A REPORT ON

MUSIC GENERATION USING LSTM MODEL

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY,
PUNE IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE

OF

BACHELOR OF ENGINEERING (COMPUTER ENGINEERING)

SUBMITTED BY

SANKLAP KAILASH WANJARI	Exam No: B150954278
VINOD GORAKH SHENDE	Exam No: B150954267
SWAPNIL SUNIL THORAT	Exam No: B150954271
SHUBHAM SATISH PATIL	Exam No: B150954252

DEPARTMENT OF COMPUTER ENGINEERING

INTERNATIONAL INSTITUTE OF INFORMATION TECHNOLOGY

P- 14, RAJIV GANDHI INFOTECH PARK HINJAWADI, PUNE 411057

SAVITRIBAI PHULE PUNE UNIVERSITY

2021 -2022

CERTIFICATE

This is to certify that the project report entitles

"MUSIC GENERATION USING LSTM MODEL"

Submitted by

SANKLAP KAILASH WANJARI	Exam No: B150954278
VINOD GORAKH SHENDE	Exam No: B150954267
SWAPNIL SUNIL THORAT	Exam No: B150954271
SHUBHAM SATISH PATIL	Exam No: B150954252

is a bonafide student of this institute and the work has been carried out by him under the supervision of **Prof. Sunil A. Sushir** and it is approved for the partial fulfillment of the requirement of Savitribai Phule Pune University, for the award of the degree of **Bachelor of Engineering** (Computer Engineering).

(Prof. Sunil A. Sushir)
Guide
Department of Computer Engineering

(Dr. Ajitkumar Shitole)
Head,
Department of Computer Engineering

(Dr. V. V. Patil)
Principal,
International Institute of Information Technology, Pune - 57

Place: Pune

Date:

ACKNOWLEDGEMENT

It gives us great pleasure in presenting the project report on **'Music Generation Using LSTM Model'**. I would like to take this opportunity to thank my internal guide **Prof. Sunil A. Sushir** for giving me all the help and guidance I needed. I am really grateful to him for his kind support. His valuable suggestions were very helpful.

I am also grateful to **Dr. Ajitkumar Shitole**, Head of Computer Engineering Department, International Institute Of Information Technology, Hinjawadi, Pune, for his indispensable support & suggestions.

We wish to record our sincere gratitude to the **Management of this college** and to our beloved Principal, **Dr. Vaishali V. Patil**, International Institute of Information Technology for her constant support and encouragement in preparation of this report and for making available library and laboratory facilities needed to prepare this project and report.

In particular I am indebted to **Prof. Sunil A. Sushir,** Project Coordinator who had faith in this idea, believed in my ability, whispered the words of encouragement and made helpful suggestions from time to time. I would forever remain indebted to him.

At last we must express our sincere heartfelt gratitude to all the faculty members of the Computer Engineering Department who helped me directly or indirectly during this project work.

SANKALP WANJARI
SWAPNIL THORAT
VINOD SHENDE
SHUBHAM PATIL

ABSTRACT

Generally, music was treated as an analogue signal and was created manually. In recent times, music is prominent to technology which can produce a suite of music automatically with practically no human intervention. To achieve this task, we have to overcome a few technical challenges which are discussed descriptively in this paper. A concise introduction about music and its components is provided in the paper alongside the reference and analysis of related work achieved by various authors in this space. Primary goal of this paper is to propose an algorithm which can be utilized to generate melodic musical notes using Recurrent Neural Networks (RNN), basically Long Short-Term Memory (LSTM) network. A model is designed to execute this algorithm where data is represented with the help of musical instrument digital interface (MIDI) file format for simpler and easier access and better understanding. Preprocessing of data prior to feeding it into the model, uncovering techniques to read, process and prepare MIDI files for input are also discussed. The model used in this paper is utilized to learn the sequences of polyphonic melodic musical notes over a single-layered LSTM network. The model must have the ability to recall past details of a musical sequence and its structure for better learning. description of layered architecture which is used in LSTM model and its intertwining connections to generate and develop a neural network is introduced in this work. This paper imparts a peek perspective on distributions of weights and biases in each layer of the model alongside a precise representation of losses and precision at each step and batches. At the point when the model was thoroughly analyzed, it produced stellar results in generating new melodies.

TABLE OF CONTENTS

Thomas Noltey, Hans Hanssony, Lucia Lo Belloz, “Communication Buses for Automotive Applications” In *Proceedings of the* 3rd *Information Survivability Workshop (ISW-2007)*, Boston, Massachusetts, USA, October 2007. IEEE Computer Society.

LIST OF ABBREVATIONS

ABBREVIATION	ILLUSTRATION
RNN	Recurrent Neural Networks
LSTM	Long Short -Term Memory

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1
INTRODUCTION

1.1 Overview

This paper construes an algorithm (Neural Network) based on the LSTM networks which can be used to generate music and melodies automatically without any human intervention. The key goal is to develop a model which can learn from a set of musical notes, analyze them and then generate a pristine set of musical notes. This task is a real challenge because the model must have capabilities to recall past details and structure of musical notes for future projection of learning sequence. The model needs to learn the original sequences adjacent to past one and transform it for the learning system. The model must learn and convert the original sequences next to the previous one for the learning system. Networks that are less complicated. This is due to the fact that, while any group of notes in music may be paired to produce a chord in theory, only a handful are employed in reality. As a result, the presenceor absence of specific notes may be used to predict whether a different note or set of notes will appear at the same moment. When used with an RNN to represent probability along the time axis, RNN is able to describe polyphonic music more accurately than simpler networks.

1.2 Motivation

This is motivated by the fact that while in principle any set of notes in music can be combined to form a chord, in practice only a few combinations are used. Thus the presence or absence of certain notes can be used to infer whether or not a certain different note or group of notes might be likely to occur at the same time. Combined with an RNN to model probabilities along the time axis, this means that RNN is able to more accurately model polyphonic music than simpler networks. Music has become increasingly visible to technology in recent years, which can now compose a suite of music without the need for human participation. To complete this goal, we must overcome a number of technological hurdles, which are described in detail in this work. The article includes a brief introduction to music and its components, as well as citations and analyses of relevant work done by many scholars in this field. The main goal of this study is to offer a method for generating musical notes using Recurrent Neural Networks (RNN), specifically Long Short-Term Memory (LSTM) networks. This technique is implemented using a model in which data is represented using the musical instrument digital interface (MIDI) file format for easy access and comprehension. Data pre-processing before putting it into the model, revealing reading techniques.

1.3 Problem Definition

The input to our algorithm will be a note or a series of notes from a MIDI file. We then use a Recurrent Neural Network, an Encoder and Decoder Recurrent Neural Network, and a Naïve to generate a new sequence of notes with the aim of making a good piece of music.

1.4 Project Scope & Limitations

Our project falls under the scope of music creation. We are going to accomplish our goal using technologies like deep learning, AI, etc. The aim of our project is to generate musicusing these technologies so that we can make computers able to create music to some extent.

They require a lot of resources and time to get train and become ready for real-world applications. In technical terms, they need high memory-bandwidth because of linear layers present in each cell which the system usually fails to provide. Thus, hardware-wise, LSTMs become quite inefficient. LSTMs get affected by different random weight initialization and hence behave quite similar to that of the feed-forward neural net. They prefer small weight initialization instead.

1.5 Methodologies of Problem solving

By using LSTM model, we are making computers be able to create/generate music that can be used for various purpose like for background sound for different games.

CHAPTER 2

LITERATURE SURVEY

2.1 The Effect of Explicit Structure Encoding of Deep Neural Networks for SymbolicMusic Generation:

Allen Huang stated that the past work in music age has primarily been centered around making a single melody. Later work on the polyphonic music demonstrating is based on time series probability density estimation, has met some partial success. Perhaps the earliest paper on deep learning generated music, composed by Chen et al, creates one music with only one music melody and no harmony. The authors likewise omitted dotted notes and all harmonies and chords. Midi records are organized as a series of concurrent tracks, each containing the list of meta messages [1].

They utilized a 2-layered Long Short-Term Memory (LSTM) recurrent neural network (RNN) architecture to deliver a character level model to predict the following note in a given sequence. In their midi data experiments, they regarded a midi message as a single token, though in piano roll experiment, they treated each unique mix of notes across all time steps as a different token. The architecture permitted the user to set different layer parameter, for example, number of layers, hidden away unit size, sequence length, batch size, and learning rate. They likewise strengthen their learning rate when they see that the rate of a training errors is decreases gradually. The conclusion by Allen Haung is to show that the multi-layer LSTM, character-level language model applied to two separate information representations is capable for creating music that is basically similar to sophisticated time series probability density strategies common in the literature.

It involves random noises as input to the generator CNN. The objective of this generator is to change irregular clamors into genuine midi record. In the meantime, the discriminator CNN that takes input from generator and predicts whether it is from a real or an obtained midi, in this way informs the generator how to appear to be real. This adds up to a generative adversarial network (GAN), which learns the generator and the discriminator iteratively. It demonstrates the way that it tends to be a strong option in contrast to RNNs [2].

2.2 Deep Composer: Deep neural hashing and retrieval approach to automatic music generation:

In this paper, introduce a novel neural system that learns hash encodings for musical segments. Music composition is performed by using the current segment’s hash code as a query to retrieve the next composable segment from the database.

2.3 Music Generation Using Bidirectional Recurrent Network:

In this paper, introduced a bidirectional LSTM network with the goal of generating harmonic music. In general, model improved the quality of generated music through learning context information of notes from horizontal and vertical level, and which are bidirectional.

2.4 Deep Learning for Expressive Music Generation:

In this paper, presented a model capable of generating artificial expressive music compositions in order to study the influence of expressiveness on human and artificial compositions. built an LSTM-based model composed of independent networks - each one in charge of a different sequence of elements. evaluated the generated pieces' expressiveness, conducting user-testing sessions where we assessed the impact of different musical elements on human and artificial excerpts.

2.5 A Unit Selection Methodology for Music Generation Using Deep Neural Networks:

In this paper, method of music generation that utilizes unit selection. Two variables essential to the quality of the system are the width and size of the unit database and the unit length. An auto-encoder was used to demonstrate the ability to reconstruct never seen before music by picking units out of a database.

Sr. No.	Title of Paper	Journal/ conferences/ Year of Publication	Author Name	Issue	Technology	Algorithm	Solution	Summary of Paper
1	The Effect of Explicit Structure Encoding of Deep Neural Networks for Symbolic Music Generation	2019 International Workshop on Multilayer Music Representation and Processing (MMRP)	Ke Chen, Weilin Zhang, Shlomo Dubnov, Gus Xia and Wei Li	Melody generation problem constrained by the given chord progression	LSTM (a type of RNN) and WaveNet (dilated temporal-CNN)	compared two models for conditioned symbolic music generation LSTM and WaveNet	Using encoding structure more explicitly using a stack of dilated convolution layers improves the performance effectively	In this paper, comparison of two representative models for conditioned symbolic music generation.
2	Deep Composer: Deep neural hashing and retrieval approach to automatic music generation.	2020 University of Central Florida IEEE	Brandon Royal, Kien Hua, Brenton Zhang	Struggling to create a full song that has structure, theme, and uniqueness	LSTM	1) Two Phase Music Segmentation 2) Segment Hash Encoding	Introduced Deep Composer, a music generation system based on music retrieval using deep neural hashing to encode the music building blocks	In this paper, introduce a novel neural system that learns hash encodings for musical segments

Sr. No.	Title of Paper	Journal/ conferences/ Year of Publication	Author Name	Issue	Technology	Algorithm	Solution	Summary of Paper
3	Music Generation Using Bidirectional Recurrent Network	2019 2nd International Conference on Electronics Technology, IEEE	Tianyu Jiang, Qinyin Xiao, Xueyuan Yin	Existing system usually ignores the information in the negative time direction which is important in music	bidirectional LSTM network	1) Note context generation in time direction 2) Note generation in note direction	Introduced bidirectional LSTM network with the goal of generating harmonic music	In this paper, introduced a bidirectional LSTM network with the goalof generating harmonic music.
4	Deep Learning for Expressive Music Generation	ARTECH 2019, 9th International Conference onDigital and Interactive Arts,ACM	José Maria Simões, Penousal Machado, Ana Cláudia Rodrigues	In existing systems, artificial pieces lack some expression when compared to music compositions performed by humans	LSTM	Train sequential relationships between specific elements - Notes Network, Velocities Network and Durations Network	Created model capable of generating artificial expressive music compositions in order to study the influence of expressiveness on human and artificial compositions	In this paper, presented a model capable of generating artificial expressive music composition s in order to study the influence of expressiveness on human and artificial compositions.

Sr. No.	Title of Paper	Journal/ conferences/ Year of Publication	Author Name	Issue	Technology	Algorithm	Solution	Summary of Paper
5	A Unit Selection Methodology for Music Generation Using Deep Neural Networks	2016 Cornell University, arxiv	Mason Bretan, Gil Weinberg , Larry Heck	Failing of Markov-based approaches to take into account the higher-level structure and semantics important to music	LSTM	1) Autoencoding 2) Deep Structured Semantic Model (DSSM) Algorithm	Introduced method to generate monophonic melodic lines based on unit selection	In this paper, method of music generation that utilizes unit selection.

Table No. 2.0 Literature Table

CHAPTER 3

SOFTWARE REQUIREMENTS SPECIFICATION

3.1 Assumptions and Dependencies

- MIDI Dataset should be used.

3.2 Functional Requirement:

3.2.1 System Feature (Functional Requirement)

Functional requirement thoroughly describes features, functioning, and the usage of a product and system or software from the perspective of the product and its user. Functional requirements are also called as the functional specifications were synonym for specification is design.

3.3 Nonfunctional Requirement:

The non-functional requirements of the framework are explained below as performance requirements and design plan constraints.

3.3.1 Performance requirements:

- The hardware ought to have the option to handle multiple requests and should give a consistent view to all users.
- The hardware should have a system for validating a client sending a request.
- There ought to be an efficient error reporting mechanism when the access to the data fails for some reasons.
- The system will respond any request in couple of seconds from the time of the request submittal. The system shall be allowed to take more time while doing large processing jobs.

3.3.2 Safety Requirements:

- No damage is expected from the use of the product either to the OS or any information that resides on the client system.
- Secure access of confidential data (client's details). Data security implies protecting data and information systems from unauthorized access, use, disclosure, disruption, alteration or destruction. The terms data security, computer security and data assurance are regularly incorrectly used interchangeably. These fields are interrelated often and share the common objectives of safeguarding the privacy, integrity and availability of information; however, there are a few subtle differences between them.
- Accuracy ought to be good since it is basic use case. The data of the users should be stored in the database safely. So, the accuracy of the system should be excellent to give exact results as it is an application created for critical use case.

3.3.3 Security Requirements

- Administrator will have full admittance to Application to resolve any issues.
- normal client can just read data yet they can't modify or edit anything aside from their personal information.
- The product is protected from unapproved users from using it. The system permits only verified users to work on the application. The users of the system are network clients (Sender Receiver)
- A lot of applications and systems have been confronted serious security threats because of the huge number of new available technologies and the lack of knowledge and investigation about them. In the past, security concerns were essentially around network infrastructure layers. Right now, because of the growing use of networks and the Internet concept dominance, for example, cloud computing, Software as a Service (SaaS), serious vulnerabilities are being found by attackers in the application layer. Hence, the concept of application security layer arose as a essential task in the development cycle. As per Federal Information Processing Standard (FIPS) (The National Institute of Standards and Technology (NIST), 2010) there are three security center principles that guide the data security area:

Confidentiality: Save the access control and revelation limitations on information Guarantee that nobody will be break the guidelines of individual privacy and exclusive information.

