Government arts and science college -Alangudi

College code-bdu752

Department: computer science

Internship program Smartinternz

Project name: Rhythmic tunes your melodic companion

Submitted by

Team id:NM2025TMID35251

Team leader:
Rajapandi.m(mrajapandi944@gmail.com)
Team member:suvendran. v
(suvendransuvendran15@gmail.com)
Team member :shiyam .k
(kdshiyam52@gmail.com)
Team member:rooban.j
(chiyanrooban@gmail.m)

RhythmicTunes: Your Melodic Companion
Ideation Phase
Brainstorm&Idea Prioritization

Brainstorm&Idea Prioritization Template:

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

Reference: https://www.mural.co/templates/brainstorm-and-idea-prioritization

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

Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

- **10 minutes** to prepare
- **1 hour** to collaborate
- **2-8 people** recommended

Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

10 minutes

A **Team gathering**
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B **Set the goal**
Think about the problem you'll be focusing on solving in the brainstorming session.

C **Learn how to use the facilitation tools**
Use the Facilitation Superpowers to run a happy and productive session.

Open article →

1 Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

5 minutes

PROBLEM

How might we [your problem statement]?

Key rules of brainstorming

To run an smooth and productive session

- Stay in topic.
- Encourage wild ideas.
- Defer judgment.
- Listen to others.
- Go for volume.
- If possible, be visual.

Step-2: Brainstorm, Idea Listing and Grouping

2 Brainstorm

Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil [switch to sketch] icon to start drawing!

Person 1	Person 2	Person 3	Person 4
Users can search songs, albums, and artists and get recommendations.	Allow users to create, edit, and share playlists.	Fetch and display real-time lyrics while playing a song.	Display top charts and trending songs based on user preferences.

3

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.

