**Week 6 Report Summer Internship**

Name: Mohamed Elfaki

University: Eastern Mediterranean University (Famagusta, Cyprus)

Department: Software Engineering

Training Organization: Cloud Solutions for IT & Communications Co, AI department

**Outline:**

1. Introduction
2. Book Reservation System (CRUD App) Implementation
3. Simple AI Chatbot Implementation
4. MSA-AI System
5. References

**1.Introduction**

In this week I’ll be focusing on developing 3 things:

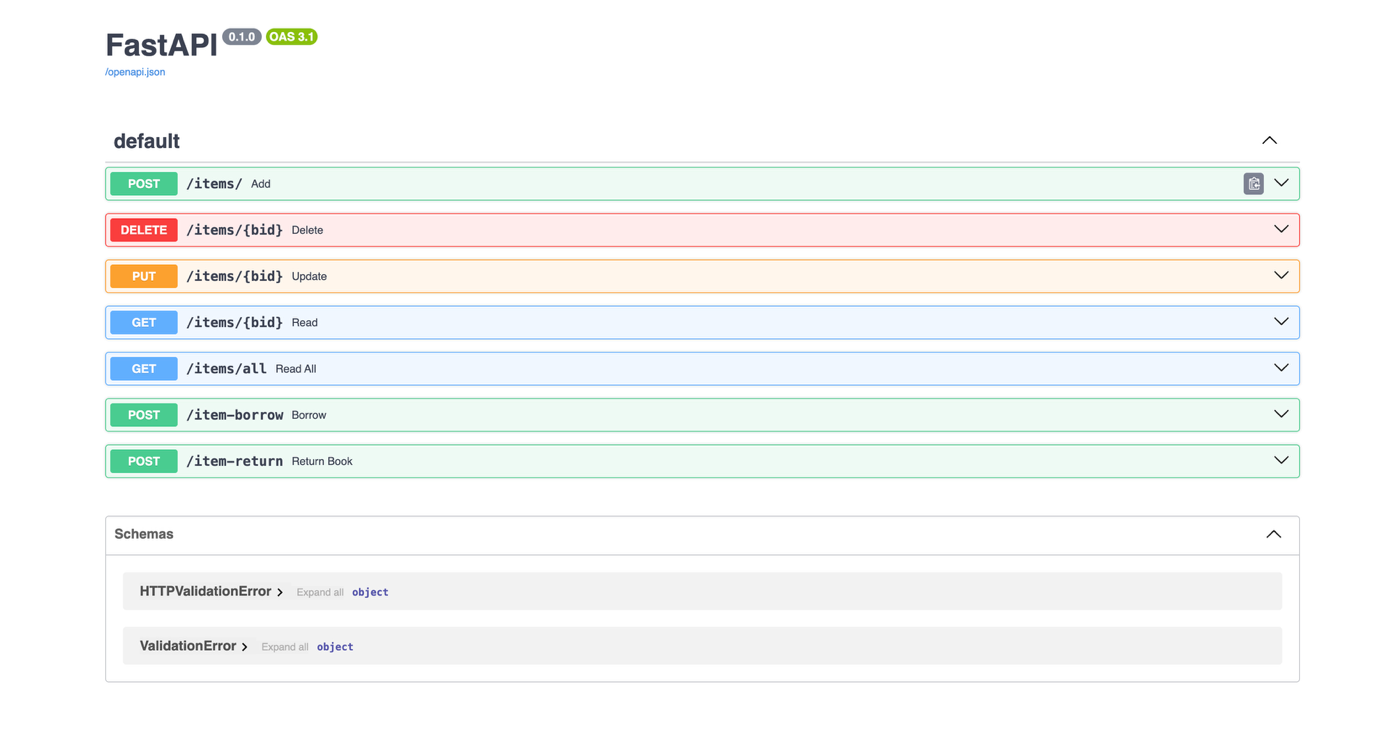
1. A mini book reservation system using FastAPI and Python.
2. A simple AI chatbot that can respond to prompts on any database the users choose to upload.
3. My MSA-AI (Medical Support Authorization) system, which is a system that aims to utilize AI to support insurance companies in making authorization decisions. These decisions are mainly the approval/denial of medical procedure requests based on patient history and other data.