Integrity: Avoid the inappropriate (unauthorized) information modification or destruction. Here is included guarantee to ensure the non-repudiation and information authenticity.

Availability: The data should be available to access and use constantly and with reliable access. Certainly, it must be valid for the people who have right of access.

3.3.4 Software Quality Attributes:

- **Usability:** The system ought to be easy to use and self-explanatory. Since all clients know about the general usage of computers, no particular preparation should be expected to operate the system. Aside from this, the system should be highly Reliable, Flexible, Robust, and effectively Testable.

- **Accuracy:** Since we will give the priority to the accuracy of the software, the performance of the Music generation will be founded on its accuracy on generation.

- **Failure handling:** System parts might fail independently of others. Therefore, system components should be built so they can handle with failure of different components they rely upon.

- **Openness:** The system should be extensible to ensure that it is helpful for a reasonable timeframe.

- **Usability:** The product will be embedded in a website. It should be scalable designed to be easily adopted by our system.

- **Reliability:** The system ought to have exact outcomes and quick responses to users changing habits.

3.3.5 Availability Requirements:

- Application should be available 24 hours in order to provide access to userwithout any server down / fail.

- Database backup and recovery plan should be proper in order to avoid any unexpected downtime of Application.

3.4 System Requirements

3.4.1 Database Requirements

In the Music Generation System, we require a database to store data in the backend, DBMSused for data storage is MySQL. The database contains various attributes which perform functionality as per requirement, the database has attributes like Name, Mobile Number, Email ID, Login password credential this was for uses side attribute requirement. For Admin Purpose the attributes like user Name, Admin Id, Admin Number, Generate music.By using these attributes, the database functionality is achieved in the Music Generation System.

3.4.2 Software Requirements (Platform Choice)

- Python
- Xampp
- Operating System: Windows/Linux

3.5.3 Hardware Requirements

- Processor: 8th Generation Intel® Core™ i3-1005G1 Processor (4MB Cache, up to 3.4GHz)
- Memory: 8GB, DDR4, 2666MHz
- Hard Drive: 256 GB

3.6 Analysis Models: Waterfall Model

Waterfall approach was first SDLC Model to be used generally in Software Engineering to ensure the success of the project. In The Waterfall approach, the entire process of software development is partitioned into different stages. In this Waterfall model, normally, the result of one phase acts as the input for the following phase sequentially.

- **Pre-Project:** Here first various requirements will be gathered for which various surveys and research will be done. Then the justification for those requirements will be done.

- **Planning:** First survey of various research papers along with other online resources will be done accordingly. Then by the learnings that we got from our literature survey we will decide the technology, algorithms, frameworks and database to be used in our project.

- **Design:** For the design of our project, we will use the web technologies that will help us to develop the attractive User Interface so that even a layman can use the application effectively. For developing UI, we have used HTML and CSS. Other tools web designers might use include markup validator and other testing tools for usability and accessibility to ensure their websites meet web accessibility guidelines.

- **Implement** - This covers everything from system install to add-ons like custom reports and test type packages.

- **Validate** – Risk assessment is performed along with scripts, IQ, OQ, PQ, DM and execution. Also, in this step we find all the bugs and whether the system fulfills all the requirements according to software requirement specification document.

- **Deployment** – Users receives training on the newly implemented system and the system is made live to use. Also, a demo will becreated which will help the new users to learn how to use it.

The following figure is an exact representation of the various phases of the Waterfall Model:

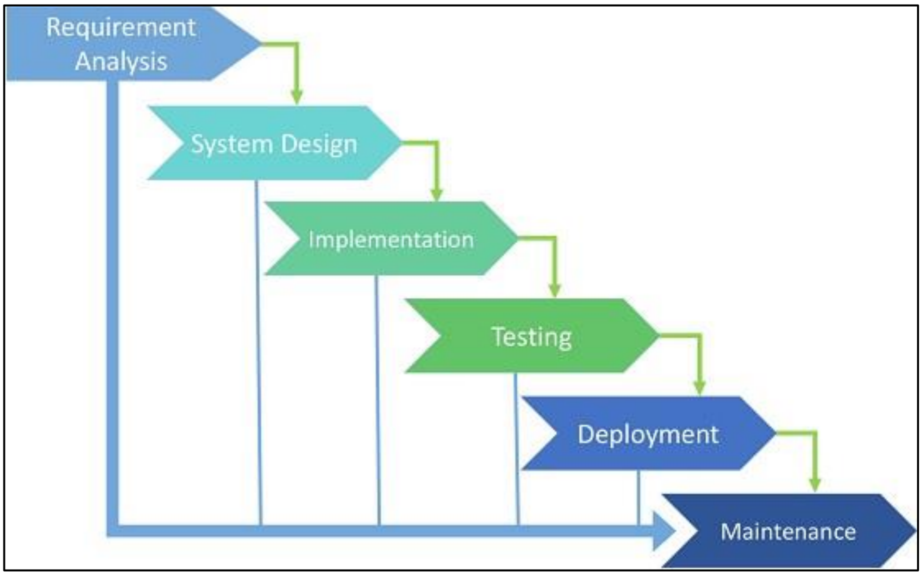

Fig. 3.6 Waterfall Model [5]

CHAPTER 4

SYSTEM DESIGN

4.1 System Architecture

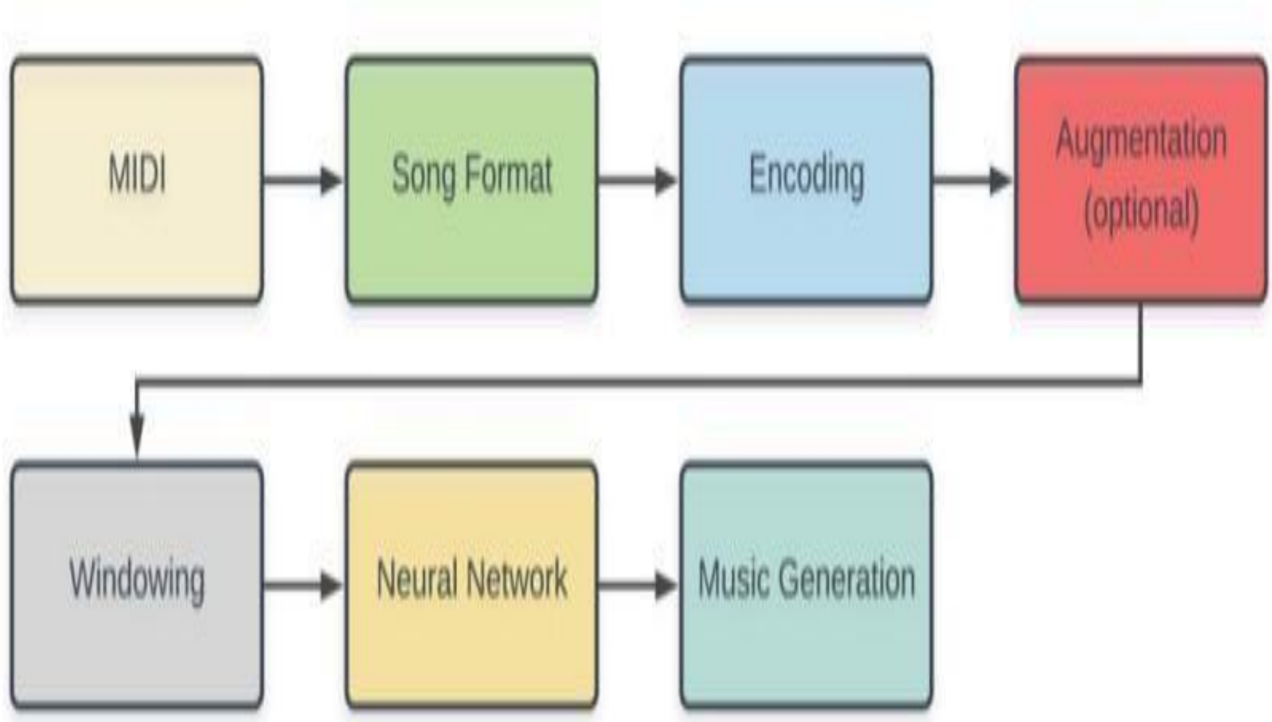

Fig 4.1 System Architecture [3]

4.2 Mathematical Model

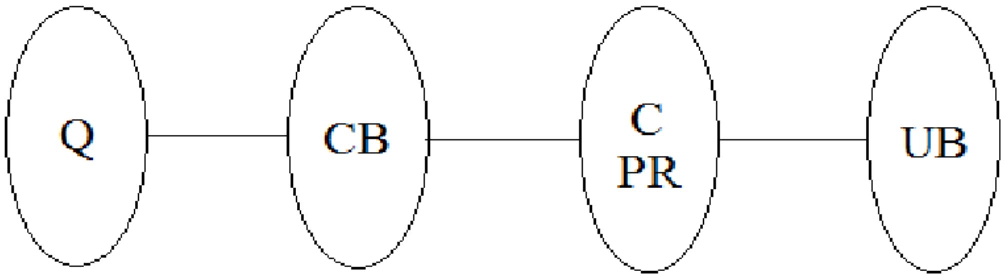

Fig. 4.2.1 Mapping Diagram

- Q = Input wav
- CB = Preprocess
- C = Features Extraction
- PR = LSTM Classifier
- UB = Update result

- **Set Theory**

1. Let S be as system which allow users to predict the music

- S = {In, P, Op}
- Identify Input In as
- In = {Q}
- Where, Q = Input wav
- Identify Process P as
- P = {CB, C, PR}
- Where,
- CB = Perform Preprocessing on the input wav
- C = Extractions of Features and storing Extracted Features for further comparison
- PR = Classification using LSTM
- Identify Output Op as

- Op = {UB}

Where, UB = Update result

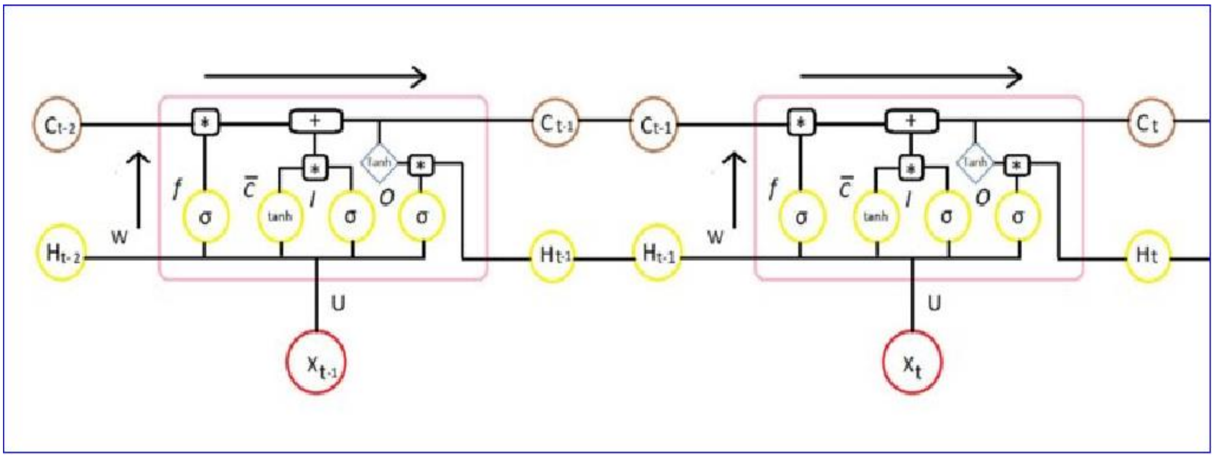

Fig. 4.2.2 Mathematical Model

$ft = \sigma\,(Xt * Uf + Ht\text{-}1 * Wf)$

$Ct = \tanh(Xt * Uc + Ht\text{-}1 * Wc)$

$It = \sigma\,(Xt * Uf + Ht\text{-}1 * Wi)$

$Ot = \sigma\,(Xt * Uf + Ht\text{-}1 * Wo)$

$Ct = ft * Ct\text{-}1 + I\,t * Ct$

$Ht = Ot * \tanh(Ct)$

Xt = Input Vector

Ht-1 = Previous Cell Output

Ct-1 = Previous Cell Memory

Ht = Current Cell Output

Ct = Current Cell Memory

W, U = weight vectors for forget gate Candidate ©, i/p gate(i) and o/p gate(o)