20 minutes

TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

Music Discovery & Search

Personalization & Playlists

Enhanced Listening Experience

Step-3: Idea Prioritization

4

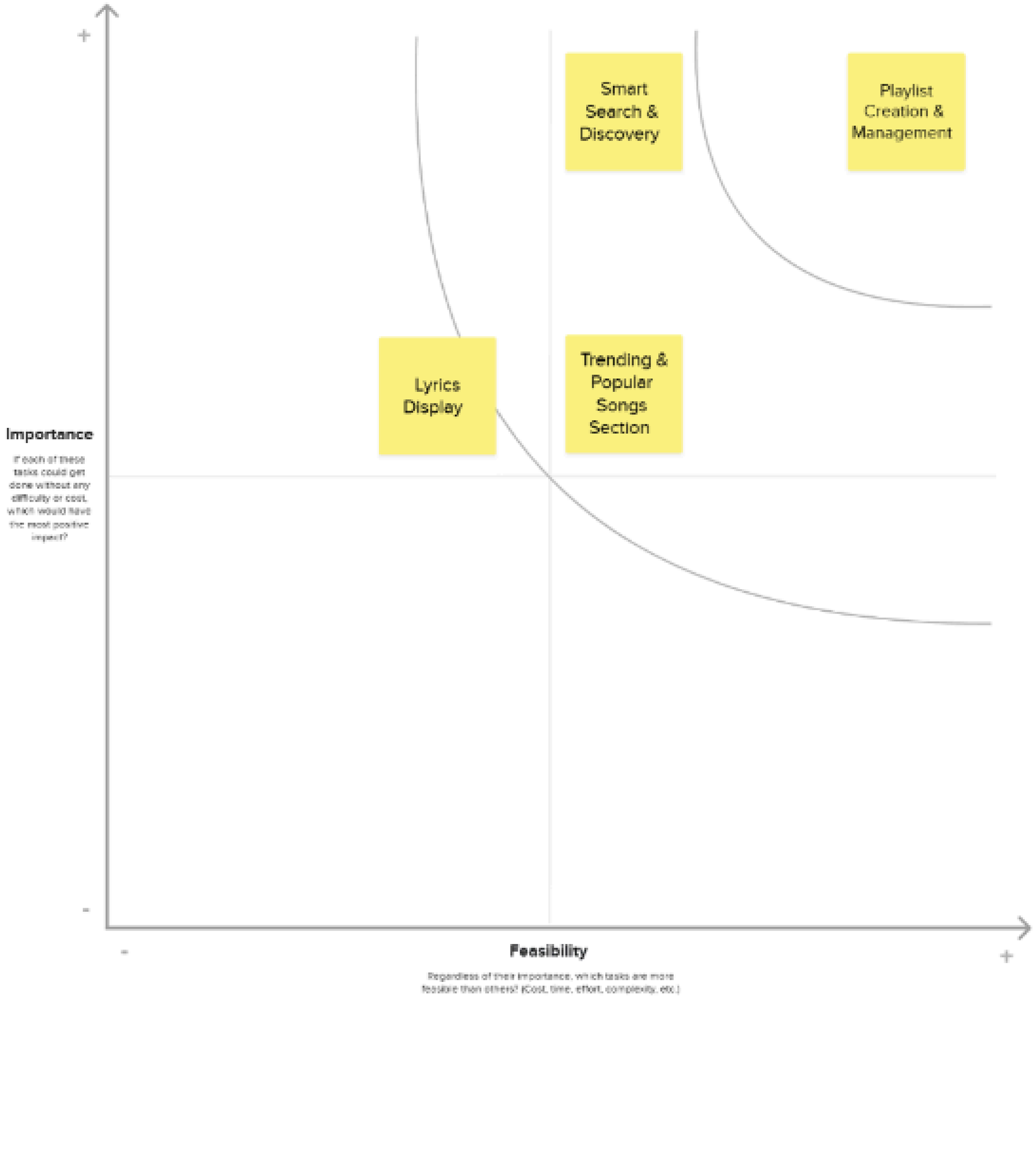
Prioritize

Your team should all be on the same page about what's important moving forward. Place your ideas on this grid to determine which ideas are important and which are feasible.

20 minutes

TIP

Participants can use their cursors to point at where sticky notes should go on the grid. The facilitator can confirm the spot by using the laser pointer holding the H key on the keyboard.



RhythmicTunes: Your Melodic Companion
Ideation Phase
Define the Problem Statements

Customer Problem Statement Template:

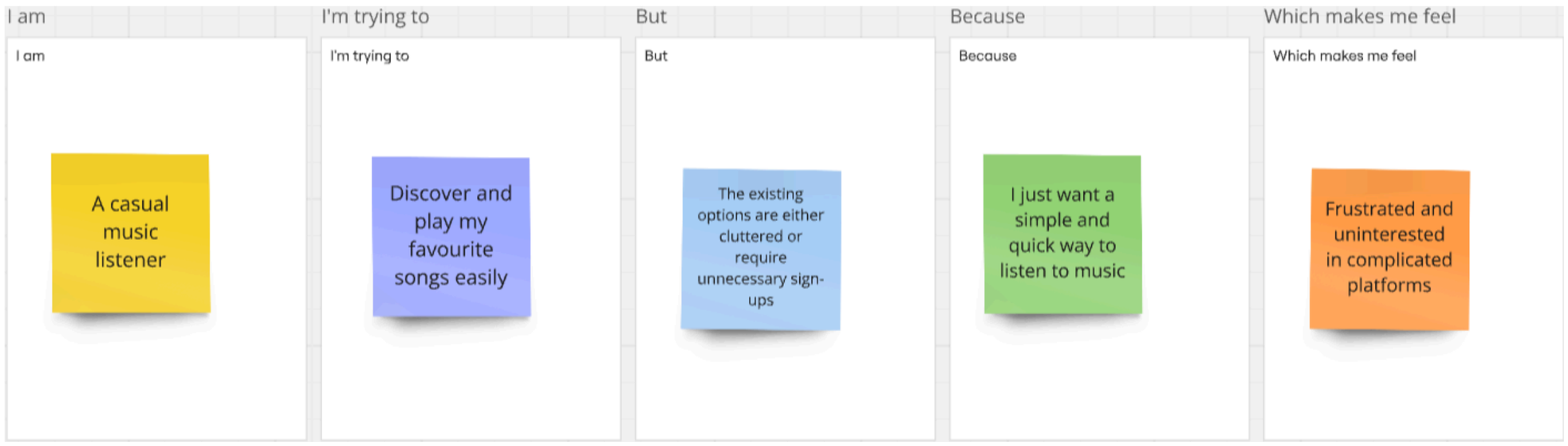
Create a problem statementto understand your customer's point of view. The Customer Problem Statement template helps you focus on what matters to create experiences people will love.

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.

I am	Describe customer with 3-4 key characteristics - *who are they?*	Describe the customer and their attributes here
I'm trying to	*List their outcome or "job" the care about - what are they trying to achieve?*	List the thing they are trying to achieve here
but	Describe what problems or barriers stand in the way – *what bothers them most?*	Describe the problems or barriers that get in the way here
because	Enter the "root cause" of why the problem or barrier exists – *what needs to be solved?*	Describe the reason the problems or barriers exist
which makes me feel	Describe the emotions from the customer's point of view – *how does it impact them emotionally?*	Describe the emotions the result from experiencing the problems or barriers

Reference: https://miro.com/templates/customer-problem-statement/

Example:



Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	A casual music listener	Discover and play my favourite songs easily	The existing options are either cluttered or require unnecessary sign-ups	I just want a simple and quick way to listen to music	Frustrated and uninterested in complicated platforms

