

# lessoncode1

L1\_App\_html.py

## Creating an HTML Preview Web App

This Python script **creates an AI-powered web app** using **Gradio** and **Ollama**. The app allows users to:

1. **Generate AI responses** from different models.
2. **Detect and preview HTML output** by opening it in a browser.

This is a foundational lesson for **handling AI-generated HTML** and displaying it dynamically.

### Code Explanation

#### 1 Importing Required Modules

```
import gradio as gr
import subprocess
import re
import tempfile
import webbrowser
import os
```

- **gradio** → Used to create the web-based UI.
- **subprocess** → Runs terminal commands (to call Ollama).
- **re** → Used for cleaning up AI responses.
- **tempfile** → Creates a temporary HTML file for preview.
- **webbrowser** → Opens HTML output in a browser.
- **os** → Manages file paths.

#### 2 Running the AI Model & Cleaning Responses

```
def run_ollama_prompt(prompt, model):
    """Runs a prompt using Ollama CLI and returns the response."""
    command = ["ollama", "run", model, prompt]

    try:
        result = subprocess.run(command, capture_output=True, text=True, c
heck=True, encoding="utf-8")
        response = result.stdout.strip()
        # Remove <think>...</think> tags if present
        response = re.sub(r'<think>.*?</think>', '', response, flags=re.DOTAL
L).strip()
        return response
    except subprocess.CalledProcessError as e:
        return f"Error running Ollama: {e}"
```

- **Runs the AI model** and returns the generated text.
- **Cleans AI-generated** `<think>` tags that may contain unnecessary reasoning before the response.
- **Handles errors gracefully** to prevent crashes.

### 3 Detecting & Previewing AI-Generated HTML

```
def save_and_open_html(response):
    if response.strip().lower().startswith("<!doctype html>") or "<html" in res
ponse.lower():
        with tempfile.NamedTemporaryFile(delete=False, suffix=".html", mode
="w", encoding="utf-8") as temp_file:
            temp_file.write(response)
            temp_file_path = temp_file.name
            webbrowser.open(f"file://{temp_file_path}")
            return "HTML preview opened in a new tab."
    return "No valid HTML detected in the response."
```

- ♦ **What This Does:**
  - **Checks if the response contains valid HTML.**
  - **If HTML is detected, it:**

- **Creates a temporary HTML file.**
- **Opens it in the browser** for preview.
- **Returns a confirmation message.**
- If **no HTML is found**, it simply states that **no valid HTML was detected**.

## 4 Creating the Gradio Web Interface

```
models = ["deepseek-r1:1.5b", "deepseek-r1:14b", "deepseek-r1:32b", "phi
4:latest", "llama2:7b"]

with gr.Blocks() as app:
    gr.Markdown("# Ollama Chat App")

    with gr.Row():
        model_selector = gr.Dropdown(models, label="Select Model", value
="deepseek-r1:1.5b")

    with gr.Row():
        prompt_input = gr.Textbox(label="Enter your prompt", placeholder="T
ype something...")

    with gr.Row():
        submit_button = gr.Button("Generate Response")
        html_preview_button = gr.Button("Open HTML Preview")

    response_output = gr.Textbox(label="Response", interactive=False)

    submit_button.click(fn=generate_response, inputs=[prompt_input, model
_selector], outputs=response_output)
    html_preview_button.click(fn=save_and_open_html, inputs=response_out
put, outputs=response_output)
```

### ♦ Key Features:

- **Dropdown for model selection** ( `deepseek` , `phi4` , `llama2` , etc.).
- **Textbox for user input** (prompt).
- **Button for generating responses**.

- Button for HTML preview.
  - Output field to display responses.
- 

## 5 Launching the Web App

```
app.launch()
```

- Starts the Gradio app, allowing interaction via a web browser.
- 

## How to Run the App

### 1 Ensure dependencies are installed:

```
pip install gradio
```

### 2 Run the script:

```
python L1_App_html.py
```

### 3 Open the provided link in your browser.

### 4

Enter a prompt and generate a response.

### 5

If the response is HTML, click "Open HTML Preview".

---

## ★ Example Usage

### Scenario 1: Normal Text Response

1. Enter: "What is the capital of France?"
2. AI responds: "Paris"
3. HTML preview button does **nothing** (since it's not HTML).

### Scenario 2: HTML Response

1. Enter: "Generate a simple HTML page with a heading and paragraph."
2. AI responds with HTML code:

```
<!DOCTYPE html>
<html>
<head><title>Test Page</title></head>
<body><h1>Hello, World!</h1><p>This is a sample HTML page.</p></
body>
</html>
```

3. Click "**Open HTML Preview**" → The page opens in the browser.
-