***PodscriptQA Web Appication***

***SOFTWARE PROJECT FINAL REPORT***

Team members: Pradyoth Reddy Gampala, 2863351

Saritha Reddy Loka, 2862662

Sponsor: Sai Vivek Reddy Yedulla, Ph. No. 703-722-4261

Email: y.saivivekreddy@gmail.com

Date 12/15/2023

**1. Introduction**

**1.1. Purpose and Scope**

PodScriptQA is an innovative application designed to revolutionize how users interact with podcast content. At its core, this application aims to make podcasts more accessible and interactive by allowing users to upload podcast audio and then pose specific queries related to that podcast in a chat format. The primary purpose of PodScriptQA is to enhance accessibility, ensuring that users can easily extract specific information from podcasts without needing to listen to the entire episode. This feature is particularly beneficial for saving time, as it provides direct answers to user queries without manual searching through the audio content. Moreover, it aims to boost content engagement by simplifying the process of interacting with and extracting information from podcasts. The scope of PodScriptQA encompasses robust audio processing capabilities, capable of interpreting and transcribing spoken content. It includes a sophisticated query handling mechanism that analyzes the processed audio to accurately respond to user inquiries, drawing insights directly from the podcast's content. Additionally, the project focuses on a user-friendly interface for seamless podcast uploads and query submissions, ensuring broad compatibility with various podcast formats and lengths. PodScriptQA is not just a tool for regular podcast listeners; it's also an invaluable resource for researchers and anyone interested in efficient information retrieval from audio content, marking a significant advancement in the way users engage with podcasts.

**1.2. Product Features**

* User can upload a desired podcast/audio clip
* The application converts uploaded audio to text
* User can upload queries related to the podcast/audio
* The application will process the questions and generate accurate answers based on the text transcribed from the podcast/audio

**2. Requirement Specifications**

Stakeholders for this application are:

* Developers
* Sponsor

**1. Functional Requirements**

1.1 Audio Upload and Processing

- Users must be able to upload podcast audio files in various formats, including MP3 and WAV.

- Integration with Google Cloud Services for audio transcription, ensuring high accuracy and efficiency in converting spoken words to text.

1.2 Query Handling and Response Generation

- A user-friendly chat interface for submitting queries related to the podcast content.

- Integration with OpenAI's GPT model for interpreting and answering user queries, leveraging its advanced natural language understanding capabilities.

1.3 User Interface

- An intuitive and responsive interface for easy uploading, querying, and navigation.

- Clear display of transcribed content and corresponding answers to queries.

1.4 Data Privacy and Security

- Compliance with data protection laws, ensuring user data confidentiality and security.

**2. Non-Functional Requirements**

2.1 Performance

- Fast and reliable performance in audio transcription and query responses, with processing time proportionate to podcast length.

2.2 Scalability

- The application must efficiently handle a high volume of users and simultaneous queries.

2.3 Usability

- The application should be easily navigable and accessible to users with varying technical backgrounds.

2.4 Compatibility

- Cross-platform compatibility to support diverse user devices and operating systems.

2.5 Reliability

- High system uptime with robust error handling and minimal service disruptions.

2.6 Maintainability

- Easy maintenance and upgradability, considering future advancements in API integrations.

**3. System Constraints**

3.1 Technical Constraints

- Dependence on external APIs (Google Cloud Services and OpenAI) for core functionalities.

- Requirement for stable internet connectivity for API communications.

3.2 Budgetary Constraints

- Budget allocation must account for ongoing costs associated with API usage and cloud services.

3.3 Legal and Compliance Constraints

- Adherence to the terms of service and usage policies of the integrated APIs.

- Compliance with global data protection regulations in all operational regions.

**4. User Documentation and Training**

4.1 Documentation

- Comprehensive guides on system setup, API integration, and general usage.

