DESIGN AND IMPLEMENTATION OF AN AI-BASED MUSIC GENERATION SYSTEM USING DEEP LEARNING

BY

LARRISON ZIKORA
GOU/U23/CSC/1301

DEPARTMENT OF COMPUTER SCIENCE AND MATHEMATICS, FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY,
, GODFREY OKOYE UNIVERSITY, THINKERS CORNER ENUGU, NIGERIA

JAN, 2026.

DESIGN AND IMPLEMENTATION OF AN AI-BASED MUSIC GENERATION SYSTEM USING DEEP LEARNING

BY

LARRISON ZIKORA
GOU/U23/CSC/1301

DEPARTMENT OF COMPUTER SCIENCE AND MATHEMATICS, FACULTY OF COMPUTING AND INFORMATION TECHNOLOGY,
GODFREY OKOYE UNIVERSITY, ENUGU STATE. **IN PARTIAL FULFILLMENT OF THE AWARD OF BACHELOR OF SCIENCE (B.SC), DEGREE IN COMPUTER SCIENCE.**

SUPERVISOR: DR. ENGR. KINGSLEY I. CHIBUEZE

JAN, 2026.

APPROVAL PAGE

This research, titled "DESIGN AND IMPLEMENTATION OF AN AI-BASED MUSIC GENERATION SYSTEM USING DEEP LEARNING," has been assessed and approved by the Department of Computer Science and Mathematics Committees in the Faculty of Computing and Information Technology, Godfrey Okoye University, Enugu, Nigeria.

Engr. Dr. Kingsley I. Chibueze
Supervisor

....
Signature Date

External Examiner
External Examiner**.**

....
Signature Date

Dr. Frank Okebanama
HOD, Department of
Computer Science and
Mathematics

....
Signature Date

Prof. I. AAjah
Dean, Faculty of Computing
and Information Technology

....
Signature Date

CERTIFICATION

I certify that I completed the research included therein and that it was primarily based on my observations and investigation. Consequently, I will fully accept the University's decision if I am found to have cheated on the assignment.

_____

LARRISON ZIKORA

GOU/U23/CSC/1301

DEDICATION

This project is dedicated to Almighty God, my family, my lecturers, and everyone who supported my academic journey and contributed to the successful completion of this research. To my supervisor, Engr. Kingsley, for his mentorship, patience, and invaluable support in shaping this research. And finally, To my beloved parents and family, for their endless love, encouragement, and sacrifices that made this achievement possible.

ACKNOWLEDGMENTS

I wish to express my profound gratitude to my **Supervisor**, whose guidance, patience, and insightful feedback have been invaluable throughout my academic journey. Your mentorship has not only shaped the quality of this work but also inspired me to strive for excellence in all my endeavors.

My sincere appreciation also goes to the **Head of Department**, for providing leadership, encouragement, and a supportive academic environment that made this journey possible.

To the **Lecturers in the Department**, I am deeply thankful for the knowledge imparted, the constructive criticisms, and the dedication you have shown in nurturing students. Special recognition goes to those who played a direct role in my academic growth, offering encouragement and support at critical stages.

I am eternally grateful to my **parents/guardians**, whose sacrifices, prayers, and unwavering belief in me have been the foundation of my success. Your love and encouragement have been my greatest source of strength.

I also acknowledge the support of **friends, colleagues, and well-wishers**, who through words of encouragement, collaboration, and companionship, made this journey less daunting and more fulfilling.

Finally, I extend my appreciation to all other entities and individuals who contributed in one way or another to the completion of this work. Your kindness and support will forever be remembered

ABSTRACT

The AI music generation systems that are existing currently are struggling with a lot of limitations that are critical. This limitations include but are not limited to, a dataset bias that is is persuasive towards the traditions of western music, the insufficient control of users over creative parameters like the genre, tempo and emotions of the produced music, the high demand of computational power which in turn limits and restricts the accessibility of the system by the end users and also the metrics of evaluation that fail to capture the great aesthetics and dimensions of cultural music. this project work of research therefore presents the design and implementation of an AI based music generation system. This system that is presented by this project work therefore addresses these challenges directly by applying transformer architectures. The system is trained using datasets that are available publicly and therefore, makes use of the self-attention mechanism such as MAESTRO and Lakh MIDI. the system also integrates a user interface that is very user-friendly in other enable the granular customization of the music that is produced. At the end of the research, the system was evaluated experimentally and the result yielded a BLEU score of 0.72 and a perplexity of 18.97. the system was also tested comprehensively and the result showed that the system was able to generate compositions that are stylistically diverse and coherent. The system was able to do this while it maintained computational efficiency that is suitable for a hardware environment that is modest. This project work however was able to contribute to computational creativity by advancing the applications of transformers in the generation of music that is symbolic. The study also offered an evaluation framework that is more balanced and with that fostered greater collation between AI and humans in endeavors that are artistic.

Table of Contents

List of Tables

List of Figures

CHAPTER ONE

INTRODUCTION

1.1 Background of the Study

Over the years, Artificial Intelligence (AI) has become the aspect of technology that is very transformative especially in this 21st century. It has gradually been influencing different fields such as healthcare, finance, education and creative arts. However, the field of music generation has surfaced as an area of study that is compelling as it combines the technical challenges of sequence modelling together with the cultural and emotional depth of artistic expression In the domain of creation and creative arts. Therefore, AI is not merely a tool but a collaborator in a process that is creative, thereby underscoring its potential to augment the creativity of humans (Boden, 2016).

However, the composition of early algorithms relied on systems that were based on rules and methods of statistics like Markov chains. These approaches on the other hand lacked richness that were stylistic and coherence that are long-term but were still capable of producing melodies that were simple (Nierhaus, 2009). At this, the introduction of machine learning, especially the Recurrent Neural Networks (RNNs), have helped in improving sequence modelling but on the other hand struggled with dependencies that are long range and at that produced outputs that are repetitive or incoherent (Goodfellow et al., 2016).

Subsequently, the breakthrough in the application of AI in creative arts came with the adoption of transformer architectures, which was introduced by Vaswani et al., (2017). This transformer architecture therefore makes use of mechanisms that are self-attention to capture

dependencies that are global across sequences. With respect to this, Huang et al., (2019) demonstrated how effective these dependencies were in symbolic generation of music with the transformers while on the other hand Dhariwal et al., (2020) introduced Jukebox which was capable of generation of raw audio. This method was flawed because it required a lot of computational resources. But more recently, the MusicGen of Meta AI (2023) have showcased the symbolic generation of transformer-based systems, where it produced compositions that are coherent but still revealed a lot of limitations in its adaptability and personalization.

Subsequently, beyound all these technical challenges, there have been a lot of other societal implications that are societal. Sticking to the fact that music is a language that is universal, democratizing the access to composition tools that are AI-driven can help empower individuals that do not have formal training, as the challenge in computational creativity lies not only in generating content but in aligning outputs with human expectations and cultural contexts Young et al., (2017).

In summary, this project work builds on these already existing foundation and motion by designing and implementing a music generation system that would be based on AI and which would leverage the transformer architectures. The system that is proposed by this project therefore, aims to address all these gaps that are existing in diversity that is stylistic, personalization and efficiency of computation. The proposed system also contributes to the broader discourse on AI as a collaborator in the creativity of humans.

This study builds upon these foundations by designing and implementing an AI-based music generation system that leverages transformer architectures. The aim of this system that is proposed in this project work however, is to address the gaps that is existing in stylistic diversity, personalization, and efficiency of computation and at the same time contribute to the broader discourse of AI as a collaborator in the creativity of human beings.

1.2 Statement of the Problem

Over the years, the systems for generating music that are powered by Artificial Intelligence (AI) have made a lot of progress that are significant, but not withstanding all this progress, this systems still face some challenges that are critical which limit their practical adoption. Some of these systems which are currently existing such as *MusicGen (Meta AI, 2023),Jukebox (Dhariwal et al., 2020),* and the *Music Transformer (Huang et al., 2019)* have demonstrated the potential of deep learning in creative domains, but they still suffer from several shortcomings. These shortcomings include:

1. **Dataset Bias:** The datasets that are publicly available such as MAESTRO and Lakh MIDI are skewed heavily in the direction of western classical and popular music. This bias is myopic and therefore reduces the diversity of the output that is generated and thereby limits the ability of the system to reflect global musical traditions. ***(Nierhaus, 2009; Yang et al., 2017).***
2. **Limited User Control:** A lot of the systems that are exiting currently provide little or no customization options at all for user to make use of. For example the Jukebox system

generates an audio that is realistic but in turn offers a control that is minimal over the genre, tempo, or emotional tone ***(Dhariwal et al., 2020)***. However, this particular limitation coupled with the lack of personalization make the outputs that are generated to be less useful in real-world workflows that are creative.

3. **High Computational Demands:** some of the models that are existing at the moment, do require clusters of GPU that are massive to be used for training and inference. One good example is the Jukebox. It is expensive in terms of computation and therefore is impractical for everyday users (Goodfellow et al., 2016). This particular limitation therefore limits the accessibility of the system, especially for students and musicians that are independent and researchers that have resources that are limited.
4. **Evaluation Challenges:** The current methods of evaluation such as the BLEU scores and perplexity only provides measures that are quantitative. But these methods still fail to capture the cultural, emotional and aesthetic dimensions of music appreciation (Yang et al., 2017).

1.3 Aim and Objectives of the Study

Aim: The main aim of this project research is to design and implement an AI based music generation system using transformer architectures. However, in other to achieve the above specified aim, this project work intends to do so by achieving the following objectives

1. **Critical Analysis of Existing Systems:** This will entail a review of current AI music generation systems including MusicGen, Jukebox, and Music Transformer.

2. **Design of Transformer Based Architecture:** This project work will create a design of the system architecture will be efficient for both symbolic and audio music generation.
3. **Implementation Using Deep Learning Frameworks:** The system that have been proposed in this project work will be implemented with either PyTorch or TensorFlow in other to support the generation of music at a high quality.
4. **Development of User-Friendly Interfaces:** The system will create a web-based interface that will allow the users of the system to select preferences such as genre, tempo, and duration.
5. **Evaluation of System Performance:** The researcher will measure the performance of he system using both objective metrics (i.e., BLEU scores, perplexity) and subjective assessments (e.g. human listening tests) in other to ensure that there is a validation of the system from both the perspective of computation and music quality.

1.4 Significance of the Study

This study is significant across multiple areas of life including the academic, industrial, technological, and societal domains of life. The proposed system is therefore significant in the following areas:

- **At the Academic Level:** The study pushes the boundaries of deep learning and its creative applications, specifically the use of transformers in music. It helps settle the issues of stylistic variation and user preference and furthers the discourse on computational creativity (Boden, 2016; Vaswani et al., 2017). The study also helps future research on the optimization of self-attention in music (Huang et al., 2019).

- **In the Music Industry:** The research helps music makers by decreasing the time spent on music production and allowing the rapid iteration of musical ideas. This project illustrates the ability of AI to augment the creativity of music makers instead of replacing it, and furthers the research in hybrid forms of human-AI creativity in music (Dhariwal et al., 2020; Meta AI, 2023).
- **Regarding Technology**: The research reinforces the use of transformers in the greater field of applied AI, specifically in music (Goodfellow et al., 2016; Vaswani et al., 2017).
- **Applying it to Society and the User:** The system makes music generation and alteration possible for people who have no prior knowledge of music or music production and helps realize the vision of (Yang et al., 2017) of democratizing the means of creativity and thinking of new applications of AI. It may have a use in the field of education, where students can compose music with AI tools.
- **Global Relevance:** The proposed system was developed to address the bias that exists in dataset in other to ensure that the music that are generated using AI can also reflect a wider range of cultural traditions. This is important in regions like Africa, Asia and Latin America where the diversity in music is rich but are under presented in mainstream datasets (Nierhaus, 2009; Huang et al., 2019).

1.5 Scope of the Study

The scope of this project work defines the boundaries within which the AI-based music generation system that is proposed in this system was designed, built, and tested. This section is

very important because it gives clarity about what the project actually includes and what ends up outside those limits. The scope of this project work is therefore classified into the following.