PS-2	A user exploring new music	Find new songs based on my preferences	Most platforms push mainstream tracks rather than personalized recommendations	I prefer fresh, unique music rather than what’s always trending	Disconnected from my personal taste and less excited about using the platform
PS-3	A multitasker	Listen to music while working or studying	The platform has interruptions like ads or buffering issues	I need a seamless experience without distractions	Annoyed and frustrated with constant disruptions
PS-4	A mobile user	Stream music on my phone with a smooth interface	The UI is not optimized for mobile or lacks essential features	I want a responsive and intuitive design that makes navigation easy	Frustrated and likely to switch to another app

RhythmicTunes: Your Melodic Companion
Ideation Phase
Empathize&Discover

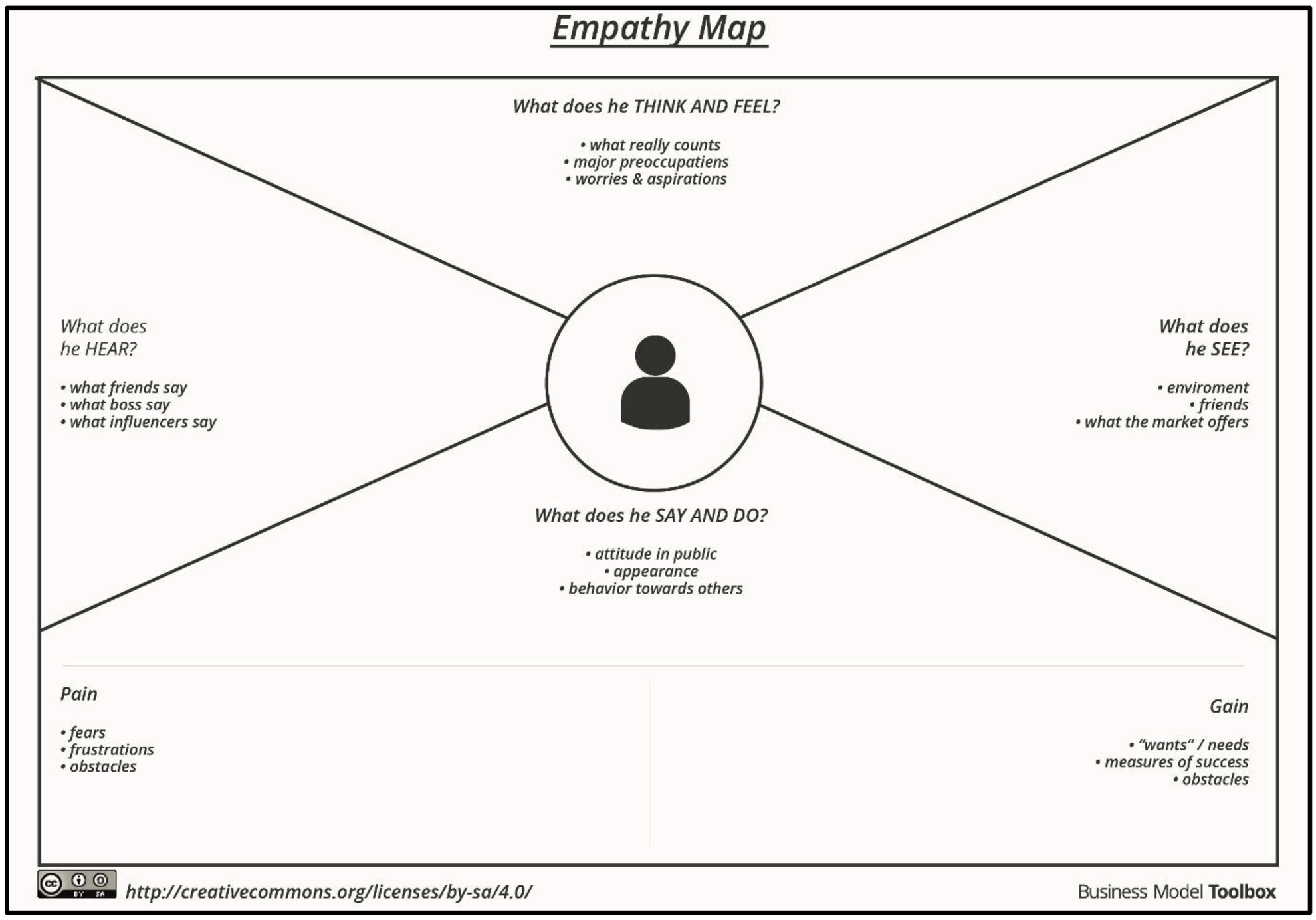
Empathy Map Canvas:

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:



Reference: https://www.mural.co/templates/empathy-map-canvas

Says

What have we heard them say?
What can we imagine them saying?

I want an easy way to discover new music.

Thinks

What are their wants, needs, hopes, and dreams?
What other thoughts might influence their behavior?

Which platform gives me the best music recommendations?

