TITLE

Software Project Management Report

End-Semester Evaluation

Submitted by:

(802332098)Ashmeet Kaur

(802332016)Aruneek Kaur

(802332091)Abhishek Sharma

(802332003)Abhinav Prasher

(802332007)Akbar Ali Ahamed

ME(CSE) First Year

Team Id: C

Submitted to: Dr Harkiran Kaur (Assistant Professor)

THAPAR INSTITUTE
OF ENGINEERING & TECHNOLOGY
(Deemed to be University)

Computer Science and Engineering Department

TIET, Patiala

December 2023

TABLE OF CONTENTS

S.No.	Assignment	Page No.
1.	Short Report on assigned SWEBOK Topics and PMBOK Topics	4 - 19
2.	Gantt Chart, Network Diagram, WBS (Hybrid), Module Structure Chart, Resource List (using Open Proj/MS Project)	20 - 23
3.	SRS & Project Scope Document	24 - 32
4.	Software Project Management Plan	33 - 43
5.	WBS Dictionary Entries	44 - 46
6.	Activity on Node Diagram – CPM, PERT method	46 - 48
7.	Change Request Form	49
8.	Use Case Diagram	50
9.	Class Diagram	50
10.	Snapshots of Working Project	51 - 52
11.	Implement Earned Value Analysis (EVA) for all your project activities along with the final software product. Compute all the metrics associated with EVA for your project at 100% completion of your project respectively.	53

12.	Risk Management – Risk Table, Risk Refinement and RMMM Plan	54
13.	Compute Function Points from the Data Flow Diagram for your projects.	55
14.	Report on CASE Tool for Software Configuration Management – *Name of tool, Vendor, Date of First and Latest Release, Features, Drawbacks*	56 - 62
15.	Testing Summary Report (signed copy by Testing Team)	63

1. **Short Report on assigned SWEBOK Topics and PMBOK Topics**

SOFTWARE CONSTRUCTION

The purpose of this report is to provide some insights for the software construction process, such that we will provide how software construction is useful and a crucial step in the field of software development cycle.In this phase the design specifications are converted down to the actual useful code stuff.

The term software construction refers to the detailed creation of working software through a combination of coding, verification, unit testing, integration testing, and debugging.

SOFTWARE CONSTRUCTION FUNDAMENTALS

There are main 5 major fundamentals of the software construction such that:

- Minimizing complexity
- Anticipating change
- Constructing for verification
- Reuse
- Standards in construction.

Minimizing Complexity

- Minimize code complexity for better optimization.
- Achieve optimization through coding standards.
- These standards enhance code readability.
- Benefits include reduced errors and easier maintenance.
- Prioritize readability over overly clever code.
- Optimization scales across the project.
- Efficient development and lower maintenance costs result from optimized, readable code.

Anticipating The Change

Most Softwares are built today such that, when the platform changes , in short when the system changes then our software should withstand those changes to the fullest leading to no system or software failure.

Constructing for the Verification

This scenario means that the faults can be found by the software engineers in an easy way by the testers and the coders when they are individually searching building up the test scenarios.

This actually involves restricting the complex to hard language structure, organizing the code to support the automated testing.

Reuse

It basically means that , whatever existing features or assets are there , using them to build a unique element.

In software construction, typical assets that are reused include libraries, modules, components, source code, and commercial off-the-shelf (COTS) assets.

Reuse has literally two facets , *construction with reuse* and the *construction for reuse.*

Which means that, constructing down the software using the existing set of technologies, certain code standards and all whereas constructing anything which is totally new in nature so that it could be reused again and again in the other software production cycles that would be there in future.

Standards In Construction

There are certain aspects which come while the designing of the software, these standards are used in almost every domain of software construction leading to some standards that are developed already.

First of all, coding standards play a key role in the software development lifecycle. Just like some classes, some libraries should be written in standard format.

Secondly, Platforms also play an important role just like on which scale the meetings are being conducted and all.

Third, which programming language we are working on.

MANAGING CONSTRUCTION

Construction In Life Cycle Models

Some models are sort of linear in nature such that they are linear from the construction point of view, such as the waterfall and the staged-delivery life cycle models.

These sort of models actually treat this construction as an activity that occurs only after certain work has been completed—including detailed requirements work, extensive design work, and detailed planning

The more linear approaches tend to emphasize the activities that precede construction (requirements and design) and to create more distinct separations between activities. In these models, the main emphasis of construction may be coding.

Construction Planning

Construction method plays a key role in the construction planning activities because these actually lay the foundation of the software development cycle.Construction prerequisites are affected by the choice of the construction method, in which order they are being formed and also in which order they were performed or initialized.

PRACTICAL CONSIDERATIONS

Construction Design

Some of the projects are allocated considerable design activities to the construction, such that others allocate the design to a phase explicitly, Regardless of the exact allocation. At the construction level, some of the design work will occur.

Construction Languages

So in the software construction process, there is a kind of communication by which a human can actually execute the problem solution. These construction languages actually affect quality attributes of performance, reliability etc. Simplest type of language is the configuration language, under this option the software engineers choose from the set of attributes to work on it.

The files which are text based used up in the windows as well as in the linux are the perfect example of this.

Scripting languages are the perfect example of the languages which are used up in the application programming languages.

Similarly, programming languages are also the best example and yet the most flexible languages which are part of the construction language, as they contain least amount of the data about the particular domain set so these are quite simple in nature.

Coding

Following considerations apply to the software construction coding activity:

- Techniques for creating the code structure understandable is very important, including the source code optimisation.
- Use of classes, enumerated types, variables, named constants, and other similar entities
- Use of control structures
- Handling up the errors

Construction for Reuse

Creating software with the intention of reusing it in the future, either for the current project or other projects, is called "construction for reuse." This approach involves designing and analyzing the software to make it versatile across different systems. To prevent issues like duplicating code, it's a good idea to organize reusable code parts into well-organized libraries or components.

PROJECT PERFORMANCE DOMAIN

Project Performance Domain refers to a specific area within a project where Performance is assessed, measured and managed. The Project performance domains form a unified whole and are independent of each other which results in the successful completion of the project and its intended outcomes.

Following are the 8 Project Performance Domains:

- Stakeholders.
- Team.
- Development approach and Life cycle.
- Planning.
- Project work.
- Delivery.
- Performance.
- Uncertainty and ambiguity.

DEVELOPMENT APPROACH AND LIFE CYCLE PERFORMANCE DOMAIN

The Development Approach and Lifecycle Performance Domain is all about the *'big picture'* of how you deliver a project:

- The type of methodology you select : But not the specific methodology you'll use, nor the adaptations you make
- The general rhythm (or '*cadence*') of major events But not the day-to-day scheduling of activities
- The broad stages or phases within your project : What each stage will focus on and how you will sequence them in time

Effective execution in a project is essential in achieving desired outcomes. There are three key components to consider when it comes to optimizing project outcomes: development approach, delivery cadence, and project life cycle.

WHAT IS DELIVERY CADENCE?

The type of deliveries of projects may change from case to case. Some projects may have a single delivery, some may have multiple deliveries, and some may have periodic deliveries.

Delivery cadence refers to the timing and frequency of project deliveries.Maintaining the correct rhythm of activities is important here. Because value creation and project delivery are like a heartbeat. When the heartbeat rhythm is steady it means your project is healthy, if it has irregularities it may point out potential problems.

PROJECT LIFE CYCLE

The project life cycle is made up of five project stages: project initiation, project planning, project execution, monitoring & control and project closing. Each of these phases is necessary for the effective delivery of the project.

If you’re managing projects, you’ll need the right tools to make the process more effective and efficient.

DEVELOPMENT APPROACHES IN PROJECT MANAGEMENT

Different people may give different names to development approaches. Some organizations even may create their own approach. However the most common ones are predictive, adaptive and hybrid development approaches.

PREDICTIVE APPROACH:

The predictive approach is the most familiar one as it was the focus of former PMBOK guides. The other name of it is the waterfall approach. It is based on planning as much as possible

before performing the project activities. It is useful when we are able to define, collect and analyze requirements at the beginning of a project.

ADAPTIVE APPROACH: When uncertainty exists at the beginning of a project, then the adaptive approach is the most applicable one. In this type of project, requirements change frequently and the project needs to adapt itself to changing requirements. There are two iterative approaches as iterative and incremental.

HYBRID APPROACH: It is simply appointed somewhere in the middle of predictive and adaptive approaches. Even if it is more adaptive than being predictive most times, it still has some characteristics of a predictive approach.

CONSIDERATIONS FOR SELECTING A DEVELOPMENT APPROACH

The criteria can be divided into three categories: product, service, or result; project; and organization

Product, Service, or Result	Project	Organization
• Degree of Innovation • Requirements Uncertainty • Scope Stability • Ease of Change • Delivery Options • Risk • Requirements and Regulations	• Stakeholders • Schedule Constraints • Funding Availability	• Organizational Structure • Culture • Organizational Capability • Project Team Size and Location

Product, Service, or Result

The factors influencing the product, service, or result consideration all have to do with the nature of a project's outcome, whether it is a product, a service, or another type of result.

Degree of Innovation

Deliverables that have a well-understood scope and requirements, that the project team has worked with before, and that allow for planning up front are well suited to the predictive approach. Deliverables involving a high degree of innovation or those with which the project team does not have experience are better suited to a more adaptive approach.

Requirements Certainty

A predictive approach works well when the requirements are well known and easy to define.

When requirements are uncertain, volatile, or complex and are expected to evolve throughout the project, a more adaptive approach is a better fit.

Scope Stability

If the scope of the deliverable is stable and not likely to change, a predictive approach is practical. If the scope is expected to undergo many changes, an approach that is closer to the spectrum's adaptive side can be helpful.

Ease of Change

Related to requirements certainty and scope stability, if the nature of the deliverable makes it

challenging to manage and incorporate changes, then a predictive approach is best. Deliverables that can quickly adapt to change can use an adaptive approach.

Delivery Options and Cadence

The nature of the deliverable, such as whether it can be delivered in components, influences the development approach. Products, services, or results that can be developed and delivered in pieces are aligned with incremental, iterative, or adaptive approaches.

Risk

You can reduce risk by building products modularly and adapting the design and development based on learning to take advantage of emerging opportunities or reduce the exposure to

threats. Adaptive approaches frequently mitigate high-risk requirements by addressing their viability first.

Safety Requirements and Regulations

Products that have rigorous safety requirements often use a predictive approach because

significant up-front planning is needed to ensure that all the safety requirements are identified, planned for, created, integrated, and tested. Likewise, environments that are subject to significant regulatory oversight may need a predictive approach due to the required process, documentation, and demonstration needs.

Project

The factors in the project consideration column all have to do with aspects of the project, such as how it is structured, constrained, and funded.

Stakeholders

Specific stakeholders, such as product owners, play a substantial role in establishing requirements from the customer's perspective and prioritizing work. If such dedicated project team staff are available to support project work, adaptive methods are preferred.

Schedule Constraints

If there is a need to deliver something early, even if it is not a finished product, an adaptive approach is beneficial.

Funding Availability

Projects that work in an environment of funding uncertainty can benefit from an adaptive

approach. A minimum viable product can be released with less investment than an elaborate product. This allows for market testing or market capture with minimum investment.

Organization

The factors in the organization consideration column all have to do with the organizational environment of the project, including the culture, structure, and complexity.

Organizational Structure

An organizational structure with many levels, a rigid reporting structure, and substantial bureaucracy is likely to use a predictive approach. In contrast, projects that use adaptive techniques are associated with a flat structure.

Culture

A predictive approach fits better in an organization with a culture of managing and directing.

Here the work is planned out and progress is measured against baselines. Adaptive approaches fit better within an organization that emphasizes project team self-management.

Organizational Capability

If organizational policies embrace attitudes and beliefs that support an agile mindset, then adaptive methods are likely to succeed.

Project Team Size and Location

Adaptive approaches often work best with project teams of five to nine individuals. Adaptive approaches also favor project teams that are located in the same physical space.