4.3 Data Flow Diagrams

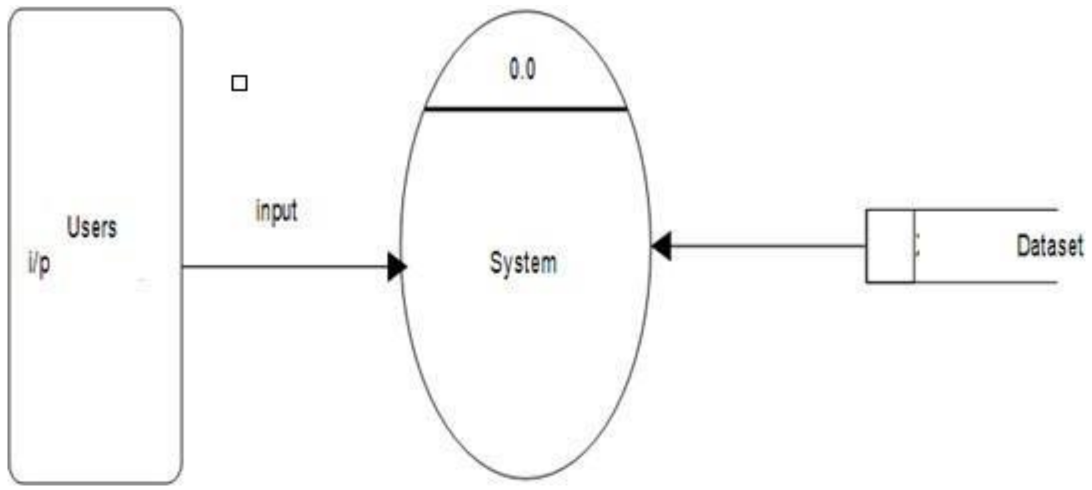

Fig.4.3.1 Level 0 Diagram

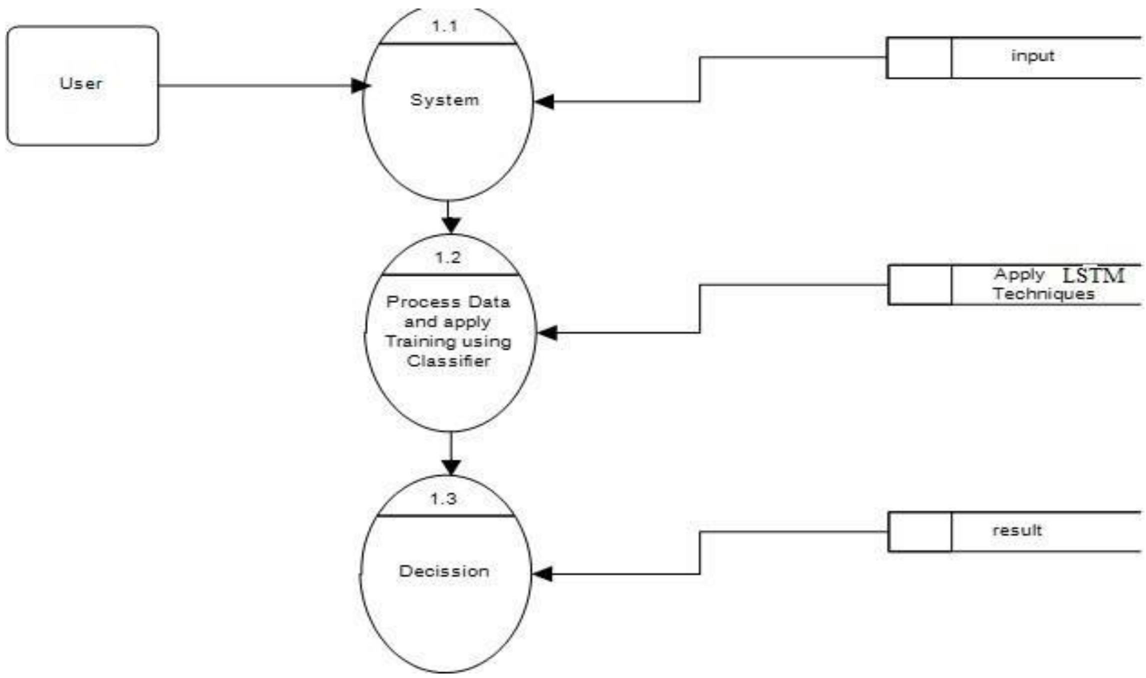

Fig. 4.3.2 Level 1 Diagram

4.4 Entity Relationship Diagram

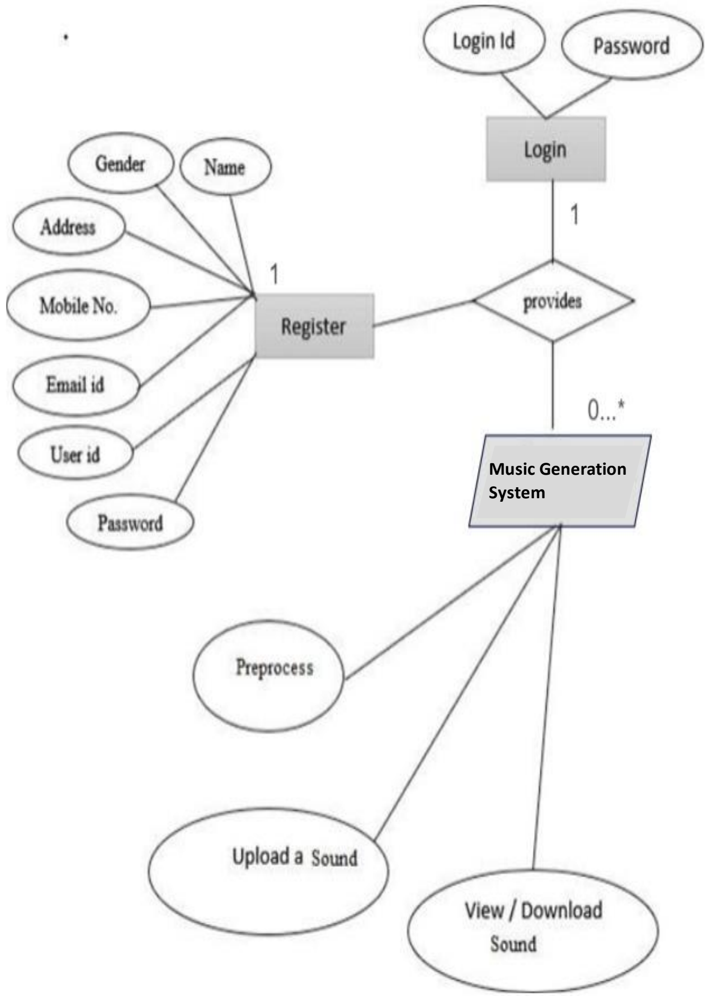

Fig. 4.4 Entity Relationship Diagram

4.5 UML Diagram

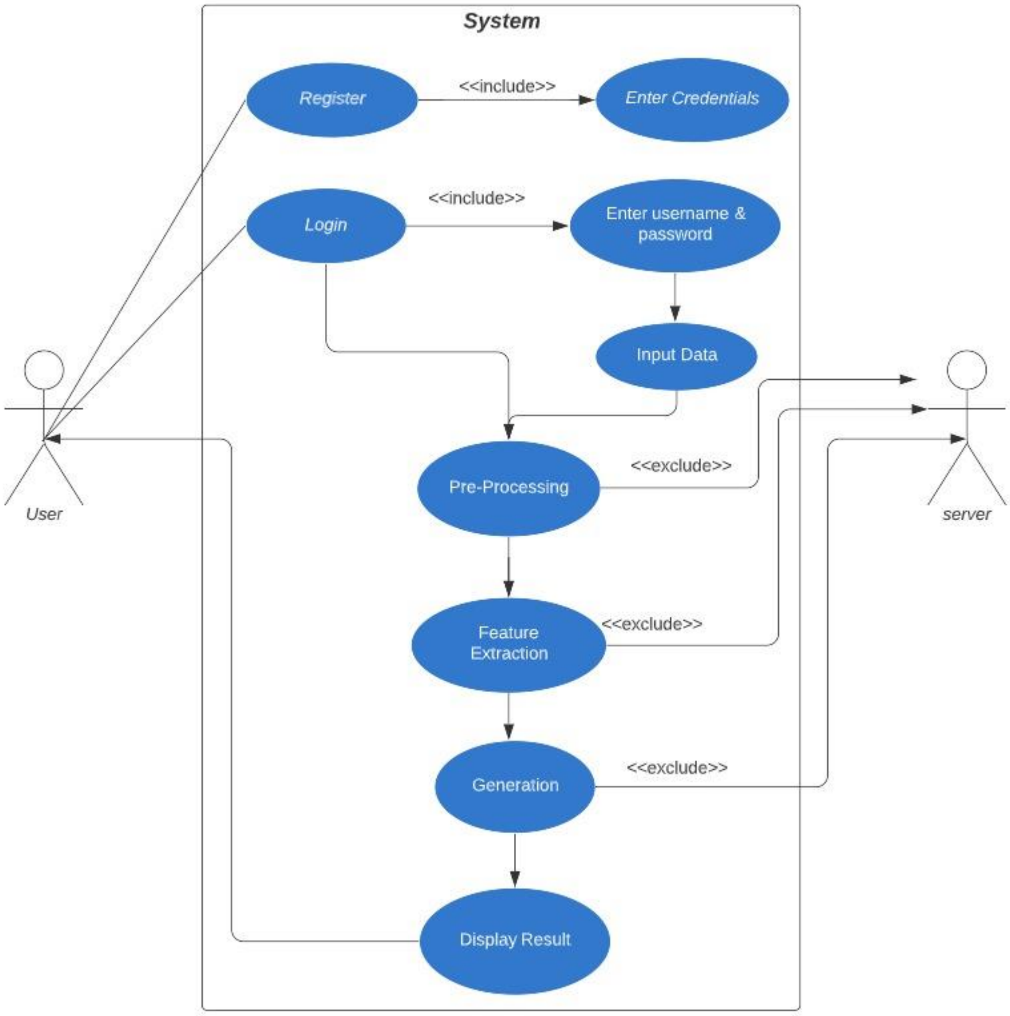

Fig 4.5 UML Diagram

4.6 Activity diagram

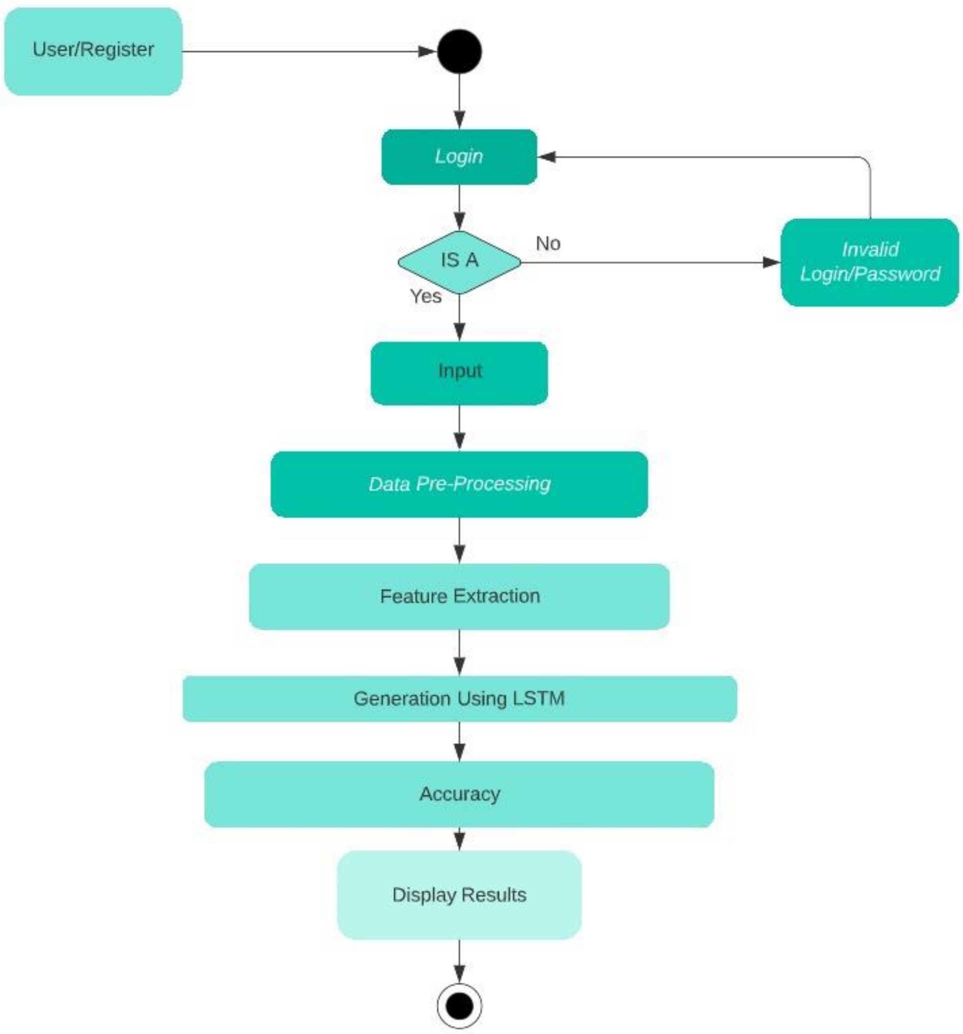

Fig. 4.5 Activity diagram

4.7 Class Diagram

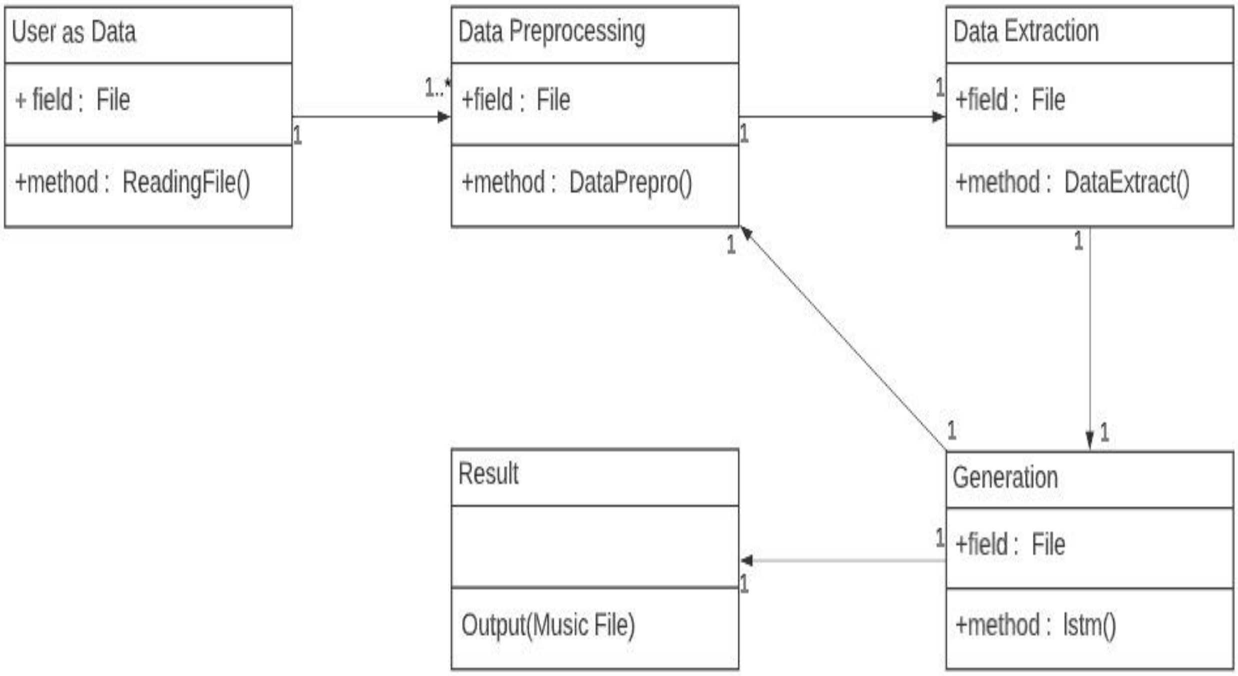

Fig. 4.7 Class Diagram

4.8 Sequence Diagram

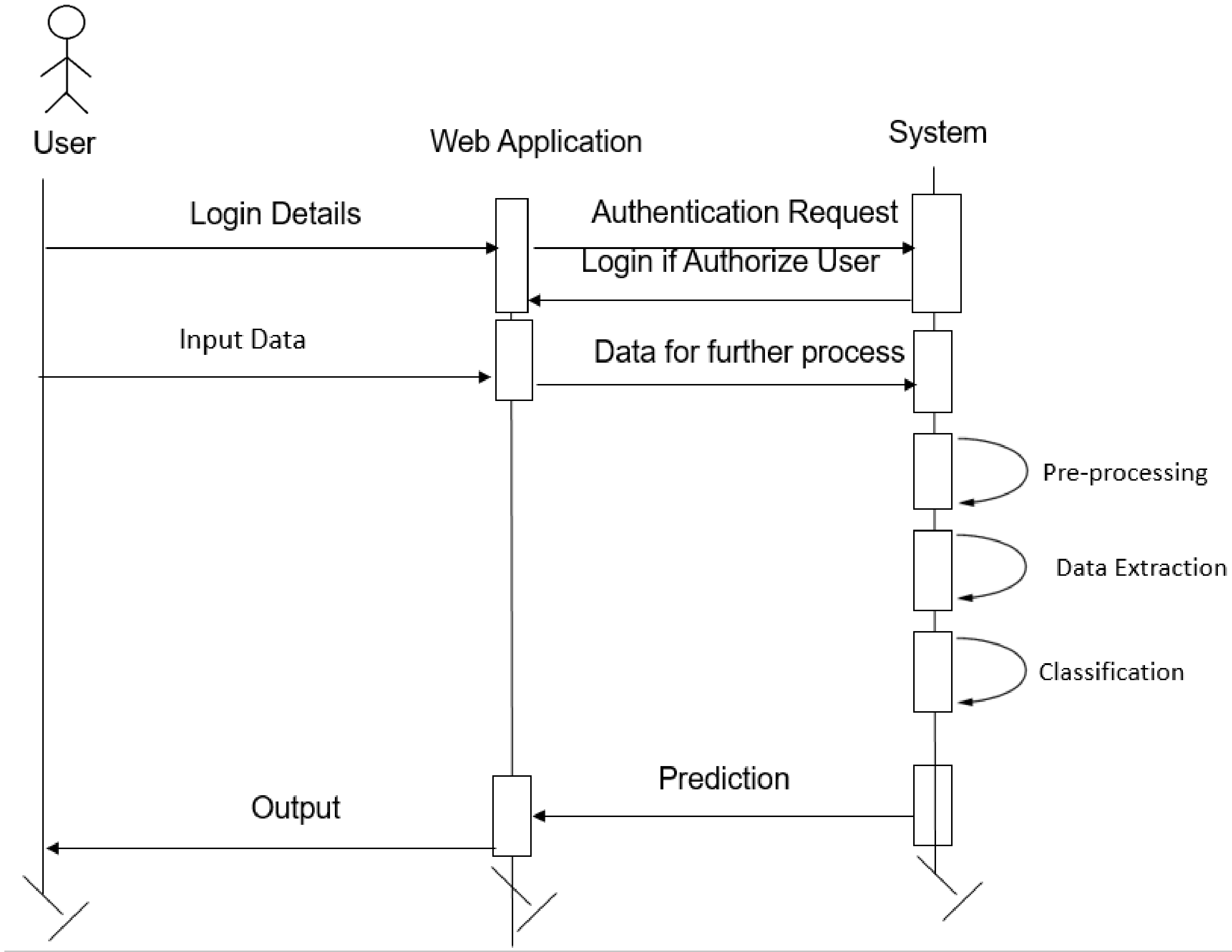

Fig. 4.8 Sequence Diagram

4.9 Communication Diagram

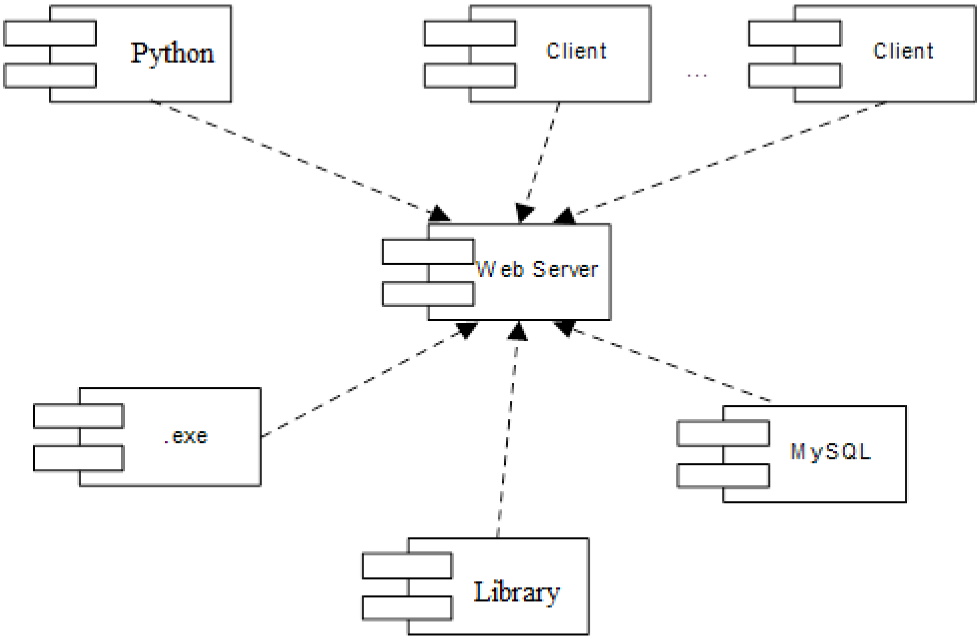

Fig. 4.9 Communication Diagram

4.10 Deployment Diagram

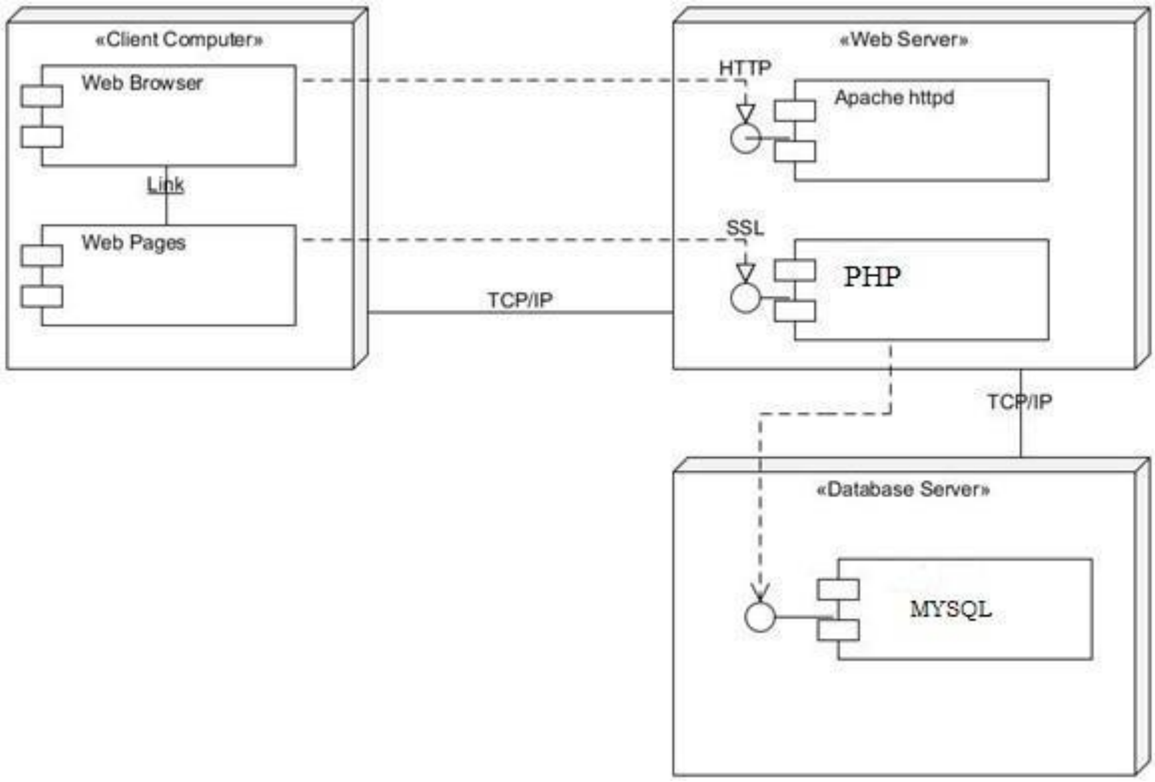

Fig. 4.10 Deployment Diagram

CHAPTER 5

PROJECT PLAN

5.1 Project Estimates

5.1.1 Reconciled Estimates:

Task	Effort weeks	Deliverables	Milestones
Analysis of existing systems & compare with proposedone	4 weeks		
Literature survey	1 weeks		
Designing & planning	2 weeks		
System flow	1 weeks		
Designing modules & its deliverables	2 weeks	Modules: design document	
Implementation	7 weeks	Primary system	
Testing	4 weeks	Test Reports	Formal
Documentation	2 weeks	Complete project report	Formal

Table 5.1: Effort Estimate Table

5.1.2 Project Resources:

In our project development we will require software and hardware resources.

Software Resources:

- Chrome Browser
- Operating System
- Jupyter Notebook
- Anaconda Navigator
- Python
- Xampp

Hardware Resources:

- Minimum Core Duo Processor
- Minimum 4gb ram
- Minimum 50 gb free Storage Space
- Server to Deploy Web Application
- Camera
- Speaker
- Continuous power supply

5.2 Risk Management

5.2.1 Risk Identification

5.2.1.1 Technical Issues

- Are facilitated application particular techniques used to support communication between the client and the developer? The development group will hold regular meetings directly with the client. No formal meetings are held (all informal). During these meetings the product is discussed and notes are taken for future use.

- Are particular methods used for software analysis process? Special methods will be used to dissect the product progress and quality. These are a series of tests and surveys to guarantee the software is up to speed. For additional data, see the Software Quality Assurance and Software Configuration Management records.

- Do you involve a specific method for information and architectural design? Information and architectural design will be generally object oriented. This considers a more significant level data encapsulation and measured quality of code.

5.2.2 Risk Analysis

5.2.2.1 Business Impact Risk

- Amount and quality of documentation that must be produced and delivered tocustomer. The customer will be supplied with a complete online help. Coincidentally, the customer will have access to all development documents for this project, as the customer will also be grading the project.
- Governmental constraints in the process of construction of the product none known.
- Costs associated with late delivery: Late delivery will prevent the customer from issuing a letter of acceptance for the product, which will result in an incomplete grade for the course for all members of the group.
- Costs associated with a defective product is unknown at this particular time.

5.2.2.2 Customer Related Risks

- Have you ever worked with any customer in the past? Yes, all team members have completed at least one project for the customer in the past, though none of them have been to the magnitude of the current project.

- Does the customer who the product have a solid idea of what is required from the product? Yes, the customer has access to both the System Requirements Specification, and the Software Requirements Specification of the software.
- Will the client agree to invest time and energy in formal requirements gathering/ meetings to distinguish project scope? Unknown. While the client will likely participate if asked, the request has not yet been made.

5.2.3 Overview of Risk Mitigation, Monitoring, Management

5.2.3.1 Risk management organizational role

Every individual member of the group will undertake risk management. The development group will reliably be observing and monitoring their progress and project status as to recognize present and future risks as fast and precisely as possible. With this said, the members who are not directly associated with the execution and of the product will also have to keep their eyes open for any potential risks that the development team didn't spot. The obligation of risk management falls on every individual of the group.

5.2.3.2 Process Risks

- Does senior administration support a written policy statement that emphasize the significance of a standard process for software development? N/A. PA Software doesn't have a senior administration. It ought to be noticed that the organized method has been adopted. Toward the finish of the project, it will be determined that if the product method is satisfactory as a standard process, or on the other hand if changes need to be implemented.

- Has your organization developed a written detailed information and description of the software process which is to be used on this project? Yes.

