Part 1: Theoretical Analysis

Q1: Explain how AI-driven code generation tools (e.g., GitHub Copilot) reduce development time. What are their limitations?

- **Boilerplate automation**: Tools like GitHub Copilot generate repetitive code (e.g., data models, utility functions), speeding up development by saving manual typing.
- **Instant suggestions**: In-context real-time suggestions accelerate coding, help enforce consistent style, and guide toward correct API usage.
- **Learning from existing codebases**: Trained on massive open-source data, these tools recommend idiomatic or optimized patterns, reducing design iteration.

Limitations

- **Overreliance**: Overuse can degrade developers' skills and reduce understanding of what the code does, leading to debugging and maintenance issues.
- **Risk of insecure or buggy code**: Generated snippets can contain vulnerabilities; developers must review and test them thoroughly.

Q2: Compare supervised and unsupervised learning in the context of automated bug detection.

Aspect	Supervised Learning	Unsupervised Learning
Data	Needs labeled examples: "buggy" vs "clean" code	Uses unlabeled logs or metrics
Training Goal	Model learns to predict bug likelihood	Finds anomalies or unexpected patterns
Limitations	Requires extensive labelled datasets, limited to known patterns	Higher false positive rates, interpretation needed

Q3: Why is bias mitigation critical when using AI for user experience personalization?

- **Fairness and ethics**: Personalization must avoid discriminatory treatment (e.g., biased hiring recommendations, uneven pricing).
- **Trust and perceptions**: Users may feel alienated or manipulated if recommendations consistently favor or ignore certain groups.

- **Legal compliance**: Personalization systems must comply with laws on non-discrimination; bias leaks could lead to litigation or reputational harm.

Part 2: Case Study Analysis

Article: AI in DevOps: Automating Deployment Pipelines

Answer: How does AIOps improve software deployment efficiency? Provide two examples.

1. Test Optimization & Build-time Reduction

- By analyzing historical deployment logs, AI predicts risky deployments before they go live, triggering **canary rollouts** or **auto-rollback** on anomalies.
- Example: Netflix's Spinnaker platform uses AI to detect likely failures and rollback bad releases, minimizing downtime

2. Test Optimization & Build-time Reduction

AI prioritizes and selects tests likely to fail based on code changes, slashing build durations. Facebook's Sapienz tool handles test generation to reduce false positives.