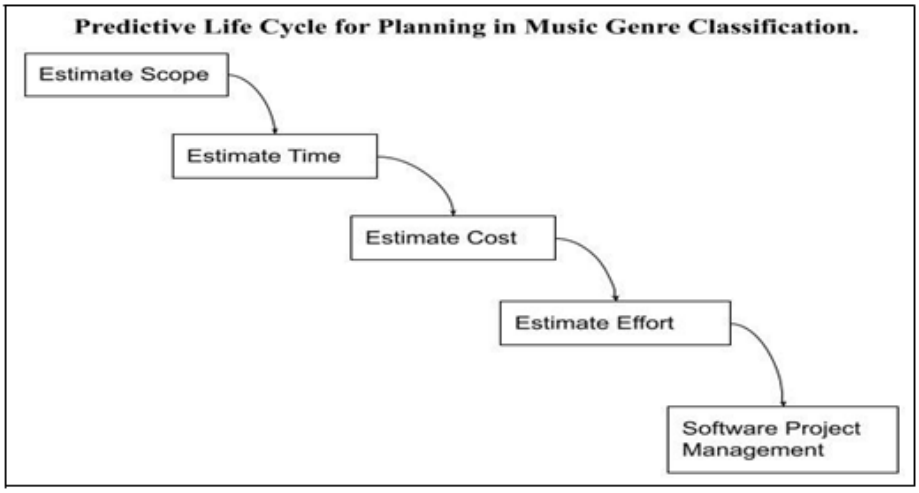
LIFE CYCLE AND PHASE DEFINITIONS

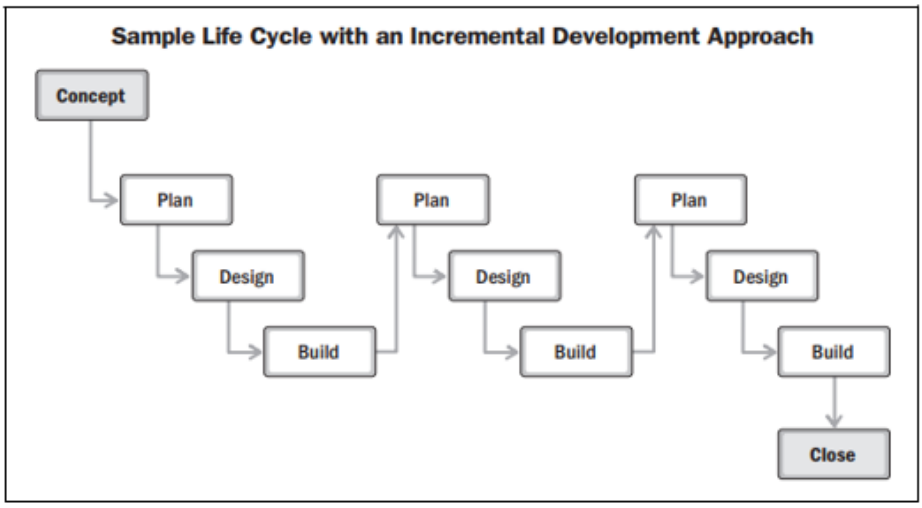
The type and number of the project's phases during its life cycle are dependent on a range of factors, The process of delivery and development approach, as described before, are the chief among them. The following examples of stages in the life cycle can be found:

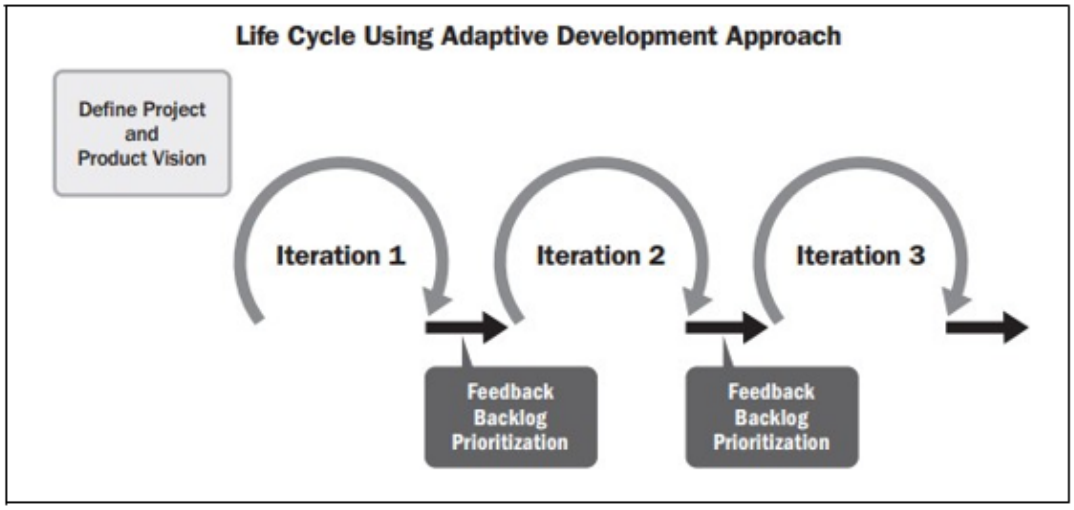
- **Feasibility** This stage will determine whether a business case can be proven and if an organization is capable of delivering the desired outcome.
- **Design** The development of the project deliverables which will be drawn up is guided by planning and analysis.

- **Build** Construction of the deliverable with integrated quality assurance activities is conducted.
- **Test** Before the transition, go live, or acceptance by the customer, the final quality review and inspection of the deliverables shall be carried out.
- **Deploy** Project deliverables are put into use and transitional activities required for sustainment, benefits realization, and organizational change management are completed.
- **Close** The project is closed, project knowledge and artifacts are archived, project team members are released, and contracts are closed.

In order to check that the desired outcomes or exit criteria have been met before proceeding into the following phase, project stages are often accompanied by a stage gate review also known as Phase Gate Review. The exit criteria may relate to acceptance conditions for deliverables, contractual obligations and achieving specific performance objectives or other tangible measures.


Predictive Life Cycle for Planning in Music Genre Classification.
Estimate Scope
Estimate Time
Estimate Cost
Estimate Effort
Software Project Management


Sample Life Cycle with an Incremental Development Approach
Concept
Plan
Design
Build
Plan
Design
Build
Plan
Design
Build
Close


Life Cycle Using Adaptive Development Approach
Define Project and Product Vision
Iteration 1
Iteration 2
Iteration 3
Feedback Backlog Prioritization
Feedback Backlog Prioritization

ALIGNING OF DELIVERY CADENCE, DEVELOPMENT APPROACH, AND LIFE CYCLE

Delivery Cadence and Development Approach

Deliverable	Delivery Cadence	Development Approach
Building	Single delivery	Predictive
Senior services	Multiple deliveries	Iterative
Website	Periodic deliveries	Adaptive
Community action patrol training	Multiple deliveries	Incremental

Based on this information, a potential life cycle might be:

1. **Start Up**
    - Entry Criteria : Business case approval and project charter signing are required entry criteria. These criteria signify the formal initiation of the project.
    - Activities in the Initiation Phase: A high-level project roadmap is developed. Initial funding requirements are specified, The project team and necessary resources are identified.
    - Completion of Deliverables : Before concluding the initiation phase, specific deliverables must be completed. These deliverables typically include the development of the project roadmap, defining initial funding needs, forming the project team, and establishing a progress schedule.
    - Exit Criteria and Origination Phase Gate Review : Exit criteria are the conditions that must be met before proceeding from one project phase to the next. In this case, exit criteria are reviewed at an origination phase gate review
2. **Plan** :

    This stage involves the decomposition of building's complex information into concrete plans. A detailed design document for the CAP training is completed. Together with a gap analysis, an analysis of the senior services offering is carried out. The first frame of the website is created. By the end of the planning phase, these outputs should have been completed. The exit criteria shall be revised as part of a gate review at the planning stage.

3. **Development**

Since the deliverables are varied in terms of their delivery timetable and different approaches, this phase will be accompanied by test and deployment phases. Early

deliveries will be made on the website, giving information to the public about progress at the Community Centre. Before the opening of the Community Centre, certain senior functions and CAP training may start to take place. Before going to the test stage, each delivery can have its own review.

4. **Test**

Overlaps with development and deploy phases. The type of testing depends on the specific deliverable. Includes inspections for building, beta delivery of CAP courses, small-scale trials for senior services, and operating in a test environment for website releases. Each deliverable undergoes relevant testing before moving to the deploy phase.

5. **Deploy**

Overlaps with development and test phases. Initial website deployment may occur early in the project. Activities evolve as more deliverables become available. Final deployment is the opening of the community center. Ongoing updates for the website and senior services become part of operations after the community center opens.

6. **Close.**

Occurs periodically as deliverables are completed. Upon initial website deployment, project personnel (including contractors) are released, and retrospectives or lessons learned are conducted. At project completion, phase gate reviews and an overall evaluation of project performance compared to baselines take place.

Prior to final closeout, the project charter and business case are reviewed to determine if intended benefits and value have been achieved.

INTERACTIONS WITH OTHER PERFORMANCE DOMAINS

Interaction with Stakeholder Domain

- The Development Approach and Life Cycle Performance Domain interact withm the Stakeholder Domain.
- The chosen life cycle impacts planning.
- Predictive life cycles involve extensive upfront planning and use rolling wave planning and progressive elaboration.
- Plans are updated as threats and opportunities arise.

Interaction with Uncertainty Domain

- Development approach and delivery cadence help reduce uncertainty in projects.
- For deliverables with regulatory risk, a predictive approach may be chosen to incorporate extra testing, documentation, and robust processes.
- For stakeholder acceptance risk, an iterative approach with a minimum viable product is chosen for feedback before further development.

Interaction with Delivery Performance Domain

- There is significant overlap with the Delivery Performance Domain particularly concerning delivery cadence and development approach.
- Delivery cadence is crucial for delivering value in alignment with the business case and benefits realization plans.
- Product and quality requirements from the Delivery Performance Domain influence the development approach.

Interaction with Team Performance Domain

- Interaction occurs regarding project team capabilities and leadership skills.
- The development approach influences the project team's way of working and the project manager's style.
- A predictive approach emphasizes upfront planning, measurement, and control.

- An adaptive approach, especially with agile methods, requires a servant leadership style and may involve self-managing project teams.

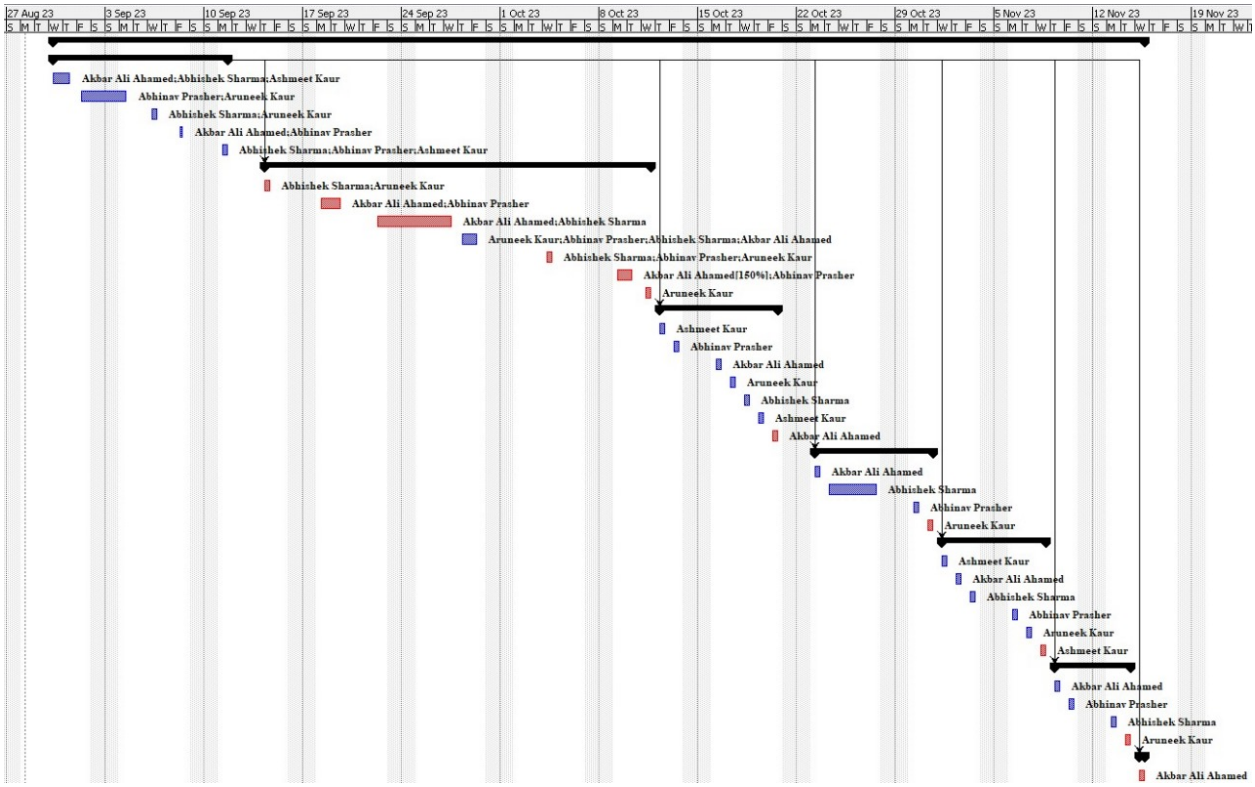
MEASURING OUTCOMES

Checking Outcomes—Development Approach and Life Cycle Performance Domain

1. **Developing methods that correspond to the project deliverables :** The development approach for deliverables (predictive, hybrid, or adaptive) reflects the product variables and is appropriate given the project and organizational variables.
2. **The project's life cycle is divided into stages in which the supply of business and stakeholder value takes place from beginning to end of a project. :** In the project phases, work from inception to closure shall be taken into account. Appropriate exit criteria are included in the phases.
3. **Phases of the project's life cycle which facilitate delivery pace and development in order to produce Project deliverables:** The development, testing and deployment times are indicated in the life cycle stages. The overlaps or repetitions of phases shall be used to represent projects with a number of deliverables that vary in delivery schedules and development methods, where appropriate.

