$y_i = f_i(x_j, x_{j+1}, \dots, x_{j+M-1}). \quad (2.11)$$

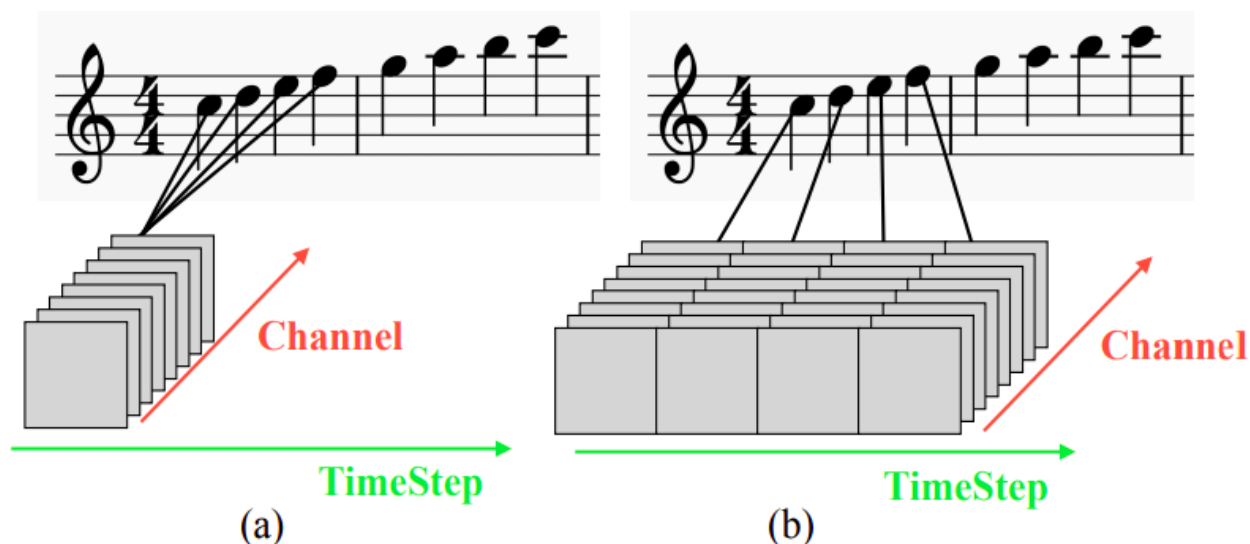


Рисунок 2.3 – Згорткова операція над музичними даними. У (b) модель локальної згортки додає більше ядер через часовий вимір, що працює набагато краще, ніж традиційна згорткова операція (a)

2.1.4.5 Генеративно-змагальні нейронні мережі (GAN)

Генеративно-змагальна мережа (GAN) — це модель машинного навчання, у якій дві нейронні мережі конкурують одна з одною, щоб стати точнішими у своїх прогнозах. Дві нейронні мережі, які утворюють GAN, називаються генератором і дискримінатором [19].

Дискримінатор є класифікатором. Він розрізняє реальні дані та дані, згенеровані генератором. Він може використовувати будь-яку мережеву архітектуру, придатну для типу даних, які має генерувати GAN. Дискримінатор тренується на двох типах даних. Перший тип - це реальні вхідні дані. Дискримінатор розглядає їх як базові мітки істинності того, як виглядає «справжній» артефакт. Другий тип - дані, створені генератором. Вони використовуються як базові мітки істинності того, як виглядає «фальшивий» артефакт.

Процес навчання дискримінатора відбувається наступним чином:

1. Дискримінатор класифікує справжні та підроблені дані, отримані від генератора.

2. Дискримінатор обчислює функцію втрат, яка штрафує його за неправильну класифікацію реального артефакту як підробленого або підробленого артефакту як справжнього.
3. Дискримінатор оновлює свої вагові коефіцієнти шляхом зворотного поширення похибки, щоб мінімізувати втрати дискримінатора, не знаючи про функцію втрат генератора.

Генератор вчиться враховувати зворотний зв'язок від дискримінатора для генерування підроблених даних. Його мета полягає в тому, щоб змусити дискримінатор класифікувати свій підроблений результат як справжній.

У своїй базовій формі GAN приймає випадковий шум як вхідний сигнал (стандартним вибором є нормальний розподіл). Потім генератор перетворює цей шум на артефакти у форматі, який задає користувач.

Вводячи шум, GAN може генерувати багато варіацій даних шляхом вибірки в різних місцях цільового розподілу.

Навчання генератора працює наступним чином:

1. Генератор приймає свій початковий вхід, як правило, випадковий шум, і перетворює його на значущий вихід.
2. Дискримінатор забезпечує класифікацію виходу, класифікуючи його як справжній або підроблений.
3. Дискримінатор обчислює свою функцію втрат.
4. Зворотне розповсюдження виконується через мережі дискримінатора та генератора для отримання градієнтів.
5. Градієнти використовуються лише для налаштування вагових коефіцієнтів генератора, щоб згенерований вміст з більшою ймовірністю “обдурив” дискримінатор наступного разу.

Високорівнева архітектура генеративно-змагальної нейронної мережі представлена на рисунку 2.4.

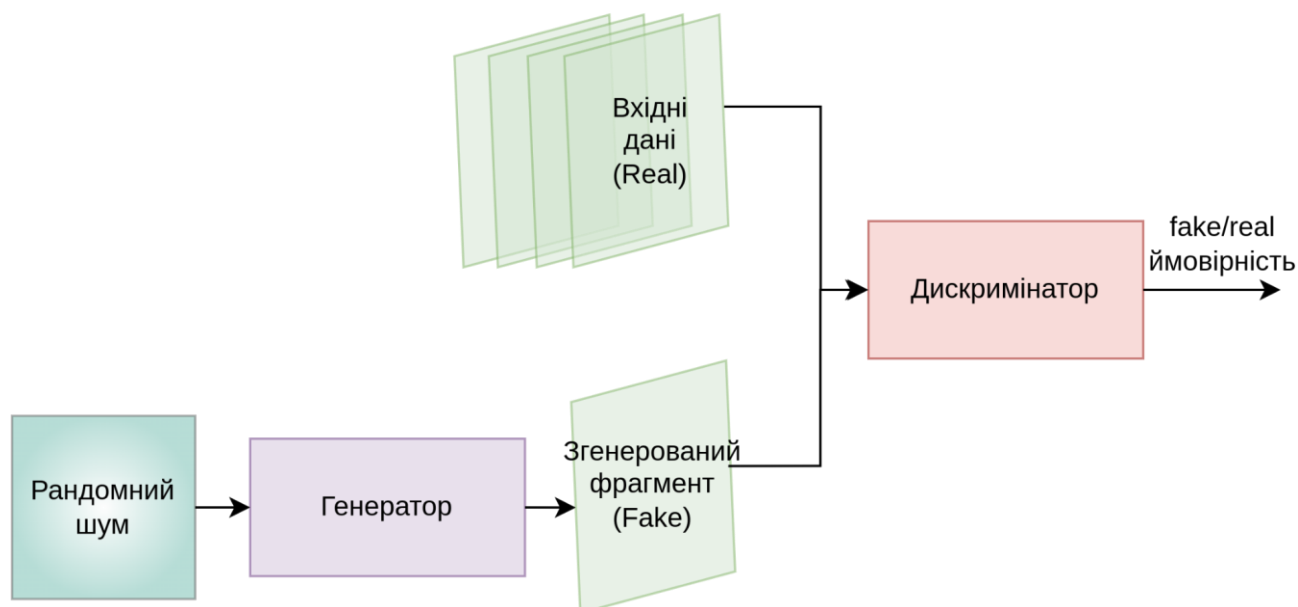


Рисунок 2.4 – Архітектура генеративно-змагальної нейронної мережі

2.2 Обґрунтування вибору використаних методів та їхній опис

У розроблюваному застосунку присутні генерація за референсним твором та генерація за референсною вибіркою.

Для задачі генерації мелодії за референсним твором обрано підхід побудови Марківського ланцюга. Вибір обумовлений тим, що підхід з використанням граматик та оптимізаційні методи краще працюють з людським втручанням для створення правил та відбору критеріїв. Для навчання нейронної мережі може не вистачити розміру вхідної вибірки (токенів вхідного твору), особливо якщо завантажений/обраний користувачем твір є монотонним та короткотривалим. У якості експерименту було спробовано генерацію твору нейронною мережею LSTM на основі композиції Моцарта “Соната №17 в B-Dur” тривалістю 12 хвилин. Припускається, що тривалість фрагментів, завантажених користувачем, найчастіше становитиме 3-5 хвилин - середню довжину музичного супроводу пісень, що є найбільш популярними музичними творами. Через це очікується, що результат генерації нейронної мережі за умови її використання в застосунку буде гіршим, ніж

отриманий в експерименті. Нейронна мережа складалась з двох шарів LSTM зі 128 виходами, шару зміщення, двох шарів зміщення методом виключення (dropout) між ними та шару активації softmax. У додатку Б представлено спектограми вхідної мелодії та результату генерації нейронної мережі.

Спектрограми демонструють залежність спектральної густини сигналу від часу. Функція спектральної густини описує потужність сигналу залежно від частоти. З побудованих спектрограм легко бачити, що нейронна мережа виділила лише найпотужніші сигнали (скоріше за все, нейрони інших токенів було обнулено як незначимі). Натомість ланцюги Маркова розглядають всі можливі переходи зі стану в стан. Навіть беручи до уваги те, що вибір наступного акорду в алгоритмі враховує ймовірності переходів, вихідна мелодія включатиме в себе більше властивостей за рахунок випадкового вибору наступного фрагменту.

Для генерації за вхідною вибіркою обрано метод генеративно-змагальних нейронних мереж. Вибірки включають у себе декілька творів одного композитора, тому для генерації вихідного твору у стилі вибірки цілком доречно застосувати можливість нейронних мереж виділяти основні фактори серед їхнього великого набору, виокремленого з різних творів. Генеративно-змагальна архітектура нейронної мережі дозволяє не лише згенерувати твір з найбільш важливих факторів композицій вибірки, але й оцінювати його відповідність вибірці завдяки дискримінанту. Таким чином, якщо дискримінатор має велику точність, це дозволяє гарантувати, що згенерований фрагмент створено у стилі заданої вибірки.

2.2.1 Побудова ланцюга Маркова

Перший етап включає в себе обробку користувацького файлу та створення датафрейму з вхідними даними. Спочатку виділяються ноти та акорди вхідного файлу. Після цього, ітеративним проходом списком нот та акордів формується список їхніх висот та тривалостей (тривалості вимірюються четвертями). Отримавши такий список, на ноти та акорди можна накласти мітки, що відповідають їхній тривалості: короткі - тривалість до однієї четверті; середні - тривалість 2-3 четверті; довгі - 4

Розглянемо приклад для переходу з акорду (стану) C5-L виділеному на рисунку 2.8.

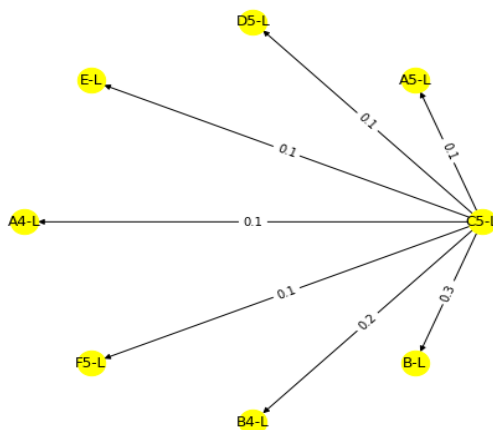


Рисунок 2.6 – Переходи зі стану C5-L

З прикладу можна зауважити, що сума ймовірностей переходів дорівнює одиниці. Ймовірності переходів у інші наявні стани, такі як G5-L, D6-L, E5-M, C#-L, B5-L, F4-L, E4-L, D4-L, C4-L, B3-L, F6-L, E5-L, C6-L, G4-L, C5-L, D5-L дорівнюють нулю та не представлені на графі. З ймовірністю 0.3 після акорду C5-L йтиме акорд B-L.

Акорд для початку мелодії вибирається випадковим чином. Для вибору наступного елементу використовується дискретна ймовірнісна функція $p_x(x) = P(X=x)$, де $X = [x_1, x_2, \dots, x_n] \subset R^n$. Таким чином, здійснюється випадковий вибір наступного елементу з урахуванням значень ймовірностей слідування елементів. Процес триває задану кількість кроків, у нашому випадку, 250.

2.2.2 Топологія генеративно-змагальної нейронної мережі

Розглянемо топологію нейронної мережі, використаної для даної розробки.

Генератором виступає згорткова нейронна мережа послідовного типу, що складається з наступних шарів:

- повнозв'язний (Dense) шар з вхідним розміром 1000 та вихідним розміром 256;
- нещільний зрізаний лінійний вузол (Leaky ReLu) з коефіцієнтом нахилу 0.2;
- шар пакетної нормалізації (BatchNormalization) з моментом ковзаючого середнього рівним 0.8;
- повнозв'язний (Dense) шар з вихідним розміром 512;
- нещільний зрізаний лінійний вузол (Leaky ReLu) з коефіцієнтом нахилу 0.2;
- шар пакетної нормалізації (BatchNormalization) з моментом ковзаючого середнього рівним 0.8;
- повнозв'язний (Dense) шар з вихідним розміром 1024;
- нещільний зрізаний лінійний вузол (Leaky ReLu) з коефіцієнтом нахилу 0.2;
- шар пакетної нормалізації (BatchNormalization) з моментом ковзаючого середнього рівним 0.8;
- повнозв'язний шар з сигмоїдальною функцією активації;
- шар перетворення форми виходу (Reshape).

Дискримінатор - нейронна мережа послідовного типу, що складається з наступних шарів:

- LSTM з 512 виходами без повернення вихідних даних;
- шар регуляризації методом виключення (dropout) з ймовірністю відкидання нейронів рівній 50%;
- повнозв'язний (Dense) шар з вихідним розміром 512;
- нещільний зрізаний лінійний вузол (Leaky ReLu) з коефіцієнтом нахилу 0.2;
- шар регуляризації методом виключення (dropout) з ймовірністю відкидання нейронів рівній 50%;
- повнозв'язний шар з сигмоїдальною функцією активації та кількістю виходів еквівалентній 1.

Зображення архітектури застосованої нейронної мережі представлено в додатку

В.

