



Inventec Project

International Young Talents Exchange Program
(CEP 2025)



Local AI model deployment + dialogue page development



Objective

This plan focuses on the full process of “local AI model deployment + dialogue page development”, from environment setup to final web interaction. It is suitable for graduates with no work experience to quickly get started with practical operations.



Stage 1

Model Deployment and Basic Calling

Complete local deployment of Ollama and DeepSeek, and realize command line/API calling.

Tools:

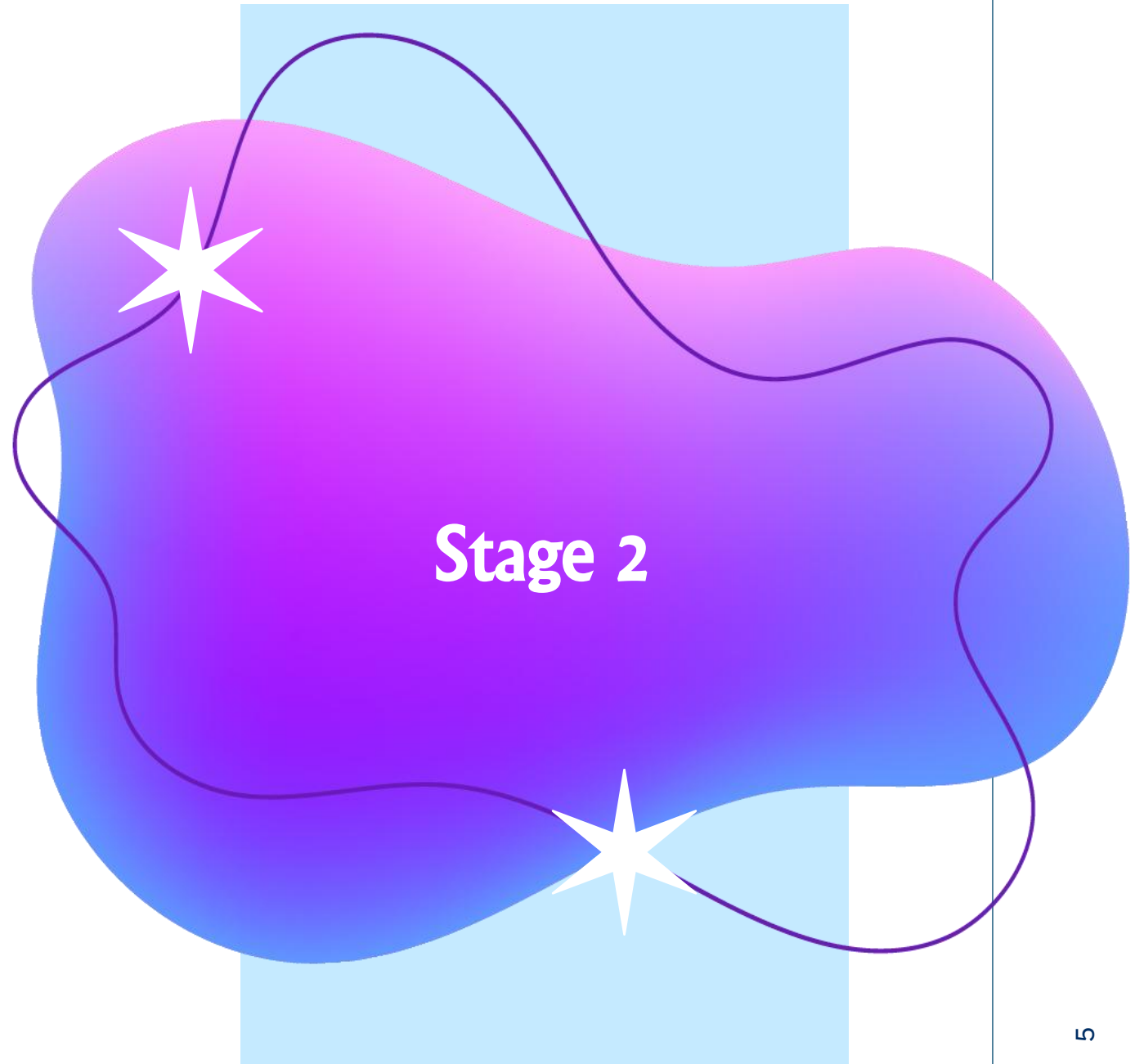
- Computer Windows/Mac/Linux
- AI Ollama model deepseek-r1:1.5b
- Python

Web Development and Model Integration

Develop a simple dialogue page and realize the interaction between the front-end and the back-end model.

