Project Report

1. INTRODUCTION

- 1.1 Project Overview
- 1.2 Purpose

2. IDEATION PHASE

- 2.1 Problem Statement
- 2.2 Empathy Map Canvas
- 2.3 Brainstorming

3. REQUIREMENT ANALYSIS

- 3.1 Customer Journey map
- 3.2 Solution Requirement
- 3.3 Data Flow Diagram
- 3.4 Technology Stack

4. PROJECT DESIGN

- 4.1 Problem Solution Fit
- 4.2 Proposed Solution
- 4.3 Solution Architecture

5. PROJECT PLANNING & SCHEDULING

5.1 Project Planning

6. FUNCTIONAL AND PERFORMANCE TESTING

- 6.1 Performance Testing
- 7. **RESULTS**
 - 7.1 Output Screenshots
- 8. ADVANTAGES & DISADVANTAGES
- 9. CONCLUSION
- 10. FUTURE SCOPE
- 11. 11. APPENDIX

Source Code(if any)

Dataset Link

GitHub & Project Demo Link

Team Members:

Ayush Gupta (22BCE10279) Mayank Kandpal (22BCE10518) Ansh Khanna (22BCE11144) Aditya Jain (22BAI10198)

Project Report: Musicify

Phase : 1 Project Ideation Phase

1 Introduction

1.1 Project Overview

MusicifyMusicify is a comprehensive music streaming platform built using the MERN (MongoDB, Express.js, React, Node.js) stack. The platform creates a seamless experience for music enthusiasts to discover, explore, and listen to music from various genres and artists.

The application features a multi-role system with distinct user types:

- **Regular Users (Listeners)**: Can browse the music catalog, create accounts, create playlists, follow artists, and listen to music with both free and premium tiers
- **Artist Users**: Have access to a specialized dashboard for uploading and managing their music, tracking listener statistics, and interacting with fans
- Admin Users: Maintain the platform, manage users, and monitor content

The project integrates Firebase for authentication and MongoDB Atlas for database management, creating a robust and scalable infrastructure. The responsive design ensures a consistent experience across desktop and mobile devices, allowing users to enjoy music anytime and anywhere. The application includes features like continuous playback, personalized playlists, and algorithmic recommendations.

1.2 Purpose

The primary purpose of Musicify is to create an accessible and user-friendly platform connecting music lovers with their favorite tracks while providing artists with efficient management tools and listener insights.

The key objectives of the Musicify project are:

- **Enhanced Music Discovery** Provide algorithmic recommendations, curated playlists, and radio functionality to help users discover music matching their interests
- **Seamless Listening Experience** Enable efficient browsing, searching, playlist creation, and a streamlined playback process with queue management
- User Account Management Allow listeners to create profiles with both free and premium tiers, track listening history, save favorite songs, and manage personal information securely
- Artist Management System Equip artists with tools to easily upload, update, and organize their music catalog, as well as monitor listener statistics
- **Security & Authentication** Implement Firebase authentication alongside secure data handling practices to protect user information and prevent unauthorized access
- **Responsive & Intuitive UI** Create a visually appealing, easy-to-navigate interface with dark mode that works seamlessly across all devices
- Database Optimization Utilize MongoDB Atlas for efficient data storage and retrieval, ensuring fast performance even with large music catalogs and user bases
- Streaming Architecture Develop robust backend systems to handle audio streaming, playlists, and user activity tracking
- Scalable Architecture Build a well-structured application using the MERN stack that
 can easily accommodate growing catalogs, increasing user traffic, and additional
 features

2 Ideation:

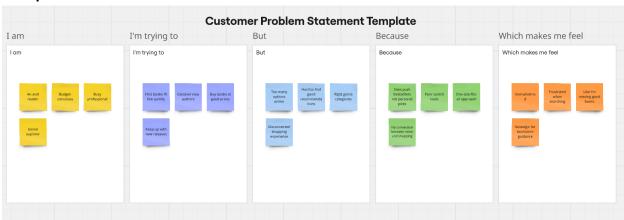
Date	27 March 2025
Team ID	SWTID1744391109
Project Title:	Musicify
Maximum Marks	2 Marks

2.1 Customer Problem Statement Template:

Create a problem statement to understand our customers' point of view. The Customer Problem Statement template helps you focus on what matters to create experiences people will love.

