Google Summer Of Code 2025-Proposal

Organisation Name: Open Technologies Alliance - GFOSS

Title: Generative AI Agent for Personalized Music Recommendations

K Sai Sri Lekha

kslekha02gmail.com

Hyderabad, Telangana, India

Timezone: Indian Standard Time (+5:30 UTC)

Table of Contents

1 Basic Information .

2 Background .

2.1 Education & Interests .

2.2 Relevant Experience .

2.3 Motivation for Selecting this Project .

3 Why do I want to work with GFOSS? .

4 My Skills .

5 Project Overview .

5.1 Technology Stack .

5.2 Core Features .

6 Implementation .

6.1 Frontend .

6.2 Backend .

6.3 AI Recommendations .

6.4 Spotify API Integration .

6.5 Database .

Flexibility For Modifications .

7 Timeline .

7 Non- Summer Of Code Plans .

8 Why am I a good fit for this? .

9 References .

Basic Information

Name	K Sai Sri Lekha
Location	Hyderabad, Telangana, India
University	Indian Institute of Technology- Madras, Anurag University- Hyderabad
Email address	kslekha02@gmail.com
Timezone	GMT+5:30
Github	Lekha0204
LinkedIn	K SAI SRI LEKHA
Timezone	UTC+5:30

Background

Education & Interests

I am currently pursuing dual studies at Anurag University and IIT Madras, where I have developed a strong foundation in artificial intelligence, machine learning, and data-driven applications. My interests span a wide range of domains, including natural language processing, recommendation systems, and real-time AI applications.

Music has always been an integral part of my daily routine. I enjoy exploring diverse genres, and a major portion of my day revolves around music, which has fueled my curiosity about the intersection of AI and personalized music experiences.

Relevant Experience

I have worked on research paper classification and rationale generation using NLP, where I employed cosine similarity on semantic vector representations to categorize research papers into the most relevant organizations based on textual and contextual alignment. Additionally, I leveraged the Gemini API to generate detailed rationale for classification decisions, ensuring interpretability and robustness in the results. This project enhanced my ability to design AI-driven solutions that bridge data insights with real-world applications.

Resume: tap here

Motivation for Selecting This Project

My motivation for this project stems from my passion for both AI and music. Understanding how mood influences music preferences presents an exciting challenge in AI-driven personalization. By developing an intelligent recommendation system that aligns music choices with users' emotional states, I aim to create a seamless and immersive listening experience. My experience in text classification, NLP**,** and AI-powered decision-making provides a strong foundation for building an adaptive and context-aware recommendation model.

Why Do I Want to Work with This Organization?

GFOSS – Open Technologies Alliance promotes open-source software, open data, and free technology, working closely with universities, research centers, and public institutions. Their efforts in digital transformation and open-source development make a real impact, especially in education and governance.

I’m interested in GFOSS because they focus on real-world applications of open-source technology. Their projects not only encourage software development but also help in making technology more accessible and useful.

Through GSoC, I see a great opportunity to contribute to an open-source project under GFOSS, learn from experienced mentors, and be part of a community that values openness and collaboration.

My Skills

- Python, C, C++, Java, HTML, CSS, JS
- DSA, Competitive Programming, Machine Learning, NLP, Data Analysis, Web Scraping, API Integrations, Vector Space Models

Project Overview

This project aims to develop an AI-powered music recommendation system that interacts with users in natural language to gauge their emotional state and music preferences. Through an interactive chatbot interface, users can express their mood, and the system will dynamically curate playlists tailored to their emotional state.

The system will integrate with the Spotify API to refine recommendations based on listening history, liked songs, and personal preferences. Users who prefer a discovery-based experience can opt out of personal data usage, allowing for a fresh set of recommendations. Additionally, real-time conversational modifications will enable users to adjust playlists by requesting changes in mood, energy level, or genre.

Technology Stack

- **Frontend**: HTML, CSS, JavaScript
- **Backend**: Django (Python)
- **AI & ML Models**: NLP and generative AI models (Python)
- **Database (if required)**: PostgreSQL
- **API Integration**: Spotify API for authentication, playlist retrieval, and playback control

Core Features

1. **Conversational Playlist Curation**
    - The AI agent will analyze user **text-based inputs** to determine mood and musical preferences.
    - It will generate a playlist based on **user history, preferences, and contextual cues**.
2. **Spotify API Integration**
    - Authentication via OAuth for secure access to user data.
    - Fetching user's liked songs, playlists, and listening habits.
    - Creating and modifying playlists dynamically based on AI recommendations.

3. **Real-Time Modifications**

   - Users can interact with the chatbot to **adjust mood, genre, tempo, or specific songs**.

   - Playlists will be **updated dynamically** with a shuffle and rearrange effect for a smooth transition.

4. **Embedded Music Player**

   - The web app will include an **embedded Spotify player** for seamless playback.

   - Users can **play, pause, skip, and control volume** directly within the interface.

   - Song details like **album art, track name, and artist** will be displayed.

5. **AI-Based Music Recommendations**

   - The system will leverage **machine learning models** for **mood detection** from conversation.

   - **LLMs (small-scale or API-based)** will generate relevant **song suggestions** based on user input.

