Project Proposal for Emotion-Aware Lyrics Generation System

1. Title:

   Emotion-Aware Lyrics Generation System

2. Project Summary

   The *Emotion-Aware Lyrics Generation System* is an AI-powered application designed to generate personalized song lyrics based on a user's facial emotion. The system utilizes computer vision and deep learning (Convolutional Neural Networks) to detect the user's emotional state from a captured image. Once the emotion is identified, a Natural Language Processing (NLP) model—such as GPT-2 from Hugging Face—is used to generate emotionally aligned lyrics that reflect the user's current mood.

   The goal is to create an engaging and creative experience where users can see AI-generated lyrics that match their emotions in real-time. The entire system will be deployed through a web-based interface using **Streamlit**, which ensures ease of use, quick interaction, and a smooth user experience. This project integrates multiple areas of artificial intelligence, including computer vision, deep learning, NLP, and web development.

   This end-to-end solution will not only enhance understanding of emotion detection and text generation but also provide practical experience in deploying machine learning models as interactive applications.

3. Objectives

   1. Emotion Detection via Computer Vision

   Develop a facial emotion recognition system using Convolutional Neural Networks (CNN) to accurately detect the emotional state (e.g., happy, sad, angry, etc.) of a user from an uploaded image or webcam input.

   ---

   2. Emotion-Driven Lyrics Generation

   Use a pre-trained Natural Language Processing (NLP) model (e.g., GPT-2 from Hugging Face) to generate lyrics that align with the detected emotion, ensuring that the tone and style of the lyrics reflect the user's mood.

---

3 . Interactive Web Application with Streamlit
Build a user-friendly and responsive web interface using Streamlit, allowing real-time image input, emotion detection display, and automatic generation and visualization of the corresponding lyrics.

---

4. Project Documentation and Presentation
Prepare comprehensive documentation, including a final report, presentation slides (PPT), and a demo video to clearly demonstrate the project objectives, methodology, results, and conclusions.

4. Methodology

The development of the *Emotion-Aware Lyrics Generation System* will follow a modular and phased approach, integrating computer vision, deep learning, and NLP into an interactive web-based application. The methodology is divided into four main stages.

---

1. Emotion Detection Module (Computer Vision + CNN)

- Collect or use an existing dataset of facial images labeled with emotions (e.g., FER-2013 or RAF-DB).
- Preprocess images (resize, grayscale/color normalization).
- Train a Convolutional Neural Network (CNN) to classify at least five emotions: **Happy, Sad, Angry, Neutral, and Surprise**.
- Integrate the model into a Python script that accepts an input image (from webcam or upload) and returns the predicted emotion.

---

2. Lyrics Generation Module (NLP using Hugging Face)

- Choose a **pre-trained language model** from Hugging Face capable of generating lyrics (e.g., GPT-2 or a fine-tuned model for song lyrics).
- Create custom prompts for each emotion, e.g.:
    - *“Write a happy song verse about sunshine.”*
    - *“Generate sad lyrics about loneliness.”*

- Use the detected emotion to dynamically craft the prompt and pass it to the model.
- Generate English lyrics that align with the user's mood.

---

3. Web Application (Streamlit Interface)

- Develop a **Streamlit-based web app** to:
  - Capture user image from **webcam** using `st.camera_input()`.
  - Display the detected emotion.
  - Show the generated lyrics on the same page.
- Ensure the interface is responsive and easy to use, with minimal steps for interaction.

---

4. Finalization and Documentation

- Combine both modules into a single working pipeline.
- Conduct testing to ensure accuracy of emotion detection and coherence of lyrics.
- Prepare final **project documentation**, including:
  - Technical report
  - PPT slides for presentation
  - Demo video showcasing system functionality

5. Tools and Technologies

Category	Tool / Technology	Purpose
Programming Language	Python	Core language for model development and web integration
Computer Vision	OpenCV, TensorFlow / PyTorch	Image capture, preprocessing, and emotion classification using CNN

Deep Learning	TensorFlow / PyTorch	Building and training the emotion detection model
Natural Language Processing (NLP)	Hugging Face Transformers (e.g., GPT-2)	Generating emotion-aligned English lyrics
Web Framework	Streamlit	Interactive user interface for uploading/taking images and displaying output
Development Tools	Jupyter Notebook / VS Code	Model training, testing, and experimentation
Version Control	Git & GitHub	Code management and collaboration
Documentation	MS Word / Google Docs / Notion / Canva	Writing reports, creating slides and visuals

6. Expected Outcomes

By the end of the project, the following outcomes are expected to be achieved:

1. **Accurate Facial Emotion Detection System**
   A trained and validated CNN model capable of detecting at least five facial emotions — **Happy, Sad, Angry, Neutral, and Surprise** — with good accuracy using webcam or uploaded images.

