



Project Launch : AI Question - Answer Helper

By :

DevTown - India's Fastest Growing Tech Learning Community

About Devtown Bootcamp :

- ▶ DevTown offers a range of bootcamps that cater to various tech skills and career paths.
- ▶ Here's a brief overview of what I learnt on:

AI Agent Development Bootcamp:

This 5 -day bootcamp focuses on building an AI agent using Langgraph. Students will learn to set up the environment, build a foundational chatbot system, create multi - agent workflows, integrate tools for smarter AI actions, and deploy and test their AI agent.



Project Description :

- ▶ A simple AI agent that:
 - Answers user questions
 - Uses a small search tool for factual queries
 - Maintains short-term memory for context
- ▶ Beginner-friendly project to understand **agent + tool behavior**

What Is This Project About?

- ▶ Build an AI agent that:
 - Responds conversationally
 - Uses tools for factual answers
 - Stores short-term memory (recent questions)
- ▶ Learn core ideas of **agentic AI**: reasoning, tool use, and memory



Project Steps :

- 1. Set up environment** (Python, pip, Streamlit)
- 2. Build search tool** (dictionary-based facts)
- 2. Implement agent logic** (factual vs conversational)
- 3. Add memory** (store recent inputs)
- 4. Create web interface** (Streamlit chatbot)
- 5. Test and improve** (add robot greeting, cute replies)

Code Snippets :

► Search Tool :

```
1 def search_tool(query):
2     query = query.lower()
3     for key in facts_db:
4         if key in query:
5             return facts_db[key]
6     return "Sorry, I don't know that."
```

► Agent Response :

```
1 def agent_response(user_input):
2     if is_factual(user_input):
3         answer = search_tool(user_input)
4         if "Sorry" not in answer:
5             return f"🤖 Here's what I found: {answer}"
6         else:
7             return "💡 Hmm, I don't know that yet!"
8     else:
9         return f"🤖 Beep beep! You said: '{user_input}'"
```



Full Code :

```
Chat_app.py X
C:\Users\TAMILARASI B\OneDrive\Desktop\AI_QA_Helper> Chat_app.py
1 # chat_app.py
2
3 import streamlit as st
4 import random
5
6 # --- Robot creating section ---
7 st.markdown(
8     """
9         

10             
11             

I'm here to answer your questions and chat with you! 🤖


12


13     """,
14     unsafe_allow_html=True
15 )
16
17 # optional: Add a cute robot image
18 st.image("https://cdn-icons-png.flaticon.com/512/413/4712189.png", width=150)
19
20 # --- Simple facts database ---
21 facts_db = {
22     "capital of France": "Paris",
23     "largest planet": "Jupiter",
24     "speed of light": "299,729 km/s",
25     "who wrote hamlet": "William Shakespeare",
26     "current year": "2025",
27     "python creator": "Guido van Rossum",
28     "meaning of AI": "Artificial intelligence is the simulation of human intelligence by machines."
29 }
30
31 # --- Search tool with keyword matching ---
32 def search_tool(query):
33     query = query.lower()
34     for key in facts_db:
35         if key in query:
36             return facts_db[key]
37     return "Sorry, I don't know that."
```

```
Chat_app.py X
C:\Users\TAMILARASI B\OneDrive\Desktop\AI_QA_Helper> Chat_app.py > search_tool
1
2
3     # --- check if question is factual ---
4     def is_factual(question):
5         keywords = ["what", "when", "where", "who", "how many", "which", "give me"]
6         return any(question.lower().startswith(k) for k in keywords)
7
8     # --- agent response logic with cute replies ---
9     def agent_response(user_input):
10         cute_replies = [
11             "Beep boop! I heard you say: \"{}\".",
12             "Robo says: \"{}\".",
13             "That's interesting! Tell me more about \"{}\".",
14             "Hi friend! You said: \"{}\"."
15         ]
16
17         if is_factual(user_input):
18             answer = search_tool(user_input)
19             if "Sorry" not in answer:
20                 return f"Here's what I found: {answer}"
21             else:
22                 return "Hm, I don't know that yet!"
23         else:
24             return random.choice(cute_replies).format(user_input)
25
26     # --- Streamlit UI ---
27     st.title("AI Question-Answer Helper")
28     st.write("Ask me anything!")
29
30     user_input = st.text_input("Your question?")
31
32     if user_input:
33         response = agent_response(user_input)
34         st.write("🤖", response)
```



Streamlit Web Interface :

- Robot greeting with emoji and image
- Text input for user questions
- Cute randomized replies for casual chat
- Factual answers with icons
- Example questions:
 1. “What is the capital of France?” → Paris
 2. “Who wrote Hamlet?” → Shakespeare
 3. “I love Paris.” → Cute robot reply



Tips for Students :

- Keep your tool simple (dictionary or basic function)
- Test both factual and non-factual queries
- Focus on the workflow: reasoning → tool call → final answer
- Use environment variables for API keys
- Use Langchain or LlamaIndex for advanced memory/tool integration



Challenges & Fixes :

ISSES	FIX
1. streamlit not found	Installed in correct Python version
2. ScriptRunContext warning	Used streamlit run Chat_app.py
3. VS Code import error	Selected correct interpreter
4. Simple replies	Added emoji, robot image, random cute responses



Learnings :

- How to build a basic agent with reasoning and memory
- Importance of environment setup and correct execution
- How to use Streamlit for interactive AI apps
- How to debug and improve chatbot behavior



Future Enhancements :

- Add **chat history panel**
- Connect to **Wikipedia or OpenAI API**
- Style chatbot with **cartoon CSS**
- Add **voice or image generation**
- Deploy online for public use

Conclusion :

- Successfully built and tested an **AI Question-Answer Helper**
- Learned agentic AI fundamentals
- Ready to expand into more advanced projects .