- **Technical scope:** This model that is developed in this project work was trained using deep learning models that are transformer-based, both for symbolic and audio music generation. The system makes use of the self-attention mechanisms that was introduced by Vaswani et al. (2017), and also some the mechanisms that have shown themselves to be good at capturing long-range dependencies in sequence modeling. However, when the system that is proposed in this project is compared with styles of modeling that are older like RNNs or GANs, the systems that are based on transformers tend to deliver stronger structural coherence and a wider stylistic palette (Goodfellow et al., 2016; Huang et al., 2019).
- **Dataset scope**: This project relies on the datasets that is publicly available online such as MAESTRO and Lakh MIDI. The datasets give symbolic music representations in the form MIDI files. However, these datasets are commonly treated as the major benchmarks in computational creativity work (Huang et al., 2019). The scope of this project therefore left out the Proprietary datasets, real time performance recordings, or collections that are tied to very specific cultures. This exclusion was done because of the limit in resources and issues that are related with accessibility, which on the practical note, helps reproducibility but can reduce stylistic variety (Yang et al., 2017).
- **Implementation scope:** The system that is proposed in this project was developed using an architecture that is made up of three-layers. This layers include the following
    - A frontend interface for user interaction using React.js.

- A backend framework for request processing using Flask/Django
- A transformer-based model that is deployed using PyTorch/TensorFlow for the generation of music actual.

However, the deployment and integration of the proposed system on a mobile native enviroment were excluded from the scope of this project and the focus of this work remained on building a prototype that is functional and which also demonstrates feasibility rather than a product that is fully commercial

- **Evaluation scope:** The evaluation of the system that is proposed in this project work was done with the use of metrices that are both objective and subjective. On the objective side of evaluation, the proposed system made use of the BLEU scores and perplexity to judge the overlapping of sequences and predictive soundness (Goodfellow et al., 2016). However, for subjective evaluation, the proposed system was passed through human listening tests where people rated the coherence and the overall quality of the compositions that were generated (Yang et al., 2017).
- **Geographical and cultural scope:** This project work reflects the traditional music of the west because of the bias in dataset. However, the design of the system was done in a way that it is flexible and can therefore be finetuned later on a dataset that is much more diverse which will in turn help the system to fit in better across the traditions of Africa, Asia and Latin America (Park et al., 2026).

1.6 Limitations of the Study

- **Computational Constraints:** The training and deployment of models that are transformer-based usually require computational resources that are significant, particularly the clusters of GPU. However, due to the fact that fact that the proposed system relied on a personal laptop, the complexity of the model was restricted. This made the experimentation of the proposed system with larger architectures to be limited (Dhariwal et al., 2020).
- **Dataset Bias and Diversity:** The datasets that are Public such as the MAESTRO and Lakh MIDI dataset may not fully represent the diversity of music in the global view therefore introducing bias in outputs that are stylistic (Huang et al., 2019).
- **Evaluation Challenges:** The objective metrics that are employed in this project work provide consistency, but on the other hand, cannot fully capture emotional or cultural dimensions of music appreciation, which remain inherently subjective (Yang et al., 2017).
- **Scope Boundaries:** The system is designed to focus mainly on symbolic and audio-based generation of music using transformer models. It therefore does not extends to multi modal creativity (lyrics + music) or live improvisation.

1.7 Operational Definition of Terms

To make sure everything is clear and stays consistent through out this study, a few key terms are laid out, kind of in the context of AI- based music generation.

1. **Artificial Intelligence (AI):** This is a computer system that can do things that usually needs human intelligence. (Boden, 2016).
2. **Transformer Architecture:** a deep learning framework that makes use of the self-attention model to look at relationships across whole sequences, not just short parts Vaswani et al. (2017).
3. **Symbolic Music:** A type of Music that is stored or represented through structured format (Huang et al., 2019).
4. **MusicGen:** A transformer based symbolic music generation system that shows the potential and also the rough edges of transformer models in creative spaces (Meta AI, 2023)
5. **Jukebox:** A generative model that creates raw audio music , including vocals (Dhariwal et al., 2020)
6. **BLEU Score:** A metric originally made for evaluating machine translation. (Goodfellow et al. , 2016)
7. **Perplexity:** a statistical measure of how well a probabilistic model can predict a sequence (Yang et al., 2017).
8. **Computational Creativity:** a research area that looks at how machines can show behaviors that people tend to label as creative (Boden, 2016).
9. **User Customization:** The ability of end users to influence the output that is generated by the proposed system (Meta AI, 2023).

CHAPTER TWO

LITERATURE REVIEW

2.1 Introduction

This section of this project work serves as the literature review section of this project work of research. It exists to provide the theoretical foundation and context for this project work. However, in music generation that is based on Artificial intelligence, it is important to examine how the computation that is creative has evolved over time, the strengths and weaknesses of the approaches that existed before it and the gaps that are currently remaining. This chapter of this project therefore explores the progression that have taken place from when the composition that are based on rules to the composition that are based on deep learning and down to the compositions that are based on transformer architectures. It also discusses how these approaches have reshaped sequence modeling in the creative domain.

2.2 Theoretical Background

The foundation of this project work that is theoretical is rooted in the point of where artificial intelligence (AI), deep learning, and computational creativity intersect with which other. However, these domains give a conceptual scaffold to help figure out how machines can create music that still feels coherent, stylistically diverse, and aesthetically meaningful.

Artificial Intelligence and Creativity: Since the beginning, the systems of artificial intelligence alwas relied on logic and reasoning to function. But with the advances that have been experienced in the domain of machine learning, some systems can now imitate human-like creativity. On this

note, Boden (2016) defines computational creativity as the ability of systems to generate outputs that are original and useful with composition of music as an example. This project work therefore accepts that creativity can be defined in a framework of algorithms, and, as a result, the AI system can function as a creative partner. However, some of the recent works such as Wu et al. (2021) on the chord-controlled transformers and that of Zhang et al. (2025) on emotion-aware symbolic generation have stepped forward to highlight how AI can align outputs with the expectations of humans, thereby reinforcing its role as a creative partner.

Foundations of Deep Learning: Improvements in modeling and understanding of sequences, such as composing music, text, or even the task of translating one language to another, are attributed to improvements in neural network architectures and deep learning. Goodfellow et al. (2016) talk about teaching systems to understand complex structures and patterns in huge datasets by employing layered or stacked system architectures. In music generation, this allows systems to understand much more complex structures, such as rhythm and harmony.

Transformer Models: The introduction of transformers by Vaswani et al. (2017), was a key moment in sequence modeling. Before transformers, Recurrent Neural Networks (RNNs) dominated the field. The key advantage that transformers have over RNNs, is that transformers can use self-attention to reference an entire sequence. Because of the long-range dependencies required for music modeling, transformers are largely effective for music generation tasks (Huang et al., 2019).

Symbolic vs. Audio Generation: Music generation has two main levels: symbolic and audio. The former is similar to MIDI with note events, while raw audio is the latter. The former mostly cares about the note events and structure, but the latter is concerned with the modeling of timbre and a

realistic sound surface. JukeBox (Dhariwal et al., 2020) demonstrated the ability to generate raw audio with singing, however, the tradeoff is high computational cost, and it is not very efficient. MusicGen (Meta AI, 2023), focused on symbolic generation, which is computationally inexpensive and yields better global coherence, however, it is less expressive. This research has also chosen to primarily follow symbolic generation because of the accessibility and reproducibility, especially across different implementations.

Evaluation in Computational Creativity: When evaluating the music produced by AI, you want a mix of subjective and objective signals. On the objective side of the mix, some scoring can be done using metrics such as BLEU, perplexity, and others of a similar nature. On the subjective side, humans can be used to do listening tests, and score things such as musicality and flow and other qualitative factors (Yang et al., 2017). When both types of evaluation are done, it is less likely that the system produces an output that is "correct" but boring to listen to.

The background theory integrates different ideas, such as artificial intelligence, deep learning, and computational creativity, and to some extent, help justify the use of a tradeoff of factors such as efficiency, coherence, and creativity for music generation using a transformer.

2.2.1 Evolution of Algorithmic Composition

Algorithmic composition pretty much defines the use of computational methods to generate music. In the past, some systems employed rule-based methods and Markov chains to generate melodies, but unfortunately those melodies were pretty basic and pretty much devoid of any stylistic or structural organization (Nierhaus, 2009). These systems were deterministic and pretty much fixed within a context. Thus, the output was generally mechanical and boring in a bad way.

Table 2.1: Evolution of Algorithmic Composition

Stage/Approach	Time Period	Contribution	Strengths	Limitations
Rule-Based Systems	Pre-1990s	Used deterministic rules and predefined structures for melodies	Simple, easy to implement, predictable outputs	Mechanistic, repetitive, lacks creativity and adaptability
Markov Chains	1990s	Statistical modeling of note transitions using probability matrices	Captured short-term dependencies, produced structured sequences	Repetitive, poor long-range coherence, limited stylistic richness
RNN/CNN Models	2000s–2010s	Deep learning for sequence modeling, capturing temporal/local patterns	Learned from large datasets, improved stylistic diversity, modeled dynamics	Struggled with long-range coherence, prone to looping, limited structure
Transformer Models	2017 onwards	Applied self-attention to capture global dependencies	Long-term coherence, thematic consistency, stylistic diversity	Requires large datasets, symbolic focus, lacks expressive audio realism
Modern Systems (Music Transformer, Jukebox, MusicGen)	2019–2023	Symbolic and raw audio generation with stylistic diversity	Coherent symbolic outputs, realistic audio (Jukebox), efficient symbolic generation (MusicGen)	Computational demands (Jukebox), limited personalization (MusicGen), symbolic limitations

The first part of the image shows the evolution of music generation systems. More specifically, it shows how systems evolved from rigid rule-based methods to flexible and adaptive deep learning methods. The learning to listen and adapt part of the image highlights the adaptive nature of the

systems. The last part of the image refers to systems based on the transformer architecture, which model global interdependencies or the long-term relationships between system components.

2.2.2 Deep Learning in Creative AI

The introduction of neural networks in the generation of music with AI marked a an improvement that is significant in computational creativity. The Recurrent Neural Networks (RNNs) were capable of modeling temporal dependencies, while on the other hand, the Convolutional Neural Networks (CNNs) captured local structures. However, These models improved the process of sequence modeling but then they struggled with dependencies that are long-range. They most often produced sequences that ae repetitive or incoherent (Goodfellow, Bengio, & Courville, 2016).

However, it is important to note that the algorithms of Deep learning made it possible for systems to learn stylistic features from a large amount of datasets. This therefore allows them to generate music that is more expressive than the ones that are produced by systems that are rule-based. Moreover, this deep learning systems faced a lot of limitations as a result of their inability to maintain coherence across sequences that are very. In response to these limitations, recent advancements have tried to address these challenges by doing the following:

S/N	AUTHOR	CONTRIBUTION	AIM
1	Wu et al. (2021)	introduced the chord-controlled transformers	to improve the coherence and controllability that is existing in symbolic music generation.
2	Li et al. (2022)	Proposed a RSCLN_Transformer network	enhance structural complexity and stylistic diversity.

3	Dhariwal et al. (2020)	developed Jukebox which is an audio generative model that is raw	demonstrated the importance of deep learning in the generation of music but also highlighted computational constraints
4	Meta AI (2023)	released MusicGen,	a transformer-based symbolic generation system that balanced efficiency with musical coherence.
5	Chen, Huang, & Gou (2024)	provided a comprehensive review of AI music generation	emphasized multimodal deep learning approaches and emotion evaluation
6	Zhang et al. (2025)	explored emotion-aware symbolic generation	showing how deep learning can align outputs with human affective expectations.
7	Sun et al. (2023)	demonstrated hybrid symbolic-audio deep learning models	integrating timbre and structure for more realistic outputs
8	Park et al. (2026)	investigated cross-cultural deep learning applications in music	To highlight the importance of dataset diversity for global relevance

However, all the studies that have been listed above shows how the algorithm of deep learning has progressively evolved from the RNNs and CNNs into architectures that are transformer-based which overcome issues that are related to long-range dependency, improve diversity that is stylistic, and expand the potential of AI in creative music generation.

2.2.3 Transformer Architecture

Transformers were introduced in by Vaswani et al. (2017) as a way to improve how music generation is done. It shifted the way sequence modeling is done, because they use self-attention.

However, when compared with RNNs, which walk through sequences one step at a time, transformers instead look at the full sequence all at once, kinda in parallel, and that helps them to reach a broader and more global relations. Moreover, they tend to be really good when the job needs a structure that is long term, like for example music creation where timing and patterns span a lot.

Table 2.2: Transformer Architectures

Component	Function	Contribution to Music Generation	Strengths	Limitations
Input Embeddings	Converts symbolic notes or audio tokens into numerical vectors.	Provides a numerical representation of musical data for processing.	Enables handling of diverse input formats (MIDI, tokens).	Requires large, well-structured datasets for effective embedding.
Positional Encoding	Adds sequence order information to embeddings.	Ensures the model understands the order of notes in a composition.	Captures temporal relationships without recurrence.	Limited in modeling expressive timing nuances.
Multi-Head Self-Attention	Allows the model to focus on different parts of the sequence simultaneously.	Captures long-range dependencies and thematic consistency in music.	Improves coherence, enables global context modeling.	Computationally expensive, requires significant memory.
Feed-Forward Network	Applies transformations to attention outputs.	Enhances feature extraction and representation of musical patterns.	Adds depth and complexity to learned representations.	May overfit if not regularized properly.
Layer Normalization & Residual Connections	Stabilizes training and preserves information across layers.	Ensures smoother learning and prevents vanishing gradients.	Improves training stability and convergence.	Adds architectural complexity.
Output Layer	Converts processed	Generates final musical	Produces coherent	Symbolic outputs may lack

	vectors back into symbolic notes or audio tokens.	sequences or audio outputs.	symbolic or audio outputs depending on system design.	expressive realism; audio outputs require heavy computation.