Shivani Kapoor
Music Listener

Searches for new songs and creates playlists.

Excited when discovering a new favorite song.

Does

What behavior have we observed?
What can we imagine them doing?

Feels

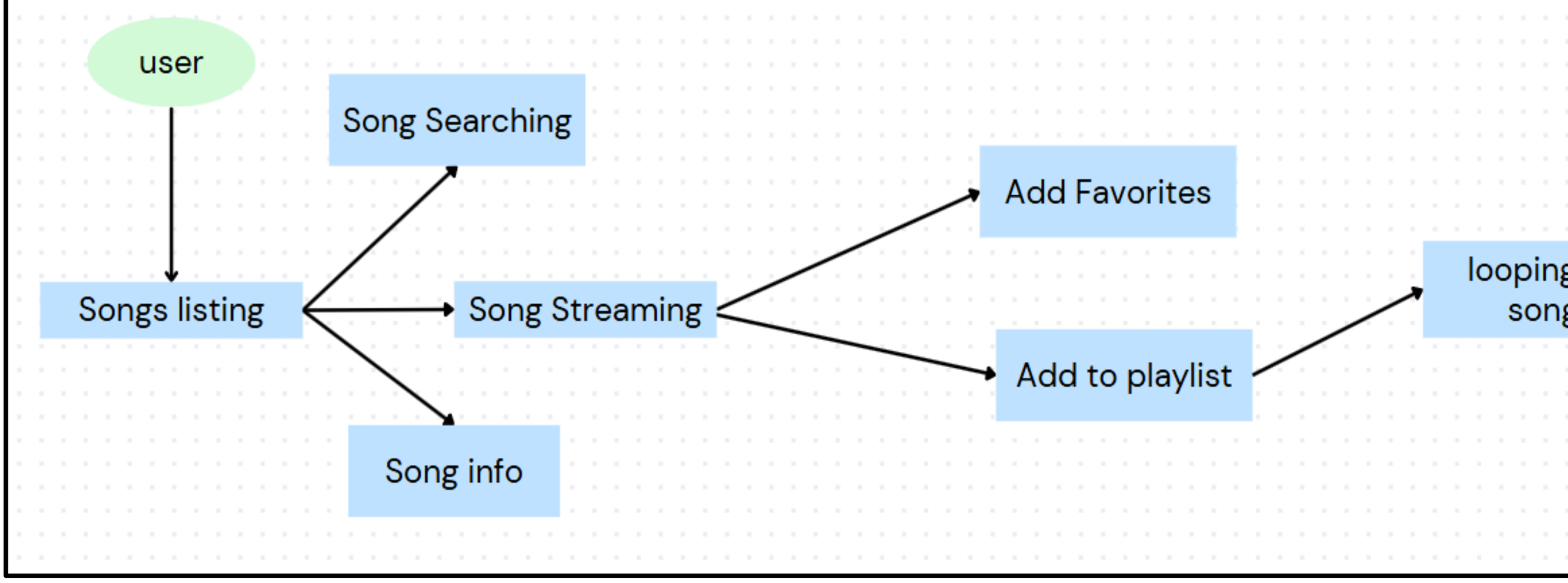
What are their fears, frustrations, and anxieties?
What other feelings might influence their behavior?

See an example

Project Design Phase-II
Data Flow Diagram&User Stories

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: DFD Level 0 (Industry Standard)



User Stories

User Story Table – Music Streaming App

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptanc
Music Discovery		USN-1	As a user, I can search for songs, albums, or artists.	I can see a l results.
		USN-2	As a user, I can view trending and recommended songs.	I can see re my dashboa
Playback		USN-3	As a user, I can play, pause, and skip songs.	I can contro buttons.
		USN-4	As a user, I can view album artwork and song details while playing a song.	I can see alb artist name.
Playlists&Favorites		USN-5	As a user, I can create my own playlists.	I can save a under a cus
		USN-6	As a user, I can add or remove songs from my playlists.	I can succes within a play
		USN-7	As a user, I can like/favorite songs.	I can save m access them
Audio Streaming		USN-8	As a user, I can stream high-quality audio.	I can listen t buffering.

Project Design Phase-II
Solution Requirements (Functional&Non-functional)

Functional Requirements:

Following are the functional requirements of the proposed solution.

Functional Requirements – Music Streaming App

FR No.	Functional Requirement (Epic)	Sub Requirement(Story / Sub-Task)
FR-1	**Music Search&Discovery**	Search for Songs, Albums, and Artists
		View Trending and Recommended Music
FR-2	**Playback&Streaming**	Play, Pause, and Skip Songs
		Display Album Art and Song Details
FR-3	**Playlist&Favorites**	Create and Manage Playlists
		Add or Remove Songs from Playlists
		Like / Favorite Songs
FR-4	**Audio Streaming**	Stream High-Quality Audio

Non-functional Requirements:

Following are the non-functional requirements of the proposed solution.