**5. Acceptance Criteria**

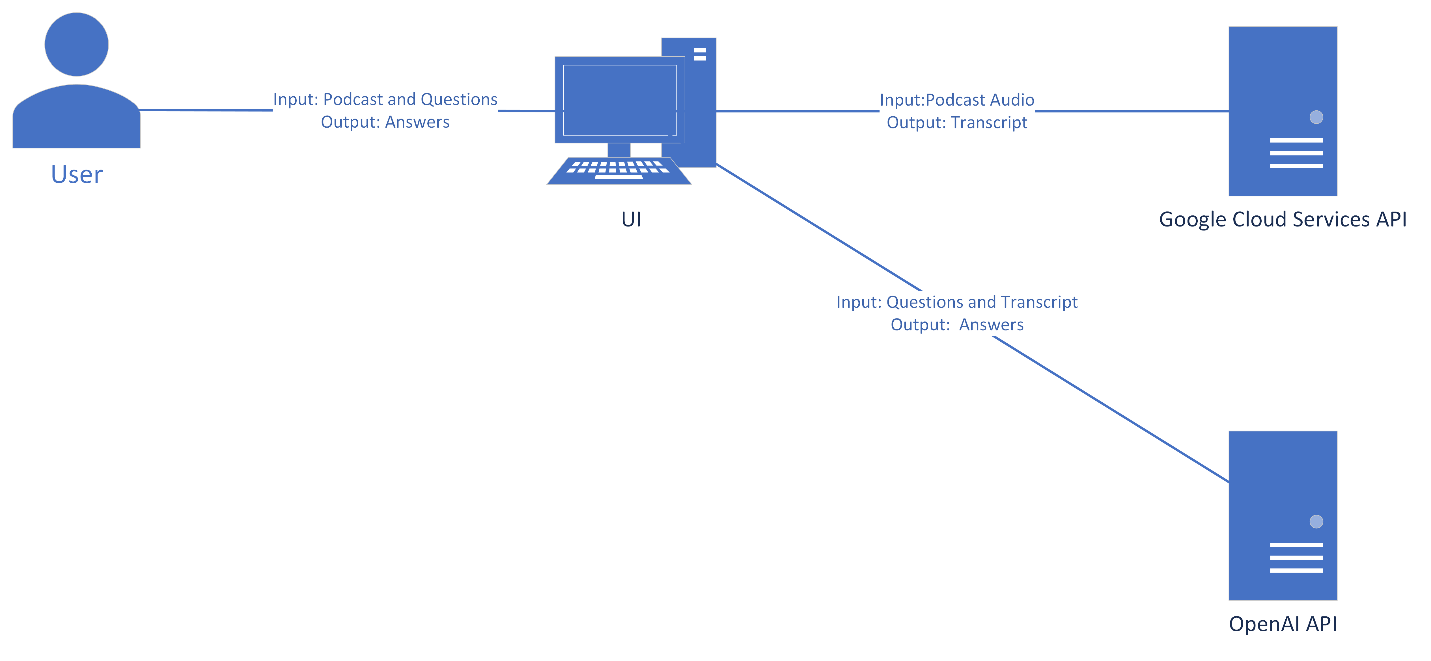
5.1 Testing and Validation

- Thorough testing of API integrations and overall system functionality.

- User testing phase to gather feedback and ensure the system meets user expectations and requirements.

By incorporating specific details about the use of Google Cloud Services for transcription and OpenAI for the QA model, this revised specification outlines a clear and detailed framework for the development of podscriptQA, ensuring it leverages these advanced technologies effectively for its intended purpose.

