GenAI Hackathon

Hackathon Project Phases

Project Title:

Audio transcriber app using open ai whisper

Team Name:

Team curious crew

Team Members:

- Bodgam.Rishika
- Radapaka.Sushmitha
- Emmadi.Tejaswi

Phase-1: Brainstorming & Ideation

Objective:

Develop an AI powered tool to transcribe an audio into text and summarize it.

Key Points:

1. **Problem Statement:**
    - Large corporations struggle document team meetings, client calls.
2. **Proposed Solution:**
    - An AI-powered application using open ai whisper to provide key takeaways from meeting calls.
    - Accurately transcribes multi accent discussions.

3. **Target Users:**

    - **Businesses and professionals.**
    - **Content creators.**
    - **Government and public services.**
4. **Expected Outcome:**

    - A functional AI powered transcriptor app to generate accurate, well formatted text from spoken audio.

Phase-2: Requirement Analysis

Objective:

1.Develop an automated system that converts spoken language from an audio file into text.

2.Detects the language of the transcribed text.

3.summarize the translated text using Open AI GPT.

Key Points:

1. **Technical Requirements:**

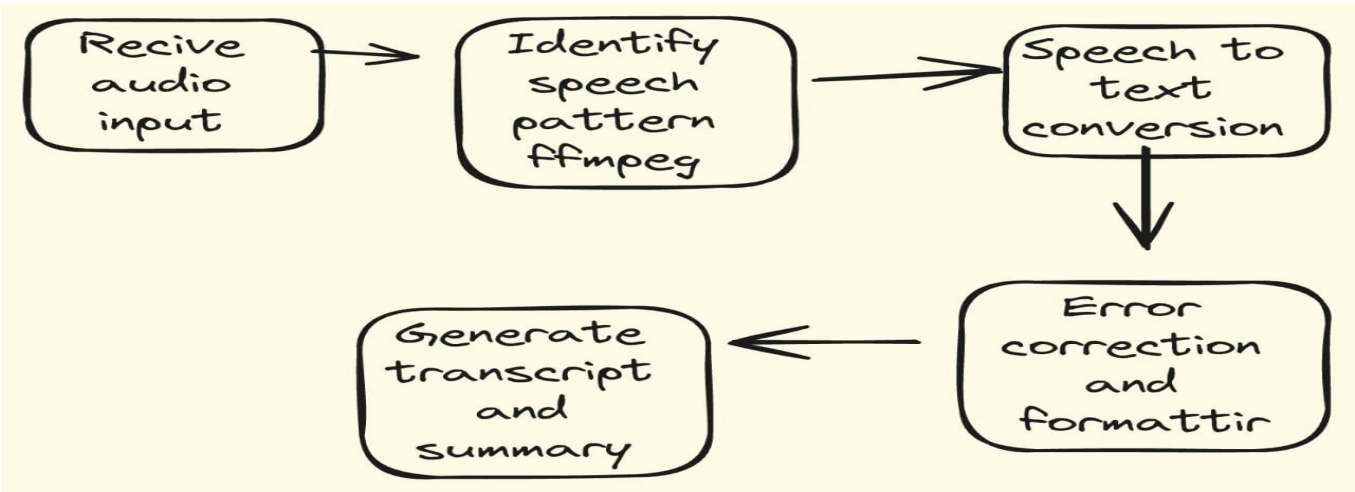
    - Programming Language: **Python.**
    - Backend: **python,ffmpeg,torch,whisper,langid,transformers.**
    - Frontend: **Stream lit Web Framework**
    - Database: **Not required initially .**
2. **Functional Requirements:**

    - Accepts an audio file(e.g.,MP3,WAV etc..)
    - Detects the spoken language.
    - Displays the original transcription .
    - Summarize the text.
3. **Constraints & Challenges:**
    - Internet requirements.
    - Providing a **smooth UI experience** with Streamlit.
    - OpenAI GPT_4 API charges based on usage
    - Processing large audio files.

Phase-3: Project Design

Objective:

Develop the architecture and user flow of the application.



