**Summer Training Report  
Week 2**

**LangChain, Agentic, and Augmented AIs, Docker, and Data conversion practice**

**Submitted by**

Name: Mahmoud Mustafa Oushi

University ID: 4312239

University: University of Prince Muqrin

Department: College of Computer Science and Cyber Security

Advisor: Dr.Ahmed Elhayek

**Training Organization**

Hulool Al Sahabah for IT & Communications Co

AI Department

**Report Topics – Week 2**

* Converting CSV Data to LLM-Compatible Formats (Practical)
* AI Agents and Agentic AI
* Augmented AI
* Docker
* FastAPI
* PostgreSQL
* Application Practice Results

**Abstract**

For week two of the summer training, the tasks were to put into practice what was researched in the previous week. How to convert the data from SQL into LLM formats, using libraries like Pandas to create natural human language settings as well as JSON format style. As for the research side, this week will go into LangChain which is a way to directly integrate the CSV with natural language without the need for conversion. You can query in human language and still get your answer. Also trying out the FLAN-T5 model for JSON summarization. Then the new topics introduced are Agentic AIs, how to put into work an agent which will have their own agents to recruit, then to take a look into Augmented AI and the Docker deployment system.

**1. Introduction**

So, for the second week. The tasks are more practical than the last. This time it is going to be more or less project oriented, using FastAPI and Postgres to create a simple project that take the name of the user and display a message to them. But before that, back to the LLM format conversion via json. I managed to do so, then start a model to do some natural language applications to it. That and more exciting things like LangChain, as well as a deeper look into Claude Augmented AI. And more Insha’Allah.

**3. Converting CSV Data to LLM-Compatible Formats (Practical)**

To be able to go through with the practice, I searched on Kaggle for a suitable dataset with a matching theme with what Cloud solutions is, I did managed to find a candidate.

**3.1. HospInfo data from Kaggle**

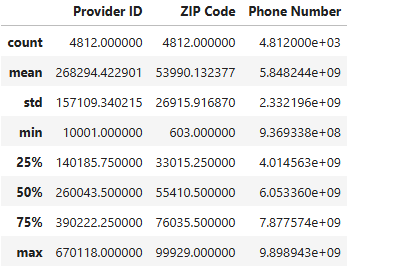
Content:

This file contains general information about all hospitals that have been registered with Medicare, including their addresses, type of hospital, and ownership structure. It also contains information about the quality of each hospital, in the form of an overall rating (1-5, where 5 is the best possible rating & 1 is the worst), and whether the hospital scored above, same as, or below the national average for a variety of measures.

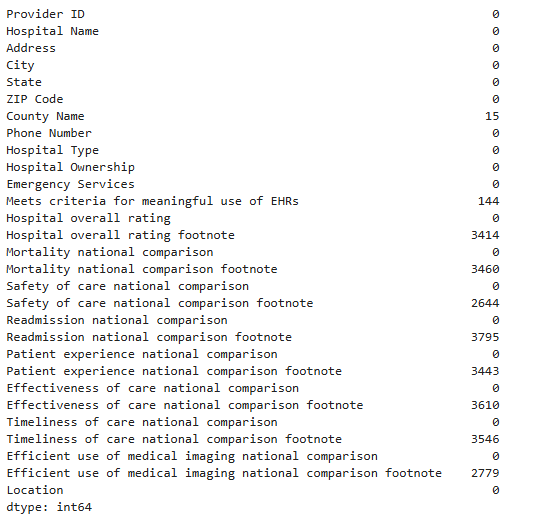
**3.2. EDA**

Performed the standard EDA processes (columns names, types, indo, statistical analysis, sum of all missing or NaN values, etc.)

* dtypes: bool(1), int64(3), object(25)
* Description



* Sum of null

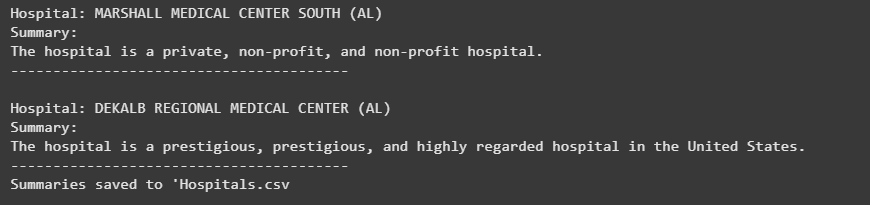


**3.3. Pandas JSON Conversion**

The process of using the Pandas Library was straight forward, after reading the file and opening it in the writing mode, I used a function called (json.dump) which **serializes** a python object like a dictionary or list into **JSON format,** and writes it directly to a file. Then it is a matter of saving the file and that is about it.