**2. Book Reservation System (Crud App)**

This week I started working on a book reservation system, the first step in this project was to get a suitable database. I searched Kaggle for a .csv file for this purpose and found a suitable dataset; I took some lines of code from stack overflow which serve to switch a .csv file to an SQLite database.

I then started the skeleton of the application with a main.py file with the main FastAPI frontpage, and another file database.py containing all the database related functions for the file such as create, delete, read, read all, update and additional borrow book and return book function.

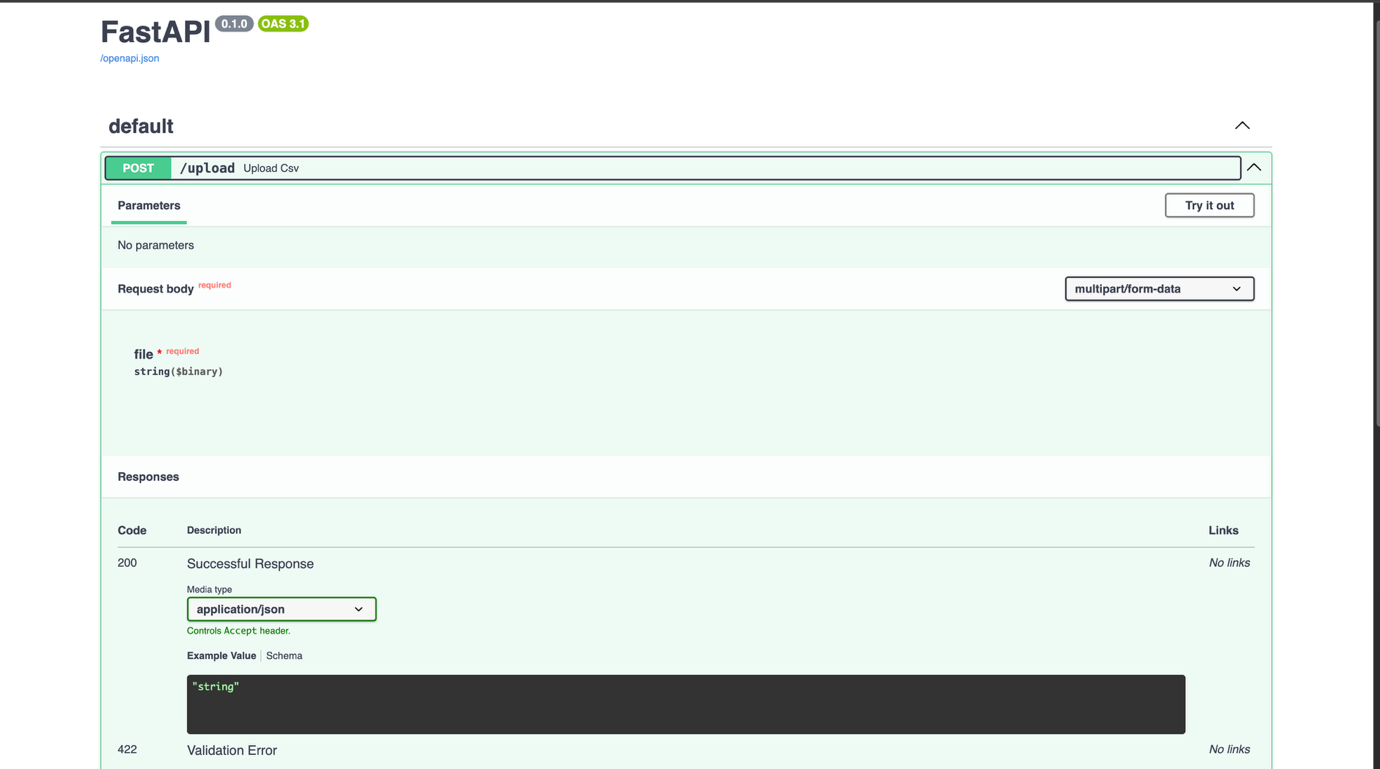
All FastAPI functions in the frontpage call the functions from the database.py file and return the message as a JSON file to show success or failure and results of the function.



**3. Simple AI Chatbot**

The next project I developed during this week is a simple AI chatbot. For this project, I used FastAPI as a lightweight frontend for the Python-based code. I split the code into two main python files: main.py and ai\_engine.py. main.py handles the FastAPI initialization logic. Ai\_engine.py loads the database and sends it to the AI, giving it instructions to only answer based on the database’s information.

