AI-Driven-Development

AI-Driven Development_30-Days Challenge

Day Two 🚀

Student Name: Asma Akbar
Instructor: Sir Hamzah Syed
Class: Friday — 6:00 PM to 9:00 PM
Date: November 17, 2025

Submission Date: November 18, 2025
Status: Completed

📁Part A: Theory

1. Nine Pillars Understanding

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

AI agents wo saaray boring setup waale kaam khud kar detay hay jis ko kary main hamy boht time lgta tha or is liye ab hmra dimagh fresh rehta hai . Phir ham asaani se system ki bari soch aur planning par focus kar sakte ho. isi wajah se ham system architect ki tarah jaldi grow kartay hain

Jab AI basic configuration aur setup handle kar leta hai, to ham architecture design, scalability, security aur problem-solving jaisi high-level skills improve kar sakte hain. Is tarah ham tez seekhte hain, zyada productive hote hain aur complex systems ko samajhne aur design karne ka mauqa milta hai, jo long-term growth ke liye bohot important hota hai.

Example: bilkul aise jaise ghar aatay hi khnna pkaa howa milta hai ab aap araam se apna kaam plan kar lete ho.

(b) Explain how the Nine Pillars of AIDD help a developer grow into an M-Shaped Developer?

AIDD ke 9 pillars aap ko coding ke saath sochne ka tareeqa bhi sikhate hain, is liye banda aik hi waqt me kai skills me strong hota. hai dheere dheere banda code bhi samajhta hai aur system ka dimaag bhi samajhne lagta hai (M-shaped).

Example: jaise aik student maths, English aur sports teeno me acha ho sirf aik cheez me expert nahi.

Nine Pillars of AIDD aik developer ko sirf code karne wale insaan se badal kar ek “M-shaped” developer bana dete hain matlab wo na sirf technical expert hota hai, balkay system architect, design thinker aur agent-orchestrator bhi banta hai. In pillars mein specification-driven development, AI-augmented workflows, agent orchestration, test aur quality gate, version control, human verification, iterative refinement, embedded documentation, aur production deployment shamil hain.

2. Vibe Coding vs Specification-Driven Development

Vibe Coding vs Specification-Driven Development

(a) Why does Vibe Coding usually create problems after one week?

Vibe coding aam tor par ek haftay baad is liye maslay paida karta hai q kay Vibe coding me banda bas mood se code likhta jata hai, is liye aik haftay baad code samajh hi nahi aata. Cheezein bikhar jati hain aur phir chhoti si change bhi mushkil lagti hai.

> **Example:** jaise aap bina plan ke jaldi jaldi room saaf karein pehle theek lagta hai, ek hafta baad phir ganda lagne lagta hai

(b) How would Specification-Driven Development prevent those problems?

Specification-driven development me pehle sab kuch clear hota hai kya banana hai aur kyun banana hai. is liye code saaf, stable aur baad me handle karna bohat aasaan rehta hai.

> **Example:** jaise shopping list bana ke market jao—phir na time waste hota hai aur na galat cheez uthti hai.

Specification-Driven Development in maslon ko is liye rokh deta hai kyunke is mein coding shuru karne se pehle clear plan, requirements aur steps likh liye jate hain. Jab sab kuch pehle se define ho, to code random ya vibe par nahi banta, balkay structured aur samajh aa jane wala hota hai. Is tarah bugs kam hotay hain, confusion nahi hoti, aur project ek haftay baad bhi smooth chal raha hota hai.

3. Architecture Thinking

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

Architecture-first soch developer ko coder se zyada sochne wala insan bana deti hai, banda pehle sochta hai system kaise chalega, phir code likhta hai aur is se us ka role zyada mature ho jata hai.

Architecture-first thinking AIDD mein developer ka role badal deta hai kyunke is approach mein developer sirf coder nahi rehta, balkay ek system architect ban jata hai jo pehle pura design, flow aur structure sochta hai. Is mein developer AI agents ko guide karta hai, clear instructions aur specs banata hai, aur AI implementation handle karta hai. Is tarah developer ka kaam zyada strategic, organized aur long-term soch wala ho jata hai, jisse behtar systems bante hain aur developer ki growth bhi tez hoti hai.

> **Example:** jaise koi ghar banana ho pehle naqsha banta hai, deewar baad me

(b) Explain why developers must think in layers and systems instead of raw code?

Developers ko layers aur system jaisi soch is liye rakhni chahiye kyun ke raw code sirf chhoti picture dikhata hai, jabke layers poora flow samjhati hain. Aise sochne se Systems thinking se aap samajh lete hain ke har hissa doosray hisse se kaise connect hota hai, jis se bugs kam hote hain, planning strong hoti hai, aur future updates smooth ho jate hain. Raw code soch sirf short-term kaam chalati hai, lekin layered aur system-level soch long-term success deti hai.

> **Example:** jaise mobile ko sirf screen dekh kar nahi samjha ja sakta andar battery, software, signals sab layers me mil kar kaam karte hain, tabhi phone chal pata hai.

📁Part A: Practical Task Solution

create a function validateEmail (Email)that first check whether the Email strinng contains the "@" character.Then esure if ends with a vaild domain such as '.com' o '.org' if the eail is missing "@" return the error message: Email must contains '@' if the doma is vaild, return : "Email must end with a vaild domainn like '.com' or '.org' if all condition are satisfied , return "vaild email"

I will create a Python function validateEmail that checks for the presence of '@' and a valid domain suffix.

```python
def validateEmail(email: str) -> str:
    """
    Validates an email string based on the presence of '@' and a valid domain.

    Args:
        email: The email string to validate.

    Returns:
        A string indicating the validation result:
        - "Email must contains '@'" if '@' is missing.
        - "Email must end with a vaild domainn like '.com' or '.org'" if the domain is invalid.
        - "vaild email" if all conditions are satisfied.
    """
    if "@" not in email:
        return "Email must contains '@'"

    if not (email.endswith(".com") or email.endswith(".org")):
        return "Email must end with a vaild domainn like '.com' or '.org'"

    return "vaild email"
```

📁Part C: MSQ's

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

- 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
