AI-Driven Development Task 2 (Official Submission)

By: Sarfaraz Ahmed
Roll Number: 00455501
Submitted to: Sir Hamzah Syed
Class Slot: Friday — 6 PM to 9 PM

PART A THEORY (Short Questions)

1. Nine Pillars Understanding

Q1: Why is using AI Development Agents for repetitive setup tasks better for your growth as a system architect?

The fact that AI Development Agents can do cyclical work helps you conserve mental power that would otherwise be drained on trivial tasks. Rather, you would be practicing higher level functions like design. Instead of busy work like templating, setting up an environment, or configuring files, you can focus on system architecture, decision making, and system structuring. You develop the instinct of an architect instead of that of a coder. The more you leave the agents to do the routine work, the more your skills get refined toward designing systems and orchestrating workflows, which is basically what a system architect does.

Q2: How do the Nine Pillars of AIDD help a developer grow into an M-Shaped Developer?

The Nine Pillars provide a complete ecosystem of tools specs, agents, tests, automation, workflows that reduce the friction of learning new domains. Since gaps in knowledge are covered by AI, developers can work in various domains (architecture, testing, DevOps, planning, and coding). Over time, this leads to building a strong foundation in many interrelated fields as opposed to a single specialization. This multi-domain proficiency forms an M-shaped developer who can function like an entire multidisciplinary team.

2. Vibe Coding vs Specification-Driven Development

Q1: Why does Vibe Coding usually create problems after one week?

Vibe Coding lacks organization, documented choices, and consistent strategies. After a couple of days, the initiative gets confusing. Features break, the overall logic becomes erratic, and the developers forget the rationale behind some of the previously made decisions. As there is no source of truth, corrective measures become extremely tiresome. This absence of clarity brings indescribable anguish as the project scales.

Q2: How would Specification-Driven Development prevent those problems?

Specification-Driven Development means that a detailed document gets completed before coding gets done. The documents act as a guiding star for the entire project. Each and every function, rule, and behavior is recorded for clarity, eliminating ambiguity. Having everything outlined at the start means that work is predictable, testable, and extensible. If more details get added over time, the original intention stays at the forefront, and over-organization is avoided.

3. Architecture Thinking

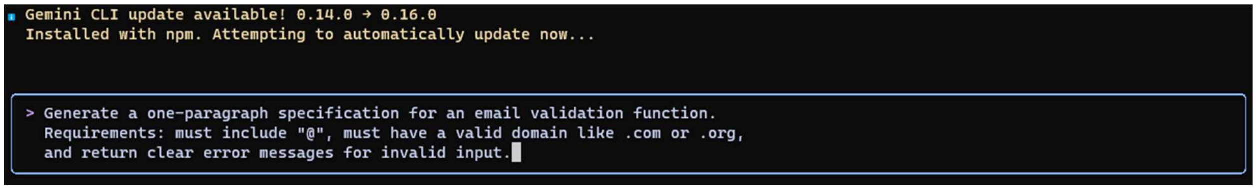
Q1: How does architecture-first thinking change the role of a developer in AIDD?

Thinking in an architecture-first manner means thinking beyond just coding. It means designing a system. Here, the developer describes the structure of the system. What the components of the system are, how do they interact? Which layers are there, and how does the layer's responsibilities are divided? AI tools do the rest and take care of most of the coding. Here, the developer pivoted into a more of a strategic position rather than just manual coding.

Q2: Why must developers think in layers and systems instead of raw code?

The layered system is scalable, maintainable, and, more importantly, easier for the AI agents to work through. Each layer has a purpose, separation of concerns, and predictable behavior. Thinking purely in raw code invites the system to tangle and become a mess of hard to extend structures. With layered system thinking, the project becomes a seamless, functioning entity, where every part is a well-oiled machine, and knows its role.

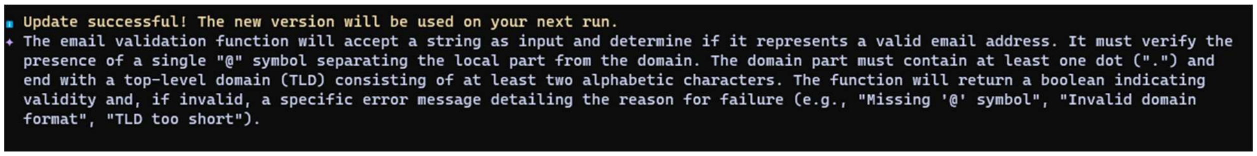
PART B — PRACTICAL TASK



```
Gemini CLI update available! 0.14.0 → 0.16.0
Installed with npm. Attempting to automatically update now...

> Generate a one-paragraph specification for an email validation function.
  Requirements: must include "@", must have a valid domain like .com or .org,
  and return clear error messages for invalid input.
```

2□ One-Paragraph Specification Generated by the AI CLI



```
▪ Update successful! The new version will be used on your next run.
✦ The email validation function will accept a string as input and determine if it represents a valid email address. It must verify the
  presence of a single "@" symbol separating the local part from the domain. The domain part must contain at least one dot (".") and
  end with a top-level domain (TLD) consisting of at least two alphabetic characters. The function will return a boolean indicating
  validity and, if invalid, a specific error message detailing the reason for failure (e.g., "Missing '@' symbol", "Invalid domain
  format", "TLD too short").
```

PART C — MULTIPLE CHOICE QUESTIONS

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**

Reflection

The move to AI-Driven development moves the developer up a level from coder to system designer." Things like Development Agents, Systems Development Discipline (Azure), and the Nine Pillars help us to think in systems instead of syntax. This enables them to scale as M-shaped professionals with strong competencies in several areas driven by intelligent automation, not shackled by manual labor.