Module 2 Session 2: Code Review Process, Git Best Practices, Merging, and Conflict Resolution

Duration: 2 hours

Session Objectives:

- Understand the importance of code review in collaborative development.

- Learn the code review process with practical examples.

- Discuss common issues encountered while working with Git, focusing on merging, branches, and resolving conflicts.

Student Capability After the Session:

- Proficiency in conducting and participating in code reviews.

- Understanding of best practices for working with Git in a collaborative environment.

- Capability to manage branches effectively and resolve merge conflicts.

Agenda:

1. Welcome and Overview (10 mins)

- Introduction to the session objectives.

- Emphasis on the significance of code reviews in a collaborative coding environment.

2. Code Review Process (20 mins)

- Explanation of the code review process: initiation, review, feedback, and resolution.

- Live demonstration of a sample code review.

3. Common Git Issues (15 mins)

- Discussion on common issues encountered while working with Git.

- Overview of best practices to avoid common pitfalls.

4. Managing Git Branches (15 mins)

- Explanation of branching in Git.

- Best practices for creating and managing branches.

- Use cases for different types of branches (feature, release, hotfix).

5. Merging in Git (15 mins)

- Understanding the importance of merging in collaborative development.

- Discussion on fast-forward and recursive merge strategies.

6. Resolving Merge Conflicts (25 mins)

- Explanation of merge conflicts and their causes.

- Step-by-step guide on resolving merge conflicts.

- Live demonstration of conflict resolution.

7. Interactive Discussion (10 mins)

- Encourage students to share their experiences and challenges with Git.

- Q&A session for clarification.

8. Q&A and Discussion (10 mins)

- Addressing questions from participants.

- Facilitating a discussion on the covered topics.

Practice Questions for Students:

1. Conduct a code review for a given set of changes, providing constructive feedback.

2. Identify and explain three common issues that developers face when working with Git.

3. Create a new feature branch for a hypothetical project and propose a merge request.

4. Simulate a merge conflict scenario and document the steps to resolve it.