**3.4. Failed APIs and FLAN-T5**

Afterwards I tried to experiment with some LLM APIs like Llama3, Mistral-7B, and some public google models all failed due to an empty response. Tried multi network choices, and created multiple tokens with playing with modes and privileges. Nothing worked. While conducting some testing, the webpage when calling the APIs turned to be blocked for some reason.

When that did not work, I used a FLAN-T5 small model which I previously used in the GEN AI using LLMs course from DeepLearning.AI introduced and explained in the last report. I deployed the model and was able to read the created json from the pandas experiment. And even generate a zero-shot inference summary of the file. Below are some examples.  
  


Now the initial results weren’t….great which is expected using the method I used. But with some PEFT-ing and maybe few-shot inferences. This does have potential.

**4. AI Agents and Agentic AI**

Agentic AI is a system that acts as an autonomous agents that perceive the environment, make decisions and take actions to achieve their specific goal without constant human surveillance and intervention. They can be seen as goal oriented which they need to achieve in dynamic uncertain situations and environments.

**4.1 Characteristics:**

* **Autonomy**

Meaning the AI can operate on its own in order to make actions and take decisions based on it surroundings and input.

* **Goal Oriented**

Designed to achieve a specific task

* **Planning and Reasoning**

Able to plan up to multiple steps ahead, as well as evaluating the alternative to each to ensure the optimal way to reach the goal is taken

* **Memory and Context**

The ability to remember past moves and decisions to learn from and gain experiences

Some examples of such models could be Robotic models integrated with an AI agent to logic like an LLM model, it can handle tasks like customer service and directional guidance. Another example which most of us use in our daily lives are virtual assistants, like Siri and the Google Assistant. With tasks like planning the schedule for the day, setting a reminder based on availability, booking an appointment at a place, or sifting through contacts to make a call. All these tasks needs some logic and decision making for them to operate perfectly, thus making them a viable to try yourself and see the full capabilities of such models.

**4.2. Are AI agents the same as Agentic AIs?**

The two terms may seem like two sides of the same coin but in core they are two different concepts.

An AI agent is any AI system that can make decision and take actions based on the perceived environment, and because it takes the decision based on that. It does not need to be autonomous, or even goal driven for an extended time period. Like a simple robot that follows a line on the floor, or a hoover that when hitting and object will change direction, or a spam filter that will check if the email is a good to go or a no, no. they mostly follow a script or may not have the memory capacity for long term goals.

So think about the Agentic AI as maybe the architecture who had to come up with the plan for the building and had to take into consideration all environmental and logical aspects before coming with the final blue print. And with the AI Agent as the construction worker who is placing the bricks according to the set in stone plan.

**5. Augmented AI**

While AI is made to be perceived as an autonomous tool to replace the human need in some tasks. Augmented AI the counter act for that, as the concept of it is to have the assistant of the AI during the given process not as the replacement but as a helper and a guidance while having the final saying to the human expert.

**5.1 Some features of the Augmented AI Systems:**

* **Human Presence**

The Human expert is present and in control during the whole process and has the making of the final decision

* **Assistance not Automation**

The AI gives helps with coming up with the ideas and potential decisions as well as highlighting some potential patterns and leads

* Transparency

The main goal is to come up with what helps the expert so the process of how the idea or advice came to be should always be explainable and easily interpretable.

Some examples of Augmented Ai models are the likes of AI diagnosis assistants in the medical field, like cancer detection with various types. Another example can be writing assistants, wither it is for spelling and grammar checking, paraphrasing tools or even coding like the GitHub copilot. They all help and suggest means for some decision making but do not possess the ability to make the final saying without the human approval.

**6. Docker**

I tried to install and work with Docker which is a great deployment tool to us for various project types. But the laptop I was provided denies that without any admin authorization. Will try to use my personal laptop, will that fix it? Idk man.

