Project Title: AI Mood-Based Music or Quote Recommender

Project Overview

Over the past week, I've been exploring the feasibility of creating an AI system that can detect how someone is feeling based on what they write or say, then offer personalized support through either uplifting music or motivational quotes. The goal was to determine if this concept could work before investing in full development.

Day 1: Understanding the PoC Concept (21/04/25)

1. What is a PoC and Why Do We Need It?

Today I started by understanding what a PoC (Proof of Concept) really is. Basically, it's like a small demo or prototype that shows whether an idea is possible to build or not.

In our case, we're trying to check:

- Can AI really understand how someone feels based on a sentence like *"I'm anxious"*?
- Can we then suggest something helpful, like a motivational quote or calming music?
- And can this whole system actually be built with tools like Google Gemini and YouTube API within a short time and limited resources?

That's what our PoC will help prove.

2. Types of PoC (and Where Ours Fits)

I explored different kinds of PoCs today:

- Technical PoC – Focuses on whether the technology works (that's our main focus).
- Business PoC – Checks if the idea has value in the real world (we'll touch on this lightly).
- Design/UX PoC – Focuses on whether the design is user-friendly (we might explore this later too).

So, ours is mostly technical – testing APIs, AI models, and system integration

3. What Our PoC Will Show

The purpose of our PoC is to prove that:

- A user can type something emotional in natural language (like “I feel exhausted”).
- The AI (Gemini or Sentiment API) can understand the mood correctly.
- Based on that, it can suggest the right content – either a calming song or a motivational quote.
- Bonus: If possible, it can support voice and multilingual input later on.

Key Takeaways from Today

- Starting small is smart. This PoC will help us avoid wasting time building something that may not work.
- We'll first test only the core features: emotion detection and content recommendation.
- Advanced features like voice input, multilingual support, and PDF generation will be tested once the basics are working.

Day 2: Identify Stakeholders & Requirements (22/04/25)

- Identified stakeholders: users (emotional support seekers), developers, mentors.
- Gathered core requirements:
    - Accept natural language input.
    - Use Gemini/Sentiment API for emotion detection.
    - Recommend music (Spotify/YouTube) or GenAI quotes.
    - Support PDF export, optional voice/multilingual input.
- Listed PoC key features:
    - Emotion detection
    - Quote/music recommendation
    - Basic UI + Flask backend

Success Metrics:

- Accurate emotion classification
- Relevant content recommendation
- Fast response time (<3s)
- Simple, intuitive UI
- Feasibility within timeframe

To measure success, we'd need to track:

- How accurately our system identifies emotions
- Whether recommendations truly match the user's mood
- Response speed
- Overall user engagement

The biggest challenge was avoiding feature creep. I kept reminding myself: "This is just proving the concept works, not building the full system."

Day 3: Looking at the Market and Technical Options(23/04/25)

- **Explored Similar Projects:**
    - Reviewed apps like *Youper*, *Replika*, and *Calm*.
    - Common features: mood tracking, chatbot-style responses, music/quote suggestions.
    - Gaps: limited multilingual support, no PDF export, minimal GenAI use.
- **Technical Feasibility:**
    - Emotion Detection: Feasible using **Google Gemini**, **TextBlob**, or **transformers**.
    - Quote Generation: Use **Gemini** or **OpenAI** with prompt tuning.
    - Music APIs: YouTube Search API or Spotify API (OAuth required).
    - PDF Export: Possible using Python's fpdf or reportlab.
    - Multilingual: Use **Google Translate API** or Gemini's multilingual prompts.
    - Voice Input: Feasible with **SpeechRecognition + pyaudio** or **Web Speech API** (JS).
- **Tools Stack Confirmed:**
    - **Frontend:** HTML, JS
    - **Backend:** Flask (Python)
    - **AI/GenAI:** Google Gemini
    - **Others:** YouTube API, fpdf, SpeechRecognition

Challenges:

- Some APIs (Spotify) require user authentication (OAuth). *Plan:* Start with YouTube for PoC simplicity.

.

Day 4: Design PoC Plan(24/04/25)

1.Implementation Strategy (Phases):

I sketched a straightforward architecture:

- A simple web interface for users to express their feelings
- A Flask backend to process this input
- Gemini AI to analyze the emotion
- Integration with quote generation or music services
- A clean display of the recommendations

2. Risk Analysis:

My implementation plan focused on building one component at a time:

1. First, get the emotion classification working
2. Then add recommendation capabilities
3. Only if time permitted, explore voice input and multiple languages
4. Finally, add the PDF export option

I identified several potential challenges:

- API rate limits or unexpected costs
- The complexity of supporting multiple languages
- Ensuring quick responses even when fetching recommendations

3. Resource & Budget Planning:

Resource-wise, we could start with free-tier API access, a simple Flask application, and basic HTML/JavaScript for the interface.

To manage complexity, I planned to build each component separately so we could test and troubleshoot independently.

Day 5: Conclusion & Report Creation(25/04/25)

1.Summarize Findings

On the final day, I compiled everything I'd learned into a comprehensive report. I created mockups of the user interface to show how someone would interact with the system and receive recommendations.

The project proved technically feasible: using readily available AI tools and APIs, we can indeed build a system that understands emotional cues and responds with relevant content. A lightweight Flask application using Google's Gemini AI forms the heart of the solution, with connections to content sources for the recommendations.

2. PoC Recommendations:

Key Takeaways and Recommendations

After this week-long exploration, I'm confident this concept is viable. The technology exists, the market shows interest, and the implementation approach is clear.

3. Final Report Structure

Moving forward, I recommend:

1. Building a working prototype that demonstrates the core functionality in real-time
2. Training a custom emotion classifier to improve accuracy beyond what general APIs offer
3. Expanding the input options to include voice recognition and support for multiple languages
4. Conducting user testing to refine the recommendation algorithms based on feedback

This emotional support tool has significant potential. By understanding how people feel and offering tailored content in response, we can create moments of connection and relief when they're needed most.