Сигмоїдальна функція активації генерує значення за наступною формулою:

$$f(x) = \frac{1}{1 + e^{-x}}. \quad (2.12)$$

Вибір функції активації пов'язаний з областю її значень - від 0 до 1. Основною задачею дискримінатора є визначення ймовірностей того, є артефакт реальними вхідними даними чи згенерованими. Оскільки ймовірність будь-чого існує лише в діапазоні від 0 до 1, сигмоїдальна функція є зручною для даної задачі.

Нейрони повнозв'язного шару Dense отримують вихідні дані від кожного нейрона свого попереднього шару та виконують множення матриці значень нейронів на вектор ваг.

Нещільний зрізаний вузол Leaky ReLu реалізує наступну функцію:

$$f(x) \begin{cases} x, & \text{якщо } x > 0 \\ 0.01x & \end{cases}. \quad (2.13)$$

Шар пакетної реалізації BatchNormalization реалізує масштабування та нормалізацію для зменшення перенавчання.

Шар LSTM реалізує архітектуру нейронної мережі LSTM описану у роботі "Long short-term memory" Зепа Хохрайтера [20]. Мережа складається з одного вхідного, одного прихованого і одного вихідного шарів. Повнозв'язний самоз'єднаний прихований шар містить комірки пам'яті та відповідні блоки вентилів. Прихований рівень також може містити "звичайні" приховані блоки, що забезпечують входи для блоків вентилів і комірок пам'яті. Усі блоки (крім блоків вентилів) на всіх шарах мають спрямовані з'єднання (служать вхідними даними) для всіх одиниць наступного шару. S-комірок пам'яті, що мають один і той самий вхідний і той самий вихідний вентиль, утворюють структуру, яка називається блоком комірок пам'яті розміру S. Блоки комірок пам'яті полегшують зберігання інформації. Блок комірки пам'яті розміром 1 є простою коміркою пам'яті.

У якості функції активації LSTM використовується функція гіперболічного тангенсу, оскільки її похідна якої може підтримуватися протягом тривалого діапазону, перш ніж перейти до нуля, що є необхідним для подолання проблеми

зникаючого градієнта. У якості функції активації рекурентного шару LSTM використовується сигмоїдальна функція.

Шар регуляризації Dropout виключає нейрони з ймовірністю $rate$. Таким чином, ймовірність того, що нейрон залишиться в мережі, становить $1 - rate$. "Виключення" нейрона означає, що при будь-яких вхідних даних або параметрах він повертає 0.

Шар Reshape використовується для перетворення результату на певну форму. У нашому випадку вихідною формою генератора є форма вхідного масиву інформації.

У якості функції втрат при компіляції дискримінатора та комбінації генератора і дискримінатора використовується функція втрат бінарної кросентропії:

$$Loss = \frac{1}{n} \sum_{i=1}^n y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i), \quad (2.14)$$

де n - розмір вихідних даних; $\hat{y}_i$ - i -те скалярне значення на виході моделі; y_i - i -те очікуване значення.

Параметри мережі необхідно змінювати задля мінімізації функції втрат. Для цього застосовуються оптимізатори. При компіляції розроблюваної мережі застосовується оптимізатор Adam. Оптимізація Adam - це метод стохастичного градієнтного спуску, який базується на адаптивній оцінці моментів першого та другого порядку [21]. Нехай α - розмір кроку алгоритму; $\beta_1, \beta_2 \in [0, 1)$ - експоненційні швидкості розпаду для поточних оцінок; $f(\theta)$ - стохастична цільова функція з параметрами θ ; m, v - вектори першого та другого моментів, t - часовий крок. Алгоритм Adam:

1. Ініціалізувати вектор першого моменту: $m_0 \leftarrow 0$.
2. Ініціалізувати вектор другого моменту: $v_0 \leftarrow 0$.
3. Ініціалізувати часовий крок: $t \leftarrow 0$.
4. Ініціалізувати вектор параметрів початковими значеннями θ_t .
5. Поки θ_t не збігається:

- a. Збільшити крок: $t \leftarrow t + 1$.
 - b. Обчислити градієнт стохастичної цільової функції на кроці t :

$$g_t \leftarrow \nabla_{\theta} f(\theta_{t-1}).$$
 - c. Оновити оцінку першого моменту: $m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$
 - d. Оновити оцінку другого моменту: $v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$
 - e. Обчислити оцінку першого моменту з виправленням зсуву:

$$\hat{m}_t \leftarrow m_t / (1 - \beta_1^t).$$
 - f. Обчислити оцінку другого моменту з виправленням зсуву:

$$\hat{v}_t \leftarrow v_t / (1 - \beta_2^t).$$
 - g. Оновити параметри: $\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \hat{m}_t / (\sqrt{\hat{v}_t} + \epsilon)$
6. Повернути список параметрів.

Даний метод оптимізації обчислювально ефективний, має невеликі вимоги до пам'яті, інваріантний до діагонального масштабування градієнтів і добре підходить для проблем, які є великими з точки зору даних/параметрів.

2.2.3 Огляд результатів роботи використаних методів

2.2.3.1 Ланцюг Маркова

Для перевірки роботи методу ланцюгів Маркова, застосуємо його до твору Клода Дебюссі “Arabesque No.1”. Побудуємо графіки частоти входження частин та тривалостей акордів вхідної та згенерованої композицій.

На рисунку 2.7 представлено графік частоти входження частин певної висоти до вхідної композиції.

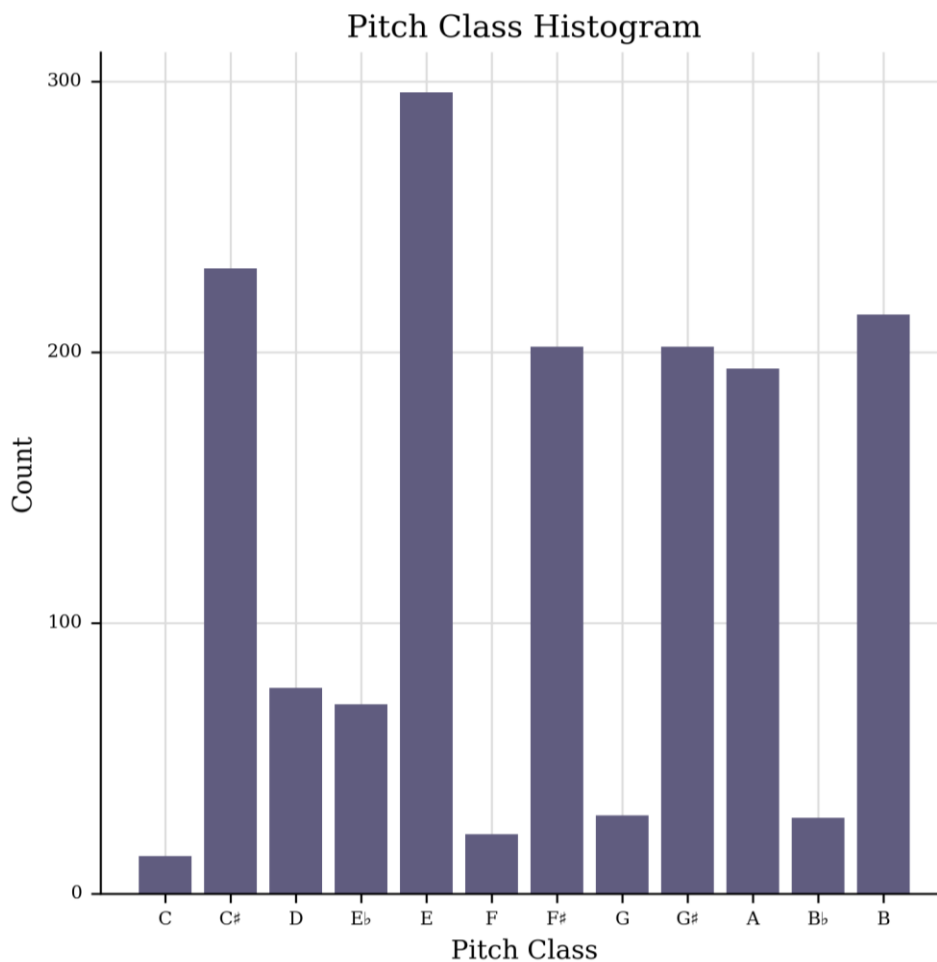


Рисунок 2.7 – Частота входження частин певної висоти у вхідній композиції

На рисунку 2.8 представлено графік частоти входження частин певної висоти у згенерованій композиції.

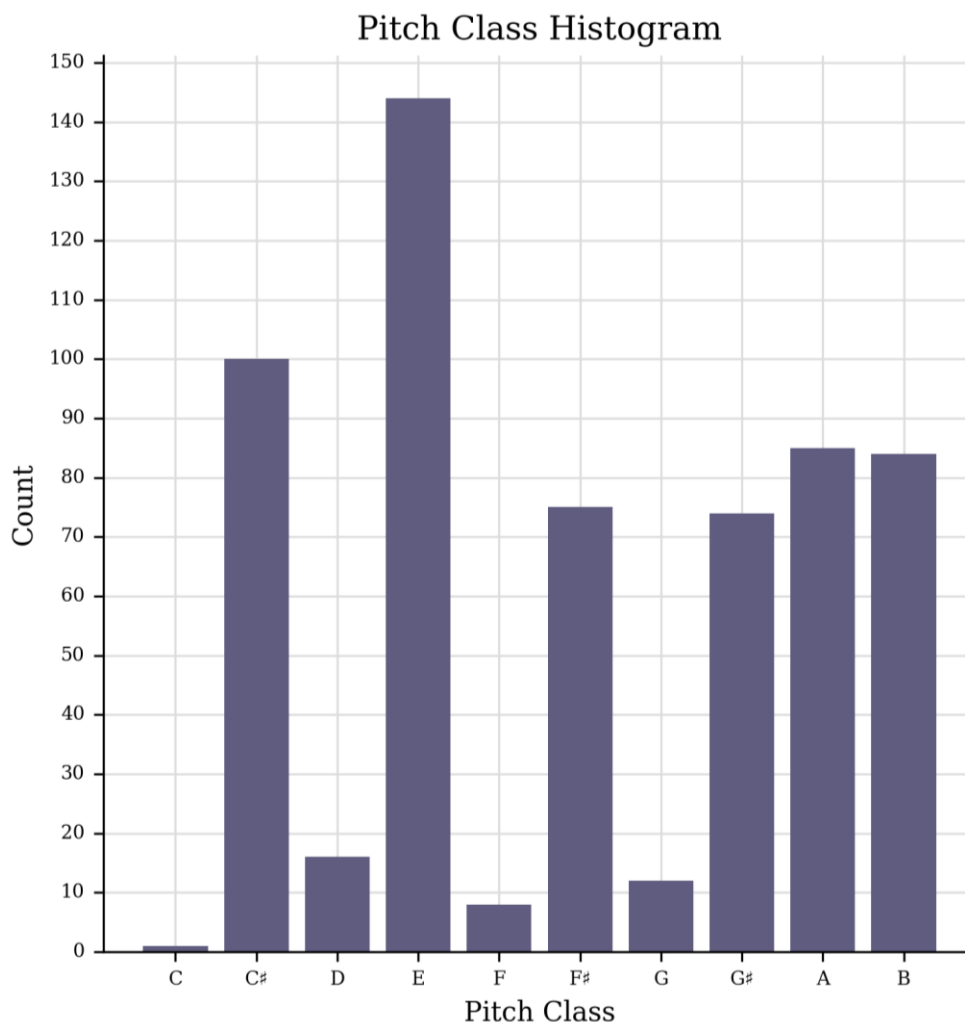


Рисунок 2.8 – Частота входження частин певної висоти у згенерованій композиції композиції

З рисунків 2.7 та 2.8 можна бачити, що найчастіше як у вхідній, так і у згенерованій композиціях зустрічаються частини висоти C#, E, F#, G#, A, B. З цього можна зробити висновок, що ланцюг було правильно побудовано за критерієм врахування ймовірностей.

У додатку Г представлено тривалості частин в четвертях. З графіків можна бачити, що у згенерованій композиції тривалості більш однотипні, на відміну від вхідної композиції. Це пов'язано з присвоєнням частинам розмірів S, M, L.

2.2.3.2 Генеративно змагальна нейронна мережа

Проаналізуємо функції втрат нейронної мережі на вибірках з композиціями Баха (29 композицій), Бартока (12 композицій), Бетховена (13 композицій), Дебюссі (7 композицій), Гендель (8 композицій).

Графіки функцій втрат для вибірок представлено у додатку Д. Функція втрат дискримінатора для усіх вибірок впродовж переважної кількості епох коливається у межах 0.6-0.8, що є адекватним результатом для стабільної генеративно-змагальної мережі (оскільки дискримінатор тренується на реальних та фейкових даних, після чого складена модель оновлює параметри генератора). Значення функції втрат дискримінатора говорить про те, що генератор добре виконує задачу «підробки» даних, тобто, генерує мелодії подібні на вхідні.

