Task 15: Build a Mini Single-Page Application (SPA) Using Vanilla JavaScript

Tools:

- Primary: VS Code, JavaScript (Vanilla), Browser
- Alternatives: CodePen, StackBlitz

Hints / Mini Guide:

1. Create a basic HTML file with a navigation menu and a main content container where page content will be dynamically injected.
2. Design multiple "views" (Home, About, Contact) as JavaScript functions or template strings instead of separate HTML files.
3. Add click event listeners to navigation links to prevent default page reload behavior.
4. Use JavaScript to update the content container dynamically based on user navigation.
5. Implement browser history handling using the History API (pushState, popstate) to maintain URL changes.
6. Ensure page state remains consistent when the user uses browser back and forward buttons.
7. Modularize your JavaScript code by separating routing logic and view rendering logic.
8. Add basic loading indicators during view switching to simulate real application behavior.
9. Thoroughly test navigation, refresh behavior, and browser history functionality.

Deliverables:

- Fully functional mini SPA without page reloads
- Clean routing and state handling using Vanilla JS

Interview Questions Related To Above Task:

- What is a Single-Page Application?
- How does the History API work?
- Difference between SPA and MPA?
- What is client-side routing?
- Why are SPAs faster?

📌 **Task Submission Guidelines**

- ⏰ **Time Window:**

Youcan complete the task anytime between 10:00 AM to 10:00 PM on the given day. Submission link closes at 10:00 PM

- 🔍 **Self-Research Allowed:**

Youare free to explore, Google, or refer to tutorials to understand concepts and complete the task effectively.

- 🛠️ **Debug Yourself:**

Tryto resolve all errors by yourself. This helps you learn problem-solving and ensures you don't face the same issues in future tasks.

- 💸 **No Paid Tools:**

Ifthe task involves any paid software/tools, do not purchase anything. Just learn the process or find free alternatives.

- 📁 **GitHub Submission:**

Create a new GitHub repository for each task.
Add everything you used for the task — code, datasets, screenshots (if any), and a short README.md explaining what you did.

📤 **Submit Here:**

After completing the task, paste your GitHub repo link and submit it using the link below:

- 👉 [Submission Link]