A well-articulated customer problem statement allows you and your team to find the ideal solution for the challenges your customers face. Throughout the process, you'll also be able to empathize with your customers, which helps you better understand how they perceive your product or service.

Example:



Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	A music enthusiast	Discover new artists and songs	Most platforms have limited discovery options	Algorithms tend to repeat similar recommendatio ns	Stuck in a music bubble
PS-2	Budget-con scious listener	Listen to high-quality music	Many platforms require premium subscription s	Free tiers have intrusive ads and limitations	Frustrated with interruptions

PS-3	Independen t artist	Share my music with potential fans	Getting visibility is extremely difficult	Major labels dominate promotional slots	Overlooked and undervalued
PS- 4	On-the-go listener	Seamlessly transition between devices	My listening state doesn't sync well	Platform limitations in cross-device experience	Disconnected and annoyed

2.2 Empathy Map Canvas:

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes.

It is a useful tool to helps teams better understand their users.

Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.

Example: Our Project on Full stack Music Streaming Platform- Musicify

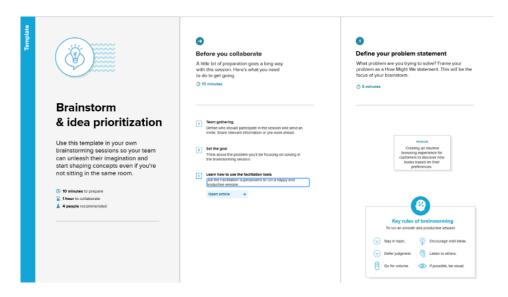
EMPATHY MAP: TUNE TRAIL MUSIC STREAMING APP SAYS **THINKS** • "I want an ad-free experience without paying too much" 🎧 • Music is essential to daily routines and activities • "I need to find new music that actually fits my taste" · Current platforms are either too expensive or too • "Why can't I easily continue listening when switching devices?" • Discovery algorithms don't truly understand persi • "I wish I could download music for my commute" · Switching between devices should be seamless • "Most playlists don't match my specific mood" · Quality matters but not at the expense of data us • "I want to support artists I like directly" · Independent artists deserve more visibility · "Why do all music apps push the same popular songs?" · Music preferences evolve and platforms should a MUSIC DOES **FEELS** LISTENER · Creates multiple playlists for different moods and activities · Overwhelmed by too many similar options · Shares music discoveries with friends · Frustrated with service limitations and interruptio · Searches for specific tracks and artists · Excited when discovering perfect tracks that mat · Listens while commuting, working, and relaxing · Annoyed by frequent ads in free services · Frequently skips recommended tracks that miss the mark · Connected to music that resonates with persona Switches between free and paid services based on features · Disappointed when recommendations are off-tar-· Uses different devices throughout the day for music · Satisfied when finding exactly the right song at the

TUNE TRAIL: USER PAINS & GAINS PAINS GAINS Disruptive Advertising Personalized Discovery · Frequent ads interrupt music flow · Al that truly understands musical taste · Premium costs too high to avoid ads · Mood-based recommendations that work · Forced visual ads in free versions · Discover similar artists to favorites **Limited Offline Access** Affordable Access · Can't listen during commutes without data · Value-priced premium tier · Download limits on free accounts · Free tier with reasonable ad frequency · Family and student discount plans · Downloaded content expires **Poor Discovery** Seamless Experience · Repetitive recommendations · Continue listening across all devices · Algorithm favors mainstream hits · Offline library syncs automatically · Can't easily find similar but new artists · Consistent interface on all platforms **Device Inconsistency Quality & Connection** · Progress lost when switching devices · High-quality audio with bandwidth options · Different interfaces across platforms · Direct artist support and updates · Can't seamlessly control from multiple devices · Community features with like-minded listeners

2.3 Brainstorm & Idea Prioritization:

Step-1: Team Gathering, Collaboration and Select the Problem Statement

- Team identified the core problem: "Music streaming platforms fail to balance affordability, discovery, and seamless experience"
- Focus on creating a service that addresses these three key pain points simultaneously



