Prepared by: Umama
Roll number: 00438580
Class Timing: 06 to 09 PM (Friday)

(AIDD) AI Driven Development 30 Days Challenge - Task 02

📁Part A — Theory (Short Questions)

1. Nine Pillars Understanding

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

AI development agents like Gemini CLI automatically handle repetitive and boring tasks (such as boilerplate code, configuration setup, and documentation), which saves our time and mental energy. This allows us to focus on high-level system design, architecture patterns, scalability planning, and strategic decision-making. It helps us rise above implementation details and think about the “why” and “what,” which is the mindset of a true system architect. In this way, we don’t just write code we learn to design resilient and scalable systems.

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

The Nine Pillars of AIDD (Prompt Engineering, AI-Assisted Coding, Automated Testing, Documentation, Code Review, Learning, Problem-Solving,) help developers build expertise across multiple domains. An M-Shaped Developer is someone who has deep expertise in 2–3 areas while maintaining broad knowledge in all others. These pillars ensure that you don’t become skilled in just one thing but instead become versatile learning how to use AI tools, perform testing, write documentation, and more.

2. Vibe Coding vs Specification-Driven Development

1.Why does Vibe Coding usually create problems after one week?

Vibe Coding means writing code based on vibe without any planning, documentation, or clear thought process. You just do whatever seems right in the moment. After a week, problems start appearing because:
1.You forget what the code was doing and why.
2.There’s no clear structure or pattern, so the code becomes messy.
3.Requirements aren’t remembered, and important bugs get missed.
4.Other developers can’t understand what’s going on in code.
5. Testing and debugging become difficult because there’s no documented behavior or logic.
Basically, what started as a temporary quick fix turns into a permanent problem.

2.How would Specification-Driven Development prevent those problems?

In Specification-Driven Development (SDD), you first write clear specifications meaning you document everything before writing any code: what needs to be built, how it should behave, what the inputs and outputs are.

This approach prevents all the problems that come from Vibe Coding because:

1.Specifications provide a clear roadmap, so you never forget what you were supposed to build.
2. Requirements are documented, which makes testing and validation much easier.
3. Every team member understands how the system works, because the behavior is written down.

4. Maintenance becomes simple, since the expected behavior is already documented.
5. You can understand the impact of changes by reviewing the specifications before modifying anything.

3. Architecture Thinking

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

Architecture-first thinking a developer from a code writer into a solution designer. Now the developer first creates the system's blueprint how components will connect, how data will flow, and how the system will scale and then uses AI tools to handle the implementation.In traditional development, people used to start coding right away. Now, planning and structure come first. The developer makes strategic decisions (design patterns, database schema, API design) while AI handles the repetitive coding tasks. This increases the developer's value because they become a problem solver and a system thinker who can translate business needs into technical solutions.

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

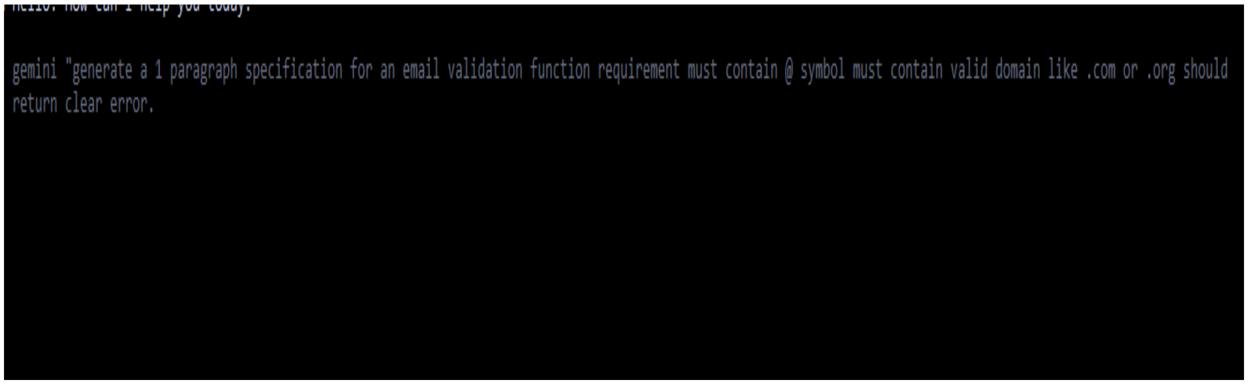
Thinking in layers (UI, Logic, Data, Infrastructure) ensures that each part works independently, testing becomes easier, and changes stay isolated. Systems thinking helps you understand how components interact, what the dependencies are, and where potential failures might occur.

🗂️ Part B — Practical Task (Screenshot Required)

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

Prompt Text: gemini Generate a 1-paragraph specification for an email validation function. Requirements: must contain @ symbol, must contain valid domain like .com or .org, should return clear error messages.

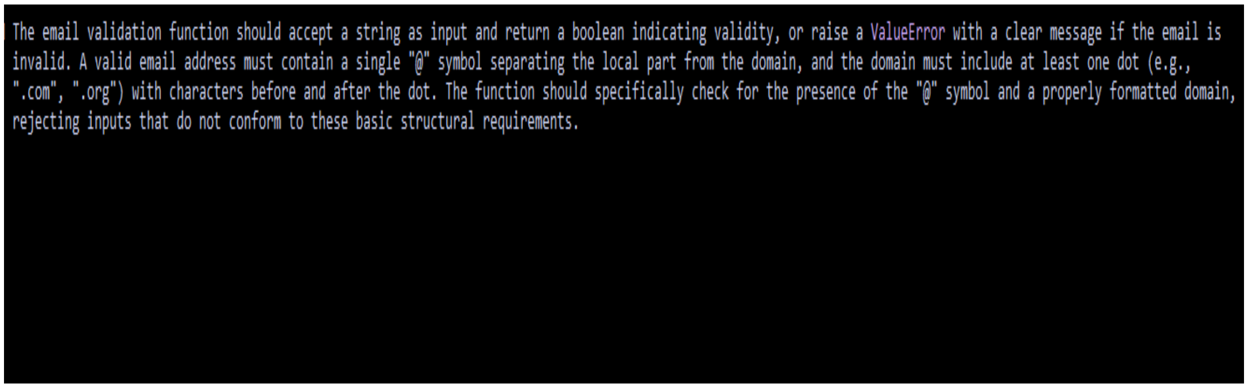
Prompt ScreenShot:



```
Hello! How can I help you today?

gemini "generate a 1 paragraph specification for an email validation function requirement must contain @ symbol must contain valid domain like .com or .org should
return clear error.
```

Prompt result:



```
The email validation function should accept a string as input and return a boolean indicating validity, or raise a ValueError with a clear message if the email is
invalid. A valid email address must contain a single "@" symbol separating the local part from the domain, and the domain must include at least one dot (e.g.,
".com", ".org") with characters before and after the dot. The function should specifically check for the presence of the "@" symbol and a properly formatted domain,
rejecting inputs that do not conform to these basic structural requirements.
```

The email validation function should accept a string as input and return a boolean indicating validity, or raise a ValueError with a clear message if the email is

invalid. A valid email address must contain a single "@" symbol separating the local part from the domain, and the domain must include at least one dot (e.g.,
".com", ".org") with characters before and after the dot. The function should specifically check for the presence of the "@" symbol and a properly formatted domain,
rejecting inputs that do not conform to these basic structural requirements.

📁*Part C — Multiple Choice Questions*

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