NFR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	The app should have an intuitive and user-friendly interface, ensuring smooth navigation and accessibility for users of all demographics.
NFR-2	**Security**	User authentication and data must be secured using encryption (e.g., HTTPS, OAuth for third-party logins). The app should prevent unauthorized access and follow best security practices.
NFR-3	**Reliability**	The app should ensure a consistent and uninterrupted music streaming experience, minimizing crashes and downtime.
NFR-4	**Performance**	Songs should load and stream with minimal buffering. The app should respond to user interactions (search, playback, playlist management) within 2 seconds.
NFR-5	**Availability**	The system should maintain an uptime of at least 99.9%, ensuring accessibility across different time zones.
NFR-6	**Scalability**	The app should handle increasing numbers of users and concurrent streams efficiently without performance degradation. The architecture should support future feature expansion.

Project Design Phase-II
Technology Stack (Architecture&Stack)

Technical Architecture:
The Deliverable shall include the architectural diagram as below and the information as per the table1&table 2
Example: Rhythmic Tunes

Reference: https://open.spotify.com/

Table-1 : Components&Technologies:

S.No	Component	Description	Tech
	User Interface	Web-based interface for music streaming	HTM
	Application Logic-1	Music streaming and metadata management	Reac
	Application Logic-2	Playlist and user preference management	Reac
	Database	Stores Songs, playlists, and metadata	JSO

Table-2: Application Characteristics:

S.No	Characteristics	Description	Techn
	Open-Source Frameworks	Frontend frameworks	React
	Scalable Architecture	3-tier architecture with RESTful APIs	Micro

References:
React.js Documentation
Node js Best Practice
JSON Web Server Referance
https://medium.com/the-internal-startup/how-to-draw-useful-technical-architecture-diagrams-2d20c9fda90d

Project Planning Phase

Project Planning Template (Product Backlog, Sprint Planning, Stories, Story points)

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Product Backlog&Sprint Schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task
Sprint-1	**Music Search&Discovery**	USN-1	As a user, I can search for songs, albums, or artists.
		USN-2	As a user, I can view trending and recommended songs.
Sprint-2	**Playback&Streaming**	USN-3	As a user, I can play, pause, and skip songs.
Sprint-3	**Playlist&Favorites**	USN-4	As a user, I can create my own playlists.
		USN-5	As a user, I can add or remove songs from my playlists.
		USN-6	As a user, I can like/favorite songs.
Sprint-3	**Audio Streaming**	USN-7	As a user, I can stream high-quality audio.
Sprint-4	**User Profile Management**	USN-8	As a user, I can update my profile information.
		USN-9	As a user, I can change my password.
Sprint-4	**Customer Support**	USN-10	As a user, I can contact support via in-app chat.

Project Tracker, Velocity&Burndown Chart

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as
Sprint-1	20	6 Days	1 Mar 2025	2 Mar 2025	20
Sprint-2	20	6 Days	3 Mar 2025	4 Mar 2025	20
Sprint-3	20	6 Days	5 Mar 2025	6 Mar 2025	20
Sprint-4	20	6 Days	7 Mar 2025	8 Apr 2025	20

Project Planning Phase

Project Planning Template (Product Backlog, Sprint Planning, Stories, Story points)

Product Backlog, Sprint Schedule, and Estimation (4 Marks)

Product Backlog&Sprint Schedule

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task
Sprint-1	**Music Search&Discovery**	USN-1	As a user, I can search for songs, albums, or artists.
		USN-2	As a user, I can view trending and recommended songs.
Sprint-2	**Playback&Streaming**	USN-3	As a user, I can play, pause, and skip songs.
Sprint-3	**Playlist&Favorites**	USN-4	As a user, I can create my own playlists.
		USN-5	As a user, I can add or remove songs from my playlists.
		USN-6	As a user, I can like/favorite songs.
Sprint-3	**Audio Streaming**	USN-7	As a user, I can stream high-quality audio.
Sprint-4	**User Profile Management**	USN-8	As a user, I can update my profile information.
		USN-9	As a user, I can change my password.
Sprint-4	**Customer Support**	USN-10	As a user, I can contact support via in-app chat.

Project Tracker, Velocity&Burndown Chart

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as
Sprint-1	20	6 Days	1 Mar 2025	2 Mar 2025	20
Sprint-2	20	6 Days	3 Mar 2025	4 Mar 2025	20
Sprint-3	20	6 Days	5 Mar 2025	6 Mar 2025	20
Sprint-4	20	6 Days	7 Mar 2025	8 Apr 2025	20

Project Design Phase
Problem – Solution Fit Template

Problem – Solution Fit Overview:

The **Problem-Solution Fit** ensures that the identified problem aligns with the needs of music listeners and that the proposed solution effectively addresses it. This validation is crucial before further development.