Key Points:

1. **System Architecture:**

    - User uploads an audio file
    - Query is processed using open ai whisper
    - AI model fetches and processes the data.
    - The frontend displays **transcript,detected language,summary**
2. **User Flow:**

    - Step 1: User upload an audio file
    - Step 2: The backend **transcripts speech to text**
    - Step 3: Some other models help finding the errors in sentences and fix them
    - Step 4: Result gives the transcript and key takeaways from given audiofile
3. **UI/UX Considerations:**

    - **Minimalist, user-friendly interface** for seamless navigation.

Phase-4: Project Planning (Agile Methodologies)

Objective:

Break down development tasks for efficient completion.

Sprint	Task	Priority	Duration	Deadline	Assigned To	Dependencies	Expected Outcome
Sprint 1	Environment Setup	🔴 High	6 hours (Day 1)	End of Day 1	Sushmitha Tejaswi	Python, Streamlit setup	established & working
Sprint 1	Frontend UI Development	🟡 Medium	2 hours (Day 1)	End of Day 1	Rishika	response format finalized	Basic UI with input fields
Sprint 2	Audio file upload	🔴 High	3 hours (Day 2)	Mid-Day 2	Tejaswi Rishika	UI elements ready	Search functionality with filters
Sprint 2	Error Handling & Debugging	🔴 High	5 hours (Day 2)	End of the day 2	Members 3	UI inputs	Improved stability
Sprint 3	Testing & UI Enhancements	🟡 Medium	2 hours (Day 2)	Mid-Day 2	Tejaswi Sushmitha	UI layout completed	Responsive UI, better user experience
Sprint 3	Final Presentation & Deployment	🟢 Low	1 hour (Day 2)	End of Day 2	Entire Team	Working prototype	Demo-ready project

Sprint Planning with Priorities

Sprint 1 – Setup & Integration (Day 1)

(🔴 **High Priority)** Set up the **environment** & install dependencies.
(🟡 **Medium Priority)** Build a **basic UI with input fields**.

Sprint 2 – Core Features & Debugging (Day 2)

(🔴 **High Priority)** Implement **search of audio file uploading**.
(🔴 **High Priority)** Debug API issues & handle **errors in queries**.

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

(🟡 **Medium Priority)** refine UI, & fix UI bugs.
(🟢 **Low Priority)** Final **demo preparation & deployment**.

Phase-5: Project Development

Objective:

Implement core features of the Audio transcriber App.

Key Points:

1. **Technology Stack Used:**
    - **Frontend:** Streamlit
    - **Backend:** ffmpeg,python,torch.
    - **Programming Language:** Python
2. **Development Process:**
    - Setup ffmpeg/transformers/langid.
    - Implement key features : speech recognition ,language detection & translation.
    - Summarization.
3. **Challenges & Fixes:**
    - **Challenge:** low quality or noisy audio formats.
      Fix: use noise reduction techniques before transcription.
    - **Challenge:** large audio files may cause performance bottlenecks.
      Fix: limit file size and implement chunk-based processing for long recordings.

Phase-6: Functional & Performance Testing

Objective:

Ensure that the AutoSage App works as expected.

Test Case ID	Category	Test Scenario	Expected Outcome	Status	Tester
TC-001	Functional Testing	A 4 seconds audio file(EN) is uploaded	To Give correct transcription	✅ Passed	Rishika
TC-002	Functional Testing	A 10 seconds audio file(EN) of different accent is uploaded	To give correct transcription and summary	✅ Passed	Sushmitha
TC-003	Performance Testing	Large audio data files	To give response quickly	⚠ Needs Optimization	Tejaswi
TC-004	Bug Fixes & Improvements	A 10 seconds audio file(mix lang) is uploaded	To give correct transcription with summary	⚠️Needs optimization	Sushmitha

TC-005	Final Validation	Ensure UI is responsive across devices.	UI should work on mobile & desktop.	✅Passed	Tejaswi
TC-006	Deployment Testing	Host the app using Streamlit Sharing	App should be accessible online.	🚀 Deployed	Entire team

---

Final Submission

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