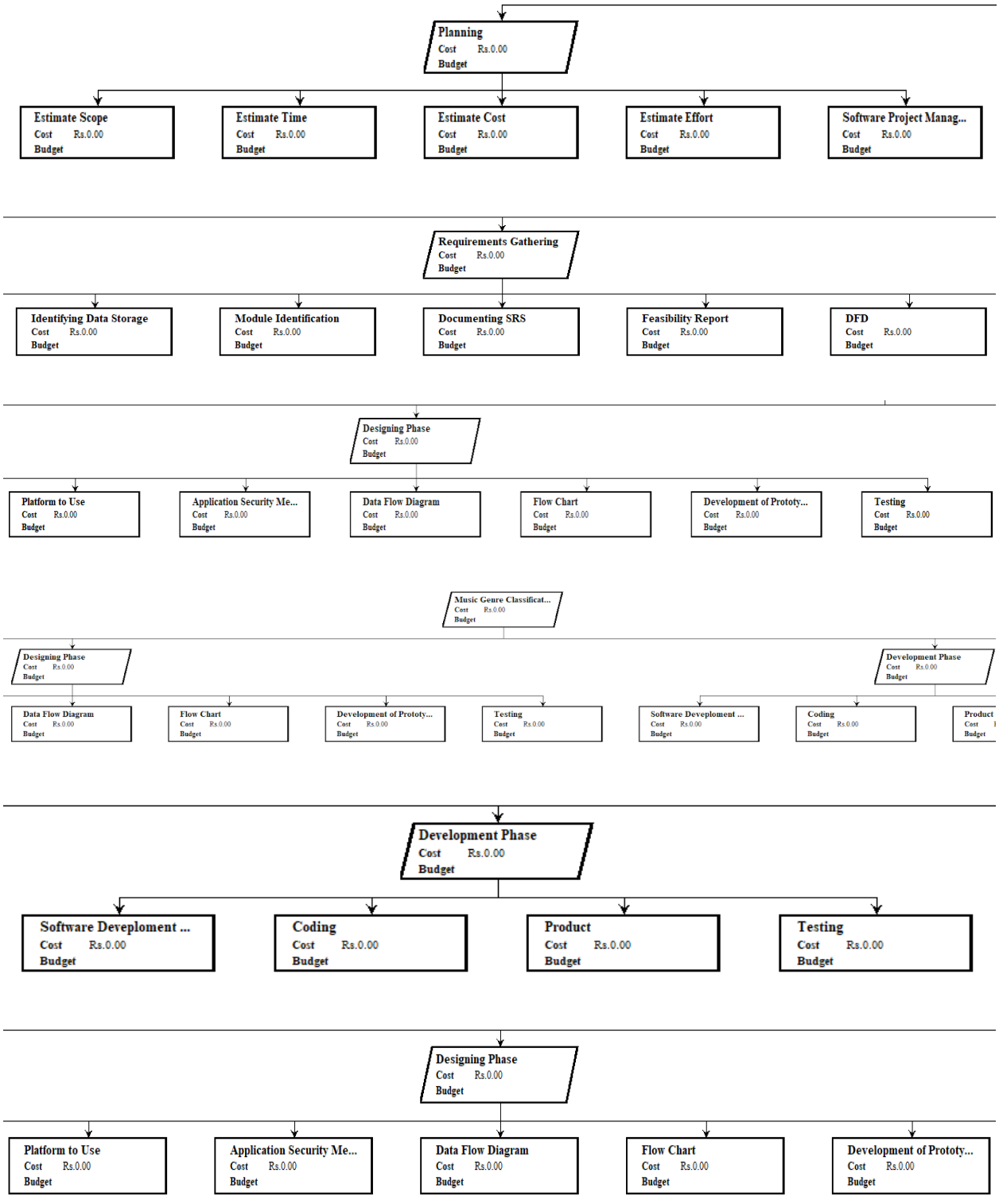
2. Gantt Chart, Network Diagram, WBS (Hybrid), Module Structure Chart, Resource List (using Open Proj/MS Project)

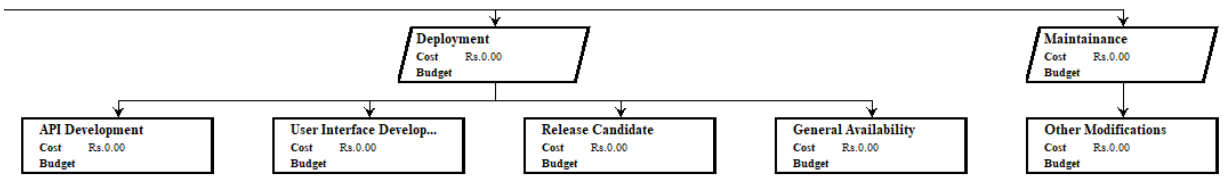
- Gantt Chart



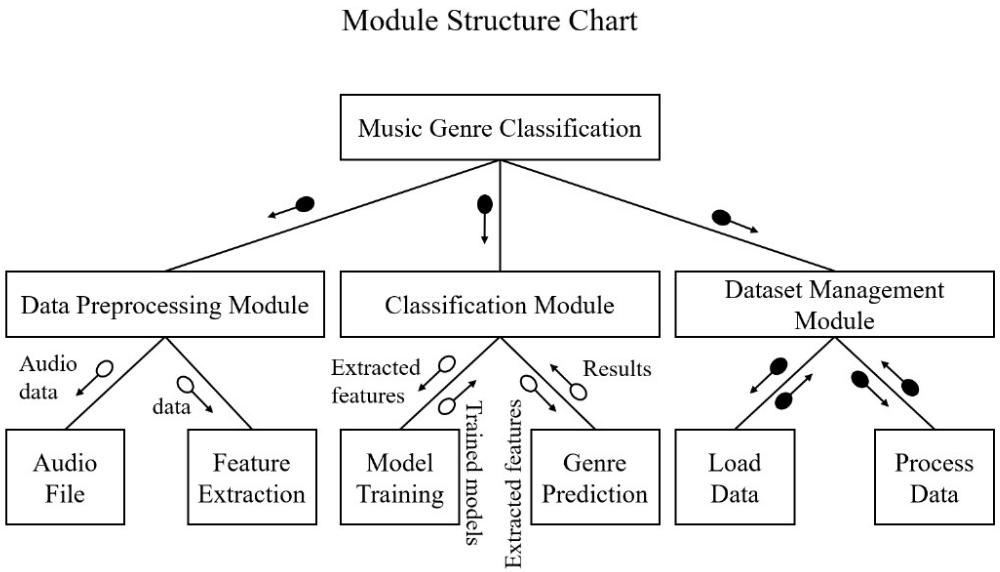
- **Network Diagram**

- **Work Break Down Structure**


Planning
Cost Rs.0.00
Budget
Estimate Scope
Cost Rs.0.00
Budget
Estimate Time
Cost Rs.0.00
Budget
Estimate Cost
Cost Rs.0.00
Budget
Estimate Effort
Cost Rs.0.00
Budget
Software Project Manag...
Cost Rs.0.00
Budget
Requirements Gathering
Cost Rs.0.00
Budget
Identifying Data Storage
Cost Rs.0.00
Budget
Module Identification
Cost Rs.0.00
Budget
Documenting SRS
Cost Rs.0.00
Budget
Feasibility Report
Cost Rs.0.00
Budget
DFD
Cost Rs.0.00
Budget
Designing Phase
Cost Rs.0.00
Budget
Platform to Use
Cost Rs.0.00
Budget
Application Security Me...
Cost Rs.0.00
Budget
Data Flow Diagram
Cost Rs.0.00
Budget
Flow Chart
Cost Rs.0.00
Budget
Development of Prototy...
Cost Rs.0.00
Budget
Testing
Cost Rs.0.00
Budget
Music Genre Classificat...
Cost Rs.0.00
Budget
Designing Phase
Cost Rs.0.00
Budget
Development Phase
Cost Rs.0.00
Budget
Data Flow Diagram
Cost Rs.0.00
Budget
Flow Chart
Cost Rs.0.00
Budget
Development of Prototy...
Cost Rs.0.00
Budget
Testing
Cost Rs.0.00
Budget
Software Deveploment ...
Cost Rs.0.00
Budget
Coding
Cost Rs.0.00
Budget
Development Phase
Cost Rs.0.00
Budget
Software Deveploment ...
Cost Rs.0.00
Budget
Coding
Cost Rs.0.00
Budget
Product
Cost Rs.0.00
Budget
Testing
Cost Rs.0.00
Budget
Designing Phase
Cost Rs.0.00
Budget
Platform to Use
Cost Rs.0.00
Budget
Application Security Me...
Cost Rs.0.00
Budget
Data Flow Diagram
Cost Rs.0.00
Budget
Flow Chart
Cost Rs.0.00
Budget
Development of Prototy...
Cost Rs.0.00
Budget



- **Module Structure Chart**



- **Resource List**

		Name	RBS	Type	E-mail Address	Material Label	Initials	Group	Max. Units
1		Akbar Ali Ahamed		Work	aahamed_me23@thapar.edu		A		100%
2		Abhishek Sharma		Work	asharma4_me23@thapar.edu		S		100%
3		Abhinav Prasher		Work	aprasher_me23@thapar.edu		P		100%
4		Aruneek Kaur		Work	akaur_me23@thapar.edu		K		100%
5		Ashmeet Kaur		Work	akaur1_me23@thapar.edu		H		100%

3. SRS & Project Scope Document

Introduction

Purpose of this Document

Music genre classification serves several valuable purposes in the field of music analysis and recommendation. The primary objectives of music genre classification include:

- **Music Recommendation:** Music streaming platforms and recommendation systems use genre classification to provide personalized playlists and song recommendations to users. By categorizing songs into genres, the system can suggest music that aligns with the user's musical taste, enhancing the overall listening experience.
- **Content Organization:** Music libraries, both personal and commercial, can benefit from genre classification to organize their vast collections efficiently. This enables users to navigate through music libraries with ease, searching for and discovering music that suits their mood or preferences.
- **Market Segmentation:** In the music industry, genre classification helps identify and target specific listener demographics. Record labels, artists, and event organizers use this data to tailor their marketing and promotional efforts to reach the appropriate audience for a particular genre.
- **Music Analysis:** Researchers and musicologists use genre classification to analyze musical trends and patterns. This aids in understanding the evolution of music over time, tracking the popularity of specific genres, and exploring the cultural and historical context of music.
- **Enhanced Music Discovery:** Genre classification allows users to discover new music within their preferred genres and related styles. It promotes music exploration and can lead to the discovery of artists and tracks that might otherwise remain hidden.
- **Curated Playlists:** Music streaming services and radio stations use genre classification to create curated playlists, themed radio stations, and genre-specific channels, delivering a more immersive and engaging listening experience.
- **Music Licensing and Rights Management:** In the context of licensing and royalty distribution, genre classification helps ensure that artists and content creators receive fair

compensation for their work when their music is used in various media, such as films, commercials, and video games.

- **Automated Content Tagging:** Music platforms and content management systems employ genre classification to automatically tag and categorize newly uploaded music. This simplifies the process of metadata assignment, making music search and retrieval more efficient.
- **Educational and Historical Analysis:** Music educators and historians can use genre classification to teach students about the rich diversity of musical styles across different eras and cultures. This supports the preservation and dissemination of musical heritage.
- **Artistic Exploration:** Musicians and composers can draw inspiration from genre classification by exploring different genres and styles, experimenting with hybrid genres, and pushing the boundaries of musical creativity.

In summary, music genre classification plays a pivotal role in the modern music ecosystem, enhancing user experiences, supporting the music industry, enabling research, and contributing to the broader understanding and appreciation of the art of music.

Scope of the Development Project

In the context of developing a music genre classification system, the scope of the project encompasses various critical dimensions that need to be considered for its successful implementation. These dimensions include:

- **Functionality:** The system's functionality pertains to its ability to accurately categorize music into distinct genres based on audio features, lyrics, and other relevant data. It should encompass a wide range of musical genres and be capable of adapting to new and emerging genres. Additionally, the system's functionality must account for scalability to accommodate a growing and evolving music library.

- **Efficiency:** Efficiency in music genre classification involves optimizing the algorithm's resource usage, such as CPU and memory, to process and classify music tracks swiftly. This includes ensuring that the classification process doesn't consume excessive

computational resources and can be deployed in real-time applications without significant delays.

- **Performance:** The system's performance is measured by how quickly it can analyze and classify a large dataset of music tracks. It should also provide efficient response times for user queries, enabling users to access genre-specific playlists and recommendations promptly.

- **Reliability:** Reliability is paramount in music genre classification. The system should consistently produce accurate genre labels for various music tracks, even when dealing with noisy or incomplete data. It should be robust against variations in audio quality, song length, and stylistic diversity, ensuring dependable results.

- **Usability:** The usability aspect involves designing a user interface that is intuitive and user-friendly. Users should be able to interact with the system effortlessly, whether they are casual listeners, music enthusiasts, or professionals in the music industry. The interface should facilitate easy navigation, music discovery, and customization of playlists.

- **Availability:** Availability refers to the system's ability to be accessible to users whenever they require its services. Music genre classification should be available and responsive as promised, offering uninterrupted access to categorized music libraries, personalized recommendations, and genre-specific content.

- **Adaptability:** The system should be adaptable to changes in the music landscape, such as evolving genres and subgenres, emerging artists, and shifts in musical trends. It should be designed to incorporate new data sources and features that enhance genre classification accuracy.

- **Interoperability:** Interoperability is crucial for integrating the music genre classification system with various music platforms, applications, and databases. It should support APIs

and data exchange formats that enable seamless interaction with other music-related services.

- **Data Security:** The project should address data security and privacy concerns, especially when dealing with user-generated content. Protecting user data and ensuring the confidentiality of sensitive information is paramount.