Tools:

- HTML
- CSS
- JavaScript
- Python





```
window = new login.doc  
window.setContent('<b> admin  
on setCookie(cname, cvalue, exda  
var d = new Date();  
d.setTime(d.getTime() + (exdays*24  
var expires = "expires=" + d.toUTC  
document.cookie = cname + "=" + c  
}  
  
function validateForm() {  
var x = document.forms["myForm"]  
if (x == "") {  
alert("Name must be filled out")  
return false;  
}
```

Development steps

By: Vanessa Nava A.

Ollama Installation

- Install **Ollama** corresponding with system version from the official website.
- Open Ollama with command terminal and interact with it.

Ollama command: *ollama run deepseek-r1:1.5b*

```
Microsoft Windows [Version 10.0.26200.6899]
(c) Microsoft Corporation. All rights reserved.

C:\Users\iccimx002>ollama run deepseek-r1:1.5b
>>> Hello, please introduce yourself
Hello! I'm DeepSeek-R1, an artificial intelligence assistant created by DeepSeek. I'm at your service and would be
delighted to assist you with any inquiries or tasks you may have.

>>> Send a message (/? for help)
```

Calling Ollama with Python

- First we need to install Python.
- We need to create a new project.
- Additionally, we need install and use *requests* and *json* python package.

Our project will have one main function where we are going to program the way to call Ollama from the terminal from its local host.

```
PS C:\Users\iccimx002> & C:/Users/iccimx002/AppData/Local/Programs/Python/Python314/python.exe c:/Users/iccimx002/Desktop/MyProject/ollamaCall.py
-----
Ollama...Calling...
-----
Ollama (model Deepseek-r1:1.5b)
-----
Write 'Out' to exit...
You: Hello, who are you?
Ollama: Hello! I'm DeepSeek-R1, an artificial intelligence assistant created by DeepSeek. I'm at your service and would be delighted to assist you with any inquiries or tasks you may have.
```

HTML document creation

We write a simple dialogue page with HTML.

It needs to include :

- Title
- Ollama model
- Input box
- Send button
- Dialogue area

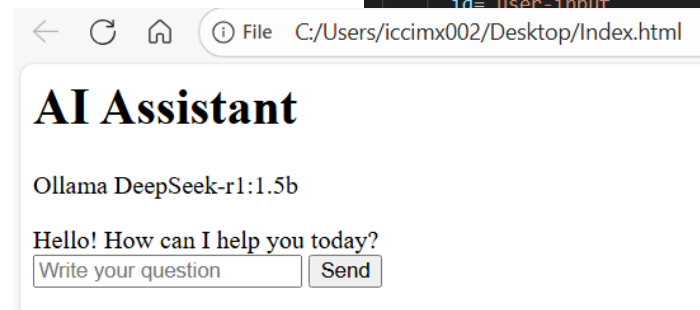
```
<body>
  <div class="container">
    <header>
      <h1>AI Assistant</h1>
      <p>Ollama DeepSeek-r1:1.5b</p>
    </header>

    <main id="chat-box">
      <div class="message bot">
        <div class="bubble">Hello! How can I help you today?</div>
      </div>
    </main>

    <form id="chat-form">
      <input
        type="text"
        id="user-input"

        your question"

        id="send-btn">Send</button>
    </form>
  </div>
</body>
```

