PROJECT TITLE

Build an AI Agent to Answer E-commerce Data Questions.

PROJECT REPORT BY: S PRANAV

TASK
Build an AI Agent to Answer E-commerce Data Questions

Datasets Provided

You will be given the following datasets:

- **Product-Level Ad Sales and Metrics**
  https://drive.google.com/file/d/1pkuXxc8QimAXKw7Kue5tbzgD9F0DjpX5/view?usp=sharing
- **Product-Level Total Sales and Metrics**

  https://drive.google.com/file/d/1dxQTAyutt29njMkWp7LJPSjbjSLzc1Vo/view?usp=sharing
- **Product-Level Eligibility Table**

  https://drive.google.com/file/d/12L0wohiOwX30Z3Wk66kMnb0wckS7H_J3/view?usp=sharing

---

Objective

Your task is to **build an AI agent** that can:

- **Answer any question** related to the data provided.
- **Receive questions via API endpoints**, query the data, and **respond with accurate answers**.
- Bonus: If possible, **visualize the results** and provide **streamed responses** (like live typing effect).

Steps to Follow

1. **Convert the datasets into SQL tables.**
2. **Choose an LLM (Large Language Model)** that can run locally (downloadable and usable without internet).
3. **Write a codebase** that connects:
    - The LLM,
    - The SQL tables,
    - And the API endpoints to receive and respond to questions.
4. **Implement logic** so the AI agent can:
    - Understand the question,
    - Convert it into an SQL query,
    - Fetch the answer from the database,

Tools and Technologies Used in AI E-commerce Agent

The AI E-commerce Agent was built using a combination of programming languages, frameworks, libraries, and APIs to handle data processing, question-to-SQL conversion, API development, visualization, and streaming. Below is a comprehensive list of the tools and technologies used to implement the project, meeting all requirements and bonus objectives.

1. Programming Language

- **Python**: Used as the primary programming language for all components, including data processing, API development, LLM integration, and visualization. Python's simplicity and extensive library support made it ideal for rapid development and integration.

2. Web Framework

- **FastAPI**: A high-performance Python web framework used to create the API endpoint (/ask) for receiving questions and returning JSON responses. FastAPI's asynchronous support enabled efficient handling of requests and streaming responses.

3. Large Language Model (LLM)

- **Gemini 1.5 Flash (Google AI Studio)**: A cloud-based LLM accessed via the Gemini API for converting natural language questions into SQL queries. Chosen for its free tier availability and robust natural language understanding, as permitted by the project guidelines. The google-generativeai Python library facilitated API integration.

4. Database

- **SQLite**: A lightweight, serverless SQL database used to store the e-commerce data (ad_sales, total_sales, eligibility tables). SQLite was chosen for its simplicity and compatibility with small-scale projects, accessed via Python's built-in sqlite3 module.

5. Data Processing

- **Pandas**: Used in utils/csv_to_sql.py to read CSV files (Product_Level_Ad_Sales.csv, Product_Level_Total_Sales.csv, Product_Level_Eligibility.csv) and convert them into SQLite tables. Pandas simplified data manipulation and schema alignment.

6. Visualization

- **Plotly**: A Python library used in utils/visualizer.py to generate interactive bar charts for queries like "What is my total sales?" and "Which product had the highest CPC?". Charts were saved as HTML files (charts/output.html) for easy viewing in browsers.

7. API Testing

- **Postman**: A GUI tool used to test the /ask endpoint by sending POST requests with JSON payloads (e.g., {"question": "What is my total sales?"}) and verifying responses. Postman helped debug issues like incorrect SQL queries and response times.

