

Міністерство освіти і науки України
Харківський національний університет імені В.Н. Каразіна
Навчально-науковий інститут комп'ютерних наук та штучного інтелекту

Рекомендовано до захисту:

протокол засідання кафедри

№ ____ від ____ . ____ .2025 р.

Завідувач кафедри _____

(підпис)

(прізвище та ініціали)

Пояснювальна записка

до кваліфікаційної роботи бакалавра
на тему «Розробка музичного ритм-тренажеру з алгоритмами аналізу
точності»

Захищено на засіданні ЕК № _____

протокол № ____ від ____ . ____ .2019 р.

Оцінка ____ / _____

Голова ЕК

(підпис)

(прізвище та ініціали)

Виконав:

студент 4 курсу, групи КС-43

Спеціальності:

122 Комп'ютерні науки

(шифр і назва спеціальності підготовки)

Ложкін Микита Іванович

(прізвище, ім'я та по батькові, підпис)

к.т.н. Малахова М.О.

(ступень, звання, прізвище та ініціали, підпис)

Рецензент _____

(ступень, звання, прізвище та ініціали, підпис)

МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Харківський національний університет імені В. Н. Каразіна

Навчально-науковий інститут комп'ютерних наук та штучного інтелекту
Кафедра кібербезпеки інформаційних систем, мереж і технологій
Рівень вищої освіти (освітньо-кваліфікаційний рівень) Бакалавр
Галузь знань: 12 Інформаційні технології
Спеціальність: 122 Комп'ютерні науки
Освітня програма: Комп'ютерні науки

ЗАТВЕРДЖУЮ
Завідувач кафедри ІПСІТ
Олег ОЛЕШКО
«___» _____ 20__ року

З А В Д А Н Н Я
НА КВАЛІФІКАЦІЙНУ РОБОТУ (ПРОЕКТ)

Ложкін Микита Іванович
(прізвище, ім'я, по батькові студента)

1. Тема роботи: Розробка музичного ритм-тренажеру з алгоритмами аналізу точності

керівник роботи Малахова Марина Олегівна, кандидат технічних наук,
(прізвище, ім'я, по батькові, науковий ступінь, вчене звання)

затверджені наказом по університету від “16” квітня 2025 року №4101-5/962

2. Строк подання студентом роботи 02 червня 2025 року.

3. Перелік питань, які потрібно розробити:

Розробка концепції музичного тренажера для розвитку ритму. Вибір алгоритмів аналізу ритмічної точності. Побудова архітектури клієнт-серверної вебсистеми. Визначення функціональних і нефункціональних вимог до системи. Вибір оптимального стеку технологій для реалізації. Проектування інтерфейсу користувача з урахуванням принципів UX/UI. Реалізація механізмів генерування ритмічних вправ та збору статистики. Розробка алгоритму оцінки точності виконання вправ у реальному часі. Тестування застосунку та аналіз відповідності поставленим завданням. Обґрунтування новизни, ефективності та потенційних сфер використання розробки.

4. План роботи

№ з/п	Назви етапів роботи	Термін виконання етапів роботи
1	Аналіз існуючих рішень та постановка задачі	Січень-початок лютого
2	Вибір алгоритмів аналізу ритмічної точності	Лютий
3	Розробка архітектури додатку	Лютий
4	Реалізація програмного забезпечення	Березень-Квітень
5	Тестування додатку	Квітень
6	Оцінка роботи системи та внесення змін	Квітень-початок травня
7	Підготовка пояснювальної записки	Травень
8	Підготовка доповіді та презентації	Травень

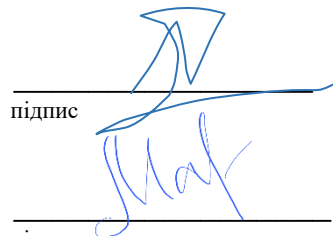
5. Дата видачі завдання 10 жовтня 2025 року.

Студент

Ложкін М. І.

ініціали, прізвище

підпис



Керівник роботи

Малахова М.О.

ініціали, прізвище

підпис



АНОТАЦІЯ

Пояснювальна записка до кваліфікаційної роботи містить 87 сторінок, включає 3 розділи, 3 рисунки, 2 таблиці, 1 додаток, 2 формули, а також 20 джерел у переліку посилань. Робота структурована відповідно до вимог щодо оформлення бакалаврських кваліфікаційних робіт.

У роботі розглянуто питання розробки та впровадження музичного ритм-тренажера з алгоритмами аналізу точності. Висвітлено сучасний стан цифрових технологій у сфері музичної освіти, проведено огляд ритм-тренажерів, досліджено алгоритми оцінювання ритмічної точності. Обґрунтовано вибір архітектури системи, розглянуто інструменти та мови програмування, а також описано основні етапи реалізації функціоналу тренажера. Метою роботи є створення функціонального та інтуїтивно зрозумілого вебзастосунку, що дозволяє користувачу покращувати навички ритмічного відчуття за допомогою інтерактивних вправ та точного аналізу виконання. У дослідженні використано методи інженерії програмного забезпечення, структурного аналізу, а також моделювання взаємодії компонентів вебзастосунку. Для реалізації проєкту застосовано HTML, CSS, JavaScript та бібліотеку WebAudioFont.

Результатом роботи є створений музичний тренажер, що дозволяє генерувати ритмічні вправи, аналізувати точність виконання та зберігати статистику успішності користувача. Новизна роботи полягає в поєднанні ритм-тренувального підходу з автоматизованим аналізом точності у форматі зручного вебінтерфейсу. Результати роботи можуть бути використані в освітніх цілях, зокрема для самостійного навчання ритміці, а також як основа для подальшого розвитку музичних освітніх застосунків, орієнтованих на індивідуальне навчання.

Ключові слова: МУЗИЧНИЙ ТРЕНАЖЕР, РИТМ, АНАЛІЗ ТОЧНОСТІ, ВЕБЗАСТОСУНОК, ПРОГРАМНЕ ЗАБЕЗПЕЧЕННЯ, JAVASCRIPT, ВЕБ-ДИЗАЙН, ІНТЕРАКТИВНЕ НАВЧАННЯ, ОСВІТНІ ТЕХНОЛОГІЇ.

ABSTRACT

The explanatory note to the bachelor's qualification work contains 87 pages and includes 3 chapters, 3 figures, 2 tables, 1 appendix, and 8 sources listed in the bibliography. The work is structured in accordance with the requirements for the formatting of bachelor's qualification works.

This paper examines the development and implementation of a musical rhythm trainer with accuracy analysis algorithms. It covers the current state of digital technologies in the field of music education, reviews existing rhythm trainers, and investigates algorithms for evaluating rhythmic accuracy. The system architecture is justified, the development tools and programming languages are analyzed, and the key stages of functionality implementation are described. The goal of the work is to create a functional and intuitive web application that allows users to improve their sense of rhythm through interactive exercises and precise performance analysis. The research uses methods of software engineering, structural analysis, and modeling of web application component interaction. The implementation of the project involves HTML, CSS, JavaScript and the WebAudioFont library.

The result is a musical training tool that enables the generation of rhythmic exercises, accuracy assessment, and user progress tracking. The novelty of the work lies in the combination of rhythm training methodology with automated accuracy analysis presented in a user-friendly web interface. The results of this work may be used for educational purposes, particularly for self-study in rhythm, and as a foundation for further development of educational music applications aimed at personalized learning.

Keywords: MUSICAL TRAINER, RHYTHM, ACCURACY ANALYSIS, WEB APPLICATION, SOFTWARE, JAVASCRIPT, WEB DESIGN, INTERACTIVE LEARNING, EDUCATIONAL TECHNOLOGIES.

ЗМІСТ

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ	8
ВСТУП	9
1. ОГЛЯД ЛІТЕРАТУРНИХ ДЖЕРЕЛ ТА СУЧАСНОГО СТАНУ РОЗРОБОК У СФЕРІ МУЗИЧНИХ ДОДАТКІВ	12
1.1. Загальні відомості про музичне навчання з використанням цифрових технологій.	12
1.1.1. Історія розвитку цифрових методів навчання музиці	12
1.1.2. Поява та поширення ритм-тренажерів.....	13
1.2. Аналіз сучасних інструментів та систем для ритмічного навчання.....	15
1.2.1. Огляд комерційних рішень.....	16
1.2.2. Огляд академічних та дослідницьких розробок.....	18
1.2.3. Порівняльний аналіз функціональних можливостей	20
1.3. Алгоритми аналізу точності у музичних додатках	22
1.3.1. Проблематика оцінювання ритмічної точності.....	23
1.3.2. Поширені підходи до аналізу точності ритмічного виконання.....	24
1.3.3. Приклади імплементації у програмних рішеннях	26
1.4. Висновки до розділу	27
2. РОЗРОБКА МУЗИЧНОГО РИТМ-ТРЕНАЖЕРА З АЛГОРИТМАМИ АНАЛІЗУ ТОЧНОСТІ	30
2.1. Постановка задачі та вимоги до системи	30
2.1.1. Формулювання основної мети та задач	30
2.1.2. Опис загальних функціональних та нефункціональних вимог	32
2.1.3. Визначення цільової аудиторії.....	34
2.2. Архітектура системи.....	36
2.2.1. Загальна структура системи та модулів.....	37
2.2.2. Взаємодія між компонентами фронтенду та бекенду	39
2.3. Вибір технологій та інструментів розробки.....	41

2.3.1. Мова програмування, бібліотеки, фреймворки.....	42
2.3.2. Причини вибору технологічного стеку	43
2.3.3. Порівняльна характеристика альтернативних технологій.....	45
2.4. Реалізація основних функцій тренажера	46
2.4.1. Генерація ритмічного прикладу та відтворення звуку	47
2.4.2. Введення відповіді користувачем.....	48
2.4.3. Перевірка правильності та точності ритму	50
2.4.4. Реалізація двох режимів: “фортепіано” та “перкусія”	51
2.5. Механізм аналізу точності виконання	53
2.5.1. Збір і обробка таймінгів.....	54
2.5.2. Алгоритм розрахунку точності	55
2.5.3. Візуалізація результатів та зворотний зв’язок користувачу	57
2.6. Дизайн інтерфейсу та взаємодія з користувачем	58
2.6.1. Загальні принципи дизайну та UX	58
2.6.2. Оформлення елементів інтерфейсу, кольорова палітра	60
2.7. Розширені можливості та додатковий функціонал	61
2.7.1. Ведення статистики користувача	62
2.7.2. Скидання статистики, підрахунок середніх значень	63
2.7.3. Можливості для майбутнього розширення	64
3. ВИСНОВКИ ДО РЕЗУЛЬТАТІВ РОБОТИ.....	66
3.1. Оцінка досягнення поставленої мети	66
3.2. Аналіз результатів та їх відповідність сучасним тенденціям	67
3.3. Можливі напрями використання та подальшого розвитку системи	68
ВИСНОВКИ.....	71
ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ	73
ДОДАТОК А. ПРИКЛАДИ ВИХІДНОГО КОДУ	75

ПЕРЕЛІК СКОРОЧЕНЬ, УМОВНИХ ПОЗНАЧЕНЬ, ТЕРМІНІВ

ВЗ – веб-застосунок;

ІТ – інформаційні технології;

ПЗ – програмне забезпечення;

СЧ – серверна частина.

AI – Artificial Intelligence (штучний інтелект);

BPM – Beats Per Minute (ударів на хвилину);

CSS – Cascading Style Sheets (каскадні таблиці стилів);

HTML – HyperText Markup Language (мова розмітки гіпертексту);

JS – JavaScript (мова сценаріїв для веброзробки);

MIDI – Musical Instrument Digital Interface (цифровий інтерфейс музичних інструментів);

UX – User Experience (досвід користувача).

ВСТУП

У сучасному світі роль цифрових технологій у галузі музичної освіти набуває дедалі більшого значення. Широке впровадження інформаційних технологій у процес навчання сприяє не лише підвищенню ефективності засвоєння матеріалу, а й формуванню нових підходів до викладання та самостійного опанування навичок. Особливої актуальності ці тенденції набувають у сфері початкової музичної підготовки, зокрема — в навчанні ритму, яке традиційно вважалося одним із найскладніших аспектів музичної грамотності. Ритмічний слух, відчуття метроритму та здатність точно відтворювати ритмічні структури є фундаментальними навичками для будь-якого музиканта, незалежно від інструменту чи стилю виконання.

Значна частина музичних педагогів відзначає, що недосконале володіння ритмом часто стає причиною труднощів у подальшому розвитку учня. На цьому етапі надзвичайно важливим є використання сучасних цифрових інструментів, які дозволяють зробити навчальний процес більш наочним, інтерактивним і доступним. Різноманітні онлайн-платформи, мобільні додатки та програмне забезпечення, що моделює виконання ритмічних вправ, стали предметом активного вивчення та застосування як в Україні, так і за її межами.

Проте, попри значну кількість вже наявних рішень, існує низка суттєвих проблем, які залишаються невирішеними. Більшість ритмічних тренажерів мають обмежений функціонал, що зводиться до простого відтворення ритму та його повторення користувачем. Системи оцінювання, як правило, є надто загальними та не надають чіткого зворотного зв'язку щодо точності виконання. У багатьох випадках користувач не має змоги побачити, в чому саме полягає його помилка — у ритмічному рисунку, таймінгу або неправильному сприйнятті структури. Це, у свою чергу, ускладнює процес самонавчання та гальмує розвиток ритмічного слуху.

Іноземні аналоги, зокрема платформи типу Rhythm Trainer, EarMaster, Complete Rhythm Trainer, мають доволі розвинений інтерфейс та функціонал, проте часто орієнтовані на англомовну аудиторію, є комерційними або потребують складної реєстрації. У вітчизняному ж сегменті освітнього програмного забезпечення подібних рішень практично немає, особливо — тих, що були б україномовними, адаптованими під потреби студентів музичних шкіл, початківців та викладачів. Це створює об'єктивну нішу для розробки нових програмних продуктів, які б поєднували в собі зручність використання, доступність, адаптивність до рівня користувача та наочність зворотного зв'язку.

Об'єктом дослідження є процес формування та розвитку ритмічних навичок у користувача за допомогою цифрових інструментів у контексті музичної освіти.

Предметом дослідження виступає програмна реалізація інтерактивного ритм-тренажера з функціями генерації вправ, аналізу точності виконання та надання візуального зворотного зв'язку.

Метою дипломної роботи є розробка україномовного веб-застосунку (ВЗ) — музичного ритм-тренажера, що дозволяє користувачу проходити ритмічні вправи, отримувати оцінку своєї точності у реальному часі, а також отримувати аналітичний звіт про виконання з метою самонавчання та вдосконалення навичок.

Для досягнення поставленої мети у роботі необхідно виконати такі завдання:

- Проаналізувати наявні програмні засоби для тренування ритму;
- Обґрунтувати вибір технологій для реалізації веб-застосунку;
- Розробити архітектуру програмного забезпечення;
- Реалізувати генерацію ритмічних вправ із різною довжиною та складністю;
- Забезпечити можливість аудіовідтворення та візуального зображення вправ;

- Реалізувати систему аналізу точності виконання ритму користувачем;
- Запровадити систему обліку статистики користувача;
- Забезпечити привабливий, зручний та адаптивний інтерфейс.

Актуальність теми визначається необхідністю запровадження сучасних, інтерактивних, локалізованих рішень у процес музичного навчання. В умовах розвитку дистанційного навчання та популяризації самоосвіти, зростає попит на прості та ефективні інструменти для розвитку базових музичних навичок. Реалізований у межах цієї роботи тренажер дозволяє не лише виконувати ритмічні вправи, а й отримувати миттєвий якісний аналіз їх точності — що надає користувачу можливість самостійно відслідковувати власний прогрес.

Новизна розробки полягає у поєднанні інтуїтивно зрозумілого інтерфейсу, реалізації двох режимів роботи (фортепіанний та перкусійний), можливості реального аналізу ритмічної точності та збереження статистики в межах одного компактного, незалежного від серверної частини застосунку. Також проєкт вирізняється повною україномовною локалізацією, адаптацією до освітніх цілей та відкритістю до подальших розширень.

Таким чином, розроблений у дипломній роботі тренажер є прикладом застосування веб-технологій у сфері музичної освіти, демонструє можливість створення ефективного інструменту з навчальним та аналітичним потенціалом та, водночас, є внеском у розвиток національного сегменту навчального програмного забезпечення.

1. ОГЛЯД ЛІТЕРАТУРНИХ ДЖЕРЕЛ ТА СУЧАСНОГО СТАНУ РОЗРОБОК У СФЕРІ МУЗИЧНИХ ДОДАТКІВ

1.1. Загальні відомості про музичне навчання з використанням цифрових технологій.

Упродовж останніх десятиліть спостерігається стрімкий розвиток цифрових технологій, що вплинув практично на всі галузі людської діяльності, зокрема й на сферу музичної освіти. Традиційні методики навчання музиці, засновані на особистому спілкуванні викладача з учнем, поступово доповнюються і навіть замінюються інноваційними інструментами, що базуються на використанні програмного забезпечення, мобільних застосунків та інтерактивних платформ [1].

Цей підрозділ спрямований на висвітлення загальних тенденцій у застосуванні цифрових технологій у музичному навчанні, починаючи з історичних передумов їх виникнення, до сучасних спеціалізованих тренажерів, зокрема ритмічних, які сьогодні активно використовуються як в інституційній освіті, так і для самостійного опанування музичних навичок [2].

1.1.1. Історія розвитку цифрових методів навчання музиці

Витоки цифрового музичного навчання сягають другої половини ХХ століття, коли з'явилися перші спроби використання комп'ютерів у процесі навчання музичної теорії та гри на інструментах [3]. Спершу ці системи мали вигляд програмованих інструкцій або мультимедійних презентацій на базі мейнфреймів чи перших персональних комп'ютерів. Попри свою обмеженість, вони стали першими прикладами поєднання технологій з музичним навчальним процесом.