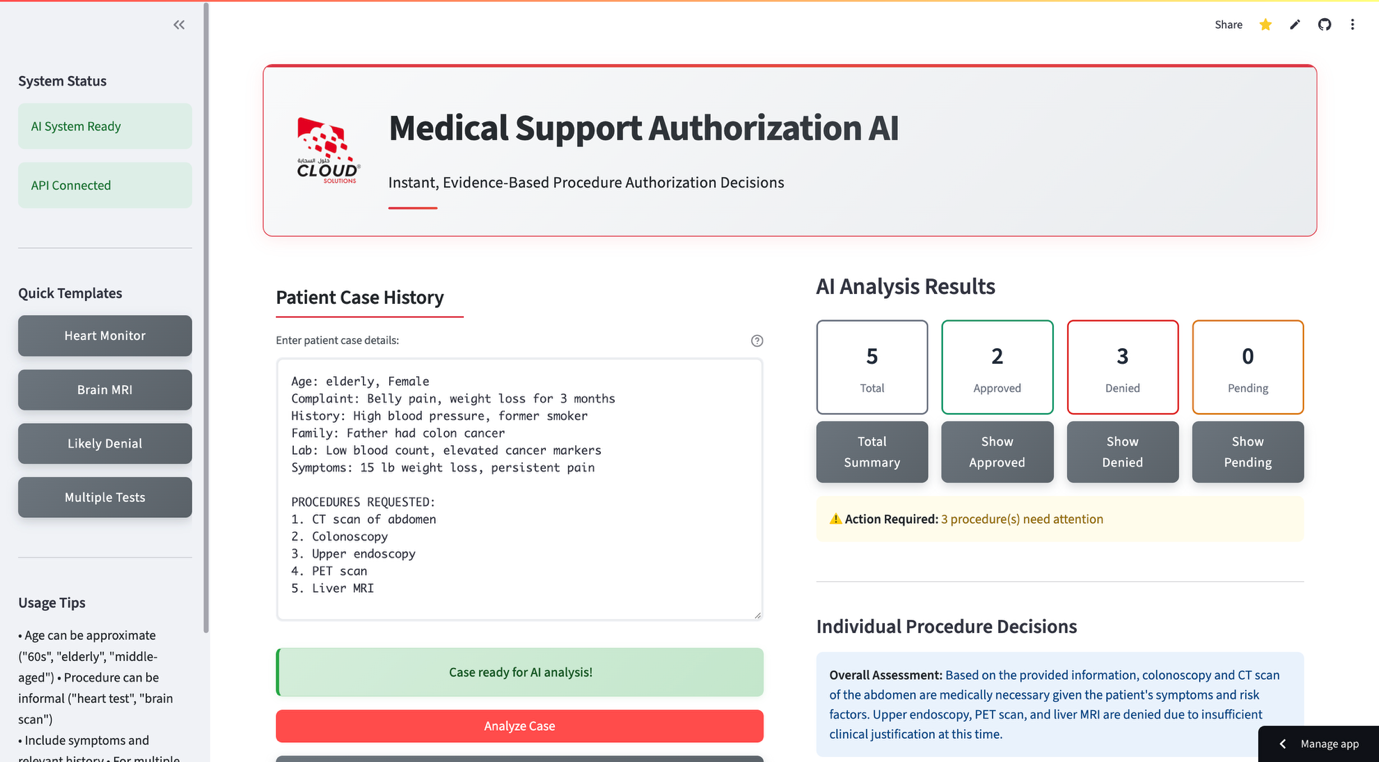
A screenshot of a computer

AI-generated content may be incorrect.