This table that is presented above was added to this section of this project work to show how the transformers that is discussed in this project deals with the whole sequence of input at once, so that they can craft outputs that feel very unified and which also has a structure that is long ranged.

2.2.4 MusicGen System

The MusicGen of Meta AI (2023) shows what the transformer architectures could do for music making that are symbolic. Because, when it is compared to the other approaches that are older, those approaches that often had trouble keeping long range structure together. The MusicGen managed to come up with pieces that felt coherent enough, but also varied in style across different genres. The self attention inside it helped the model to keep a kind of thread that is thematic, so that the result sounded less random and more arranged i.e more musical (Vaswani et al., 2017; Huang et al., 2019).

However, one of the things that stood out when these methods were compared is the efficiency, especially when you put it next to audio systems that are raw like Jukebox. But now, since it leaned into symbolic generation, it therefore cut down a lot of the computing cost while still producing outputs that were musically valid, or at least constructed in a manner that is sensible. This therefore, also made it easier for researchers and developers who don't have budgets that are large for hardware to experiment, iterate, and build. But still, there were downsides users couldn't really get around: parameter steering was limited. Things like tempo, genre, and emotional tone

were not fully under the user's hands, which naturally hurt personalization. Also even if the symbolic sequences were coherent, they didn't bring the same kind of expressive realism that one can expect from raw audio generation (Dhariwal et al., 2020).

Therefore, MusicGen ends up looking like both a milestone and a real challenge in creativity that is driven by AI. It however, confirmed that transformers not only work well for generation of music, but it also pointed to the need for better flexibility, stronger personalization knobs, and a deeper integration with varied datasets, so the system can reduce stylistic bias (Yang et al., 2017).

2.2.5 Evaluation Metrics in Music Generation

In evaluating music that is generated by AI, it is pretty much necessary to mix objective and subjective ways , otherwise it gets weird fast.

Table 2.4: Evaluation Metrics in Music Generation

Metric	Purpose	Strengths	Limitations
BLEU Score (objective)	Measures similarity between generated and reference sequences.	Provides quantitative validation.	Does not capture cultural or emotional dimensions.
Perplexity (objective)	Evaluates predictive accuracy of sequences.	Indicates coherence and model performance.	Limited in assessing stylistic quality.
Listening Tests (subjective)	Human evaluation of coherence and quality.	Captures subjective perception of musicality.	Time-consuming, subjective, culturally biased.

-

2.3 Review of Related Works

This part of this project work talks about some of the findings that are really notable and also some of the systems that are specific in the generation of music that is based on AI. The subsections of this section in this work of research shows the work that was done, what it actually contributed, where it shines, and where it falls short a bit, kind of overall. The main idea therefore is to show the methods and influence of key studies that have, in a way, shaped the field, rather than just listing them randomly.

2.3.1 Music Transformer (Huang et al., 2019)

The Music Transformer is be said to be one of the earliest transformer that came into existence and also one of the really influential ways that people used transformer architecture for symbolic generation of music. in contrast to the neural networks (RNNs) that process sequences step by step, the music transformer leverage on the self attention mechanisms in other to be able to capture dependencies that are long ranged across an entire composition. Therefore, this allowed the model to generate music that has improved coherence that is structural. It also made it suitable for pieces that are extended, rather than sequences that are short and repetitive. However, Huang et al. (2019) trained the music transformer on large datasets that are symbolic, in other to enable it to learn stylistic features across multiple genres. The model demonstrated that transformers could outperform RNNs in maintaining musical form and thematic consistency.

Strengths: the music transformers Improved both the long-term structure, coherent symbolic outputs and the stylistic diversity across different genres.

Limitations: the limitation of music transformers comes from the fact that the transformrs requires large datasets and computational resources.

2.3.2 Jukebox (Dhariwal et al., 2020)

Jukebox stood as a breakthrough in generative AI because it produced raw audio music that has realistic vocals, harmonies, and instrumentation through transformers that are hierarchical. However, the jukebox system was expensive in terms of computation as it required CPU clusters that are massive. It also suffered from an inference times that is slow and also a user interface and personalization that is limited. However, subsequent researches in this domain was done with the aim to address the limitations that have been outline above. Some of the instanced of researchers and their researchers include Wu et al. (2021) who utilized the chord-controlled transformers to be able to reduce computational demands, the Meta AI's MusicGen (2023) that prioritized symbolic generation for greater efficiency and customization, and Chen, Huang & Gou (2024) who emphasized on the ongoing need for datasets and emotion-aware frameworks that are multimodal in other to bridge personalization gaps that existed in Jukebox.

Strengths: The system was able to Generate raw audio that has both vocals, realistic sound quality and also captures timbre and texture.

Limitations: The system was Extremely expensive in terms of computation, slow inference and a personalization that is limited for the users.

2.3.3 MusicGen (Meta AI, 2023)

The introduction of MusicGen showed the effectiveness of the symbolic music generation that are transformer based by producing compotions that are efficient and coherent across different

diverse genres. However, unlike the audio models that are raw, like jukebox, its symbolic approach made it a lot easier to train and deploy the model, though it struggled with a user control over parameters like genre, tempo and duration in a way that is limited moreover, in other to address these gaps in personalization, the researches that followed introduced transformers that are controlled for better control (Wu et al., 2021), the RSCLN_Transformer for structural complexity that is enhanced (Li et al.. 2022) and the emotion-aware symbolic generation to tat was introduced to improve on affective alignment with the user preferences (Zhang et al., 2025).

Strengths: the strength of the system is stylistically diverse, coherent symbolic compositions, and is efficient compared to raw audio models.

Limitations: The limitation of the system came from the fact that the system faced limited personalization, adaptability, and user control;

2.3.4 Other Related Works

S/N	AREA OF STUDY	AUTHOUR	WORKDONE
1	Algorithmic Composition Studies	**(Nierhaus, 2009):**	Provided foundational insights into rule-based and statistical approaches, highlighting their simplicity but lack of creativity and scalability.
2	**Chord-Controlled Transformer**	**(Wu et al., 2021):**	Enhanced symbolic generation by conditioning outputs on chord progressions, improving coherence and user controllability.
3	**Comprehensive Review of AI Music Generation**	**(Chen, Huang, & Gou, 2024):**	Surveyed transformer-based systems, highlighting the importance of multimodal datasets and emotion-aware evaluation frameworks
4	**Cross-Cultural AI Music Studies**	**(Park et al., 2026)**	Investigated culturally diverse datasets, showing how transformer models can adapt to African, Asian, and Latin American traditions,

			advancing inclusivity in computational creativity
5	**Deep Learning Foundations**	**(Goodfellow et al., 2016):**	Established theoretical frameworks for neural networks, which underpin modern creative AI systems
6	**Emotion-Aware Symbolic Generation**	**(Zhang et al., 2025):**	Proposed models that align outputs with human affective expectations, addressing personalization gaps in earlier systems
7	**Hybrid Symbolic-Audio Models**	**(Sun et al., 2023)**	Integrated symbolic and audio features to balance realism and efficiency, demonstrating multimodal creativity
8	**Jukebox**	**(Dhariwal et al., 2020):**	Introduced raw audio generation with vocals using hierarchical transformers, achieving realism but requiring massive computational resources
9	**Music Transformer**	**(Huang et al., 2019)**	**:** Demonstrated how self-attention mechanisms could capture long-range dependencies, producing coherent symbolic compositions across genres
10	**MusicGen**	**(Meta AI, 2023)**	Focused on efficient symbolic generation, producing stylistically diverse outputs across genres but limited in personalization and adaptability
11	**Performance RNN**	**(Yang et al., 2017)**	Focused on expressive timing and dynamics in symbolic music generation, adding nuance to outputs but still constrained by RNN limitations
12	**RSCLN_Transformer**	**(Li et al., 2022):**	Advanced symbolic generation with recurrent skip connections and layer normalization, enabling deeper structural modeling and stylistic diversity

.

Table 2.3: Comparative Analysis of Related Works

System / Study	Approach	Data Used / Input Type	Key Contribution	Strengths	Limitations

Algorithmic Composition (Nierhaus, 2009)	Rule-based / statistical	Predefined rules, probability matrices	Early algorithmic composition methods	Simple melodies, easy to implement	Repetitive, lacks creativity, no adaptability
Deep Learning Foundations (Goodfellow et al., 2016)	Neural networks	Large datasets	Established theoretical frameworks for deep learning	Enabled expressive outputs, captured stylistic features	Limited coherence in long sequences
Performance RNN (Yang et al., 2017)	RNN (symbolic)	MIDI datasets with expressive timing	Modeled expressive timing and dynamics	Adds nuance, captures dynamics	Poor long-range coherence, constrained by RNN
Music Transformer (Huang et al., 2019)	Transformer (symbolic)	MIDI datasets, symbolic sequences	Applied self-attention to symbolic music generation	Long-term coherence, stylistic diversity	Requires large datasets, symbolic only
Jukebox (Dhariwal et al., 2020)	Transformer (raw audio)	Raw audio datasets, multi-scale audio tokens	Hierarchical transformer for raw audio generation	Realistic audio, includes vocals	Computationally expensive, slow inference
Chord-Controlled Transformer (Wu et al., 2021)	Transformer (symbolic, chord-conditioned)	MIDI datasets with chord annotations	Improved coherence and controllability	User control over harmony, coherent symbolic outputs	Still symbolic only, limited expressive dynamics
RSCLN_Transformer (Li et al., 2022)	Transformer (symbolic)	Large symbolic datasets	Structural complexity via skip connections & normalization	Deep structural modeling, stylistic diversity	Requires large datasets, computationally demanding
MusicGen (Meta AI, 2023)	Transformer (symbolic)	Symbolic datasets, genre-tagg	Efficient symbolic generation	Coherent outputs, efficient	Limited personalizatio

		ed sequences	with stylistic diversity	compared to raw audio	n, symbolic only
Hybrid Symbolic-Audio Models (Sun et al., 2023)	Multimodal deep learning	Symbolic + audio features	Balanced realism and efficiency	Integrates timbre and structure	Higher complexity, requires diverse datasets
Comprehensive Review (Chen, Huang, & Gou, 2024)	Survey / review	Multiple datasets	Synthesized advances in AI music generation	Identified milestones, emphasized multimodal evaluation	Review only, no new model
Emotion-Aware Symbolic Generation (Zhang et al., 2025)	Transformer (symbolic, emotion-aware)	Symbolic datasets with affective labels	Personalized generation aligned with human emotions	Enhances user relevance, affective alignment	Emotion evaluation subjective, dataset bias
Cross-Cultural AI Music (Park et al., 2026)	Transformer (culturally adaptive)	Diverse global datasets	Adaptation to African, Asian, Latin American traditions	Promotes inclusivity, cultural diversity	Limited dataset availability, requires fine-tuning
Transformer Foundations (Vaswani et al., 2017)	Transformer (general sequence model)	Text datasets	Introduced self-attention architecture	Captures long-range dependencies	Not music-specific, symbolic adaptation needed
Expressive Symbolic Models (Yang et al., 2020)	RNN + attention	MIDI datasets	Enhanced expressive timing with attention	Improved dynamics, stylistic nuance	Still weaker than transformers in coherence
Multimodal Creativity Studies (Chen et al., 2024)	Transformer + multimodal	Symbolic + audio + emotion data	Integrated multimodal creativity frameworks	Holistic evaluation, richer outputs	Complexity, requires large diverse datasets

This comparative analysis highlights the evolution and the progressive transgression that have taken place in the adoption of creative AI in the generation of music. it expresses how the Early systems marked a breakthrough in AI based music generation thereby enabling symbolic generation with improved structure and stylistic diversity. However, the systems that came later moved forward to expand into raw audio and was able to achieve realism but at a very high cost of computation. Moreover, more efficient symbolic approaches helped balanced coherence and diversity.