З появою MIDI (Musical Instrument Digital Interface) у 1983 році відкрилися нові горизонти для розробки навчальних програм, що дозволяли не лише відтворювати звуки, але й аналізувати гру користувача в режимі

реального часу [4]. Програмне забезпечення стало здатне фіксувати точність, темп, інтервали, силу натискання клавіш, що суттєво підвищило інтерактивність та об'єктивність навчального процесу. Ці зміни сприяли переходу від одностороннього викладання до динамічної взаємодії між системою та користувачем.

У 1990-х роках відбувся стрімкий розвиток мультимедійних CD-ROM програм, орієнтованих на вивчення нотної грамоти, основ гармонії, техніки гри на інструментах [5]. Згодом з'явилися інтерактивні програми, такі як Sibelius Student, EarMaster, Piano Marvel, що дозволяли не лише навчатись теорії, а й тренуватись у практичних аспектах гри на інструменті. Таким чином, цифрові методи навчання музиці почали набувати ознак повноцінної навчальної екосистеми.

З початку XXI століття розвиток інтернету, мобільних пристроїв та хмарних технологій зробив цифрове музичне навчання ще доступнішим [6]. Онлайн-платформи, інтерактивні мобільні застосунки, системи з доповненою реальністю (AR) розширили можливості для самостійного навчання. Тепер учні мають доступ до широкого спектру курсів, вправ, методичних матеріалів незалежно від місця перебування.

Сьогодні цифрові технології у музичній освіті стали не лише допоміжним засобом, а й важливим інструментом, що змінює саму парадигму навчального процесу — від статичного до динамічного, від механічного повторення до адаптивного навчання, що враховує індивідуальні особливості користувача [7].

1.1.2. Поява та поширення ритм-тренажерів

Ритміка є одним з базових компонентів музичної грамотності, тому поява інструментів для її опанування стала логічним продовженням розвитку цифрових технологій у сфері музичного навчання. Ритм-тренажери — це спеціалізовані програмні продукти або мобільні додатки, що дозволяють

користувачу тренувати ритмічне відчуття через виконання вправ з визначеним темпом, тривалістю нот і пауз, метричними структурами тощо [8].

Перші ритм-тренажери були досить простими. Вони пропонували користувачу повторювати задану ритмічну послідовність, прослуховуючи її у вигляді звукових сигналів або візуального відображення. Згодом ці тренажери отримали функції зворотного зв'язку — система почала фіксувати відхилення користувача від ритмічного еталону та надавати оцінку точності виконання[9]. Це дозволило значно покращити ефективність занять.

Сьогодні існує безліч сучасних ритм-тренажерів, таких як Rhythm Sight Reading Trainer, Complete Rhythm Trainer, Soundbrenner, Rhythm Lab та інші[10]. Вони поєднують високоточні алгоритми аналізу ритмічної точності з гейміфікаційними елементами, що мотивують користувача до регулярних занять. Багато з них мають адаптивну систему складності, що підлаштовується під рівень користувача, а також можливість інтеграції з MIDI-клавіатурами, сенсорними панелями або мобільними пристроями.

Ритм-тренажери активно використовуються як в індивідуальному навчанні, так і в академічному середовищі [11]. Викладачі музичних шкіл та університетів впроваджують їх у навчальні програми для розвитку внутрішнього ритму, точності виконання, координації рухів, що особливо важливо для виконавців на ударних інструментах, фортепіано, духових та струнних інструментах.

Отже, зростання популярності ритм-тренажерів є обґрунтованим та логічним кроком у напрямі цифрової трансформації музичної освіти [12]. Це інструменти, що дозволяють значно покращити якість навчального процесу, зробити його більш інтерактивним, гнучким і доступним для широкого кола користувачів.

1.2. Аналіз сучасних інструментів та систем для ритмічного навчання

В наш час цифрові технології значно змінили парадигму музичної освіти. Особливо активно це спостерігається у впровадженні інноваційних навчальних платформ і тренажерів, що дозволяють удосконалювати специфічні музичні навички, зокрема ритмічні. Завдяки інтеграції інформаційних технологій у сферу музичної педагогіки стало можливим створення адаптивних, персоналізованих і динамічних інструментів навчання, які враховують індивідуальні потреби, здібності та темп засвоєння матеріалу кожного учня.

Сучасні музичні тренажери вирізняються широким спектром функцій, серед яких можна виокремити: аналіз точності відтворення ритму, візуалізацію нотного матеріалу, зворотний зв'язок у режимі реального часу, моделювання ситуацій гри в ансамблі, гейміфікацію навчального процесу, а також інтеграцію з мобільними платформами й хмарними сервісами. Розробники програмних продуктів намагаються охопити як новачків, так і професійних музикантів, пропонуючи різні рівні складності, жанрові підходи та підтримку різноманітних музичних інструментів.

На сьогодні існує велика кількість програмних рішень, спрямованих на розвиток ритмічного відчуття. Деякі з них мають виключно тренувальну функцію, інші — включають повноцінні навчальні модулі та комплексні освітні курси. Усі ці інструменти розрізняються за функціональністю, педагогічною ефективністю, рівнем технічної реалізації та ціною, що обумовлює необхідність їх систематичного аналізу та порівняння.

Окрім суто комерційних продуктів, важливе місце займають також академічні розробки, що створюються у межах дослідницьких проєктів, університетських лабораторій або з підтримкою міжнародних грантів. Такі продукти зазвичай спрямовані на глибоке вивчення аспектів музичного навчання, мають відкритий код або безкоштовну модель розповсюдження та виконують не лише освітню, а й експериментальну функцію.

З урахуванням вищевикладеного, у наступних пунктах підрозділу буде здійснено огляд як найпоширеніших комерційних рішень, так і академічних проєктів, а також проведено порівняльний аналіз їх функціональних характеристик з метою виявлення тенденцій та прогалин у сучасному ринку музичних тренажерів для ритмічного навчання.

1.2.1. Огляд комерційних рішень

На сучасному етапі розвитку цифрових технологій на ринку навчального програмного забезпечення для музикантів спостерігається значне зростання кількості комерційних рішень, які пропонують інноваційний підхід до формування музичних навичок. Ці рішення зазвичай є продуктами команд розробників, які поєднують у своїй діяльності програмістів, музичних педагогів, дизайнерів інтерфейсів та спеціалістів із когнітивної психології. У результаті створюються гнучкі, адаптивні системи, що відповідають запитам сучасного користувача, орієнтовані на різний рівень підготовки та мають високий ступінь персоналізації.

Однією з найбільш впізнаваних і широко застосовуваних платформ є Yousician — інтерактивна програма, яка підтримує навчання гри на гітарі, фортепіано, бас-гітарі, укулеле та вокалу. Система використовує мікрофон пристрою для аналізу виконання та надає зворотний зв'язок у режимі реального часу. Інтерфейс адаптується до рівня користувача, пропонуючи поступове ускладнення завдань. Програма містить гейміфіковані елементи, що підвищують мотивацію до навчання, зокрема візуальні винагороди, рейтинги та систему досягнень. Водночас слід зазначити, що ритмічному компоненту в Yousician приділяється лише часткова увага — переважно у вигляді вправ на таймінг, синхронізацію та правильне розміщення нот у часовому просторі.

Інше популярне рішення — Melodics, що фокусується насамперед на розвитку ритмічного мислення та моторики, зокрема при роботі з MIDI-контролерами, електронними ударними установками та клавіатурами. Melodics відзначається яскравим, динамічним інтерфейсом, великою

кількістю вправ та можливістю побудови індивідуальної траєкторії навчання. Система пропонує користувачеві вправи у вигляді ритмічних патернів, які необхідно повторити з максимальною точністю. Користувач отримує оцінку якості виконання з детальним розбором помилок, що сприяє глибшому розумінню власних слабких сторін. Незважаючи на комерційний характер, Melodics активно позиціонує себе як навчальний інструмент для музикантів-початківців і професіоналів, а також викладачів, які використовують цифрові тренажери у своїй педагогічній практиці.

У контексті навчання нотної грамоти варто згадати програму Tenuto, яка є доповненням до більш масштабної онлайн-платформи MusicTheory.net. Tenuto орієнтована на розвиток базових навичок музичної теорії: розпізнавання нот, інтервалів, акордів, ритмічних структур. Інтерфейс програми є стриманим, без зайвої візуальної динаміки, що робить її зручною для освітнього середовища. Ритмічні вправи у Tenuto дозволяють тренуватися у впізнаванні тривалостей, акцентів, синкоп, а також у побудові правильного ритмічного малюнка. Програма дає змогу самостійно створювати завдання з різними рівнями складності, що дозволяє її використовувати як у формальному навчанні, так і для самостійної підготовки.

Ще одним прикладом програмного засобу, що спеціалізується на розвитку ритмічного слуху, є Rhythm Sight Reading Trainer. Ця програма призначена для музикантів, які прагнуть вдосконалити навички читання з листа ритмічного нотного тексту. Вона дозволяє практикуватися в різних розмірах, темпах і ритмічних структурах. Особливістю програми є наявність голосового відліку, візуального таймера та можливості поступової побудови складності вправ. Rhythm Sight Reading Trainer активно використовується в музичних школах і коледжах, особливо у тих випадках, коли виникає потреба в інтенсивній роботі над читанням ритмічних рисунків.

Таким чином, сучасні комерційні рішення, що спеціалізуються на музичному навчанні, демонструють широкий спектр підходів до реалізації ритмічного компонента. Деякі з них (як-от Melodics чи Rhythm Sight Reading

Trainer) акцентують увагу саме на розвитку ритму, тоді як інші (наприклад, Yousician або Tenuto) інтегрують ритм у ширший контекст музичної підготовки. Всі ці рішення мають різний рівень деталізації аналізу ритмічного виконання, ступінь адаптивності та можливості персоналізації, що створює підґрунтя для їх подальшого аналізу з точки зору ефективності та доцільності використання в різних освітніх середовищах.

1.2.2. Огляд академічних та дослідницьких розробок

Окрім комерційних продуктів, які активно використовуються широким загалом, вагомий внесок у розвиток цифрових музичних тренажерів здійснюють академічні установи, науково-дослідні лабораторії та університетські спільноти. Саме в академічному середовищі часто формуються концептуальні основи, на яких згодом ґрунтуються комерційні реалізації. Крім того, дослідницькі розробки зазвичай орієнтовані не лише на масового користувача, а й на більш глибоке вивчення психофізіологічних, когнітивних та педагогічних аспектів музичного навчання.

У світовій практиці відомі численні приклади використання ритм-тренажерів у межах експериментальних освітніх програм або як частини дослідницьких проєктів. Наприклад, у рамках освітніх ініціатив провідних університетів (зокрема, Massachusetts Institute of Technology, Stanford University, University of Rochester) розроблялися інтерактивні платформи, які дозволяли відслідковувати точність ритмічного виконання студентами музичних факультетів. У таких проєктах акцент робився не тільки на зручність використання, а й на обґрунтованість алгоритмів оцінювання, їх відповідність педагогічним принципам та науковим підходам.

Деякі академічні дослідження зосереджуються на вивченні ритмічного сприйняття та моторної координації під час гри на інструменті. Наприклад, у роботах, що проводилися в межах європейських дослідницьких програм Erasmus+ та Horizon 2020, вивчалися методики використання біо-зворотного зв'язку (biofeedback) у ритм-тренажерах. Це дозволяло інтегрувати дані про

частоту серцевих скорочень, м'язову активність або рівень стресу для більш індивідуалізованого підходу до навчання.

Окрему увагу в академічному середовищі приділяють вивченню інтерфейсів взаємодії у музичних тренажерах. Дослідники аналізують, які типи візуального зворотного зв'язку, кольорових маркерів, анімацій або звукових сигналів є найбільш ефективними для формування стабільного ритмічного навичку. У деяких експериментах застосовуються навіть технології eye-tracking для відстеження рухів очей під час навчання та виявлення критичних зон інтерфейсу, які потребують покращення. Це дозволяє з'ясувати, як учень взаємодіє з візуальною інформацією і як це впливає на результат.

Академічні розробки, на відміну від комерційних, часто мають відкритий код або надаються у вільному доступі для наукового використання. Це створює умови для розвитку відкритої спільноти розробників та дослідників, які можуть вдосконалювати, модифікувати або застосовувати наявні рішення у власних освітніх практиках. Наприклад, у рамках проєкту OpenMusicTheory, що реалізується кількома північноамериканськими університетами, створено низку інтерактивних тренажерів, зокрема з ритміки, які не тільки навчають, але й дозволяють оцінити прогрес користувача.

Попри свою академічну спрямованість, такі розробки часто демонструють глибший рівень інноваційності та відкривають нові напрями в дослідженні музичної педагогіки. Вони нерідко стають підґрунтям для магістерських та докторських дисертацій, а також — прототипами для комерційних продуктів майбутнього. Зокрема, окремі інститути розробляють повноцінні навчальні середовища на базі хмарних технологій, які можуть використовуватися не тільки у вищих навчальних закладах, а й у дистанційній освіті або в роботі з дітьми з особливими потребами.

У цілому, академічні розробки відіграють важливу роль у формуванні фундаменту для нових підходів до музичного навчання. Їхній вклад полягає не лише у створенні інструментів, а й у виробленні методологій, критеріїв оцінки ефективності та педагогічної доцільності цифрових засобів. Це особливо

важливо в умовах зростання ролі цифрових технологій в освіті загалом та у музичній підготовці зокрема.

1.2.3. Порівняльний аналіз функціональних можливостей

Після розгляду прикладів комерційних та академічних рішень у сфері музичних ритм-тренажерів доцільно провести порівняльний аналіз їхніх функціональних характеристик. Такий аналіз дозволяє виявити спільні риси, унікальні особливості, а також наявні обмеження, що мають ключове значення при розробці нових або вдосконаленні існуючих навчальних систем.

Одним із основних критеріїв порівняння є рівень інтерактивності. У більшості сучасних комерційних рішень, таких як Yousician чи Melodics, реалізовано високий рівень інтерактивності завдяки використанню звукового розпізнавання у реальному часі, інтерфейсів з візуальним зворотним зв'язком та систем досягнень, що стимулюють користувача. Програми такого типу дозволяють учню миттєво отримувати зворотній зв'язок про точність та ритмічність свого виконання, що сприяє формуванню стійких навичок через адаптивне навчання.

Натомість академічні розробки частіше зосереджуються на точності аналізу, науковій обґрунтованості методів оцінки та можливості детального налаштування навчального процесу. Наприклад, у деяких дослідницьких проєктах особлива увага приділяється алгоритмам вимірювання ритмічної точності, що базуються на математичному моделюванні, а не лише на співпадінні у часі. Такі системи враховують варіативність людського виконання, дозволяючи відрізнити незначні відхилення, які не мають критичного впливу на загальне ритмічне відчуття, від суттєвих помилок.

Ще одним важливим аспектом є масштабованість та адаптивність навчальних платформ. Комерційні сервіси зазвичай орієнтовані на масового користувача й пропонують попередньо визначені рівні складності, що забезпечують швидкий старт, проте можуть бути недостатньо гнучкими для індивідуального підходу. Академічні рішення, натомість, часто надають

можливість гнучкого налаштування параметрів — як з боку викладача, так і самого учня, включно з вибором тривалості вправи, типів ритмічних фігур, метричних шаблонів тощо.

Крім того, до уваги береться обсяг навчального матеріалу та варіативність контенту. У комерційних додатках користувач має доступ до великого каталогу вправ, що охоплюють різноманітні музичні стилі та рівні складності. Більшість таких сервісів постійно оновлюється завдяки платній підписці або великій спільноті викладачів і користувачів. Академічні системи, у свою чергу, можуть обмежуватися навчальними модулями, створеними у рамках одного курсу, проте часто доповнені теоретичними поясненнями, які дозволяють глибше зрозуміти принципи побудови ритму.

Особливу увагу також приділяють сумісності з пристроями. Сучасні тренажери здебільшого кросплатформні, що дає можливість використовувати їх як на комп'ютерах, так і на мобільних пристроях. Деякі з них навіть передбачають підтримку MIDI-інтерфейсу або інтеграцію з цифровими інструментами, що значно розширює можливості навчання. У той же час частина академічних прототипів обмежена роботою у певному середовищі або не адаптована до широкого кола пристроїв, що ускладнює їх використання за межами дослідницького контексту.

Підсумовуючи порівняльний аналіз, можна зазначити, що обидва напрями розвитку мають власні сильні сторони. Комерційні рішення вирізняються зручністю, широким охопленням та гейміфікацією, що робить їх привабливими для початківців і самостійного навчання. Академічні — пропонують глибину, точність та адаптивність, що особливо важливо для освітніх установ і цілеспрямованого формування ритмічної компетентності. Саме синтез цих підходів — інтеграція точності та гнучкості академічних рішень з привабливістю та доступністю комерційних — може стати оптимальною моделлю для створення ефективного ритм-тренажера нового покоління.

1.3. Алгоритми аналізу точності у музичних додатках

У межах цифрового навчання музиці особливу роль відіграє наявність ефективного механізму зворотного зв'язку, що дозволяє користувачам не лише повторювати певні вправи, а й аналізувати власні помилки та прогрес у реальному часі. Саме завдяки наявності алгоритмів аналізу точності виконання можна забезпечити об'єктивне оцінювання дій користувача, яке слугує основою для формування ефективної траєкторії навчання.

Зі зростанням популярності музичних додатків, призначених як для початкового, так і для поглибленого навчання, все більше уваги приділяється саме питанням точності: наскільки чітко користувач влучає у ритмічну сітку, чи правильно виконує темп, інтервали, а також чи є його виконання синхронним щодо очікуваної моделі. Аналіз таких параметрів є надзвичайно важливим — як для мотивації учня, так і для ефективного контролю засвоєного матеріалу.