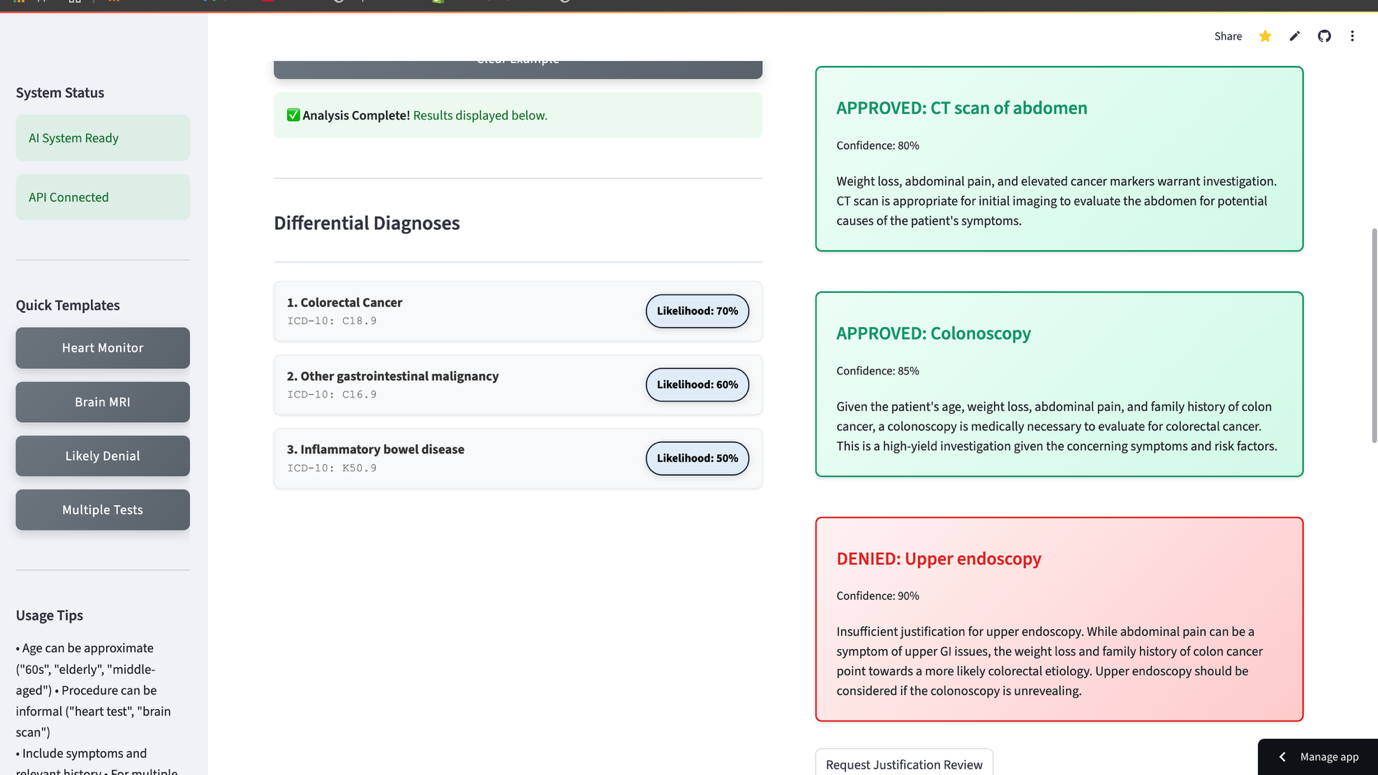
**4. MSA-AI System**

A project I have been working on for a while with my fellow intern is MSA-AI. It is a Medical Support Authorization system, designed to assist insurance companies in making medical decisions such as approving or denying medical procedures for patients. These decisions are AI-generated based on the patient history. The LLM we used for this project is Gemini 1.5 Flash, due to its swiftness and large number of tokens.

For the frontend of this system, we used the Streamlit library, and Python for the backend. You may choose to provide additional justification for denied procedures. The frontend of this system is as follows:



The GUI is split into 2 columns in addition to a sidebar. The left column allows you to enter the patient case history, which will be sent to Gemini 1.5 Flash for decisions. The right column shows the output of the AI and it shows all decisions regarding the procedures.



As you scroll down, ICD codes for the most likely diagnoses are shown. On the right you can see the many procedure decisions along with sufficient reasoning, you may also request justification review for denied procedures.

**5. References**

* <https://fastapi.tiangolo.com/tutorial/>
* <https://medium.com/@aberrospic1/crud-operations-with-fastapi-c2de026e5862>
* <https://docs.python.org/3/library/sqlite3.html>
* <https://console.groq.com/docs/quickstart>
* <https://docs.streamlit.io/get-started/tutorials/create-an-app>
* <https://ai.google.dev/gemini-api/docs/quickstart>