Розмір навчальної вибірки для композицій Баха становить 13019. Точність моделі після 100 епох навчання склала 65.62% (рис. 2.9).

```
99 / 100 [D loss: 0.674278, acc.: 67.58%]  
4/4 [=====] - 1s  
100 / 100 [D loss: 0.679603, acc.: 65.62%]
```

Рисунок 2.9 – Точність моделі, навченої на вибірці Баха, після 100 епох навчання

Розмір навчальної вибірки для композицій Бартока становить 10075. Точність моделі після 100 епох навчання склала 71.48% (рис. 2.10).

```
98 / 100 [D loss: 0.459684, acc.: 87.89%]  
4/4 [=====] - 1s  
99 / 100 [D loss: 0.796573, acc.: 62.50%]  
4/4 [=====] - 1s  
100 / 100 [D loss: 0.532898, acc.: 71.48%]  
The G-DNN-GAN model has been trained with
```

Рисунок 2.10 – Точність моделі, навченої на вибірці Бартока після 100 епох навчання

Розмір навчальної вибірки для композицій Бетховена становить 105596. Точність моделі після 100 епох навчання склала 51.56% (рис. 2.11).

```

97 / 100 [D loss: 0.642691, acc.: 72.66%]
4/4 [=====] - 1s
98 / 100 [D loss: 0.869106, acc.: 69.92%]
4/4 [=====] - 1s
99 / 100 [D loss: 0.682491, acc.: 57.03%]
4/4 [=====] - 1s
100 / 100 [D loss: 0.682601, acc.: 51.56%]

```

Рисунок 2.11 – Точність моделі, навченої на вибірці Бетховена після 100 епох навчання

Розмір навчальної вибірки для композицій Дебюссі становить 17039. Точність моделі після 100 епох навчання склала 66.8% (рис. 2.12).

```

99 / 100 [D loss: 0.644801, acc.: 64.06%]
4/4 [=====] - 1s
100 / 100 [D loss: 0.589385, acc.: 66.80%]

```

Рисунок 2.12 – Точність моделі, навченої на вибірці Дебюссі після 100 епох навчання

Розмір навчальної вибірки для композицій Генделя становить 30032. Точність моделі після 100 епох навчання склала 95.31% (рис. 2.13).

```

98 / 100 [D loss: 0.367228, acc.: 80.08%]
4/4 [=====] - 1s
99 / 100 [D loss: 0.175835, acc.: 95.70%]
4/4 [=====] - 1s
100 / 100 [D loss: 0.145753, acc.: 95.31%]

```

Рисунок 2.13 – Точність моделі, навченої на вибірці Генделя після 100 епох навчання

Для усіх вибірок, окрім композицій Бетховена, точність нейронної мережі перевищує 60%. Низька точність для даної вибірки пов'язана з великим розміром навчальної вибірки, яка включала у себе фрагменти музичних творів з різними ознаками. Це призвело до того, що значення втрат дискримінатора (а відповідно і генератора) були стабільними. Найкраща точність досягнута для вибірки з

композиціями Генделя. Це пов'язано з тим, що у вибірку входили подібні між собою твори. Нейронна мережа дає кращі результати, коли твори не лише написані одним автором, але й мають спільні характеристики.

Висновки до розділу

У даному розділі виконано огляд наявних методів для вирішення задачі генерації музики. Обґрунтовано вибір методів для реалізації розроблюваного застосунку та наведено їхній опис. Проведено огляд отриманих результатів роботи методів.

3 ПРОГРАМНЕ ТА ТЕХНІЧНЕ ЗАБЕЗПЕЧЕННЯ

3.1 Опис засобів розробки

Для реалізації програмного застосунку використано мову програмування Python версії 3.8. Дана мова є зручною для реалізації задач розробки, адже містить широкий спектр бібліотек для машинного навчання, обробки музики та математичних розрахунків.

Для вибору інструментів розробки веб-застосунку було розглянуто такі фреймворки Python як Flask та Django. Django - це передбачає високорівневий фреймворк для створення веб-застосунків, що передбачає застосування патерну проектування MVC (Model-View-Controller) [22]. Фреймворк Flask належить до класу мікрофреймворків та обмежується базовим інструментарієм для реалізації веб-застосунків. Даний фреймворк використовує шаблонізатор Jinja2 та набір інструментів Werkzeug і дозволяє реалізовувати будь-яку архітектуру [23]. Розроблюваний стосунок є достатньо простим з точки зору веб-частини, тому для його реалізації обрано мікрофреймворк Flask.

У якості середовищ розробки використано інтегроване середовище розробки PyCharm та марний сервіс для обчислень Google Colab. Середовище PyCharm є зручним для роботи з мовою Python та надає інструментарій для якісного та швидкого аналізу коду [24]. Вибір хмарного середовища Google Colab пов'язаний з наявною у ньому можливістю використання GPU для пришвидшення процесу навчання нейронних мереж. Після тренування нейронних мереж в середовищі Google Colab моделі зберігались у файли для подальшої інтеграції з застосунком.

Перелік використаних засобів розробки наведено у таблиці 3.1

Таблиця 3.1 – Перелік засобів розробки

Бібліотека/фреймворк	Опис
Flask	Мікрофреймворк для розробки веб-застосунків мовою Python.
TensorFlow, Keras	Keras - бібліотека програмного забезпечення з відкритим кодом, яка забезпечує інтерфейс Python для розробки нейронних мереж. Keras виступає інтерфейсом для бібліотеки TensorFlow. Застосовується в рамках роботи для побудови моделі нейронної мережі, що вирішує задачу генерації композиції на основі творів композитора.
Music21	Бібліотека з набором інструментів для автоматизованого аналізу музики. Застосовується в рамках розробки парсингу вхідної композиції, конвертації категоріальної інформації в числову та навпаки, задання інструментів композиції, конвертації даних в музичний файл та побудови графіків частоти акордів в композиціях.
Dlib	Багатоплатформова бібліотека, інструментів для вирішення задач машинного навчання.
NumPy	Бібліотека з відкритим кодом, що містить функціонал для математичної обробки масивів.

Продовження таблиці 3.1

Бібліотека/фреймворк	Опис
Os	Модуль для роботи та управління процесами файлової системи та середовища.
Midi2Audio	Бібліотека для перетворювання .midi файлів у файли інших типів.
Mido	Бібліотека для роботи з MIDI-повідомленнями та портами.
collections	Реалізує спеціалізовані контейнерні типи даних, що є альтернативою вбудованим контейнерам загального призначення Python.
random	Бібліотека, що реалізує генератори псевдовипадкових чисел для різних розподілів.
Bootstrap	Фреймворк CSS із відкритим вихідним кодом, спрямований на адаптивну інтерфейсну веб-розробку. Він містить HTML, CSS і шаблони дизайну на основі JavaScript для типографіки, форм, кнопок, навігації та інших компонентів інтерфейсу.

3.2 Архітектура програмного забезпечення

3.2.1 Опис класів

Для розробки застосунку застосовано парадигму ООП, базовими елементами якої є класи. Розроблюваний програмний продукт складається з чотирьох класів:

- App: містить методи для отримання та обробки даних користувача, повернення оброблених даних, підтримки логіки веб-застосунку;
- Instruments: містить методи для додавання потоків музичних інструментів до згенерованого фрагменту, накладання нотних інтервалів;
- MarkovChain: містить методи для попередньої обробки вхідного файлу, реалізації алгоритму побудови ланцюга Маркова та створення вихідного аудіо. Діаграму класів представлено в додатку Е.

3.2.2 Сценарій роботи

Діаграму послідовності наведено в додатку Ж. Опис кроків послідовності представлено в таблиці 3.2.

Таблиця 3.2 – Опис кроків послідовності

№	Назва кроку	Опис	Запит
1.1	Натискання кнопки [Завантажити]	Користувач обрав мелодію на ПК та починає процес натисканням кнопки [Завантажити] на сторінці “Головна”.	-

Продовження таблиці 3.2

№	Назва кроку	Опис	Запит
1.2	Запит на збереження файлу	Браузер надсилає контролеру запит на збереження файлу.	POST /music/<audiofile={file}>
1.3	Перевірити формат файлу	Контролер перевіряє файл на відповідність умові: ALLOWED_EXTENSIONS = {'mid', 'midi'}. Якщо умову виконано - перехід в крок 1.4. У протилежному випадку - перехід в крок 1.04.	-
1.04	Викликати повідомлення про помилку	Контролер надсилає запит на відтворення повідомлення з помилкою.	GET /index/error
1.05	Відтворення повідомлення про помилку	На веб-сторінці “Головна” відтворюється вікно з повідомленням про помилку.	-
1.4	Зберегти файл	Контролер виконує збереження аудіофайлу.	-
1.5	Запит на сторінку "Вибір інструменталу композиції"	Контролер надсилає запит на відкриття сторінки "Вибір інструменталу композиції".	GET /instruments

Продовження таблиці 3.2

№	Назва кроку	Опис	Запит
1.6	Відтворення форми для вибору інструменталу	Браузер відтворює веб-сторінку “Вибір інструменталу композиції”.	-
1.7	Заповнення форми для вибору інструменталу	Користувач заповнює форму та натискає кнопку [Обрати] на веб-сторінці “Вибір інструменталу композиції”.	-
1.8	Запит на збереження результатів	Браузер надсилає контролеру запит з результатами заповнення форми користувачем.	POST /instruments/<instrument={formInput}>
1.9	Тимчасове збереження результату	Контролер виконує збереження користувацького вибору.	-
1.10	Згенерувати аудіофайл	Контролер виконує генерацію аудіофайлу методом ланцюгів Маркова.	-
1.11	Згенерувати графіки	Контролер виконує генерацію статистичних показників вхідної та згенерованої композицій.	-
1.12	Запит на сторінку "Результат генерації"	Контролер надсилає запит на відкриття сторінки "Результат генерації".	GET /markovresult

Продовження таблиці 3.2

№	Назва кроку	Опис	Запит
1.13	Відтворення результату	Браузер відтворює сторінку "Результат генерації".	-
1.14	Натискання кнопки [Завантажити]	Користувач натискає кнопку [Завантажити] в контекстному меню аудіопрогравача на сторінці "Результат генерації".	-
1.15	Запит на завантаження	Браузер надсилає контролеру запит на завантаження композиції.	-
1.16	Повернути файл на ПК користувача	Контролер виконує операції для завантаження композиції на ПК користувача	Згенероване URL (метод GET)
2.1	Натискання кнопки [Обрати зі списку]	Користувач обрав мелодію на ПК та починає процес натисканням кнопки [Обрати зі списку] на сторінці "Головна".	-
2.1	Запит на відкриття списку композицій	Браузер надсилає контролеру запит на перенаправлення на сторінку "Список композицій".	-
2.3	Запит на сторінку "Список композицій"	Контролер надсилає запит на відкриття сторінки "Список композицій"	GET /playlist_instruments

Продовження таблиці 3.2

№	Назва кроку	Опис	Запит
2.4	Відтворення списку композицій	Браузер відтворює сторінку “Список композицій”.	-
2.5	Натискання кнопки [Обрати]	Користувач, після прослуховування композицій, натискає кнопку [Обрати] біля аудіопрогравача вподобаної композиції на сторінці “Список композицій”.	-
2.6	Запит на збереження вибору користувача	Браузер надсилає контролеру інформацію про вибір користувача.	POST /playlist_instruments/ <int:_id>
2.7	Зберегти вибір користувача	Контролер зберігає інформацію про обрану користувачем композицію.	-
2.8	Запит на сторінку "Вибір інструменталу композиції"	Контролер надсилає запит на відкриття сторінки "Вибір інструменталу композиції".	GET /instruments
2.9	Відтворення форми для вибору інструменталу	Браузер відтворює веб-сторінку “Вибір інструменталу композиції”.	-

Продовження таблиці 3.2

№	Назва кроку	Опис	Запит
2.10	Заповнення форми для вибору інструменталу	Користувач заповнює форму та натискає кнопку [Обрати] на веб-сторінці “Вибір інструменталу композиції”.	-
2.11	Запит на збереження результатів	Браузер надсилає контролеру запит з результатами заповнення форми користувачем.	POST /instruments/<instrument={formInput}>
2.12	Тимчасове збереження результату	Контролер виконує збереження користувацького вибору.	-
2.13	Згенерувати аудіофайл	Контролер виконує генерацію аудіофайлу методом ланцюгів Маркова.	-
2.14	Згенерувати графіки	Контролер виконує генерацію статистичних показників вхідної та згенерованої композицій.	-
2.15	Запит на сторінку "Результат генерації"	Контролер надсилає запит на відкриття сторінки "Результат генерації".	GET /markovresult
2.16	Відтворення результату	Браузер відтворює сторінку "Результат генерації".	-

Продовження таблиці 3.2

№	Назва кроку	Опис	Запит
2.17	Натискання кнопки [Завантажити]	Користувач натискає кнопку [Завантажити] в контекстному меню аудіопрогравача на сторінці "Результат генерації".	-
2.18	Запит на завантаження	Браузер надсилає контролеру запит на завантаження композиції.	-
2.19	Повернути файл на ПК користувача	Контролер виконує операції для завантаження композиції на ПК користувача	Згенероване URL (метод GET)
3.1	Натискання кнопки [Вибірки]	Користувач обрав мелодію на ПК та починає процес натисканням кнопки [Вибірки] на будь-якій сторінці веб-застосунку.	-
3.2	Запит на відкриття списку вибірок	Браузер надсилає контролеру запит на перенаправлення на сторінку "Список вибірок".	-
3.3	Запит на сторінку "Список вибірок"	Контролер надсилає запит на відкриття сторінки "Список вибірок"	GET /samples
3.4	Відтворення списку вибірок	Браузер відтворює сторінку "Список вибірок".	-

Продовження таблиці 3.2

№	Назва кроку	Опис	Запит
3.5	Натискання кнопки [Обрати вибірку]	Користувач, після прослуховування композицій, натискає кнопку [Обрати вибірку] на сторінці “Список вибірок”.	-
3.6	Запит на збереження вибору користувача	Браузер надсилає контролеру інформацію про вибір користувача.	-
3.7	Зберегти вибір користувача	Контролер зберігає інформацію про обрану користувачем композицію.	-
3.8	Запит на сторінку "Вибір інструменталу композиції"	Контролер надсилає запит на відкриття сторінки "Вибір інструменталу композиції".	GET /{composer}/instruments
3.9	Відтворення форми для вибору інструменталу	Браузер відтворює веб-сторінку “Вибір інструменталу композиції”.	-
3.10	Заповнення форми для вибору інструменталу	Користувач заповнює форму та натискає кнопку [Обрати] на веб-сторінці “Вибір інструменталу композиції”.	-

Продовження таблиці 3.2

№	Назва кроку	Опис	Запит
3.11	Запит на збереження результатів	Браузер надсилає контролеру запит з результатами заповнення форми користувачем.	POST /{composer}/instruments/<instrument={formInput}>
3.12	Тимчасове збереження результату	Контролер виконує збереження користувацького вибору.	-
3.13	Запит на сторінку "Вибір нотного інтервалу"	Контролер надсилає запит на відкриття сторінки "Вибір нотного інтервалу".	GET /{composer}/intervals
3.14	Відтворення форми для задання нотного інтервалу	Браузер відтворює веб-сторінку "Вибір нотного інтервалу".	-
3.15	Заповнення форми для задання нотного інтервалу	Користувач заповнює форму та натискає кнопку [Задати] на веб-сторінці "Вибір інструменталу композиції".	-
3.16	Запит на збереження результатів	Браузер надсилає контролеру запит з результатами заповнення форми користувачем.	POST /{composer}/intervals /<interval={formInput}>
3.17	Тимчасове збереження результату	Контролер виконує збереження користувацького вибору.	-

Продовження таблиці 3.2

№	Назва кроку	Опис	Запит
3.18	Згенерувати аудіофайл	Контролер виконує генерацію аудіофайлу методом ланцюгів Маркова.	-
3.19	Згенерувати графіки	Контролер виконує генерацію статистичних показників вхідної та згенерованої композицій.	-
3.20	Запит на сторінку "Результат генерації"	Контролер надсилає запит на відкриття сторінки "Результат генерації".	GET /gan_result
3.21	Відтворення результату	Браузер відтворює сторінку "Результат генерації".	-
3.22	Натискання кнопки [Завантажити]	Користувач натискає кнопку [Завантажити] в контекстному меню аудіопрогравача на сторінці "Результат генерації".	-
3.23	Запит на завантаження	Браузер надсилає контролеру запит на завантаження композиції.	-
3.24	Повернути файл на ПК користувача	Контролер виконує операції для завантаження композиції на ПК користувача	Згенероване URL (метод GET)

3.2.3 Розгортання

Користувач взаємодіє з системою через робочу станцію (ПК) з відкритим веб-браузером. Веб-браузер з'єднано з веб-сервером, який включає в себе веб-інтерфейс, що взаємодіє з моделлю, яка звертається до вбудованої інформації.