З одного боку, точність можна оцінювати у вузькому розумінні — як розбіжність у часі між очікуваним та фактичним натисканням. З іншого боку, у сучасних програмах усе частіше використовуються складніші метрики, які враховують варіативність виконання, темп, акценти та інші елементи ритмічної організації. Саме тому, на мою думку, огляд підходів до аналізу точності є необхідним етапом дослідження теми розробки ритм-тренажера.

У наступних пунктах даного підрозділу мною буде висвітлено низку важливих питань, пов'язаних з проблематикою оцінювання ритмічної точності, розглянуто сучасні підходи до реалізації алгоритмів аналізу виконання, а також наведено приклади їх імплементації у програмних рішеннях, що вже існують. Такий огляд дозволить не лише систематизувати знання, а й обґрунтувати обраний у дипломному проєкті підхід до побудови аналізатора точності.

1.3.1. Проблематика оцінювання ритмічної точності

Оцінювання ритмічної точності є одним із ключових завдань музичних освітніх програм, які орієнтовані на практичне закріплення навичок гри на інструменті або вокального виконання. Особливої актуальності це питання набуває у цифрових тренажерах, де замість викладача контроль за правильністю виконання здійснює програмна система. Проте, попри зовнішню простоту задачі — «натисни вчасно» — вона приховує низку складнощів як технічного, так і методологічного характеру.

По-перше, сам по собі термін «ритмічна точність» є дещо абстрактним, оскільки не має єдиного канонічного визначення. В різних контекстах точність може розумітися як відсутність помітних темпових коливань, як мінімальна розбіжність між очікуваним та фактичним моментом звукової події або як здатність зберігати синхронізацію з метрономом чи акомпанементом. Усе це зумовлює варіативність у підходах до аналізу, що ускладнює розробку універсального інструменту оцінювання.

По-друге, важливо враховувати, що ритміка в музиці не завжди є абсолютно математичною. Наприклад, у джазі або фолкових жанрах легітимними є відхилення від метрономічної точності, які надають виконанню «живого» характеру. Така «творча неточність» з погляду цифрового аналізатора може бути розцінена як помилка, хоча у музичному контексті це — цілком допустима варіація. Отже, система повинна не лише порівнювати час подій, але й інтерпретувати контекст.

По-третє, слід враховувати технологічні обмеження апаратури та середовища виконання. Так, затримка (latency) при натисканні клавіші чи обробці звуку може суттєво впливати на зафіксовані значення таймінгу. У професійному середовищі подібні речі компенсуються програмними методами калібрування або згладжуванням, однак у освітніх тренажерах така функціональність реалізується не завжди або виконується спрощено. Це створює ризик несправедливого оцінювання учня навіть у разі об'єктивно вдалого виконання.

Крім того, проблемою залишається і питання чутливості аналізу. Якщо встановити надто жорсткі критерії — це може демотивувати початківців. Якщо ж система буде надто «поблажливою», вона втратить свою навчальну ефективність. На мою думку, ідеальна система має бути адаптивною, підлаштовуючи точність до рівня користувача або складності завдання, що вимагає реалізації складних алгоритмів з елементами машинного навчання або гнучкого налаштування параметрів аналізу.

Загалом, оцінка ритмічної точності у цифровому середовищі — це завдання, що лежить на перетині технічної інженерії, акустики, психології сприйняття музики та педагогіки. Через це більшість існуючих рішень мають компромісний характер: вони або спрощують аналіз, або ж вимагають складної технічної реалізації з високими вимогами до апаратних ресурсів і програмної точності.

1.3.2. Поширені підходи до аналізу точності ритмічного виконання

Розробка алгоритмів аналізу ритмічної точності в музичних додатках передбачає обробку часових параметрів, отриманих від користувача під час виконання вправи, та порівняння їх з еталонними значеннями. У контексті цифрового навчання найважливішими критеріями є точність таймінгу, стабільність ритму та узгодженість виконання з метрономом або аудіосупроводом. Протягом останніх років сформувалося кілька основних підходів до реалізації таких алгоритмів.

Один із найпоширеніших способів оцінки ритмічної точності базується на порівнянні часових міток подій, які генерує користувач (наприклад, натискання клавіші, клік мишкою, удар по екрану), з таймінгом еталонних ритмічних одиниць. Це так званий метод порогового аналізу відхилення. Він полягає в тому, що кожна подія користувача оцінюється на відповідність до найближчої очікуваної події. Якщо відхилення у часі не перевищує допустимого порогу (наприклад, ± 100 мс), виконання вважається правильним. Цей підхід простий у реалізації, однак має ряд недоліків: зокрема, він не

враховує музичний контекст і не робить різниці між незначними та систематичними помилками.

Більш просунуті рішення використовують статистичний аналіз таймінгу. Наприклад, розраховуються середнє квадратичне відхилення, середнє абсолютне відхилення або коефіцієнт варіації між очікуваними і фактичними подіями. Такі метрики дозволяють отримати загальну оцінку рівномірності виконання, що є важливим у завданнях типу "зіграй рівно в темпі". Проте, з точки зору користувача, подібна статистика іноді виглядає надто абстрактно і не зовсім зрозумілою без візуалізації.

Окрему категорію становлять методи, засновані на інтервальному аналізі. У цьому випадку не порівнюється час кожної окремої події, а аналізуються інтервали між послідовними ударами або нотами. Таким чином, система визначає, наскільки стабільно користувач тримає заданий ритм, навіть якщо вся фраза трохи зміщена в часі. На мою думку, це один із найбільш природних способів аналізу, оскільки він дозволяє абстрагуватися від одиничних помилок і зосередитися на загальному відчутті ритму.

Сучасні системи також активно використовують методи машинного навчання. Зокрема, нейронні мережі тренуються на великому обсязі даних, щоб навчитися розрізняти "правильне" та "неправильне" виконання не лише за часовими характеристиками, а й за музичними параметрами — динамікою, артикуляцією, тощо. Проте такі рішення потребують значних обчислювальних ресурсів, наявності репрезентативної вибірки та професійної оцінки для формування правильних міток у процесі навчання.

Незалежно від обраного підходу, практично всі рішення зіштовхуються з однією загальною проблемою — необхідністю знаходити баланс між жорсткістю аналізу і лояльністю до початківців. Ідеальний алгоритм повинен бути адаптивним, враховувати рівень користувача, тип завдання, музичний стиль і навіть апаратне середовище, в якому виконується вправа. Цей баланс — одна з найбільших методичних і технічних викликів при створенні освітніх музичних інструментів.

1.3.3. Приклади імплементації у програмних рішеннях

Застосування алгоритмів аналізу ритмічної точності у програмних засобах — це вже не теоретична концепція, а цілком прикладна задача, яка успішно реалізується в низці сучасних музичних тренажерів. У цьому підпункті буде розглянуто, яким чином ті чи інші програми реалізують аналіз точності, які методи використовують на практиці, а також яким чином подається результат користувачеві.

Одним із яскравих прикладів реалізації алгоритмів ритмічного аналізу є програма Yousician. Цей додаток здійснює аналіз гри користувача за допомогою мікрофона або MIDI-інтерфейсу. Звук, що надходить до системи, проходить спектральний аналіз, визначається висота та момент початку кожної ноти. Система порівнює таймінг реального виконання з еталонним і виводить результат у формі візуального зворотного зв'язку: зелений — правильно, жовтий — близько до правильного, червоний — помилка. Цей підхід поєднує простоту сприйняття для користувача із достатнім рівнем точності для навчальних цілей.

Інший приклад — Melodics, програмне забезпечення для гри на MIDI-клавіатурах та ударних пад-контролерах. Тут аналіз точності ґрунтується на синхронізації MIDI-подій з еталонною сіткою. Програма фіксує затримки або передчасні удари та виводить детальний звіт по кожному такту. Такий підхід дуже корисний для поглибленого навчання, однак може бути занадто технічним для новачків. На мою думку, важливо, щоб інтерфейс таких систем дозволяв адаптувати рівень деталізації зворотного зв'язку до потреб конкретного учня.

Ще один цікавий приклад — Rhythm Sight Reading Trainer, який має дещо іншу концепцію: замість реального виконання він аналізує точність візуального читання ритму. Користувач натискає клавішу або торкається екрана згідно з представленим ритмом, а система фіксує кожен удар і порівнює з очікуваним. У цьому випадку акцент зроблено на зорову координацію та

розвиток відчуття часу без необхідності повноцінної гри на інструменті. Така методика є корисною для ранніх етапів музичного навчання.

Слід також згадати деякі прикладні проєкти відкритого коду, що реалізують аналіз точності з використанням мов програмування. Наприклад, у середовищі Pure Data або Max/MSP створюються патчі, які вимірюють інтервали між натисканнями користувача, будують гістограми ритмічних відхилень, обчислюють середнє відхилення або навіть подають зворотний зв'язок у реальному часі. Ці рішення не є масовими, проте мають велике значення для експериментальної музичної педагогіки.

На мою думку, з огляду на приклади, що були розглянуті, очевидно, що більшість програм орієнтуються на візуальну подачу результату — за допомогою кольорів, графіків, часових шкал. Це сприяє кращому розумінню своїх помилок, що є надзвичайно важливим у навчальному процесі. Однак у більшості випадків глибина аналізу обмежується досить простими метриками, і лише незначна частка систем використовує адаптивний або інтелектуальний аналіз.

Отже, впровадження алгоритмів ритмічного аналізу у програмні рішення вже стало поширеною практикою, однак їх точність, адаптивність і педагогічна цінність все ще залишаються напрямками для подальших досліджень і вдосконалень.

1.4. Висновки до розділу

У результаті проведеного огляду літературних джерел та аналізу сучасного стану розробок у сфері музичних ритм-тренажерів можна зробити низку важливих висновків, що підкріплюють актуальність та доцільність обраного напряму дослідження.

По-перше, було встановлено, що впродовж останніх десятиліть цифрові технології суттєво трансформували підходи до навчання музиці, відкривши нові можливості як для інституційної освіти, так і для самостійного засвоєння музичних навичок. Зокрема, зростаюча популярність онлайн-платформ,

мобільних додатків і спеціалізованого ПЗ сприяє активному впровадженню інноваційних методик викладання, які поєднують в собі інтерактивність, адаптивність і персоналізацію навчального процесу.

По-друге, було виявлено, що ринок музичних освітніх продуктів активно розвивається, проте існуючі рішення мають як сильні сторони, так і певні обмеження. Комерційні платформи пропонують потужні інструменти для навчання, однак не завжди враховують індивідуальні потреби користувача або мають обмежену аналітичну складову. Академічні розробки, своєю чергою, часто є експериментальними та не адаптованими для масового використання, хоча й відкривають нові горизонти в дослідженні методик навчання.

По-третє, було доведено, що аналіз точності ритмічного виконання є одним із найменш опрацьованих аспектів у музичних тренажерах. Попри наявність спроб реалізації таких алгоритмів у деяких додатках, більшість з них використовують досить примітивні методи оцінки, не враховуючи індивідуальні особливості темпового відчуття чи адаптацію до прогресу користувача. На мою думку, ця проблема є ключовою, оскільки ритм — це фундаментальна складова музичного виконання, а точність — критичний чинник його якісної реалізації.

На основі проведеного аналізу можна аргументовано стверджувати, що існує об'єктивна потреба у створенні нового інструменту — ритм-тренажера, який поєднає зручність використання, наочність та гнучкі алгоритми аналізу точності. Такий підхід дозволить не лише покращити ефективність навчального процесу, а й надати користувачу детальний зворотний зв'язок щодо його ритмічного виконання.

Таким чином, обрана тема дипломної роботи є актуальною як з наукової, так і з практичної точки зору. Її реалізація має потенціал покращити якість музичної освіти, особливо в умовах зростаючого попиту на дистанційні та самостійні форми навчання. Крім того, подальше удосконалення алгоритмів аналізу точності та впровадження адаптивних підходів сприятиме

персоналізації освітніх технологій, що відповідає сучасним тенденціям у галузі.

2. РОЗРОБКА МУЗИЧНОГО РИТМ-ТРЕНАЖЕРА З АЛГОРИТМАМИ АНАЛІЗУ ТОЧНОСТІ

2.1. Постановка задачі та вимоги до системи

Перш ніж розпочати безпосередню реалізацію програмного забезпечення, необхідно чітко визначити проблематику, яку воно має вирішувати, а також сформулювати вимоги до функціональності, надійності та зручності використання системи. Постановка задачі — це важливий етап проєктування, що дозволяє окреслити загальну концепцію майбутнього застосунку, його мету та способи досягнення поставленої мети.

На мою думку, саме коректно сформульована задача дозволяє не лише уникнути зайвого функціонального навантаження, але й забезпечити відповідність системи потребам кінцевих користувачів. Це особливо актуально в умовах розробки освітніх застосунків, де взаємодія з користувачем має бути інтуїтивно зрозумілою, а результати — легко оцінюваними.

Крім постановки задачі, важливо сформувати низку функціональних та нефункціональних вимог до системи. Вони слугують орієнтиром під час розробки й оцінки якості реалізованого рішення. У цьому підрозділі буде наведено:

- формулювання основної мети та задач дослідження;
- вимоги до функціональності системи;
- вимоги до інтерфейсу користувача;
- обмеження та умови функціонування;
- очікувані результати та критерії їх оцінки.

2.1.1. Формулювання основної мети та задач

На сучасному етапі розвитку цифрових технологій особливу актуальність набуває розробка програмних засобів, орієнтованих на

підвищення якості освітнього процесу. У сфері музичної освіти все частіше використовуються спеціалізовані програмні продукти, які дозволяють автоматизувати або значною мірою спростити процес навчання. У цьому контексті особливий інтерес становлять ритм-тренажери — програмні інструменти, що дають змогу користувачу тренувати відчуття ритму, координацію та точність виконання ритмічних фраз.

Основною метою даної роботи є розробка інтерактивного веб-застосунку — музичного ритм-тренажера, що дозволяє не лише створювати й проходити ритмічні вправи, а й аналізувати точність їх виконання за певними алгоритмами. На мою думку, ключовим моментом у досягненні цієї мети є створення такого інтерфейсу, який би поєднував простоту використання з гнучкістю функціоналу, а також забезпечував якісний зворотний зв'язок для користувача у вигляді точних статистичних показників.

З метою досягнення поставленої мети в рамках роботи були сформульовані наступні основні задачі:

- Провести аналіз предметної області з урахуванням наявних програмних рішень у сфері музичного навчання та ритм-тренажерів.
- Визначити функціональні вимоги до ритм-тренажера, виходячи з потреб потенційних користувачів, серед яких можуть бути як новачки, так і музиканти з базовими знаннями.
- Розробити архітектуру програмного продукту, що передбачає модульну побудову з можливістю подальшого масштабування функціоналу.
- Реалізувати основні режими роботи тренажера — зокрема, «фортепіанний» режим із нотами різної висоти та «перкусійний» режим з одним типом удару.
- Створити механізм генерації вправ різного рівня складності, який враховуватиме як довжину вправи, так і її ритмічну наповненість.

- Імплементувати алгоритми оцінки точності виконання, які б аналізували відхилення у ритміці (таймінгу) від еталонного зразка.
- Розробити систему статистики та аналізу результатів, яка б дозволяла користувачеві оцінювати власний прогрес у динаміці.
- Забезпечити інтуїтивно зрозумілий інтерфейс, придатний для взаємодії як на настільних ПК, так і на мобільних пристроях.
- Здійснити тестування працездатності розробленого програмного забезпечення та провести аналіз його ефективності у навчальному процесі.

Таким чином, сформульовані задачі охоплюють повний цикл розробки: від теоретичного аналізу до реалізації і тестування кінцевого продукту. Їх поетапне виконання дозволить досягти головної мети дипломної роботи — створення зручного, функціонального та ефективного засобу для тренування музичного ритму з можливістю аналізу точності.

2.1.2. Опис загальних функціональних та нефункціональних вимог

На етапі формування концепції програмного ритм-тренажера вкрай важливим завданням є чітке визначення вимог до системи, що забезпечить узгодженість подальших етапів проєктування, реалізації та тестування. Вимоги до програмного забезпечення поділяються на функціональні та нефункціональні. Такий поділ дозволяє не лише формалізувати очікування щодо поведінки системи, а й упорядкувати її архітектуру, структуру користувацького інтерфейсу, а також принципи взаємодії з кінцевим користувачем.

Функціональні вимоги визначають, які саме дії повинна виконувати система, тобто її основну поведінку. У контексті музичного ритм-тренажера, на мою думку, серед першочергових функціональних вимог варто виділити можливість генерації ритмічних прикладів із заданими характеристиками (довжина, складність, режим). Важливою є й функція озвучення вправи, тобто звукове відтворення прикладу для зразка, який користувач повинен повторити.

Окрім того, система має забезпечити інструмент для введення відповіді користувачем — як у формі натискання клавіш у заданому ритмі, так і, потенційно, через інші інтерактивні методи введення. Не менш важливою є реалізація механізму порівняння виконаного ритму з еталонним, а також подальшого аналізу точності виконання, який враховує не лише правильність нот, а й час натискання, ритмічну послідовність і динамічні характеристики.

Окремою складовою функціональних вимог є реалізація зворотного зв'язку — система повинна демонструвати користувачу результат виконання у зручній формі: текстове повідомлення, оцінка у відсотках, графічне відображення помилок або відхилень тощо. Важливою є й наявність можливості повторного виконання вправи для закріплення навичок, а також ведення базової статистики, яка б дозволила користувачу простежити власний прогрес за певний період.

Нефункціональні вимоги, у свою чергу, охоплюють характеристики системи, які не пов'язані безпосередньо з її функціональністю, проте визначають її якість, ефективність та зручність експлуатації. Насамперед, це продуктивність системи, яка повинна забезпечувати обробку взаємодії користувача в реальному часі, з мінімальними затримками, особливо в частині аналізу таймінгу. Іншими словами, відгук системи на дії користувача має бути максимально наближеним до миттєвого, щоб не спотворювати відчуття ритму та не ускладнювати навчальний процес.

