Software Design Document (SDD)
AI Music Recommender System
Snapshot 1 - Core System

Project Group 5

December 7, 2025

Version Description

Version	Description	Date
1.0	Initial SDD for Snapshot 1 of the AI Music Recommender System. Describes base architecture, core components, and database design.	December 7, 2025

1 Introduction

1.1 Purpose

This document describes the internal design and architecture for Snapshot 1 of the AI Music Recommender System, mapping the SRS requirements to implementable components.

1.2 Scope

Snapshot 1 includes a client interface, backend services, a simple recommendation component, and a relational database for user, catalog, and interaction data.

1.3 Intended Audience

The audience includes project team members, the course instructor, teaching assistants, and future developers who need to understand the initial design.

2 System Overview

The system is composed of:

- A client application for user interaction.

- Backend services providing REST APIs for authentication, catalog, recommendations, and interaction logging.
- A recommendation engine using past interactions for basic personalization.
- A database layer storing users, songs, artists, and interactions.

3 Architecture Design

3.1 Workflow

1. The user opens the client and logs in.
2. The client sends credentials to the authentication API.
3. On success, the client requests recommended songs for the user.
4. The backend reads interaction history, runs recommendation logic, and returns a ranked list of tracks.
5. The user plays songs and can like or skip them.
6. The client logs play, like, and skip events via the interaction API, which stores them in the database.

3.2 Main Components

3.2.1 Client Application

- Login/registration screens.
- Home screen displaying recommended songs.
- Search screen.
- Now-playing screen with playback and feedback controls.

3.2.2 Backend Services

- Authentication module (register, login).
- Catalog module (list songs, search songs).
- Recommendation module (fetch recommendations).
- Interaction logger (store plays, likes, skips).

3.2.3 Recommendation Engine

- Reads the user's interaction history.
- Computes simple scores based on likes and skips.
- Produces a ranked list of candidate tracks for the backend.

4 Database Design

4.1 Schema

Entities for Snapshot 1:

- User(userId, name, email, passwordHash, createdAt).
- Artist(artistId, name).
- Track(trackId, title, artistId, album, genre, duration).
- Interaction(interactionId, userId, trackId, actionType, timestamp).
- Playlist(optional)(playlistId, userId, name).

4.2 Relationships

- One User to many Interactions.
- One Track to many Interactions.
- One Artist to many Tracks.
- One User to zero or many Playlists.

5 Interface Design

5.1 API Endpoints (Logical)

Example logical endpoints:

- POST /auth/register
- POST /auth/login
- GET /songs/recommended
- GET /songs/search
- POST /interactions

6 Design Constraints

- Design must be simple enough for implementation within the course schedule.
- Components should be modular to support additional features in later snapshots.

7 Glossary

- UI: User Interface.
- API: Application Programming Interface.
- DB: Database.
- SRS: Software Requirements Specification.
- SDD: Software Design Document.