In summary, the scope of the music genre classification development project encompasses the functional, efficiency, performance, reliability, usability, availability, scalability, adaptability, interoperability, and data security aspects required to create a robust and user-friendly system for classifying and accessing music genres.

Overview

Music genre classifier is a kind of data driven approach , under this we will be using a certain machine learning algorithm to categorize the music tracks into their individual genres. This technique has a variation in its applications including the following:

- **Understanding User Preferences:**Music genre classification provides valuable insights into how users group and categorize music. This understanding enables music platforms, artists, and record labels to create more targeted marketing campaigns, allowing them to tailor their content and promotions to specific listener preferences.

- **Providing Personalized Recommendations:** The application of music genre classification is instrumental in generating personalized music recommendations for users. By identifying the genres that resonate with a listener based on their listening history, music platforms can offer curated playlists, artist suggestions, and song recommendations that align with individual tastes and preferences.

- **Enhancing Music Discovery:** Music genre classification plays a pivotal role in helping users discover new music within their preferred genres and related styles. It facilitates the

creation of genre-specific playlists and thematic radio stations, enriching the overall music exploration experience.

- **Supporting Content Licensing and Rights Management:** Accurate genre classification is crucial for music licensing and rights management. It ensures that artists, songwriters, and copyright holders receive fair compensation when their music is used in various media, including films, commercials, and video games. Accurate genre labeling is key to ensuring that music is appropriately licensed and attributed.

- **Enabling Cultural and Historical Analysis:** Researchers, musicologists, and educators use music genre classification to study the evolution of music, track genre popularity over time, and explore the cultural and historical context of musical styles. This supports a deeper understanding of the art form and its societal impact.

- **User-Driven Playlist Creation:** Music genre classification enables users to create their own playlists and organize their music libraries effectively. By categorizing tracks into genres, users can curate their own music collections, fostering a more personalized and organized listening experience.
- **Industry Insights:** For the music industry, genre classification provides valuable insights into market segmentation. Record labels, artists, and event organizers can tailor their promotional strategies, music releases, and live performances to target specific listener demographics.

- **Efficient Content Management:** Music libraries and content management systems benefit from the organization and categorization provided by genre classification. It simplifies the process of metadata assignment, content tagging, and search and retrieval, leading to more efficient content management.

- **Scalability and Adaptability:** The world of music is constantly evolving with new genres and subgenres emerging. Music genre classification systems must be scalable and

adaptable to incorporate these changes, ensuring they remain relevant in an ever-expanding musical landscape.

Product Functions

Its functionality will be accessed by evaluation of the data set that we will provide. We can evaluate the overall accuracy of the model which we will make. Another important thing to consider here is how many different genres there are in the world. As this would help us out in training the model better.

It must be efficient and can be defined as the ability to achieve an end goal with little to no waste, effort, or energy. Being efficient means you can achieve your results by putting the resources you have in the best way possible.

How quickly can the CNN algorithm process a large dataset of wines? What is the response time for queries?

How dependable is the CNN algorithm? Does it produce consistent results, and how does it handle noisy or incomplete data?

Availability is the measurement of how “available” a service is. Quite simply, if a customer wants to use the service at a time when we promised it would be there for them, they expect it to be available.

User Characteristics

Internal Users:

- **Testers:** These are individuals responsible for testing the functionality and accuracy of the music genre classification system. They evaluate the system's performance, report issues, and help ensure its reliability.
- **Music Curators:** Music cultivators or curators play a significant role in organizing and categorizing music content. They use the genre classification system to maintain music

libraries, create playlists, and ensure that tracks are appropriately tagged with the correct genres.

- **Board Members:** Board members may have a strategic interest in the success and development of the music genre classification system, particularly if it is part of a larger music-related organization.

External Users:

- **Music Communities:** Music communities, including listeners and enthusiasts, have a strong interest in the accuracy and efficiency of music genre classification. They rely on genre information to explore and discover music that aligns with their preferences.
- **Educational Institutions:** Educational institutions play a vital role in training individuals interested in music technology and data analysis. They promote initiatives related to music genre classification and may offer courses or programs in this field.
- **Musicians and Artists:** Musicians and artists may use genre classification as a tool for understanding the market and the preferences of their audience. They may also explore genre classification to experiment with different musical styles.
- **Music Platforms and Streaming Services:** Music platforms and streaming services use genre classification to provide users with personalized playlists and recommendations. Accurate genre information is crucial for delivering a satisfying user experience.

General Constraints, Assumptions and Dependencies

Constraints:

- The Data set must be clean and well-formatted.
- The Deep learning method used is Parallel Recurrent Convolutional Neural Network which will allow our classifier to learn better and efficiently.

Assumptions:

- The data set which is feeding for the training purpose or for validation purpose should only contain the different musics, as this must be the prior requirement before training the model, otherwise no point going further.

- CNN algorithm must be able to correctly identify the hidden patterns that are there within the dataset by correctly identifying the data and training the model to utmost accuracy.

Apportioning of the requirements

CNN Analysis:

- Know about the Goal
- Planning
- Data collection and analysis
- Data Pre-processing
- Exploratory Data analysis
- Feature Engineering
- Model testing and Evaluation
- Data Visualisation
- Completion

Detailed Description of Functional Requirements

Purpose - A description of the functional requirements and its reasons

Inputs -What are the inputs.

Processing - It Describes the outcome rather than the implementation

Outputs - The Genre of the Music.

Table 3:

Activities	**Description**
Data set	Collection of data
Data Pre-processing	It involves cleaning and transforming the data to prepare it for the Prediction of music

Load Data	All features are relevant for CNN, so it is important to select the features that are most informative.
Data cleaning	It involves transforming the data so that all of the features have the same scale.
Feature Selection	Involves selecting the features that will the most to the prediction of the music genre
CNN Algorithm	It aims to predict the genre using an audio signal as its input. The objective of automating the music classification is to make the selection of songs quick and less cumbersome.

Performance requirements

- **Speed**: The Parallel Recurrent CNN must be able to correctly identify the music genre in a reasonable amount of time.
- **Accuracy:** The Accuracy of the model should be good because , if a new music comes for validation then , it should correctly identify the genre associated with that music.
- **Usability:** The Model should be usable for the end users , that means, it should be user friendly in nature, interface should not be that complex.
- **Flexibility:** The model which we are developing should accommodate different musics into it , so that if a new music comes for classification then it should correctly tell the prediction based on the training on the large amount of datasets containing multiple music genres.

4. Software Project Management Plan

PROJECT - "MUSIC GENRE CLASSIFICATION"

Introduction:

The project aims to classify music into various genres using machine learning techniques. This involves data acquisition, preprocessing, and modeling. The end goal is to accurately predict the genre of a given piece of music, aiding in music

categorization and recommendation systems.

Project Deliverables:

- Cleaned and preprocessed GTZAN music dataset.
- An interactive user interface for music genre prediction.
- Trained and tested machine learning model for music genre classification.
- Comprehensive documentation including user and technical guides.
- Software Requirements Specification (SRS) and Software Design Description (SDD) documents.

Definitions and Acronyms:

GTZAN	A widely recognized dataset used for music genre classification.
MFCC	Mel-frequency cepstral coefficients - a representation of the short-term power spectrum of sound.
CNN	Convolutional Neural Networks, a deep learning algorithm that can recognize patterns.
ML	Machine Learning.

UI	User Interface.

Project Organization:

Project organization refers to the way a project is set up to be managed and executed. It encompasses the system of roles, responsibilities, and processes established to ensure the effective management, planning, and execution of a project.

Process Model:

The Waterfall model is used, ensuring a linear progression from one phase to the next, starting with gathering data and ending with deploying the model.
Waterfall model: A traditional method in software development where one phase must be completed before moving on to the next. It's sequential, and each phase depends on the deliverables of the previous one.

Organizational Structure:

Organizational structure outlines how activities related to the project (like task allocation, supervision, and coordination) are directed towards the achievement of project objectives. It defines a hierarchy within the project team and delineates the relationships between team members.

- Project Manager: Oversees project tasks, communicates with stakeholders, and ensures timely achievement of milestones.
- Data Acquisition Team: Sources the GTZAN dataset and confirms its authenticity and completeness.
- Data Cleaning Team: Processes audio files and extracts essential audio features.
- Model Building Team: Builds, trains, and validates the machine learning algorithm.
- UI Development Team: Constructs a platform for users to engage with the model.
- Documentation Team: Produces all necessary documentation.

Project Responsibilities:

Project responsibilities delineate the specific duties, tasks, or actions that individuals or teams are expected to undertake within the project. They are defined to ensure that every task associated with the project has a clear owner.

- Project Manager: Oversees project tasks, communicates with stakeholders, and ensures timely achievement of milestones.
- Data Acquisition Team: Sources the GTZAN dataset and confirms its authenticity and completeness.
- Data Cleaning Team: Processes audio files and extracts essential audio features.
- Model Building Team: Builds, trains, and validates the machine learning algorithm.
- UI Development Team: Constructs a platform for users to engage with the model.
- Documentation Team: Produces all necessary documentation.

Managerial Process:

The managerial process refers to the systematic series of actions, methods, or procedures that managers follow to design, plan, organize, lead, control, and evaluate project tasks, resources, and goals to achieve desired outcomes. It's the way managers oversee and direct the activities of the project or organization to ensure its objectives are met efficiently and effectively.

Management Objectives and Priorities:

Definition:

These are the primary goals and focal points that guide the managerial decision-making process throughout the project. They serve as the yardstick against which project progress and success are measured.

Importance:

- Sets clear targets for the team.
- Helps in resource allocation and prioritization of tasks.
- Provides a direction for the project, ensuring that all team members are aligned towards a common goal.

Assumptions,Dependencies, and Constraints:

Assumptions:

Assumptions are beliefs or things we consider to be true or certain without concrete evidence, yet they form the basis on which we plan and act.

- Data Availability: It's assumed that the GTZAN dataset is readily available and can be used without legal or licensing issues.
- Tool Efficacy: Libraries like Librosa and frameworks like TensorFlow or Scikit-learn will be suitable and efficient for the required data processing and model building tasks.
- Team Expertise: The project team possesses the necessary skills in machine learning, data processing, and software development to complete the project.
- Hardware and Software Compatibility: The tools and technologies chosen for the project will be compatible with the available hardware infrastructure.

Dependencies:

Dependencies are conditions that rely on something else for the project to move forward.

- Data Preprocessing: The success of the model-building phase is highly dependent on the proper cleaning and preprocessing of the GTZAN dataset.
- Team Availability: Timely completion of tasks depends on the consistent availability of team members. If one team finishes late (e.g., Data Cleaning), it could delay the subsequent teams (e.g., Model Building).
- External Libraries: The project relies on third-party libraries and tools. Any updates or changes to these tools could impact the project's progress.
- Infrastructure: Efficient model training and testing are dependent on the availability of required computational resources, especially if deep learning models are employed.

Constraints:

Constraints are the limitations or restrictions the project must work within.

- Timeline: The project has a fixed start and end date, which constrains the amount of time available for each phase of the project.

- Budget: There may be a fixed budget allocated for the project. This could limit the tools, resources, or manpower that can be utilized.
- Computational Resources: There might be limitations on available computational power, especially for intensive tasks like training machine learning models.
- Dataset Limitations: The GTZAN dataset, while popular, has its own limitations in terms of diversity, size, and possible biases. This could constrain the model's overall performance and generalization capabilities.
- Software and Licensing: There could be constraints related to the software licenses, especially if proprietary tools are used. This may limit the number of installations or the duration of usage.

Risk Management:

- It is the process of identifying, evaluating, and prioritizing potential risks, followed by coordinated actions to minimize, control, or accept the consequences of uncertain and unpredictable events.
- In this project, it entails foreseeing potential problems that might arise during the phases of data acquisition, preprocessing, model building, and deployment and ensuring mechanisms are in place to address these problems.

Risk Management Plan:

A Risk Management Plan is a detailed roadmap for how a project or organization will address potential challenges. It identifies risks, assesses their potential impact, and outlines strategies to manage and mitigate those risks. This plan acts as a guideline, ensuring that risks are managed proactively, rather than reactively.

For this project, this means foreseeing challenges that could arise during the various phases (data acquisition, preprocessing, modeling, etc.) and ensuring there are mechanisms in place to tackle these challenges efficiently.

Risk Analysis:

- It is the process of understanding the nature, sources, and causes of the risks that you face. It also involves determining the level of risk by considering the consequences and the likelihood of potential outcomes.
- This could involve studying the previous projects of a similar nature, discussions with experts in the field, and researching common pitfalls in music genre classification endeavors.

Risk Identification:

It is the initial step in the risk management process. It involves spotting : sources of risk, areas of impacts, events, and any other scenarios that could potentially harm the project.

Common risks might include:

- Data-related risks: Data corruption, missing data, or data that isn't representative of diverse music genres.
- Technical risks: Model overfitting, inefficiencies in the preprocessing phase, or the inability to scale the model.
- Resource risks: Lapses in team communication, unavailability of specific expertise, or computational resource constraints.

Risk Estimation:

Once risks are identified, risk estimation involves assessing the potential impact of each risk and the likelihood that it will occur. This is often a combination of quantitative (numerical) and qualitative (descriptive) analysis.