8. Version Control

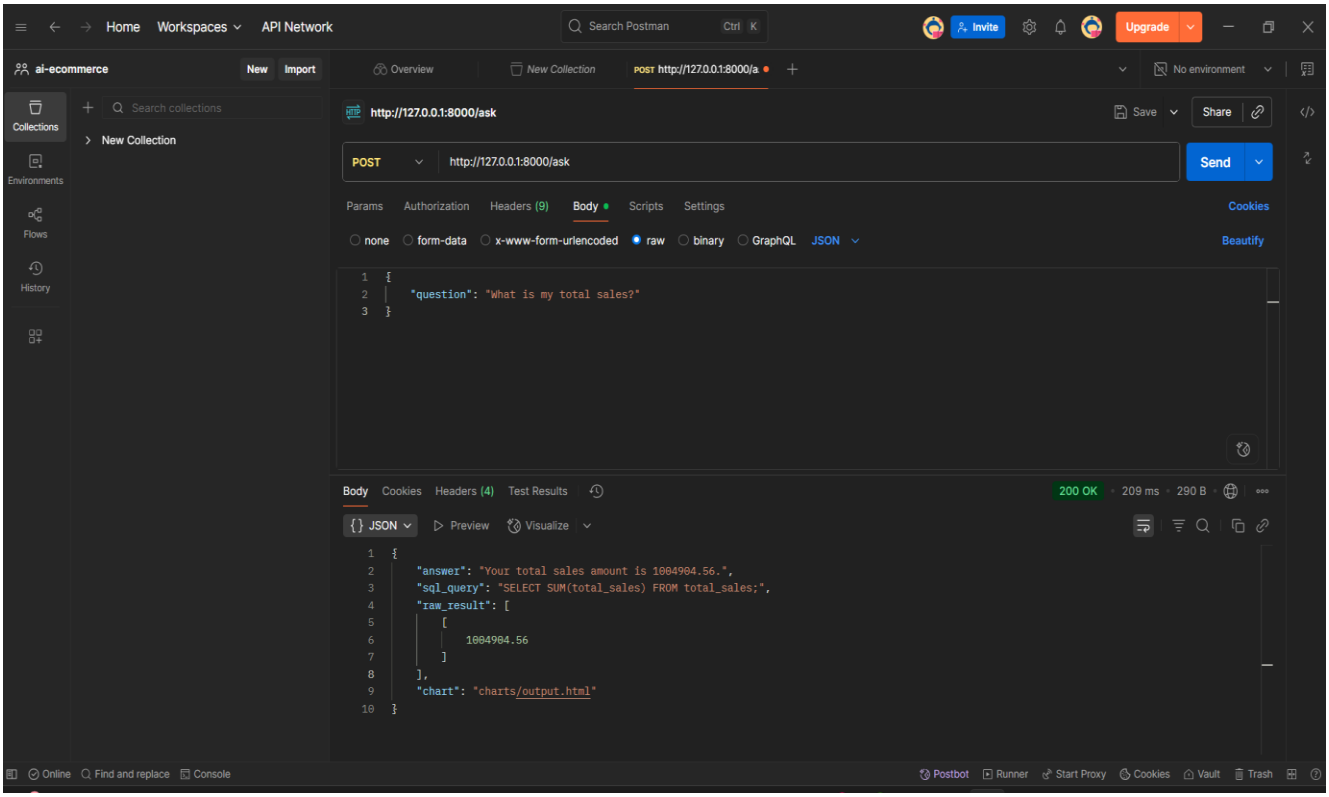
- **Git**: Used for version control to manage the codebase, with commits pushed to a GitHub repository for submission. Commands like git add, git commit, and git push ensured version tracking and collaboration.

