GEN AI HACKATHON PROJECT TEMPLATE

Project Title: Audio Transcription App Using OpenAI whisper

Team Name: CodeCrew

Team Members:

- Neshmitha Burgu
- Navya Alikanti
- Sri Lasya Challa
- Poojitha Boda

Phase-1: Brainstorming & Ideation

Objective: Develop an AI-powered audio transcription app using OpenAI Whisper to help users convert speech to text, translate audio into English, and support both file uploads and live speech input.

Key Points:

1. **Problem Statement:**

- Many users struggle to transcribe and translate audio recordings accurately and efficiently.
- There is a need for a user-friendly tool that supports real-time speech transcription and multi-language translation.

2. **Proposed Solution:**

- An AI-powered application using OpenAI Whisper to provide real-time audio-to-text transcription.
- The app supports audio file uploads and live speech recording, with automatic language detection and translation to English.

3. **Target Users:**

- Content creators and journalists needing quick transcriptions.
- Students and researchers transcribing lectures or interviews.
- Multilingual users requiring audio translations.

4. **Expected Outcome:**

- A functional AI-powered audio transcription app that processes both live and uploaded audio, offers language detection, and translates text into English seamlessly.

Phase-2: Requirement Analysis

Objective:

Define the technical and functional requirements for the **Audio Transcription App using OpenAI Whisper**.

Key Points:

1. Technical Requirements:

- **Programming Language:** Python
- **Backend:** OpenAI Whisper API, Torch, FFmpeg
- **Frontend:** Streamlit Web Framework
- **Audio Processing:** PyAudio for live speech input
- **Deployment:** local VS Code setup

2. Functional Requirements:

- **Audio File Transcription:** Upload audio/video files (MP3, WAV, MP4, etc.) and transcribe speech to text.
- **Live Speech Transcription:** Record live audio from the microphone and transcribe it in real-time.
- **Language Detection:** Automatically detect the spoken language.
- **English Translation:** Provide English translations for non-English audio.
- **User-Friendly Interface:** Display transcriptions, detected language, and translations in a clean UI.
- **Audio Playback:** Allow users to listen to uploaded or recorded audio.

3. Constraints & Challenges:

- **Model Size:** Ensuring Whisper’s base model runs efficiently without performance issues.
- **Real-Time Processing:** Minimizing latency during live transcription.
- **Audio Quality:** Handling noisy input or low-quality audio recordings.

Phase-3: Project Design

Objective: Develop the architecture and user flow of the Audio Transcription App using OpenAI Whisper.

Key Points:

1. System Architecture:

using the OpenAI Whisper model.

The model generates transcriptions in Telugu and translates them into English.

The frontend displays both the original Telugu transcription and the English translation for User uploads an audio file or records live speech via the app's UI.

The audio input is processed users.

2. User Flow:

Step 1: User selects an option — upload an audio file or record live speech.

Step 2: The backend processes the audio using the Whisper model.

Step 3: The app displays the original Telugu transcription along with the English translation in an easy-to-read format.

3. UI/UX Considerations:

Intuitive, user-friendly interface for smooth navigation.

Clear options to choose between uploading a file or live recording.

Language selection dropdown for transcription.

Real-time status indicators (e.g., 'Recording...', 'Processing...', 'Done!').

Phase-4: Project Planning

Sprint	Task	Priority	Duration	Deadline	Assigned To	Dependencies	Expected Outcome
Sprint 1	Environment Setup & API Integration	Medium	2 hours (Day 1)	End of Day 1	lasya	Python, Streamlit, OpenAI Whisper API, FFmpeg	API connection established & working
Sprint 2	Frontend UI Development	medium	3 hours (Day 2)	Mid-Day 2	Navya	API response format finalized	Basic UI with file upload & record button
Sprint 2	Audio File Processing & Transcription	Medium	1.5 hours (Day 2)	Mid-Day 2	Neshmitha	Whisper model integration	Transcription working for uploaded files
Sprint 3	Live Audio Transcription And Error handling and debugging	Medium	1.5 hours (Day 2)	Mid-Day 2	poojitha	PyAudio setup complete API logs, user inputs	real time time speech to text enabled
Sprint 3	Final Presentation & Deployment	Medium	1 hour (Day 2)	End of Day 2	Entire Team	Working prototype	Demo-ready project

Sprint 1 – Setup & Integration (Day 1)

🔴 **High Priority:** Set up the environment & install dependencies.

🟠 **Medium Priority:** Integrate OpenAI Whisper API & FFmpeg. **:** Build a basic UI with file upload & record button.

Sprint 2 – Core Features & Debugging (Day 2)

🔴 **High Priority:** Implement audio file processing & transcription.Enable real-time speech-to-text using PyAudio.

Sprint 3 – Testing, Enhancements & Submission (Day 2)

🟠 **Medium Priority:** Debug API issues, handle errors in live transcription, and refine the UI.

🟢 **Low Priority:** Final demo preparation & deployment.

Phase-5: Project Development

Objective: Implement core features of the Audio Transcription App using OpenAI Whisper.

Key Points:

1. Technology Stack Used:

Frontend: Streamlit

Backend: OpenAI Whisper model

Programming Language: Python

2. Development Process:

Integrate OpenAI Whisper for audio processing and transcription.

Implement real-time recording and file upload features.

Develop logic for Telugu transcription and English translation.

Optimize transcription accuracy and processing speed.

3. Challenges & Fixes:

Challenge: Inaccurate Telugu transcriptions. Fix: Fine-tune model parameters and test with diverse audio samples.

Challenge: Slow processing time for long audio files. Fix: Implement audio chunking and parallel processing.

Phase-6: Functional & Performance Testing

Objective:

Test Case ID	Category	Test Scenario	Expected Outcome	Status	Tester
TC-001	Functional Testing	Upload an audio file for transcription	Accurate transcription displayed	✅ Passed	Neshmitha
TC-002	Functional Testing	Test live audio transcription (real-time speech input)	Real-time text output on screen.	✅ Passed	Navya
TC-003	Performance Testing	Test transcription speed for a 1-minute audio file	Transcription completed in 5 to 10 seconds	✅ Passed	Lasya
TC-004	Bug Fixes & Improvements	Check app response with empty or invalid audio files	Data accuracy should be improved.	✅ Fixed	poojitha
TC-005	Final Validation	Ensure UI responsiveness (file upload & record button.	UI should work on mobile & desktop.	❌ Failed	Entire team
TC-006	Deployment Testing	Host the app using Streamlit Sharing	App should be accessible online.	🚀 Deployed	DevOps

---

Final Submission

1. **Project Report Based on the templates**
2. **Demo Video (3-5 Minutes)**
3. **GitHub/Code Repository Link**
4. **Presentation**