DATA-236 Sec 12 - Distributed Systems for Data Engineering

HOMEWORK 1(2 Parts)

Instructions:

- Please provide the code solution for each question along with its intended output. Ensure that the code and corresponding output screenshots are placed together, one below the other.
- Screenshots must be provided for the output of each question.
- Submission should be in PDF Format

Questions:

I. HTML

1. Create an HTML page with the title "HW1-{Your Name}". (0.5 points)
2. Add a heading tag to name the title of the blog topic. *(Note. you have to use the header tag which renders the biggest font size).* (0.5 points)
3. Create a form for the blog. The form should include the following:
    a. A text input for the blog title, placeholder text “Enter the title of your blog”. This field should be required and the cursor should automatically focus on this field when the page loads. (0.5 points)
    b. A text input for the author name, placeholder "Enter your name”, and required. (0.5 points)
    c. An email input for the email address, placeholder “Enter your email”, and required. (0.5 points)
    d. A text area for the blog content, placeholder “Write your content here…”, and required. (0.5 points)
    e. A dropdown for category selection, and options "Technology," "Lifestyle," "Travel," and "Education." (0.5 points)
    f. A checkbox and a label with the text “I agree to the terms and conditions.” (0.5 points)
    g. A submit button with the text “Publish Blog”. (0.5 points)
4. Add a script tag to link your javascript code for part II at the end of your HTML file. (0.5 points)

II. Javascript

1. Write a javascript function using an arrow function to validate:
    a. Verify if the blog content is more than 25 characters. If the validation fails, display an alert with the message “Blog content should be more than 25 characters”. (1 points)
    b. Verify if the terms and conditions check box is checked. If the validation fails, display an alert with the message “You must agree to the terms and conditions”. (1 points)

2. After the form submission is successful, convert the form data into a JSON string and log the output in the console. (2 points)
3. Use object destructuring to extract the title and email fields from the parsed object and log their values in the console. (2 points)
4. Use the spread operator to add a new field “submissionDate” with the current date and time to the parsed object. Log the updated parsed object in the console. (2 points)
5. Create a closure to track how many times the form has been successfully submitted and log the submission count each time the form is submitted. (2 points)

DEPLOYMENT:

Docker: Create a docker image of the above application, and build and run the application using docker in your local.
Provide the screenshot of your app running on your localhost.

AWS ECS: Create an AWS ECS service (only one task) running the above application using the docker image created above.
Provide the screenshot of your app running on your public IP address.

Agentic AI (Part 2)

Build two tiny “agents” that talk to each other using a small local LLM (via **Ollama**).
Input: a blog **title** and **content**.
Output: exactly **3 topical tags** + a **≤25-word summary** — produced through a short Planner → Reviewer → Finalizer flow — and printed as **valid JSON**.

Requirements:

- Python **3.11+** (3.11 or 3.12 recommended). Note: 3.13 has compatibility issues with langchain and numpy(internal dependency)
- **Ollama** installed and running.
- A local model:
    - ollama pull smollm:1.7b

- agents_demo.py (file should have two agents + finalizer, strict JSON, no domain hardcoding)

Questions:

1. **Set up** a small local LLM with Ollama.(use smollm:1.7b)
2. **Write and Run** a Python script that creates two agents (Planner, Reviewer) and a finalization step.
3. **Capture** outputs and submit a PDF with your console screenshots + answers.

What you should see (format, not exact text):

- Three tags like ["vector clocks","partial ordering","conflict resolution"]
- A one-sentence summary (≤25 words), and a Planner/Reviewer transcript.
- A final **Publish output** JSON printed to console.

Deliverables

1. **Command used** (the exact python agents_demo.py line).
2. **Console screenshot(s)** showing:
    - Planner output
    - Reviewer output
    - Finalized Output
    -
3. **Short answers (1–2 lines each):**
    - Q1. Write the 3 final tags you obtained.
    - Q2. Paste the final summary (≤25 words).
    - Q3. Did the Reviewer agent change anything? (yes/no + explanation)

Place the command, screenshots, and answers together — **code snippet above, screenshot below** (same pattern as HW1).

4. **Explanation** of the each step in your own words

Reading Materials:
https://ollama.readthedocs.io/en/quickstart
https://huggingface.co/HuggingFaceTB/SmolLM-1.7B
https://python.langchain.com/api_reference/ollama/chat_models/langchain_ollama.chat_models.ChatOllama.html#langchain_ollama.chat_models.ChatOllama