**7. FastAPI**

FastAPI is a modern, high-performance Python web framework designed for building APIs. It leverages Python's type hints and asynchronous capabilities to create efficient and robust web applications quickly. This performance is partly due to its use of Starlette and Pydantic.

**7.1. Characteristics:**

* **Asynchronous Support**

Handling large number of tasks and requests in a concurrent fashion

* **Automatic Data Validation and Serialization**

The use of Pydantic to validate, serialize and desacralize the data based on python type hints

* **Type Hints**

Checking for static type hints and reducing the errors

* **API Documentation**

The ability to generate interactive API documentations to simplify developing and testing

**8. Postgres**

PostgreSQL is a powerful, open-source, object-relational database management system (ORDBMS) known for its reliability, feature richness, and adherence to SQL standards. It supports both relational and non-relational (JSON) data types, making it versatile for various applications. Combining the aspects of relational databases with user defined functions and types allows it to be one powerful tool which is freely available on demand. Being also adherent to the standards of the usual SQL, while being able to deal with none relational data types like in the next case JSON, and even geospatial data types.

**8.1. Some Characteristics in Details**

* **Concurrency Control:**

It uses multi-version concurrency control (MVCC) to manage concurrent access to the database, ensuring data consistency and avoiding conflicts.

* **Extensibility:**

PostgreSQL can be extended with user-defined functions, data types, and procedural languages, making it highly adaptable to different needs.

* **Use Cases:**

PostgreSQL is used in various applications, including web development, data warehousing, geospatial data management, and social media platforms

The next step will be to develop a simple program using FastAPI and PostgreSQL tp implement CRUD (Create, Read, Update, Delete) concepts. Take the username, display a message for them.

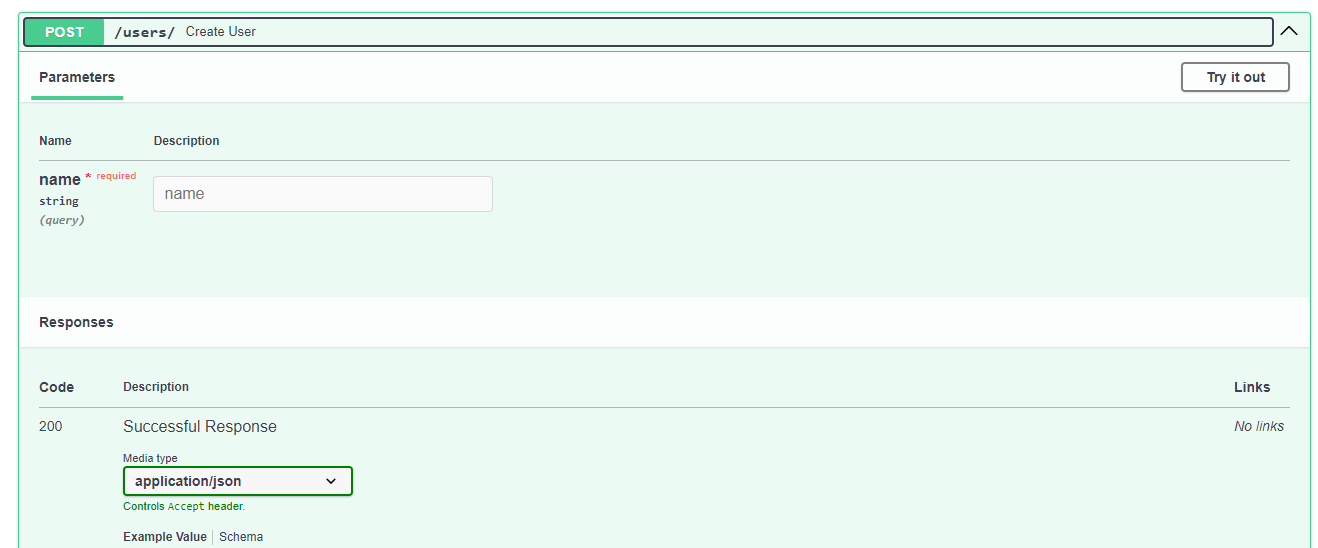
**9. Application Practice Results**

To start with the practice for this week. I started with using SQL-Lite with FastAPI to create a host that will take in the username, display a message for them. Then add an id to that user, starting from 1 till whatever. Then the usernames can be edited or deleted.