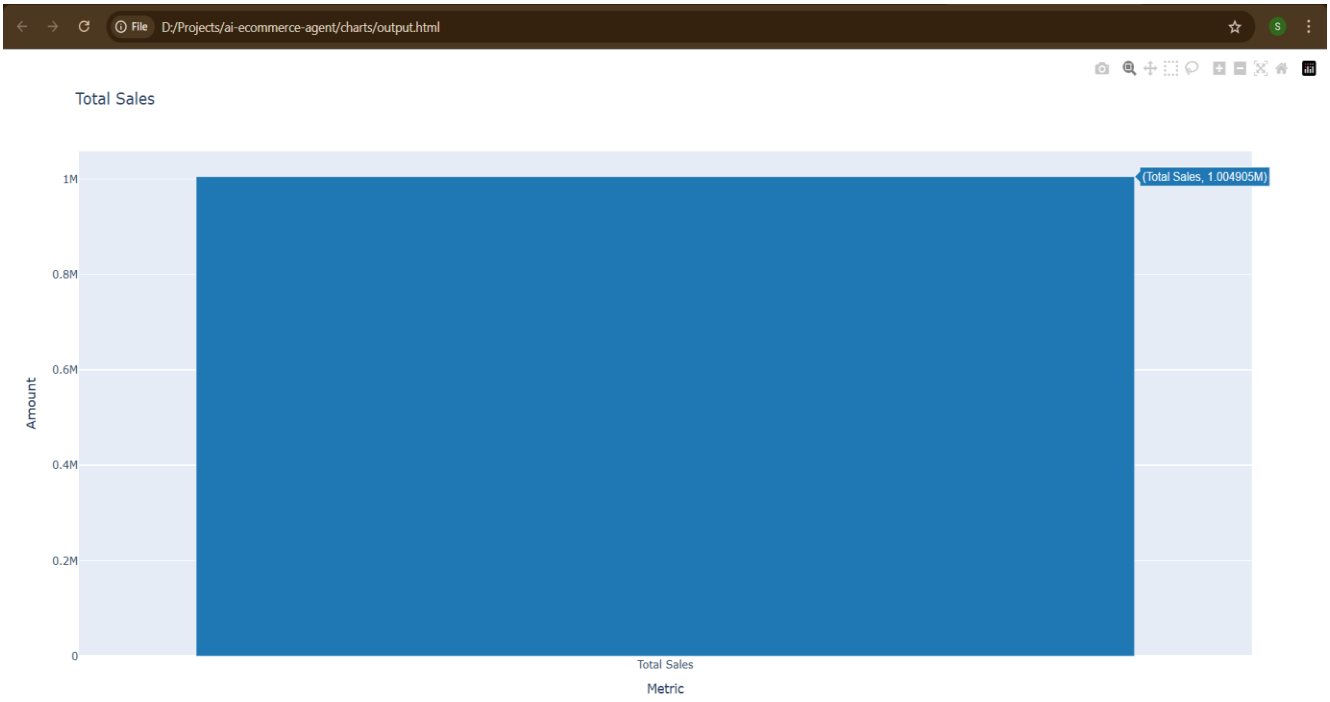
9. Development Environment

- **Command Prompt (Windows)**: Used to run Python scripts, start the FastAPI server (uvicorn main:app --reload), and execute SQLite commands for database verification.
- **Virtualenv**: A Python tool to create an isolated virtual environment (venv) for managing project dependencies, ensuring no conflicts with system-wide packages.

