Part A — Theory (Short Questions)

Q.1 Why is using AI Development Agents (like Gemini CLI) for repetitive setup tasks

better for your growth as a system architect?

A. AI agents repetitive aur boring kaam jaise setup, installation aur boilerplate handle kar lete hain, is liye developer ka time bach jata hai. Jab yeh sab AI kar deta hai to developer ka focus architecture, logic aur planning par reh jata hai. Repetitive tasks AI karne se human mistakes bhi bohot kam ho jati hain, aur AI developer ko sirf coder nahi balkay ek system architect ki tarah sochne me help karta hai.

Q.1B Explain how the Nine Pillars of AIDD help a developer grow into an M-Shaped Developer.

A. Nine Pillars basically aik complete system hota hai jisme tools aur processes dono shamil hote hain. Ye developer ko coding, architecture, testing, agents, CLI aur specifications jaise important kaam sahi tareeke se karna sikhata hai. Is process se developer ek hi field tak limited nahi rehta balkay multiple domains me strong ho jata hai, jise M-Shaped Developer kehte hain. Nine Pillars ki wajah se developer ki speed, understanding aur overall capability bohot improve ho jati hai.

Q.2 Why does Vibe Coding usually create problems after one week?

A. Vibe coding me developer bina planning ke bas andazay se code likhta jata hai, is liye kuch din baad code messy aur confuse ho jata hai. Phir changes karna mushkil ho jata hai aur bugs barh jate hain, kabhi kabhi developer ko khud samajh nahi aata ke usne kya likha tha. Ek haftay ke baad system unstable lagne lagta hai kyun ke koi proper direction nahi hoti.

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

A. Specification-Driven Development me pehle clear instructions likh liye jate hain ke system ko kya karna hai, is liye developer bilkul focused rehta hai. Har feature ka scope clear hota hai to mistakes kam hoti hain aur future me changes krna buht easy ho jata hai. SDD code ko clean, organized aur maintainable bana deta hai, iss tarah vibe coding wali problems paida hi nahi hoti.

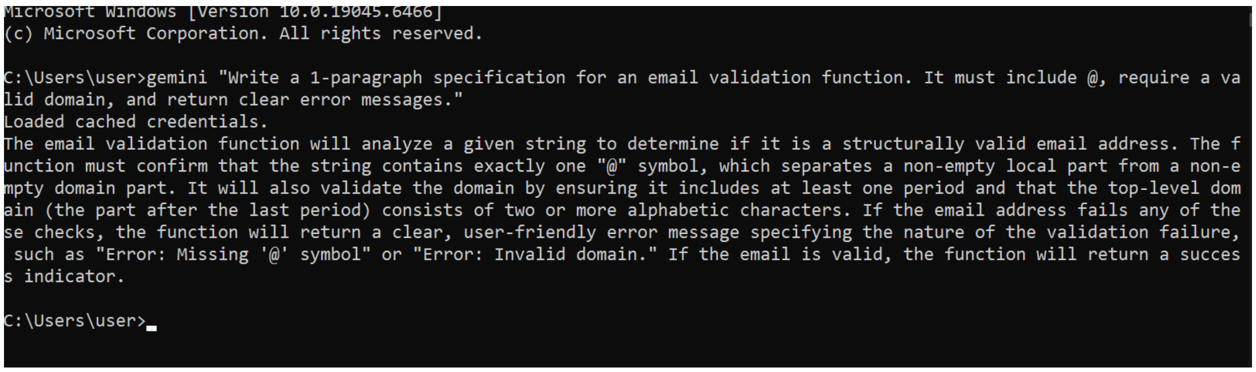
Q3. How does architecture-first thinking change the role of a developer in AIDD?

Architecture-first thinking developer ko sirf code likhne wale insaan se utha kar ek proper system designer bana deta hai. Jab developer pehle pura structure, flow aur system ka design sochta hai to uski approach zyada mature ho jati hai. Is tarah developer ka role sirf coding tak limited nahi rehta, balkay wo planning, decision-making aur overall system ki quality ka zimmedar ban jata hai.

Q3B. Explain why developers must think in layers and systems instead of raw code.

A. Layers me sochne se system clear blocks me divide ho jata hai jisse samajhna, manage karna aur update karna buht easy hota hai. Har layer ka apna role hota hai, jese models, IDEs aur agents, aur ye separation system ko stable aur organized rakhta hai. Jab developer layers me sochta hai to complex software bhi simple lagta hai aur problems ko identify krna buht asaan hota hai.

Part B (CLI prompt + output paragraph)



```
Microsoft Windows [Version 10.0.19045.6466]
(c) Microsoft Corporation. All rights reserved.

C:\Users\user>gemini "Write a 1-paragraph specification for an email validation function. It must include @, require a va
lid domain, and return clear error messages."
Loaded cached credentials.
The email validation function will analyze a given string to determine if it is a structurally valid email address. The f
unction must confirm that the string contains exactly one "@" symbol, which separates a non-empty local part from a non-e
mpty domain part. It will also validate the domain by ensuring it includes at least one period and that the top-level dom
ain (the part after the last period) consists of two or more alphabetic characters. If the email address fails any of the
se checks, the function will return a clear, user-friendly error message specifying the nature of the validation failure,
 such as "Error: Missing '@' symbol" or "Error: Invalid domain." If the email is valid, the function will return a succes
s indicator.

C:\Users\user>
```

Part C: MCQs

Question: 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

Answer: B Kyun k SDD ka main maqsad hai pehle clear instructions/specifications likhna before coding.

Question: 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

Answer: B AI-Driven Development me developer system aur instructions pe focus karta hai, na ke bas code likhne pe.

Question: Biggest failure of Vibe Coding?
A. AI stops responding
B. Architecture becomes hard to extend ✓
C. Code runs slow
D. Fewer comments written

Answer: B Vibe coding me planning nahi hoti, isliye architecture messy aur extend karna mushkil ho jata hai.

Question: 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

Answer: B CLI agents repetitive tasks handle karte hain, developer focus design aur problem solving pe rakhta hai.

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

Answer: C M-Shaped Developer multiple domains me deep knowledge rakhta hai, na ke sirf ek field me.