* Then I added another notebook to the GitHub directory that had the PostgreSQL. It was fairly the same workflow, with just the addition of the following.   
  Install psycopg2-binary.
* Using the pgAdmin4. logged in to the server, created my DB (lefastapi), created my user (leuser), sat the password and the privilages, granted the wizard.
* Updated the url

Then created a (.env) file so that I can log in and out without having the sensitive information in the notebook.

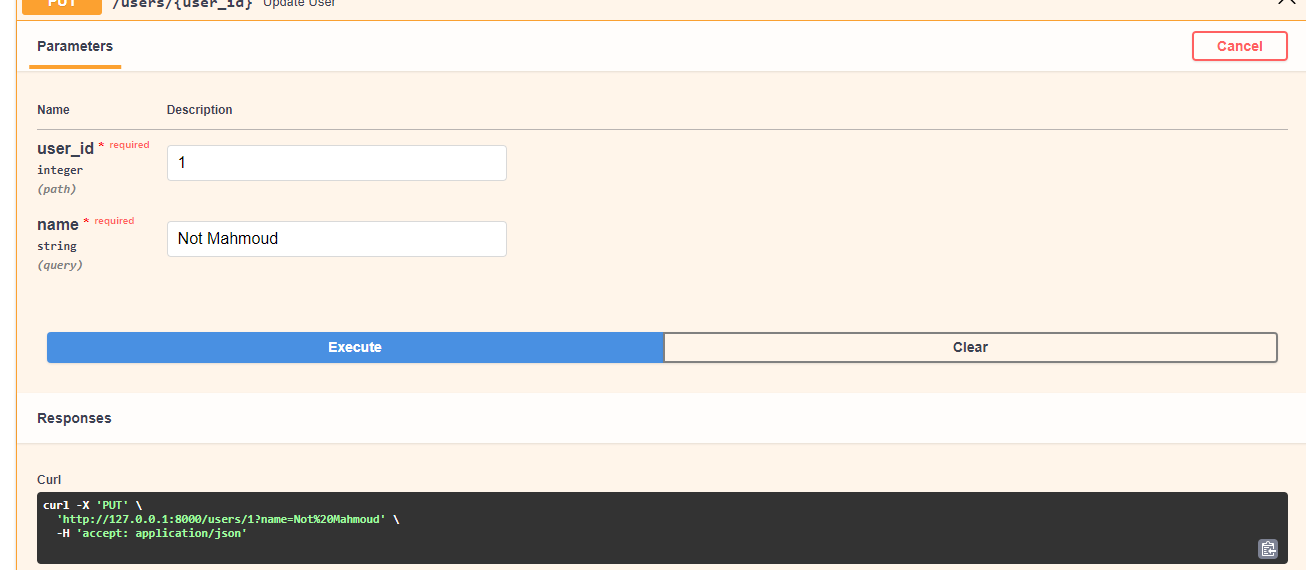
Bellow are the examples of how the page looks



A screenshot of a computer

Description automatically generatedA screenshot of a computer

Description automatically generatedA black and white striped sign

Description automatically generated with medium confidenceA screenshot of a computer

Description automatically generated

**References**

Langchain, “Introduction to Langchain,” Langchain Documentation. [Online]. Available: <https://python.langchain.com/docs/introduction/> [Accessed: Jul. 6, 2025].

Centers for Medicare & Medicaid Services, “Hospital General Information Dataset,” Kaggle. [Online]. Available: <https://www.kaggle.com/datasets/cms/hospital-general-information> [Accessed: Jul. 6, 2025].

Coursera, “LangChain for LLM Application Development Project,” [Online]. Available: <https://www.coursera.org/learn/langchain-for-llm-application-development-project/home/welcome> [Accessed: Jul. 6, 2025].

Sebastián Ramírez, FastAPI, [Online]. Available: <https://fastapi.tiangolo.com>. [Accessed: 07-Jul-2025].

[1] PostgreSQL Global Development Group, "PostgreSQL," PostgreSQL.org. <https://www.postgresql.org> (accessed Jul. 7, 2025).