**3. Architecture**



**Data Flow:**

1. **Audio Upload**:
   * A **User** uploads a podcast audio file to the **PodscriptQA Website**. This audio needs to be in mono-channel format, as required by GCS.
2. **Audio Segmentation**:
   * The **Website** segments the podcast audio into one-minute chunks, complying with GCS's processing limitation.
3. **Transcription Process**:
   * Each audio segment is sent sequentially to **Google Cloud Services** for transcription.
   * **Google Cloud Services** transcribes each segment and returns the text to the **Website**.
4. **Transcript Concatenation**:
   * The **Website** receives transcriptions in parts and concatenates them to form a complete transcript.
   * This complete transcript is then stored as the context for the podcast on the **Website's Server**.
5. **User Queries**:
   * The **User** inputs questions into the **Chat Interface** on the **Website**.
   * The **Chat Interface** resembles ChatGPT’s interface, which retains the history of previous queries and answers for the session.
6. **Contextual Question Answering**:
   * When a query is entered, the **Website** sends both the complete transcript (as context) and the new question to the **OpenAI API**.
7. **Answer Retrieval**:
   * The **OpenAI API** processes the context and the question to generate an appropriate answer.
   * The answer is sent back from the **OpenAI API** to the **PodscriptQA Website**.
8. **Displaying Answers**:
   * The **Website** presents the answer from OpenAI within the **Chat Interface** to the **User**.
   * The user can continue to ask more questions, with each interaction being logged in the chat history for continuity.

**4. Tools and Technologies**

**Development Environment**

*Visual Studio Code (VSCode):* The primary integrated development environment (IDE) used for writing and editing the codebase. It provides comprehensive support for Python, version control integration, and debugging tools.

**Programming Language**

*Python:* The main programming language used to develop the backend logic of the PodScriptQA application due to its robust libraries and frameworks for web development and API integration.

**APIs**

*Google Cloud Services (GCS) API:* Utilized for the transcription service that converts podcast audio content into text format. This API is particularly chosen for its advanced speech recognition capabilities.

*OpenAI API:* Employed to power the question-answering model that interprets user queries and provides relevant responses based on the context provided by the transcribed podcast text.

**Audio Processing Library**

*PyDub:* A Python library used to manipulate audio. In this project, PyDub is instrumental for segmenting the podcast audio into one-minute chunks to comply with the transcription limitations of GCS.

**Web Framework**

*Flask:* A lightweight web framework for Python that is used to create the web server and RESTful API endpoints. Flask facilitates the integration of the frontend user interface with the backend services.

**5. Limitations of the application:**

1. Larger audio files take longer duration of transcription because of the API limits.
2. Larger transcripts also mean longer processing for QA model as well.

**6. Future Scope:**

PodScriptQA, as it stands, is an innovative platform that transforms the way users engage with podcast content. Looking ahead, the project can be expanded in several directions:

1. **Multilingual Support**: To cater to a global audience, future versions could include transcription and query answering in multiple languages.
2. **Real-Time Interaction**: Enhancing the platform to provide real-time transcription and query responses would greatly improve user engagement.
3. **Deepened Contextual Understanding**: Implementing more advanced natural language processing models could enable the system to understand complex queries and deliver more precise answers.
4. **User Customization**: Introducing personalized experiences based on individual user history and preferences could make PodScriptQA a more user-centric platform.
5. **Podcast Platform Integration**: Seamless integration with existing podcast platforms could simplify the user experience, allowing listeners to utilize PodScriptQA's functionality without leaving their preferred platform.
6. **Offline Capabilities**: Developing the ability to transcribe and answer queries offline would make the service more versatile and widely usable.
7. **Content Analytics**: Offering content creators insights into the questions their listeners are asking could inform future podcast topics and strategies.