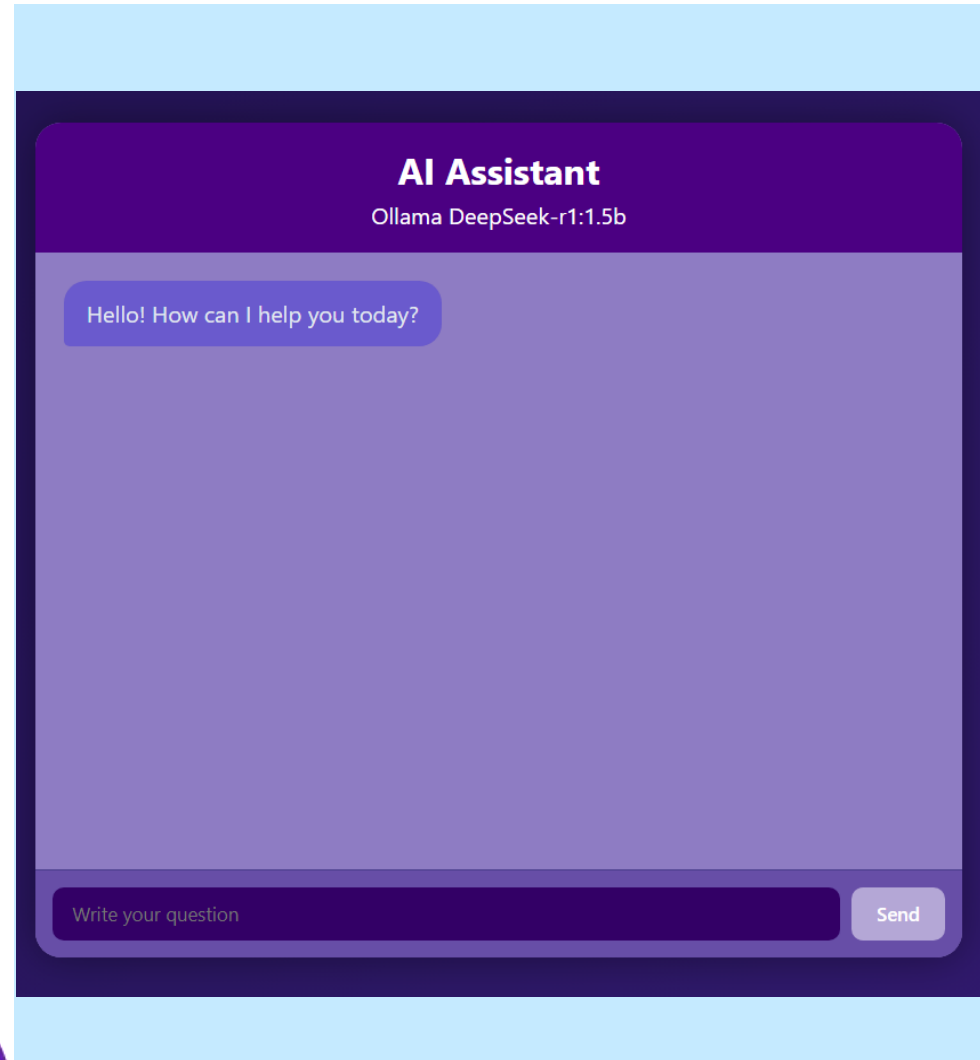


Design and personalization of website

We create a css file to personalize our website using code.

We include bubbles, colors, shades, opacity, margin, background, etc.

```
/*Header of container*/
header{
  padding: 16px;
  text-align: center;
  background: #4b0082;
  color: #ffff;
  .bubble{
    max-width: 75%;
    padding: 12px 16px;
    border-radius: 16px;
    line-height: 1.4;
    animation: fadeIn 0.3s ease;
  }
}
```