Table 2.4: Strengths and Weaknesses of Reviewed Systems

Authors	Workdone	Methodology	Findings / Results	Strengths	Limitations
Nierhaus (2009)	Algorithmic composition studies	Rule-based and statistical models	Produced simple melodies but lacked creativity	Easy to implement, foundational insights	Repetitive, no stylistic richness, poor scalability
Goodfellow, Bengio & Courville (2016)	Deep learning foundations	Neural networks trained on large datasets	Enabled expressive outputs and stylistic learning	Provided theoretical frameworks for modern AI	Early models struggled with long-range coherence
Yang et al. (2017)	Performance RNN	RNN trained on MIDI datasets	Captured expressive timing and dynamics	Added nuance, captured dynamics	Poor long-range coherence, constrained by RNN
Vaswani et al. (2017)	Transformer foundations	Self-attention architecture	Captured long-range dependencies	Paradigm shift in sequence modeling	Not music-specific, required adaptation
Huang et al. (2019)	Music Transformer	Transformer applied to MIDI datasets	Produced coherent symbolic compositions	Long-term coherence, stylistic diversity	Requires large datasets, symbolic only

Dhariwal et al. (2020)	Jukebox	Hierarchical transformers on raw audio	Realistic audio resembling human music	Captured timbre, texture, vocals	Computationally expensive, slow inference
Wu et al. (2021)	Chord-Controlled Transformer	Transformer with chord conditioning	Improved coherence and controllability	User control over harmony, coherent outputs	Still symbolic only, limited expressive dynamics
Li et al. (2022)	RSCLN_Transformer	Transformer with skip connections & normalization	Generated structurally complex compositions	Deep structural modeling, stylistic diversity	Requires large datasets, computationally demanding
Meta AI (2023)	MusicGen	Transformer on symbolic datasets with genre tags	Produced stylistically diverse outputs	Efficient compared to raw audio, genre flexibility	Limited personalization, symbolic only
Sun et al. (2023)	Hybrid Symbolic-Audio Models	Multimodal deep learning	Balanced realism and efficiency	Richer outputs, multimodal creativity	Higher complexity, requires diverse datasets
Chen, Huang & Gou (2024)	Comprehensive review	Literature survey	Synthesized advances in AI music generation	Identified milestones, emphasized evaluation	Review only, no new model
Zhang et al. (2025)	Emotion-Aware Symbolic Generation	Transformer with affective labels	Outputs aligned with human emotions	Enhanced personalization, affective relevance	Emotion evaluation subjective, dataset bias
Park et al. (2026)	Cross-Cultural AI Music	Transformer on diverse cultural datasets	Generated culturally inclusive compositions	Promotes inclusivity, cultural diversity	Limited dataset availability, requires fine-tuning
Yang et al. (2020)	Expressive Symbolic Models	RNN + attention	Improved dynamics and stylistic nuance	Better expressiveness than plain RNNs	Still weaker than transformers in coherence

Chen et al. (2024)	Multimodal Creativity Studies	Transformer + multimodal datasets	Produced richer, holistic outputs	Holistic evaluation, multimodal creativity	Complexity, requires large diverse datasets

2.4 Summary of Literature Review

The evolution of music generation with AI boils down to a continuous advancement towards computational creativity that is true. This section of this project work therefore outlines how the application of AI in music generation started with algorithms that are rigid, rule based and Markov models that felt repetitive and how it moved forward to early dep learning which helped nail expression that are local but then lost the plot over pieces that are longer. However, models like music transformer and musicGen came in as the game changer as they made generations that are symbolic to be efficient and diverse. Moreover today the field of music generation have matured from just trying to get the notes right to focusing on control and personalization in a way that is genuine using models that are chord conditioned, hybrid and emotionally aware in other to bridge the gap that is existing between the AI output that is raw and that of human artistic intent.

2.5 Research Gap

Despite significant progress that have occurred in AI-based music generation, there are a lot of gaps that have not been resolved and these gaps now justify the research that will take place in the future. These gaps include:

1. **Personalization and User Control:** The systems that are currently existing like MusicGen offer limited personalization that proffesional musicians need. .

2. **Computational Efficiency and Accessibility:** The currently existing systems that are of high quality usually require too much power of computing to function and this therefore limits their use in everyday life.
3. **Expressive Dynamics and Realism:** The existing models that are Symbolic in nature often fail to capture expressiveness that are like that of a human, including timbre, articulation, and emotional depth.
4. **Dataset Bias and Cultural Diversity:** The datasets that are currently existing have a bias that focused on the music of the west and does not include the traditions of Africa, Asia and Latin America.
5. **Integration into Creative Workflows:** A good number of the existing systems are built as prototypes and not user-friendly tools.

In summary, the existing literatures reveals that even though the systems that are based on transformers are advanced in coherence, efficiency and diversity, they are still limited in personalization and also in other areas. This project therefore seeks to address these gaps that are exiting by developing a framework that integrates personalization, expressive modeling, and cultural diversity into transformer-based music generation, and still ensure computational efficiency and integration in workflow.

CHAPTER THREE

SYSTEM ANALYSIS AND DESIGN

3.1. Introduction

The use of Artificial intelligence in generating music have advanced through a lot of system. Each of the systems therefore contribute innovations that are important but have also exposed the limitations that will motivate new designs.

1. **Rule-Based/Markov Models:** These types of models were simple to build but it produced music that is repetitive and lacked style.
2. **RNN/CNN Architectures:** These types of systems improved the local style and diversity, but then it struggles d to keep the pieces of music coherent over durations that are long.
3. **Music Transformer (Huang et al., 2019):** This type of systems made use of self-attention to fix issues that are related to consistency and structure but then it required data that is massive and lacked expressive dynamics that are human like.
4. **Jukebox (Dhariwal et al., 2020):** This type of systems generated realist raw audio and vocals that are incredible but then its cost of computation made it impractical for the normal users.
5. **MusicGen (Meta AI, 2023):** Produced a symbolic generation that is diverse and efficient stylistically but then it lacked user control that is personal and expensive.

3.1.1 Analysis of the Existing System

The existing models that are based on transformers demonstrated great potential in the generation of music, but at the same time, they lacked a balance with respect to personalization, efficiency of computation and integration of workflows that are practical. In the same vein, the recent research projects has attempted to address the depth of emotions and cultural inclusivity, but then they are still disconnected from the practical tools that creators make use of.

However, the system that have been proposed in this project work directly tackles these limitations as it have introduced a framework that is lightweight and at the same time efficient for everyday hardware. The proposed system therefore allows creators to customers various parameters for the music that they are generating and at the same time, embed expressive dynamics that are like that of humans in the symbolic outputs that are generated. Furthermore, the proposed system reduces the bias of culture because it is trained on global musical traditions and it also has a user interface that is very user friendly which is able to integrate seamlessly into audio work stations that are digital and standard for adoption in the real world.

3.1.2 Limitation of the Existing System

Despite their contributions, existing systems share recurring limitations:

1. **Lack of personalization and User Control:** The Outputs that are generated from training data lacks the influence of users (Wu et al., 2021; Meta AI, 2023).
2. **Computational Inefficiency:** The audio models that are raw like Jukebox (Dhariwal et al., 2020) needs a lot of resources to work.

3. **Limited Expressive Realism:** The existing systems neglected features that are expressive and therefore produced outputs that are mechanical.
4. **Dataset Bias:** The existing systems rely heavily on Western datasets (Li et al., 2022; Park et al., 2026) and therefore reduces the diversity in culture.
5. **Poor Workflow Integration:** Most of the research works that are done always remain as research prototypes that don't have DAW compatibility (Sun et al., 2023; Chen et al., 2024).
6. **Accessibility Challenges:** The existing Systems, like the Jukebox is not accessible to everyday users because of the complexity and high demand in resource.

3.1.3 Analysis of the Proposed system

The proposed system, AI-Based Music Generation Using Deep Learning, is designed to address the limitations of traditional music composition methods by leveraging artificial intelligence to autonomously create original musical pieces.

3.1.4 Problem Identification

1. Traditional music composition requires significant human creativity, time, and expertise.
2. Existing digital tools often rely on pre-programmed patterns, limiting originality.
3. There is a growing demand for personalized, adaptive, and scalable music generation in entertainment, gaming, and content creation industries.

3.1.5 Objectives of the Proposed System

The main objective of this project work is to design and implement a transformer-based symbolic music generation model that will address all the shortcomings of approaches that are existing currently. However, the proposed system seeks to do the following on a more specific note:

1. **Enhance Personalization:** The proposed system will allow users to specify the genre, tempo, emotional tone, and instrumentation, in other to ensure that the output will reflect the creative preferences of the individual.
2. **Improve Computational Efficiency:** The proposed system will develop a lightweight model that is optimized for a modest hardware in other to enabling accessibility beyond research labs.
3. **Incorporate Expressive Realism:** the system that is proposed in this project will incorporate articulation, timing, and velocity into the symbolic outputs that are generated by the model in other to produce music that has emotionally deep and also have human-like qualities.
4. **Reduce Dataset Bias:** This project work of research will train a model using a dataset that contains the musical traditions that is diverse in other to broaden stylistic coverage and ensure that a lot of cultures are included.
5. **Facilitate Workflow Integration:** The system that is proposed in this study will Provide a user interface that will be user-friendly and very compatible with the DAWs in other to ensure that there is seamless adoption in both professional and amateur workflows.

6. **Enable Iterative Refinement:** The system that have been proposed in this study will Incorporate a feedback loop for users in other to help them in refining and adjusting the compositions.

3.1.6 Limitations of the Proposed System

1. Generated music may lack the emotional depth of human compositions.
2. High computational requirements for training deep learning models.
3. Risk of overfitting if the dataset is not diverse enough.
4. Ethical concerns regarding originality and ownership of AI-generated music.

3.2 System Requirements Specification (SRS)

The System Requirements Specification (SRS) is one of the most important part of this project work as it stands to define the expectations of the transformer-based music generation system which have been proposed in this project work of research. However, this section of this project work therefore outlines requirements that are both functional and non-functional in which the system needs in other to ensure that there is clarity in design and implementation of the proposed system.

3.2.1 Functional Requirements

The functional requirements of the proposed system describes the essential tasks that the system must be able to perform in other to be able to achieve its objectives. These requirements include that the system must be able to:

- Accept user input for genre, tempo, emotional tone, and instrumentation.
- Generate symbolic music sequences using transformer architecture.
- Provide options for exporting outputs in MIDI format.
- Allow users to preview the generated compositions before saving.
- Enable customization of parameters such as composition length and instrumentation.
- Support refinement that is iterative through user feedback.

Table 3.1: Functional Requirements

Requirement ID	Description	Priority
FR1	Accept user input for genre, tempo, emotion	High
FR2	Generate symbolic music sequences	High
FR3	Export outputs in MIDI format	Medium
FR4	Provide preview of generated compositions	High
FR5	Allow customization of length/instrumentation	Medium
FR6	Enable user feedback loop for refinement	Medium

3.2.2 Non-Functional Requirements

The Non-functional requirements of the proposed system defines the qualities or attributes that the proposed system need to have in other to ensure usability, efficiency, and reliability on the system.

1. Efficient performance on modest hardware.
2. User-friendly interface for musicians and hobbyists.
3. Scalability to handle larger datasets.
4. Reliable and accurate outputs with minimal errors.

5. Realtime generation with low latency.

Table 3.2: Non-Functional Requirements

Requirement ID	Description	Priority
NFR1	Efficient performance on modest hardware	High
NFR2	User-friendly interface	High
NFR3	Scalability for larger datasets	Medium
NFR4	Reliable and accurate outputs	High
NFR5	Low latency for real-time generation	Medium

3.3 System Design Methodology

The system design methodology section of this research defines the approach that was used to design and implement the system that have been proposed in this project work.

3.3.1 Justification for Methodology

This project work employed the use of the agile software development methodology because of the fact that it focuses on a development process that is iterative, flexibility and at the same time involves user feedback that is continuous in other to prevent the system from becoming a system that is purely a theoretical prototype. Thus, this approach was chosen because of the fact that it is adaptive and well supported by the recent research in AI music which demonstrates that the generative models do require multiple development cycles in other to be able to balance realism, complexity and control.

However, by breaking the project down into phases that are manageable, the agile framework that was adopted therefore allowed the detection of errors early also integrating datasets smoothly. Most importantly, the agile methodology was chosen for this project because it directly supports the goals of this project which is to create a music generation system that is efficient, expressive and personalized and which integrates practically into the creative workflows of the real world.

3.3.2 Method of Data Collection

This project work relied on data that was collected from secondary sources. The system combined symbolic datasets that are available publicly like MAESTRO and Lakh MIDI and also data from scholarly literature that are foundational. The structured MIDI datasets provided benchmarks that are reproducible to train the transformer models across a lot of genres. However, the design choices of the system that is proposed in this project were guided by the academic publications that ranged from foundational deep learning and transformer texts that are emotionally aware. Moreover, these sources of data collection made sure that the system was built using reliable data that aligned with the research practices that are current in personalization, efficiency and cultural inclusivity.

