**Part 1: Theoretical Analysis**

**1. Short Answer Questions**

**Q1: Explain how AI-driven code generation tools (e.g., GitHub Copilot) reduce development time. What are their limitations?**  
AI-driven code generation tools like GitHub Copilot reduce development time by providing real-time code suggestions, auto-completing functions, and generating boilerplate code. This helps developers focus on higher-level logic and reduces the need for repetitive coding tasks. These tools also assist in learning new frameworks by showing code patterns based on context.

However, limitations include:

* **Contextual errors**: AI may generate syntactically correct but logically flawed code.
* **Security risks**: Generated code might introduce vulnerabilities if not reviewed carefully.
* **Dependence**: Over-reliance can hinder learning or understanding core concepts.
* **Training bias**: Suggestions may reflect outdated or non-optimal practices.

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

* **Supervised learning** uses labeled datasets (e.g., code marked as buggy or clean). It learns patterns associated with bugs and can accurately classify new code if trained well. It's effective but requires large, well-labeled datasets.
* **Unsupervised learning**, on the other hand, identifies anomalies or unusual patterns in code without labeled data. It is useful for detecting unknown or novel bugs but may result in more false positives due to a lack of labeled guidance.

**Q3: Why is bias mitigation critical when using AI for user experience personalization?**  
Bias mitigation is critical because personalized AI systems rely on user data to make recommendations or adapt interfaces. If biases in the training data (e.g., gender, age, or cultural preferences) are not addressed, the system can reinforce stereotypes, exclude minority groups, or provide an unfair user experience. Ethical personalization must ensure fairness, inclusivity, and transparency to maintain trust and usability across diverse user bases.

**2. Case Study Analysis**

**Reading the article: AI in DevOps: Automating Deployment Pipelines.**  
**Answer: How does AIOps improve software deployment efficiency? Provide two examples.**

AIOps enhances software deployment efficiency by leveraging machine learning and big data analytics to automate and optimize various DevOps processes. It reduces human error, shortens feedback loops, and provides predictive insights that accelerate delivery.

**Example 1: Automated Root Cause Analysis (RCA)**  
AIOps can analyze system logs and telemetry data to pinpoint the root cause of deployment failures, drastically reducing diagnosis time and enabling faster rollbacks or fixes.

**Example 2: Intelligent Resource Allocation**  
AIOps tools can dynamically scale infrastructure based on predicted traffic or resource usage during deployment, optimizing performance and reducing downtime or cost overruns.