Step 2: Theoretical Analysis

Q1: AI-Driven Code Generation (e.g., GitHub Copilot)

Answer:
AI-driven code generation tools like GitHub Copilot use large language models (LLMs) trained on public code repositories. They assist developers by autocompleting code snippets, generating functions, and offering suggestions, thereby **reducing development time**.

Benefits:

- Faster prototyping
- Reduces repetitive coding
- Enhances productivity for boilerplate code

Limitations:

- Can produce incorrect or insecure code
- May not follow project-specific standards
- Risk of code plagiarism if not reviewed

Q2: Supervised vs Unsupervised Learning in Bug Detection

Supervised Learning	Unsupervised Learning
Uses labeled data (bug/no bug)	Uses unlabeled data
Can predict specific bugs	Detects anomalies (potential bugs)
Example: Classifying bug types	Example: Finding outliers in logs
Needs historical labeled data	Doesn't require labeled data

Q3: Why Bias Mitigation is Critical in UX Personalization

Answer:
Bias mitigation ensures AI systems don't unfairly favor or disadvantage certain user groups. In UX personalization:

- **Without mitigation**, the system may over-personalize based on gender, location, or ethnicity.
- **Bias** can lead to exclusion, unfair recommendations, or unequal experiences.
- Tools like **AI Fairness 360** help audit and correct biased models.

Case Study: AIOps in Deployment

How AIOps improves efficiency:

- **Automates issue detection:** Identifies failures during deployment faster than humans.
- **Predictive scaling:** Adjusts server resources based on past usage patterns.

Examples:

1. **Automated rollback:** If an error occurs, AIOps can revert to a stable version automatically.
2. **Anomaly detection in pipelines:** Detects unusual delays or build failures and alerts developers.