6. **User Privacy & Control**

   - Users can **opt-out** of personal data usage for a more **exploratory** music experience.

- The system will allow **manual refinement** of recommendations beyond AI-based suggestions.

Implementation

1. Frontend (User Interface)

- **Technologies**: HTML, CSS, JavaScript
- **Description**: A web-based chatbot-style UI that interacts with users, collects mood preferences, and displays the generated playlist.

2. Backend (Django - Python)

- **Technologies**: Django for request handling, API communication, and user authentication
- **Description**: Backend will process user inputs, communicate with AI models, and manage playlist recommendations via Spotify API.

3. AI Recommendations (Python - NLP)

- **Technologies**: NLP for mood detection, AI models for recommendations
- **Description**: A lightweight AI model will analyze user inputs and suggest songs based on sentiment analysis.

4. Spotify API Integration

- **Technologies**: Spotify Web API

- **Description**: Retrieves user playlists, adds recommended songs, and enables real-time music control.

5. Database (PostgreSQL - If Required)

- **Description**: Store user preferences, playlist history, and session data.

Flexibility for Modifications:

I am open to any modifications or improvements suggested by the mentors to refine the approach, optimize the system, or explore additional features that enhance the project. The plan can be adapted based on expert insights to ensure the best possible outcome.

Timeline

Time Period	Tasks
Community Bonding Period (May 8 - June 1)	- Research Spotify API and Spotipy library. - Learn about chatbot frameworks (Rasa, ChatterBot, LangChain, or custom NLP). - Set up the development environment (Django backend, PostgreSQL, and chatbot framework). - Define chatbot scope: Commands, queries (playlist management, song search, recommendations). - Discuss project milestones and API rate limits with mentors.

Coding Begins (June 2)	- Implement OAuth authentication for user login via Spotify. - Develop basic Django backend structure. - Set up the chatbot framework (initial intents & responses).
Work Period (Phase 1) (June 2 - July 14)	- Develop core chatbot functionalities: ◦ Fetch user playlists & song data. ◦ Search for songs, albums, artists via chatbot commands. ◦ Basic natural language processing (NLP) for user queries. - Integrate chatbot with Spotipy API for real-time responses. - Implement a simple UI for chatbot interactions (React/HTML+JS). - Conduct initial testing of bot responses. - Write documentation for chatbot setup & API usage.
Midterm Evaluation Submission (July 14 - July 18)	- Submit midterm evaluation with a working chatbot prototype. - Review mentor feedback and refine chatbot features.
Work Period (Phase 2) (July 14 - August 18)	- Enhance NLP capabilities (better intent recognition, context awareness). - Implement song recommendations based on listening history. - Optimize API request handling (reduce latency, improve caching). - Improve chatbot UI (smooth animations, dark mode, responsiveness).

	- Implement voice command support (optional feature). - Conduct extensive bug fixing, performance improvements. - Write user guide and finalize documentation.
Buffer Period (August 18 - August 25)	- Address pending tasks & bugs. - Review chatbot edge cases & test against different user inputs. - Polish UI/UX improvements based on feedback. - Finalize and review documentation.
Final Week (Submission & Evaluation) (August 25 - September 1)	- Conduct final debugging & testing. - Submit the final work product. - Complete final mentor evaluation.
Mentor Evaluations (September 1 - 8)	- Mentors submit final evaluations.
Flexibility Note	The buffer week ensures extra time for refinements & unforeseen issues.

Non-Summer of Code Plans

My final exams for the current semester will be completed by the first week of May 2025. My university commences a new academic session by June, 2025. During the GSoC period, my only commitment apart from this project will be my academics. I can devote about 25-30 hours per week, or more if required, ensuring a balance between my coursework and GSoC without compromising on either.

Why am I a good fit for this?

I have a strong foundation in Python, machine learning, and web development, which aligns well with the requirements of this project. My experience includes working with Django for backend development, along with HTML, CSS, and JavaScript for frontend development. Additionally, I have explored APIs and external integrations, making me comfortable with handling data and implementing various functionalities within a project.

I am highly adaptable and quick to grasp new concepts, always ready to learn and improve. I approach problem-solving with determination and enjoy taking on challenges that push me to grow. My dedication and enthusiasm for this project ensure that I will contribute effectively while continuously refining my skills.

Furthermore, I am open to feedback and willing to make modifications based on mentor guidance. This ensures that my contributions align with the project's objectives while maintaining flexibility in implementation.

References

- https://medium.com/@gajbhiyeshreya23/personalized-music-playlist-recommendation-system-776bfabd50c2
  This blog explains how music recommendation systems work, covering three types: content-based (suggests songs by analyzing audio features), collaborative filtering (recommendations based on user preferences), and hybrid models (a mix of both for better accuracy).

- https://moodify.toasted.ai/
- https://www.ischool.berkeley.edu/sites/default/files/projects/w205-1moodify-finalreport.pdf

  Moodify - *A Music Recommender by Mood*: Developed as a student project at UC Berkeley's School of Information, this web application recommends songs to users based on their selected mood. It also features an interactive map displaying the most popular moods in various regions worldwide.

- https://github.com/mahnoorshafi/Moodify
- Spotify Web API Documentation: https://developer.spotify.com/documentation/web-api