2. **Emotion-Driven Lyrics Generator**
   A functioning NLP pipeline that uses a pre-trained model from Hugging Face (e.g., GPT-2) to generate **emotion-specific English lyrics** based on the detected facial emotion.

3. **Integrated Web Application**
   A fully operational and interactive **Streamlit web app** that allows users to capture or upload a photo, detect their emotion, and view generated lyrics

in real time.

4. **User-Centric Interface**
   A smooth, intuitive, and responsive interface that simplifies user interaction without requiring any technical knowledge.

5. **Complete Project Documentation**
   A well-documented solution that includes:
   i. Final project report
   ii. Presentation slides
   iii. Demonstration video
   iv. Source code repository

6. **Portfolio-Ready Deliverable**

   A unique, creative, and end-to-end AI project suitable for sharing on **GitHub, LinkedIn**, and academic or professional portfolios, demonstrating strong skills in computer vision, NLP, and app development.

7. Timeline

📌 Phase 1: Project Setup & Research (9–12 June)

Goal: Define scope, prepare the environment, and complete initial research.

Date	Task	Tools Used	Output
9 June	Finalize idea, objectives, deliverables	Notion / Google Docs	Project proposal draft
10 June	Research emotion datasets & models	Paperswithcode, Kaggle	Notes on datasets/models
11 June	Install dependencies, test webcam	Python, OpenCV	Working webcam setup

12 June	Draft Software Requirements Specification (SRS)	MS Word / LaTeX	`SRS.docx` file

---

📌 Phase 2: Model Development (13–20 June)

Goal: Build emotion recognition and lyrics generation components.

Date(s)	Task	Tools Used	Output
13–14 June	Train/test emotion recognition model (e.g., FER2013)	TensorFlow / PyTorch	`.h5` or `.pt` model file
15 June	Create label-to-emotion mapping	Python	`emotion_map.py`
16–17 June	Music recommendation logic or alternative flow	Spotify API / Dataset	`recommend_song.py`
18 June	Generate lyrics using Hugging Face (e.g., GPT-2)	Transformers library	`generate_lyrics.py`
19 June	Test full emotion → lyrics flow	Python	Functional prototype
20 June	Write Model Design section in report	Word / LaTeX	`report_section2.docx`

📌 Phase 3: Integration & UI (21–28 June)

Goal: Integrate models and build user interface.

Date(s)	Task	Tools Used	Output
21–23 June	Build Streamlit/Flask interface	Streamlit or Flask	`app.py`
24 June	Display lyrics or song info with emotion result	YouTube Embed / Text Display	UI update
25 June	Store outputs/logs in local DB *(optional)*	SQLite / MongoDB	`db.py`
26 June	Improve UI (buttons, emojis, colors)	Streamlit	Final interface
27–28 June	Write System Architecture & UI design documentation	Word / LaTeX	`report_section3.docx`

📌 Phase 4: Testing, Documentation & Finalization (29 June – 9 July)

Goal: Test all components, document results, and prepare for submission/demo.

Date(s)	Task	Tools Used	Output
29 June	Test modules & workflows	Manual/Unit Testing	`test_results.txt`

30 June – 1 July	Write Evaluation & Results section	Word / LaTeX	`report_section4.docx`
2 July	Add limitations & future work	Word / LaTeX	`report_section5.docx`
3 July	Assemble final report	Word / LaTeX	`Final_Report.pdf`
4 July	Create presentation slides	PowerPoint	`FYP_Presentation.pptx`
5–6 July	Record or prepare demo	OBS / Mobile Camera	`demo.mp4`
7 July	Submit project online	GitHub / Google Drive	✅ Submission Complete
8 July	Final backup and review	Git / Notion	✅ Project Wrapped
9 July	Reserved buffer or CampusX project	—	Flex day

8. Deliverables

i. Project Proposal Document
Clearly defines the idea, objectives, methodology, tools, and timeline.

ii. Trained Emotion Detection Model
Deep learning model (CNN-based) trained to detect 5+ facial emotions: Happy, Sad, Angry, Neutral, Surprise, etc.

iii. Lyrics Generation Module
A Natural Language Processing model (e.g., GPT-2 via Hugging Face) that generates English lyrics based on the detected emotion.

iv. Integrated Web Application
A fully functional web interface developed using **Streamlit**, allowing real-time webcam input, emotion detection, and lyric generation.

v. System Documentation

Includes:

- Software Requirements Specification (SRS)
- Model design details
- System architecture diagrams
- UI/UX design explanations
- Testing and evaluation reports
- Limitations and future scope

vi. Final Report (PDF)
Complete technical report covering all aspects of the project in a professional format.

vii. Presentation Slides (PPT)

Concise and engaging slides to present the project during viva/demo sessions.

viii. Demo Video
A recorded walkthrough of the project demonstrating real-time functionality.

ix. GitHub Repository

Full source code, model files, documentation, and demo video, made public for evaluation and showcasing.