Діаграму розгортання представлено в додатку И.

3.2.4 Специфікація методів

У таблиці 3.3 наведено опис методів розроблюваного застосунку.

Таблиця 3.3 – Опис методів

Метод	Опис
app.py	
main_page()	Повертає сторінку “Головна” веб-застосунку.
allowed_file(filename)	Перевіряє завантажений користувачем файл на відповідність формату .mid/.midi
upload_file()	Виконує завантаження файлу користувача в файлову систему та виклику сторінки “Вибір інструменталу композиції” у випадку методу POST та коректного формату файлу. У випадку методу POST та некоректного формату файлу повертає повідомлення про помилку. У випадку методу GET повертає сторінку “Головна” веб-застосунку.

Продовження таблиці 3.3

Метод	Опис
<code>render_instruments()</code>	У випадку методу GET повертає сторінку “Вибір інструменталу композиції”. У випадку методу POST виконує тимчасове збереження даних форми з інструментами та викликає методи для генерації мелодії методом ланцюгів Маркова і генерації зображень; повертає сторінку “Результат генерації”.
<code>render_from_playlist(_id)</code>	У випадку методу GET повертає сторінку “Вибір інструменталу композиції”. У випадку методу POST виконує тимчасове збереження даних форми з інструментами та викликає методи для генерації мелодії методом ланцюгів Маркова (з передачею id обраної користувачем мелодії) і генерації зображень; повертає сторінку “Результат генерації”.
<code>render_composer(composer)</code>	У випадку методу GET повертає сторінку “Вибір інструменталу композиції”. У випадку методу POST виконує тимчасове збереження даних форми з інструментами та повертає сторінку “Вибір нотного інтервалу”.

Продовження таблиці 3.3

Метод	Опис
<code>render_sample_result()</code>	У випадку методу GET повертає сторінку “Вибір нотного інтервалу”. У випадку методу POST виконує тимчасове збереження даних форми з інтервалами та викликає методи для генерації мелодії генеративно-змагальною нейронною мережею і генерації зображень; повертає сторінку “Результат генерації”.
<code>gan_result()</code>	Повертає сторінку “Результат генерації”. Застосовується лише для випадку генерації на основі вбудованої вибірки.
<code>instruments.py</code>	
<code>part_generate_model(self, model_filename, input_instrument, note_interval, audio_folder)</code>	Генерує аудіофайл, використовуючи натреновану нейронну мережу; додає до згенерованого фрагменту потоки обраних користувачем інструментів та відповідні нотні інтервали.
<code>markov_chain.py</code>	
<code>parse_midi_to_list(self, path)</code>	Виконує парсинг вхідного аудіофайлу та перетворює інформацію на список з акордами та тривалостями.
<code>create_markov_chain(self, list_input)</code>	Реалізує алгоритм побудови ланцюга Маркова.

Продовження таблиці 3.3

Метод	Опис
build_audio(self, data, num, input_instruments)	Створює мелодію заданої довжини (кількості акордів), додаючи потоки обраних користувачем музичних інструментів.
GAN.ipynb : Midi	
parser(self, folder)	Здійснює парсинг .midi файлів, розташованих в одному репозиторії, виділяє токени з творів та перетворює категоріальні дані в числові, повертаючи словник.
sequences(self)	Нормалізує дані в межах від -1 до 1 (необхідно для коректної обробки нейронною мережею).
build_audio(self, prediction, filename)	Перетворює вихід нейронної мережі на аудіофайл формату .midi.
GAN.ipynb : Model	
build_discriminator(self)	Будує модель дискримінатора.
build_generator(self)	Будує модель генератора.
train(self, epochs, folder, batch_size, sample_interval)	Метод для тренування нейронної мережі.
save(self)	Зберігає натреновану модель нейронної мережі.
generate(self)	Генерує передбачену послідовність токенів музичного твору.

3.3 Інструкція користувача

3.3.1 Опис екранних форм

Екранну форму сторінки “Головна” наведено на рисунку 3.1.

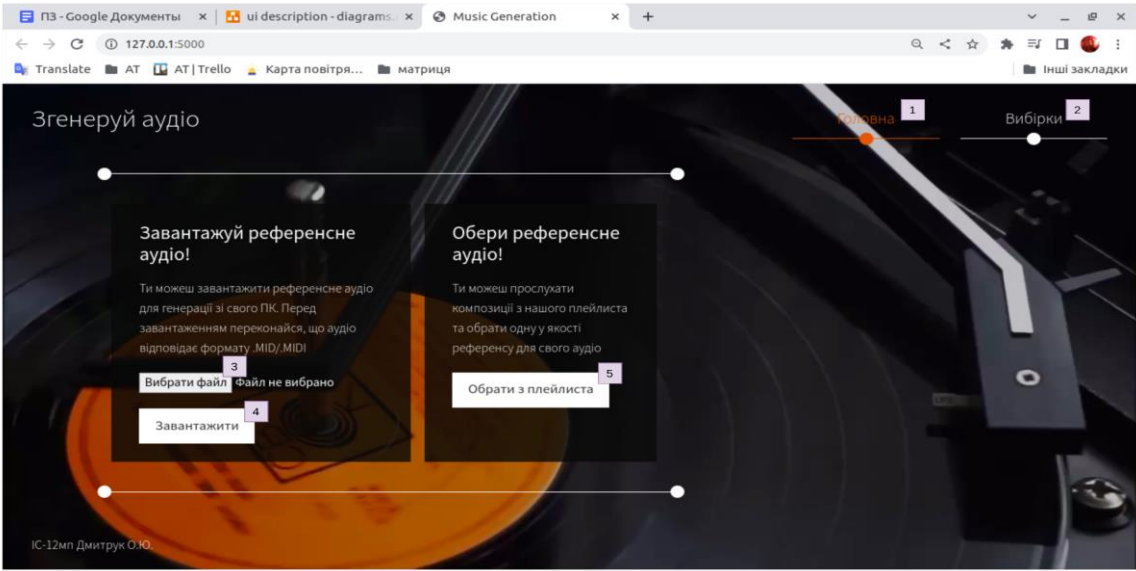


Рисунок 3.1 – Екранна форма сторінки “Головна”

Опис екранної форми сторінки “Головна” наведено у таблиці 3.4.

Таблиця 3.4 – Опис екранної форми сторінки “Головна”

№	Тип елемента та назва	Опис
1	Кнопка навігації [Головна]	Скеровує користувача на сторінку “Головна” веб-застосунку.
2	Кнопка навігації [Вибірки]	Скеровує користувача на сторінку “Список вибірок” веб-застосунку.
3	Кнопка [Вибрати файл]	Дозволяє обрати файл для завантаження з файлової системи користувача. Приклад вибору файлів наведено на рисунках 3.2 - 3.3.

Продовження таблиці 3.4

№	Тип елемента та назва	Опис
4	Кнопка [Завантажити]	Після натискання на кнопку користувачем, обраний файл завантажується в систему. У випадку вдалого завантаження система скерує користувача на сторінку “Вибір інструменталу композиції”.
5	Кнопка [Обрати з плейлиста]	Скерує користувача на сторінку “Список композицій” веб-застосунку.

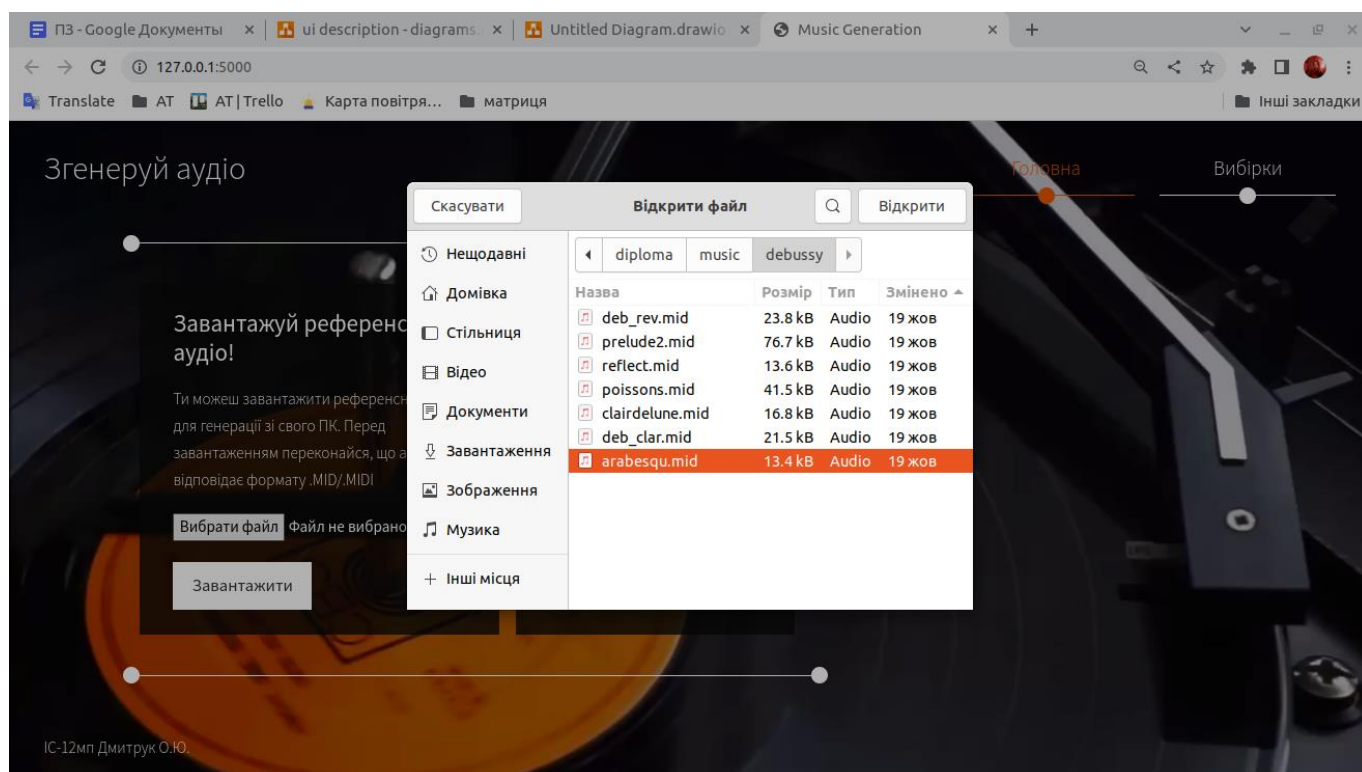


Рисунок 3.2 – Вибір файлу для завантаження

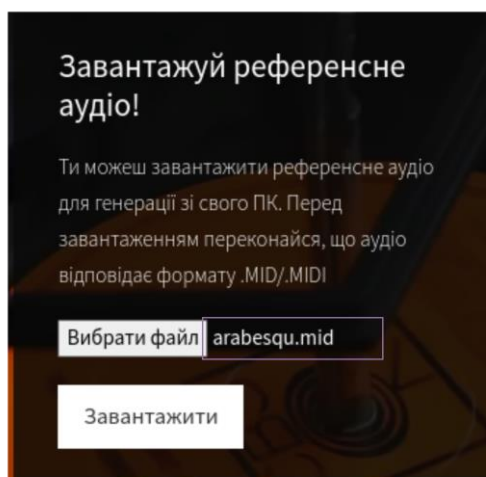


Рисунок 3.3 – Відтворення імені обраного файлу

Екранну форму сторінки “Список композицій” наведено на рисунку 3.4.

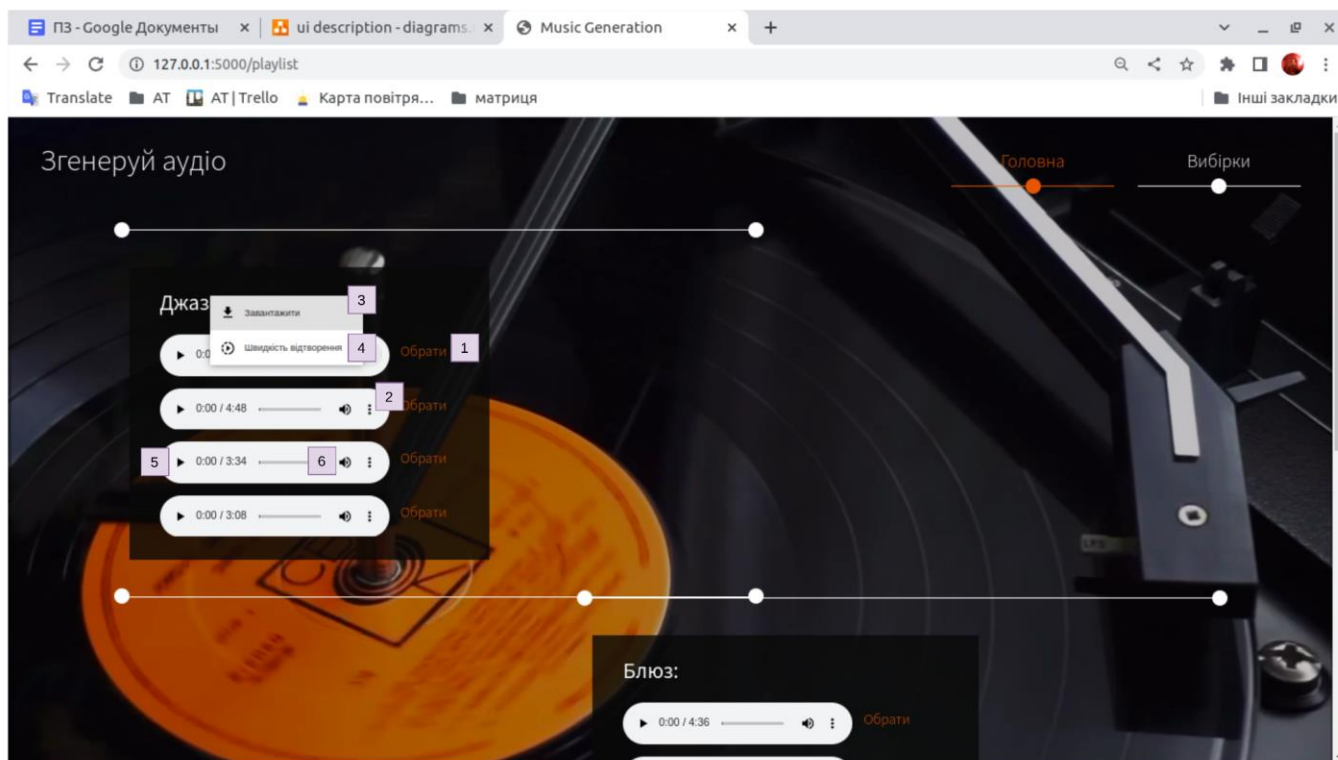


Рисунок 3.4 – Екранна форма сторінки “Список вбудованих композицій”

Опис екранної форми сторінки “Список композицій” наведено у таблиці 3.5.