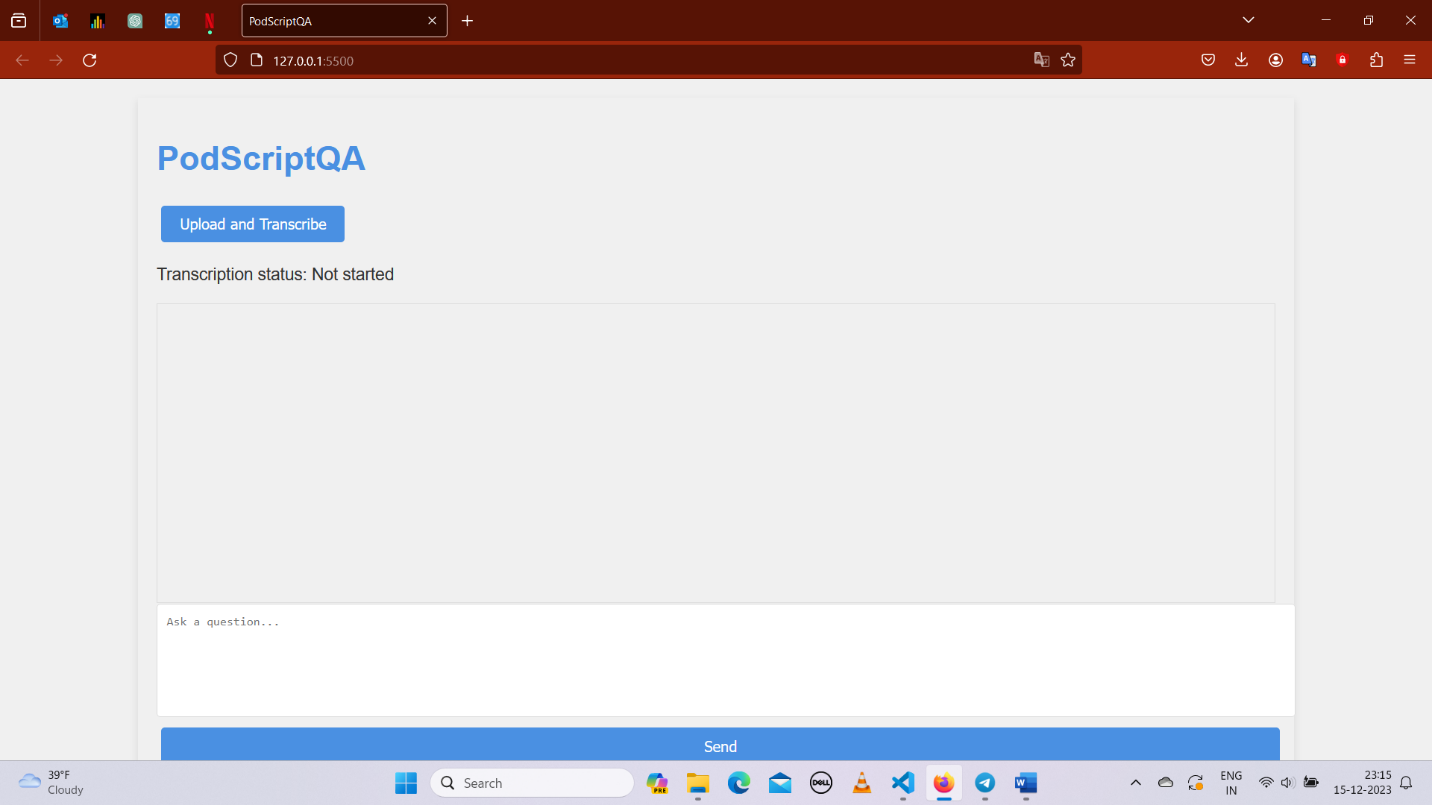
**7. Conclusion:**

PodScriptQA marks a notable advancement in podcast technology by providing a unique service that makes audio content searchable and interactive. Utilizing tools like VSCode, Python, PyDub, and Flask, along with powerful APIs from Google Cloud Services and OpenAI, the project has established a scalable and maintainable solution that greatly benefits its users.