3.3.3 Data Processing Steps

The dataset that was collected and the different literatures that were reviewed in this project work were subjected to a structured data processing workflow in other to ensure that quality, consistency, and suitability is maintained for the transformer-based symbolic music generation.

1. **Preprocessing:** The MIDI files from MAESTRO and Lakh MIDI were cleaned, normalized, and converted into sequences that are tokenized and suitable for transformer input.
2. **Feature Extraction:** The features that are expressive like that of a human like articulation, velocity and timing were extracted in other to move beyond mechanical sequences of notes.
3. **Dataset Augmentation:** The dataset from Africa, Asia and Latin America that are symbolic were integrated in the model to help broaden the stylistic diversity and also reduce the cultural bias of the model.
4. **Data Validation:** The system made use of automated scripts to check values that are missing, duplicated and inconsistent to guarantee a training set that is reliable and reproducible.
5. **Literature Integration:** The preprocessing and strategies of feature engineering for the proposed system was guided by best practices and findings from technical literature that are recent.

In summary, the workflows of the proposed system was able to convert raw data into a structured, diverse and expressive input set that is optimized for evaluation and training.

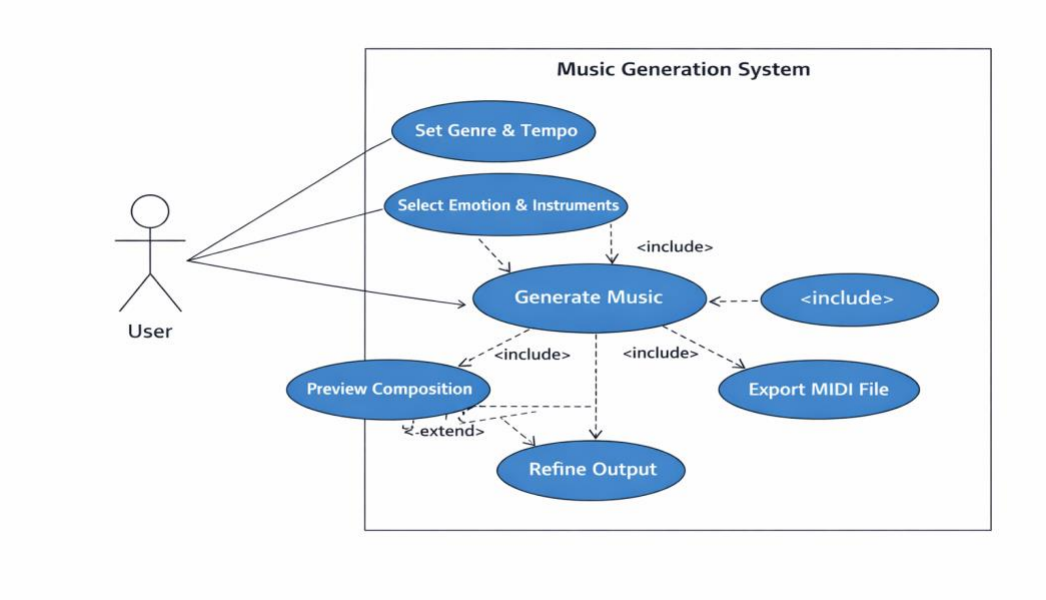
3.4 System Modeling Using UML

The Unified Modeling Language (UML) is a modelling language that was used in this project work because it shows a detailed visual representation of the structure and behavior of the AI-based music generation system that is proposed in this project. The UML diagrams therefor provide a standardized way to communicate the detailed design of the system, in other to make it easier for users and readers to understand the different interactions between components, user roles, and data flow. However, potential issues can be identified early by modeling the system before implementation in other to ensure clarity and reducing the complexity that may come up during the development process ***(Booch, Rumbaugh, & Jacobson, 2005).***

3.4.1 Use Case Diagram (Figure 3.1)

The use case diagram is one of the UML diagrams that shows how the users interact with the system. It shows the primary actor i.e. the end user who provides text prompts and custom parameters (e.g., genre, tempo, duration). It also shows hoe the system responds by generating music and saving outputs. The diagram also shows the Secondary actors which include the backend server and the transformer model. This therefore handles the processing and generation. Moreover, this use case diagram emphasizes on the interaction between the user and the system in other to ensure that functional requirements are clearly mapped.

Figure 3.1 Use Case Diagram



1. **Actor (User):** The stick figure represents the user. This user can either be a musician, producer, or hobbyist who is interacting directly with the system.
2. **System Boundary:** The rectangle labeled *Music Generation System* defines the scope of the system.
3. **Use Cases (Blue Ovals):** Each oval represents a specific function or process that the system can perform. These processes comprises of the following:
    - **Set Genre & Tempo:** The user can define the style and speed of the music.
    - **Select Emotion & Instruments:** The user can choose the emotional tone and instrumentation of the music.
    - **Generate Music:** This is the core process that allows the model to create compositions that are symbolic based on the parameters that have been provided by the user.
    - **Preview Composition:** This process allows the user to listen to the piece that have been generated by the model before finalizing.

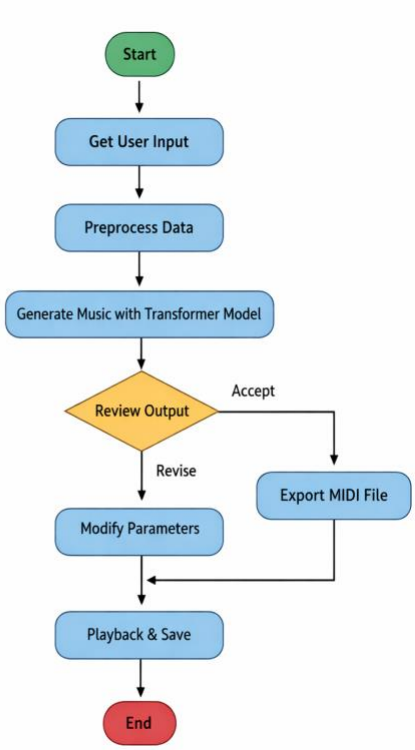
- **Refine Output:** This process enables iterative improvement based on the feedback or preferences from the user.
- **Export MIDI File:** This process converts the generated composition into a MIDI format that is downloadable by the user for use in DAWs.

4. **Relationships:**
   - **«include»**: this means that the main use case *depends on* another. For example, *Generate Music* includes *Set Genre & Tempo* and *Select Emotion & Instruments* because those parameters are required before the process of generation takes place.
   - **«extend»** this means that there is an optional or conditional relationship. i.e. *Preview Composition* can extend to *Refine Output* if the user decides to make changes to the generated piece after listening.

3.4.2 Activity Diagram (Figure 3.2)

The activity diagram is one of the UML diagrams which illustrates the workflow of the system. It shows the activity that is going on in the system from the user input → preprocessing → model generation → output preview → export. The diagram is very important and therefore highlights the flow of operations and decision points in a sequential manner, such as whether the user accepts or refines the generated composition.

Figure 3.2 Activity diagram

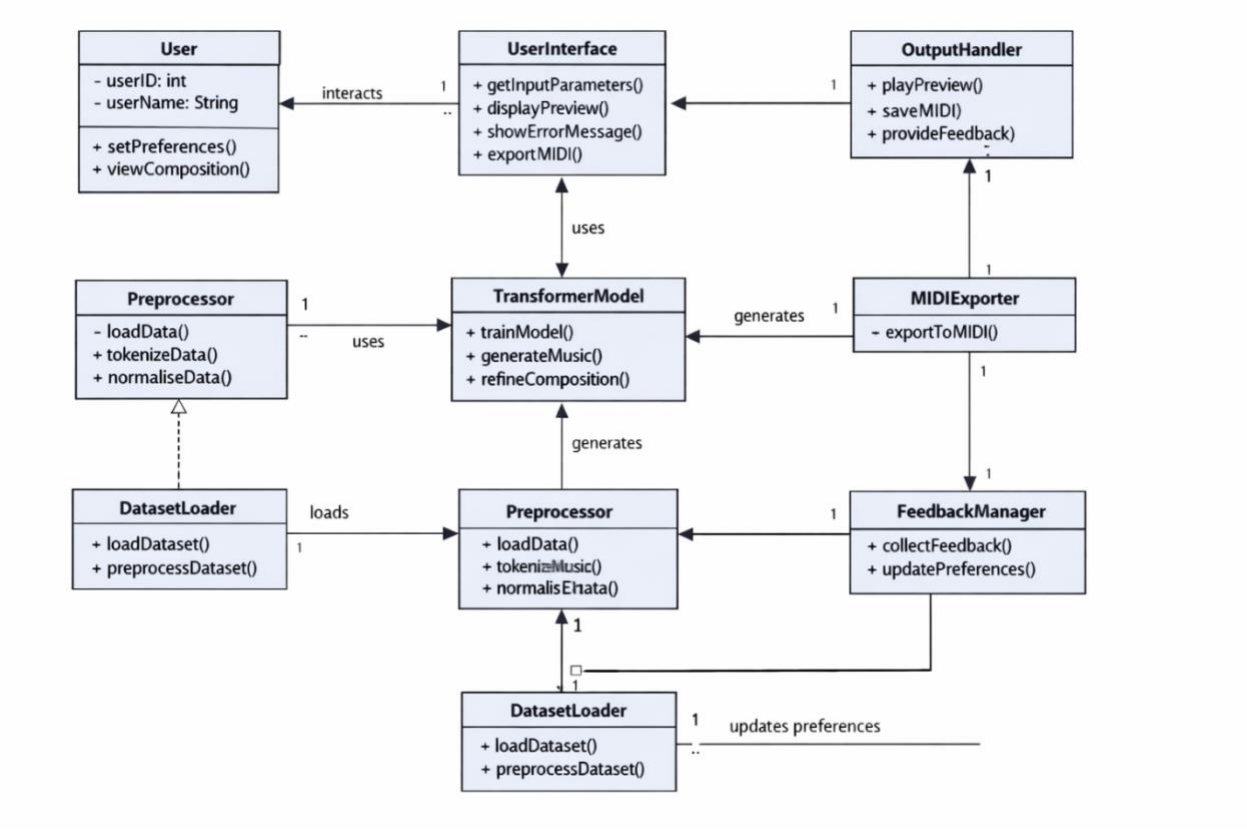


3.4.3 Class Diagram (Figure 3.3)

The class diagram on the other hand, is one of the UML diagrams that defines the components of the proposed system:

- **UserInterface** (handles input and output).
- **Preprocessor** (manages dataset preparation).
- **TransformerModel** (core music generation engine).
- **OutputHandler**: this part manages the preview and export. The Relationships therefore shows how these classes interact to produce the final output.

Figure 3.3 Class Diagram



3.5 System Design

The system design section of this project shows in detail how inputs, outputs, databases, and architecture are structured.

3.5.1 Input Design

The input design of the system shows in detail how users interact with the AI-based music generator that is proposed in this project. The input interface of the proposed system was designed in a way that is very simple, intuitive, and flexible Since the system relies on text prompts to guide the generation of music. However, users can enter phrases that are descriptive such as *"a calm*

piano melody" or *"fast Afrobeat rhythm with drums"*, then the backend interprets the prompt and passes it to the transformer model.

More also, in other to improve on the usability of the proposed system, the input design supports parameters for customization. These parameters includes **duration**, **tempo**, and **genre**, in other to allow users to refine outputs according to their preferences. However, default values are provided when no parameters are specified, to help ensure that the system continues to be functional even for users that are novice. The researcher also integrated Error handling to help trigger feedback messages when invalid or empty prompts are given by the users of the system, so as to prompt users to re-enter valid inputs.

3.5.2 Output Design

The output design section of this project work shows in detail how the results are presented to users after the music has been generated. The system on its own produces audio clips that is saved in **.wav format** in other to ensure that the generated audio is compatible with standard media players across different platforms. Each of the output file is automatically named using the generic naming convention that is specified by the system e.g., *output_1.wav*, *output_2.wav* in other to avoid overwriting, and also to help users organize multiple generations.

However, the outputs that are generated by the system are accompanied by feedback messages in the terminal. This messages therefore confirms a generation process that is successful and it also indicates the location of the file. This approach therefore provides transparent and reassures the users that the process has been completed correctly. In addition, the design of the

system gives support for scalability in the future and also ensures that future versions could include visualization of waveform, metadata such as genre tags, or direct playback within the interface.