Таблиця 3.5 – Опис екранної форми сторінки “Список композицій”

№	Тип елемента та назва	Опис
1	Кнопка [Обрати]	Дозволяє обрати композицію у якості референсу для генерації мелодії. Скеровує користувача на сторінку “Вибір інструменталу композиції”.
2	Контекстне меню програвача	Відкриває список дій, які можна здійснити над аудіофайлом.
3	Кнопка [Завантажити]	Дозволяє завантажити композицію на ПК користувача у форматі .wav.
4	Кнопка [Швидкість відтворення]	Дозволяє регулювати швидкість відтворення файлу.
5	Кнопка [Грати/Призупинити]	Починає/призупиняє відтворення аудіо.
6	Мішкер звуку	Дозволяє регулювати гусність відтворення.

Екранну форму сторінки “Список вибірок” наведено на рисунку 3.5.

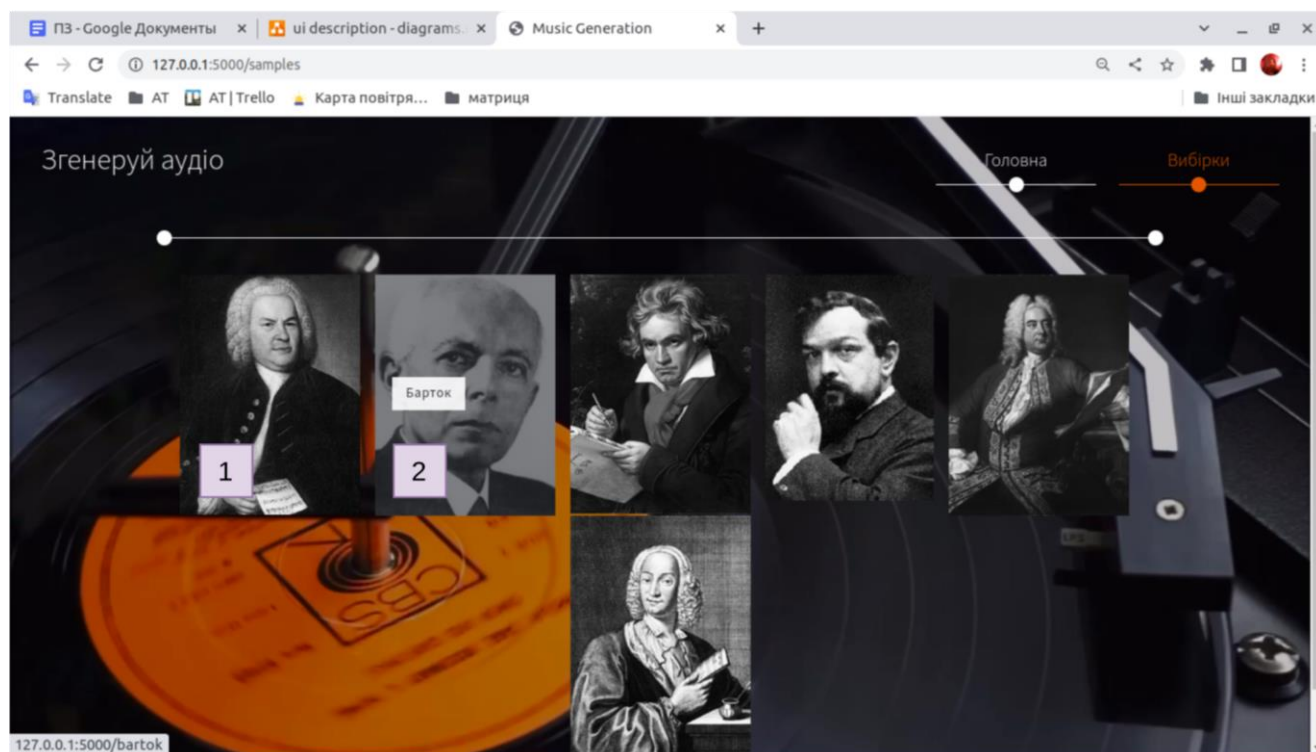


Рисунок 3.5 – Екранна форма “Список вибірок”

Опис екранної форми сторінки “Список вибірок” наведено у таблиці 3.6.

Таблиця 3.6 – Опис екранної форми сторінки “Список вибірок”

№	Тип елемента та назва	Опис
1	Кнопка [Обрати вибірку]	Дозволяє обрати вибірку композитора у якості референсу для генерації мелодії. Скеровує користувача на сторінку “Вибір інструменталу композиції”.
2	Кнопка [Обрати вибірку]	Вигляд кнопки при наведенні на неї курсору.

Екранну форму сторінки “Вибір інструменталу композиції” наведено на рисунку 3.6.

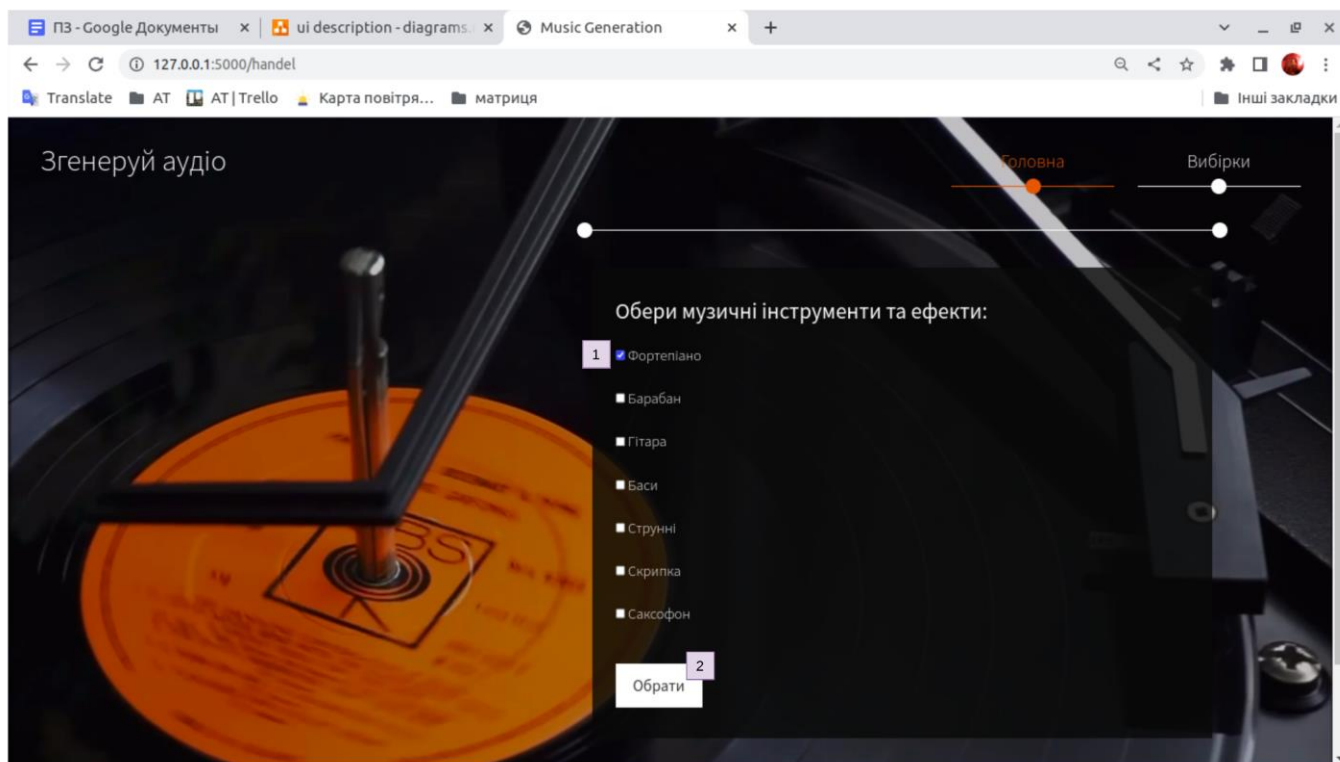


Рисунок 3.6 – Екранна форма “Вибір інструменталу композиції”

Опис екранної форми сторінки “Вибір інструменталу композиції” наведено у таблиці 3.7.

Таблиця 3.7 – Опис екранної форми сторінки “Вибір інструменталу композиції”

№	Тип елемента та назва	Опис
1	Чек-бокс	Заповнений чек-бокс додає вказаний інструмент до списку інструментів композиції, що генерується.
2	Кнопка [Обрати]	Зберігає вибір інструменталу та скеровує користувача на сторінку “Вибір нотного інтервалу” або “Результат” (залежно від сценарію).

Екранну форму сторінки “Вибір нотного інтервалу” наведено на рисунку 3.7.

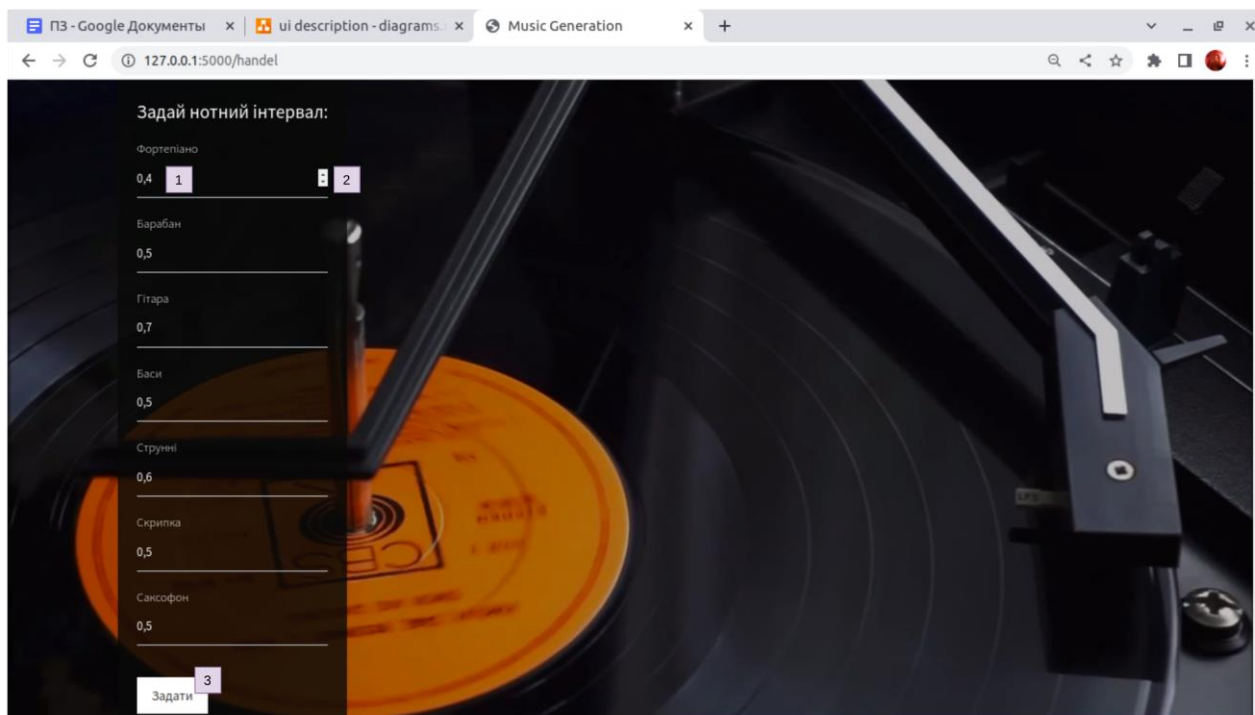


Рисунок 3.7 – Екранна форма “Вибір нотного інтервалу”

Опис екранної форми “Вибір нотного інтервалу” наведено у таблиці 3.8.

Таблиця 3.8 – Опис екранної форми сторінки “Вибір нотного інтервалу”

№	Тип елемента та назва	Опис
1	Текстове поле значення інтервалу	Дозволяє задати інтервал в форматі дробу (розділювач - кома) в діапазоні від 0 до 1.
2	Регулятори значення	З’являються при наведенні на текстове поле. Дозволяють збільшувати/зменшувати значення інтервалу на 0.1
3	Кнопка [Задати]	Зберігає задані нотні інтервали та скеровує користувача на сторінку “Результат”.

Екранну форму сторінки “Результат” наведено на рисунках 3.8 - 3.9.

