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 Tools for repetitive setup work is really helpful because it saves time, reduces errors and let us focus on designing the system as whole, this way we learn to plan, organize and think like a system architect instead of getting stuck in small tasks.

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

the Nine Pillars of AIDD teach develop many important skills like designing effective prompts for tasks measures AI performance and reliability testing and launching projects help in handling organizing data design pipe line that combines multiple tools/models these pillars help developer understands more think bigger so M shaped developers deep in some area broad in many.

2. Vibe Coding vs Specification-Driven Development

Why does Vibe Coding usually create problems after one week?

vibe coding usually gives problems after a week because after week new topics include and old exercise feel difficult if they were not practiced and understood properly this is help in learning new ideas and not forget what you have learned.
How would Specification-Driven Development prevent those problems?
Specification-Driven Development stop problems by planning everything before coding so the code is organized easy to fix easy to understand it helps to avoid problems later even after a week.

3. Architecture Thinking

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

with architecture first thinking developers spend more time plaining the system instead of just writing codes they decide how parts of systems connect make sure everything work well and think about future growth this make them like system designer while AI handle boring and repetitive coding task.

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

developers should think in layers and systems instead of raw code because software is made of many connected parts. By organizing code into layers like UI, logic, and data, it becomes easier to understand, fix, and improve. This also helps in adding new features without breaking anything and makes it simpler for team members to work together. Thinking this way makes your code cleaner, stronger, and ready to grow.

MCQs
Q1) B. Clear requirements before coding begins
Q2) B. Thinking in systems and clear instructions
Q3) B. Architecture becomes hard to extend
Q4) B. Handle repetitive tasks so dev focuses on design & problem-solving
Q5) C. Deep skills in multiple related domains