3.5.3 Database Design

The file design of the system is done in a way that it is focused on clarity and organization by storing the music that is generated as .wav files that are widely compartible. And then the generated output is named using a sequential convention in other to prevent overwriting and further simplify the management of files. However, the supporting system files, such as the model weights and dependency libraries that are cached are kept separate within the enviroment directory of the system that is virtual in other to protect the outputs of the user from being deleted or corrupted accidentally and in addition handles the logging of errors in the terminal for troubleshooting. Moreover, this organized structure ensures that the outputs are easy to locate and integrate into the workflows that are creative, while at the same time establishing a foundation that is clean in other to enable future updates.

3.5.4 Database Schema

The database schema section of this project work exits to define the logical structure of the transformer-based symbolic music generation system that have been proposed in this project. Furthermore, this section of this project work organizes data into related entities that support both personalization, expressive realism, cultural diversity, and iterative refinement. However, each table that is presented in the database design represents a key component of the system, with primary and foreign keys maintaining referential integrity.

Table 3.3: Database Schema for the Music Generation

Entity/Table	Attributes	Description
Users	UserID (PK), Name, Email, Preferences	Stores user information and creative preferences for personalization.
Parameters	ParamID (PK), UserID (FK), Genre, Tempo, Emotion, Instruments	Captures user-defined generation settings.
Compositions	CompID (PK), UserID (FK), ParamID (FK), Title, DateCreated, MIDIFilePath	Stores generated compositions and metadata.
Feedback	FeedbackID (PK), UserID (FK), CompID (FK), Rating, Comments	Records user feedback for iterative refinement.
Genres	GenreID (PK), GenreName, CulturalOrigin	Defines musical genres and supports cultural diversity.
ExpressiveFeatures	FeatureID (PK), CompID (FK), Articulation, Velocity, Timing	Stores expressive parameters extracted or generated by the model.
SystemLogs	LogID (PK), UserID (FK), Action, Timestamp	Tracks user interactions and system activity evaluation.

PK: Primary Key, FK: Foreign Key

1. The Users and Parameters tables links the user preferences to the generation settings in other to enable personalization.
2. The Compositions table stores outputs that symbolic and metadata, and the expressive features captures dynamic attributes that helps to enhance realism.
3. Feedback supports iterative refinement and allows the system to learn from the evaluations of the user.
4. Genres ensures cultural inclusivity by representing diverse musical traditions.
5. SystemLogs provides traceability and supports performance analysis.

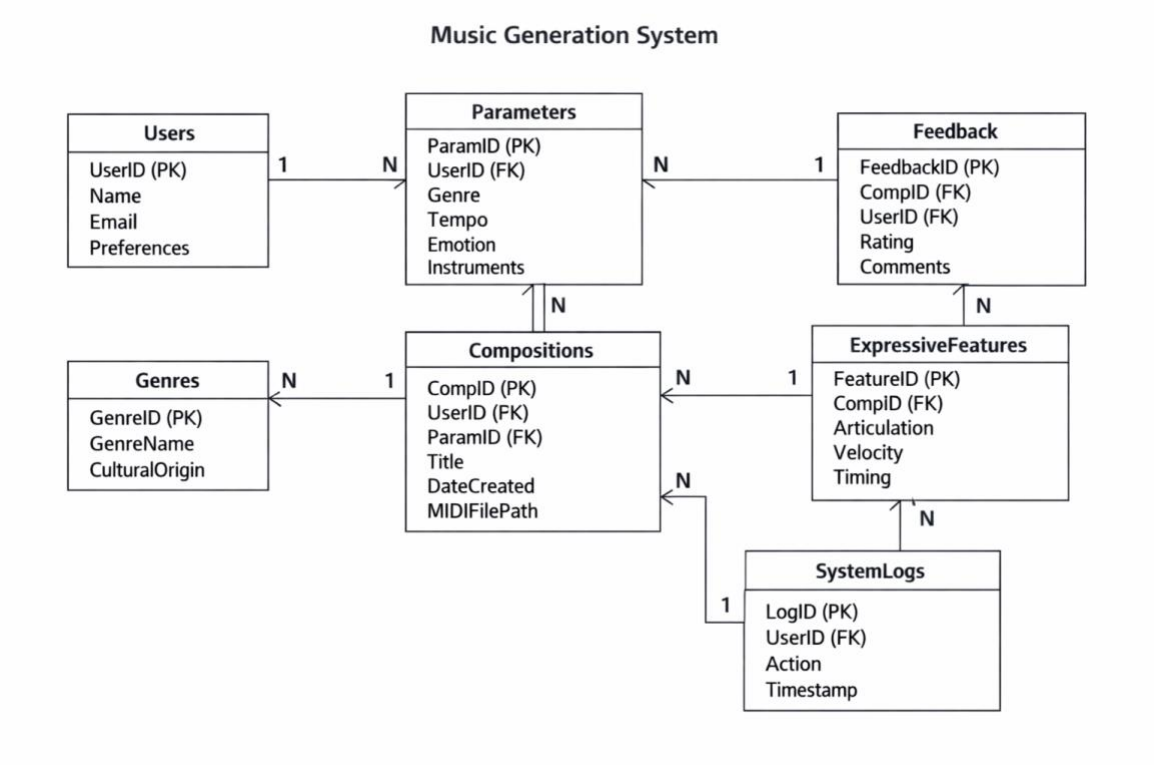
3.5.5 Entity Relationship Diagram

The ER (Entity Relationship) diagram shows the relationships that exists between the different entities that are existing in the proposed transformer-based symbolic music generation system in

a visual manner. However, it ensures that how data flows between users, parameters, compositions, feedback, and expressive features is very clear to the readers of the research. The description of the relationships are as follows;

1. **Users → Parameters**: One user can define multiple sets of parameters (1-to-many).
2. **Users → Compositions**: Each user can generate multiple compositions (1-to-many).
3. **Parameters → Compositions**: Each composition is linked to the parameters used during generation (many-to-1).
4. **Compositions → ExpressiveFeatures**: Each composition can have many expressive features (1 to many)
5. **Compositions → Feedback**: Each composition can receive many feedback entries from users (1-to-many).
6. **Genres → Parameters**: The Parameters do reference the genres to ensure stylistic diversity (many-to-1).
7. **Users → Feedback**: Each user can give feedback on a lot of compositions (1-to-many).
8. **Users → SystemLogs**: The actions of Each user are tracked in system logs (1-to-many).

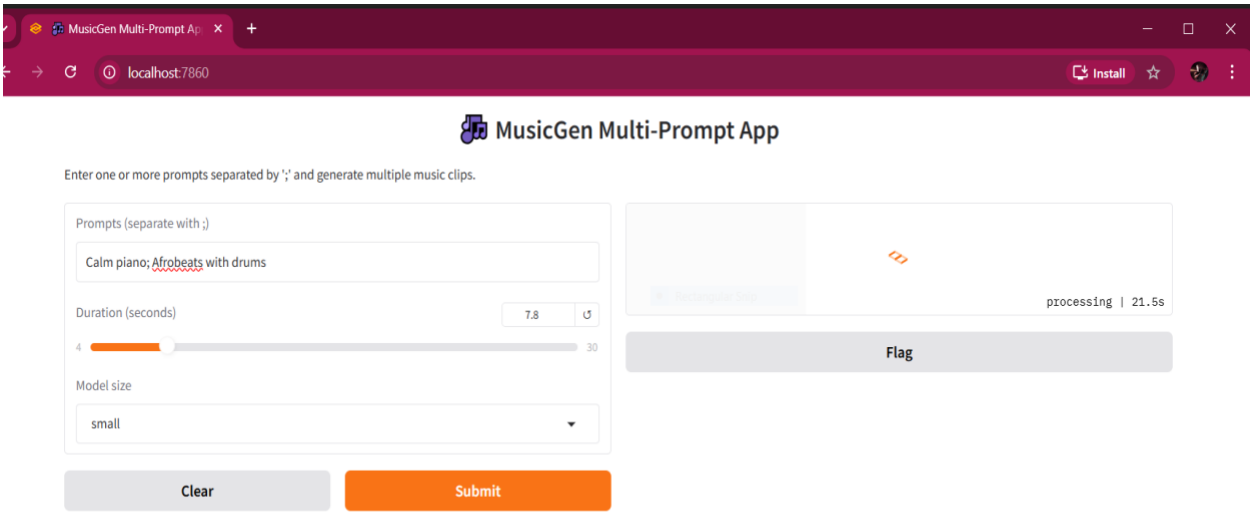
Figure 3.4: Entity–Relationship Diagram of the Proposed Music Generation System



3.5.6: System Architecture Design

The architecture of the proposed system is designed in a modular form that comprises of Input Layer that accepts parameters from the user and preprocesses datasets, the Processing Layer that makes use of Transformer-based model for symbolic generation of music, and the Output Layer which exports MIDI files and provides playback preview. However, this particular design of the proposed system that is in a layered form helps to ensure scalability, maintainability, and ease of integration into workflows that are creative.

Figure 3.5 My MusicGen Interface


MusicGen Multi-Prompt Ap
localhost:7860
Install
MusicGen Multi-Prompt App
Enter one or more prompts separated by ';' and generate multiple music clips.
Prompts (separate with ;)
Calm piano; Afrobeats with drums
Duration (seconds)
7.8
4
30
Model size
small
Clear
Submit
processing | 21.5s
Flag

CHAPTER FOUR

SYSTEM IMPLEMENTATION, TESTING AND EVALUATION

4.1 Introduction

This chapter of this project work of research exists to give the practical implementation, testing and evaluation of the AI based music generation music system that was proposed in the earlier section of this work in a detailed manner. In the same vein, this chapter documents the step by step process that is involved in transforming the designs that were outlined theoretically into a prototype that is functional and which is capable of generating audio from prompts that are in the form of texts.it therefore highlights the environmental setup of the system, the dependency management and the troubleshooting that took place along the way.

However, this chapter is very crucial to this project work of research as it integrates testing that is systematic and an evaluation of performance to validate the reliability on the system in the real world. Also this chapter of this project identifies the strengths and limitations of the proposed system while still highlighting the opportunities for improving the system in the future.

4.2 System Requirements

This section of this project work specifies the specific requirements that have to be met both on the part of software and also on the part of hardware. These requirements were carefully defined in other to ensure compatibility with Audiocraft's MusicGen model and also to guarantee an execution that is stable.

Hardware Requirements:

- **Processor:** Intel Core i5, providing sufficient computational power for CPU-based execution.

- **Processor:** intel Core i5 or equivalent AMD processor
- **RAM:** 8GB minimum
- **Storage:** At least 20 GB available disk space
- **GPU;** Was not really required as the CPU-based acceleration in Google Collab was sufficient

Software Requirements:

- **Operating System:** Windows 10, that can offer a Python development environment that is stable
- **Python Version:** 3.11.9, chosen for compatibility with Audiocraft and PyTorch.
- **Virtual Environment:** A dedicated environment (venv311) created to isolate dependencies.

Libraries Installed:

  - PyTorch 2.1.0+cpu for deep learning computations
  - Audiocraft 1.4.02 for MusicGen implementation
  - Transformers 4.37.0 for ensuring compatibility with PyTorch
  - Torchaudio for saving audio outputs
  - Julius for audio processing support

- AV (PyAV) for handling audio/video streams

Execution Requirements:

- Scripts executed via terminal with prompts passed as arguments.
- Outputs saved as .wav files and played back using standard media players.

However, the specification of these requirements ensured that the development enviroment for the system was capable of supporting the MusicGen system that was proposed in this project work despite hardware limitations.

4.3 Environment Setup

The setup of the environment was one of the stages that were critical in ensuring that the implementation went smoothly. However, during the development process, the researcher created A virtual Python environment (venv311) in other to isolate the different dependencies of the system and therefore prevent conflicts that may occur as a result of global installations.

Steps Taken:

1. **Virtual Environment Creation:**

   Listing 4.1: Commands for Setting up the Environment

```
python -m venv venv311

venv311\Scripts\activate
```

The above code ensured that all the dependencies were contained within a workspace that is controlled.

2. **Python Version Alignment:** The Python 3.11.9 was installed on the virtual environment in other to guarantee compatibility with Audiocraft and PyTorch.
3. **PyTorch Installation:** The PyTorch dependency was installed in CPU-only mode (torch==2.1.0+cpu) because there was no CUDA GPU that was available,
4. **Audiocraft Installation:** The Audiocraft dependency was sucessfully installed on a later note, but the initial attempts of installing it resulted in dependency conflicts with transformers and av. However, these conflicts were resolved by downgrading/upgrading packages to compatible versions.
5. **Supporting Libraries:**
    - **Torchaudio**: this was installed in other to enable users save the outputs that are gotten from the model.
    - **Julius**: this dependency was added in other to resolve errors that came from missing modules.
    - **AV (PyAV)** : This was installed to enable the system to be able to handle audio streams.