- Quantitative: For example, determining that there's a 70% chance that the model might overfit given the complexity of the features and the size of the dataset.
- Qualitative: Describing the potential impact of a risk, like "High impact" for data corruption since it would halt the project, or "Medium impact" for overfitting, as it can be managed with certain techniques.

Monitoring and Controlling Mechanisms:

Tools, practices, and systems set in place to keep track of project performance against the established plan, ensuring corrective actions are taken when necessary.

Importance:

- Ensures that the project stays on track and within its defined constraints.
- Allows for timely interventions and corrections.
- Reduces the chances of cost overruns, delays, and quality issues.

Staffing Approach:

The strategy and methodology used to source, hire, train, and manage the human resources required for the project.

Importance:

- Ensures the right skills are available for the project's success.
- Impacts the efficiency, productivity, and morale of the project team.
- Affects the overall quality and timeline of the project deliverables.

Technical Process:

This pertains to the specific technical activities, methodologies, and tools that the project will employ. For the music genre classification project, the technical process will largely revolve around software development, modeling, and documentation.

Software Documentation:

Documentation is pivotal for understanding, maintaining, and scaling the software. It provides a clear overview of how the software was designed, its requirements, and how to use it.

Software Requirements Specification (SRS):

Definition: An SRS is a document that captures the complete software requirements in detailed terms. It defines the expected behavior, features, and constraints of the software.

In context: For the project, the SRS would detail the need for a model to classify music genres, specific requirements regarding accuracy, the dataset to be used (GTZAN), and requirements related to the integration of the model into systems or platforms.

Software Design Description (SDD):

The SDD provides a detailed description of the software's architecture and design decisions. It's more about 'how' the software will achieve its requirements.

This would include the architecture of the machine learning model (e.g., CNN layers, activation functions), data preprocessing steps (using MFCC, other audio features), and possibly the user interface design if there's a frontend to the system.

- **Work Breakdown Structure (WBS):** A hierarchical decomposition of the total scope of work to be carried out by the project team to achieve project objectives.
  The WBS would segment tasks like "Data Gathering", "Data Cleaning", "Dataset Transformation", "Model Building", and "Documentation".
- **Gantt Chart Representation:** A visual representation of the project schedule, showing task durations, start and end dates, and dependencies.
  This chart would visually layout the timeline of each segment from the WBS, showing when "Data Gathering" starts and ends in relation to "Model Building", for example.
- **(Critical Path Method):** A technique used to identify the critical (longest) path through a set of tasks, considering dependencies. Tasks on this path define the project's shortest duration. CPM would help identify the most critical tasks for the music genre classification project—those that, if delayed, would delay the whole project.
- **PERT Chart Representation (Program Evaluation and Review Technique):** A graphical representation of a project's timeline, showing the order of tasks and their interdependencies.
  It can showcase the sequence and dependencies, for instance, highlighting that "Data Cleaning" can't commence until "Data Gathering" is complete.
- **Data Flow Diagram (DFD):** A graphical representation of the flow of data within a system, detailing where data comes from, how it's processed, and where it goes.

The DFD would depict how music data flows from the GTZAN dataset, through preprocessing steps, into the machine learning model, and then outputs as a genre classification.

Software Test Plan:

A detailed plan that outlines the strategy, objectives, resources, schedule, and scope of testing activities.

It would include testing the model's accuracy, ensuring no overfitting, and possibly user testing if there's an interface for genre predictions.

- **User Documentation:** Guides that help end-users understand how to use the software or system.

  This would be a guide on how to input a piece of music into the system and retrieve a genre prediction, detailing any interfaces or commands needed.

- **Project Support Function:** The infrastructure, tools, and resources supporting the successful execution of the project.

  This could include cloud storage for data backup, GPU resources for training, and tools like Slack or Teams for team communication.

Work Packages, Schedule, and Budget:

Work Packages: These are essentially smaller, manageable tasks or units of work that make up a larger project. For this project, they might include:

- Data Acquisition: Fetching the GTZAN dataset.
- Data Preprocessing: Audio signal processing and feature extraction.
- Model Construction: Building and training the genre prediction model.
- UI Development: Crafting the user interface for genre predictions.
- Documentation: Preparing technical guides, user manuals, and architectural documentation.

Dependencies:

Dependencies refer to the sequence in which tasks should be performed and if one task relies on the completion of another. For this project:

- Data Preprocessing depends on Data Acquisition.
- Model Construction can only commence after Data Preprocessing.
- UI Development can start in parallel with Model Construction but will need final model integration.
- Documentation can partly start in parallel but will need finalized components for complete documentation.

Resource Requirements:

This encompasses both the human and technical resources necessary for the project.

- **Human Resources:**
    - Data scientists for model construction.
    - Data engineers for data acquisition and preprocessing.
    - UI/UX designers and developers for the user interface.
    - Technical writers for documentation.
- **Technical Resources:**
    - Servers or cloud space for data storage.
    - Computing power (like GPUs) for model training.
    - Software licenses, if any specific tools are being used.
    - Platforms like GitHub for version control.

Budget and Resource Allocation:

Define the financial allocation for each work package and resource.

- Data Acquisition and Preprocessing might need budgeting for tools, platforms, or cloud storage.
- Model Construction could be resource-intensive, necessitating higher financial allocation especially for computational power.
- UI Development would need resources for designing and hosting the application.
- Documentation would require less budget compared to others, mostly for manpower.
- A clear budget allocation ensures that you have adequate funds for each phase, and it helps in keeping track of expenses to ensure the project doesn't overshoot its financial limits.

Schedule:

The schedule outlines the timeline for the entire project, including when each work package will start and finish, ensuring that tasks are completed in a coordinated and timely manner for successful wine clustering analysis.The project has started from 24.08.2023 August and will be finished around 11th November.

Start Date	24.08.2023
Finish Date	9.11.2023

5. WBS Dictionary Entries

WBS LEVEL	WBS CODE	WBS NAME	WBS DESCRIPTION	WBS DURATION
1	1	Music Genre Classification	The overarching project aiming to classify music into distinct genres using machine learning.	56
2	1.1	Know About the Goal	Understanding and defining the primary objective of the project.	3
3	1.1.1	Choose Problem	Selecting the specific challenge or issue within music genre classification to address.	1
3	1.1.2	Problem Understanding	Gaining a deeper comprehension of the nuances, implications, and specifics of the chosen problem.	1
3	1.1.3	Problem Analysis	Investigating the problem's intricacies, potential solutions, and expected challenges.	1
2	1.2	Planning	Outlining the approach, resources, and timeline for addressing the problem.	5
3	1.2.1	Scope/Size Estimation	Gauging the breadth and depth of the project, including the data volume and expected outcomes.	1
3	1.2.2	Cost Estimation	Projecting the financial requirements for successfully executing the project.	1
3	1.2.3	Time Estimation	Forecasting the duration and milestones for the project's completion.	1
3	1.2.4	Effort Estimation	Evaluating the manpower and hours required for each phase of the project.	1
3	1.2.5	Software Project Management Planning (SPMP) Validation.	Confirming that the project management plan aligns with best practices and project goals.	1
2	1.3	Data Collection and Analysis.	Gathering relevant data for the project and conducting preliminary analysis.	11
3	1.3.1	Dataset Preparation	Organizing the collected data into a structured format suitable for model training.	2

3	1.3.2	Dataset Analysis	Examining the dataset's properties, distribution, and potential challenges	2
3	1.3.3	Model Requirement Specification.	Documenting the technical requirements for the model and validating these requirements.	2
3	1.3.4	SRS Validation.	Documenting the technical requirements for the model and validating these requirements.	3
2	1.4	Data Preprocessing.	Enhancing the quality and structure of the data to ensure its readiness for model training.	10
3	1.4.1	Cleaning	Removing inconsistencies, translating data into a uniform format, and merging different data sources.	2
3	1.4.2	Mapping	Removing inconsistencies, translating data into a uniform format, and merging different data sources.	4
3	1.4.3	Integration	Removing inconsistencies, translating data into a uniform format, and merging different data sources.	2
3	1.4.4	Unit Testing of Data Preprocessing Module.	Checking the data preprocessing steps to ensure accuracy and correctness.	2
2	1.5	Data Transformation	Modifying data to better fit the model's needs, including normalization or standardization.	7
3	1.5.1	Train Test Split.	Dividing data into training and testing subsets, and applying any final transformations	1
3	1.5.2	Selection and Transformation.	Dividing data into training and testing subsets, and applying any final transformations	3
3	1.5.3	Processed Labeled Data	The final output after preprocessing, ready for model training.	2
3	1.5.4	Unit Testing of Data Transformation Module	Verifying that data transformation has been correctly executed.	1
2	1.6	Model Building	Crafting and training the machine learning model for genre classification.	22
3	1.6.1	Sequential Model Creation	Implementing different algorithms and neural network architectures to find the best model.	3

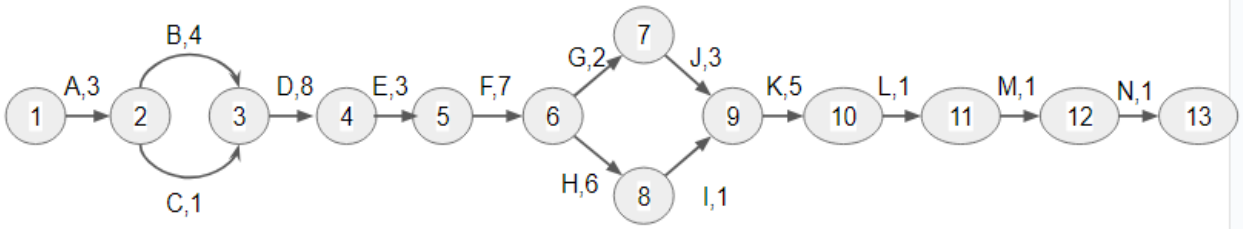
3	1.6.2	Accuracy Testing	Evaluating the model's performance against known benchmarks.	1
3	1..6.3	Implement K-Nearest Neighbor (KNN) algorithm	Implementing different algorithms and neural network architectures to find the best model.	3
3	1.6.4	Use Convolutional Neural Network	Implementing different algorithms and neural network architectures to find the best model.	4
3	1.6.5	Accuracy Testing	Evaluating the model's performance against known benchmarks.	1
3	1.6.6	Use Parallel Recurrent Convolutional Neural Network	Implementing different algorithms and neural network architectures to find the best model.	4
3	1.6.7	Overfitting Prevention Techniques	Implementing strategies to ensure the model generalizes well to new, unseen data.	2
3	1.6.8	Model optimization (using various optimizers)	Fine-tuning the model for better performance using different optimization techniques.	2
3	1.6.9	Unit Testing of Data Model Building Module	Checking the integrity and functionality of the model-building process.	1
2	1.7	Model Testing and Evaluation.	Assessing the model's overall performance and ensuring it meets project objectives.	6
3	1.7.1	Training Accuracy vs Testing Accuracy.	Comparing the model's performance on training data versus unseen testing data.	2
3	1.7.2	White Box Testing.	Testing the model's internal workings and its functionality from an end-user's perspective.	1
3	1.7.3	Black Box Testing.	Testing the model's internal workings and its functionality from an end-user's perspective.	1
3	1.7.4	Testing on Random Data.	Evaluating the model's predictions on new, unstructured data.	1
3	1.7.5	Model Output Analysis and Testing.	Reviewing the model's predictions for accuracy and relevance.	1

2	1.8	Data Visualization.	Representing data and results graphically for better understanding and interpretation	2
3	1.8.1	Correlation Analysis using Seaborn/Matplotlib.	Assessing relationships between variables in the dataset using visualization libraries.	1
3	1.8.2	Unit Testing on Data Visualization.	Ensuring the integrity and accuracy of data visualization components.	1

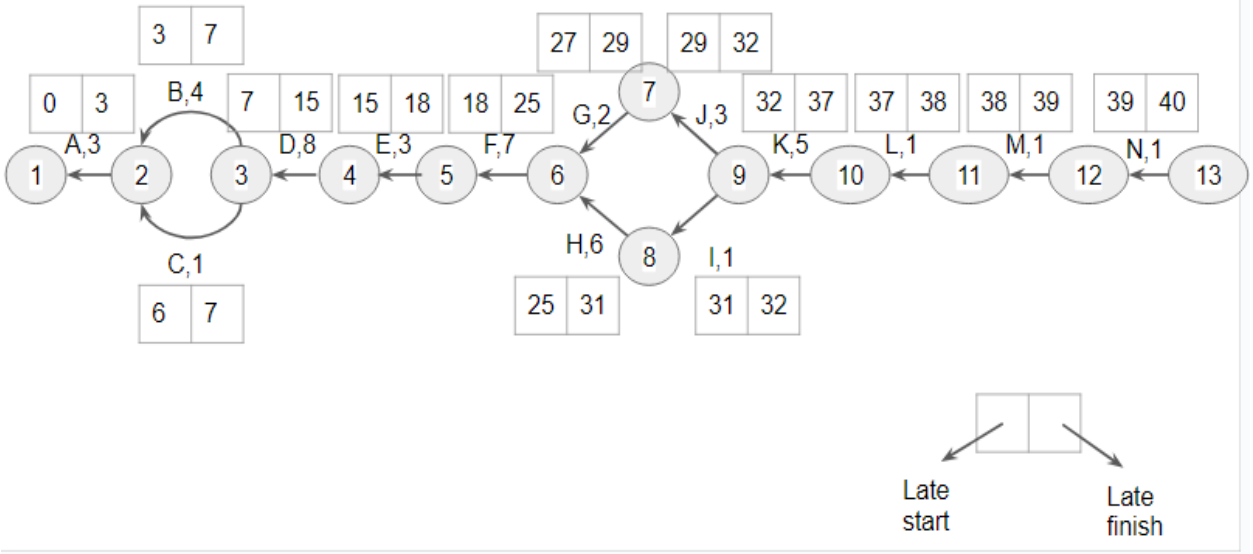
6. Activity On Node Diagram :- CPM and Pert Method