Purpose:

Address the fragmented music streaming experience, where users struggle to find a comprehensive platform that caters to diverse musical tastes and offers personalized recommendations.
Provide an intuitive and engaging platform for users to discover new music, artists, and playlists without relying on multiple sources.
Offer seamless playback, offline listening, and social sharing features to enhance user engagement and satisfaction.
Provide a platform that empowers independent artist to gain exposure.
Improve accessibility and engagement through an **interactive UI, responsive design, and well-structured data flow**.

Problem Statement:

Many music enthusiasts face challenges in finding a single platform that offers:
A vast and diverse music library.
Accurate and personalized music recommendations.
Reliable offline listening capabilities.
A strong social community around music.
Fair exposure for independent artists.

Solution:

“Rhythmic Tunes”, a music streaming web and mobile application, will provide:

An extensive music library through partnerships with major and independent record labels.
personalized recommendations based on user listening history and preferences.
Offline listening mode for downloaded playlists and albums.
Dedicated artist pages, to help users find more information about the artist.
A freemium business model, that allows for free and paid users.

Project Design Phase

Proposed Solution Template

Proposed Solution for Music Streaming App

S. No.	Parameter	Description
1	**Problem Statement** (Problem to be solved)	Users often struggle to find a seamless, personalized music streaming experience that offers high-quality playback, real-time recommendations, and an intuitive user interface. This project aims to provide a responsive and engaging music streaming platform that integrates with a third-party API.
2	**Idea / Solution Description**	The Music Streaming App is a React-based frontend solution that allows users to search, stream, and manage their favourite music. It leverages a third-party API to fetch music data and ensures an intuitive and interactive UI/UX for a seamless experience. Users can create and manage playlists, search for songs/artists, and enjoy personalized recommendations.
3	**Novelty / Uniqueness**	- **Personalized Playlists&Recommendations** using AI-powered API suggestions - **Intuitive UI/UX** with responsive design across devices - **Cross-Platform Support** for both mobile and web users - **Seamless Integration** with third-party music APIs for vast content availability
4	**Social Impact / Customer Satisfaction**	- Provides an **ad-free music streaming experience** with customizable playlists - **Brings emerging artists** to a wider audience through recommendations - **Increases user engagement** through interactive features like favourite tracks, genre-based suggestions, and social sharing
5	**Business Model (Revenue Model)**	- **Freemium Model**: Free tier with ads, premium subscription for an ad-free experience - **In-App Purchases**: Exclusive access to curated playlists, offline downloads - **Affiliate Partnerships**: Integration with brands and music merchandise stores
6	**Scalability of the Solution**	- Can be **expanded globally** by integrating multiple third-party music APIs - Supports **multi-user profiles** for enhanced personalization - Can be extended to **mobile applications** for iOS & Android using React Native - Potential integration with **AI-driven recommendation engines** for improved user experience

Project Design Phase
Solution Architecture

Solution Architecture:

The solution architecture for **HarmonyStream**, the Rhythmic Tunes Application, ensures a **scalable, high-performance, and immersive platform** for discovering, streaming, and organizing music across genres, artists, and personalized playlists. The architecture prioritizes seamless audio delivery, real-time recommendations, and cross-device synchronization to enhance user engagement.

