**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

1. **Ways to answer analytical questions:**
2. Data in the **CSV format**
3. SQL tables
4. Pdf with tables => markdown format is preferred (needs more research on this one)

**Type 1:** We can first target the data in the CSV format i.e., **Chat with CSV data**

ask questions to a CSV file

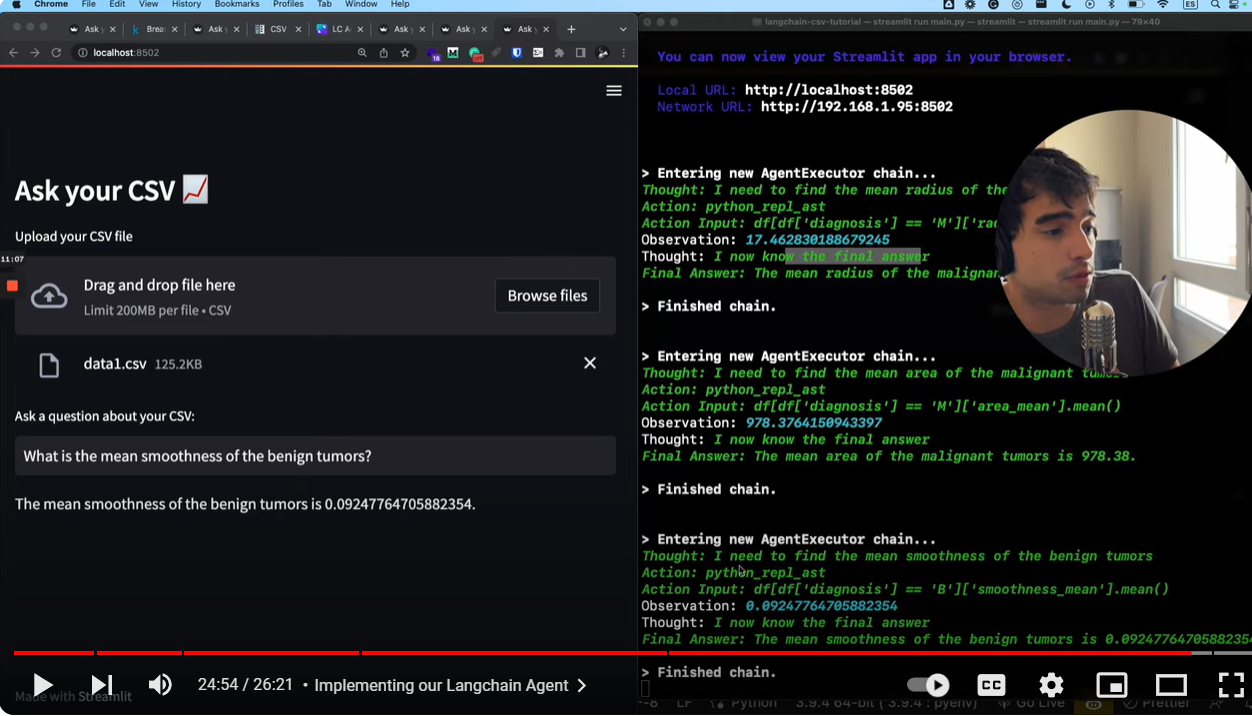
* Langchain has an agent called
* <https://python.langchain.com/v0.2/docs/integrations/toolkits/csv/> => this link contains the langchain agent to chat with the csv data for the questions like
* **Example questions:**
  1. How many people have more than 3 silblings?
  2. How many rows are there in the file?
  3. How many patients are older than 30 years?
  4. How many rows in the age column are different between the two dfs?

Question: How to do:? We will use the **LangChain’s** function of **create\_csv\_agent**

Answer:

1. <https://python.langchain.com/v0.2/docs/integrations/toolkits/csv/>
2. **Langchain** has a method called **create\_csv\_agent** and this is used for the chatting with the CSV files.
3. Agents or nodes is a new concept in the Langchain’s **LangGraph**.
4. The csv agent will have pandas functions which will create the data frame queries and get the analytical answer.
5. Then the analytical output should be sent to the another agent of the LLM which will create the output response.

This function will think in steps and construct the Natural Language Output Response as shown in the screenshot below.



**Important Point:**

The important point to be noted down is that the data is organized in the CSV files and we can get the **Analytical Question’s** numeric data through the **Pandas functions.** So, we don’t need Chroma DB like in RAG applications.

**References:**

1. <https://python.langchain.com/v0.1/docs/langgraph/>
2. <https://python.langchain.com/v0.2/docs/integrations/toolkits/csv/>
3. <https://youtu.be/tjeti5vXWOU?si=KTC-KSP1gi9uIy_q>