- Are group members willing to use the software process of the project? Yes. The software development process has been agreed upon before development work began.

- Is the software process of your project is used for other products? N/A. PA Software has no other projects as this particular time.

5.2.3.3 Technology Risk

- Is the technology to be built new to your group members? No

- Does the software interface is new with or unproven hardware? No

- Is a specialized user interface is provided by the product requirements? Yes.

5.2.3.4 Development Environment Risks

Software tools are to be utilized for development. Because of the current deadline, the development team felt it would be more useful and productive to start implementing the project than attempting to learn new software tools. After the completing the project software tools might be implemented for future work.

5.3 Project Schedule:

In project management, a timetable is a listing of a project's milestones, different activities, and deliverables, normally with planned start and finish dates. Those things are often estimated by other data included in the project timetable of resource allocation, financial plan, task span, and linkages of dependencies and scheduled events. A schedule is generally used in the project planning and project portfolio management parts of project management.

5.3.1 Project Task Set

Task No.	Task Name	Task Description	Deadline
1	Analysis	Analyze the information given in the IEEE paper	Week 2
2	Literature survey	Collect raw data and elaborate on literature surveys	week 5
3	Module 1: Gather Dataset and Preprocess	Performing preprocess	week 7
4	Module 2: Training	Perform training on 80% of dataset	week 11
5	Design	Assign the module and design the process flow control	week 13
6	Implementation Algorithm	Implement the code for all the modules and integrate all themodules	week 14
7	Testing	Test the code and overall process weather the process works properly	week 16
8	Documentation	Prepare the document for this project with conclusion and future enhancement.	week 18

Table 5.3: Project Scheduling

5.3.2 Timeline Chart

Fig. 5.1 Timeline Chart

5.4 Team Organization

The team for B.E. final year project consists of a team of college students, a college professor as an internal guide making collaborative efforts for fulfillment and implementation of project problem statement. The team is passionate about using technology to solve problems and innovate new system.

5.4.1 Team Structure

The project is being worked upon by a team of five people (1 project internal guideand four project developers). Each project developer is aware of the entire workingof the project. This is possible due to the fact that project group is small. Thus, distribution of work is according to the need of the hour. It is decided to keep the teamstructure highly flexible throughout the project. Each individual shall contribute equally through all the plans of the project namely Problem Definition, Requirements Gathering and Analysis, Design, Coding, Testing and Documentation. ProjectDevelopers are Sankalp Wanjari, Shubham Patil, Swapnil Thorat, Vinod Shende. Prof. Sunil A. Sushir is the internal college guide for providing thorough domain guidance, doubt removal and suggesting approaches andensuring timely completion of activities.

5.4.2 Management reporting and communication

We reported the progress of our project to our internal guide twice a week. We showed our weekly status to our guide and incorporate the necessary changes. A project diary is maintained where all the date wise implementation details are mentioned and is regularly check by our internal guide. We communicated among ourselves in case we want suggestions while executing our tasks. Also, team collaboration tools were used post lockdown to ensure proper development and execution of overall system.

CHAPTER 6
PROJECT IMPLEMENTATION

6.1 Overview of Project Modules

We have used LSTM Model to generate music. With use of various genre dataset like Pop, Jazz, Rock, Pop-Rock.

6.2 Tools and Technologies Used

- The Music Generation System is implemented using Python, PHP, HTML, CSS, Bootstrap, and various libraries which assist in predicting the output.
- MySQL database is used.
- Xampp server is used.

6.3 Algorithm Details

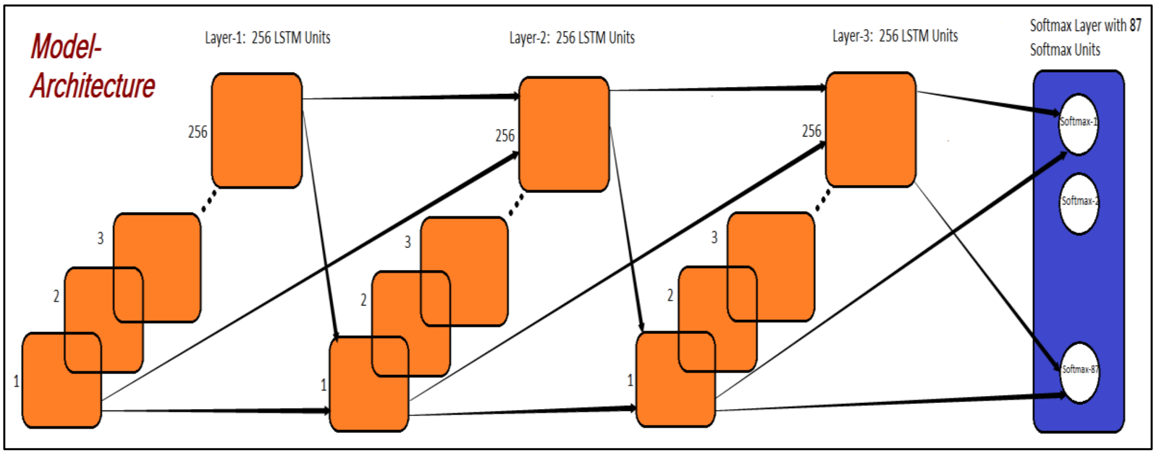

Fig. 6.3 Architecture Diagram

The work with music generation utilizing deep learning included training a RNN to learn to predict sequences of notes. The model is prepared to predict the next note in a monophonic melody sequence; consequently, we call it a Note RNN. Frequently, the Note RNN is implemented using a Long Short-Term Memory (LSTM) model. LSTMs are networks in which each repetitive cell learns how to control the storage of information using an input gate, output gate and forget gate. The first two controls whether data can flow into and out of the cell, and the latter controls whether the items in the cell ought to be reset. Because of these properties, LSTMs are better at learning long-term dependencies in the data, and can adapt more quickly to new data. A softmax function can be applied to the final results of the network, to get the probability the network puts on each note.