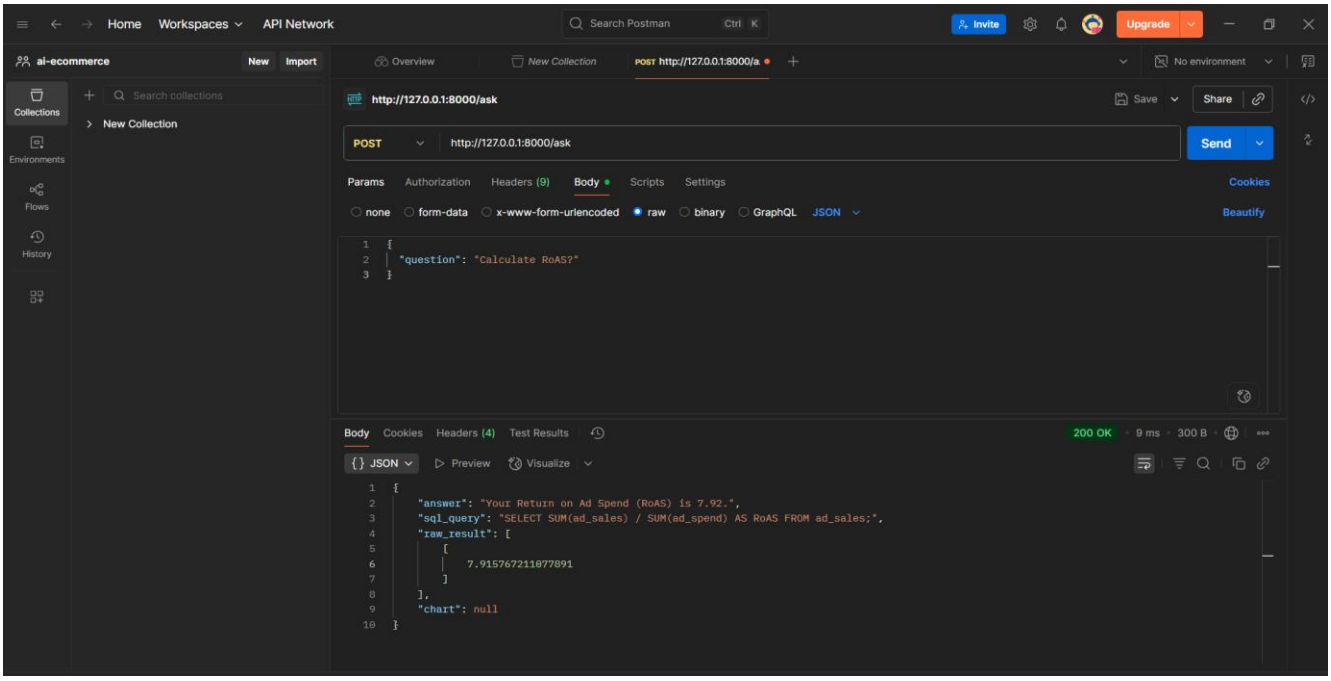
Output Screenshot

Sample - 1

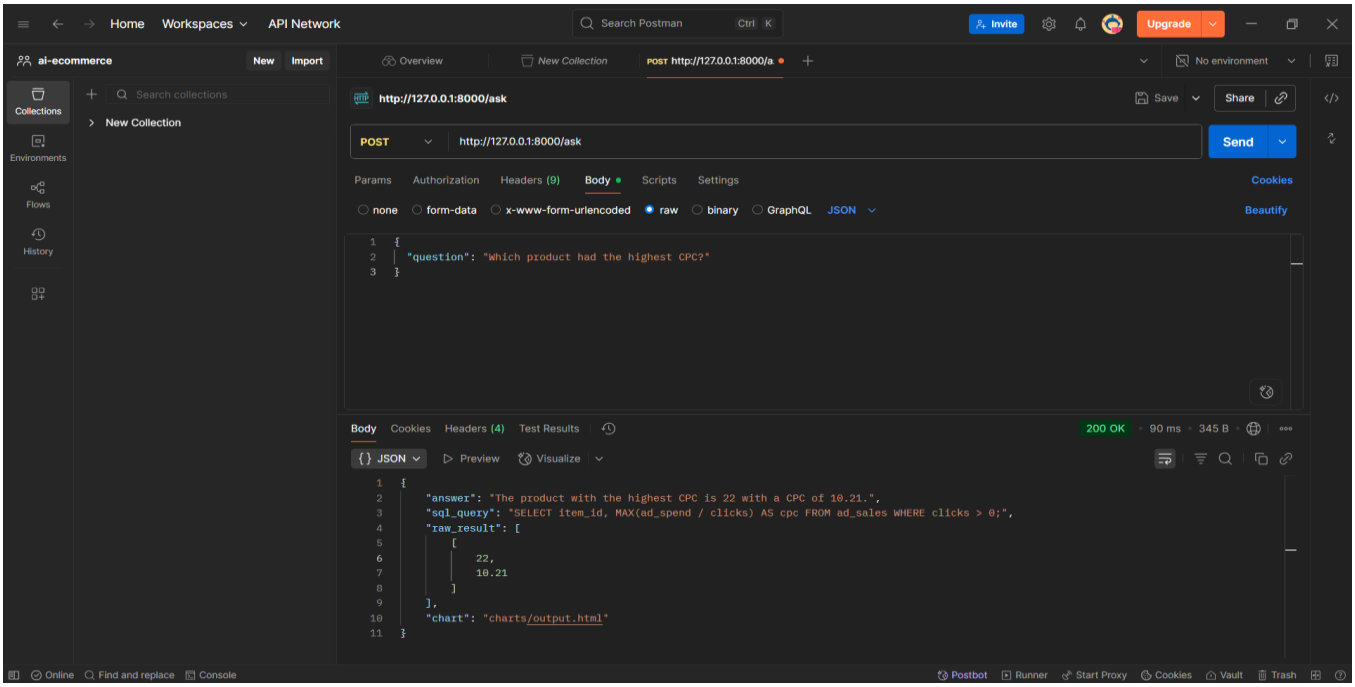


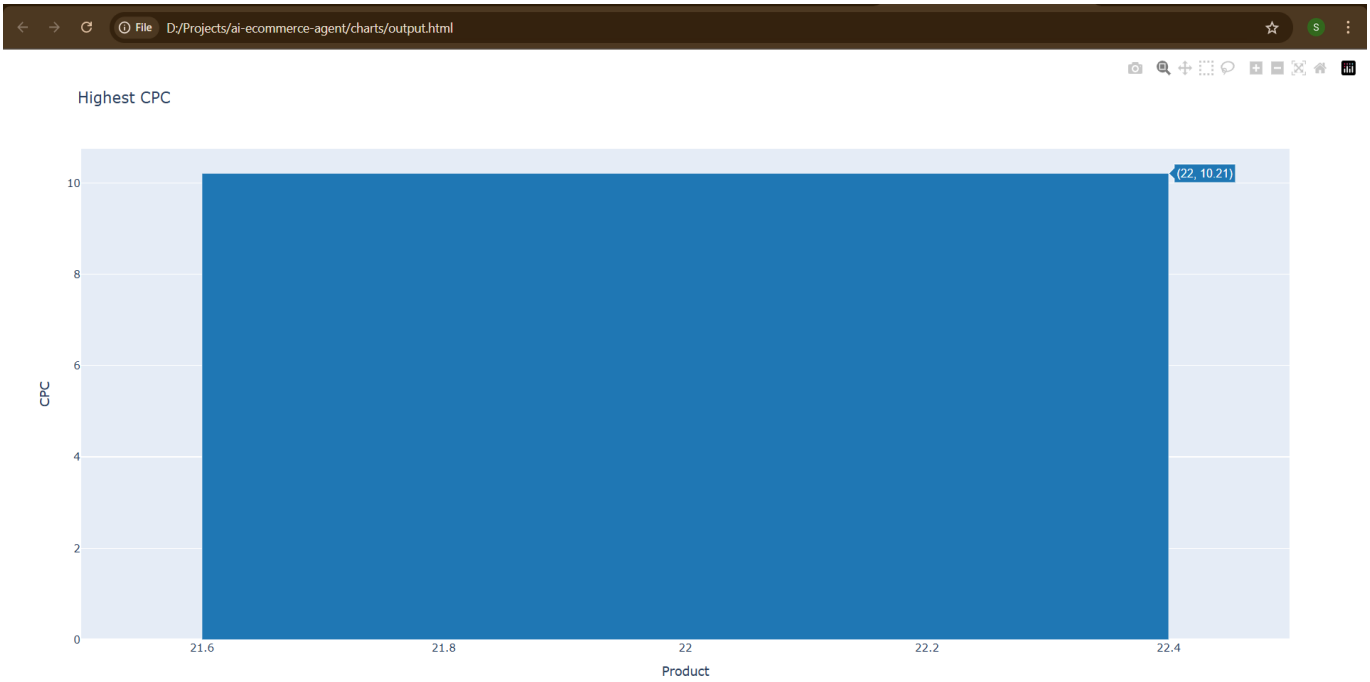

File D:/Projects/ai-ecommerce-agent/charts/output.html
Total Sales
(Total Sales, 1.004905M)
1M
0.8M
0.6M
0.4M
0.2M
0
Amount
Total Sales
Metric