CPM

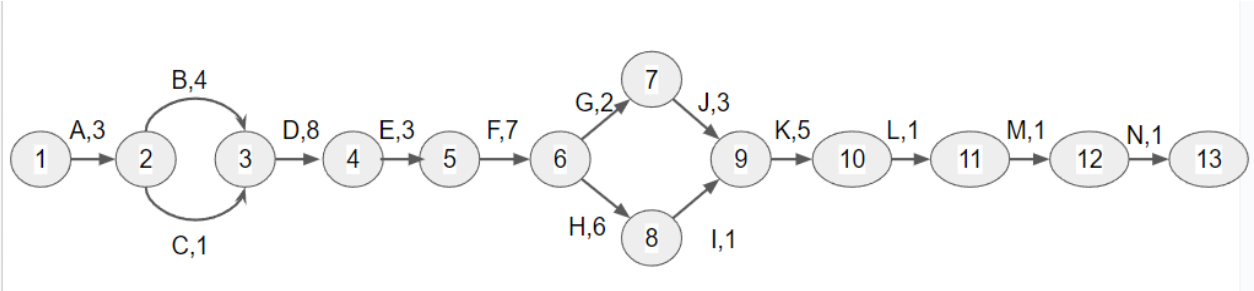
Forward Pass



Backward Pass



PERT Network



A -> B -> D -> E -> F -> H -> I -> K -> L -> M -> N

Optimistic Time, OT : 2+2+7+2+5+4+1+4+1+1+1 = 30

Pessimistic Time, PT : 5+6+10+4+10+7+3+7+3+3+3 = 61

Most Probable Time, MP : 3+4+8+3+7+6+1+5+1+1+1 = 40

Expected Time : (OT + 4(MP) + PT) / 6 = (30 + 160 + 61) / 6 = 41.83

Balancing Factor : (PT - OT) / 6 = (61 – 30) / 6 = 5.16

Range : 40 +5.16 = 45.16

: 40 – 5.16 = 34.84

7. Change Request Form

CHANGE REQUEST FORM

Project: Music Genre Classification **Number:** 06/23

Change Requester: Amrita Kundu **Date:** 5-11-2023

Requested change: Optimize the Model Accuracy using Transfer Learning.

Change Analyser: Abhishek Sharma **Analysis Date:**15-11-23
Components Affected:

Model Training Process, Performance Metrics are Affected.

Associated Components: Implementation of Transfer Learning techniques may increase model accuracy and efficiency.

Change Assessment: To Enhance the model accuracy use DenseNet121 Transfer Learning Model which has Optimized Architecture.

Change Priority: High

Change Implementation: The implementation involves integrating transfer learning methodologies into the existing model architecture. This process will likely involve pre-trained models and fine-tuning.

Estimated Effort: 2 Days.

Date to CCB: 20-11-2023 **CCB Decision Date:**20-11-2023

CCB Decision: Accept change .The change to be implemented in release 1.1.

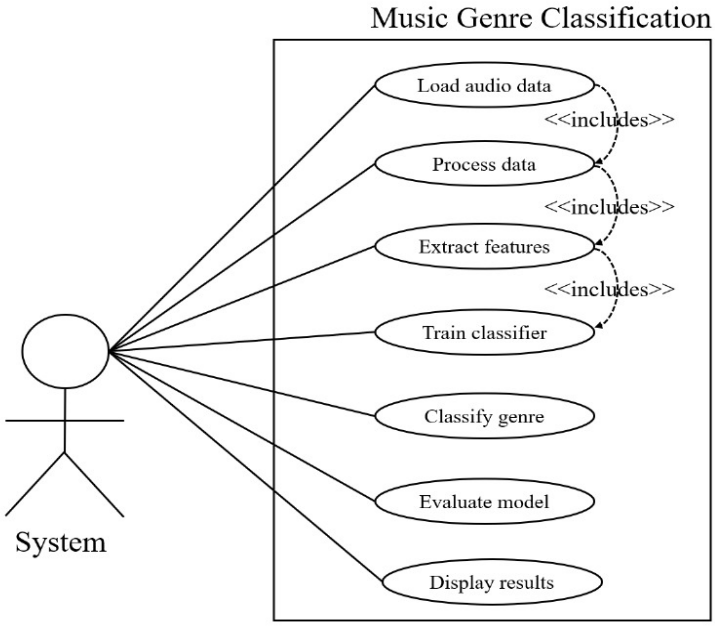
CCB Implementor: Ashmeet Kaur **Date Of Change:** 24-11-2023