Step-2: Brainstorm, Idea Listing and Grouping

- Affordability Ideas: Tiered pricing, ad-supported free tier with reasonable limits, student/family plans
- **Discovery Ideas**: Advanced algorithm using listening habits, mood detection, playlist radio, genre exploration
- **Experience Ideas**: Cross-device syncing, offline mode, background play, queue management, high-quality audio options
- **Differentiation Ideas**: Lyric integration, song information, artist insights, visualization features



Step-3: Idea Prioritization

- **High Impact/Low Effort**: Cross-device listening state sync, basic recommendation engine, playlist management
- **High Impact/High Effort**: Advanced discovery algorithm, offline mode, premium tier infrastructure
- Low Impact/Low Effort: Basic user profiles, dark/light themes, sharing capabilities

notes to make it easier to find, browse, organize, and categorize important ideas as

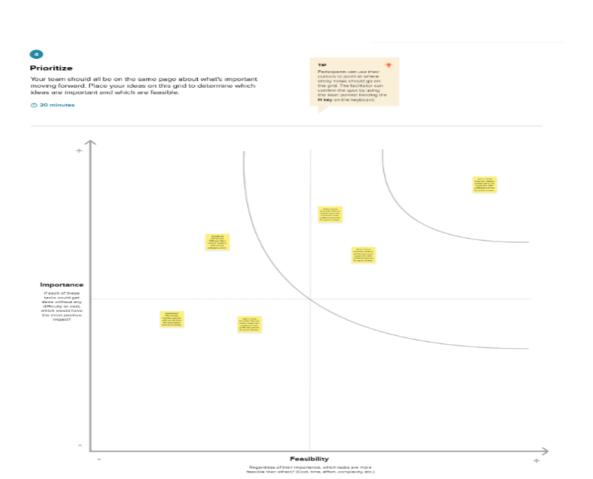
• Low Impact/High Effort: Visualization features, social network integration, lyrics synchronization



Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

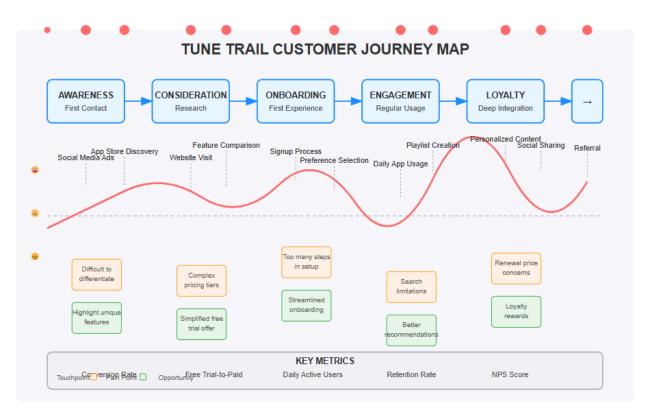




3 REQUIREMENT ANALYSIS

Date	30 March 2025
Team ID	SWTID1744391109
Project Title	Musicify
Maximum Marks	4 Marks

3.1 Customer Journey Map:



3.2 Solution Requirements (Functional & Non-functional)

Functional Requirements:

The following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration & Authentication	Registration through Form, Social Login Integration, Role selection (Listener, Artist, Admin), Free/Premium tier selection
FR-2	Music Playback & Control	Play/Pause/Skip Controls, Volume Control, Queue Management, Continuous Playback, Background Playing
FR-3	Role-Based Dashboard Access	Admin: Content Moderation & Statistics, Artist: Upload & Analytics, Listener: Personalized Home & Library
FR-4	Music Discovery	Search functionality, Browse categories, New releases section, Algorithmic recommendations
FR-5	Playlist & Library Management	Create/Edit/Delete Playlists, Add/Remove Tracks, Like Songs, Follow Artists
FR-6	Personalized Experience	Weekly Discovery Playlist, Recently Played, Favorite Artists Updates, For You Section

Non-functional Requirements:

The following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	Intuitive UI with dark theme, mobile responsiveness, and accessibility features
NFR-2	Security	Firebase Auth, token-based API security, secure payment processing
NFR-3	Reliability	99.9% uptime guarantee with failover systems and data backups

NFR-4	Performance	Fast stream initialization (<2s), minimal buffering, efficient API responses
NFR-5	Availability	Cloud-hosted infrastructure with load balancing and g distribution
NFR-6	Scalability	Microservice architecture capable of handling millions of concurrent streams

3.3 Data Flow Diagrams:

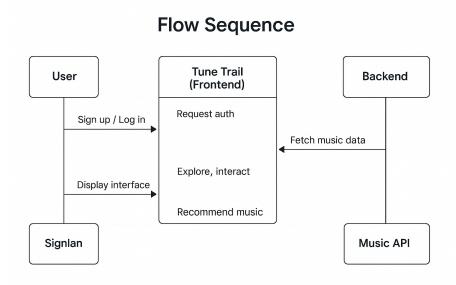
A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the right amount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.