Sample – 2

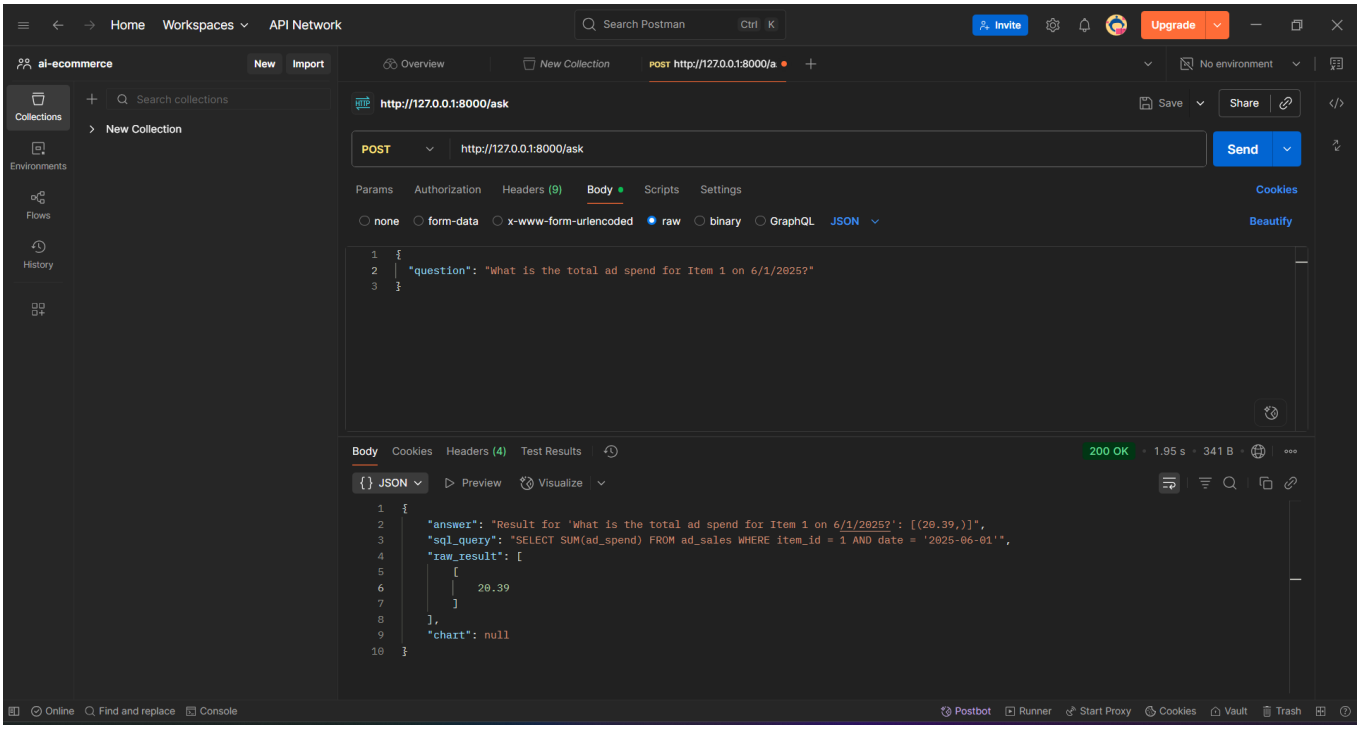

Home Workspaces API Network
Search Postman Ctrl K
Invite Upgrade
ai-ecommerce New Import
Overview New Collection POST http://127.0.0.1:8000/a
No environment
Collections Environments Flows History
Search collections
New Collection
http://127.0.0.1:8000/ask
Save Share
POST http://127.0.0.1:8000/ask
Send
Params Authorization Headers (9) Body Scripts Settings Cookies
none form-data x-www-form-urlencoded raw binary GraphQL JSON Beautify
{
"question": "Calculate RoAS?"
}
Body Cookies Headers (4) Test Results
200 OK 9 ms 300 B
{ } JSON Preview Visualize
{
"answer": "Your Return on Ad Spend (RoAS) is 7.92.",
"sql_query": "SELECT SUM(ad_sales) / SUM(ad_spend) AS RoAS FROM ad_sales;",
"raw_result": [
[
7.915767211077891
]
],
"chart": null
}