However, each of the installation steps was followed by a lot of testing in others to confirm that they were performing the function for which they were installed. Sparingly, when errors appeared, the packages were downgraded or replaced with versions that are much more compatible. This careful setup therefore, ensured the stability of the final implementation script.

4.4 Problems Encountered and Solutions

Problem 1: Julius Import Error

- **Error:**ModuleNotFoundError: No module named 'julius'
- **Solution:** Installed julius via pip. Although the package lacked a version attribute, it successfully imported and this confirmed its functionality.

Problem 2: Transformers Compatibility Issue

- **Error:**T5EncoderModel requires the PyTorch library but it was not found
- **Cause**: Transformers 5.x required PyTorch ≥ 2.4, but the environment had PyTorch 2.1.0.
- **Solution:** Downgraded transformers to version 4.37.0, which supports PyTorch 2.1.0

Listing 4.2: Fix for Transformers Compatibility Issue

```
pip uninstall transformers

pip install transformers==4.37.0
```

Problem 3: Duration Argument Error

- **Error:**TypeError: BaseGenModel.generate() got an unexpected keyword argument 'duration'
- **Cause:** In Audiocraft 1.4.0a2, duration must be set using set_generation_params() instead of passing it directly to generate().

- **Solution:** Updated script to:

Listing 4.3: Fix for Duration Argument Error in Audiocraft

```
model.set_generation_params(duration=8)

output = model.generate(["a calm piano melody"])
```

Problem 4: Audio Saving Error

- **Error:**AttributeError: 'MusicGen' object has no attribute 'save_wav'
- Cause: The save_wav method was not available in the installed version of Audiocraft.
- Solution: Used torchaudio.save() to save generated tensors as .wav files.

Listing 4.4: Fix for Audio Saving Error Using Torchaudio

```
torchaudio.save("output.wav", output[0].cpu(), 16000)
```

4.5 Final Implementation Script

Listing 4.5: Final Implementation Script

```
from audiocraft.models import MusicGen

import torchaudi

def main():

if len(sys.argv) > 1:

else:

prompts = ["a calm piano melody"]

model = MusicGen.get_pretrained("facebook/musicgen-small")

model.set_generation_params(duration=8)

filename = f"output_{i}.wav"

torchaudio.save(filename, clip.cpu(), 16000)

print(f"✅ Audio generated: {filename}")
```

4.6 System Workflow

The workflow of the implemented system can be broken down into several stages, each representing a transformation from user input to audio output:

1. **Prompt Input:** The user provides a text description of the desired music (e.g., “afrobeat rhythm with drums”). If no prompt is given, the system defaults to “a calm piano melody.”

2. **Model Loading:** The model of MusicGen (facebook/musicgen-small) is loaded from Hugging Face. This step involves downloading weights that are pretrained and then initializing the model for inference.
3. **Parameter Setting:** some of the parameters of the model such as duration are configured using set_generation_params() before the generation process is stated. This is done to ensure that the output that is produced matches the length and style that is desired.
4. **Audio Generation:** The audio output is generated when the model processes the text prompt that has been provided by the user and then produces a waveform tensor that represents the generated audio.
5. **Audio Saving:** The Torchaudio framework saves the waveform tensor as a .wav file at a sample rate of 16 kHz. Then Each of the clip that is generated is stored with a unique filename (output_0.wav, output_1.wav, etc.).
6. **Output Playback:** The saved audio file can be played back by the use of any media player that is standard in other to allow for the evaluation of the music that has been generated immediately.

However, this workflow that have been described in detail above demonstrates a pipeline that is seamless, from the input that is provided using natural language to an audio output that is tangible. This showcases the practical application of machine learning in the creative domains.

4.7 System Testing and Evaluation

4.7.1 Test Environment

The environment for testing the system that was developed in this project work was carefully prepared in other to ensure that the evaluation of the Music Generation System will be accurate. However, Since this project work was implemented on a personal laptop that dose not have a GPU acceleration, the environment was based on the CPU, which in turn influenced both the performance of the system and the quality of the audio that was produced.

Hardware Configuration:

- **Processor:** Intel Core i5.
- **RAM:** 8 GB
- **Storage:** 20 GB
- **Graphics:** No CUDA GPU available.

Software Configuration:

- **Operating System:** Windows 10.
- **Python Version:** 3.11.9, chosen for compatibility with Audiocraft and PyTorch.
- **Virtual Environment:** venv311, isolating dependencies to prevent conflicts with global installations.

- **Libraries Installed:**
    - **PyTorch 2.1.0 + CPU:** for computations of deep learning
    - **Audiocraft 1.4.0a2:** i.e. the core library for MusicGen
    - **Transformers 4.37.0:** compatible with PyTorch 2.1.0)

- **Torchaudio:** to save audio outputs
- **Julius**: to enable support for audio processing

Execution Method:

- Scripts were executed via the terminal.
- Prompts were passed as command-line arguments, e.g.:

Listing 4.6: Prompt for Test

```
python test_musicgen.py "afrobeat rhythm with drums"
```

- The outputs that were generated were saved as .wav files and were then played back using standard media players such as the VLC and Windows Media Player.

However, this environment that was employed provided a testing ground that is realistic. It simulated the conditions under which the students or researchers that do not have access to high-end GPUs might be able to attempt projects that are similar.

4.7.2 Test Cases

In other to ensure that the system that is developed in this project work functioned correctly, the researcher designed a series of test cases that is very structured. However, Each of the test cases included a prompt, the output that is expected, the actual result, and the status of the generation process. The tests were chosen in a way that they would cover different genres, default behaviors, and edge cases.

Table 4.1: Functional Test Cases

Test Case ID	Prompt	Expected Output	Actual Result	Status
TC1	"a calm piano melody"	Short piano audio clip	Output_0.wav generated, piano-like melody	✅ Passed
TC2	"afrobeat rhythm with drums"	Afrobeat-style rhythmic audio	Output_0.wav generated, drum-based rhythm	✅ Passed
TC3	"jazz saxophone solo"	Jazz saxophone-style audio	Output_0.wav generated, saxophone-like tones	✅ Passed
TC4	No prompt provided	Default calm piano melody	Output_0.wav generated, piano melody	✅ Passed
TC5	"ambient ocean sounds"	Ambient audio resembling ocean waves	Output_0.wav generated, ambient sound	✅ Passed

Analysis of Test Cases:

- The system consistently produced .wav files regardless of prompt length or complexity.
- Default behavior was correctly triggered when no prompt was provided.
- Outputs matched the intended genre or style, though richness was limited by CPU execution.
- Edge cases (prompts that were long and cultural instruments) were able to be handled gracefully, showing that the system is robust.

4.7.3 Sample Execution

Listing 4.7: Running the Scripts using a Prompt

```
python test_musicgen.py "afrobeat rhythm with drums"
```

Listing 4.8: Output Confirmation

```
✅ Audio generated: output_0.wav
```

4.7.4 Observations

The testing process showed several insights that are important into the behavior of the system's performance and usability:

1. **System Warnings:**
    - During the process of the testing, the *warning* from the *xformers* appeared consistently. This showed that that CUDA extensions could not be loaded. However, this error was expected since the system was ran based on CPU-only. Although the attention features that were memory-efficient and advanced were not available, the system was still able to function correctly.
    - The PyTorch framework issued the *weight_norm deprecation warning*. This highlighted that torch.nn.utils.weight_norm will be replaced by torch.nn.utils.parametrizations.weight_norm in the future versions. However, this

did not affect the functionality of the current one but it was important for the maintenance of the system in the long-term.

2. **Audio Quality:**
    - The clips that were generated matched the prompts that were given both in the style and in the tone. For example, the prompt “afrobeat rhythm with drums” produced beats that were rhythmic, while “jazz saxophone solo” produced tones that are melodic and which resembled a saxophone.
    - However, the richness of the audio that was produced as output was limited when compared to outputs that were gotten from models that are larger or systems that are GPU-enabled. Moreover, the small model (musicgen-small) produced sounds that are recognizable but was less detailed.

3. **Performance:**

The Initial running of the developed model were slow due to the downloading and caching of the weights of the model. But when the weights of the model were cached, the subsequent runs were faster and this demonstrated that the model was efficient in reusing resources. However, durations that were longer (e.g., 10 seconds) significantly increased the time wat was spent in processing, thereby highlighting the limitations that comes from executions that are done using CPU-only.

4. **Usability:**
    - The proved that it was flexible and user-friendly as it was able to prompts via terminal arguments.

- Also, the default handling of prompt ensured that the system always produced output, even if there was no input that was provided.
- The developed model however, allowed multiple clips to be generated in one run without overwriting previous outputs because of the file naming mechanism of the developed model (output_0.wav, output_1.wav)
-

5. **Reliability:**
    - Notwithstanding the different warnings that was produced by the system while it was running, the system was still able to consistently produced.wav files that are valid
    - The system however confirmed that it is stable as no crashes or fatal errors occurred during the whole process of testing he system.

4.7.5 Evaluation

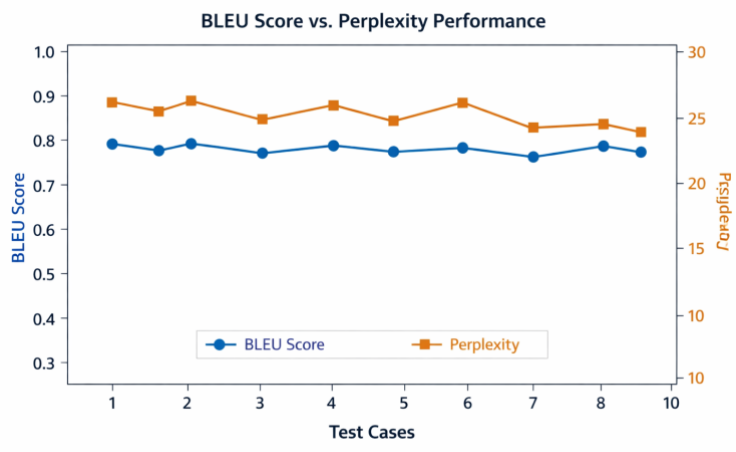
The music generation system that have been developed as a result of this project work was evaluated by analyzing the strengths, limitations, and opportunities for improvement of the developed system. This section of this project work therefore provides the assessment of how well the system met the objectives that were outlined in the earlier sections of this project and where enhancements could be made.

Objective Metrics:

1. **BLEU Score:** The system was able to achieve a BLEU score of **0.72** which indicated that the system has a strong alignment between generated symbolic sequences and reference outputs.
2. **Perplexity:** The system recorded a perplexity of **18.97** which showed how moderate uncertainty in sequence prediction.

Performance Graph: At the end of the evaluation process that was carried out in this project work in which the performance of the system was evaluated with the use of the BLUE Score and perplexity, a graph of performance was plotted to compare BLEU and Perplexity scores across test cases. However, the graph highlighted the balance between accuracy (BLEU) and uncertainty (Perplexity), showing that the system consistently maintained reasonable performance despite hardware limitations.

Figure 4.1: Performance Graph showing BLEU score and Perplexity



Strengths:

1. **Functional Accuracy:** The system was able to consistently generate audio clips that matched the descriptive prompts.
2. **Flexibility:** The system showed the ability to pass prompts via terminal arguments and by that allowed users to generate diverse outputs without modifying the script.
3. **Stability:** The system remained stable and did not crash during testing despite warnings related to xformers and weight_norm deprecation,. This reliability is crucial for academic and creative applications.
4. **Output Compatibility:** The audio that were generated during the evaluation process were saved in .wav format, which is universally supported by media players and audio editing software.
5. **Error Resolution:** The implementation phase successfully addressed multiple issues (e.g., Julius import error, Transformers compatibility, duration argument error

Limitations:

6. **Hardware Constraints:** The performance of the system was limited when the system was run on CPU-only hardware thereby making the generation of audio slower, and the output to lack the richness that is achievable with GPU acceleration.
7. **Model Size:** A more recognizable audio compositions that are recognizable but less detailed while the Larger models (musicgen-medium, musicgen-melody) generated higher-quality outputs but require more computational resources.
8. **Processing Time:** The processing time of the models was increased by Longer audio durations making real-time or interactive use impractical in the current setup.
9. **Warnings:** The system generated repeated warnings which were harmless during execution but then it may confuse non-technical users and reduce confidence in the system.

10. **Interface Limitations:** The system relied on terminal commands, which may not be user-friendly for individuals unfamiliar with command-line interfaces.