Training the LSTM can be achieved using softmax cross-entropy loss and back propagation through time (BPTT). To produce melodies from this model, it is first prepared with a short sequence of notes. Then, at each time step, the following note is picked by sampling from the output distribution given by the model's softmax layer. The note that is sampled is taken back once again into the network as the input at the next time step. However, as previously described, the tunes created by this model will generally wander, and lack melodic structure. In the next section, we will describe how to force rules of music theory on our model utilizing reinforcement learning.

Deep learning neural networks are extremely simple and easy to make and evaluate in Python with Keras, yet you should follow a strict model lifecycle. The step-by-step life-cycle for making, training, and evaluating Long Short-Term Memory (LSTM) Recurrent Neural Networks in Keras and how to make predictions with a trained and prepared model. The following is an overview of the 5 steps in the LSTM model lifecycle in Keras that we will look at.

1) Define Network
2) Compile Network
3) Fit Network
4) Evaluate Network
5) Make Predictions

- **Define Network:** The initial step is to make an instance of the Sequential class. Then you can create your layers and add them in the manner that they ought to be connected. The LSTM recurrent layer contained memory units is called LSTM(). A completely connected layer that often follows LSTM layers and is utilized for yielding a prediction. LSTM layers can be stacked and formed by adding them to the Sequential model. Most importantly, while stacking LSTM layers, we should output a sequence instead of a single value for each input so that the resulting LSTM layer can have the necessary input.

- **Compile Network:** Once we have characterized our network, we should compile it. Compilation is an essential and efficiency step. It changes the simple sequence of layers that we defined into an exceptionally efficient series of matrix transforms in a format intended to be executed on our GPU or CPU, depending on how Keras is structured. Consider compilation as a precompute step for your network. It is generally always required after defining the model. Compilation requires various parameters to be specified, explicitly tailored to training your network. In particular, the optimization algorithm to use to train the network and the loss work function to evaluate the network that is minimized by the optimization algorithm.

- **Fit Network:** Once the network is compiled, it very well can fit, and that implies adapt the weights on a training dataset. Fitting the network requires the training data to be determined, both a matrix of input patterns, X, and an array of matching output result patterns, y. The network is trained utilizing the backpropagation algorithm and optimized by the optimization algorithm and loss function specified while compiling the model. The backpropagation algorithm expects that the network be trained for a specified number of epochs or exposures to the training dataset. Each epoch can be partitioned into groups of data input-output pattern pairs called batches. This defines the number of examples and patterns that the network is presented to before the weights are updated inside an epoch. It is also an efficiency optimization, guaranteeing that not too many input patterns are loaded into memory at a particular time.

- **Evaluate Network:** Once the network is trained, then we can evaluate it. The network can be evaluated on the training the data, however this won't give a valuable indication of the performance of the network as a highly predictive model, as it has seen all of this data previously. We can evaluate the performance of the network on a different dataset, unseen during testing. This will give an estimate of the performance of the network at making predictions for unseen data later on. The compiled model evaluates the loss across all of the test patterns, as well as some other metrics specified when the model was compiled, similar to classification accuracy. A list of all the evaluation metrics is returned.

- **Make Predictions:** When we are satisfied with the performance and result of our fit model, we can utilize it to make predictions on new data. This is basically as simple as calling the predict() function on the model with a variety of new input patterns. The predictions will be returned in the format given by the output layer of the network.

CHAPTER 7
SOFTWARE TESTING

A system should always be tested thoroughly before implementing it, as regards its individual programs. This is because implementing a new system is a major job which a lot of man hours and a lot of other resources, so an error not detected before implementation may cost a lot. Effective testing early in the process translates directly into long term cost saving from reduced number of errors. This is also necessary because in some cases, a small error is not detected and corrected before installation, which may explode into much larger problem. Programming and testing is followed by the stage of installing the new computer based system. Actual implementation of the system can begin at this point using either a parallel or a direct changeover plan, or some blend of two. Software testing is a critical element of Software Quality Assurance (SQA) and represents the ultimate review of specification, design and coding. The purpose of product testing is toverify and validate the various work products viz. units, integrate unit, final product to ensure that they meet their respective requirements.

7.1 Type of Testing

7.1.1 Unit Testing

Unit testing is a method by which individual units of source code are tested to determine if they are ready to use. It is performed by developers.

7.1.1.1 Entry Criteria

- Requirements are at least 80
- Technical Design has been finalized and approved.
- Environment setup is completed and is stable.
- Code development for the module is complete.

7.1.1.2 Exit Criteria

- Code has version control in place.
- No known major or critical defects are pending.
- A testing transition meeting has be held and the developers signed.

- Project manager approval has been received.

7.1.2 Integration Testing

It is the phase in software testing in which individual software modules are combined and tested as a group. Primarily performed by testers with support from developers where so ever required.

7.1.2.1 Entry Criteria

- Code development for the module is complete.
- Unit testing is completed.
- All priority bugs are detected in unit testing phase are fixed and closed.

7.1.2.2 Exit Criteria

- All Integration test cases are executed successfully.
- All priority bugs are detected during unit and integration testing are fixed.
- Installation and deployment ability of integrated product has been successfully tested.
- Project manager approval has been received.

7.2 Test Cases & Test Results

7.2.1 Test Case 1:

- Test Item: User Registration Process
- Action: Register
- Test Notes and Preconditions: Enter valid data.
- Verification Steps: Verify whether user can register successfully or not.

Test Cases	Feature	Description	Steps To Execute	Test Data / Input	Expected Results	Actual Results	Status P/F
TC-01	User Interface	Check all the text boxes, buttons, etc.	1) Click on buttons and dropdowns.	N/a	UI should be perfect	Same as expected	Pass
TC-02	Required fields	Check the required fields by not filling any data	1) Do not enter any value in the field. 2) Click on the Register button.	N/a	It should show a message for filling out all the mandatory fields.	Same as expected	Pass
TC-03	Required fields	Check user should Register by filling all the required fields	1) Enter valid values in the required fields. 2) Click the Register button.	All the valid values in required fields	1. Users should be registered successfully. 2. A successful registration message should show.	Same as expected	Pass
TC-04	Email validation	Check the Email text field that has an Invalid email address.	1) Enter Invalid Emails 2) Click on the Register Button.	1.testAtgmail.com 2.test@gmailcom 3.test@gmail 4.@gmail	It should show the validation message for valid email	Same as expected	Pass
TC-05	Email validation	Check all the valid emails	1) Enter valid Emails 2) Click on the Register Button.	1.test.22@gmail.com 2.test@gmail.com	It should not show any validation message	Same as expected	Pass

Test Cases	Feature	Description	Steps To Execute	Test Data / Input	Expected Results	Actual Results	Status P/F
TC-06	Phone Number validation	Check the phone number when passing alphanumeric data and verify if the length of the phone number is incorrect	1) Enter alphanumeric data in phone field / Enter phone number less or more than 10 digits. 2) Click on Register button	1. dada5$77# 2. 8989895 3. 84889459894	It should show the validation message for Phone Number	Same as expected	Pass
TC-07	Phone Number Validation	Check all the valid Phone numbers	1) Enter valid phone number. 2) Enter all required fields. 3) Click on Register Button	9190112255	It should not show any validation error message for phone number.	Same as expected	Pass
TC-08	Password Validation	Check the password when passing valid data	1) Enter valid password. 2) Click on Register button.	Pass123456	It should not show any validation message	Same a expected	Pass

Table No. 7.1 Test Case 1

7.2.2 Test Case 2:

- Test Item: User Sign In Process
- Action: Sign In
- Test Notes and Preconditions: Enter valid data.
- Verification Steps: Verify whether user can sign in to the system successfully or not.

Test Cases	Feature	Description	Steps To Execute	Expected Results	Actual Results	Status P/F
TC-01	User Interface	Check all the text boxes, buttons, etc.	Click on buttons and dropdowns.	UI should be perfect	Same as expected	Pass
TC-02	Required fields	Check the required fields by not filling any data	1) Do not enter any value in the field. 2) Click on the Register button.	It should show a message for filling out all the mandatory fields.	Same as expected	Pass
TC-03	User Login	Check by passing a correct username and invalid password	1) Enter valid username. 2) Enter incorrect password. 3)Click on login button.	1. User should not get log in and should show proper error message.	Same as expected	Pass
TC-04	User Login	Check when passing correct username and password.	1) Enter valid username and password. 2) Click on the Login Button.	User should log in successfully.	Same as expected	Pass
TC-05	Signup option for new users	Check whether the signup link for the new user is working.	1) Click Signup link.	Clicking signup link takes the user to signup page successfully.	Same as expected	Pass
TC-06	Forgot password	Verify user should get an error message when he/she enters unregistered email id.	1) Click on the Forgot Password link. 2) Enter unregistered email id and click on the send button.	User should get an error message	Same as expected	Pass
TC-07	Reset Password	Verify user able to reset his/her password.	1) Go to reset password link. 2) Enter a new password and confirm the password.	It should not show any validation error message for phone number.	Same as expected	Pass

Table No. 7.2 Test Case 2

7.2.3 Test Case 3:

• Test Item: Dashboard

• Action: Profile Update and logging out

• Test Notes and Preconditions: Enter valid data in profile update process.

• Verification Steps: Check functionality of different tabs and buttons on the dashboard.

Test Cases	Feature	Description	Steps To Execute	Expected Results	Actual Results	Status
TC-01	User Interface	Check all the text boxes, buttons, etc.	Click on buttons and dropdowns.	UI should be perfect.	Same as expected	Pass
TC-02	My Profile Tab	Check functionality of "My Profile" button.	Click on "My Profile" button.	User should directed to his/her profile.	Same as expected	Pass
TC-03	Profile Update	Check user can update his/her information by giving valid input.	Enter all the valid data in respective fields and Click on update profile.	User profile should get updated.	Same as expected	Pass
TC-04	Create Music Tab	Check functionality of "Create Music" button.	Click on "Create Music" Tab.	User should get directed to music creation page.	Same as expected	Pass
TC-05	Setting Button	Check functionality of Setting button	Click on "Setting" button.	User should be able see "My account" and "Log out" tab.	Same as expected	Pass
TC-06	Log Out	Check functionality of Log out button	Click on Log Out button.	User should get logged out of profile.	Same as expected	Pass

Table No. 7.3 Test Case 3

7.2.4 Test Case 4:

- Test Item: Side Menu.
- Action: working of different tabs.
- Verification Steps: Check functionality of different buttons like Profile, Change Password and Dashboard on the dashboard.

Test Cases	Feature	Description	Steps To Execute	Expected Results	Actual Results	Status
TC-01	My Profile Tab	Check functionality of “My Profile” button.	Click on “My Profile” button.	User should directed to his/her profile.	Same as expected	Pass
TC-02	Change Password	Check functionality of update password.	Enter valid data in different fields like Current Password, New Password and Confirm new password.	Password should get updated.	Same as expected	Pass
TC-04	Dashboard Tab	Check functionality of “Dashboard” button.	Click on “Dashboard” button.	User should directed to the dashboard page.	Same as expected	Pass

Table No. 7.4 Test Case 4

7.2.5 Test Case 5:

- Test Item: Music Category Buttons.
- Action: Uploading a dataset files and training the model.
- Verification Steps: Check functionality of different music category buttons on the dashboard.

Test Cases	Feature	Description	Steps To Execute	Expected Results	Actual Results	Status
TC-01	Music Buttons	Uploading dataset and training the model.	Click on any of the four buttons representing music categories pop, jazz, rock, pop-rock.	In the backend, dataset files should get uploaded and model should start training and generate output file.	Same as expected	Pass

Test Cases	Feature	Description	Steps To Execute	Expected Results	Actual Results	Status
TC-02	Music File	Showing created music file to the user.	NA	User should able to see newly created file on the dashboard after model training.	Same as expected	Pass

Table No. 7.5 Test Case 5

7.2.6 Test Case 6:

- Test Item: Music Player
- Action: Click on Play button, Download and Delete.
- Verification Steps: Check functionality of Play, Download and Delete buttons on the dashboard.

Test Cases	Feature	Description	Steps To Execute	Expected Results	Actual Results	Status
TC-01	Player Button	Playing music created by the user.	Click on the play button icon.	Music should start playing after hitting Play button icon.	Same as expected	Pass
TC-02	Player Button	Pausing music which is currently playing.	Click on the pause button icon.	Music should pause after hitting Pause button icon.	Same as expected	Pass
TC-03	Download Button	Downloading music file created by the user.	Click on the Download Button.	Music file should get download.	Same as expected	Pass
TC-04	Delete Button	Deleting music created by the user from the database.	Click on the Delete Button.	Music file should get deleted from the database.	Same as expected	Pass

Table No. 7.6 Test Case 6

CHAPTER 8

RESULTS

8.1 Outcomes:

The Music Generation System successfully generates melodic music and user can also download or delete created music from the database. Also, it successfully helps the user by providing the admin dashboard so that he can easily create the music and listen to it.

