
Project Title:

Audio transcription app using OpenAi Whisper

Team Name:

Hackoholics

Team Members:

- A. Praharika
- CH. Harshini
- A.Sahitya
- Brunda Siddenki

Phase-1: Brainstorming & Ideation

Objective:

Developing an app which has an audio transcription tool that utilizes Whisper, a powerful AI model for speech recognition. It allows users to upload audio files (MP3 or WAV format) and transcribes the speech into text.

Key Points:

1. **Problem Statement:**

- Many users struggle with converting spoken content into written text efficiently and accurately.
- Manual transcription is time-consuming and prone to errors.

.

2. **Proposed Solution:**

- An AI-powered application using OpenAI Whisper to transcribe audio into text with high accuracy.
- The app will support multiple languages and dialects for broader usability.

3. **Target Users:**

- Content creators, journalists, and students needing transcription services.
- Businesses looking for automated meeting transcription.
- Researchers working with audio data.

4. **Expected Outcome:**

- A fully functional audio transcription application that allows users to upload MP3 or WAV files and obtain accurate text transcriptions.

Phase-2: Requirement Analysis

Objective:

Define the technical and functional requirements for the Audio Transcription App.

Key Points:

1. **Technical Requirements:**

- Programming Language: Python
- Backend: OpenAI Whisper API
- Frontend: Streamlit Web Framework
- Database: Not required initially (API-based processing)

2. **Functional Requirements:**

- Ability to upload MP3 and WAV files.
- Real-time or near real-time transcription processing.
- Support for multiple languages and accents.

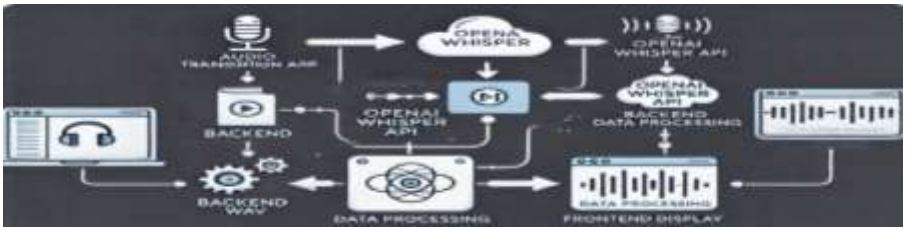
3. **Constraints & Challenges:**

- Handling large audio file processing efficiently.
- Ensuring accurate transcription across different speech patterns and noise levels.

Phase-3: Project Design

Objective:

Develop the architecture and user flow of the application.



Key Points:

1. **System Architecture:**

- User uploads an audio file via the UI.
- File is processed using OpenAI Whisper API.
- AI model transcribes speech into text.
- Frontend displays the transcribed text.

2. **User Flow:**

- Step 1: User uploads an audio file.
- Step 2: The backend processes the file using Whisper API.
- Step 3: Transcribed text is displayed for download or editing.

3. **UI/UX Considerations:**

- Minimalist, user-friendly interface for seamless navigation.
- Dark & light mode for better user experience.
- Downloadable transcript option.

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 & API Integration	□ High	6 hours (Day 1)	End of Day 1	Praharika	OpenAI API Key, Python, Streamlit setup	API connection established & working
Sprint 1	Frontend UI Development	□ Medium	2 hours (Day 1)	End of Day 1	Harshini	Basic UI with file upload field	Basic UI with input fields
Sprint 2	Vehicle Search & Comparison	□ High	3 hours (Day 2)	Mid-Day 2	Sahitya	API response, UI elements ready	Transcription Functionality Implemented
Sprint 2	Error Handling & Debugging	□ High	1.5 hours (Day 2)	Mid-Day 2	Brunda& Praharika	API logs, UI inputs	Improved API stability
Sprint 3	Testing & UI Enhancements	□ Medium	1.5 hours (Day 2)	Mid-Day 2	Harshini& Sahitya	API response, 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.
(□ High Priority) Integrate **OpenAI Whisper API**.
(□ Medium Priority) Build a **basic UI with file upload fields**.

Sprint 2 – Core Features & Debugging (Day 2)

(□ High Priority) Implement **transcription functionalities**.
(□ High Priority) Debug API issues & handle **errors in queries**.

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

(□ Medium Priority) Test API responses, refine UI, & fix UI bugs.
(□ Low Priority) Final **demo preparation & deployment**.

Phase-5: Project Development

Objective:

Implement core features of the Audio Transcription App.

Key Points:

1. **Technology Stack Used:**

- Frontend: Streamlit
- Backend: OpenAI Whisper API
- Programming Language: Python

2. **Development Process:**

- Implement API key authentication and Whisper API integration.
- Develop audio processing and transcription logic.
- Optimize response times for large audio files.

3. **Challenges & Fixes:**

- Challenge: Large file processing delays.
- Fix: Implement file size limits and chunked processing.
- Challenge: Background noise affecting transcription accuracy.
- Fix: Pre-processing noise reduction techniques.

Phase-6: Functional & Performance Testing

Objective:

Ensure that the Audio Transcription App works as expected.

Test Case ID	Category	Test Scenario	Expected Outcome	Status	Tester
TC-001	Functional Testing	Upload clear MP3 audio file	Accurate transcription should be displayed.	☐ Passed	Tester 1
TC-002	Functional Testing	Upload noisy WAV audio file	Transcription with minor errors.	Needs Optimization	Tester 2
TC-003	Performance Testing	Process large audio file (1 hour+)	Response time under 30 seconds.	Failed - Needs optimization	Tester 3
TC-004	Bug Fixes & Improvements	Fixed API timeout errors.	App should handle long processing times.	Fixed	Developer
TC-005	Final Validation	Ensure UI is responsive across devices.	UI should work on mobile & desktop.	Passed	Tester 2
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**