AI FOR SOFTWARE ENGINEERING REPORT

QUESTIONS

Q1: How do 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 automatically suggesting code snippets, functions, and even full algorithms based on comments and existing code context. These tools:

- Speed up repetitive coding tasks.
- Reduce boilerplate code writing.
- Help developers learn new syntax or libraries quickly.
- Improve prototyping by generating functional blocks instantly.

Limitations:

- Copilot may suggest incorrect or insecure code.
- It can generate solutions that don't match the project's context or style.
- Over-reliance may reduce deeper understanding of the code.
- It may include code derived from open-source examples with unclear licenses.

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

- **Supervised learning** uses labeled datasets where bugs are clearly identified (e.g., "bug" or "no bug"). In automated bug detection, this approach is used to train models to recognize known patterns of buggy code or issue reports. For example, a model can learn to classify code snippets as "buggy" or "clean" if trained on historical bug-labeled codebases.
- **Unsupervised learning** works without labeled data. In bug detection, it can be used to detect **anomalies**—code or behavior that deviates from

the norm. For instance, clustering algorithms can group similar functions together and flag outliers that may represent potential bugs.

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

Bias mitigation is crucial in AI-driven personalization because biased models can lead to **unfair, exclusive, or even discriminatory user experiences**. For example, if a recommendation system favors one user group over another based on biased training data, some users may consistently receive less relevant or lower-quality suggestions.

Key reasons to mitigate bias:

- To ensure **fair access** to features and content for all users.
- To prevent reinforcing **social, gender, or cultural stereotypes**.
- To comply with ethical standards and **regulatory frameworks** like GDPR or AI ethics principles.
- To build **user trust** and maintain long-term engagement.

Case Study: AIOps (AI in DevOps)

Prompt: Read the article *"AI in DevOps: Automating Deployment Pipelines"*.
Question: How does **AIOps** improve software deployment efficiency? Provide **two examples**.

✍️ **Suggested Answer:**

AIOps (Artificial Intelligence for IT Operations) enhances software deployment by automating the detection, diagnosis, and resolution of issues within the CI/CD pipeline. It improves efficiency through **predictive insights**, **anomaly detection**, and **automated response mechanisms**.

🔍 **Key Benefits:**

1. **Faster Issue Resolution:**
   AIOps tools can monitor deployment logs and detect failures or slowdowns in real time, allowing teams to fix problems faster than with manual monitoring.
2. **Proactive Resource Management:**
   AI models can predict traffic spikes and scale resources (e.g., containers or services) accordingly—preventing downtimes or crashes during deployment.

✅ **Two Real-World Examples:**

1. **Predictive Rollbacks:**
   AIOps platforms like Dynatrace or Moogsoft can automatically detect performance degradation during deployment and **roll back to a stable version** before users are affected.
2. **Smart Alerting in CI/CD:**
   Instead of flooding developers with alerts, AIOps uses **event correlation** to send only meaningful notifications — helping teams focus on true issues and reduce noise.