Date Submitted to QA: 24-11-23. **QA Decision:** Model Verified

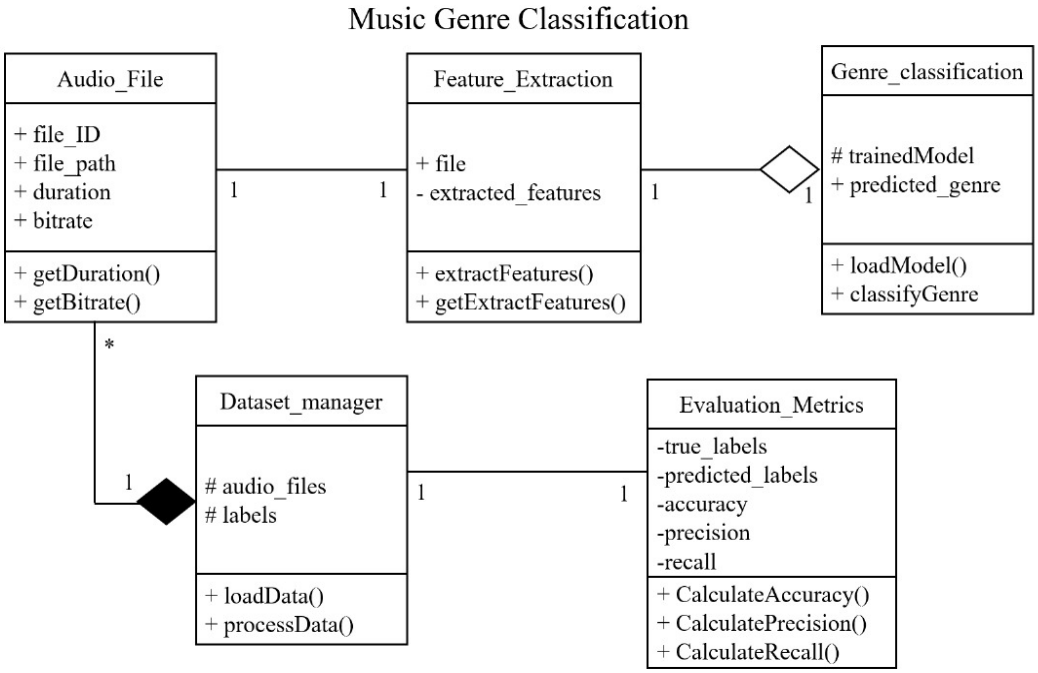
Date Submitted to CM: 26-11-23

Comments: No Comments

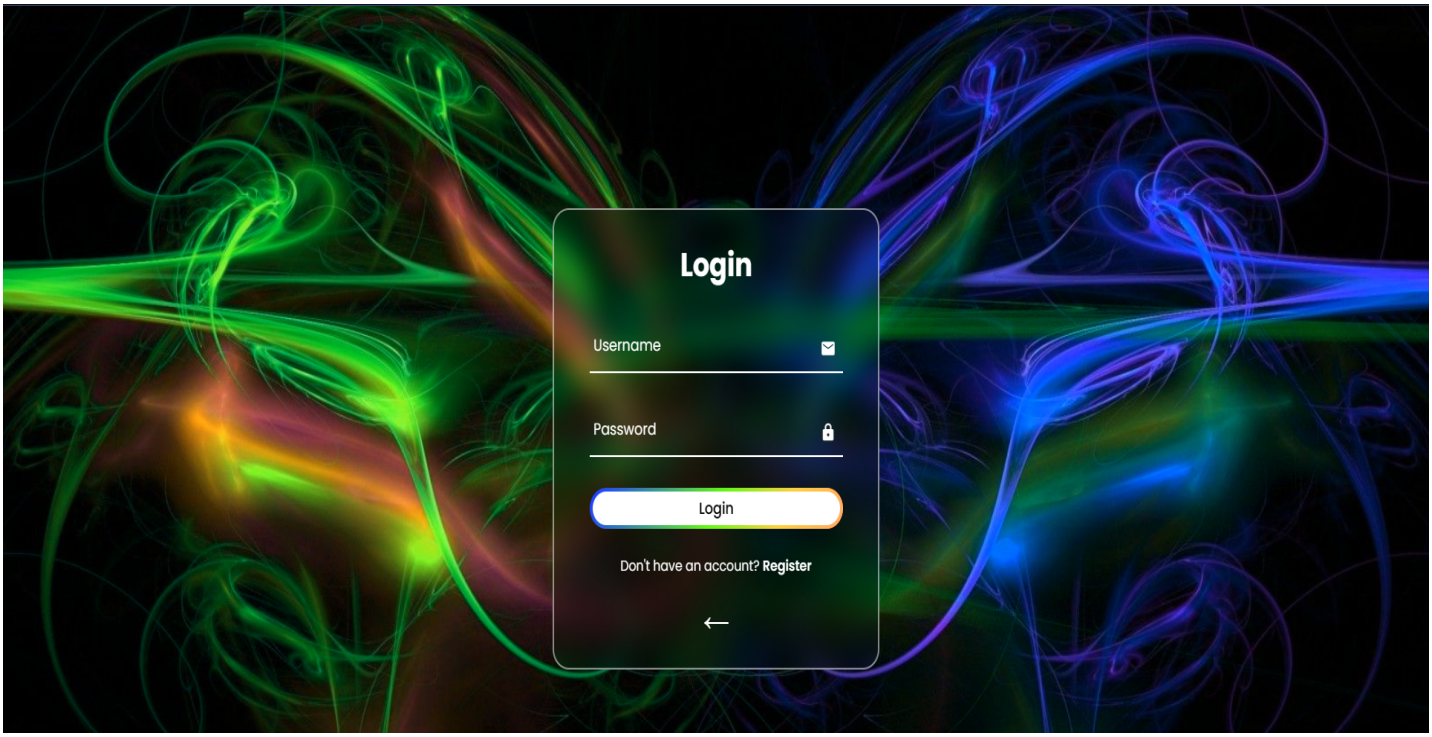
8. Use Case Diagram

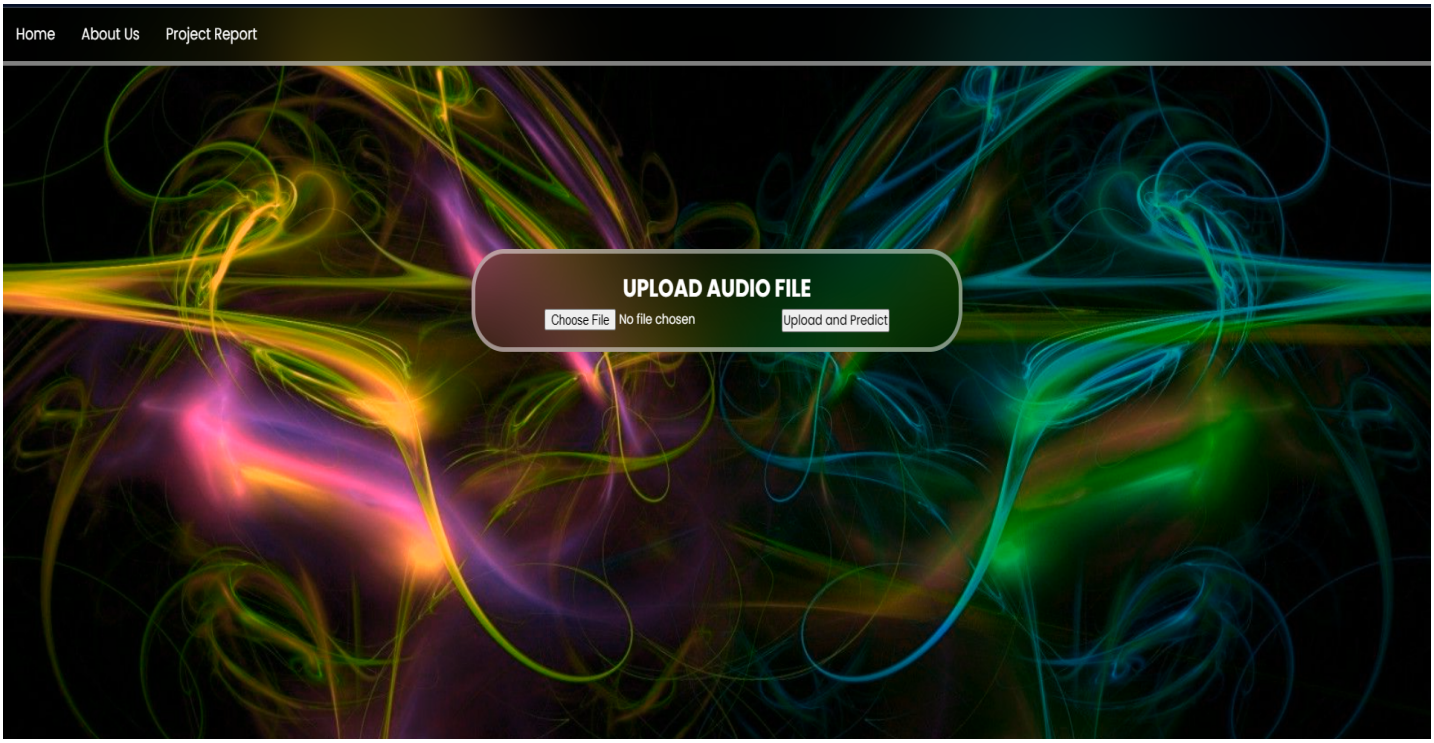


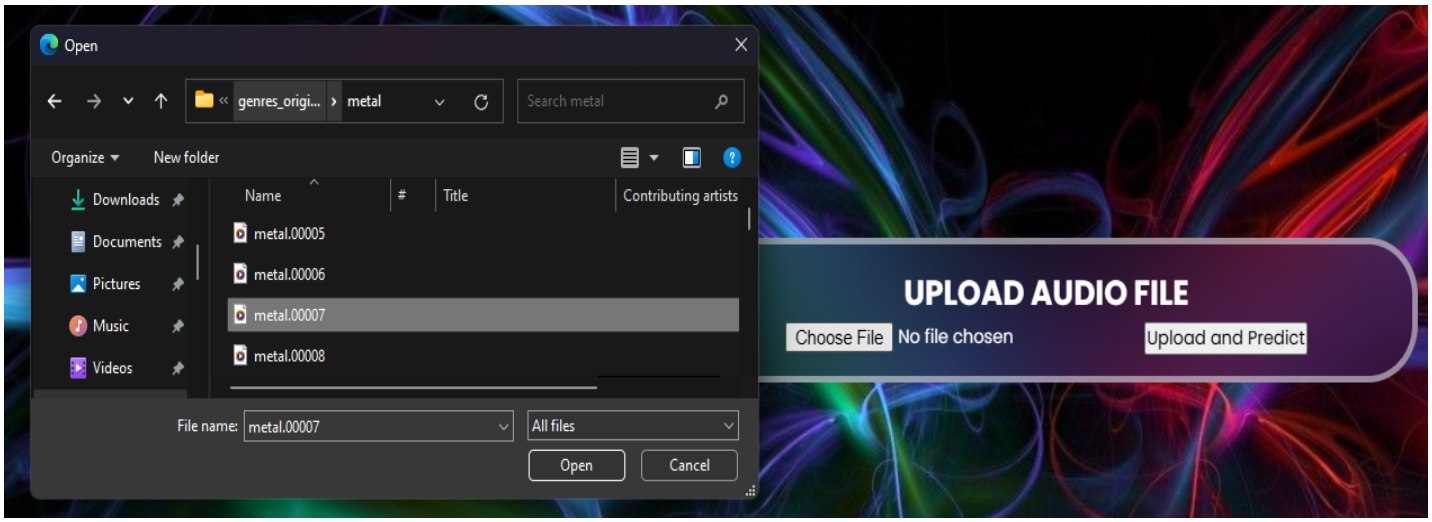
9. Class Diagram

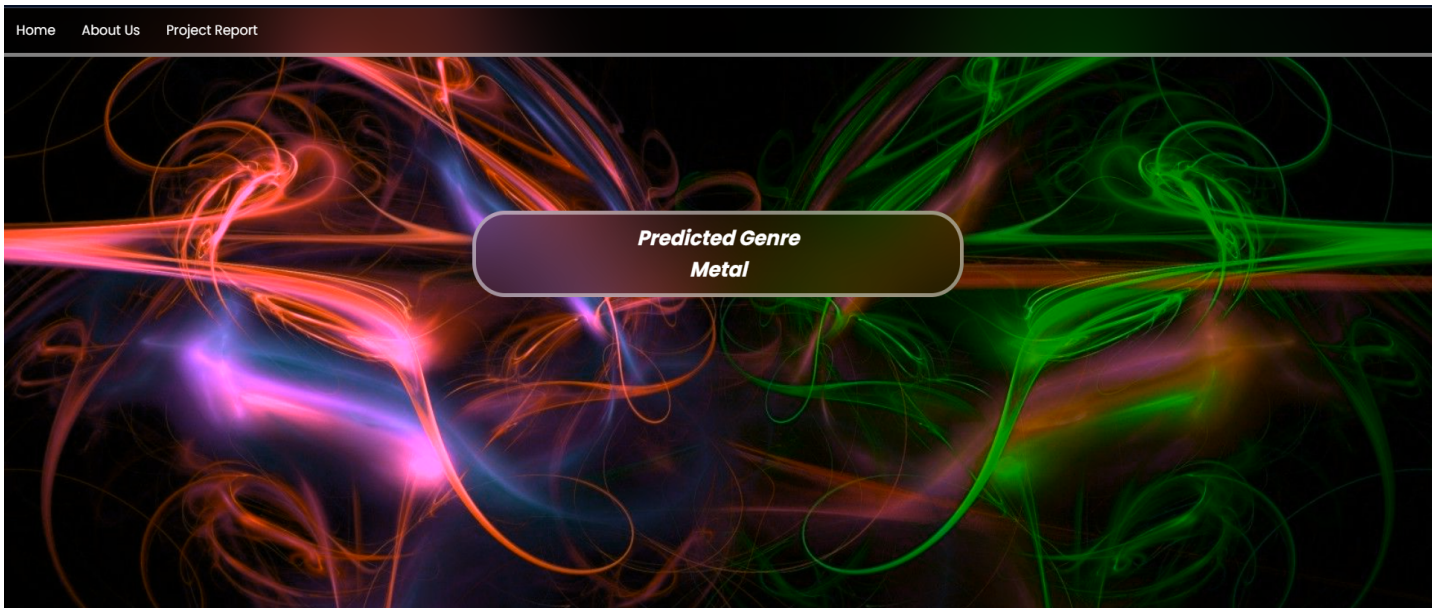


10.Snapshots Of The Working Project






Open
genres_origi...
metal
Search metal
Organize
New folder
Downloads
Documents
Pictures
Music
Videos
Name
#
Title
Contributing artists
metal.00005
metal.00006
metal.00007
metal.00008
File name:
metal.00007
All files
Open
Cancel
UPLOAD AUDIO FILE
Choose File
No file chosen
Upload and Predict


Home
About Us
Project Report
Predicted Genre
Metal

11. **EVA → Estimated Value Analysis**

Start Date	End Date	PV	BAC	AC	EV
30/8/23	15/11/23	20,000	15,000	25,000	40,000

Earned Value Analysis :

PV = Planned value

BAC = Budget At Completion

AC = Actual Cost

EV = Earned Value

Cost variance, CV = EV – AC

= 40,000 – 25,000

= Rs. 15,000 (Under budget)

Schedule variance, SV = EV – PV

= 40,000 – 20,000

= Rs. 20,000 (Ahead of schedule)

Cost Performance Index, CPI = EV/AC

= 40,000/25,000

= 1.6 (Under budget)

Schedule Performance index, SPI = EV/PV

= 40,000/20,000

= 2 (Ahead of schedule)

Estimate At Completion, EAC = BAC/CPI

= 15,000/1.6

= Rs. 9375

Estimate Time to Complete, ETC = Original time estimate/SPI

= 56/2

= 28 days

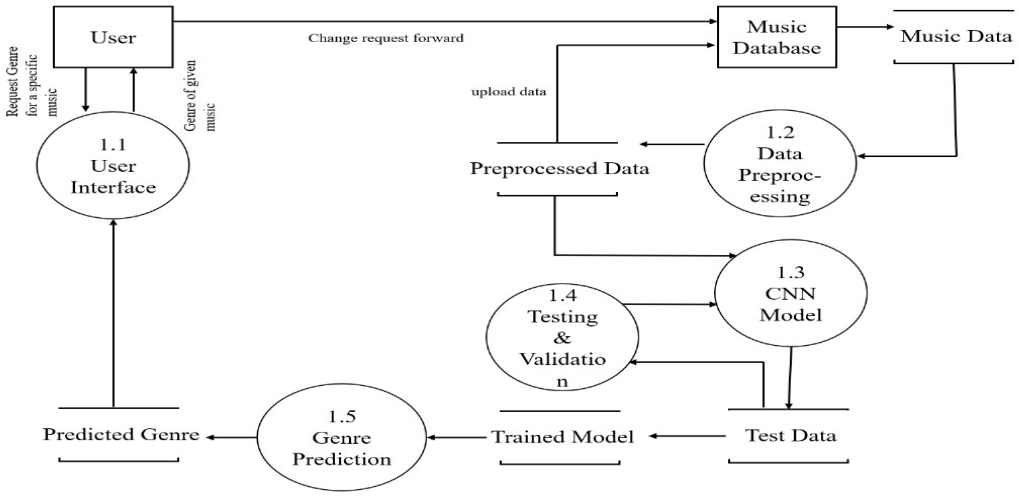
CV	SV	CPI	SPI	EAC	ETC
Rs. 15,000	Rs. 20,000	1.6	2	Rs. 9375	28 days

12. Risk Management Plan

Risk	Category	Probability	Impact	RMMM
Given that <due to the efficiency of developers in different Coding Languages> Then there is a concern <that the project might get delayed>	Team Risk	60%	4	**
If given that <the market competition is high> There could be a concern that<the performance rate might have to be increased to reduce the time of cluster formation>	Organizational risk	35%	3	***
Given that if <there is a lack of coordination among the team members> Then there is a concern <that the project might take more time to complete>	Team Risk	20%	2	**
As the members are from different places There might arise some problem to meet up for work	Environmental Risk	25%	1	*
<As the customer wanted the work to be done in python but the developers were efficient in other languages> <the quality of coding might not be up to the mark>	Technical Risk	50%	3	***
As the developers were new to the unsupervised learning The product might take more time to develop	Developer Risk	40%	3	**
As the modules being developed in different platform It can be problematic to integrate them together	Technical Risk	20%	2	***

13. Compute Function Points from the Data Flow Diagram for your projects.

- **Data Flow Diagram**



Function Point Analysis :

S.No.	Functional Units	Count Complexity	Functional Unit Totals
1	External Inputs (1)	Medium – 4	1 * 4 = 4
2	External Outputs (1)	Low – 4	1 * 4 = 4
3	External Enquiries (2)	High – 6	2 * 6 = 12
4	Internal Logical Files (3)	Medium – 10	3 * 10 = 30
5	External Interface Files (4)	Low - 5	4 * 5 = 20

Unadjusted Function Point Count, UFP = 4+4+12+30+20

= 70

Complexity Adjustment Factor, CAF = [0.65 + 0.01*14(3)]

= 1.07

Functional Point, FP = UFP *CAF

= 70 * I.07

= 74.9

Report on CASE Tool for Software Configuration Management GitHub

Introduction:

Software Configuration Management (SCM) is a critical discipline in software development, ensuring the systematic management of the various components of a software project. SCM activities include identifying changes, controlling these changes, ensuring proper implementation, and reporting changes to relevant stakeholders. GitHub, a popular web-based hosting service for version control using Git, plays a significant role in facilitating SCM.

Vendor Information:

Original Entity: Founded as: Logical Awesome LLC

Current Ownership: Owned by Microsoft Corporation

Acquisition Date: 2018

Release Information:

First Release:

Date: April 10, 2008

Details: Launch of GitHub as a platform for hosting version-controlled software repositories.

Latest Release:

Status as of April 2023: GitHub does not follow a conventional versioning system for its platform updates since it operates as a web service.

Update Frequency: Regular updates and feature enhancements are made available and are typically documented on GitHub's official blog or updates page.

Recommendation for Latest Info: Users should refer to the official GitHub website or directly access the GitHub platform for the most recent update details..

Version Control and Release Management:

Version Control System (VCS): GitHub is built on top of Git, a distributed VCS that allows developers to track and manage changes to code over time.

Branching and Merging: These features enable simultaneous updates and collaboration, minimizing conflicts in shared code bases.

Tags and Releases: GitHub allows users to create annotated tags in the repository to mark a published version or release.

Release Versioning:

Semantic Versioning: GitHub facilitates Semantic Versioning (SemVer), which typically follows the MAJOR.MINOR.PATCH format.

- MAJOR: Incompatible API changes.
- MINOR: Add functionality in a backward-compatible manner.
- PATCH: Backward-compatible bug fixes.

Release Drafting: GitHub provides a feature to draft releases, which can be shared and reviewed before final publication.

Pre-releases: GitHub also supports pre-releases to deliver beta, alpha, or release candidate versions to users for testing purposes.

Version Identification:

Each commit in GitHub is identified by a SHA-1 hash, ensuring the integrity and traceability of every change.

Version Numbering in GitHub:GitHub does not enforce a specific version numbering system but supports any system preferred by the development team, be it Semantic Versioning, Calendar Versioning, or Sequential Versioning.