Серед інших важливих нефункціональних вимог можна виокремити кросплатформеність, тобто здатність працювати у більшості сучасних браузерів та на різних типах пристроїв — від настільних ПК до мобільних телефонів. Також важливо забезпечити високий рівень доступності інтерфейсу: навіть користувач без музичної чи технічної підготовки повинен мати змогу інтуїтивно розібратися у функціоналі системи. Не останнє місце займає питання масштабованості: архітектура програми повинна дозволяти подальше розширення функціональності, наприклад, додавання нових типів вправ, режимів навчання або методів аналізу.

Окремої уваги заслуговує надійність системи та її стійкість до некоректних або непередбачених дій користувача. Програма не повинна аварійно завершувати роботу або втрачати дані при введенні помилкової інформації. І хоча ритм-тренажер не оперує критично важливою або конфіденційною інформацією, базові принципи безпечного зберігання статистики та запобігання несанкціонованому доступу все ж мають бути реалізовані.

Узагальнюючи викладене, слід зазначити, що саме на цьому етапі — етапі формування вимог — закладається основа для всієї подальшої розробки. Від точності й повноти формулювання вимог залежить як успішність реалізації, так і кінцева ефективність програмного продукту. Саме тому я вважаю цей етап одним із найбільш відповідальних у всьому процесі створення програмного ритм-тренажера.

2.1.3. Визначення цільової аудиторії

У процесі проектування будь-якого програмного забезпечення надзвичайно важливо чітко визначити, для кого саме створюється система, тобто окреслити цільову аудиторію. Це дозволяє краще зрозуміти очікування потенційних користувачів, їх рівень підготовки, потреби та технічні можливості. Правильне визначення аудиторії є основою для побудови як логіки інтерфейсу, так і вибору функціонального наповнення. На мою думку, ігнорування цього аспекту часто призводить до створення надто складних або, навпаки, надто примітивних систем, що не відповідають реальним потребам користувачів.

Ритм-тренажер, що розробляється в межах цієї кваліфікаційної роботи, передбачає широкий спектр користувачів, які можуть суттєво відрізнятися за віком, рівнем музичної підготовки, цілями використання продукту та технічним досвідом. Тому для коректного урахування їхніх потреб було доцільно класифікувати основні типи потенційних користувачів. На підставі аналізу ринку освітніх додатків, спостережень за існуючими платформами, а

також узагальнення особистого досвіду використання подібних інструментів, сформовано умовний поділ цільової аудиторії на кілька категорій.

До першої категорії відносяться новачки без музичної підготовки. Це особи, які тільки починають своє знайомство з музикою й хочуть розвинути базове відчуття ритму без академічного підходу. Для цієї групи необхідно забезпечити максимально простий та інтуїтивно зрозумілий інтерфейс, мінімум складних термінів і чіткий візуальний зворотний зв'язок.

Наступну категорію складають студенти музичних шкіл або аматори, які вже мають певну теоретичну й практичну базу. Їх очікування від системи вищі: вони зацікавлені в точнішому аналізі виконання, можливості самостійно варіювати складність вправ, зберігати історію виконань, повторювати вправи з метою вдосконалення. Це, на мою думку, одна з найширших груп користувачів.

До окремої категорії слід віднести педагогів та викладачів, які можуть використовувати тренажер як допоміжний інструмент для навчання учнів. У такому разі важливо передбачити можливість генерації вправ, які відповідають навчальним програмам, а також прості способи перевірки результатів учнів.

Окремо варто згадати про користувачів з особливими потребами, зокрема осіб з порушеннями слуху або моторики. Хоча розробка спеціалізованого функціоналу для цієї групи не є пріоритетною у рамках цієї кваліфікаційної роботи, її наявність у структурі користувачів дозволяє зважати на принципи інклюзивності під час побудови інтерфейсу та організації зворотного зв'язку.

Для зручного узагальнення наведених категорій цільової аудиторії доцільно представити їх у вигляді таблиці (див. Таблицю 2.1), що містить основні характеристики кожної групи користувачів.

Таблиця 2.1 - Основні категорії цільової аудиторії ритм-тренажера

Категорія користувачів	Рівень підготовки	Основні потреби	Особливості інтерфейсу
Новачки без підготовки	Початковий	Просте навчання ритму, інтерактивність	Простота, візуальні підказки
Студенти, аматори	Середній	Тренування ритму, аналіз точності, повторення	Збалансований функціонал
Викладачі	Високий	Генерація вправ, оцінювання учнів	Гнучке налаштування, аналітика
Користувачі з особливими потребами	Змінний	Інклюзивність, зручність, доступність	Контраст, доступна навігація

2.2. Архітектура системи

Під час розробки будь-якого програмного забезпечення вкрай важливою складовою є визначення загальної архітектури системи. Саме на цьому етапі приймаються рішення, що визначають логічну структуру взаємодії основних компонентів, розподіл обов'язків між модулями, способи комунікації між підсистемами, а також технічні особливості реалізації ключових функцій. Як показує практика програмної інженерії, саме від вдало побудованої архітектурної моделі часто залежить подальша масштабованість, підтримуваність і навіть працездатність усього проєкту в довгостроковій перспективі.

У випадку створення ритм-тренажера, побудова архітектури відіграє особливу роль з огляду на специфіку самого інструмента. Система повинна не тільки забезпечувати інтерактивну взаємодію з користувачем у реальному часі, а й ефективно обробляти вхідні дії, здійснювати точний аналіз таймінгів, формувати візуальний зворотний зв'язок і зберігати статистику виконання

вправ. Усе це вимагає не лише логічної побудови інтерфейсної частини, але й ефективної організації бекенд-механізмів. Саме тому при розробці було прийнято рішення про розділення логіки на фронтенд і бекенд з чітко визначеними зонами відповідальності.

Окрім того, однією з ключових вимог було забезпечення можливості майбутнього розширення системи — зокрема, додавання нових режимів, інтеграція з іншими освітніми платформами, підключення баз користувачів або персоналізованих профілів. Саме тому при побудові архітектурної моделі застосовувався модульний підхід, який, на мою думку, є одним із найдоцільніших у контексті сучасної розробки веб-застосунків.

Також важливо зазначити, що вибір технологій і логіки взаємодії між компонентами був тісно пов'язаний із вимогами до продуктивності та простоти підтримки коду. Було враховано, що система має ефективно працювати у браузері без потреби в складному інсталяційному процесі, бути кросплатформенною та дружньою до користувача з мінімальним технічним досвідом. Усе це, у підсумку, визначило загальні контури архітектури, які будуть розглянуті в наступних пунктах цього підрозділу.

2.2.1. Загальна структура системи та модулів

Побудова ритм-тренажера вимагала чіткого структурування програмного забезпечення з урахуванням як функціональних, так і нефункціональних вимог. Відповідно до принципів модульного проєктування, архітектура системи була поділена на логічні компоненти, кожен із яких відповідає за окремий аспект функціонування. Такий підхід дозволяє досягти високого рівня ізоляції, забезпечити легкість тестування, масштабованість та спрощення подальшої підтримки.

На верхньому рівні систему умовно можна поділити на дві основні частини — клієнтську (frontend) та серверну (backend). Кожна з них, у свою чергу, містить низку функціональних модулів, які взаємодіють між собою через стандартизовані інтерфейси.

Клієнтська частина виконує роль інтерфейсу користувача. Вона відповідає за візуалізацію нот, обробку взаємодії з користувачем (натискання клавіш, керування вправами), відображення результатів та статистики. До її складу входять такі основні модулі:

- Модуль генерації прикладу — формує послідовність ритмічних нот згідно з обраними параметрами;
- Модуль відтворення — забезпечує синхронізацію звуку з візуальним ритмом;
- Модуль введення відповіді — фіксує дії користувача у реальному часі;
- Інтерфейсний модуль — відповідає за зовнішнє оформлення, зручність взаємодії та UX.

Серверна частина (СЧ), яка умовно включає в себе логіку перевірки результатів, розрахунок точності, збереження статистики (у випадку повноцінного розширення), виконує роль обчислювального ядра. У спрощеній локальній реалізації частину цієї логіки реалізовано безпосередньо у клієнті, проте з погляду архітектури передбачено можливість відокремлення цієї логіки у backend, якщо система масштабуватиметься.

Взаємодія між модулями забезпечується внутрішніми API-викликами. У випадку реалізації повноцінного серверного компонента взаємодія відбуватиметься за протоколом HTTP із використанням REST-архітектури. Це дозволяє за потреби перенести окремі частини логіки на інші сервери або навіть реалізувати мультикористувацький режим.

Щоб краще проілюструвати архітектурну побудову системи, нижче наведено блок-схему, що відображає загальну структуру компонентів та напрямки їх взаємодії.

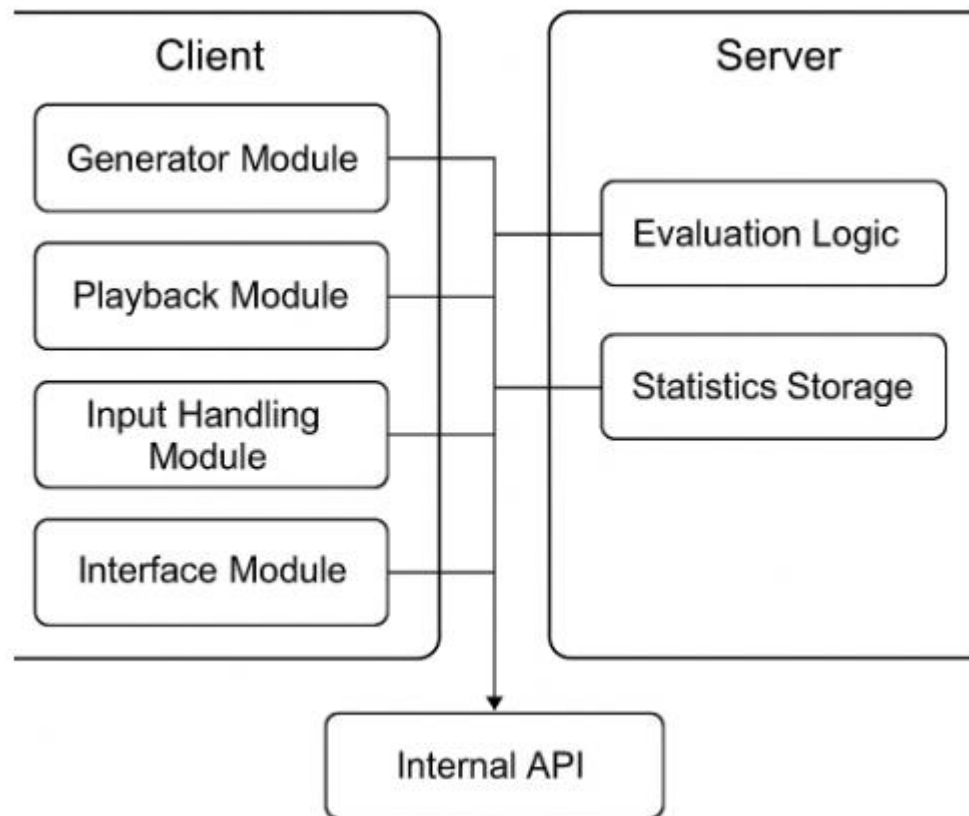


Рисунок 2.1 – Блок-схема архітектури системи

2.2.2. Взаємодія між компонентами фронтенду та бекенду

У будь-якій сучасній веб-орієнтованій системі, особливо тій, що спрямована на взаємодію з кінцевим користувачем у режимі реального часу, важливу роль відіграє чітко налагоджена комунікація між двома ключовими частинами: фронтом (інтерфейсом користувача) та бекендом (серверною логікою). Саме ця взаємодія є критично важливою для коректного, надійного і безперебійного функціонування системи в цілому. Вона забезпечує передачу даних, реакцію на дії користувача, обробку логіки тренажера, збереження статистики, перевірку правильності ритму та інші важливі аспекти роботи додатку.

З архітектурної точки зору, фронтенд виступає як “обличчя” системи, з яким безпосередньо взаємодіє користувач. У рамках даного ритм-тренажера фронтенд реалізує основні інтерфейсні компоненти: відображення нотних прикладів, введення відповіді користувача, виведення зворотного зв’язку,

статистики та іншої візуальної інформації. Водночас фронтенд є лише “клієнтом”, який потребує доступу до функціональностей, реалізованих на стороні сервера, або, інакше кажучи, у бекенді.

Бекенд, у свою чергу, виконує функції “мозкового центру” системи. Саме він відповідає за генерацію ритмічних прикладів, збереження та обробку даних, обчислення точності виконання та логіку перевірки правильності ритму. Також на стороні сервера відбувається облік статистики, формування звітів та зберігання історії взаємодії з користувачем. Таким чином, фронтенд та бекенд функціонують у тісній симбіозі, доповнюючи одне одного.

Для реалізації ефективної взаємодії між цими частинами системи було обрано модель взаємодії клієнт-сервер, що є найбільш розповсюдженою у веб-розробці. Комунікація відбувається за допомогою HTTP-запитів — здебільшого типу POST (для передачі даних, наприклад, при надсиланні ритмічної відповіді користувача на перевірку) та GET (для отримання даних — наприклад, нової вправи або статистики). Дані між фронтендом і бекендом передаються у форматі JSON, що дозволяє забезпечити легку серіалізацію та десеріалізацію об’єктів, підтримку кросплатформенності та простоту у налагодженні.

Варто відзначити, що особливу увагу було приділено асинхронній взаємодії між компонентами. Це дозволяє уникнути зависань або затримок у роботі інтерфейсу при обробці складних логічних операцій на сервері. Для цього на фронтенді застосовується механізм `fetch` з обробкою `Promise`, що дозволяє виконувати запити до бекенду у фоновому режимі та оновлювати інтерфейс лише після отримання відповіді.

З точки зору захищеності та стійкості системи, реалізовано базову обробку помилок: при виникненні проблем із сервером або мережею користувачу виводиться повідомлення з інструкцією щодо подальших дій. Це дозволяє підвищити надійність роботи навіть у нестабільних умовах підключення.

На мою думку, ефективна взаємодія між фронтендом і бекендом — це не лише технічне завдання, а й важливий елемент, що безпосередньо впливає на якість користувацького досвіду. Якщо цей міст між двома частинами системи функціонує безперешкодно, користувач сприймає тренажер як зручний, швидкий та інтуїтивно зрозумілий інструмент, що мотивує до подальшого використання.

2.3. Вибір технологій та інструментів розробки

У процесі розробки будь-якого сучасного програмного забезпечення, а особливо інтерактивних веб-додатків із елементами мультимедійної взаємодії, одним із найважливіших етапів є обґрунтований вибір технологічного стеку. Від правильності цього вибору значною мірою залежить не лише продуктивність і масштабованість програмного продукту, але й зручність розробки, підтримки та потенціал для подальшого розвитку системи.

Ритм-тренажер, як програмний засіб для навчання музичним навичкам, потребує ефективної реалізації одночасно кількох критично важливих складових. Серед них можна виділити обробку звуку в реальному часі, візуалізацію нотних прикладів, збирання точних таймінгів дій користувача, а також гнучкий інтерфейс із привабливим дизайном. Усе це, у свою чергу, висуває особливі вимоги до обраних мов програмування, бібліотек, фреймворків, а також серверної логіки.

При виборі інструментів для реалізації тренажера слід було враховувати низку факторів, серед яких: відкритість і доступність технологій (зокрема безкоштовність використання), активна спільнота розробників, наявність документації, легкість інтеграції між фронтендом і бекендом, можливість обробки подій у реальному часі, а також адаптивність інтерфейсу під різні пристрої.

На мою думку, зважаючи на сучасні тенденції у сфері веб-розробки, найбільш доцільним підходом є використання добре підтримуваних і перевірених часом рішень, які вже мають успішні приклади застосування в

суміжних проєктах. Саме тому вибір технологій у даному проєкті базується не лише на технічній доцільності, а й на досягненні оптимального балансу між швидкістю розробки, функціональністю та стабільністю.

У цьому підрозділі буде детально проаналізовано обрані мови програмування, бібліотеки та фреймворки, а також наведено причини, що зумовили їх застосування в рамках даної кваліфікаційної роботи. Крім того, буде запропоновано коротке порівняння з альтернативними варіантами, що дозволить обґрунтувати остаточне технічне рішення.

2.3.1. Мова програмування, бібліотеки, фреймворки

Для розробки ритм-тренажера було обрано веб-орієнтований підхід, що передбачає реалізацію інтерфейсу у вигляді клієнтської частини (frontend) із застосуванням HTML, CSS та JavaScript, а також серверної частини (backend), яка за потреби може бути доповнена функціоналом для зберігання даних, обробки логіки, аналітики тощо.

Основною мовою клієнтської частини стала JavaScript, що зумовлено її універсальністю, широкою підтримкою браузерами, а також можливістю гнучкої інтерактивної взаємодії з користувачем у режимі реального часу. JavaScript є де-факто стандартом для побудови веб-інтерфейсів, а завдяки сучасним можливостям, таким як підтримка об'єктно-орієнтованого підходу, асинхронних викликів (async/await), взаємодії з API, — дозволяє реалізувати функціонал, необхідний для роботи ритм-тренажера.

Для стилізації інтерфейсу використано CSS (Cascading Style Sheets) у поєднанні з базовими можливостями Flexbox та Grid, що забезпечило адаптивну верстку інтерфейсу для різних типів екранів. На мою думку, саме простота та прозорість CSS дозволили уникнути надмірного перевантаження стилів, зосередивши увагу на мінімалізмі та зручності взаємодії з тренажером.

Для обробки аудіо у веб-браузері було використано Web Audio API — стандарт, підтримуваний сучасними браузерами, який забезпечує гнучке керування аудіо-потокami, можливість програвання нот із певною частотою,

накладання ефектів, затримок, тощо. Це дозволяє реалізувати базову аудіосинтезацію для перкусійного та фортепіанного режимів без потреби у складних зовнішніх залежностях.