8.2 Screenshots:

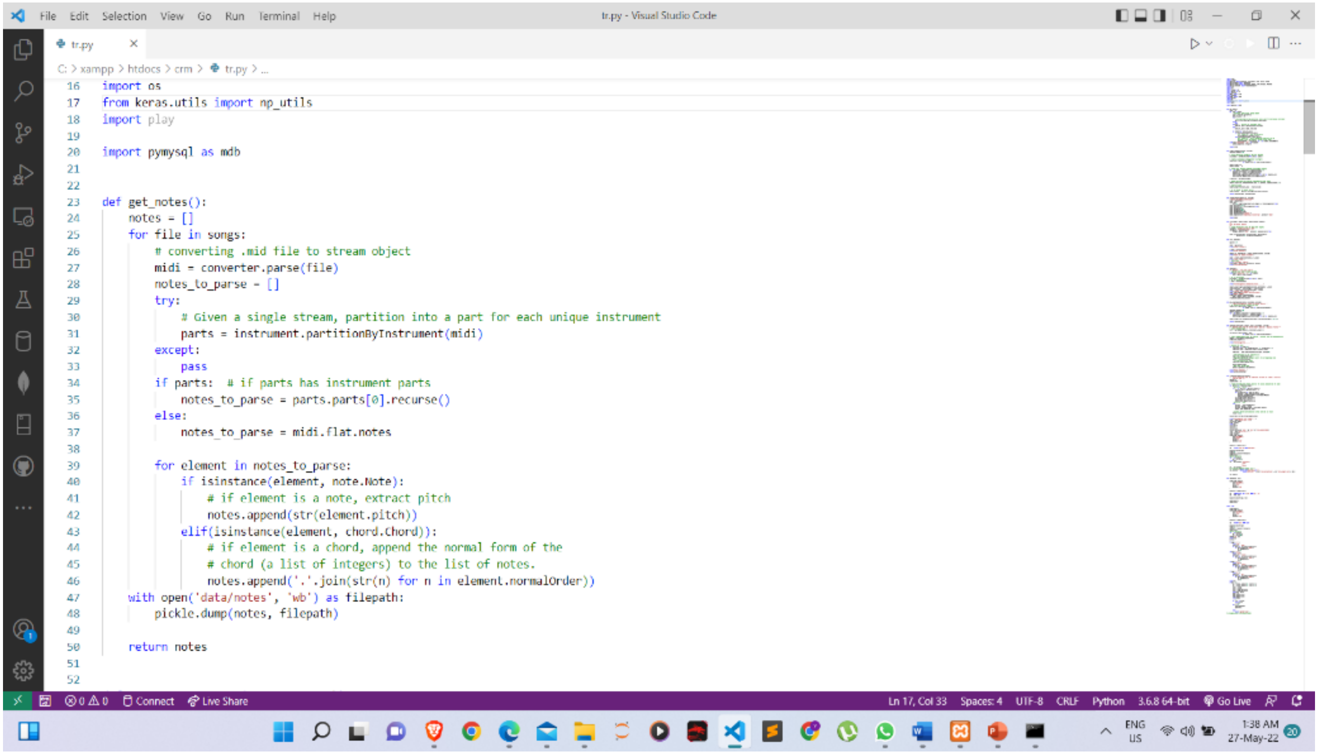

Fig. 8.1 Getting notes

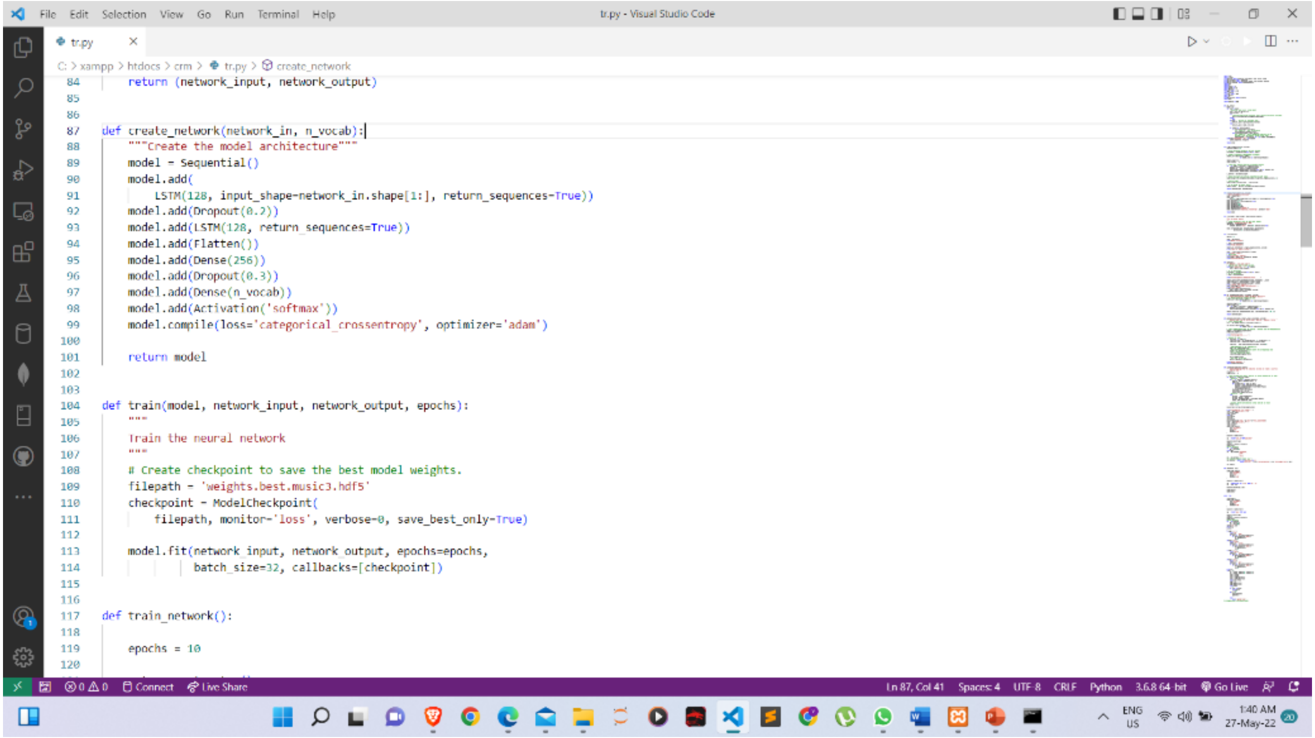

Fig. 8.2 Creating Network

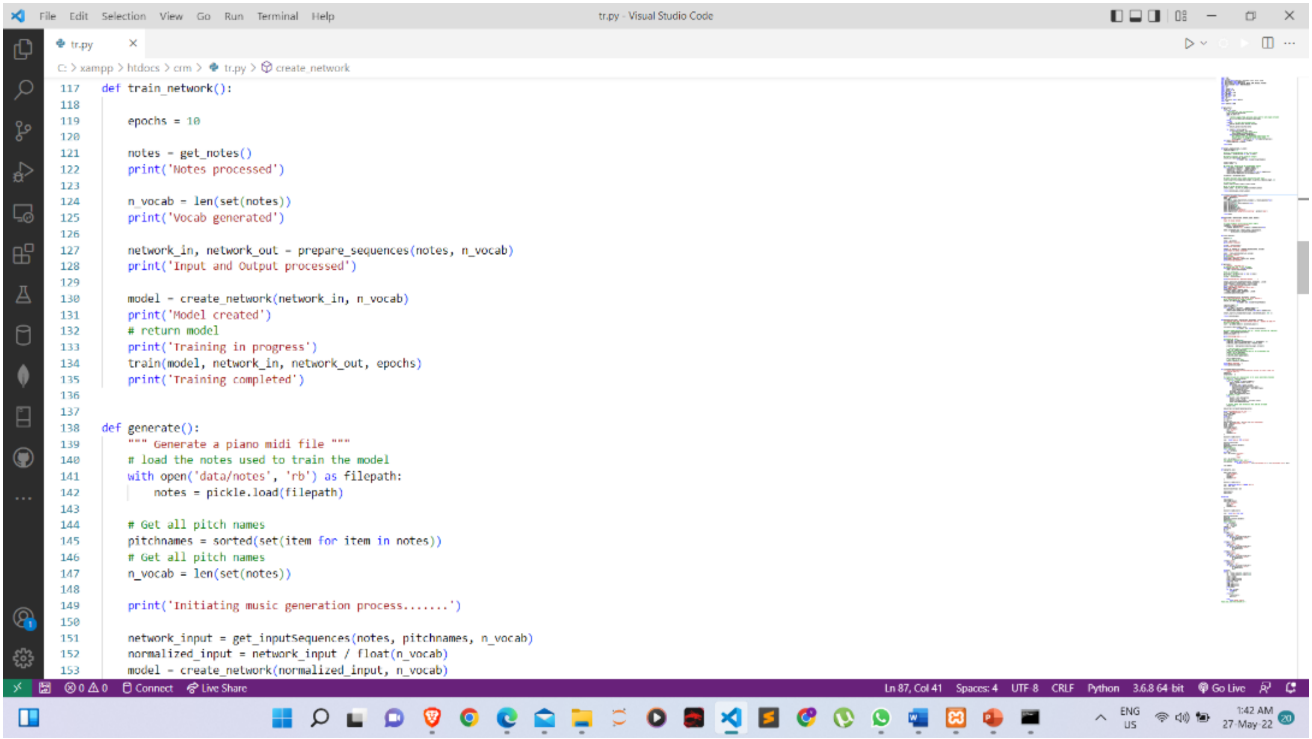

Fig. 8.3 Training Network

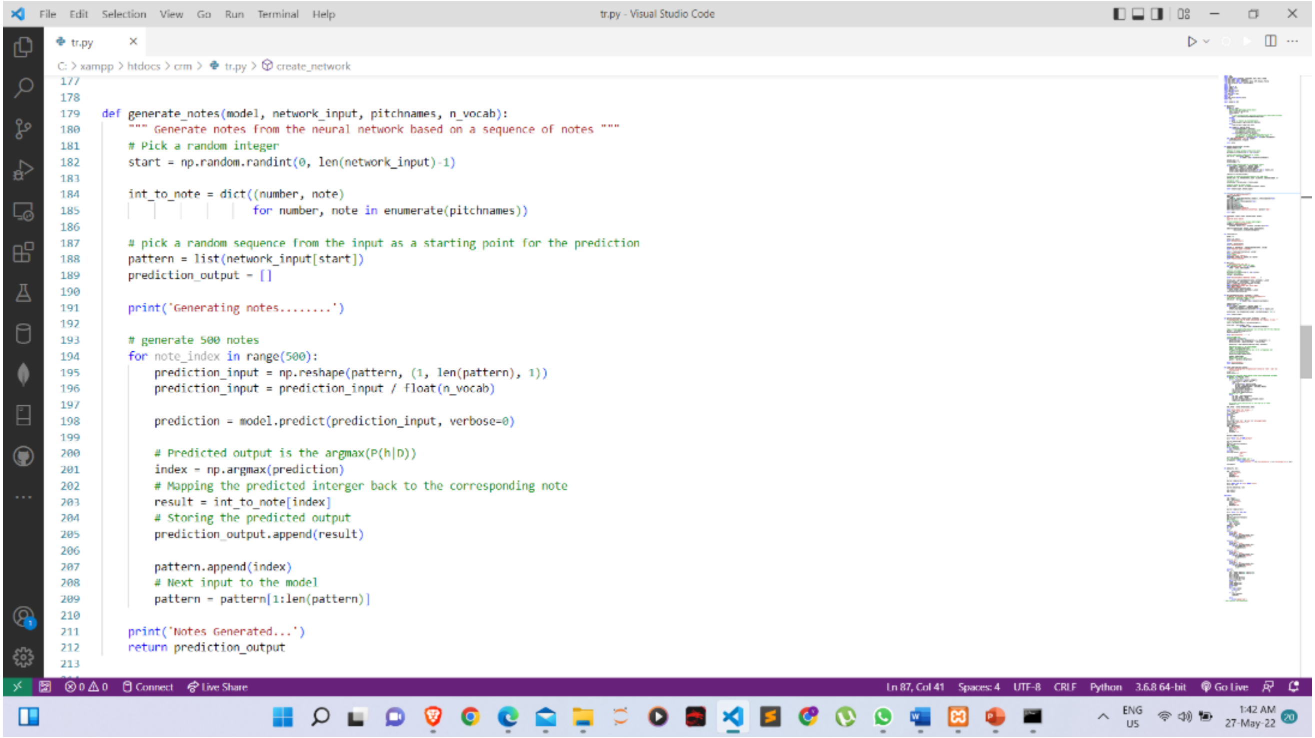

Fig. 8.4 Generating Notes

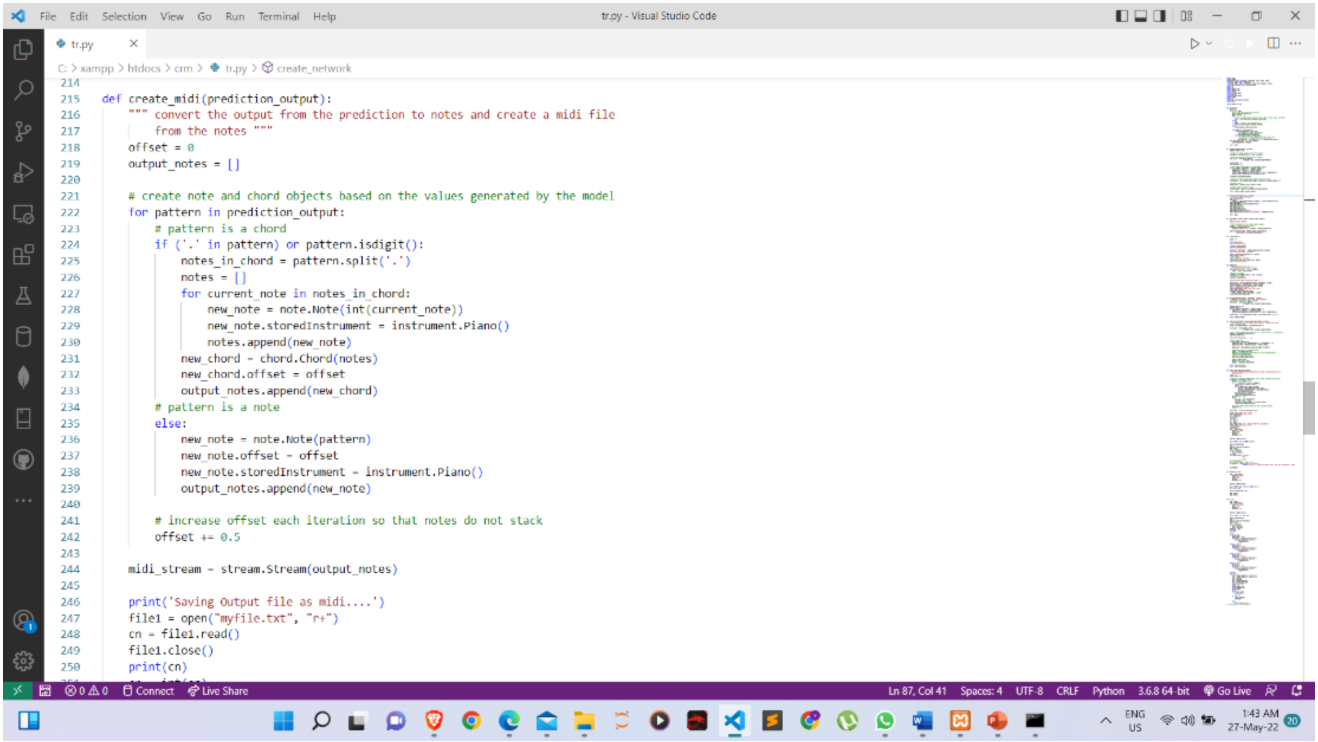

Fig. 8.5 Predicting Output

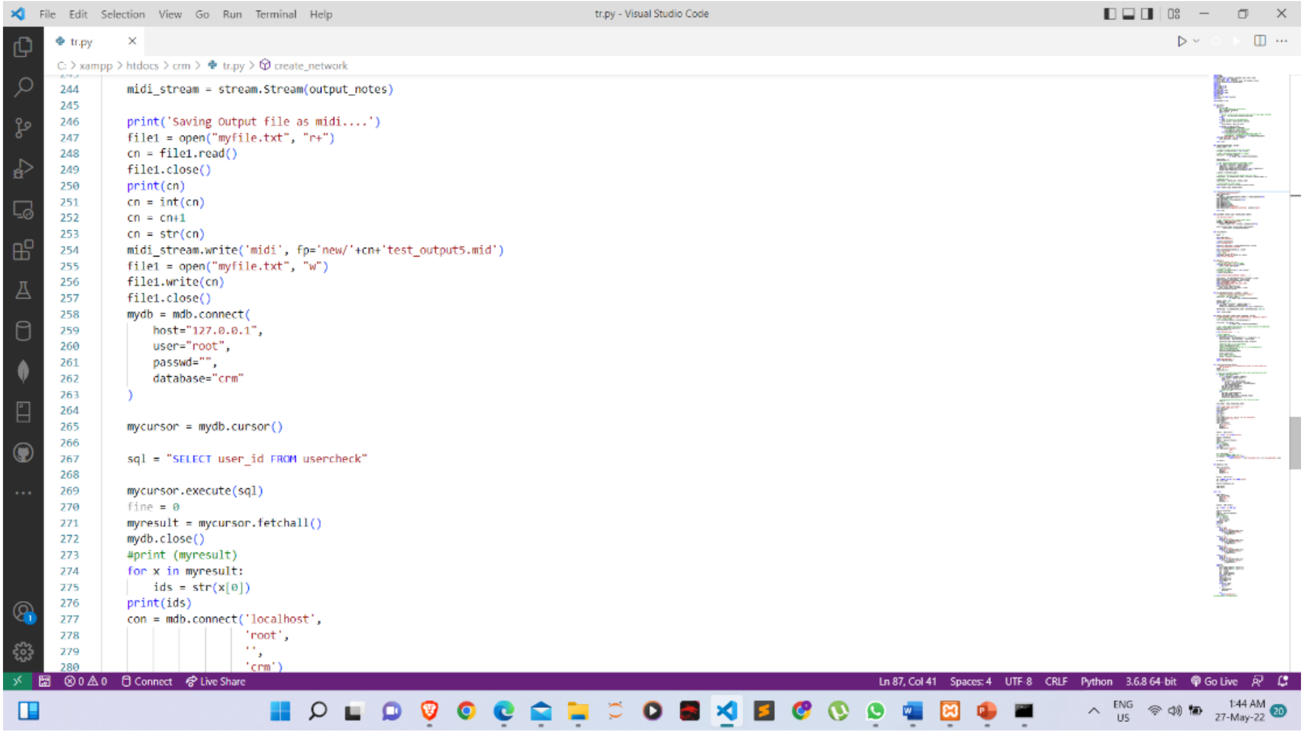

Fig. 8.6 Backend Connectivity

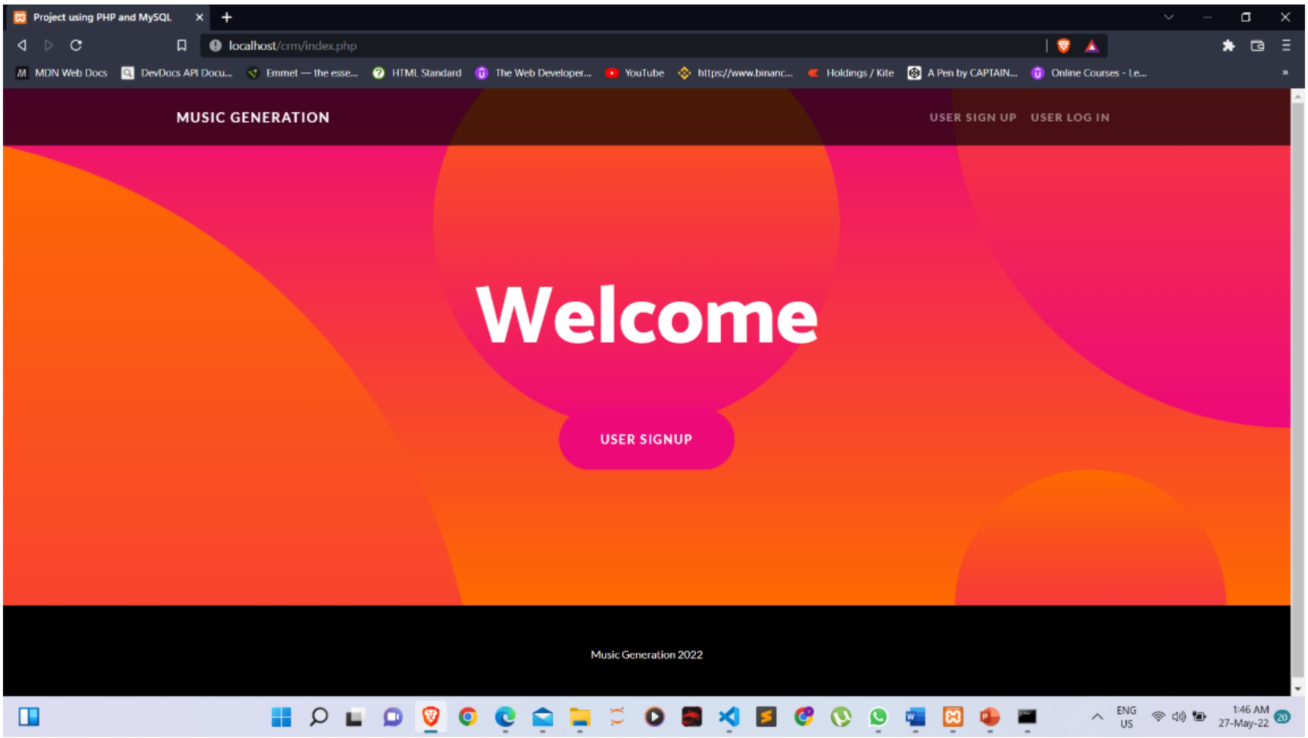

Fig. 8.7 Welcome Page

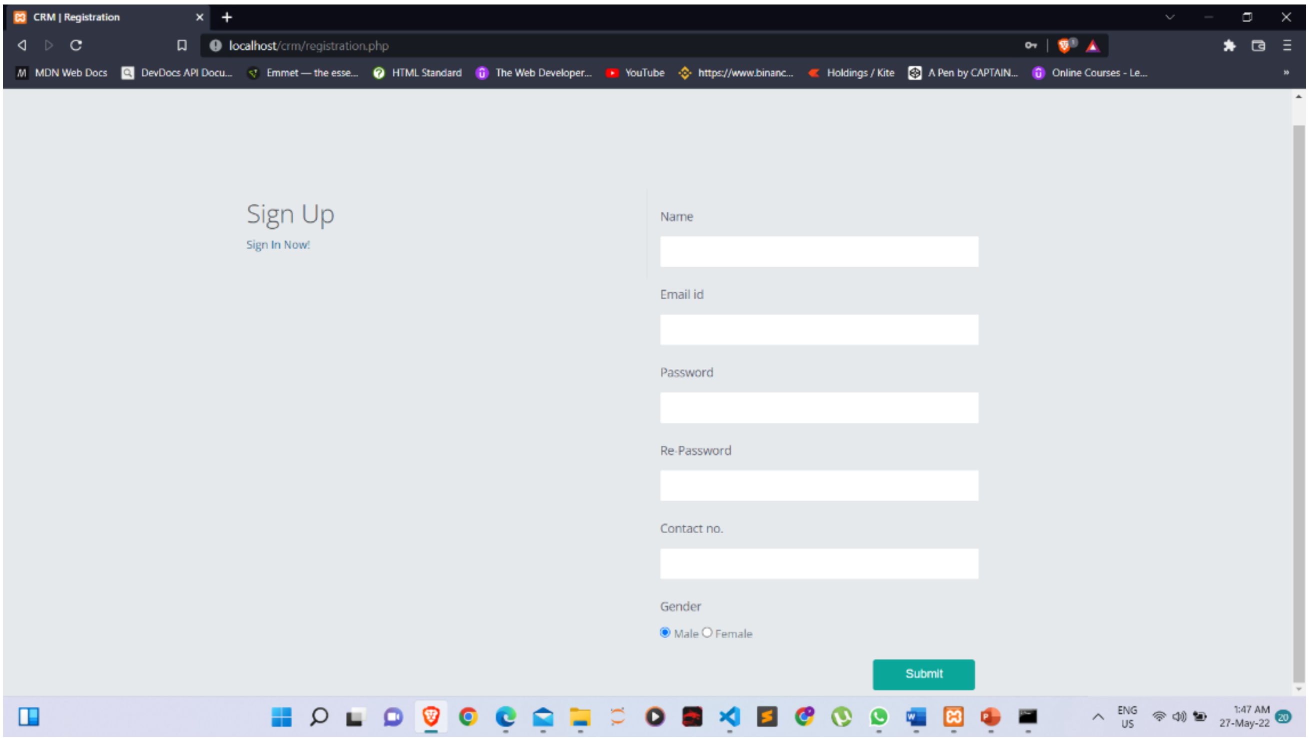

Fig. 8.8 Registration Page

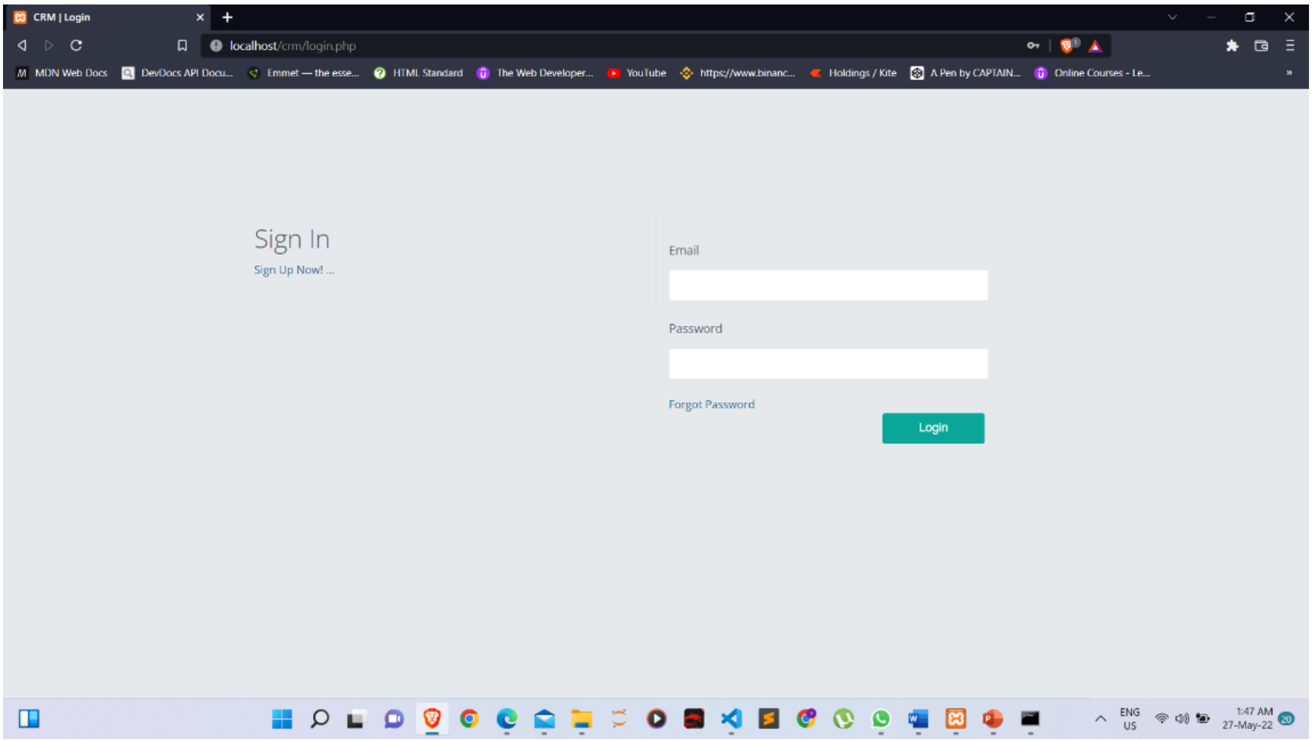

Fig. 8.9 Sign-In Page

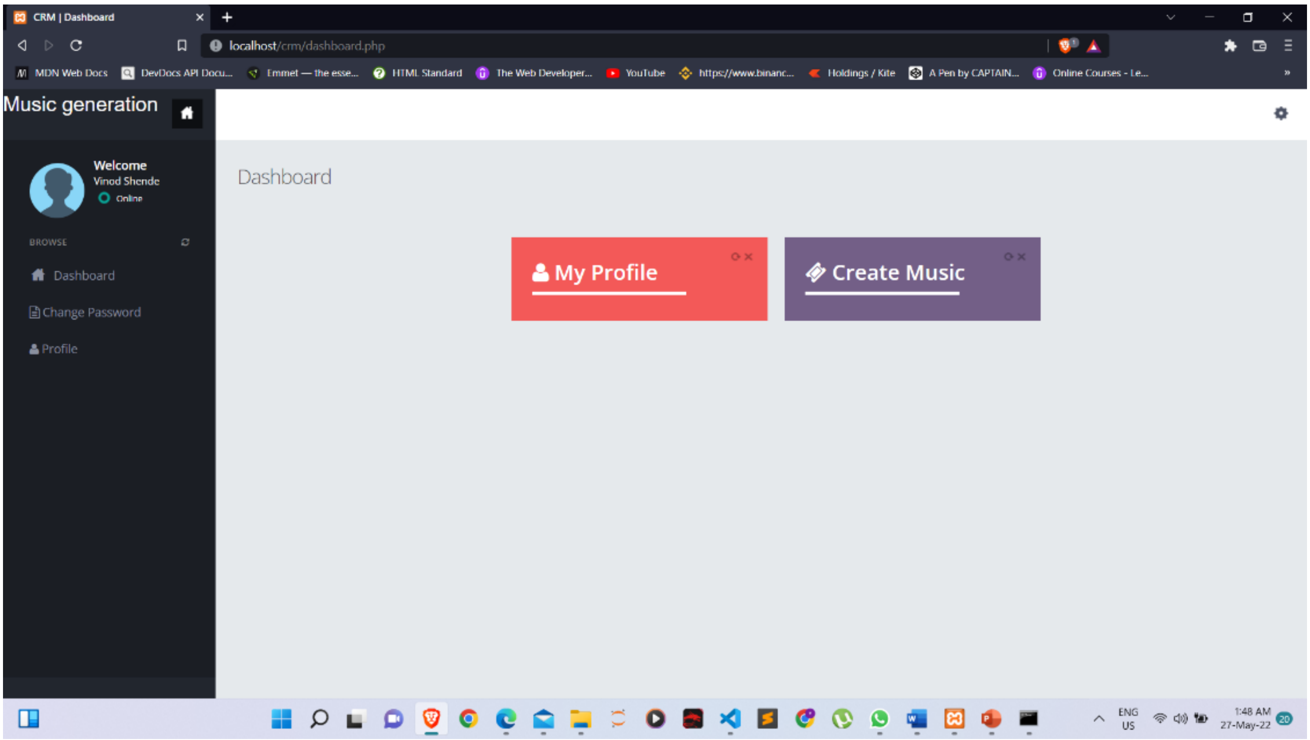

Fig. 8.10 Dashboard

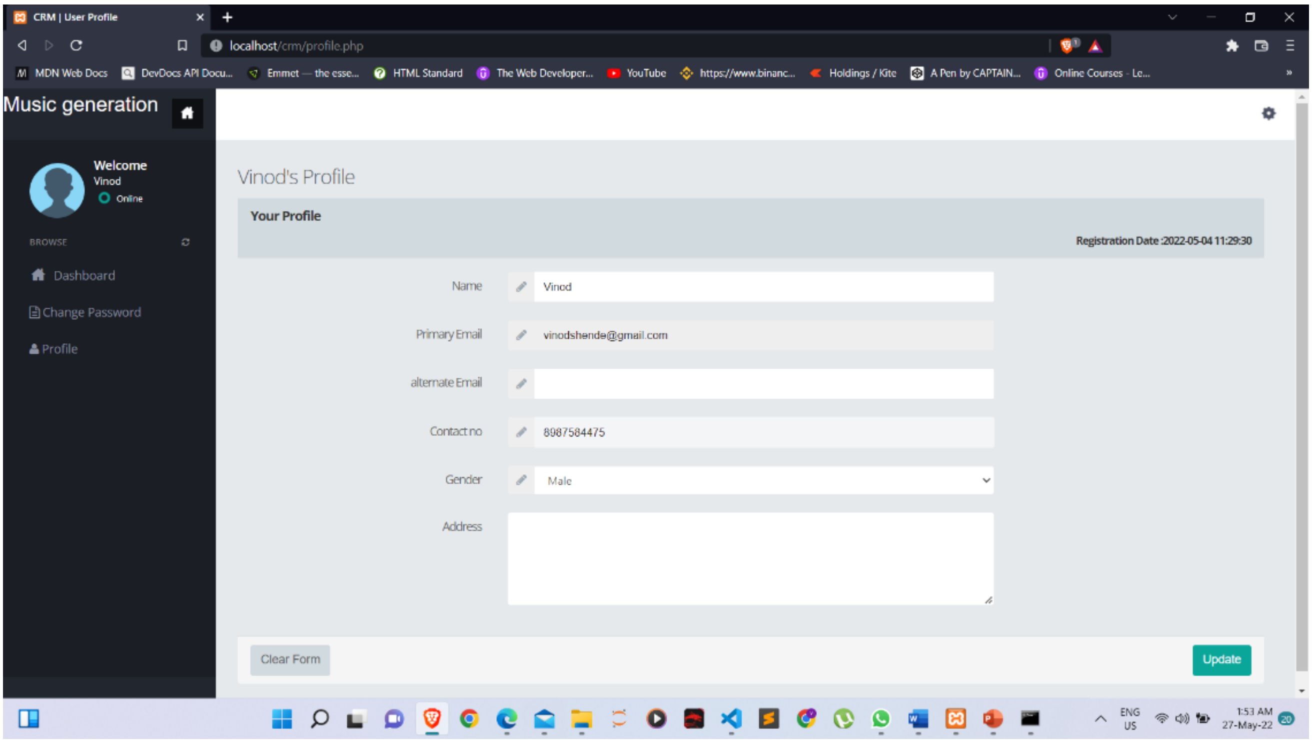

Fig. 8.11 Profile Update Page

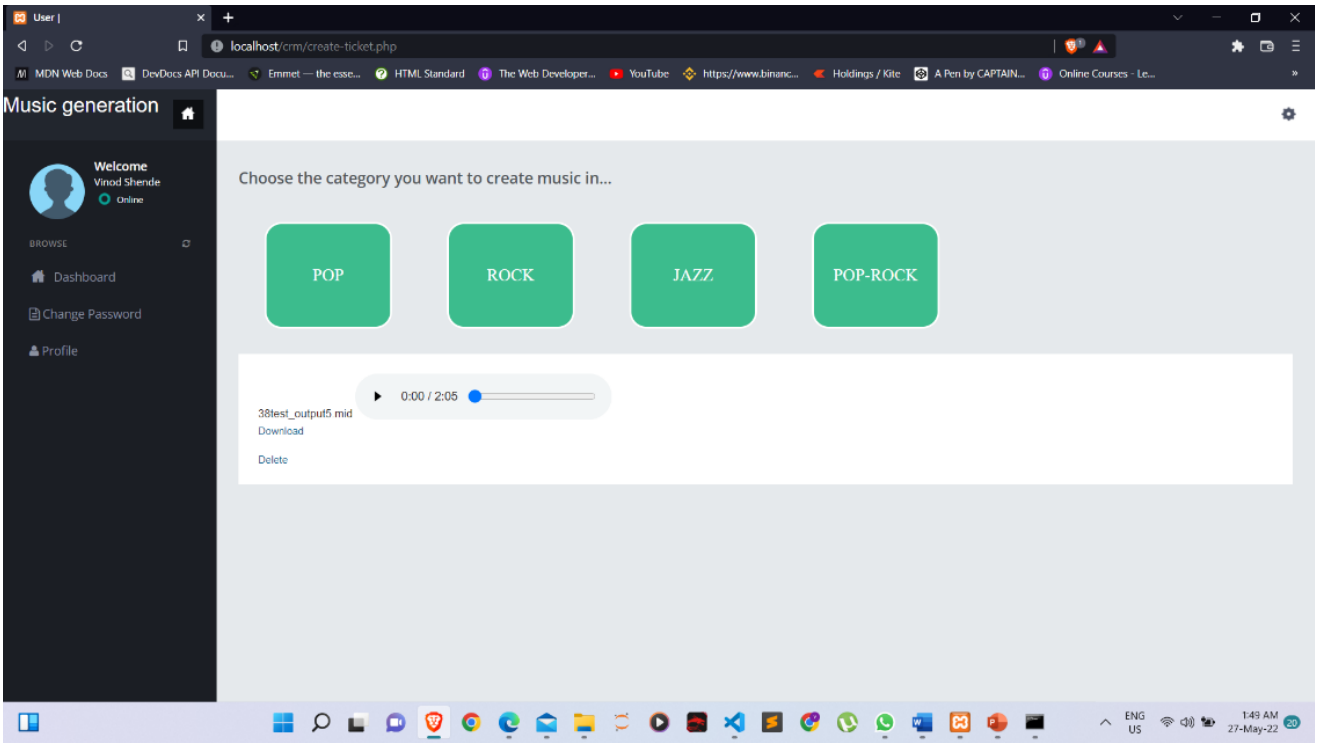

Fig. 8.12 Create Music Page with Output

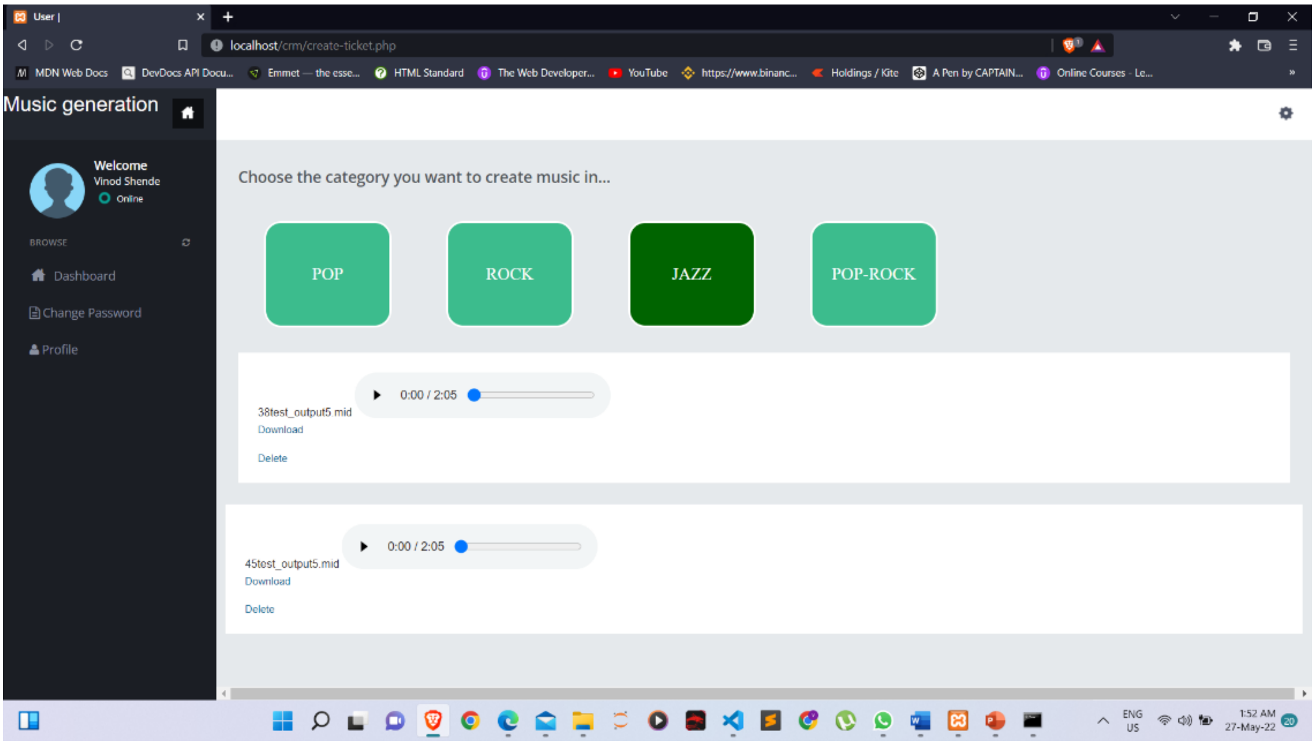

Fig. 8.13 Output 2

CHAPTER 9

CONCLUSIONS

9.1 Conclusions

This system achieves the goal of designing a model which can be used to generate music and melodiesautomatically without any human intervention. The model is capable to recall the previous details of the dataset and generate a polyphonic music using a single layered LSTM model, proficient enough tolearn harmonic and melodic note sequence from MIDI files of Pop music

9.2 Future Work

In the future, we would like to try training the model on datasets other than the collection of NES songswhich was used in this work – on soundtracks from the Super Nintendo and Sega Genesis consoles forexample. It would be interesting to see how well our model is able to learn the more complex music ofthese systems. It may even be possible to implement a system similar to that, in which we are able to learn and reproduce the different "styles" of music of each console

9.3 Applications

There are several practical and theoretical applications of music language models like ours. Perhaps one of the most common, especially in the field of music information retrieval, is for use in automatic music transcription, or AMT. AMT uses machine learning techniques in order to automatically generate a symbolic representation from an audio recording of music. This is useful if, for example, one wouldlike to study in detail the musical structure of a song or other piece of recorded music. In it is shown that music language models using LSTM networks can be used for this purpose.