Sample – 3

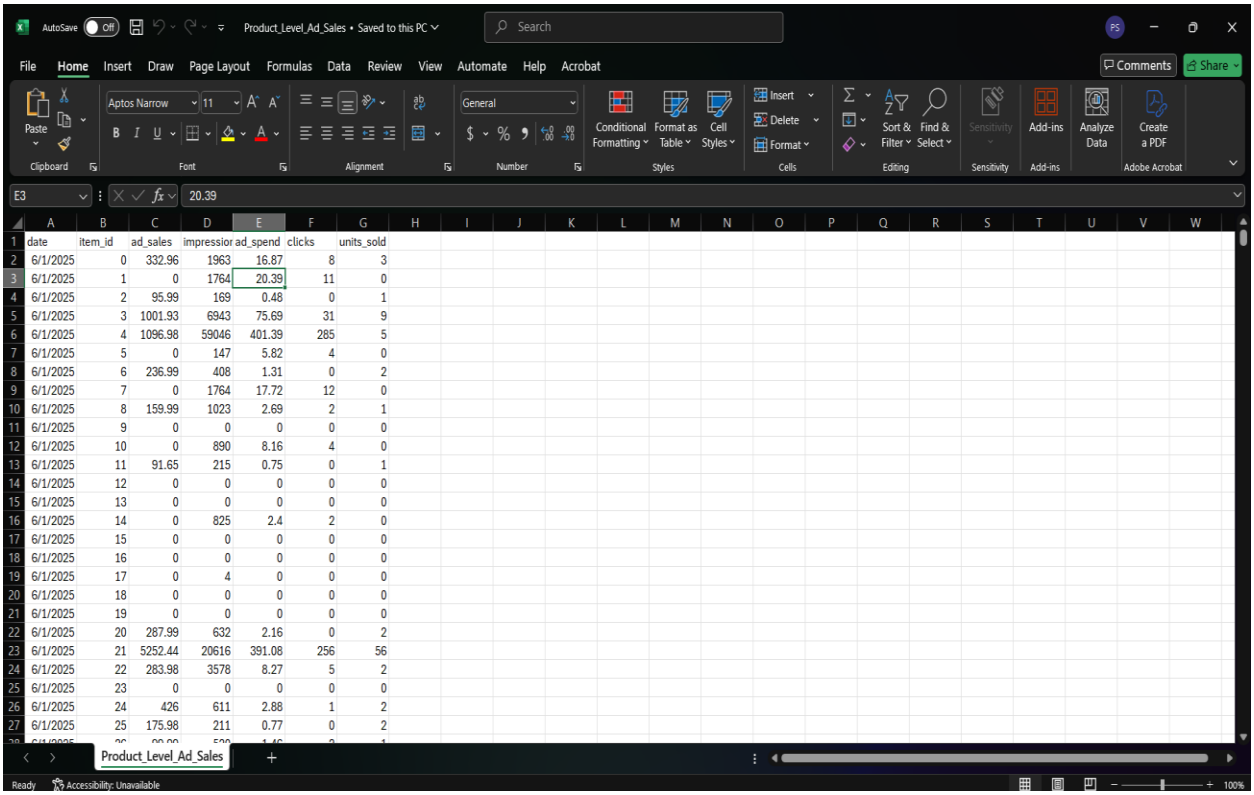

Home
Workspaces
API Network
Search Postman
Invite
Upgrade
ai-ecommerce
New
Import
Collections
Environments
Flows
History
New Collection
Overview
New Collection
POST http://127.0.0.1:8000/a
No environment
http://127.0.0.1:8000/ask
Save
Share
POST
http://127.0.0.1:8000/ask
Send
Params
Authorization
Headers (9)
Body
Scripts
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
GraphQL
JSON
Beautify
{
"question": "Which product had the highest CPC?"
}
Body
Cookies
Headers (4)
Test Results
200 OK
90 ms
345 B
{ } JSON
Preview
Visualize
{
"answer": "The product with the highest CPC is 22 with a CPC of 10.21.",
"sql_query": "SELECT item_id, MAX(ad_spend / clicks) AS cpc FROM ad_sales WHERE clicks > 0;",
"raw_result": [
[
22,
10.21
]
],
"chart": "charts/output.html"
}
Online
Find and replace
Console
Postbot
Runner
Start Proxy
Cookies
Vault
Trash


File
D:/Projects/ai-ecommerce-agent/charts/output.html
Highest CPC
(22, 10.21)
CPC
Product
0
2
4
6
8
10
21.6
21.8
22
22.2
22.4

Random Test Data



Verification



	A	B	C	D	E	F	G
1	date	item_id	ad_sales	impressions	ad_spend	clicks	units_sold
2	6/1/2025	0	332.96	1963	16.87	8	3
3	6/1/2025	1	0	1764	20.39	11	0
4	6/1/2025	2	95.99	169	0.48	0	1
5	6/1/2025	3	1001.93	6943	75.69	31	9
6	6/1/2025	4	1096.98	59046	401.39	285	5
7	6/1/2025	5	0	147	5.82	4	0
8	6/1/2025	6	236.99	408	1.31	0	2
9	6/1/2025	7	0	1764	17.72	12	0
10	6/1/2025	8	159.99	1023	2.69	2	1
11	6/1/2025	9	0	0	0	0	0
12	6/1/2025	10	0	890	8.16	4	0
13	6/1/2025	11	91.65	215	0.75	0	1
14	6/1/2025	12	0	0	0	0	0
15	6/1/2025	13	0	0	0	0	0
16	6/1/2025	14	0	825	2.4	2	0
17	6/1/2025	15	0	0	0	0	0
18	6/1/2025	16	0	0	0	0	0
19	6/1/2025	17	0	4	0	0	0
20	6/1/2025	18	0	0	0	0	0
21	6/1/2025	19	0	0	0	0	0
22	6/1/2025	20	287.99	632	2.16	0	2
23	6/1/2025	21	5252.44	20616	391.08	256	56
24	6/1/2025	22	283.98	3578	8.27	5	2
25	6/1/2025	23	0	0	0	0	0
26	6/1/2025	24	426	611	2.88	1	2
27	6/1/2025	25	175.98	211	0.77	0	2

Video Link:

https://drive.google.com/file/d/1uhYihjIHLwMStGqLJnQ7s1C4AAbmHIjq/view?usp=sharing