Відтворення музичних інструментів реалізовано з використанням бібліотеки WebAudioFont, яка забезпечує симульоване звучання MIDI-інструментів у браузері. Її перевагою є можливість легко завантажувати різні інструменти, використовуючи доступні soundfont-файли, і керувати висотою та тривалістю звуку програмно. Це значно спрощує створення багатоголосся та подальшу синхронізацію із нотним записом.

Хоча в рамках цієї кваліфікаційної роботи не було потреби в реалізації складного бекенду, варто зазначити, що у разі розширення функціоналу (наприклад, введення акаунтів користувачів або збереження результатів), доречним було б використання Node.js — як легкого серверного середовища, яке дозволяє писати backend тим же JavaScript. Це дозволяє досягнути єдності мовного середовища між клієнтом і сервером, що в перспективі значно спрощує підтримку коду.

Отже, обраний стек технологій є доцільним і виправданим для реалізації проєкту такого типу, оскільки забезпечує баланс між функціональністю, продуктивністю та простотою реалізації. До того ж, уся використана база інструментів є відкритою, активно підтримується спільнотою і не потребує додаткових ліцензій, що важливо для академічної та некомерційної розробки.

2.3.2. Причини вибору технологічного стеку

Під час початкового етапу планування розробки музичного ритм-тренажера особливу увагу було приділено вибору технологічного стеку. Це рішення визначає не лише архітектуру та принципи побудови системи, а й зручність подальшого масштабування, стабільність, швидкодію та простоту підтримки проєкту. Відповідно, вибір технологій не був випадковим, а ґрунтувався на аналізі кількох важливих факторів, які, на мою думку, є ключовими в контексті академічного програмного продукту.

Перш за все, основною метою розробки було створення простого у використанні та доступного тренажера, який не потребує інсталяції, складних налаштувань або специфічних технічних знань з боку користувача. У цьому контексті веб-платформа є найбільш універсальним рішенням, адже доступ до неї здійснюється безпосередньо через браузер, що значно спрощує розповсюдження та використання тренажера.

Вибір HTML, CSS та JavaScript як основних технологій фронтенду був зумовлений їхньою стабільністю, широким розповсюдженням, відкритістю та наявністю величезної кількості документації. На сучасному етапі розвитку інтернет-технологій ці інструменти стали незамінними при реалізації інтерактивних інтерфейсів. Крім того, для студентської роботи велике значення має прозорість та зрозумілість коду, а класичний JavaScript дозволяє реалізувати навіть складну логіку без використання важких фреймворків.

Також варто зазначити, що бібліотека WebAudioFont стала оптимальним вибором для реалізації музичного супроводу. Її використання забезпечило гнучке керування відтворенням нот і дало можливість імітувати звуки справжніх музичних інструментів. Важливо, що ця бібліотека підтримує легке підключення звуків MIDI-інструментів і сумісна з Web Audio API — стандартом для роботи з аудіо в браузерах.

Ще однією перевагою вибраного стеку є легкість адаптації під різні типи користувачів і пристроїв. Завдяки можливостям CSS (зокрема, Flexbox і Grid), тренажер було реалізовано таким чином, щоб він коректно відображався на різних розмірах екранів, що особливо важливо для сучасних користувачів мобільних пристроїв та планшетів. Таким чином, забезпечено кросплатформність без залучення додаткових фреймворків чи бібліотек.

Не менш важливою причиною такого вибору є відсутність складної серверної інфраструктури, що дозволило значно спростити розгортання системи та уникнути витрат на хостинг або підтримку баз даних. Це особливо актуально для студентських проєктів, які не мають комерційної складової, але потребують демонстрації реального, працездатного продукту.

На мою думку, саме така комбінація простоти, функціональності та відкритості інструментів забезпечила ефективну реалізацію поставлених цілей. Обрані технології дозволяють не лише реалізувати заявлений функціонал, а й у майбутньому при бажанні доповнювати або розширювати систему без суттєвих переробок.

2.3.3. Порівняльна характеристика альтернативних технологій

Під час вибору стеку технологій для реалізації музичного ритм-тренажера було проаналізовано низку альтернативних варіантів, кожен з яких має як свої переваги, так і недоліки. Такий аналіз дозволяє не лише краще обґрунтувати прийняті технічні рішення, а й сформувати базу для потенційного подальшого розвитку або масштабування проєкту.

У таблиці 2.2 наведено коротке порівняння декількох популярних технологічних рішень, які могли бути застосовані під час розробки як фронтенд-, так і бекенд-частини проєкту. Аналіз проведено за кількома критеріями: складність використання, швидкість розробки, продуктивність, підтримка, можливості інтеграції, вимоги до серверної частини тощо.

Таблиця 2.2 - Порівняння альтернативних технологій для реалізації ритм-тренажера

Технологія	Складність освоєння	Швидкість розробки	Підтримка спільнотою	Продуктивність	Потреба у сервері	Коментар
HTML/CSS/JS	Низька	Висока	Висока	Середня	Ні	Ідеально для швидкого створення веб-прототипу
React.js	Середня	Середня	Висока	Висока	Ні	Добре підходить для складніших SPA, але потребує більше ресурсів

Закінчення таблиці 2.2

Electron	Висока	Низька	Середня	Висока	Ні	Орієнтований на настільні застосунки; ресурсозатратний
Python + Flask	Середня	Середня	Висока	Висока	Так	Потужна серверна частина, але ускладнює розгортання
Unity + C#	Висока	Низька	Висока	Висока	Можливо	Підходить для складної графіки та 3D, але надлишковий для простої веб-програми

Як видно з наведеної таблиці 2.1, найдоцільнішим рішенням для нашого випадку виявилось використання стандартного стеку HTML/CSS/JS. Це дозволяє зосередитися на функціональності та дизайні, не витрачаючи надмірних ресурсів на конфігурацію середовища розробки або серверного забезпечення.

Водночас, слід зазначити, що у разі розширення функціональності тренажера, можлива поступова інтеграція серверної частини, наприклад, через Python+Flask або Node.js.

2.4. Реалізація основних функцій тренажера

Після визначення архітектури системи та вибору необхідного технологічного стеку особливу увагу було зосереджено на реалізації основної функціональності, що забезпечує ключовий користувацький досвід. У випадку ритм-тренажера — це не просто візуальна або аудіоінтеракція, а цілісний освітній процес, побудований на послідовності дій: генерація ритмічного прикладу, його звукове відтворення, введення ритму користувачем, перевірка правильності та виведення результатів.

Функціональність ритм-тренажера повинна бути логічно структурованою, інтуїтивно зрозумілою та такою, що враховує потреби і можливості різних категорій користувачів. Реалізація кожного окремого кроку

в цьому процесі потребувала як програмної чіткості, так і зручності взаємодії. Саме тому функціональні модулі розроблялися з фокусом на їхню простоту, стабільність, адаптивність до змінної складності вправ, а також готовність до подальшого масштабування.

У наступних пунктах буде детально розглянуто реалізацію кожного з етапів, що формують єдиний освітній цикл роботи з тренажером.

2.4.1. Генерація ритмічного прикладу та відтворення звуку

Однією з базових функціональних складових музичного ритм-тренажера є генерація ритмічного прикладу, який користувач має прослухати та відтворити. Саме з цього компонента фактично починається будь-яка навчальна взаємодія з додатком, оскільки завдання користувача ґрунтується на точному сприйнятті й повторенні прослуханого ритму. Генерація прикладу, таким чином, виконує роль відправної точки в логіці усієї системи.

На перший погляд, створення ритмічного прикладу може здаватися простим процесом, однак, як показує практика, навіть такий етап потребує серйозного концептуального та технічного підходу. У нашій системі було прийнято рішення формувати ритмічні патерни динамічно — тобто генерувати їх у момент створення вправи, залежно від заданої довжини, рівня складності, та обраного режиму (фортепіанного або перкусійного). Це дозволяє не лише уникнути шаблонності, а й урізноманітнити вправи, зробивши процес тренування цікавим і гнучким.

З технічного боку, генерація ритму реалізована за допомогою випадкових алгоритмів, що дотримуються заздалегідь визначених правил ритмічного поділу. Наприклад, на рівні складності 1 генеруються прості ритми, які складаються лише з чвертей або восьмих нот, тоді як на рівні 3 вже можуть зустрічатися синкопи, триолі та паузи різної тривалості. Таким чином, складність визначає не лише кількість ударів у такті, а й ступінь ритмічної різноманітності.

Щодо відтворення звуку — воно реалізоване на основі бібліотеки WebAudioFont. Цей інструмент дозволяє зручно завантажувати та відтворювати MIDI-звуки в браузері без необхідності у сторонніх плагінах або надлишковому завантаженні ресурсів. Звуки кожної ноти (або удару — в перкусійному режимі) програватимуться у чітко визначені моменти часу, що дозволяє користувачу отримати точне уявлення про ритмічний малюнок прикладу.

Слід зазначити, що велика увага приділялася також синхронності звукового ряду з візуальним поданням нот. На мою думку, це дозволяє значно покращити сприйняття ритму, особливо для початківців. Крім того, реалізовано можливість повторного прослуховування прикладу, що дає змогу користувачу більш впевнено перейти до етапу відтворення.

У перспективі можливим є впровадження декількох стилів ритмічного супроводу або тембрових варіантів інструментів, що, на мою думку, додасть нових вимірів у навчальний процес. Однак вже на поточному етапі функція генерації та відтворення ритмічних прикладів забезпечує достатню варіативність, точність і зручність для повноцінного навчання ритму у форматі тренажера.

2.4.2. Введення відповіді користувачем

Після прослуховування згенерованого ритмічного прикладу користувачу надається можливість ввести власну відповідь, повторивши ритм, що прозвучав. Цей етап є ключовим з точки зору оцінки точності та засвоєння матеріалу, адже саме тут відбувається активне залучення користувача до процесу навчання, який передбачає не пасивне сприйняття, а цілеспрямоване відтворення почутого. Таким чином, введення ритму виконує роль активного контролю знань і навичок.

У розробленому тренажері реалізовано два основні способи введення відповіді: за допомогою клавіш клавіатури та через натискання інтерактивних віртуальних кнопок на екрані. Обидва ці варіанти дозволяють задовольнити

потреби різних типів користувачів — від тих, хто користується десктопною версією застосунку, до тих, хто взаємодіє з мобільним пристроєм або планшетом. Така гнучкість, на мою думку, є важливою умовою зручності та інклюзивності цифрового навчального інструменту.

Введення ритму здійснюється шляхом фіксації таймінгів натискань користувача — кожне натискання відповідає одній ноті чи ритмічному удару. Система аналізує часові мітки натискань у контексті раніше заданого прикладу, дозволяючи порівняти їх з еталонною ритмічною послідовністю. Таким чином, формується цифрова модель виконання, яка в подальшому аналізується спеціальними алгоритмами (див. пункт 2.5.2).

Візуально відповідь користувача відображається у вигляді нотних символів або ритмічних блоків, які з'являються в окремому полі під прикладом. Це надає додатковий зоровий зворотний зв'язок, який, на мою думку, допомагає користувачеві краще усвідомити власні дії, особливо у випадку повторного проходження тієї самої вправи. Введення здійснюється поетапно, в темпі, який задається самим користувачем, що дозволяє уникнути зайвої напруги і зробити процес навчання комфортнішим.

З технічної точки зору обробка введення реалізована з використанням JavaScript API, зокрема подій `keydown` та `click`, що забезпечує оперативну фіксацію взаємодії в реальному часі. Також здійснюється перевірка на тривалість і синхронізацію з попередньо визначеним таймінгом, що дозволяє уникати випадкових подвійних натискань або технічних похибок.

Окремо варто згадати, що в системі передбачено можливість очищення останнього введеного елемента, а також повного скидання відповіді. Це зроблено для підтримки багаторазового повторення і, відповідно, формування стійких ритмічних навичок. Як показує практика, надання користувачеві можливості самокорекції суттєво підвищує мотивацію та впевненість у власних діях.

Загалом, механізм введення відповіді в нашому тренажері поєднує технічну простоту реалізації з високим рівнем зручності користування, що є, на мою думку, важливою перевагою у сфері музичної цифрової освіти.

2.4.3. Перевірка правильності та точності ритму

Перевірка правильності та точності виконання ритмічного прикладу є центральним етапом роботи музичного ритм-тренажера, адже саме вона дозволяє оцінити рівень засвоєння користувачем заданого матеріалу. У даному контексті йдеться не лише про констатацію факту правильного чи неправильного натискання, а про комплексний аналіз часових характеристик кожного удару, зіставлення їх з еталоном та формування узагальненої оцінки точності.

На першому етапі здійснюється аналіз кількості ритмічних одиниць, що були введені користувачем. Якщо кількість елементів не збігається з кількістю у прикладі, система автоматично видає попередження, що аналіз може бути неточним. Цей етап є формальним, однак він дозволяє уникнути грубих помилок на рівні структури відповіді.

Далі проводиться детальний аналіз таймінгів — проміжків часу між кожним натисканням клавіші або елементом введення. Відповідно до заздалегідь збережених еталонних значень, кожен проміжок порівнюється з очікуваним. Якщо відхилення не перевищує допустимий поріг (який у нашій системі можна налаштовувати відповідно до складності завдання), удар вважається правильним з точки зору ритмічної точності. Інакше — позначається як помилка.

З технічної точки зору, для реалізації перевірки було створено спеціальний алгоритм, який працює з масивами міток часу. Кожен масив представляє послідовність таймінгів: один — з боку системи (еталон), інший — з боку користувача. Після нормалізації тривалостей обчислюється середнє відхилення та будується умовний відсоток точності виконання. Такий підхід,

на мою думку, дозволяє об'єктивно оцінювати як загальний рівень виконання, так і окремі ритмічні помилки.

Щоб забезпечити зворотний зв'язок, результати перевірки відображаються на екрані. Користувач бачить загальний відсоток точності, кольорове маркування нот (наприклад, червоне — помилка, зелене — правильне натискання), а також текстове повідомлення, яке пояснює результат. Це дозволяє не тільки зафіксувати результат, але й зробити навчання більш усвідомленим, адже користувач бачить, де саме і в чому допустив помилку.

Крім цього, реалізовано систему накопичення статистики: середній рівень точності, кількість спроб, кількість правильних вправ тощо. Ці дані не тільки створюють додаткову мотивацію, а й дають змогу відстежити динаміку прогресу у вивченні ритму.

Таким чином, система перевірки ритмічного виконання виконує не просто функцію контролю, а є повноцінним компонентом навчального процесу. Вона інтегрує в собі аналіз точності, наочне відображення помилок та формування конструктивного зворотного зв'язку, що, на мою думку, є ознакою якісного сучасного освітнього інструменту.

2.4.4. Реалізація двох режимів: “фортепіано” та “перкусія”

Однією з ключових особливостей створеного ритм-тренажера є можливість вибору між двома режимами роботи — “фортепіано” та “перкусія”. Це рішення було прийнято з метою забезпечення ширшої адаптивності програмного продукту до потреб різних категорій користувачів. На мою думку, така варіативність дозволяє не лише розширити аудиторію, але й запропонувати гнучкі сценарії навчання з урахуванням різного рівня музичної підготовки.

Режим “фортепіано” реалізований як традиційна нотаційна система, у якій кожен ритмічний елемент прив'язаний до певної висоти звуку (C4, D4, E4 тощо). В інтерфейсі програми користувач бачить ритмічний приклад, поданий

у вигляді нот на нотному стані. Для кожної ноти використано унікальне зображення, що відображає не лише ритм, але й висоту. Введення відповіді в цьому режимі передбачає натискання відповідної клавіші на віртуальній клавіатурі або фізичній клавіші на клавіатурі комп'ютера, яка відповідає конкретній ноті. Це дозволяє реалізувати більш “музикальне” тренування — з одночасним відпрацюванням як ритму, так і інтонування.

Режим “перкусія”, у свою чергу, є спрощеним варіантом, орієнтованим виключно на ритмічний компонент. Тут не враховується висота звуку — лише часові характеристики. У цьому режимі користувач натискає одну й ту саму клавішу для кожного ритмічного удару, а система аналізує лише точність таймінгів.

У технічному сенсі перемикання між режимами реалізовано через окремий селектор у головному інтерфейсі програми. Залежно від обраного значення, тренажер завантажує відповідний набір візуальних і аудіо-асетів, а також активує специфічну логіку перевірки відповідей. Це дозволяє не лише кастомізувати навчальний процес, але й уникнути дублювання коду — архітектурне рішення передбачає повторне використання більшості базових компонентів з мінімальними модифікаціями. Для кращої ілюстрації роботи обох режимів, у тексті роботи наведено скріншоти інтерфейсу користувача (рисунок 2.4.1 та рисунок 2.4.2), які демонструють візуальні та функціональні відмінності між ними.