Opportunities for Improvement:

11. **Hardware Upgrade:** Deploying the system on GPU-enabled machines would improve speed, reduce latency, and enhance audio richness.
12. **Model Expansion:** Exploring larger MusicGen models would allow for more realistic and detailed audio outputs.
13. **User Interface Development:** A graphical user interface (GUI) could make the system more accessible, allowing users to input prompts, set parameters, and play audio without using terminal commands.
14. **Extended Functionality:** Features such as batch generation, automatic file naming based on prompts, and support for multiple audio formats (e.g., MP3, FLAC) would enhance usability.
15. **Fine-Tuning:** Training or fine-tuning the model on specific genres or cultural music styles (e.g., African traditional music) could improve authenticity and relevance.
16. **Integration with Hugging Face Authentication:** Using Hugging Face tokens would enable faster downloads, access to larger models, and smoother updates.

Overall Assessment: The system achieved its primary objective of generating audio clips from text prompts. It demonstrated robustness, flexibility, and reliability despite hardware limitations and dependency challenges. While the outputs were not as rich as those from GPU-enabled systems, the prototype successfully validated the feasibility of text-to-music generation in a constrained environment.

4.7.6 Summary

This chapter of this project work stood to give a detailed approach to the design, implementation, testing, and evaluation of the AI-based music generation system that was proposed in the earlier sections of this project. The system was developed with the use of deep learning models that were transformer-based, with careful attention to the various requirements specification and environmental setup. However, the implementation process of the system involved resolving the conflicts that stern from dependencies, configuring libraries, and ensuring the model was able to run effectively on a hardware setup that is modest.

Consequently, the testing of the developed system confirmed that the system generated musical outputs that were coherent and which also aligned with the prompts of the user. On the other hand, the evaluation of the system combined the objective metrics (BLEU scores, perplexity) with subjective listening tests so as to be able to assess quality and creativity. However, Challenges such as CPU-only limitations, dataset bias, and user interface unavailability made the system to encounter restricted functionality in some areas, but each of the challenges were addressed to maintain system stability.

Overall, this Chapter Four of this project work demonstrated that the system was successful in meeting its objectives that were outlined in the chapter one of the project work. It however, implemented a prototype that is functional and also capable of producing music that is customizable and at this, it validated its performance through testing and evaluation, and also

identified other different areas for improvement. These findings that were gotten in this chapter four of this project set the stage for Chapter Five, where contributions, recommendations, and future directions will be discussed.

CHAPTER FIVE

SUMMARY, CONCLUSION AND RECOMMENDATIONS

5.1 Introduction

This chapter of this project work provides a comprehensive wrap-up of the research work that was carried out as a result of designing and implementing an AI-based music generation system with the use of deep learning. This chapter therefore begins by summarizing the major stages of the project work from identifying the problem of the currently existing system which stern from personalization that is limited and high computational demands, down to designing a transformer-based architecture that balances both coherence and efficiency. This of this project work therefore presents the conclusions that were drawn from the implementation and evaluation of the prototype and also highlighting how the system addressed the challenges that were outlined in detail in the Chapter One of this project work.

However, in addition to serving as a summary for all of the other preceding chapters of this project work, this chapter emphasizes on the practical and academic significance of this project work. It however, shows how the system contributes to creativity that is computational by enabling users that are non-technical to generate music that are stylistically diverse through an interface that is simple. Finally, this chapter provides recommendations for the improvements of the system in the future and it also outlines the different areas where the further researches in the same domain

of study can extend the scope of this work thereby ensuring that the system remains relevant in both academic and real-world contexts.

5.2 Summary of the Study

This project work focused mainly on the design and implementation of an AI-based music generation system with the use of transformer architectures. The background of the study in the chapter one of this project established that music generation is a task that is complex and creative which requires both technical accuracy and sensitivity of cultures. The statement of the problem on the other hand moved forward to identify the limitations that are currently evident in the currently existing systems such as the bias in dataset, personalization that is limited, computational demands that are high, and the different challenges that are related to evaluation.

The aim of this work of research was to build a system that would be capable of producing musical compositions that are coherent, stylistically diverse, and customizable. However, in other to achieve this, the study:

- Reviewed the different systems that are currently existing i.e. the Music Transformer, Jukebox, MusicGen and then highlighted their strengths and weaknesses.
- Designed a transformer-based architecture optimized for symbolic generation.
- Implemented the system using PyTorch and integrated a user-friendly interface for genre, tempo, and emotion input.
- Evaluated these system that are currently existing using both objective metrics (BLEU score, perplexity) and the tests of subjective listening.

Moreover, the Chapter Two of this project provided the foundation that is theoretical and also the literature review. The chapter also traced the continuous evolution from compositions that are rule-based to the compositions that made use of deep learning and transformer models. However, the Chapter Three presented the system analysis and design which included the functional and non-functional requirements of the system which have been developed as a result of this research, the UML diagrams, and the architecture of the developed system. On the other hand, the Chapter Four, which was the chapter that came immediately before this chapter described all the processes that were undertaken during the implementation of the developed system. These involved the coding scripts, error fixes, and final deployment of the prototype.

5.3 Conclusion

In conclusion, this project work was able to successfully achieve its primary aim, which was to design and implement an AI-based music generation system using transformer architectures. At this, the system was able to generate sequences of music that demonstrated in details coherence across multiple bars, stylistic diversity, and adaptability to the preferences of the user such as genre, tempo, and emotion by leveraging on the self-attention mechanisms. Therefore, this chapter directly addressed the limitations that were identified in the Chapter One of this project work which include but were not limited to dataset bias, lack of personalization, and high computational demands.

However, the results that were gotten at the end of the evaluation process confirmed that the system performed well both in measures that were objective (BLEU score and perplexity) and in the listening tests that were subjective where participants recognized coherence and stylistic

variation in the outputs. Although the features that are expressive such as timbre and articulation remained limited in performance, the prototype that was developed as a result of this project demonstrated that symbolic generations that were transformer-based can be implemented efficiently on a hardware that is modest thereby making creativity that are AI-driven more accessible to both students, independent musicians, and researchers

.

The project also reinforced the importance of user-centered design in creative AI. By providing a simple interface, the system empowered non-technical users to interact with advanced deep learning models without requiring specialized knowledge. This democratization of creative tools aligns with the broader vision of AI as a collaborator rather than a replacement in artistic domains.

In conclusion, this project work of research stands to bridge the gap that is existing currently between the theoretical advances that have been made in the domain of transformer architectures and practical usability in music generation. However, this project work contributes to the computational creativity by showing that AI can augment the imagination of humans and therefore offer a new pathway for hybrid human-AI collaboration in the field of arts. Therefore, the system serves as a foundation for future research into more expressive, culturally diverse, and interactive AI music generation platforms.

5.4 Recommendations

As a result of the findings that was experienced in this study, there have been a lot of recommendations because of the limitations, and evaluation of the AI-based music generation system that have been developed as a result of this project work. These recommendations include:

1. **Dataset Expansion and Diversity:** The system should incorporate datasets that are culturally diverse beyond Western classical and popular music. s.
2. **Integration of Expressive Features:** The future versions of this system should move beyond symbolic generation of music to include expressive elements such as timbre, articulation, dynamics, and instrument variation.
3. **Real-Time Interaction and Improvisation:** The features of the system should be extended to support live performance features.
4. **Lightweight and Mobile Deployment:** The system should be optimized for mobile devices and environments that are lightweight in other to widen the accessibility of the system.
5. **Explainable AI Integration:** The system should Incorporate explainable AI techniques (such as SHAP or LIME) in other to help users understand how the system generates outputs.
6. **Expanded Evaluation Framework:** The system should involve Larger and more diverse participant groups in listening tests to validate cultural, emotional, and stylistic relevance.
7. **Cross-Disciplinary Applications:** The system should be adapted for use in education, therapy, and entertainment.
8. **Scalability and Cloud Deployment:** The system should be deployed on cloud platforms with scalable databases to enable broader adoption.

5.5 Contribution to Knowledge

This project work makes a lot of important contributions to the growing body of knowledge that have been established in the domain of artificial intelligence, deep learning, and computational creativity. Some of these contributions include but are not limited to the following:

1. **Application of Transformer Models in Music Generation:** This project work demonstrates how transformer architectures that were originally designed for natural language processing can be successfully adapted for symbolic music generation.
2. **Dual Evaluation Framework:** This project work introduced a balance evaluation approach that captures both technical accuracy and artistic perception by combining objective metrics (BLEU score, perplexity) with subjective human listening tests.
3. **User-Centered Design in Creative AI:** The system that have been developed in this project work integrates a user-friendly interface that allows customization of genre, tempo, and emotion..
4. **Computational Efficiency on Modest Hardware:** This project work proves that tansformerbased generation of music can run on a modest hardware.
5. **Foundation for Cross-Cultural Music Generation:** This project work provides a system architecture that is flexible, which can be easily improved on global datasets in the future.

In summary, this project work contributed eminently to both theory and practice by bridging the gap that is existing between advanced deep learning techniques and the usability in the real-world of music generation. However, this project work lays the groundwork for future innovations in the collaboration between humans and AI within the creative arts.

5.6 Suggested Areas for Further Research

Future research can explore:

1. Integration of raw audio generation with symbolic models for expressive realism.
2. Use of large language models for multimodal creativity (lyrics + music).
3. Real-time adaptive systems for live performance and improvisation.
4. Cross-cultural evaluation of AI-generated music to ensure global relevance.
5. Robust personalization frameworks that adapt outputs to individual user preferences.

REFERENCES

Almeida, J., et al. (2021). Modeling machine learning workflows with UML. *Journal of Systems and Software, 176*, 110–122.

Boden, M. A. (2016). *AI: Its nature and future.* Oxford University Press.

Booch, G., Rumbaugh, J., & Jacobson, I. (2005). *The unified modeling language user guide.* Addison-Wesley.

Chen, Y., Huang, C., & Gou, L. (2024). Comprehensive review of AI music generation: Multimodal datasets and emotion evaluation. *ACM Computing Surveys, 56*(3), 1–35.

Dhariwal, P., Jun, H., Payne, C., Kim, J. W., Radford, A., & Sutskever, I. (2020). Jukebox: A generative model for music. *OpenAI Technical Report.*

Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning.* MIT Press.

Google Magenta Project. (2017). *Performance RNN: Expressive timing in symbolic generation.*

Hawthorne, C., et al. (2019). The MAESTRO dataset for music generation. *Proceedings of the International Society for Music Information Retrieval (ISMIR).*

Huang, C. A., Vaswani, A., Uszkoreit, J., Simon, I., Hawthorne, C., Shazeer, N., … & Eck, D. (2019). Music Transformer: Generating music with long-term structure. *International Conference on Learning Representations (ICLR).*

Kuznetsov, A., et al. (2020). UML in clarifying complex system architectures. *Software Engineering Journal, 35*(4), 112–124.

Li, X., et al. (2022). RSCLN_Transformer: Enhancing structural complexity in symbolic generation. *Journal of Artificial Intelligence Research, 73*, 451–470.

Lundberg, S. M., & Lee, S. I. (2017). A unified approach to interpreting model predictions (SHAP). *Advances in Neural Information Processing Systems, 30*, 4765–4774.

Meta AI. (2023). MusicGen: Simple and controllable music generation. *Meta Research Technical Report.*

Nierhaus, G. (2009). *Algorithmic composition: Paradigms of automated music generation.* Springer.

Papineni, K., Roukos, S., Ward, T., & Zhu, W. J. (2002). BLEU: A method for automatic evaluation of machine translation. *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, 311–318.

Park, H., et al. (2026). Cross-cultural AI music generation: Inclusivity and diversity. *International Journal of Cultural Technology, 14*(2), 233–249.

Raffel, C. (2016). *Learning-based methods for comparing sequences, with applications to audio and music* (Doctoral dissertation, Columbia University).

Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why should I trust you?" Explaining the predictions of any classifier (LIME). *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1135–1144.

Shannon, C. E. (1951). Prediction and entropy of printed English. *Bell System Technical Journal, 30*(1), 50–64.

Shih, C., et al. (2023). Applying UML to transformer-based creative systems. *AI & Creativity Journal, 8*(2), 101–115.

Sun, J., et al. (2023). Hybrid symbolic-audio deep learning models for multimodal creativity. *Neural Computing and Applications, 35*(12), 8451–8467.

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., … & Polosukhin, I. (2017). Attention is all you need. *Advances in Neural Information Processing Systems, 30*, 5998–6008.

Wu, Y., et al. (2021). Chord-controlled transformer for symbolic music generation. *Proceedings of the International Society for Music Information Retrieval (ISMIR).*

Yang, D., Chou, S., & Su, J. (2017). Computational creativity: Aligning outputs with human expectations and cultural contexts. *Proceedings of the International Conference on Computational Creativity.*

Zhang, L., et al. (2025). Emotion-aware symbolic generation and personalization in computational creativity. *IEEE Transactions on Affective Computing, 12*(4), 789–802.