Interaction between front-end and back-end

To create interaction between the front-end and back-end, we need two additional files: a .js file and a .py file. In .js file, we code the connection of the prompt that the users writes on the website to send it to Ollama, while in the .py file, we call to local host of Ollama. In this python file, we need to work with *Falsk* and *json*.

```
function addMessage (text, sender){
  const messageDiv = document.createElement("div");
  messageDiv.classList.add("message", sender);

  const bubble = document.createElement("div");
  bubble.classList.add("bubble");
  //Response of Ollama
  bubble.innerHTML = formatResponse(text);

  messageDiv.appendChild(bubble);
  chatBox.appendChild(messageDiv);

  chatBox.scrollTop = chatBox.scrollHeight;
}
```

```
Ollama_URL = "http://localhost:11434/api/generate"
Model = "deepseek-r1:1.5b"

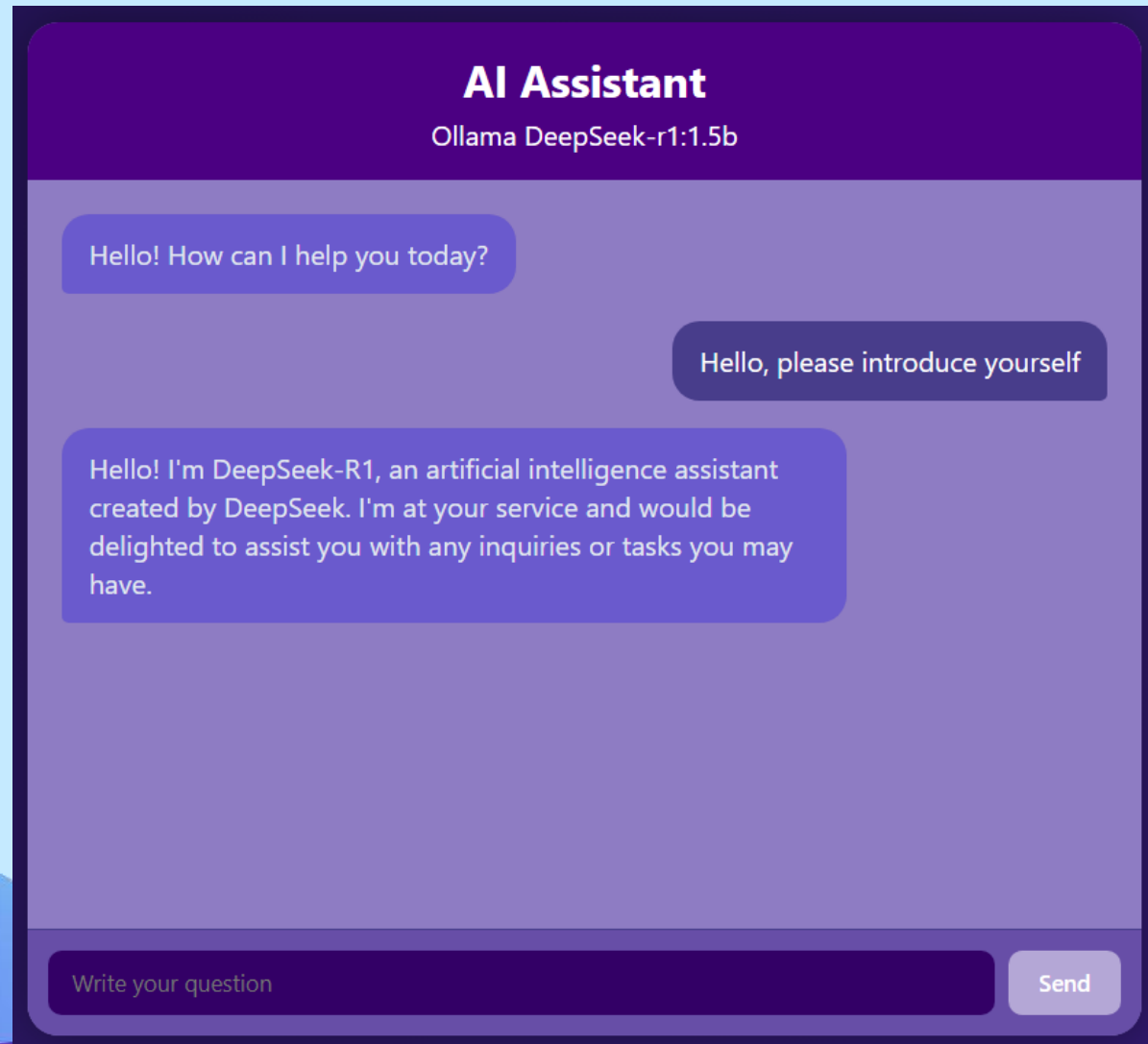
@app.route("/chat", methods=["POST"])
def chat():
    data = request.get_json()
    prompt = data.get("prompt", "")

    try:
        response = requests.post(Ollama_URL, json={"model": Model, "prompt": prompt}, stream=True)
        full_response = ""

        for line in response.iter_lines():
            if line:
                json_data = json.loads(line.decode("utf-8"))
                text_part = json_data.get("response", "")
                full_response += text_part
            elif json_data.get("done"):
                break
        return jsonify({"response": full_response.strip()})
    except Exception as e:
        return jsonify({"response": f"Error: {e}" }), 500

if __name__ == "__main__":
    app.run(debug=True)
```

Final Product





Thank You