Our network is designed to model a very specific kind of music early video game music and although it could theoretically be used for automatic transcription of such music, the already wide availability of video game music transcriptions makes this somewhat unnecessary.

REFERENCES

[1] Lejaren A. Hiller and Leonard M. Isaacson. Experimental Music: Composition with an Electronic Computer. McGraw-Hill, 1959.

[2] Jose D. Fern´andez and Francisco Vico. AI methods in algorithmic composition:A comprehensive survey. Journal of Artificial Intelligence Research, 2013.

[3] Hochreiter S, Schmidhuber J. Long short-term memory[J]. Neural computation, 1997, 9(8): 1735-1780.

[4] Mark Steedman. A generative grammar for Jazz chord sequences. Music Perception, 2(1):52–77, 1984.

[5] Franc¸ois Pachet, Pierre Roy, and Gabriele Barbieri. Finite-length markov processes with constraints. In Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI 2011), pages 635–642, Barcelona, Spain, July 2011.

[6] Jer´ ome Nika, Marc Chemillier, and G ˆ erard Assayag, ´ "Improtek: introducingscenarios into human-computer music improvisation," Computers inEntertainment(CIE), vol. 14, no. 2, pp. 4, 2016.

[7] David Cope and Melanie J Mayer, Experiments in musical intelligence, vol. 12, AR editions Madison, 1996.

[8] Raymond P Whorley and Darrell Conklin, "Music generation from statistical models of harmony," Journal of New Music Research, vol. 45, no. 2, pp. 160–183,2016.

[9] Mason Bretan, GilWeinberg, and Larry Heck, "A unit selection methodology for music generation using deep neural networks," arXiv preprint arXiv:1612.03789, 2016.

[10] Derek Wells and Hala ElAarag, "A novel approach for automated music composition using memetic algorithms," in Proceedings of the 49th Annual Southeast Regional Conference. ACM, 2011, pp. 155–159.

[11] Gabriele Medeot, Srikanth Cherla, Katerina Kosta, Matt McVicar, S Abdalla, M Selvi, E Rex, and K Webster, "Structure net: Inducing structure in generated melodies," in The 19th International Society for Music Information Retrieval Conference, 2018.