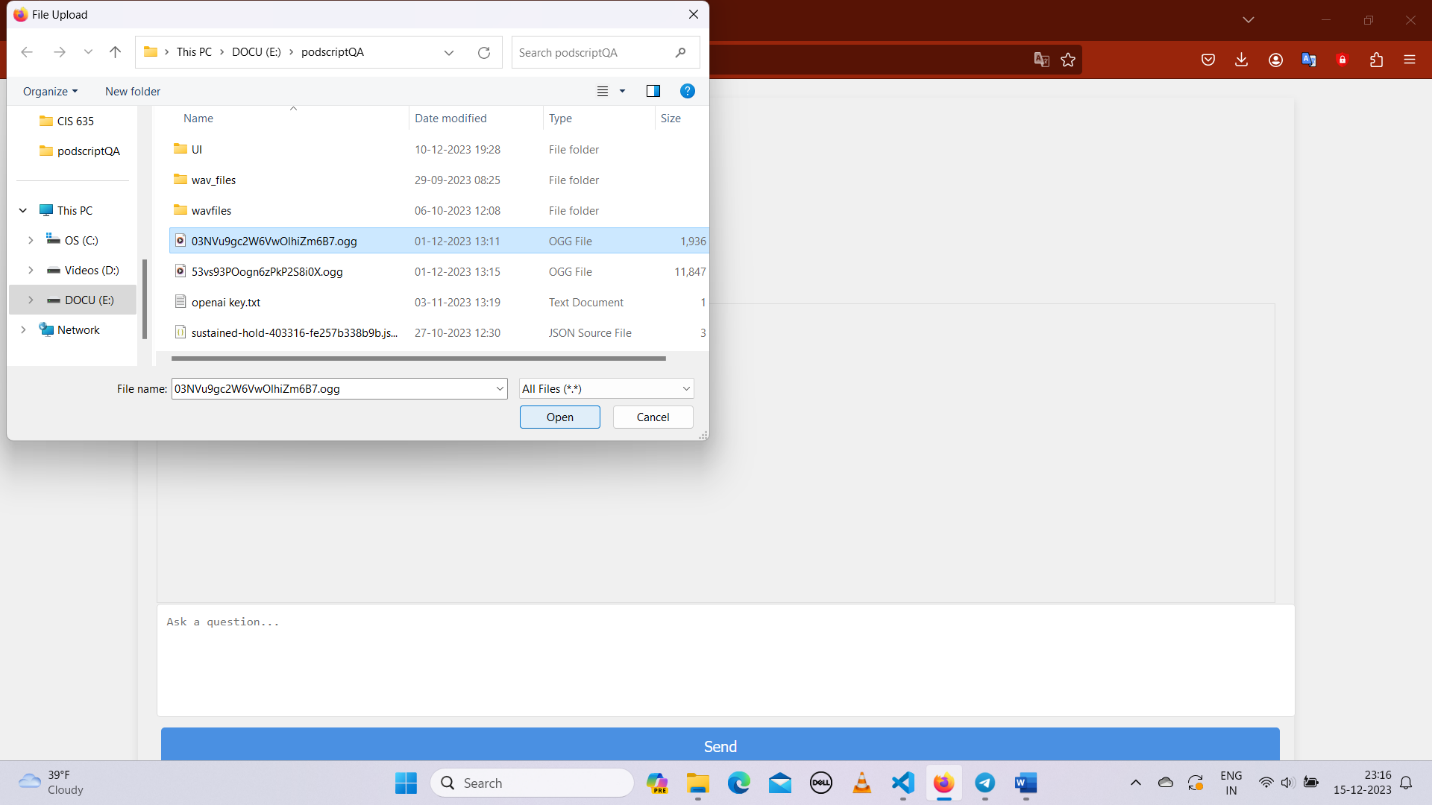
The application’s potential for growth and its capacity to adapt to the evolving landscape of digital media consumption position PodScriptQA as a vital tool for podcast listeners and creators. The current success sets the stage for future enhancements, which will continue to be informed by user feedback and technological progress.

In conclusion, PodScriptQA's development embodies more than a technological triumph; it is a step toward a future where media is not only consumed but interacted with in a meaningful way. The project team is committed to pushing the boundaries of this innovative service, striving to enrich the user experience and expand the platform's capabilities.