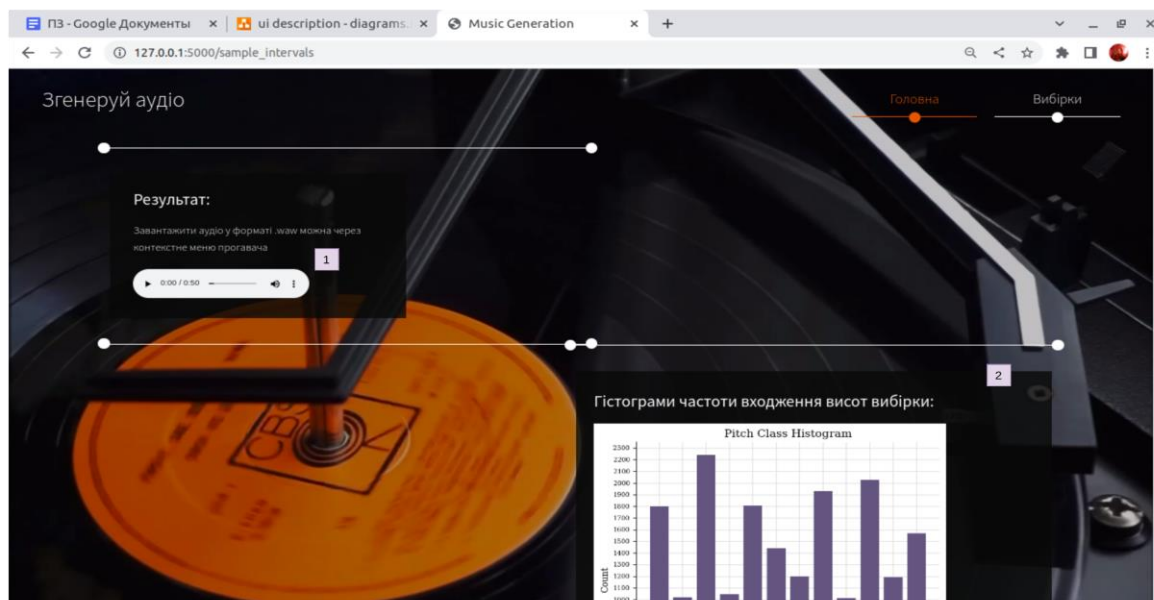


Рисунок 3.8 – Екранна форма сторінки “Результат”

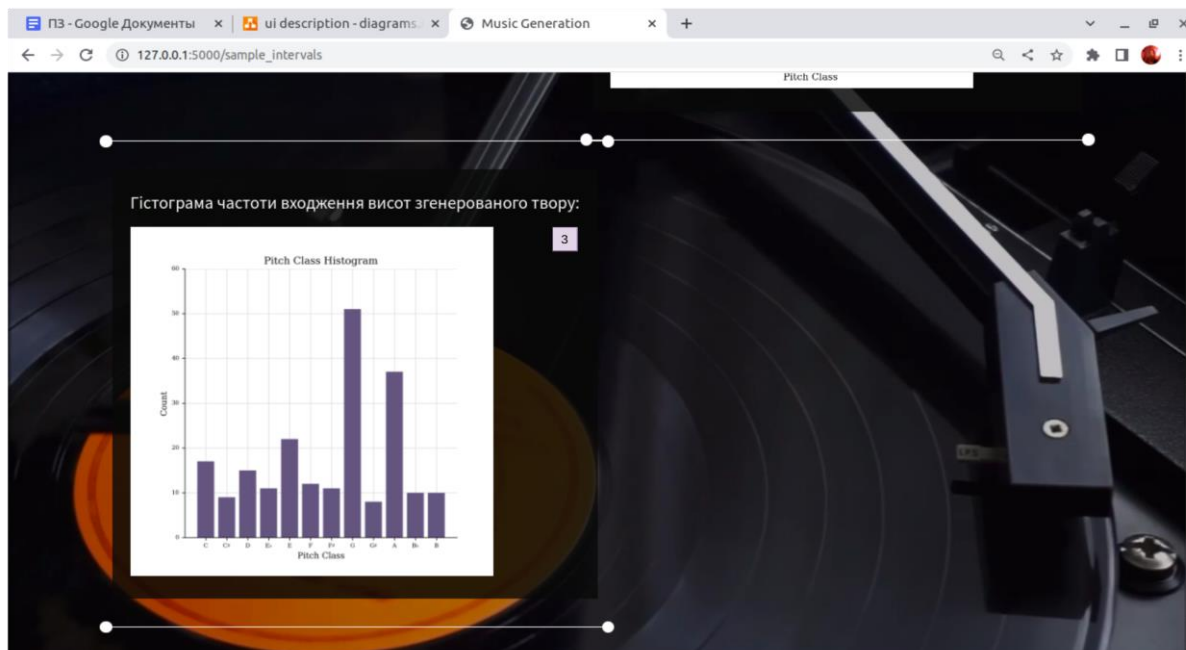


Рисунок 3.9 – Екранна форма сторінки “Результат”

Опис екранної форми сторінки “Результат” наведено у таблиці 3.9.

Таблиця 3.9 – Опис екранної форми сторінки “Результат”

№	Тип елемента та назва	Опис
1	Аудіопрогравач	Дозволяє прослуховувати та завантажувати згенерований файл у форматі .wav на ПК користувача (дії аналогічні до описаних в таблиці 3.5).
2	Гістограма частоти входження висот вибірки/вхідної композиції (залежно від сценарію)	Відтворює графік частоти входження частин певної висоти в композиції референсної вибірки/твору (залежно від сценарію).
3	Гістограма частоти входження висот згенерованого твору	Відтворює графік частоти входження частин певної висоти в згенеровану композицію.

3.3.2 Послідовність кроків виконання

Для генерації твору за користувацькою референсною композицією користувач:

1. Натискає кнопку [Вибрати файл] на сторінці “Головна” та обирає файл з файлової системи ПК.
2. Натискає кнопку [Завантажити] на сторінці “Головна”.
3. Заповнює форму з чек-боксами та натискає кнопку [Обрати] на сторінці “Вибір інструменталу композиції”.

Для генерації твору за референсною композицією зі списку вбудованих користувач:

1. Натискає кнопку [Обрати з плейлиста] на сторінці “Головна”.
2. Натискає кнопку [Обрати] біля вподобаної композиції на сторінці “Список композицій”.
3. Заповнює форму з чек-боксами та натискає кнопку [Обрати] на сторінці “Вибір інструменталу композиції”.

Для генерації твору за вбудованою вибіркою користувач:

1. Натискає кнопку [Вибірки] на будь-якій сторінці застосунку.
2. Натискає кнопку [Обрати вибірку] на сторінці “Список вибірок”
3. Заповнює форму з чек-боксами та натискає кнопку [Обрати] на сторінці “Вибір інструменталу композиції”.
4. Заповнює форму з текстовими полями та натискає кнопку [Задати] на сторінці “Вибір нотних інтервалів”.

Висновки до розділу

У даному розділі наведено опис засобів розробки. Окреслено архітектуру програмного забезпечення, а саме: наведено перелік класів, сценарій роботи системи, опис розгортання та специфікацію використаних методів. Наведено інструкцію користувача.

4 РОЗРОБЛЕННЯ СТАРТАП-ПРОЄКТУ

4.1 Опис ідеї проєкту

Ідея стартап проєкту полягає у використанні розробленого програмного застосунку у якості комерційного сервісу для генерації мелодій на основі референсних композицій або вибірок. Можливі напрямки розвитку ідеї наведено у таблиці 4.1.

Таблиця 4.1 – Опис ідеї стартап проєкту

Зміст ідеї	Напрямки застосування	Вигода для користувача
Веб-застосунок для генерації музики на основі референсної композиції/вибірки.	Генерація музичного супроводу реклами бренду в соціальних мережах.	Компанії малого бізнесу, що лише починають розвиток, можуть зекономити на придбанні ліцензії на використання музичного твору в рекламі/послугах професійного композитора. Окрім цього, музичний супровід їхньої реклами буде унікальним, що дозволить запам'ятатись клієнтові та встановити з ним емоційний контакт.
	Генерація музичного супроводу контенту в соціальних мережах для блогерів.	Дозволяє заощадити на покупці ліцензій та не обмежуватися набором дозволених соціальними мережами композицій.

Продовження таблиці 1.4

Зміст ідеї	Напрямки застосування	Вигода для користувача
	Генерація музичного супроводу до постановок/фільмів/кліпів студентів режисерських та акторських напрямків.	Дозволить створювати унікальні постановки та поширювати їх в соціальних мережах і на платформах без порушення авторського права.
	Платформа для популяризації творчої діяльності та заробітку.	Застосунок може використовуватись невідомими авторами музичних треків, оскільки у застосунку наявна можливість обрати референсну композицію з плейлиста або ж завантажити готову композицію з плейлиста. У випадку додавання можливості оплати скачаного твору, молоді автори можуть монетизувати та популяризувати свою діяльність за рахунок інших користувачів сервісу.

Для визначення сильних, нейтральних та слабких сторін розробки виконано порівняльний аналіз її техніко-економічних характеристик з конкурентами та наявними замінниками на ринку. Результати аналізу наведено в таблиці 4.2.

Таблиця 4.2 – Визначення сильних, слабких та нейтральних характеристик ідеї проекту

№	Техніко-економічні характеристики ідеї	(потенційні) товари/концепції конкурентів				W (слабкі)	N (нейтральні)	S (сильні)
		Мій проект	Jukebox	Magenta	Soundraw			
1	Є готовим до використання користувачем застосунком	+	-	-	+			+
2	Надає ліцензію на використання згенерованого фрагменту	-	-	-	+	+		
3	Дозволяє генерувати музичні фрагменти у різних стилях	+	+	+	+		+	
4	Дозволяє генерувати фрагменти на основі вбудованих вибірок	+	+	-	-			+

Продовження таблиці 4.2

№	Техніко- економічні характеристики ідеї	(потенційні) товари/концепції конкурентів				W	N	S
		Мій проект	Jukebox	Magenta	Soundraw			
5	Дозволяє генерувати фрагменти на основі референсної композиції користувача	+	-	+	-			+
6	Дозволяє обрати референсну композицію з вбудованого плейлиста	+	-	-	-			+
7	Дозволяє задавати користувацькі параметри композиції	+	+	+	+		+	
8	Накладає обмеження на користувацькі параметри зادля кращого звучання твору	-	-	-	+	+		

Продовження таблиці 4.2

№	Техніко- економічні характеристики ідеї	(потенційні) товари/концепції конкурентів				W	N	S
		Мій проект	Jukebox	Magenta	Soundraw			
9	Генерує декілька творів на основі заданих параметрів	-	-	-	+	-		
10	Масштабованість (можливість запуску в різних браузерах)	+	+	+	-		+	

Проект має декілька слабких сторін в порівнянні з рішенням Soundraw. Проте, на відміну від даного рішення, проект надає можливість генерації аудіофайлів за референсними композиціями та може підтримуватись у різних браузерах. Окрім цього, перевагою проекту є готовий користувацький інтерфейс, оскільки рішення Jukebox та Magenta надають лише програмний інструментарій та простий інтерфейс для тестування.

4.2 Технологічний аудит ідеї проекту

Для оцінки можливості технологічної реалізації проекту проведено технологічний аудит. Результати аудиту наведено в таблиці 4.3.

Таблиця 4.3 – Технологічна здійсненність ідеї проекту

№	Ідея проекту	Технології реалізації	Наявність технологій	Доступність технологій
1	Back-end частина веб-застосунку	Flask Framework	Наявна	Безкоштовна
		PyCharm IDE	Наявна	Доступна на платній (Professional) основі, має безкоштовну обмежену версію (Community)
2	Front-end частина веб-застосунку	Bootstrap	Наявна	Безкоштовна (вимагає зображення ліцензії та примітки про авторське право в коді)
		JavaScript	Наявна	Безкоштовна
3	Генерація музики на основі референсу	TensorFlow, Keras	Наявна	Безкоштовні, з відкритим кодом
		Music21	Наявна	Безкоштовна, з відкритим кодом
		Dlib	Наявна	Безкоштовна, з відкритим кодом
		NumPy	Наявна	Безкоштовна, з відкритим кодом
		Midi2Audio	Наявна	Безкоштовна, з відкритим кодом
		Google Colab GPU	Наявна	Доступна на платній основі, має безкоштовну квоту на використання

Усі необхідні технології для розробки застосунку наявні, більшість є доступними на безкоштовній основі. Можна зробити висновок, що проєкт можливо реалізувати з технічної точки зору.

4.3 Аналіз ринкових можливостей запуску стартап-проєкту

Для оцінки можливості запуску стартап-проєкту проведено аналіз поточної ситуації на потенційному ринку. Результати проведеного аналізу наведено в таблиці 4.4.

Таблиця 4.4 – Попередня характеристика потенційного ринку стартап-проєкту

№	Показники стану ринку (найменування)	Характеристика
1	Кількість головних гравців, од	10
2	Загальний обсяг продажів, \$	204000\$ на рік [25][26].
3	Динаміка ринку (якісна оцінка)	Зростає
4	Наявність обмежень для входу	Відсутність ліцензії, виробничих ресурсів (співробітників та техніки), відсутність ресурсу для розгортання застосунку.
5	Специфічні вимоги до стандартизації та сертифікації	ДСТУ ISO/IEC/IEEE 16326:2015 Розроблення систем та програмного забезпечення
6	Середня норма рентабельності в галузі (або по ринку), %	$\frac{204000\$}{723600\$} \times 100 \approx 28$

Кількість головних гравців на ринку невелика, оскільки переважно розробляються бібліотеки для вирішення задачі генерації музики, а не готові програмні застосунки. Динаміка ринку зростає, оскільки зростає попит на музичний супровід за рахунок поширення соціальних мереж. Загальний обсяг продажів та середня норма рентабельності в галузі дозволяє зробити висновок, що станом на даний момент ситуація на ринку є сприятливою.

Визначені потенційні групи клієнтів наведено в таблиці 4.5.

Таблиця 4.5 – Характеристика потенційних клієнтів стартап-проекту

№	Потреба, що формує ринок	Цільова аудиторія (цільові сегменти ринку)	Відмінності у поведінці різних потенційних цільових груп клієнтів	Вимоги споживачів до товару
1	Створення унікальних музичних фрагментів	Компанії малого бізнесу	Цільові групи мають різний рівень доходу та різну мету використання застосунку	Зрозумілий інтерфейс, якісна генерація музичних творів за референсами
		Блогери		
		Студенти творчих напрямків		
2	Популяризація своєї творчої діяльності	Невідомі автори музичних творів		Наявність функціоналу для відтворення списків композицій

Оскільки потенційні клієнти мають різні сфери застосування програмного рішення та різний рівень доходу, необхідно розробити декілька можливих варіантів підписок на послуги.