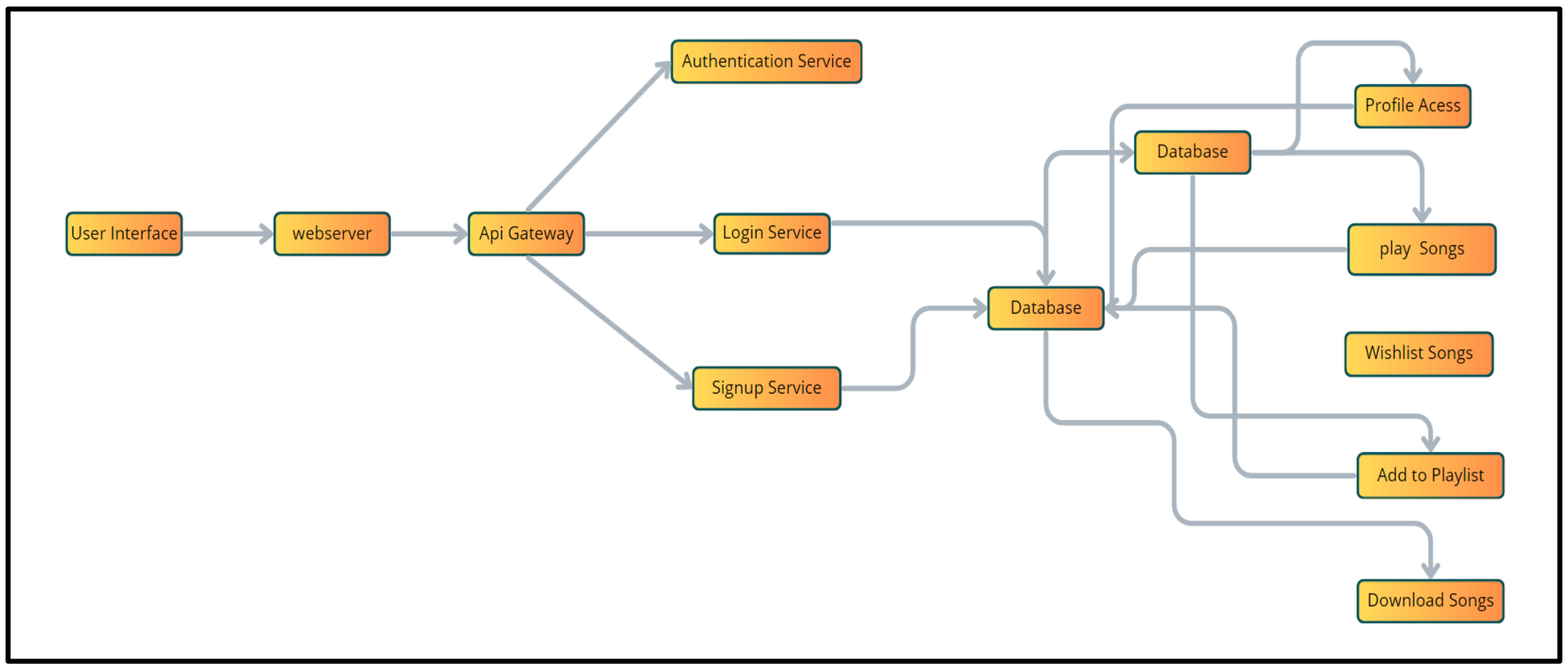
Find the best tech solution to solve existing business problems.

Describe the structure, characteristics, behavior, and other aspects of the software to project stakeholders.

Define features, development phases, and solution requirements.

Provide specifications according to which the solution is defined, managed, and delivered.

Example - Solution Architecture Diagram:



User Acceptance Testing (UAT) Template

Project Overview:-
Project Name: Rhythmic Tunes
Project Description: A React-based music streaming application that allows users to search, play, and manage music using a third-party API. Features include,search, playback, playlists, and profile management.
Project Version: v1.0
Testing Period: March 1, 2025 - March 8, 2025

Testing Scope:-
Features and Functionalities to be Tested

✅ **Music Search&Discovery**

✅ **Audio Playback (Play, Pause, Skip)**

✅ **Playlist Management (Create, Edit, Delete Playlists)**

✅ **Profile Management (Edit Profile, Change Password)**

✅ **Streaming Quality&Performance Testing**

✅ **Responsive UI Testing (Mobile, Tablet, Desktop)**

User Stories or Requirements to be Tested

📌 **User Registration&Authentication**
📌 **Searching&Viewing Music Recommendations**
📌 **Playing&Controlling Music Playback**
📌 **Creating and Managing Playlists**
📌 **Updating Profile Information**

Testing Environment
URL/Location: [Enter Web URL]
Credentials (if required):
Test User: testuser@example.com / Test@1234
Admin User: admin@example.com / Admin@1234

Test Cases

Test Case ID	Test Scenario	Test Steps	Expected Result	Actual Result	Pass/Fail
TC-001	Music Search Functionality	1. Enter a song name in the search bar 2. Click Search	Matching songs should be displayed	[Actual Result]	[Pass/Fail]
TC-002	Music Playback (Play/Pause)	1. Click on a song 2. Click Play/Pause button	Song should start/stop playing	[Actual Result]	[Pass/Fail]
TC-003	Playlist Creation	1. Navigate to Playlists 2. Click"Create New Playlist"3. Enter Name&Save	Playlist should be created successfully	[Actual Result]	[Pass/Fail]
TC-004	UI Responsiveness (Mobile)	1. Open app on mobile device 2. Navigate through pages	UI should be responsive and properly displayed	[Actual Result]	[Pass/Fail]

Bug Tracking

Bug ID	Bug Description	Steps to Reproduce	Severity	Status	Additional Feedback
BG-001	Music playback is lagging on slow networks	1. Play a song on 3G network 2. Observe buffering issues	Medium	In Progress	Optimization required for low-speed connections
BG-002	UI overlaps on small screen devices	1. Open app on iPhone SE 2. Observe overlapping UI	Low	Open	Adjust CSS for small screens

Sign-off
Tester Name: [Enter Name]
Date: [Enter Date of Completion]
Signature: [Enter Signature]

Notes
Ensure testing covers both positive&negative cases
Bug tracking should includeseverity levels&reproduction steps
Final sign-off required before deployment

Frontend Development with React.js

Project Documentation for Rhythmic Tunes

3. Architecture

- **Component Structure**:

  The application is built using React.js with a component-based architecture. Major components include:

  - **Header**: Contains the navigation bar and search bar.
    - **Player**: Music player controls (play, pause, volume, etc.).
    - **Sidebar**: Displays user playlists and navigation links.
    - **HomePage**: Displays featured tracks, recommended playlists, and new releases.
    - **SearchPage**: Allows users to search for songs, albums, and artists.
    - **PlaylistPage**: Displays user-created playlists and allows playlist management.