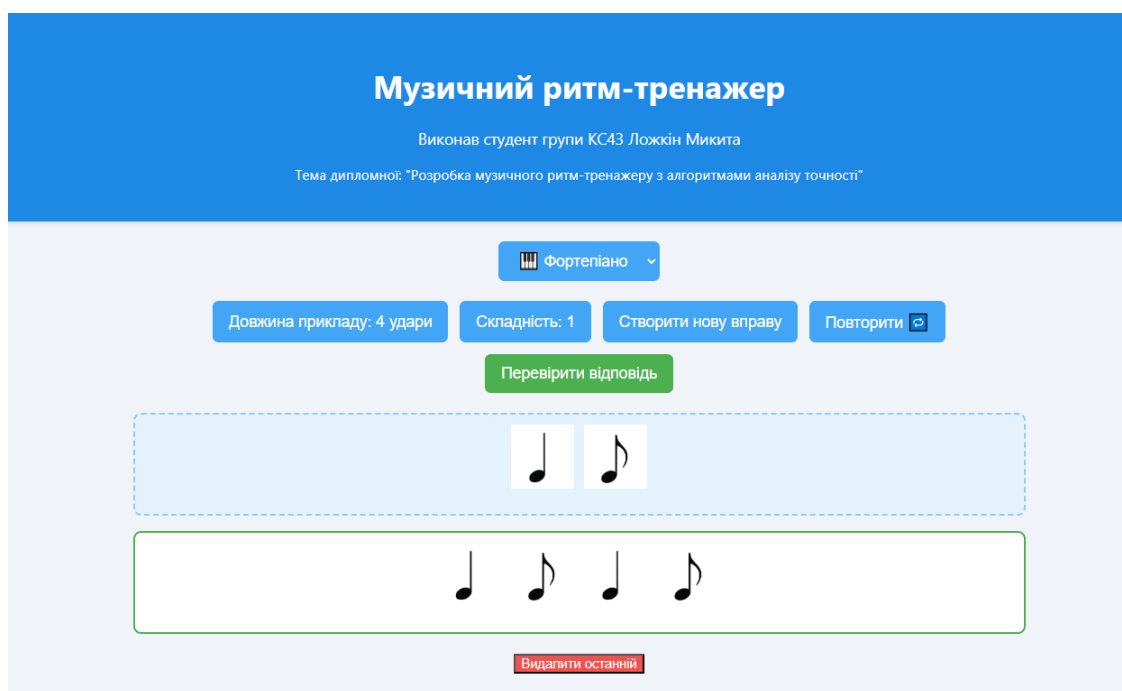


Рисунок 2.4.1 – Інтерфейс режиму “фортепіано”

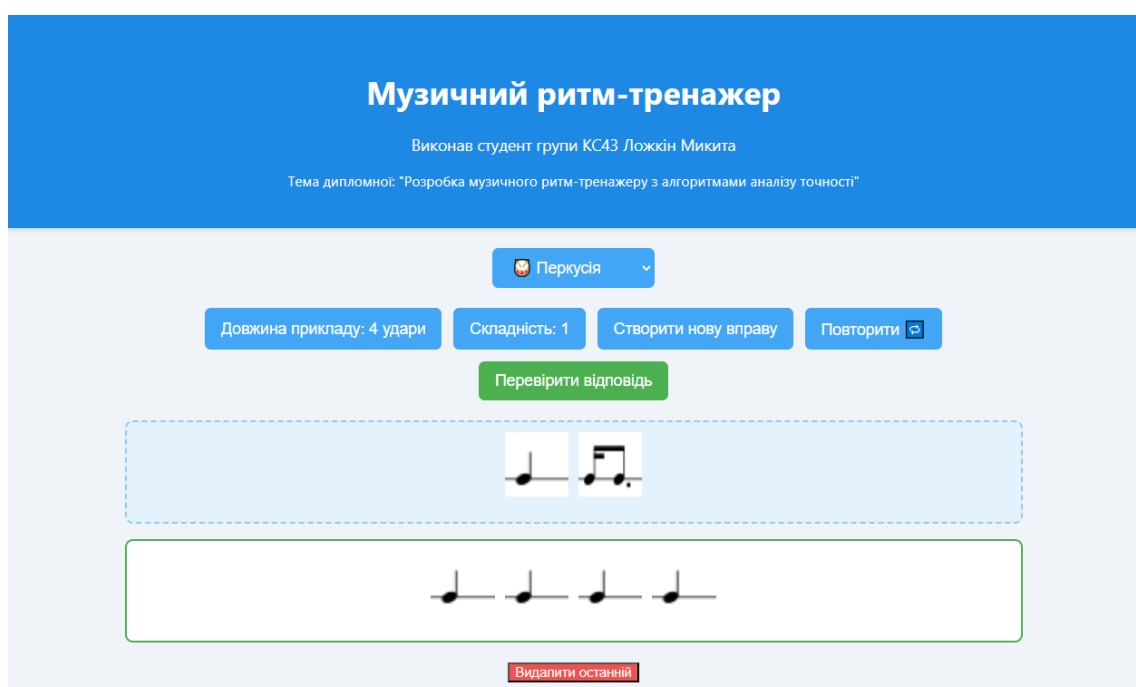


Рисунок 2.4.2 – Інтерфейс режиму “перкусія”

2.5. Механізм аналізу точності виконання

Оцінка точності виконання ритмічного завдання — це ключовий компонент, який перетворює ритм-тренажер із простого музичного редактора

на повноцінний інструмент навчання. Саме ця функціональність забезпечує реальний зворотний зв'язок, дозволяючи користувачу зрозуміти, наскільки точно він або вона відтворює заданий ритм, які помилки допускаються, і як їх можна усунути в подальшому навчанні. На мою думку, саме елемент аналізу виконавської точності надає системі педагогічної цінності й наближає її до професійних тренажерів, які використовуються в академічному музичному навчанні.

Цей підрозділ розкриває технічні аспекти реалізації механізму аналізу: як саме фіксуються й обробляються таймінги, за якими критеріями обчислюється відхилення, якими алгоритмами здійснюється оцінка правильності виконання. Не менш важливим є й аспект візуалізації — користувач має не просто отримати суху числову оцінку, а й наочно побачити свою похибку, бажано в зручній та інтуїтивно зрозумілій формі.

У межах цього підрозділу також буде звернено увагу на деякі особливості, що виникали в процесі розробки — зокрема, проблеми із затримками під час введення, обробки введених значень у реальному часі, складність синхронізації аудіо та візуальних елементів тощо. Всі ці моменти виявилися критичними для досягнення точного й об'єктивного аналізу.

2.5.1. Збір і обробка таймінгів

Одним з основних викликів під час розробки системи аналізу точності ритмічного виконання є коректний і своєчасний збір даних про момент натискання клавіші користувачем. Цей процес, на перший погляд, виглядає простим, однак у реальності він супроводжується низкою технічних нюансів, що потребують особливої уваги.

Збір таймінгів, тобто часових позначок моментів натискань, здійснюється за допомогою прослуховування подій натискання клавіш у браузері. У нашій реалізації це відбувається шляхом прив'язки до події `keydown`, яка фіксує системний час у момент взаємодії користувача. Часовою одиницею вимірювання є мілісекунди від початку сесії або початку

відтворення вправи. Такий підхід дозволяє створити масив часових точок, які відображають реальне ритмічне виконання.

Після первинного збору, ці значення зберігаються у внутрішній структурі даних, яка в подальшому використовується для зіставлення з еталонною ритмічною послідовністю. Саме на цьому етапі важливо враховувати затримки, що можуть виникати внаслідок роботи браузера, операційної системи або фізичних характеристик пристрою користувача. Ці затримки не завжди є критичними, однак у деяких випадках можуть суттєво впливати на результати аналізу.

Ще однією важливою проблемою, яку довелося врахувати, є ситуації, коли користувач вводить значення передчасно або з великим запізненням. Тому до зібраного масиву даних застосовується попередня обробка — зокрема, фільтрація значень, які суттєво виходять за межі очікуваних вікон натискання. Крім того, реалізовано механізм очищення дублікатів, що можуть виникати у випадку випадкових подвійних натискань.

На мою думку, саме точність збору таймінгів створює фундамент для подальшого коректного аналізу. Якщо на цьому етапі будуть допущені суттєві похибки, жоден алгоритм не зможе достовірно оцінити правильність виконання. Тому реалізація цієї частини вимагала найбільшої уваги, тестування на різних пристроях та в різних умовах.

У наступному пункті буде описано, як саме ці дані використовуються для розрахунку точності виконання.

2.5.2. Алгоритм розрахунку точності

Після того як система збрала часові мітки натискань користувача під час виконання вправи, необхідно оцінити, наскільки ці натискання відповідають очікуваному ритму. Для цього реалізовано алгоритм порівняння фактичних таймінгів із заздалегідь заданими референсними часовими точками.

Суть алгоритму полягає у визначенні відхилення кожного натискання користувача від ідеального часу. Кожен ритмічний елемент має свою очікувану позицію в часі (еталонну мітку), і на її основі обчислюється абсолютна похибка. Чим менше ця похибка — тим точнішим вважається натискання. Однак, з огляду на особливості сприйняття ритму та природні людські коливання, допустимий інтервал (вікно похибки) встановлюється заздалегідь — наприклад, ± 150 мілісекунд.

Алгоритм проходить по кожному елементу вправи та зіставляє його з найближчим натисканням користувача. Далі обчислюється точність за наступною простою формулою:

$$\text{Точність} = \left(1 - \frac{\sum_{i=1}^n |t_i - r_i|}{n \cdot W}\right) \cdot 100\% \quad (2.5)$$

де:

t_i — час натискання користувача для i -го елемента,

r_i — очікуваний (референсний) час для i -го елемента,

W — максимально допустиме відхилення (вікно похибки),

n — загальна кількість ритмічних елементів у вправі.

Таким чином, чим ближче натискання до ідеального моменту — тим вищий показник точності. Якщо похибка перевищує допустиме вікно, то відповідне натискання вважається неточним і не зараховується у загальну формулу.

На мою думку, запропонована модель оцінювання має достатню гнучкість і простоту, водночас забезпечує об'єктивне оцінювання ритмічної точності. Вона не лише дозволяє враховувати природну варіативність користувача, але й адаптується до вправи будь-якої складності.

Завдяки такому підходу система здатна формувати зворотний зв'язок, що є корисним для навчання — користувач бачить не лише загальну оцінку, але й розуміє, в яких саме моментах були допущені похибки.

2.5.3. Візуалізація результатів та зворотний зв'язок користувачу

Одним із ключових елементів ефективного навчального процесу є надання своєчасного, інформативного та візуально зрозумілого зворотного зв'язку. У рамках розробленого музичного ритм-тренажера передбачена система візуалізації результатів виконаної вправи, яка дозволяє користувачеві не лише побачити загальну оцінку, а й зрозуміти свої сильні та слабкі сторони.

Візуалізація реалізована у вигляді зміни кольору рамки навколо поля введених нот: зелений колір свідчить про правильне виконання ритму, тоді як червоний — про наявність помилок. Такий спосіб миттєвого зворотного зв'язку дозволяє користувачеві швидко оцінити результат, не заглиблюючись у складні аналітичні показники.

На додаток до кольорової індикації, система виводить числові значення:

- загальний відсоток точності виконання вправи;
- середній час реагування;
- кількість правильних і неправильних нот;
- середню точність за всіма вправами (якщо статистика не скинута вручну).

На мою думку, такий багаторівневий підхід до зворотного зв'язку дозволяє не лише надати користувачеві загальну оцінку, а й мотивує до подальшого вдосконалення. Поєднання простих візуальних сигналів із більш детальною статистикою створює збалансовану систему навчання — вона інтуїтивно зрозуміла початківцю, але при цьому достатньо інформативна для досвідченого користувача.

Особливо важливим є той факт, що система підтримує кумулятивну статистику — показники оновлюються з кожною вправою, що дозволяє відстежувати прогрес у динаміці. Цей аспект, на мою думку, створює додаткову мотивацію для регулярних занять та самостійного контролю успішності.

Таким чином, візуалізація та зворотний зв'язок виступають невід'ємною частиною тренажера, підвищуючи ефективність навчання через простоту, зрозумілість та доступність інформації про власні результати.

2.6. Дизайн інтерфейсу та взаємодія з користувачем

У сучасному програмному забезпеченні, особливо в галузі освітніх технологій, дизайн інтерфейсу та організація взаємодії з користувачем відіграють надзвичайно важливу роль. Від того, наскільки зручно, інтуїтивно та приємно працювати з додатком, залежить не лише рівень задоволеності користувача, а й ефективність самого навчального процесу. Саме тому розробка інтерфейсу музичного ритм-тренажера розглядалася як один із пріоритетних етапів проєктування системи.

Завданням інтерфейсу є не просто відображення інформації, а створення цілісного користувацького досвіду, який сприяє навчанню, мінімізує перешкоди у взаємодії та створює позитивне враження від використання. Особливу увагу було приділено візуальній структурі елементів, логіці переходів між діями, адаптивності під різні пристрої та врахуванню психологічних особливостей сприйняття музичного контенту.

На мою думку, саме в освітніх застосунках інтерфейс не може бути лише естетичним оформленням — він повинен стати повноцінним педагогічним інструментом, який сприяє формуванню навичок, мотивує до самостійної роботи та забезпечує комфортне середовище для навчання.

У цьому підрозділі буде детально розглянуто основні принципи побудови інтерфейсу, використані підходи до UX/UI-дизайну, кольорові рішення, а також реалізацію адаптивності під різні типи пристроїв.

2.6.1. Загальні принципи дизайну та UX

Розробка дизайну інтерфейсу музичного ритм-тренажера базувалась на принципах сучасного UX/UI-дизайну, що сформувались на перетині психології, ергономіки та технологій. У центрі уваги завжди перебуває

користувач, а всі рішення мають на меті зробити взаємодію з програмою максимально простою, інтуїтивною та приємною.

Одним із основоположних принципів при проєктуванні інтерфейсу є принцип простоти. Він полягає в тому, що користувач повинен з мінімальними зусиллями орієнтуватися у додатку та розуміти, які дії потрібно виконати для досягнення мети. Тому кожен елемент інтерфейсу був ретельно продуманий, щоб не перевантажувати екран зайвою інформацією, а водночас забезпечувати достатній рівень функціональності.

Іншим важливим принципом є єдність стилю та логічна структура. Усі кнопки, панелі, шрифти та кольори витримані в одному візуальному стилі, що формує цілісне сприйняття додатку. Структура сторінки була побудована так, щоб користувач міг легко переміщатися між блоками — від генерації вправ до перевірки відповідей та перегляду статистики.

Також було враховано принцип візуального ієрархування — найбільш важливі елементи, такі як кнопки взаємодії, розміщено в центральній частині інтерфейсу з помітними кольорами та зрозумілими підписами. Менш значущі інформаційні блоки, наприклад статистика, розташовані внизу, щоб не відволікати увагу в процесі виконання вправ.

Особливу увагу було приділено юзабіліті (зручності використання), яке проявляється не лише в розміщенні елементів, а й у відгуку на дії користувача. Наприклад, кольорова індикація рамки навколо поля відповіді дозволяє миттєво зрозуміти результат перевірки — зелений сигналізує про правильну відповідь, а червоний — про помилку. Ці деталі мають на меті зменшити когнітивне навантаження на користувача, забезпечуючи швидкий і зручний фідбек.

На мою думку, однією з ключових переваг інтерфейсу є його емоційна привабливість. Через використання музичних символів, приємних кольорів та плавної анімації користувач отримує не лише функціональний, але й емоційно позитивний досвід. Це особливо важливо для освітнього програмного

забезпечення, де мотивація та естетичне задоволення грають неабияку роль у процесі навчання.

Таким чином, дотримання сучасних принципів UX/UI-дизайну забезпечило створення інтерфейсу, який не лише відповідає вимогам функціональності, але й сприяє ефективному засвоєнню матеріалу, викликає довіру в користувача та стимулює повторне використання програми.

2.6.2. Оформлення елементів інтерфейсу, кольорова палітра

Оформлення інтерфейсу є важливою складовою користувацького досвіду, оскільки візуальна привабливість програми безпосередньо впливає на зацікавленість і комфорт користувача під час роботи з нею. Візуальне середовище ритм-тренажера було розроблено з урахуванням принципів сучасного UI-дизайну, що передбачають гармонійне поєднання кольорів, логічне розміщення елементів і виразну типографіку.

Основний фон інтерфейсу виконано у світлих тонах (світло-сірий), що створює спокійне тло, не перевантажує зорове сприйняття та дозволяє зосередити увагу користувача на ключових функціональних зонах. Верхня панель має яскравіший синій відтінок, який служить орієнтиром і візуально розділяє навігаційний блок від основного контенту.

У якості кольорів акцентів використано насичені відтінки зеленого (для підтвердження правильності дій) та червоного (для позначення помилок), що відповідає універсальним візуальним кодам сприйняття інформації. Такі сигнали сприймаються інтуїтивно і не потребують додаткових пояснень, що особливо важливо в умовах освітнього інструмента.

Шрифти в інтерфейсі підбрані таким чином, щоб забезпечити читабельність навіть на невеликих екранах. Основний текст подано стандартним безсерифним шрифтом із помірною жирністю, що забезпечує баланс між візуальною легкістю і виразністю. Назви кнопок та заголовків зроблено більш контрастними, з акцентованим стилем для кращої навігації.

Кожен елемент інтерфейсу — кнопки, панелі, поля вводу — має чітко визначені межі та реакцію на дії користувача. Наприклад, при наведенні миші на інтерактивну кнопку вона змінює відтінок або рамку, сигналізуючи про можливість взаємодії. Такі мікровзаємодії є важливою частиною UX, оскільки вони підвищують відчуття контролю й зворотного зв'язку.

Особливої уваги заслуговує використання музичних зображень — нот, хвиль, графічних символів, що органічно вписуються в інтерфейс і підсилюють тематичну складову додатку. Ці елементи не лише покращують естетичне сприйняття, а й відіграють роль візуальних маркерів, що орієнтують користувача в межах інтерфейсу.

На мою думку, завдяки виваженому підходу до кольорової палітри та оформлення елементів вдалося створити інтерфейс, що є водночас функціональним і емоційно привабливим. Це особливо важливо для навчального ПЗ, де дизайн не має відволікати, а навпаки — допомагати в процесі засвоєння матеріалу.

2.7. Розширені можливості та додатковий функціонал

У процесі розробки будь-якого сучасного освітнього програмного забезпечення особливе значення має не лише реалізація базових функцій, а й впровадження додаткових можливостей, які покращують взаємодію з користувачем, підвищують мотивацію до навчання та забезпечують зручність використання системи. У випадку ритм-тренажера, функціональні розширення відіграють роль «надбудови» до основного механізму перевірки ритму та дозволяють створити повноцінне, адаптивне та дружнє середовище для користувача.

До таких функцій можна віднести ведення статистики успішності, збереження історії тренувань, автоматичний підрахунок середніх значень точності, адаптацію рівня складності тощо. Такі розширення, з одного боку, не є критично необхідними для мінімальної працездатності додатку, але з

іншого — роблять його по-справжньому ефективним і привабливим для широкої аудиторії.

