AI-Driven Development — 30-Day Challenge

Task 2 — Official Submission

Name: Nida Khurram
Class Slot: Friday — 6:00 PM to 9:00 PM
Instructor: Sir Hamzah Syed
GitHub Repository: https://github.com/nida1khurram

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?
Using AI for boring, repetitive tasks (like project setup) is better because it frees up your time. Instead of doing the same simple things, you can focus on the bigger picture—like designing the whole system and solving harder problems. This is how you grow from a coder into an architect.

How do the Nine Pillars of AIDD help a developer grow into an M-Shaped Developer?
The Nine Pillars integrate tools like TDD, AI CLI, MCP, and agents into a cohesive system. By leveraging these pillars, developers gain expertise in multiple domains (e.g., testing, automation, system design) without deep manual effort in each. This bridges knowledge gaps and enables them to function as M-Shaped Developers—deeply skilled in 2–4 complementary areas.

2. Vibe Coding vs Specification-Driven Development

Why does Vibe Coding usually create problems after one week?
Vibe Coding relies on intuition and immediate context, leading to unstructured code. Over time, this lack of planning results in technical debt, inconsistent logic, and difficulties in extending features. After a week, the initial "vibe" fades, and the code becomes hard to maintain or debug.

How would Specification-Driven Development prevent those problems?
Specification-Driven Development requires clear, executable specs before coding begins. This formalizes intent, ensures alignment with requirements, and provides a roadmap for implementation. By prioritizing clarity and structure, SDD prevents ambiguity and technical debt, making the codebase scalable and maintainable.

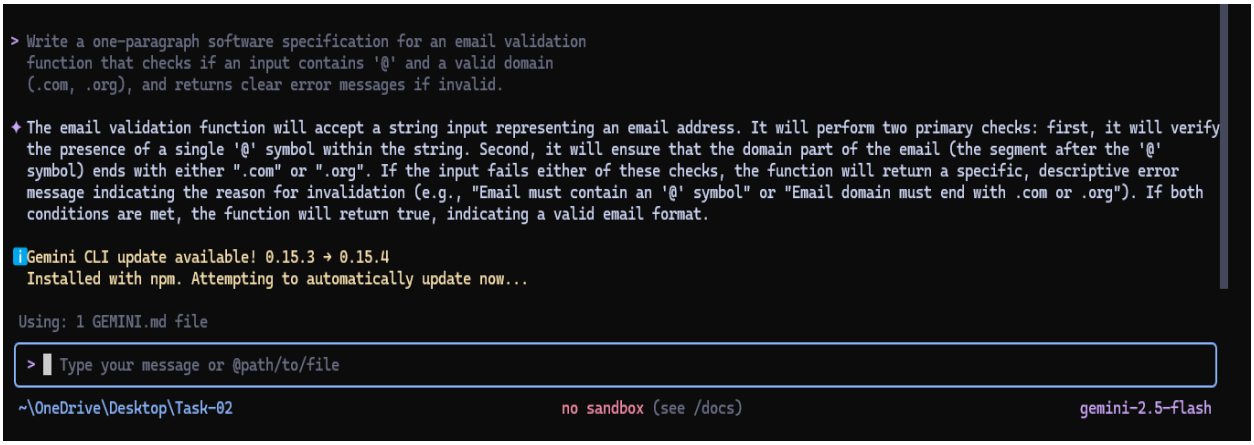
3. Architecture Thinking

How does architecture-first thinking change the role of a developer in AIDD?
Architecture-first thinking shifts the developer's focus from writing code to designing systems. Instead of reacting to immediate tasks, developers proactively plan modular, scalable solutions. This transforms them into system architects who orchestrate AI tools and agents to execute their vision.

Why must developers think in layers and systems instead of raw code?
Thinking in layers and systems ensures separation of concerns, modularity, and scalability. Raw code alone cannot address complex workflows or long-term maintenance. By designing systems, developers create resilient architectures that adapt to change and integrate seamlessly with AI-driven tools.

Part B — Practical Task (Screenshot Required)

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



Part C — Multiple Choice Questions

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

B. Clear requirements before coding begins ☑

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

B. Thinking in systems and clear instructions ☑

3. Biggest failure of Vibe Coding?

B. Architecture becomes hard to extend ☑

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

B. Handle repetitive tasks so dev focuses on design & problem-solving ☑

5. What defines an M-Shaped Developer?

C. Deep skills in multiple related domains ☑