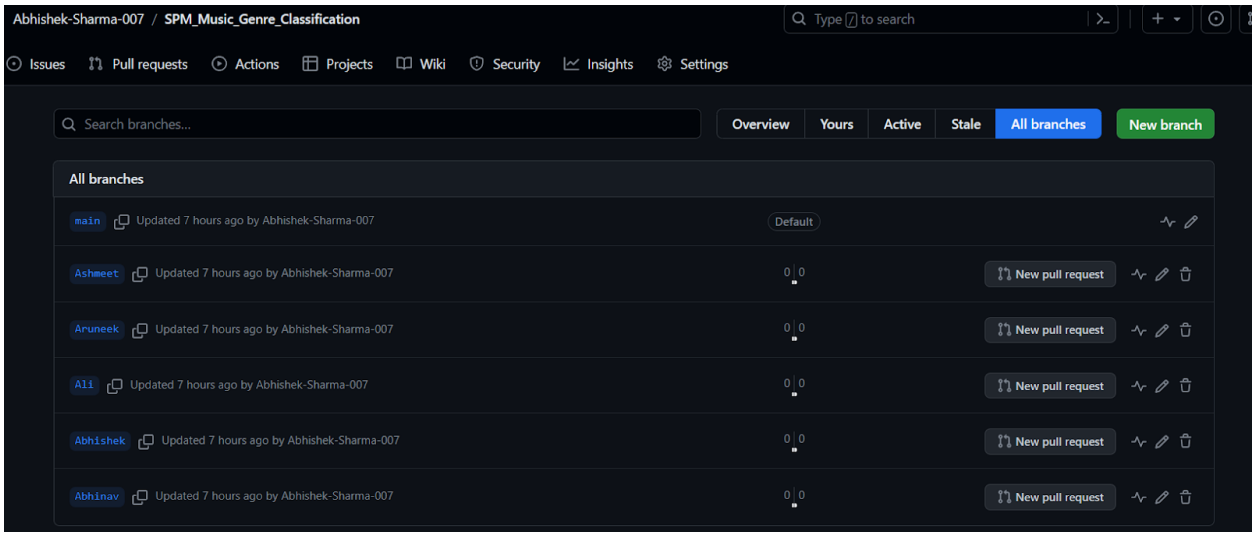
Features:

I. Version Control

Git-based System: GitHub uses Git for version control, enabling developers to track the history of changes, branch out, merge updates, and collaborate on software development.

Change Tracking: All modifications to the codebase are tracked. Each commit to the repository records a snapshot of the entire project, which can be compared to past versions or restored if necessary.

Branching and Merging: GitHub's branching model helps manage simultaneous development streams, allowing for feature development, hotfixes, and experimental work to proceed in parallel without disrupting the main codebase.

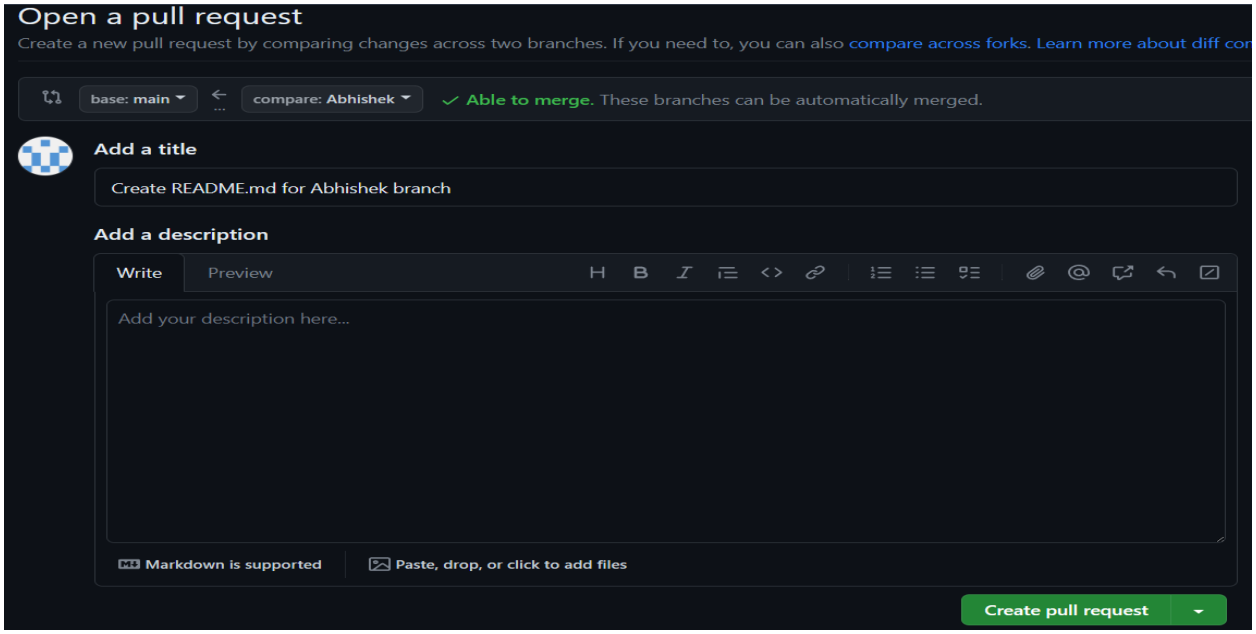


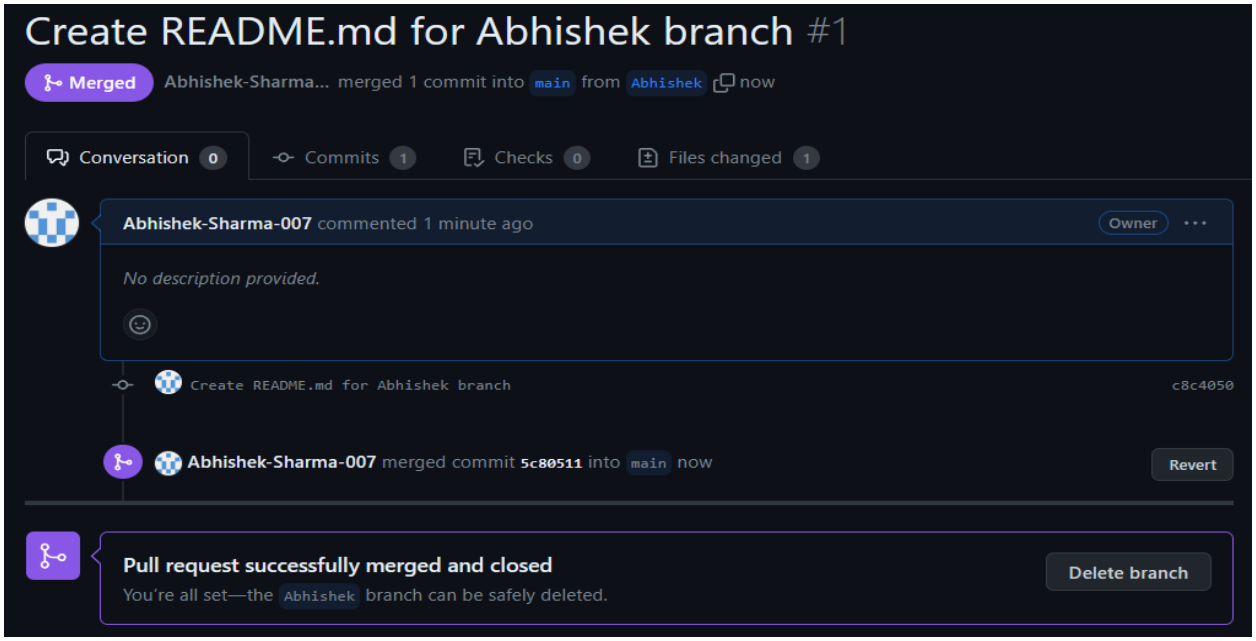
II. Code Review

Pull Requests: GitHub's pull requests are a cornerstone for code review, allowing developers to propose changes which can then be reviewed and discussed. This fosters high code quality and collaborative problem-solving.

Code Discussions: Pull requests enable in-line comments on code differences, facilitating focused conversations about specific changes.

Change Integration: Once approved, pull requests can be merged, seamlessly integrating new changes into the desired branch.

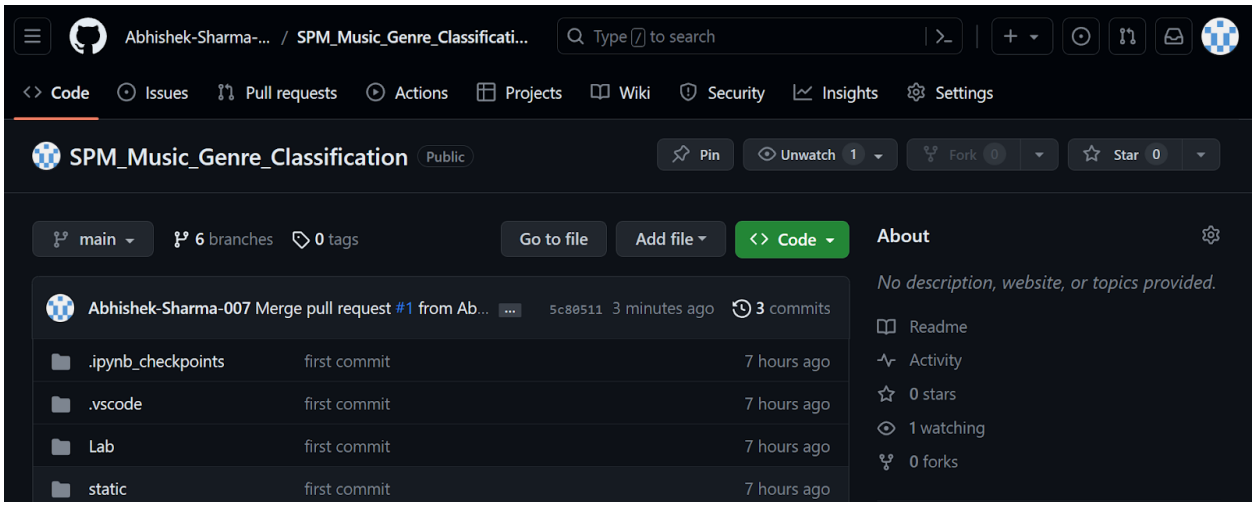




III. Collaboration Tools

- Issue Tracking: GitHub provides a robust issue tracking system where bugs, enhancements, and tasks can be logged, assigned, and managed through to completion.
- Project Management: Features like projects boards, milestones, and labels help organize and prioritize work within a repository, aligning development efforts with project goals.

- Wikis: GitHub repositories can include wikis, providing a space for documentation, design specifications, and other important project information, all within the context of the codebase.



IV. Search Functionality

- Code Searching: GitHub allows users to search within a repository or across all of GitHub for specific code snippets, commit messages, or even specific changes.
- Advanced Filtering: Search can be refined using various criteria, such as file type, language, repository, or author, enabling quick location of relevant pieces of code.

V. Code Visualization

- Comparisons: GitHub visually displays differences between file versions, showing line-by-line changes with color-coding to indicate additions, modifications, and deletions.
- Blame View: The "blame" feature shows line-by-line authorship, allowing users to see who last modified a particular line of code and in which commit, providing context for changes.

VI. Customization

- Custom Workflows: Through GitHub Actions, users can automate their software workflows, creating custom operations that build, test, and deploy their code.
- Integration with Tools: GitHub's Marketplace offers a wide array of integrations with third-party tools, enabling teams to tailor their development environment to their specific needs.

Drawbacks:

- Learning Curve: Git and GitHub can be complex for beginners.
- Public Repositories in Free Accounts: Private repositories are limited in the free tier, which can be a concern for proprietary code.
- Centralization: Over-reliance on GitHub poses risks of centralization and single points of failure.
- Limited Free CI/CD Resources: GitHub Actions, while powerful, offer limited resources in the free tier.
- Integration Costs: Some integrations with third-party tools may incur additional costs.
- Performance with Large Repositories: Handling very large repositories or files can be challenging.
- Security Concerns: Being a centralized platform, it can be a target for security threats, despite GitHub's strong emphasis on security.

Conclusion:

GitHub is a robust and versatile tool for Software Configuration Management, offering a wide range of features that facilitate the management of software projects. Its capabilities in version control, collaboration, and project management make it a vital tool in modern software development. However, organizations must also consider its limitations and drawbacks, especially regarding its learning curve, costs, and centralization, to ensure it aligns with their specific SCM needs and practices.

14. Testing Summary Report (signed copy by Testing Team)

Testing Summary Report

Project Title:	Music Genre Classification.
Developed by (Team Id):	Team C
Tested By (Team Id)	Team F
Brief Project Description:	The "Music Genre Classification" project aims to develop a machine learning system that can automatically categorise music tracks into different genres based on their audio content. This project involves the use of various audio processing and machine learning techniques to analyse audio features, such as spectrograms or MFCCs (Mel-Frequency Cepstral Coefficients), and build a classification model.
Errors/Limitations in the Project (Point wise):	1. Music Recommendation can 2. The system is not scalable.

Final Remarks:

1. The project is working as intended.

802332091 – Abhishek Sharma	Signature
802332098 – Ashmeet Kaur	Signature
802332003 – Abhinav Prasher	Signature
802332007 – Akbar Ali Ahamed	Signature
802332016 – Aruneek Kaur	Signature