На мою думку, саме ці додаткові елементи підсилюють навчальний потенціал системи, дозволяючи користувачу не лише отримувати зворотній зв'язок у реальному часі, а й відстежувати свій прогрес у довгостроковій перспективі. Нижче розглянемо реалізацію окремих із них детальніше.

2.7.1. Ведення статистики користувача

Система ведення статистики користувача є ключовим компонентом у побудові навчального прогресу та наданні персоналізованого досвіду. У контексті музичного ритм-тренажера ця функція дозволяє зберігати дані про кожну вправу, що виконувалася користувачем, включаючи такі параметри, як: кількість спроб, середній показник точності, час виконання, кількість правильних відповідей, дата та час проходження вправи.

Зібрані показники зберігаються у структурованому форматі, що дозволяє не лише переглядати їх в інтерфейсі користувача у вигляді таблиці або картки, а й аналізувати динаміку змін за певний період. Так, наприклад, користувач може побачити, що його середня точність зросла з 72% до 84% за тиждень регулярних занять — така інформація суттєво підвищує мотивацію до подальших тренувань.

На технічному рівні для реалізації цієї функції було використано локальне збереження даних (через LocalStorage браузера), що дозволяє уникнути залежності від серверної частини та забезпечити автономну роботу системи. У подальшому, за потреби, система може бути доповнена збереженням статистики в базі даних через API — це забезпечить кросплатформений доступ до результатів і синхронізацію між пристроями.

З мого погляду, наявність такої функції робить інструмент не просто тренажером, а повноцінним помічником у навчальному процесі. Статистика не лише інформує, а й формує відчуття досягнення, дозволяючи користувачу відчувати прогрес, що є надзвичайно важливим у сфері освіти.

2.7.2. Скидання статистики, підрахунок середніх значень

Однією з важливих складових функціоналу системи статистики є можливість її обнулення, а також коректний розрахунок середніх показників на основі накопичених даних. Ці функції, хоча й можуть видаватися другорядними на перший погляд, насправді мають суттєвий вплив на зручність використання тренажера, інформативність виведеної інформації та підтримку мотивації користувача.

Механізм скидання статистики передбачає можливість повного видалення попередніх результатів тренувань. Це корисно у випадках, коли користувач розпочинає тренування заново після тривалої перерви або ж хоче протестувати себе в новому режимі, не зважаючи на попередні результати. З технічного боку, скидання реалізується через очищення локального сховища браузера (LocalStorage), після чого всі статистичні параметри повертаються до значень за замовчуванням.

Щодо розрахунку середніх значень, то система забезпечує автоматичний підрахунок середньої точності та середнього часу виконання на основі всіх наявних результатів. Ці значення дають змогу користувачу бачити узагальнений рівень своєї продуктивності за обраний період або за всі сесії тренувань. Для прикладу, середня точність обчислюється за формулою:

$$\text{Середня точність} = \frac{\sum_{i=1}^n T_i}{n} \quad (2.7)$$

де:

T_i — точність виконання i -ої вправи,

n — кількість вправ.

Аналогічно обчислюється середній час виконання. Ці значення виводяться в окремому блоці статистики в інтерфейсі, що дає змогу швидко оцінити динаміку прогресу.

На мою думку, навіть така на перший погляд дрібна функція, як можливість очищення статистики, виконує важливу психологічну роль —

вона дає змогу почати з «чистого аркуша» без тиску попередніх результатів, а відтак, знижує бар'єр входження у процес навчання. А розрахунок середніх значень дозволяє формувати реалістичну картину результатів, уникаючи суб'єктивних вражень.

2.7.3. Можливості для майбутнього розширення

Хоча поточна версія музичного ритм-тренажера реалізує базовий функціонал для тренування ритмічної точності, з точки зору подальшого розвитку існує чимало перспективних напрямків, які дозволяють значно розширити функціональність системи та підвищити її освітню цінність.

1. Режим навчання.

Наразі тренажер працює в режимі тестування — користувачеві пропонується приклад, а його завдання полягає у точному повторенні. У майбутньому можливо реалізувати повноцінний навчальний режим, який включатиме поступове введення теоретичного матеріалу (наприклад, що таке доля, тривалість, синкопа тощо) та інтерактивні вправи зі зростаючим рівнем складності. Такий підхід дозволить зробити тренажер більш придатним для новачків, які тільки починають знайомство з музичним ритмом.

2. Підключення до онлайн-бази вправ.

Ще одним логічним кроком розвитку системи є інтеграція з онлайн-сховищем ритмічних вправ. У такому випадку користувач зможе отримувати нові приклади не лише згенеровані локально, але й завантажені з репозиторію, який може бути постійно оновлюваним викладачами або спільнотою. Це дозволить підвищити різноманітність завдань та зробити процес тренування динамічнішим і багатшим.

3. Адаптивність вправ.

На мою думку, важливим напрямком розвитку є створення адаптивної системи, яка автоматично змінює складність вправ залежно від попередніх результатів користувача. Якщо точність знижується, система може зменшити

темп або спростити ритмічний малюнок. І навпаки, у разі високої точності — ускладнювати ритми або зменшувати допустимий розкид у таймінгу.

4. Розширення режимів.

Окрім двох основних режимів — «фортепіано» та «перкусія», — можна додати нові варіанти, наприклад:

- тренування поліритмії (одночасне виконання різних ритмів);
- вивчення метричних акцентів у складних розмірах;
- імітація ансамблевого виконання з візуальною підтримкою інших «учасників».

5. Інтеграція з іншими пристроями.

У майбутньому передбачено можливість використання тренажера не лише на комп'ютері, але й на мобільних пристроях з підтримкою мультитач. Крім того, потенційно можливе підключення зовнішніх MIDI-контролерів, що дозволить тренуватися з реальними інструментами, наприклад, електронними барабанами.

6. Зворотний зв'язок та рекомендації.

Для покращення навчального процесу можна реалізувати механізм персоналізованого зворотного зв'язку. Наприклад, після кожної вправи користувач отримує не лише оцінку, але й конкретні поради: «запізнення в другому ударі», «нестабільний темп наприкінці» тощо. Це дозволить більш глибоко аналізувати свої помилки.

Таким чином, запропонований інструмент має широке поле для еволюції. Впровадження вищезазначених напрямів зробить систему більш універсальною, гнучкою та привабливою як для індивідуального навчання, так і для використання у професійному освітньому середовищі.

3. ВИСНОВКИ ДО РЕЗУЛЬТАТІВ РОБОТИ

3.1. Оцінка досягнення поставленої мети

У процесі виконання цієї кваліфікаційної роботи основною метою дослідження та практичної реалізації було створення інтерактивного музичного ритм-тренажера з можливістю аналізу точності виконання ритмічних завдань. Початково сформульовану мету можна вважати повністю досягнутою, адже реалізована система забезпечує основні функції, передбачені під час планування: генерацію ритмічних вправ, їхнє програвання, можливість взаємодії з користувачем через інтерфейс клавіш, а також автоматичну перевірку точності виконання з виведенням статистичних результатів.

Поставлені завдання, які включали формулювання функціональних та нефункціональних вимог, проектування архітектури застосунку, вибір інструментів реалізації, створення інтерфейсу користувача, розробку алгоритмів аналізу точності, а також впровадження механізмів статистичного підрахунку результатів, були успішно виконані в межах дослідження. На мою думку, особливо важливим було знайти баланс між простотою користування системою та забезпеченням глибокого аналізу виконання — і це завдання було реалізовано достатньо вдало.

Ще на початковому етапі розробки передбачалося створення тренажера, який би мав два режими: фортепіанний та перкусійний. Обидва режими були впроваджені й пройшли базове тестування. Режим фортепіано дозволяє користувачам вводити ноти з урахуванням висоти звуку, тоді як режим перкусії фокусується виключно на ритмічному аспекті. Такий підхід дав змогу охопити ширшу аудиторію потенційних користувачів і краще реалізувати поставлену мету.

Не менш важливим завданням було створення візуально привабливого та інтуїтивно зрозумілого інтерфейсу. Завдяки використанню сучасних

вебтехнологій (HTML, CSS, JavaScript) вдалося досягти відповідного рівня юзабіліті, що робить систему зручною у використанні як для новачків, так і для більш досвідчених користувачів.

Таким чином, як з технічної, так і з функціональної точки зору, результати, отримані в ході розробки, підтверджують успішне досягнення поставленої мети. Система демонструє стабільну роботу, відповідає заявленим вимогам і здатна виконувати ті завдання, що були поставлені на початковому етапі розробки. Крім того, реалізація додаткових елементів, таких як статистика виконання вправ, кольорове маркування правильності відповіді та зручні елементи керування, слугує підтвердженням того, що проєкт розвивався не лише відповідно до мінімальних цілей, а й включав елементи вдосконалення та розширення функціональності.

3.2. Аналіз результатів та їх відповідність сучасним тенденціям

У сучасному інформаційному просторі спостерігається стрімкий розвиток цифрових технологій у сфері освіти, зокрема музичної. Все більше інструментів створюється з метою автоматизації, спрощення та підвищення ефективності процесу навчання. На цьому тлі особливої уваги заслуговують інтерактивні ритм-тренажери, здатні поєднувати навчальну методику з миттєвим зворотним зв'язком, а також оцінюванням дій користувача. З цієї точки зору розроблений у межах дипломної роботи музичний ритм-тренажер не лише виконує свою основну навчальну функцію, а й відповідає ключовим тенденціям розвитку галузі.

На мою думку, одним з найсуттєвіших досягнень реалізованої системи є впровадження алгоритмів аналізу точності ритмічного відтворення. Це дозволяє вийти за межі традиційного «натисни кнопку – почуй звук» і перейти до глибшого рівня навчання, коли користувач отримує адекватну оцінку власної ритмічної точності. Такий підхід є надзвичайно актуальним у світлі глобального тренду на персоналізацію освітніх процесів та використання елементів штучного інтелекту у педагогіці.

Система дозволяє враховувати тимчасові відхилення користувача від еталонного ритму, надає кількісні оцінки точності, зберігає статистику успішності та візуалізує результати взаємодії. Все це сприяє формуванню в учня навичок самостійного аналізу власних помилок, а отже – розвитку музичної саморефлексії, що є критично важливим для ефективного музичного навчання.

Варто відзначити, що проєкт не використовує складних або надто ресурсомістких технологій, що також збігається із сучасною тенденцією до легковагових, оптимізованих веб-додатків, придатних для роботи на широкому спектрі пристроїв – від настільних ПК до мобільних гаджетів. Така кросплатформенність дозволяє охопити ширше коло користувачів та розширити сферу застосування тренажера.

Крім того, відповідність сучасним тенденціям підтверджується і архітектурою самого додатку. Його побудовано за принципами модульності, що дозволяє легко масштабувати та оновлювати функціонал у майбутньому. Наприклад, уже на етапі створення було передбачено можливість роботи в кількох режимах – фортепіанному та перкусійному – що є ознакою гнучкості проєкту.

Загалом, аналіз отриманих результатів показує, що реалізована система не тільки відповідає поставленим цілям, але й органічно вписується в контекст сучасних технологічних рішень, спрямованих на удосконалення навчального процесу в галузі музики. Саме завдяки поєднанню технічної реалізації, методичної обґрунтованості та інтуїтивної взаємодії з користувачем система може бути розглянута як сучасне інноваційне рішення, здатне конкурувати з існуючими аналогами.

3.3. Можливі напрями використання та подальшого розвитку системи

У процесі розробки музичного ритм-тренажера особливу увагу було приділено створенню універсального, зручного та масштабованого

інструменту, придатного для широкого спектру користувачів. Проте, як показує аналіз поточних результатів і сучасних освітніх потреб, розроблена система має значний потенціал для подальшого удосконалення та розширення сфер застосування.

На першому етапі найбільш очевидним є використання тренажера в навчальних закладах – як частина інтерактивного контенту на уроках музики в загальноосвітніх школах, спеціалізованих музичних школах, а також в межах додаткової освіти. Простота інтерфейсу та гнучкість налаштувань дозволяють адаптувати вправи під вік, навички та цілі користувача. На мою думку, такий підхід може значно підвищити мотивацію учнів, адже навчання через гру в поєднанні з миттєвим зворотним зв'язком – це перевірена часом методика ефективного засвоєння знань.

Ще одним перспективним напрямом може стати розширення можливостей тренажера для професійних музикантів, зокрема барабанщиків, піаністів, а також студентів музичних вишів. Для цієї категорії користувачів доцільно було б реалізувати більш складні режими оцінювання, включаючи аналіз динаміки натискання, тембрової диференціації та варіативності темпу.

З технічної точки зору перспективним виглядає також створення мобільної версії програми, що дозволить користувачам тренуватись у будь-який зручний час. З огляду на те, що мобільні пристрої сьогодні фактично замінили традиційні комп'ютери для повсякденних задач, адаптація інтерфейсу під екрани різних розмірів та розширень відкриє нові горизонти для використання розробки.

Крім того, можна розглянути можливість інтеграції із зовнішніми MIDI-пристроями та цифровими клавіатурами. Такий крок дозволить підвищити точність взаємодії користувача із системою, а також розширити сферу застосування тренажера у студіях звукозапису та навчальних класах із відповідним обладнанням.

Окремої уваги заслуговує перспектива колаборативного навчання. В майбутньому можливо реалізувати багатокористувацький режим, у якому учні

зможуть виконувати завдання синхронно або в режимі виклику. Такий функціонал, на мою думку, зробить процес навчання ще цікавішим, сприятиме розвитку командної роботи та формуванню здорової конкуренції між учасниками.

Також перспективним є напрям збору й обробки статистичних даних про навчальний прогрес. Це дозволить формувати персоналізовані поради, аналізувати типові помилки, а згодом — навіть будувати адаптивні сценарії тренувань на основі результатів попередніх сесій.

У підсумку варто наголосити, що описаний у дипломній роботі ритм-тренажер є лише першою і, можна сказати, базовою реалізацією більш широкої концепції музичного навчального середовища. На основі вже реалізованих функцій можна формувати більш складні продукти — як для індивідуального використання, так і для роботи у групах, для дітей або дорослих, для новачків чи професіоналів.

ВИСНОВКИ

1. У процесі виконання кваліфікаційної роботи була розроблена функціональна вебсистема — музичний ритм-тренажер, що реалізує інтерактивне навчання ритмічним шаблонам та оцінює точність виконання завдань користувачем. Такий підхід поєднує у собі сучасні цифрові методики навчання з гейміфікацією, що сприяє підвищенню мотивації та ефективності засвоєння матеріалу.

2. На основі огляду літератури та аналізу наявних програмних рішень було встановлено, що хоча існують інструменти для розвитку музичного слуху та гри на інструментах, напрямок точного ритмічного тренування з інтерактивною оцінкою у реальному часі є відносно малодослідженим. Це підтверджує актуальність обраної тематики та доцільність реалізованої розробки.

3. Під час реалізації було визначено низку функціональних і нефункціональних вимог до системи. Вибрані технології (HTML, CSS, JavaScript) забезпечили кросплатформеність, простоту розгортання та зручність використання. Особливу увагу приділено модульності коду, що дозволяє легко масштабувати функціональність у майбутньому.

4. Основною технічною особливістю реалізованої системи є алгоритм аналізу точності ритмічного введення, який дозволяє надавати зворотний зв'язок у вигляді точного відсоткового показника. Цей механізм є ключовим елементом навчального процесу, оскільки забезпечує об'єктивну оцінку результатів виконання вправ.

5. Інтерфейс тренажера створено з урахуванням принципів простоти, інтуїтивності та привабливості. Важливою перевагою є наявність візуального нотного представлення, адаптації до двох режимів роботи (фортепіано та перкусія), а також статистики прогресу користувача.

6. На мою думку, найбільш цінним у роботі стало поєднання технологічної простоти з педагогічною ефективністю. Розробка не лише

виконує свою освітню функцію, а й може стати базисом для майбутніх досліджень та розробок у галузі цифрової музичної освіти.

7. У подальшому розробку можна вдосконалити, реалізувавши мобільну версію, багатокористувацький режим, підтримку MIDI-пристроїв, а також адаптивні тренування на основі штучного інтелекту. Це значно розширить можливості системи та дозволить їй конкурувати з комерційними продуктами світового рівня.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Лаврищева К. М. Програмна інженерія: підручник. Київ: КНЕУ, 2008. 319 с. URL: <http://cyb.univ.kiev.ua/library/books/lavrishcheva-6.pdf>
2. Семеріков С. О., Триус Ю. В. Інформаційно-комунікаційні технології в освіті. Київ: Міленіум, 2020. 312 с.
3. URL: <https://lib.iitta.gov.ua/723177/>Рогальська Ю. Ю. Мультимедійні технології в освіті. Київ: Центр учбової літератури, 2021. 288 с.
4. Mishra, S., Koehler, M. J. Technological Pedagogical Content Knowledge: A Framework for Teacher Knowledge // Teachers College Record. 2006. Vol. 108(6). P. 1017–1054.
5. Nielsen, S. Interactive Piano Instruction and the Internet // Music Education Research, 2001. Vol. 3, No. 2. P. 155–167.
6. Stepanova, N. V. Using Rhythm Apps in Music Education for Preschoolers // ICTERI 2020 Workshops, Vol. 2732. P. 234–241. URL: http://ceur-ws.org/Vol-2732/paper_25.pdf
7. Співак С. В. Інтерактивні тренажери у підготовці студентів музичних спеціальностей // Педагогіка та психологія професійної освіти. 2020. №1. С. 99–103.
8. Gamzayev, R., Vekshin, A. A Model-Based Framework for Adaptive Resource Management in Mobile Augmented Reality System // ICTERI-2016, Kyiv, CEUR-WS Vol. 1614, P. 41–56.
9. Parncutt, R. Systematic training of rhythm perception and production // Music Perception, 1994. Vol. 11, No. 4. P. 449–456.
10. Leong, S., & Nakazone, A. Using rhythm training software in music classrooms // Journal of Music Technology & Education. 2015. Vol. 8(3). P. 275–290.
11. Nakra, T. C. Digital Conducting: Human–Computer Interaction in Music Performance // Proceedings of the 2006 Conference on New Interfaces for Musical Expression. P. 147–152.