Фактори, що перешкоджають впровадженню проєкту на ринку, наведено в таблиці 4.6.

Таблиця 4.6 – Фактори загроз

№	Фактор	Зміст загрози	Можлива реакція компанії
1	Геополітична ситуація	Нестабільна геополітична ситуація, що спричиняє проблеми з інфраструктурою, завадить виконати запуск та просування проєкту	Релокація команди проєкту; розміщення сервісів на ресурсах за межами країни; відкладання запуску проєкту
2	Конкуренція	Стрімкий розвиток продукції конкурентів	Створення унікального функціоналу за рахунок опитувань відносно потреб цільових груп
3	Відсутність попиту	Користувачі не будуть зацікавлені в застосунках	Запуск реклами та «трендів» у соціальних мережах з використанням музики застосунку

Фактори, що сприяють впровадженню проєкту на ринку, наведено в таблиці 4.7.

Таблиця 4.7 – Фактори можливостей

№	Фактор	Зміст можливості	Можлива реакція компанії
1	Розвиток технологій	Поява нових форматів аудіо та програм для аналізу музики	Розширення математичної моделі виявлення характеристик музичних творів для покращення результату генерації

Продовження таблиці 4.7.

№	Фактор	Зміст можливості	Можлива реакція компанії
2	Розширення потреб	Поява нових потреб користувачів	Розширення функціоналу
3	Співпраця з маловідомими музикантами	Фінансова співпраця з маловідомими музикантами: музиканти платять за розміщення музики на сервісі, проєкт платить музикантам за завантаження їхніх композицій та використання їх в якості референсу	Залучення музикантів до фінансової співпраці; розробка сервісу для монетизації музикантів; розробка функціоналу, що дозволяє розміщувати свої твори на платформі; закупівля додаткових ресурсів для зберігання даних.

Опис загальних рис конкуренції на ринку представлено в таблиці 4.8.

Таблиця 4.8 – Ступеневий аналіз конкуренції на ринку

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства (можливі дії компанії, щоб бути конкурентоспроможною)
Монополістичний тип конкуренції	Продукти кожної фірми не є досконалими замінниками одне одного; присутня кореляція цін між конкурентами	Характеристики розробки повинні покривати недостачу функціоналу конкурентів; ціна повинна урівнюватись з середньринковою

Продовження таблиці 4.8

Особливості конкурентного середовища	В чому проявляється дана характеристика	Вплив на діяльність підприємства
Світовий рівень конкурентної боротьби	Усі продукти ринку є інтернаціональними	Популяризація продукту серед потенційних клієнтів за межами країни
Внутрішньогалузева конкуренція	Конкуренція лише між продуктами, пов'язаними з генерацією музики	Виявлення та задоволення унікальних потреб користувачів в галузі
Товарно-видова конкуренція	Конкурують послуги (товари) одного виду	Робота над вдосконаленням алгоритмів генерації музичних творів
Нецінова перевага	Перевага за рахунок якості згенерованого твору та унікального функціоналу	Робота над зручністю інтерфейсу та якістю результатів
Не марочна	Торгова марка відіграє не значну роль	Застосовувати в маркетинговій стратегії результати роботи продукту, без акценту на бренд

Детальніший аналіз умов конкуренції за моделлю 5 сил М. Портера наведено у таблиці 4.9.

Таблиця 4.9 – Аналіз конкуренції в галузі за М. Портером

	Прямі конкуренти в галузі	Потенційні конкуренти	Постачальники	Клієнти	Товари-замінники
Складові аналізу	Jukebox Magenta Soundraw Musenet WaveNet	Можливе створення програмних продуктів, що використовуватимуть відкриті технології для створення музики	Постачальники забезпечення можуть змінювати ціни та політики використання	Є споживачі як з прибутком вище середнього, так і з незначним прибутком	Вигідніша ціна, ширший функціонал
Висновки	Широка нецінова конкуренція на світовому ринку	На ринку є потенційні конкуренти, проте можливість виходу на ринок цілком реальні	Постачальники не диктують умови на ринку, оскільки продукти мають замінники	Варіативність цін повинна дозволяти приваблювати клієнтів різного рівня доходу	Розширення функціоналу, робота над ціновою політикою

З проведеного аналізу можна зробити висновок, що можливість для входу на ринок є. Продукт є конкурентоспроможним, оскільки жоден продукт на ринку не є повністю взаємозамінним, тому можна привабити цільову аудиторію функціоналом,

відсутнім у конкурентів. Обґрунтування факторів конкурентоспроможності представлено в таблиці 4.10.

Таблиця 4.10 – Обґрунтування факторів конкурентоспроможності

№	Фактор конкурентоспроможності	Обґрунтування
1	Врахування потреб споживачів	Періодично проводитиметься опитування користувачів та розроблятиметься функціонал, що відповідатиме новим потребам
2	Цінова політика	Розробка цінової політики, що варіюється відповідно до затребуваного користувачем функціоналу та середнього рівня доходів країни користувача (оскільки товар представлятиметься на міжнародному ринку)
3	Можливість співпраці з маловідомими музикантами	Продукт виділяється серед конкурентів тим, що є не лише сервісом для генерації музики, але й сервісом для прослуховування та завантаження готових творів
4	Маркетингова компанія	Залучення соціальних мереж до просування продукту

Таблиця 4.11 – Порівняльний аналіз сильних та слабких сторін проєкту

№	Фактор конкурентоспроможності	Бали 1-20	Рейтинг товарів-конкурентів у порівнянні з проєктом						
			-3	-2	-1	0	+1	+2	+3
1	Врахування потреб споживачів	17							+
2	Цінова політика	15	+						
3	Можливість співпраці з маловідомими музикантами	17	+						
4	Маркетингова компанія	10			+				

На основі зібраних загроз та можливостей, а також, слабких і сильних сторін, побудовано SWOT-матрицю. Матрицю представлено у таблиці 4.12.

Таблиця 4.12 – SWOT- аналіз стартап-проєкту

Сильні сторони: – можливість співпраці з музикантами; – задоволення потреб споживачів.	Слабкі сторони: – вузька ніша; – висока собівартість запуску.
Можливості: – розвиток технологій; – розширення потреб споживачів.	Загрози: – геополітична ситуація; – конкуренція.

На основі SWOT-аналізу та огляду потенційних проєктів конкурентів розроблено модель альтернативної ринкової поведінки для виведення стартап-проєкту на ринок (табл. 4.13).

Таблиця 4.13 – Альтернативи ринкового впровадження стартап-проєкту

№	Альтернатива ринкової поведінки	Ймовірність отримання ресурсів	Строки реалізації
1	Таргетована реклама	Необхідно вкладати власні кошти; ймовірна за наявності інвесторів	2-4 місяці
2	Запуск «трендів» у соціальних мережах з музичними супровадами застосунку	Ресурси наявні	1 місяць
3	Надання користувачам апробаційного періоду	Ресурси наявні	3 місяці

Альтернативними шляхами виходу на ринок можуть стати надання користувачам апробаційного періоду в поєднанні з запуском «трендів» у соціальних мережах. «Тренди» допоможуть ознайомити аудиторію з товаром, а апробаційний період – накопичити базу користувачів застосунку.

4.4 Розроблення ринкової стратегії проєкту

Для прийняття рішення про сегмент аудиторії проведено аналіз цільових груп користувачів. Опис цільових груп наведено в таблиці 4.14.

Таблиця 4.14 – Вибір цільових груп потенційних споживачів

№	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
1	Блогери, що потребують музичний супровід для контенту	Висока, особливо у соціальній мережу TikTok	90%	Для застосунків конкурентів – низька, проте необхідно конкурувати з авторами наявного звукового супроводу	Увійти в сегмент просто

Продовження таблиці 4.14

№	Опис профілю цільової групи потенційних клієнтів	Готовність споживачів сприйняти продукт	Орієнтовний попит в межах цільової групи (сегменту)	Інтенсивність конкуренції в сегменті	Простота входу у сегмент
2	Маловідомі музиканти	Середня, оскільки наявні такі аналогові продукти для популяризації творчості як SoundCloud, YouTube	50%	Висока, оскільки потрібно конкурувати зі стрімінговими платформами	Середня складність входу в сегмент – вхід можливий після популяризації застосування
3	Компанії малого бізнесу	Середня, оскільки з одного боку продукт є для них корисним, з іншого – компанії швидше вкладатимуть гроші в готові композиції	30%	Висока	Достатньо складно, оскільки бізнес готовий співпрацювати лише з перевіреними рішеннями

Таким чином, при вході на ринок варто обрати стратегію концентрованого маркетингу з орієнтацією на сегмент блогерства. Після популяризації застосування блогерами, у ньому цілком ймовірно зацікавляться маловідомі музиканти. У цьому випадку слід застосувати стратегію диференційованого маркетингу та розробляти додатковий функціонал, що допоможе маловідомим авторам музики популяризувати та монетизувати свою творчість. Удаватись до масового маркетингу та залучати клієнтів малого бізнесу можна за декілька років, коли можна буде зібрати статистику про вдалу генерацію музики та вдосконалити функціонал.

Опис базової стратегії розвитку проекту наведено в таблиці 4.15.

Таблиця 4.15 – Визначення базової стратегії розвитку

Обрана альтернатива розвитку проекту	Стратегія охоплення ринку	Ключові конкурентоспроможні позиції відповідно до обраної альтернативи	Базова стратегія розвитку
Представлення можливостей, що відсутні у конкуренти	Маркетингова кампанія; залучення цільової аудиторії унікального функціоналу до просування застосування; розробка нового функціоналу відповідно до потреб споживачів	Додаткові можливості застосування; відмінні характеристики	Стратегія диференціації

Визначення базової стратегії ринкової поведінки окреслено в таблиці 4.16.

Таблиця 4.16 – Визначення базової стратегії конкурентної поведінки

Чи є проект «першопрохідцем» на ринку?	Чи буде компанія шукати нових споживачів, або забирати існуючих у конкурентів?	Чи буде компанія копіювати основні характеристики товару конкурента, і які?	Стратегія конкурентної поведінки
Ні, наявні конкуренти, що вже працюють на ринку. Проте сфера набула популярності відносно нещодавно, тому конкуренція незначна.	Компанія шукатиме нових користувачів, яких приваблюватиме функціоналом відмінним від конкурентського. У випадку вдалої маркетингової компанії може вдатись відити споживачів у існуючих конкурентів.	Частково характеристики можуть здаватись скопійованими, оскільки реалізують ідентичний функціонал генерації. Проте для генерації використовуватимуться інші підходи та моделі, тому результат відрізнятиметься.	Стратегія заняття конкурентної ніші

З врахуванням вимог користувачів, обраних для входу на ринок сегментів (блогери та маловідомі музиканти), та орієнтацією на базові стратегії розвитку (стратегія диференціації) і конкурентної поведінки (стратегія заняття конкурентної ніші), розроблено стратегію позиціонування стартап-проекту. Стратегія

позиціонування передбачає вибір комплексу асоціацій, за якими користувачі повинні ідентифікувати продукт. Огляд стратегії наведено в таблиці 4.17.

Таблиця 4.17 – Визначення стратегії позиціонування

№	Вимоги до товару цільової аудиторії	Базова стратегія розвитку	Ключові конкурентоспроможні позиції власного стартап-проєкту	Вибір асоціацій, які мають сформувати комплексну позицію власного проєкту
1	Якісна генерація музики	Диференціації	Можливість генерувати музику за користувацьким референсом	Генерація музики за вподобаннями
2	Популяризація музики	Диференціації	Можливість завантаження готових треків або обрання їх в якості референсу	Популяризація творчості; бібліотека музики
3	Зручність використання	Диференціації	Підказки щодо сценарію генерації музики	Інтуїтивний інтерфейс

4.5 Розроблення маркетингової програми стартап-проєкту

Для формування маркетингової концепції застосунку використано аналіз його конкурентоспроможності. Перелік ключових переваг концепції застосунку наведено в таблиці 4.18

Таблиця 4.18 – Визначення ключових переваг концепції потенційного товару

№	Потреба	Вигода, яку пропонує товар	Ключові переваги перед конкурентами
1	Генерація музичних творів	Генерація на основі користувацького референсу	Застосунок надає можливість згенерувати мелодію на основі референсу, завантаженого з ПК користувача
2	Популяризація творчості	Наявність вбудованого плейлиста, з якого можна завантажувати готову мелодію або обирати її у якості референсу	Вбудований плейлист уже наявний. Необхідно розробити: – функціонал, що надасть авторам музики можливість викладати свої твори на ресурс; – функціонал для оплати завантаження мелодії та переказу частини коштів автору музики.
3	Зручність використання	Інтуїтивно зрозумілий інтерфейс	Наявні підказки на екранних формах щодо сценарію використання. Необхідно розробити екранну форму з довідником користувача
4	Оформлення підписки на послуги	Можливість генерувати музику регулярно за вигідною ціною	Необхідно розробити: – систему для контролю взаємодії з клієнтами (CRM); – модуль реєстрації; – оформлення підписки; – інтеграцію з платіжними системами.

На основі визначеної концепції сформовано трирівневу маркетингову модель товару (табл. 4.19)

Таблиця 4.19 – Опис трьох рівнів моделі товару

Рівні товару	Сутність та складові		
I. Товар за задумом	Застосунок для генерації музики за користувацькими референсом та параметрами		
II. Товар у реальному виконанні	Властивості/характеристики	М/Нм	Вр/Тх /Тл/Е/Ор
	1. Інтерфейс (виконаний у чорних, оранжевих та білих відтінках; містить вказівки щодо виконання кроків сценаріїв генерації)	М	Тх/Ор
	2. Генерує музику за референсом тривалістю від 5 до 9 хвилин	Нм	Тл/Тх
	3. Генерує музику за вибіркою тривалістю 52 секунди	М	Тл/Тх
	4. Якісна генерація музики	М	Тл/Тх/Ор
	Якість: відповідність вимогам ДСТУ ISO/IEC/IEEE 16326:2015 Розроблення систем та програмного забезпечення.		
	Пакування: електронна технічна документація, електронна довідка користувача		
III. Товар із підкріпленням	До продажу: підготовка застосунку, консультації зі спеціалістами щодо юридичних аспектів та фінансової інтеграції, побудова бизнес-стратегії.		
	Після продажу: гарантія; ліцензія; підтримка; регулярне оновлення функціоналу		
За рахунок чого потенційний товар буде захищено від копіювання: патент			

Визначення меж встановлення ціни наведено в таблиці 4.20 (під товарами заміниками розглядається дозвіл на використання готової мелодії у якості музичного супроводу).

