

# AIDD 30-Day Challenge — Task 2

## Part A — Theory (Short Questions)

### 1. Nine Pillars Understanding:

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

Because AI Agents work as our coding assistants, doing all repetitive tasks, that reduces our work load and saves a major part of our time that was spent writing code, reading documentations, debugging cryptic errors etc and lets us focus more on creativity, design, problem-solving and making better architectural decisions rather than memorizing patterns which makes us think and grow as a system architect.

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

Each of Nine Pillars of AIDD reduce the barriers that people faced earlier, making it easier to transform to an M-Shaped developer because it's no longer needed to memorize patterns and commands, following different standards and protocols for different tools etc. These pillars help a developer become an M shaped developer and handle work of multiple domains efficiently, which earlier required different specialists.

Using all these pillars we can have assistance in any domain just in our terminal and IDE (pillar 1 AI CLI & Coding Agents and pillar 4 AI-First IDEs), and re-usable domain-specific skills (pillar 8 Composable vertical skills), so, by writing clear specs (pillar 7 Spec Driven Development) in natural language (pillar 2 Markdown as Programming Language) and using the standard tools (pillar 3 MCP Standard) when needed we can ship and deploy (pillar 9 Universal Cloud Deployment) solutions that are well tested (pillar 6 Test-Driven Development) which won't break during production, works on every os (pillar 5 Linux Universal Dev Environment).

## **2. Vibe Coding vs Specification-Driven Development**

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

Because the code is not well designed and doesn't follow a standard pattern for that project. It was just coded without a proper plan. It's not even tested for different edge cases so it could break anytime.

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

It prevents it by following an opposite order of vibe coding, we write clear specifications first for a project, then plan its architecture and design followed by breaking it into executable tasks, then we later implement those tasks along with tests that cover different edge cases. In the end we have a well structured project with proper documentation which can be maintained as well.

## **3. Architecture Thinking**

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

Before architecture-first thinking developers used to implement their idea to code first, now they have to write their intent into clear specifications that their AI Agent can understand and implement as code, that's how it changed the role of a developer as typist to spec-engineer or orchestrator.

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

Because the role of developers has evolved from code writers to orchestrators who design systems with AI as their coding assistant. So they must think more about layers and systems than raw code, like which models would be best for their work and how they're going to develop their project with AI.

## Part B — Practical Task (With Screenshot)

### Task:

Using any AI CLI tool, generate a 1-paragraph specification for an email validation function.

### Requirements:

- Must contain “@”
- Must contain a valid domain (e.g., .com, .org)
- Should return clear error messages

*My CLI prompt and the 1-paragraph specification generated by the CLI ( screenshot):*



```


Tips for getting started:
1. Ask questions, edit files, or run commands.
2. Be specific for the best results.
3. /help for more information.

> generate a 1-paragraph specification for an email validation function. It must contain “@”, a valid domain (e.g., .com, .org), and should return clear error messages

♦ The email validation function shall accept a string input representing an email address and verify its adherence to standard email format requirements. Specifically, it must ensure the presence of a single “@” symbol separating the local part from the domain, and that the domain includes a top-level domain (TLD) such as “.com”, “.org”, “.net”, etc., preceded by at least one subdomain component. The function should return True for valid email addresses and False for invalid ones, accompanied by a clear, descriptive error message indicating the specific reason for validation failure (e.g., “Missing ‘@’ symbol”, “Invalid domain format”, “Missing TLD”).


Using: 1 GEMINI.md file | 10 MCP servers

>  Type your message or @path/to/file

```

## Part C —Self-Test (MCQs)

1. What is the main purpose of Spec-Driven Development?

- A. Make coding faster
- B. Clear requirements before coding begins 

- C. Remove developers
- D. Avoid documentation

**2. What is the biggest mindset shift in AI-Driven Development?**

- A. Writing more code manually
- B. Thinking in systems and clear instructions ✓
- C. Memorizing more syntax
- D. Working without any tools

**3. Biggest failure of Vibe Coding?**

- A. AI stops responding
- B. Architecture becomes hard to extend ✓
- C. Code runs slow
- D. Fewer comments written

**4. Main advantage of using AI CLI agents (like Gemini CLI)?**

- A. They replace the developer completely
- B. Handle repetitive tasks so dev focuses on design & problem-solving ✓
- C. Make coding faster but less reliable
- D. Make coding optional

**5. What defines an M-Shaped Developer?**

- A. Knows little about everything
- B. Deep in only one field
- C. Deep skills in multiple related domains ✓
- D. Works without AI tools

***Submitted By: Fatima Faisal***