12. Ho, W. C. Technology Integration in Music Classrooms: A Review of Empirical Research // Update: Applications of Research in Music Education, 2007. Vol. 25(2). P. 57–70.
13. Interactive Rhythm Learning System by Combining Tablet Computers and Robots. Applied Sciences. URL: <https://www.mdpi.com/2076-3417/7/3/258>
14. Digital rhythm training improves reading fluency in children. // Developmental Science. URL: <https://onlinelibrary.wiley.com/doi/10.1111/desc.13473>
15. Tuning Music Education: AI-Powered Personalization in Learning Music. // arXiv. 2024 URL: <https://arxiv.org/abs/2412.13514>
16. A Computational Approach to Rhythm Training. // arXiv 2021. URL: <https://arxiv.org/abs/2109.04440>
17. EarMaster: The #1 App for Ear Training, Sight-Singing, and Rhythm. URL: <https://www.earmaster.com>
18. Perfect Ear – Ear Trainer. URL: <https://www.perfectear.app>
19. MusiQuest: A creative new way to learn music. URL: <https://app.musiquest.com>
20. Yousician – Learn Guitar, Piano, Ukulele, Bass & Singing. URL: <https://yousician.com>

ДОДАТОК А. ПРИКЛАДИ ВИХІДНОГО КОДУ

Лістинг А.1 (index.html):

```

<!DOCTYPE html>
<html lang="uk">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width,
initial-scale=1.0">
  <title>Музичний ритм-тренажер</title>
  <link rel="stylesheet" href="style.css">
</head>
<body>
  <header>
    
    

    <h1>Музичний ритм-тренажер</h1>
    <p class="version">Виконав студент групи КС43 Ложкін
Микита</p>
    <p class="subtitle">Тема дипломної: "Розробка музичного
ритм-тренажеру з алгоритмами аналізу точності"</p>
  </header>

  <main>
    <div style="text-align: center;">
      <select id="mode-selector">
        <option value="piano" selected>🎹
Фортепіано</option>
        <option value="drum">🥁 Перкусія</option>
      </select>
    </div>

    <section class="controls">
      <button id="length-btn">Довжина прикладу: 4
удари</button>
      <button id="difficulty-btn">Складність: 3</button>
      <button id="new-exercise">Створити нову
вправу</button>
      <button id="replay">Повторити 🔄</button>
      <button id="check">Перевірити відповідь</button>
    </section>

    <section class="notation-area">
      <div id="notation"></div>
      <button id="delete">Видалити останній</button>

```

```

    </section>

    <section      id="accuracy-analysis"      style="display:
none;"></section>

    <section      id="statistics"      class="stat-block"
style="display: none;">
        <h3>Статистика</h3>
        <p>Кількість      вправ:      <span      id="total-
exercises">0</span></p>
        <p>Середня      точність:      <span      id="average-
accuracy">0%</span></p>
        <p>Кількість правильних відповідей: <span id="correct-
answers">0</span></p>
        <p>Середній час виконання: <span id="average-time">0
c</span></p>
        <button id="reset-stats">Скинути статистику</button>
    </section>
</main>

<footer>
    <p>Розроблено у рамках дипломної роботи (2025)</p>
    <p>Використано      бібліотеку      <a
href="https://github.com/surikov/webaudiofont">webaudiofont</a><
/p>
</footer>

<script
src="https://surikov.github.io/webaudiofont/npm/dist/WebAudioFon
tPlayer.js"></script>
<script
src="https://surikov.github.io/webaudiofontdata/sound/0000_JCLiv
e_sf2_file.js"></script>
<script src="rhythm.js"></script>
</body>
</html>

```

Лістинг A.2 (style.css):

```

body {
    font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-
serif;
    background-color: #f0f4f8;
    color: #333;
    margin: 0;
    padding: 0;
}

header {
    background-color: #1e88e5;
    color: white;

```

```

padding: 20px 20px 40px;
text-align: center;
box-shadow: 0 2px 4px rgba(0,0,0,0.1);
position: relative;
}

.header-note {
  position: absolute;
  top: 0;
  width: 300px;
  height: auto;
  opacity: 0.9;
  z-index: 1;
}

.header-note.left {
  left: 0;
}

.header-note.right {
  right: 0;
  transform: scaleX(-1);
}

header {
  position: relative;
  background-color: #1e88e5;
  color: white;
  text-align: center;
  padding: 40px 20px 20px 20px;
  overflow: hidden;
  min-height: 160px;
}

header h1, header .version, header .subtitle {
  position: relative;
  z-index: 2;
}

.subtitle {
  font-size: 14px;
  margin-top: 5px;
}

main {
  padding: 20px;
  max-width: 900px;
  margin: auto;
}

.controls {

```

```

    display: flex;
    flex-wrap: wrap;
    justify-content: center;
    gap: 12px;
    margin-bottom: 20px;
    margin-top: 20px;
  }

  .controls button,
  #mode-selector {
    padding: 10px 16px;
    font-size: 16px;
    border: none;
    border-radius: 6px;
    background-color: #42a5f5;
    color: white;
    cursor: pointer;
    transition: background-color 0.3s ease;
  }

  .controls button:hover,
  #mode-selector:hover {
    background-color: #1e88e5;
  }

  #check {
    background-color: #4caf50;
  }

  #check:hover {
    background-color: #388e3c;
  }

  #mode-selector {
    font-size: 16px;
    padding: 10px 16px;
    border-radius: 6px;
    border: none;
    background-color: #42a5f5;
    color: white;
    display: block;
    margin: 0 auto 10px auto;
  }

  .notation-area {
    margin-top: 20px;
    text-align: center;
  }

  #notation, #user-input {
    display: flex;
    justify-content: center;
    flex-wrap: wrap;

```

```

    gap: 10px;
    min-height: 80px;
    margin-bottom: 15px;
}

#notation {
    padding: 10px;
    border: 2px dashed #90caf9;
    border-radius: 8px;
    background-color: #e3f2fd;
}

#user-input {
    padding: 10px;
    border: 2px solid #cfd8dc;
    border-radius: 8px;
    background-color: #ffffff;
    transition: border-color 0.3s ease;
}

.note-img {
    cursor: pointer;
    transition: transform 0.2s ease;
}

#notation .note-img:hover {
    transform: scale(1.1);
}

#user-input .note-img {
    cursor: default;
    transform: none;
}

#delete {
    margin: 20px auto 0;
    display: block;
    background-color: #ef5350;
    color: white;
}

#delete:hover {
    background-color: #d32f2f;
}

#accuracy-analysis {
    margin-top: 30px;
    padding: 15px;
    border-radius: 6px;
    background-color: #e3f2fd;
    color: #1565c0;
    text-align: center;
    font-size: 18px;
}

```

```

    display: none;
    max-width: 600px;
    margin-left: auto;
    margin-right: auto;
}

footer {
    text-align: center;
    font-size: 14px;
    padding: 20px;
    color: #888;
}

.stat-block {
    background-color: #e3f2fd;
    border-radius: 10px;
    padding: 20px;
    margin: 30px auto 0;
    max-width: 360px;
    box-shadow: 0 2px 6px rgba(0, 0, 0, 0.1);
    text-align: center;
}

.stat-block h3 {
    margin-top: 0;
    font-size: 18px;
    color: #1565c0;
}

#reset-stats {
    margin-top: 15px;
    padding: 8px 14px;
    background-color: #ef5350;
    border: none;
    border-radius: 6px;
    color: white;
    cursor: pointer;
}

#reset-stats:hover {
    background-color: #c62828;
}

.correct-border {
    border: 2px solid #4caf50 !important;
}

.incorrect-border {
    border: 2px solid #ef5350 !important;
}

```

Лістинг А.3 (rhythm.js):

```

const notationArea = document.getElementById('notation');
const deleteBtn = document.getElementById('delete');
const replayBtn = document.getElementById('replay');
const checkBtn = document.getElementById('check');
const accuracyBlock = document.getElementById('accuracy-
analysis');
const lengthBtn = document.getElementById('length-btn');
const difficultyBtn = document.getElementById('difficulty-
btn');

const modeSelector = document.getElementById('mode-
selector');
const inputArea = document.createElement('div');
inputArea.id = 'user-input';
notationArea.after(inputArea);

let userRhythm = [];
let correctRhythm = [];
let userTimestamps = [];

let currentLength = 4;
let currentDifficulty = 3;
let currentMode = 'piano';
let firstPianoPlayback = true;

const DIFFICULTY_LEVELS = {
  1: ['quarter', 'eighth'],
  2: ['quarter', 'eighth', 'triplet'],
  3: ['quarter', 'eighth', 'triplet', 'sixteenth']
};

const NOTE_DATA_PIANO = {
  quarter: { img: 'piano/note_c4.png', pitch: 60 },
  eighth: { img: 'piano/note_d4.png', pitch: 62 },
  triplet: { img: 'piano/note_e4.png', pitch: 64 },
  sixteenth: { img: 'piano/note_f4.png', pitch: 65 }
};

const NOTE_DATA_DRUM = {
  quarter: { img: 'cheese.png', hits: 1 },
  eighth: { img: 'bacon.png', hits: 2 },
  triplet: { img: 'triplet.png', hits: 3 },
  sixteenth: { img: 'applejuice.png', hits: 4 }
};

const AudioContextFunc = window.AudioContext ||
window.webkitAudioContext;
const audioContext = new AudioContextFunc();
const player = new WebAudioFontPlayer();
const instrument = _tone_0000_JCLive_sf2_file;

```

```

const drumSound = new Audio('assets/sounds/drum.mp3');
const metronomeSound = new Audio('assets/sounds/metronome.mp3');

window.addEventListener('load', () => {
  player.queueWaveTable(audioContext,
    audioContext.destination, instrument, audioContext.currentTime,
    60, 0.05);
  difficultyBtn.innerText = `Складність:
  ${currentDifficulty}`;
  showAvailableNotes();
});

lengthBtn.addEventListener('click', () => {
  const options = [4, 6, 8, 10];
  const currentIndex = options.indexOf(currentLength);
  currentLength = options[(currentIndex + 1) %
options.length];
  const suffix = getUdarSuffix(currentLength);
  lengthBtn.innerText = `Довжина прикладу: ${currentLength}
  ${suffix}`;
});

difficultyBtn.addEventListener('click', () => {
  currentDifficulty = currentDifficulty % 3 + 1;
  difficultyBtn.innerText = `Складність:
  ${currentDifficulty}`;
  showAvailableNotes();
});

modeSelector.addEventListener('change', () => {
  currentMode = modeSelector.value;
  showAvailableNotes();
});

function getUdarSuffix(n) {
  if (n % 10 === 1 && n !== 11) return 'удар';
  if ([2, 3, 4].includes(n % 10) && ![12, 13, 14].includes(n))
return 'удари';
  return 'ударів';
}

function getNoteData() {
  return currentMode === 'piano' ? NOTE_DATA_PIANO :
NOTE_DATA_DRUM;
}

function showAvailableNotes() {
  notationArea.innerHTML = '';
  const notes = DIFFICULTY_LEVELS[currentDifficulty];
  const NOTE_DATA = getNoteData();
  notes.forEach(note => {
    const el = document.createElement('img');

```

```

        el.src = `assets/images/${NOTE_DATA[note].img}`;
        el.classList.add('note-img', 'note-btn');
        el.dataset.note = note;
        el.width = 64;
        el.height = 64;
        notationArea.appendChild(el);
    });
}

document.getElementById('new-
exercise').addEventListener('click', () => {
    userRhythm = [];
    userTimestamps = [];
    accuracyBlock.innerText = '';
    accuracyBlock.style.display = 'none';
    inputArea.innerHTML = '';
    inputArea.classList.remove('correct-border', 'incorrect-
border');

    const availableNotes =
DIFFICULTY_LEVELS[currentDifficulty];
    correctRhythm = generateRandomRhythm(currentLength,
availableNotes);

    showAvailableNotes();
    playRhythm(correctRhythm);
});

notationArea.addEventListener('click', (e) => {
    if (e.target.classList.contains('note-img')) {
        const note = e.target.dataset.note;
        userRhythm.push(note);
        userTimestamps.push(performance.now());
        playNote(note);

        const NOTE_DATA = getNoteData();
        const el = document.createElement('img');
        el.src = `assets/images/${NOTE_DATA[note].img}`;
        el.classList.add('note-img');
        el.width = 64;
        el.height = 64;
        inputArea.appendChild(el);
    }
});

deleteBtn.addEventListener('click', () => {
    userRhythm.pop();
    userTimestamps.pop();
    if (inputArea.lastChild) {
        inputArea.removeChild(inputArea.lastChild);
    }
});

```

```

replayBtn.addEventListener('click', () => {
  playRhythm(correctRhythm);
});

checkBtn.addEventListener('click', () => {
  const result = compareRhythm(userRhythm, correctRhythm);
  const timingScore = analyzeTimingAccuracy(userTimestamps);

  const timeTaken = (userTimestamps.length > 1)
    ? (userTimestamps[userTimestamps.length - 1] -
userTimestamps[0]) / 1000
    : 0;

  recordExerciseResult(result, timeTaken);

  inputArea.classList.remove('correct-border', 'incorrect-
border');
  inputArea.classList.add(result ? 'correct-border' :
'incorrect-border');
  accuracyBlock.innerHTML = result ? `YOU WERE RIGHT! Точність
виконання: ${timingScore}%` : `Try again... Точність виконання:
${timingScore}%`;
  accuracyBlock.style.display = 'block';

  inputArea.classList.remove('correct-border', 'incorrect-
border');
  inputArea.classList.add(result ? 'correct-border' :
'incorrect-border');

  const sound = new Audio(`assets/sounds/${result ? 'woohoo'
: 'tryagain'}.mp3`);
  sound.play();
});

let stats = {
  totalExercises: 0,
  correctAnswers: 0,
  totalAccuracy: 0,
  totalTime: 0
};

if (localStorage.getItem('stats')) {
  stats = JSON.parse(localStorage.getItem('stats'));
  updateStatistics();
}

function updateStatistics() {
  document.getElementById('total-exercises').textContent =
stats.totalExercises;
  document.getElementById('correct-answers').textContent =
stats.correctAnswers;

```

```

        document.getElementById('average-accuracy').textContent =
stats.totalExercises === 0 ? '0%' : Math.round(stats.totalAccuracy
/ stats.totalExercises) + '%';
        document.getElementById('average-time').textContent =
stats.totalExercises === 0 ? '0 c' : (stats.totalTime /
stats.totalExercises).toFixed(2) + ' c';

        document.getElementById('statistics').style.display =
'block';
    }

    function recordExerciseResult(isCorrect, timeTaken) {
        stats.totalExercises += 1;
        stats.totalTime += timeTaken;
        if (isCorrect) {
            stats.correctAnswers += 1;
            stats.totalAccuracy += 100;
        } else {
            stats.totalAccuracy += 0;
        }

        localStorage.setItem('stats', JSON.stringify(stats));
        updateStatistics();
    }

    function generateRandomRhythm(length, notes) {
        let rhythm = [];
        for (let i = 0; i < length; i++) {
            const randNote = notes[Math.floor(Math.random() *
notes.length)];
            rhythm.push(randNote);
        }
        return rhythm;
    }

    function playNote(note) {
        if (currentMode === 'piano') {
            const pitch = NOTE_DATA_PIANO[note].pitch;
            player.queueWaveTable(audioContext,
audioContext.destination, instrument, audioContext.currentTime,
pitch, 0.5);
        } else {
            const count = NOTE_DATA_DRUM[note].hits;
            for (let i = 0; i < count; i++) {
                setTimeout(() => {
                    drumSound.currentTime = 0;
                    drumSound.play();
                }, i * 150);
            }
        }
    }

    function playMetronome(count = 2, delay = 600) {

```

```

    for (let i = 0; i < count; i++) {
      setTimeout(() => {
        metronomeSound.currentTime = 0;
        metronomeSound.play();
      }, i * delay);
    }
  }

function playRhythm(rhythm) {
  playMetronome();
  const isPiano = currentMode === 'piano';
  let delay = 1400;
  if (isPiano && firstPianoPlayback) {
    delay += 1000;
    firstPianoPlayback = false;
  }

  setTimeout(() => {
    let i = 0;
    const interval = setInterval(() => {
      if (i >= rhythm.length) {
        clearInterval(interval);
        return;
      }
      playNote(rhythm[i]);
      i++;
    }, 800);
  }, delay);
}

function analyzeTimingAccuracy(timestamps) {
  if (timestamps.length < 2) return 0;
  let diffs = [];
  for (let i = 1; i < timestamps.length; i++) {
    diffs.push(timestamps[i] - timestamps[i - 1]);
  }
  const avg = diffs.reduce((a, b) => a + b, 0) / diffs.length;
  let variance = diffs.map(x => Math.abs(x - avg));
  const score = Math.max(0, 100 - (variance.reduce((a, b) =>
a + b, 0) / diffs.length));
  return Math.round(score);
}

function compareRhythm(user, correct) {
  if (user.length !== correct.length) return false;
  for (let i = 0; i < user.length; i++) {
    if (user[i] !== correct[i]) return false;
  }
  return true;
}

// Обробник кнопки скидання статистики

```

```
document.getElementById('reset-
stats').addEventListener('click', () => {
  if (confirm('Ви дійсно хочете скинути статистику?')) {
    stats = {
      totalExercises: 0,
      correctAnswers: 0,
      totalAccuracy: 0,
      totalTime: 0
    };
    localStorage.removeItem('stats');
    updateStatistics();
  }
});
```