Таблиця 4.20 – Визначення меж встановлення ціни

Рівень цін на товари-замінники	Рівень цін на товари-аналоги	Рівень доходів цільової групи споживачів	Верхня та нижня межі встановлення ціни на товар/послугу
30-50\$	19.9\$ за місяць використання	500-2125\$ на місяць	8-12\$ за місяць користування, 0.3\$ за одноразове використання

Визначення системи збуту наведено в таблиці 4.21.

Таблиця 4.21 – Формування системи збуту

Специфіка закупівельної поведінки цільових клієнтів	Функції збуту, які має виконувати постачальник товару	Глибина каналу збуту	Оптимальна система збуту
Клієнти купують послугу безпосередньо у виробника шляхом оформлення підписки	Технічна підтримка клієнтів, аналіз ринку задля врегулювання цін	0 (товар продається безпосередньо виробником)	Веб-застосунок

Концепцію маркетингових комунікацій, що спирається на попередньо обрану основу для позиціонування та визначену специфіку поведінки клієнтів наведено в таблиці 4.22.

Таблиця 4.22 – Концепція маркетингових комунікацій

Специфіка поведінки цільових клієнтів	Канали комунікацій, якими користуються цільові клієнти	Ключові позиції, обрані для позиціонування	Завдання рекламного повідомлення	Концепція рекламного звернення
Дізнаються про товар через соціальні мережі	Інтернет і соціальні мережі	Продукт з цікавим функціоналом, задовільняє потреби сегменту	Проінформувати про існування продукту та зацікавити користувача у тому, щоб спробувати функціонал застосунку	Рекламне відео показує генерацію музики в застосунку, після чого людина накладає музику на своє відео для соціальних мереж

Висновки до розділу

У даному розділі було описано ідею стартап-проєкту. Проведено технологічний аудит, за результатами якого продукт можливо реалізувати (технології існують та мають доступну ціну). Виконано аналіз ринкових можливостей, який показав, що проєкт є конкурентоспроможним. Розроблено ринкову стратегію: базова стратегія – диференціація; стратегія конкурентної поведінки – заняття конкурентної ніші. Встановлено маркетингову стратегію, а саме, визначено ключові переваги продукту та концепцію його збуту, встановлено межі цін.

ВИСНОВКИ

У процесі написання роботи було проведено огляд предметного середовища, а саме, вивчено наявні рішення для генерації музичних творів та виконано аналіз досліджень, пов'язаних з вирішенням даної задачі. За мету було поставлено надання користувачеві можливості генерації мелодії на основі референсного фрагменту та користувацьких параметрів. Окреслено сценарій взаємодії користувача з застосунком.

Для вирішення задачі генерації було вивчено наявні підходи, а саме: використання стохастичних процесів, граматик, оптимізаційних методів та штучного інтелекту. Було обрано підхід ланцюгів Маркова для генерації мелодії на основі однієї референсної композиції та підхід генеративно-змагальних нейронних мереж для генерації на основі вбудованих вибірок. У роботі наведено обґрунтування вибору методів та огляд результатів їхньої роботи.

Визначено архітектуру програмного забезпечення та технології його реалізації. Розроблено веб-застосунок для генерації музики. У роботі наведено специфікацію методів застосунку. Створено інструкцію користувача.

У рамках розробки стартап-проєкту сформовано опис ідеї, проведено технологічний аудит, обрано ринкову стратегію поведінки та розроблено маркетингову стратегію. Встановлено, що умови для виходу на ринок цілком сприятливі. Застосунок є конкурентоспроможним. Окреслено можливості та характеристики застосунку, які необхідно додати перед введенням розробки на ринок.

СПИСОК ЛІТЕРАТУРИ

1. Гавриленко О.В., Дмитрук О.Ю, А.В. Тетерук. Використання математичних моделей та засобів штучного інтелекту для автоматизації написання музичних творів. Вісник Національного транспортного університету. Серія «Технічні науки». Науковий журнал. – К.: НТУ, 2022. – 3 (53).
2. Can anybody hear me? The power of music in advertising. URL: <https://www.kantar.com/inspiration/advertising-media/can-anybody-hear-me-the-power-of-music-in-advertising> (дата звернення: 04.12.2022).
3. Huron D. Music in advertising: an analytic paradigm. The musical quarterly. 1989. Т. 73, № 4. С. 557–574. URL: <https://academic.oup.com/mq/article-abstract/73/4/557/1143144?redirectedFrom=fulltext> (дата звернення: 04.12.2022).
4. Jon D. Morris, Mary Anne Boone. The effects of music on emotional response, brand attitude, and purchase intent in an emotional advertising condition. Advances in consumer research. 1998. Т. 25. С. 518–526. URL: <https://www.acrwebsite.org/volumes/8207/volumes/v25/NA-25> (дата звернення: 04.12.2022).
5. Jukebox. URL: <https://openai.com/blog/jukebox/> (дата звернення: 04.12.2022).
6. Autoregressive long-context music generation with Perceiver AR / Catalina Cangea та ін. URL: <https://magenta.tensorflow.org/perceiver-ar> (дата звернення: 04.12.2022).
7. Freely customize High quality Royalty-free music. URL: <https://soundraw.io/>.
8. Shulei Ji, Jing Luo, Xinyu Yang. A Comprehensive Survey on Deep Music generation: Multi-level Representations, Algorithms, Evaluations, and Future Directions. 2020 URL: <https://arxiv.org/pdf/2011.06801.pdf>.
9. Shuqi Dai. Personalized Popular Music Generation Using Imitation and Structure. 2021 URL: <https://arxiv.org/pdf/2105.04709.pdf>.
10. Aashiq Muhamed. Symbolic Music Generation with Transformer-GANs. 2021 URL: <https://ojs.aaai.org/index.php/AAAI/article/view/16117>.

11. Ilana Shapiro, Mark Huber. Markov Chains for Computer Music Generation. Journal of Humanistic Mathematics. 2021. T. 11 URL: <https://scholarship.claremont.edu/cgi/viewcontent.cgi?article=1848&context=jhm>
12. Monika Jadwiga Galla. Stochastic processes and probability theory in music. 2015. URL: <https://etheses.whiterose.ac.uk/13286/1/Thesis%20Final%20Corrections%20MG.pdf> (дата звернення: 04.12.2022).
13. Frank Zalkow, Stephan Brand, Benjamin Graf. Musical style modification as an optimization problem. Proceedings of the international computer music conference. 2016. C. 206–211. URL: <https://quod.lib.umich.edu/cgi/p/pod/dod-idx/musical-style-modification-as-an-optimization-problem.pdf?c=icmc;idno=bbp2372.2016.041;format=pdf> (дата звернення: 04.12.2022).
14. Iyad Abu Doush, Ayah Sawalha. Automatic music composition using genetic algorithm and artificial neural networks. Malaysian journal of computer science. 2020. T. 33, № 1. С. 35–51.
15. Luca Casini, Bob L. T. Sturm. Tradformer: a transformer model of traditional music transcriptions. Proceedings of the thirty-first international joint conference on artificial intelligence (IJCAI-22). C. 4915–4920. URL: <https://www.ijcai.org/proceedings/2022/0681.pdf> (дата звернення: 04.12.2022).
16. Yugesh Verma. A Hands-On Guide to Automatic Music Generation using RNN. URL: <https://analyticsindiamag.com/a-hands-on-guide-to-automatic-music-generation-using-rnn/> (дата звернення: 04.12.2022).
17. Hasim Sak, Andrew Senior, Francoise Beaufays. Long Short-Term Memory Recurrent Neural Network Architectures for Large Scale Acoustic Modeling URL: <https://static.googleusercontent.com/media/research.google.com/en//pubs/archive/43905.pdf>.
18. Music generation with local connected convolutional neural network / Zhihao Ouyang та ін. URL: <http://ouyangzhihao.com/wp-content/uploads/2018/12/MUSIC-GENERATION-WITH-LOCAL-CONNECTED-CONVOLUTIONAL-NEURAL-NETWORK.pdf> (дата звернення: 04.12.2022).

19. What is a Generative Model?. URL: <https://developers.google.com/machine-learning/gan/generative> (дата звернення: 04.12.2022).
20. Sepp Hochreiter. Long short-term memory. Neural computation. 1997. Т. 9, № 8. URL: <http://bioinf.jku.at/publications/older/2604.pdf> (дата звернення: 04.12.2022).
21. Diederik P. Kingma, Jimmy Lei Ba. Adam: a method for stochastic optimization. Iclr. 2015. URL: <https://arxiv.org/pdf/1412.6980.pdf> (дата звернення: 04.12.2022).
22. Django documentation. URL: <https://www.djangoproject.com/> (дата звернення: 04.12.2022).
23. Flask's documentation. URL: <https://flask.palletsprojects.com/en/2.0.x/> (дата звернення: 04.12.2022).
24. PyCharm. URL: <https://www.jetbrains.com/pycharm/>.
25. AuthorRose Salia. Top 10 AI Music Generators 2022. URL: <https://topten.ai/music-generators-review/> (дата звернення: 04.12.2022).
26. Soundraw startup. URL: <https://startup.network/startups/447616.html> (дата звернення: 04.12.2022).

ДОДАТОК К. ЛІСТИНГ ЧАСТИНИ КОДУ

```
from flask import Flask, render_template, request
from werkzeug.utils import secure_filename
from music21 import *
from instruments import Instruments
from midi2audio import FluidSynth
from markov_chain import MarkovChainAudio

UPLOAD_FOLDER = '/'
ALLOWED_EXTENSIONS = {'mid', 'midi'}
app = Flask(name)
app.secret_key = "super secret key"
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER

FILENAME = ""
INSTRUMENTS = []
INTERVALS = []
SAMPLE_GAN = ""
GAN_MUSIC = ""
SAMPLE_FOLDER = ""

def allowed_file(filename):
    return '.' in filename and \
        filename.rsplit('.', 1)[1].lower() in ALLOWED_EXTENSIONS

@app.route('/music', methods=['GET', 'POST'])
def upload_file():
    if request.method == 'POST':
        file = request.files['audiofile']
        if file and allowed_file(file.filename):
            global FILENAME
            filename = secure_filename(file.filename)
            FILENAME = filename
            print(FILENAME)
            f = os.path.join(app.config['UPLOAD_FOLDER'], filename)
            file.save(f)
            return render_template('index-instruments.html')
        else:
            return "Невірний формат файлу. Спробуй завантажити файл .mid/.midi :)"
    else:
        return render_template('index.html')
```

```

@app.route('/instruments', methods=['GET', 'POST'])
def render_instruments():
    if request.method == 'POST':
        global INSTRUMENTS, FILENAME
        INSTRUMENTS = request.form.getlist('instrument')
        markov_chain = MarkovChain ()
        str = (".../" + FILENAME)
        Convlist = markov_chain.parse_midi_to_list(str)
        Chainfirstorder = markov_chain.create_markov_chain(Convlist)
        markov_chain.buildsong(Chainfirstorder, 500, INSTRUMENTS)
        s = converter.parse(str)
        p = graph.plot.HistogramPitchClass(s)
        p.run()
        p.write("")
        s = converter.parse('/')
        p = graph.plot.HistogramPitchClass(s)
        p.run()
        p.write("")
        s = converter.parse('/')
        p = graph.plot.HistogramPitchClass(s)
        p.run()
        p.write("/")
        # s = converter.parse('/')
        # p = graph.plot.HorizontalBarPitchSpaceOffset(s)
        # p.run()
        # p.write("/")
        # render_intervals(filename)
        return render_template('markov_result.html')
    else:
        return render_template("index-instruments.html")

@app.route('/playlist_instruments/<int:_id>', methods=['GET', 'POST'])
def render_from_playlist(_id):
    if request.method == 'POST':
        global INSTRUMENTS, FILENAME
        INSTRUMENTS = request.form.getlist('instrument')
        markov_chain = MarkovChain ()
        Convlist = markov_chain.parse_midi_to_list (_str)
        Chainfirstorder = markov_chain.create_markov_chain (Convlist)
        markov_chain.buildsong(Chainfirstorder, 500, INSTRUMENTS)
        s = converter.parse(_str)
        p = graph.plot.HistogramPitchClass(s)

```

```

    p.run()
    p.write("/...")
    s = converter.parse('/')
    p = graph.plot.HistogramPitchClass(s)
    p.run()
    p.write("/...")
    return render_template('markov_result.html')
else:
    _action= '/playlist_instruments/' + str(_id)
    return render_template("sample_instruments.html", action=_action)
@app.route('/samples')
def render_samples():
    return render_template("samples.html")
@app.route("/<composer>", methods=['GET', 'POST'])
def render_composer (composer):
    global SAMPLE_GAN, GAN_MUSIC, SAMPLE_FOLDER
    GAN_MUSIC = '/'
    SAMPLE_GAN = '/... generator.h5'
    SAMPLE_FOLDER = "composer"
    if request.method == 'POST':
        global INSTRUMENTS
        INSTRUMENTS = request.form.getlist('instrument')
        print(INSTRUMENTS)
        return render_template('sample_intervals.html')
    else:
        return render_template("sample_instruments.html", action="/ ")
@app.route("/sample_intervals", methods=['GET', 'POST'])
def render_sample_result():
    global SAMPLE_GAN, INSTRUMENTS, INTERVALS, GAN_MUSIC
    if request.method == 'POST':
        INTERVALS = [request.form["piano"], request.form["drum"], request.form["guitar"],
                     request.form["bass"], request.form["string"], request.form["violin"],
                     request.form["saxophone"]]
        midi_stream = stream.Score()
        for _instrument in INSTRUMENTS:
            p = Instruments().part_generate_model(SAMPLE_GAN, _instrument, INTERVALS, GAN_MUSIC)
            midi_stream.insert(0, p)
        midi_stream.write('midi', fp= '/')
        s = converter.parse('/')
        p = graph.plot.HistogramPitchClass(s)
        p.run()
        p.write("/")

```

```
    fs = FluidSynth()
    fs.midi_to_audio('/',          '/')
    return gan_result()
else:
    return render_template("sample_intervals.html")
@app.route('/result', methods=['GET'])
def gan_result():
    return render_template('gan_result.html', composer=SAMPLE_FOLDER)
if name == 'main':
    app.run(debug=True)
```