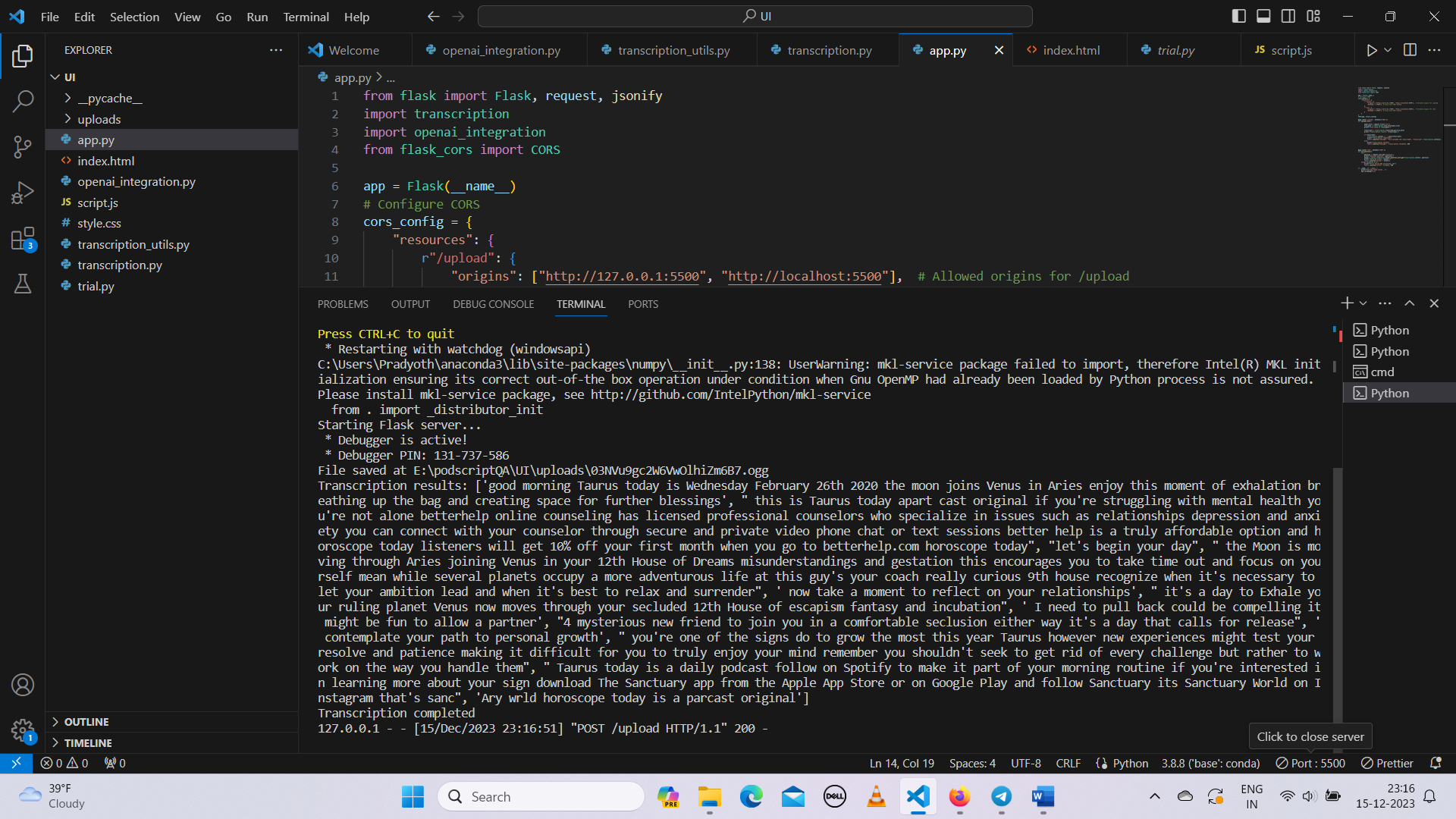
**8. Screenshots:**

**1. User Interface**  


**2.** **Uploading podcast audio file**



**3. Transcription complete**



**4. User asks questions and the application returns responses**