Example: (Simplified)

Flow Sequence:



Example: DFD Level 0 (Industry Standard)



User Stories:

User Type	Functional Requireme nt (Epic)	User Story Number	User Story / Task	Acceptance Criteria	Priorit y	Releas e
Listener	Registration	USN-1	As a listener, I can create an account and select my music preferences	I can successfully register and see personalized content	High	Sprint- 1
Listener	Music Playback	USN-2	As a listener, I can play music with standard controls	I can play/pause/skip tracks and adjust volume	High	Sprint- 1

Listener	Playlist Managemen t	USN-3	As a listener, I can create and manage playlists	I can create, edit, and delete playlists	High	Sprint- 1
Listener	Discovery	USN-4	As a listener, I can browse music by genres, moods, and activities	I can navigate categories and find relevant music	Mediu m	Sprint- 2
Listener	Premium Features	USN-5	As a premium listener, I can download music for offline listening	I can access downloaded music without internet connection	Mediu m	Sprint- 2
Listener	Cross-devic e	USN-6	As a listener, I can continue playback across multiple devices	My playback state syncs when I switch devices	High	Sprint- 2
Artist	Registration	USN-7	As an artist, I can create an artist account	I can access artist-specific features after verification	High	Sprint- 1
Artist	Music Upload	USN-8	As an artist, I can upload and manage my music	I can add, edit, and remove my tracks from the platform	High	Sprint- 1
Artist	Analytics	USN-9	As an artist, I can view listener statistics for my music	I can see play counts, geographic data, and listener demographics	Mediu m	Sprint- 2
Admin	User Managemen t	USN-10	As an admin, I can manage user accounts	I can view, edit, and deactivate user accounts when necessary	High	Sprint- 1

Admin	Content Moderation	USN-11	As an admin, I can review and moderate uploaded content	I can approve, reject, or flag content for review	Mediu m	Sprint- 2
Admin	Platform Monitoring	USN-12	As an admin, I can view system performance metrics	I can access dashboards showing usage statistics and performance data	Low	Sprint- 3

3.4 Technical Architecture:

Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	Web-based responsive interface with dark theme	React.js, Tailwind CSS
2.	Authentication Service	Handles user registration, login, and session management	Firebase Authentication, JWT
3.	Streaming Service	Manages audio delivery and playback	Node.js, Express.js, Web Audio API
4.	Discovery Service	Handles recommendations and browsing functionality	Node.js, Express.js, Machine Learning algorithms
5.	User Management Service	Manages user profiles, preferences, and subscription status	Node.js, Express.js
6.	Database	Stores user profiles, music metadata, playlists, listening history	MongoDB (NoSQL)
7.	File Storage	Stores music files and related media	AWS S3 / Firebase Storage
8.	Cloud Database	Cloud-hosted database instance	MongoDB Atlas
9.	Payment Processing	Handles premium subscription transactions	Stripe / PayPal integration
10.	Infrastructure (Server/Cloud)	Cloud-hosted platform with load balancing	AWS / Google Cloud / Azure

Table 2: Application Characteristics:

S.No	Characteristics	Description	Technology
1.	Open-Source	Modern web technologies for	MERN Stack, Web Audio
	Frameworks	scalable application development	API
2.	Security	Multi-layer security approach	HTTPS, OAuth 2.0, JWT,
	Implementations		Content Encryption
3.	Scalable	Microservice design for independent	Docker, Kubernetes, Load
	Architecture	scaling of components	Balancing
4.	Availability	Redundant systems with automatic	Multi-region deployment,
		failover	Health monitoring
5.	Performance	Efficient streaming and caching	CDN integration,
		strategies	Progressive loading,
			Audio compression