# Documentation for Case-Chat-Gpt **Project**

## Introduction

This documentation provides an overview of a Flask application designed to process uploaded Excel files, extract data, generate responses using OpenAI's GPT-4 model, and create Word documents based on the extracted data and generated responses.

## Application Components

The Flask application consists of the following components:

1. app.py: This Python script contains the main Flask application logic, including routes, database operations, file processing functions, and integration with OpenAI's GPT-4 model.
2. index.html: This HTML file defines the user interface for the web application, including forms for uploading files, displaying uploaded files, and handling user interactions.
3. main.css: This CSS file contains styles for the user interface elements defined in index.html.
4. main.js: This JavaScript file contains client-side scripting for handling form submissions, displaying messages, and interacting with the backend server.

## Functionality

The Flask application offers the following functionality:

1. Upload File: Users can upload Excel files (.xlsx) containing case data.
2. Prompt Generation: Users can enter prompts for generating responses using the GPT-4 model.
3. File Processing: Upon file upload, the application extracts data from the Excel file and generates responses for each case using the provided prompt and the GPT-4 model.
4. Document Generation: The application creates Word documents (.docx) containing summaries and responses for each case.
5. Status Tracking: The application tracks the status of uploaded files (In Progress or Completed) and updates the status accordingly.
6. Download and Delete: Users can download generated documents and delete uploaded files.

## Installation and Setup

To run the Flask application locally, follow these steps:

1. Install the required Python packages listed in the requirements.txt file using pip install -r requirements.txt.
2. Set up environment variables:
   * Set OPENAI\_API\_KEY with your OpenAI API key.
   * Ensure proper configuration of SQLite database URI (SQLALCHEMY\_DATABASE\_URI) in app.py.
3. Run the Flask application using python app.py.
4. Access the application in a web browser at http://localhost:8080.

## Usage

1. Upload File: Click on the "Choose File" button to select an Excel file (.xlsx) containing case data.
2. Enter Prompt: Enter a prompt in the provided text area to generate responses.
3. Submit: Click on the "Continue And Submit" button to process the uploaded file and generate documents.
4. Track Status: View the status of uploaded files in the table. Files are marked as "In Progress" during processing and "Completed" once processing is finished.
5. Download and Delete: Click on the "Download" icon to download generated documents. To delete an uploaded file, click on the "Delete" link.

## Notes:-

* Word documents are saved in the static/docx directory.
* Uploaded files are stored in the upload\_file\_/ directory.
* Python3 is required for run the project.
* Use the following command to launch the project directly in the terminal with run\_project.sh: ./run\_project.sh
* open terminal in the file location where project directory present.