[12] Hang Chu, Raquel Urtasun, and Sanja Fidler, "Song from pi: A musically plausible network for pop music generation," arXiv preprint arXiv:1611.03477, 2016. [12] Daniel D Johnson, "Generating polyphonic music using tied parallel networks," in International conference on evolutionary and biologically inspired music and art. Springer, 2017, pp. 128–143.

[13] Michael Chan, John Potter, and Emery Schubert, "Improving algorithmic musiccomposition with machine learning," in Proceedings of the 9th International Conference on Music Perception and Cognition, ICMPC, 2006.

[14] Sepp Hochreiter and Jurgen Schmidhuber, “Long short- ¨ term memory,” Neural computation, vol. 9, no. 8, pp. 1735–1780, 1997.

[15] Kevin Joslyn, Naifan Zhuang, and Kien A Hua, “Deep segment hash learning for music generation,” arXiv preprint arXiv:1805.12176, 2018.

[16] Rongshu Sun, Jingiing Zhang, Wei Jiang, and Yuexin Hu, “Segmentation of pop music based on histogram clustering,” in 2018 11th International Congress on Image and Signal Processing, BioMedical Engineering and Informatics (CISP- BMEI). IEEE, 2018, pp. 1–5.

[17] Cheng-i Wang, Gautham J Mysore, and Shlomo Dubnov, “Re-visiting the music segmentation problem with crowdsourcing.,” in ISMIR, 2017, pp. 738–744.

[18] Gabriel Sargent, Fred´ eric Bimbot, and Emmanuel Vin- ´ cent, “Estimating thestructural segmentation of popular music pieces under regularity constraints,” IEEE/ACM Transactions on Audio, Speech, and Language Processing, vol. 25, no.2, pp. 344–358, 2017.

[19] Kevin Joslyn, Kai Li, and Kien A Hua, “Cross-modal retrieval using deep de- correlated subspace ranking hashing,” in Proceedings of the 2018 ACM on International Conference on Multimedia Retrieval. ACM, 2018, pp. 55–63.

[20] Sarthak Agarwal, Vaibhav Saxena, Vaibhav Singal, and Swati Aggarwal, “Lstm based music generation with dataset preprocessing and reconstruction techniques,” in 2018 IEEE Symposium Series on Computational Intelligence (SSCI). IEEE, 2018, pp. 455–462.

[21] Y Cem Subakan and Paris Smaragdis, “Diagonal rnns in symbolic music modeling,” in 2017 IEEE Workshop on Applications of Signal Processing to Audio and Acoustics (WASPAA). IEEE, 2017, pp. 354–358.

[22] Nicolas Boulanger-Lewandowski, Yoshua Bengio, and Pascal Vincent, “Modeling temporal dependencies in high-dimensional sequences: Application topolyphonic music generation and transcription,” arXiv preprint arXiv:1206.6392,2012.

Plagiarism Report