🌐 AI-Driven Development — 30-Day Challenge

🗁 Part A — Theory (Short Questions)

1. Nine Pillar Understanding

- **Why is using AI Development Agents (like Gemini CLI) for repetitive setup tasks better for your growth as a system architect?**

By using AI Developments Agents (like Gemini CLI) for repetitive setup task, we can just focus on high level problem not just getting burn-out or tired by setting low level problem or setup tasks. We can focus on the bigger picture, this help us not to get tired before working on a project.

- **Explain how the Nine Pillars of AIDD help a developer grow into an M-Shaped Developer.**

The Nine Pillars of AIDD usually refers to as (Designing, Testing, Documentation, Coding, Deployment etc.). In past learning a second skill took years and now with AIDD you can learn in quickly. Like if you are expert in backend and using AI to create an Fronted using a best practice of pillars. You can implement these pillars through AI while also learning new skill.

2. Vibe Coding vs Specification-Driven Development

- **Why does Vibe Coding usually create problems after one week?**

The Vibe Coding fails or create problems after one week is because of proper planning. While you are vibe coding you doesn't create plan and create function on top of the other function in this way your code became a mess and if the problem or error is faced you and the AI doesn't know **where** and **how** to fix these error as AI Context Window has a limit and after some time AI doesn't remember that implementation it had done before.

- **How would Specification-Driven Development prevent those problems?**

Spec Driven Development is the writing the detail instruction in text of the project (or SPEC) before writing a code.

You can avoid these problem as this detailed instruction act as a “blueprint” of your project. When you provide the blueprint of the code to the AI then it will know where to implement your code or function properly. And if there is an error it can be fixed without tangling the code structure. A good spec insure weather if you start over it will give you the best result.

3. Architecture Thinking

- **How does architecture-first thinking change the role of a developer in AIDD?**

AIDD Changes the role of a Developer from **“Typist”** to **“Architect”**. You are no longer needed to worry about writing the loop of code. You are just needed to focus on end-result “What the loop is achieving (logic)”.

- **Explain why developers must think in layers and systems instead of raw code.**

Thinking about the raw code is fragile: If you think only about the raw code you will get lost. You may focus on writing a single function while the whole code structure is wrong. You must view you application as an interacting layers that are interacting which each other (e.g User interface layer -> Business logic layer -> Data Layer). You must provide the architecture of your application how different layer or component are interacting with each other. Otherwise, AI will get lost and just provide the piece of code which is not interacting with anything.

🗁 Part B — Practical Task (Screenshot Required)

CLI PROMPT

Write a 1-paragraph technical specification for an email validation function. It must require an '@' symbol, validate the domain extension (e.g., .com), and specify that the function returns distinct error messages for invalid inputs.

Specification Generated By CLI

✦ The validate_email function will analyze a given input string to confirm it adheres to fundamental email format requirements. It will first verify the presence of exactly one "@"

symbol, which is essential for separating the local part from the domain. The function will then inspect the domain portion to ensure it contains at least one period (.) and that the top-level domain (e.g., "com", "org") following the final period is not empty. For any validation failure, the function will return a distinct, descriptive error string indicating the specific issue, such as "Missing '@' symbol" or "Invalid domain extension." If the email string passes all validation checks, the function will return a boolean True.

🗀 Part C — Multiple Choice Questions

Corrected Answers

1. B. Clear requirements before coding begins
2. B. Thinking in systems and clear instructions
3. B. Architecture becomes hard to extend
4. B. Handle repetitive tasks so dev focuses on design & problem-solving
5. C. Deep skills in multiple related domains

☑ Submitted By:

Name: Muhammad Jawad Ullah

Roll no: 00282950

Class Slot: Friday (6 pm – 9 pm)