

**SOBIA RAO**

**CLASS SLOT: FRIDAY – 6:00 PM TO 9:00 PM**

## **AI-DRIVEN DEVELOPMENT – 30-DAY CHALLENGE – TASK 2**

### **Reflection Activity:**

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?

**English**

Using AI agents for repetitive setup lets you focus on system design, strategic decisions, and problem-solving, helping you grow as a system architect instead of getting stuck in manual tasks.

**(Roman Urdu)**

AI agents ko bar-bar hone wale setup ke kaamон ke liye use karne se aap apna waqt aur energy important kaamон jaise system design aur maslay hal karne par laga sakte hain, aur aap as a system architect behtar ban sakte hain, bajaye ke simple kaamон mein waqt zaya karein.

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

**English**

The Nine Pillars of AIDD help a developer become M-Shaped by automating repetitive tasks, improving system design and planning, building deep expertise in key areas, and broadening skills across AI, tools, and cloud. This lets the developer focus on high-value problem-solving, strategic decisions, and integration, combining depth and breadth in their skills.

**(Roman Urdu)**

Nine Pillars of AIDD developer k M-Shaped bane nee m help kart e ha kinky ye repetitive tasks automate kart e ha system design or planning improve kart e h key areas me deep expertise develop karate ha, or AI, tools, or cloud k skills broaden kart h. Is se developer high-value problem-solving, strategic decisions, aur integration par focus kar sakta hai, jisse unki skills mein depth or breadth dono aati hain.

## 2. Vibe Coding vs Specification-Driven Development

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

**English:**

Vibe Coding causes problems after a week because it lacks discipline. Quick AI-generated code may work at first but builds up errors and technical debt, leading to bugs and maintenance issues.

**Roman :**

Vibe Coding problems tab create hotein hain kyunki is me discipline nah hoti. AI-generated code shuru mein kaam karta hai, lekin errors aur technical debt jama ho jate hain, jisse bugs aur maintenance issues aate hain.

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

**English:**

Spec-Driven Development prevents problems by using clear specifications and structure, keeping AI-generated code consistent, tested, and error-free.

**Roman:**

Spec-Driven Development problems prevent karta hai kyunki ye clear specifications aur structure provide karta hai, jisse AI-generated code consistent, tested, aur error-free rehta hai.

## 3) Architecture Thinking

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

## **English:**

Architecture-first thinking shifts a developer's role from writing every line of code to designing the system, making strategic decisions, and orchestrating AI to do the mechanical work. Developers focus on problem understanding, quality, and integration, increasing their impact and value.

## **Roman:**

Architecture-first thinking developer ka role change karta hai ab wo har line of code nahi likhta, balk system design, strategic decisions, or AI ko tasks karne ke liye orchestrate karta hai. Developer problem samajhne, quality aur integration par focus karta hai, jisse unka impact aur value barhta hai.

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

## **English:**

Developers must think in layers and systems to maintain architecture, modularity, and scalability, focusing on design and problem-solving instead of just coding.

## **Roman:**

Developers ko layers aur systems mein sochna chahiye taake architecture, modularity, aur scalability maintain ho aur wo design aur problem-solving par focus kar saken.

## **Part B — Practical Task (Screenshot Required)**

### **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

### **Submission:**

- 1 Your exact CLI prompt (text or screenshot)
- 2 The 1-paragraph specification generated by the CLI

```

Gemini.md • image.png
↓ Gemini.md
8

PROBLEMS OUTPUT DEBUG CONSOLE PORTS TERMINAL GITLENS
node + v 🗑️ ... | ⌂ x

checks for an "@" symbol and a valid-appearing domain, and then provides clear validation messages. I'll propose this plan, get approval, implement it, and then demonstrate its functionality with both valid and invalid emails. I'm now presenting this plan to the user.

i Update successful! The new version will be used on your next run.

> "Generate a 1-paragraph specification for a Python email validation function. The function must check that the input contains '@' and a valid domain (like .com or .org), and it should return clear error messages if invalid."
+ I'll create a Python function to validate email addresses. It will check for a single '@' symbol and a domain with a TLD of at least two characters (e.g., '.com'). Valid emails return True; invalid ones raise a ValueError with a clear error message, such as 'Missing @ symbol' or 'Invalid domain format'.

```

## generate a 1-paragraph specification

```

File Edit Selection View Go Run ...
Gemini.md • task-cli
↓ Gemini.md
1 Generate a 1-paragraph specification for an email validation function.
2 Requirements:
3 - Must contain "@"
4 - Must contain a valid domain (e.g., .com, .org)
5 - Should return clear error messages
6

PROBLEMS OUTPUT DEBUG CONSOLE PORTS TERMINAL GITLENS
node + v 🗑️ ... | ⌂ x

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

+ Okay, the user wants an email validation CLI. My plan is to build a Python-based tool that takes an email as an argument, checks for an "@" symbol and a valid-appearing domain, and then provides clear validation messages. I'll propose this plan, get approval, implement it, and then demonstrate its functionality with both valid and invalid emails. I'm now presenting this plan to the user.

i Update successful! The new version will be used on your next run.

> Type your message or @path/to/file
~\Desktop\task-cli (main*)
no sandbox (see /docs)
auto

```

## CORRECT ANSWERS MCQS

- 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)Deep skills in multiple related domains ↗

## Reflection

AI-Native development shifts developers from just writing code to **thinking in systems**, using **Spec-Driven Development (SDD)** and **AI agents** to plan, build, and maintain software efficiently, growing into **M-Shaped professionals** with multiple deep skills.