- **State Management**:

  The application uses **Redux** for global state management. The Redux store manages user authentication, current playing track, playlist data, and search results.

- **Routing**:

  The application uses **React Router** for navigation. Routes include:

  - /: Home page
  - /search: Search page
  - /playlist/:id: Playlist details page
  - /login: Userlogin page

4. Setup Instructions

- **Prerequisites**:
  - Node.js (v16 or higher)
  - npm (v8 or higher)
  - Git
- **Installation**:

1. Clone the repository: git clone

https://github.com/mithunan472/Project--rhythmic-tunes-your-melodic-companion-.git

2. Navigate to the client directory: cd rhythmic-tunes/client
3. Install dependencies: npm install
4. Configure environment variables: Create a .env file in the client directory and add the necessary variables (e.g., API keys).
5. Start the development server: npm start

5. Folder Structure

- **Client**:
    - **src/components:** # Reusable components (Header, Player, etc.)
    - **src/pages:** # Page components (HomePage, SearchPage, etc.)
    - **src/assets:** # Images, icons, and other static files
    - **src/redux:** # Redux store, actions, and reducers
    - **src/utils:** # Utility functions and helpers
    - **App.js:** # Main application component
    - **index.js:** # Entry point
- **Utilities**:
    - **api.js**: Handles API requests to the backend.
    - **auth.js**: Manages user authentication and token storage.
    - **hooks/usePlayer.js**: Custom hook for managing the music player state.

---

6. Running the Application

Frontend:

- To start the frontend server, run the following command in the client directory: npm start
- npm install
- npx json-server ./db/db.json
- npm run dev
- The application will be available at http://localhost:3000

---

7. Component Documentation

- **Key Components**:
    - **Header**: Displays the navigation bar and search bar.
        - Props: onSearch (function to handle search queries).
    - **Player**: Controls the music playback.
        - Props: currentTrack (object containing track details), onPlay, onPause, onSkip.

- **PlaylistCard**: Displays a playlist with its name and cover image.
    - Props: playlist (object containing playlist details), onClick (function to handle playlistselection).

- **Reusable Components**:
    - **Button**: A customizable button component.
        - Props: text, onClick, disabled.
    - **Input**: A reusable input field for forms and search.
        - Props: type, placeholder, value, onChange.

---

8. State Management

- **Global State**:
  The Redux store manages the following global states:
    - **user:** Current authenticated user.
    - **player:** Current playing track, playback status (playing/paused), and volume.
    - **playlists:** User-created playlists.
    - **searchResults:** Results from the search functionality.

- **Local State**:
  Local state is managed using React's useState hook within components. For example, the SearchPage component manages the search query input locally.

---

9. User Interface

- **Screenshots**
    - **Home Page:** Display featured tracks and recommended playlists.
    - **Search Page:** Allows users to search for songs, albums, and artists.
    - **Playlist Page:** Displays user-created playlists and allows playlist management.

10. Styling

- **CSS Frameworks/Libraries**:

  The application uses **Styled-Components** for styling. This allows for modular and scoped CSS within components.

- **Theming**:

  A custom theme is implemented using Styled-Components, with support for light and dark modes.

---

11. Testing

- **Testing Strategy**:
  - **Unit Testing:** Using **Jest** and **React Testing Library**.
  - **Integration Testing**: Is performed to ensure that components work together as expected.
  - **End-to-End Testing: Cypress** is used for end-to-end testing of user flows.
- **Code Coverage**:
  - Code coverage is monitored using Jest's built in coverage tool. The current coverage is 85%.

---

12. Screenshots or Demo

- **Demo Link:** https://drive.google.com/drive/folders/1W0f_M9ANcEajhLlkEVaJSTYvzWAC4OpB
- **Screenshots:** See section 9 for UI screenshots.

13. Known Issues

- **Issue 1**: The music player sometimes skips tracks unexpectedly.
- **Issue 2**: The search functionality is slow with large datasets.

14. Future Enhancements

- **Future Features**:
    - Add support for user profiles and social sharing.
    - Implement a recommendation engine for personalized music suggestions.
    - Add animations and transitions for a smoother user experience.

This documentation provides a comprehensive overview of the **Rhythmic Tunes** project, including its architecture, setup instructions, and future plans.
