School:...Campus:...

AcademicYear:......................SubjectName:..SubjectCode:..........................

Semester:...............Program:.......................................Branch:.........................Specialization:..........................

Date: ....

Applied and Action Learning

(Learning by Doing and Discovery)

Name of the Experiement : Team Dev – Git and Collaboration in Projects lab

* Coding Phase: Pseudo Code / Flow Chart / Algorithm

Git Basics

- Version control system to track changes in code.
- Key commands:
- git init → Initialize repository
- git clone <repo> → Copy remote repo locally
- git status → Check repository status
- git add <file> → Stage changes
- git commit -m "message" → Commit changes

Branches

- Enable multiple developers to work simultaneously.
- Key commands:
- git branch <branch-name> → Create a branch
- git checkout <branch-name> → Switch branch
- git merge <branch-name> → Merge changes
- Remote Repository & Collaboration
- GitHub / GitLab used for storing and collaborating on projects.

Key commands:

- git remote add origin <url> → Link local repo to remote
- git push origin <branch> → Push local commits to remote
- git pull origin <branch> → Fetch latest changes from remote

Collaboration Workflow

- Fork → Clone → Branch → Commit → Pull Request → Merge
- Use Pull Requests (PR) for code review before merging.
- Resolve conflicts using git merge or git rebase.
- Team Practices
- Follow GitFlow or Feature Branch Workflow

* Softwares used

1. Git (Version Control System)
2. GitHub / GitLab / Bitbucket (Remote Repositories)
3. VS Code

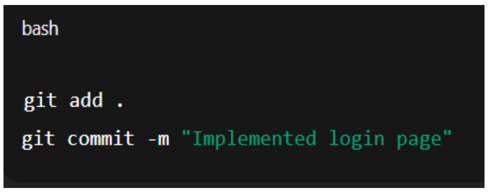
*Asapplicableaccordingtotheexperiment.
Twosheetsperexperiment(10-20)tobeused.

* Implementation Phase: Final Output (no error

1. **Set up Git Repository**

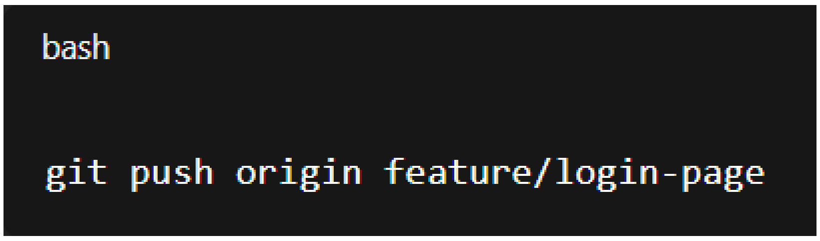
- git init or clone an existing repo.

2. Make Changes & Commit



```bash
git add .
git commit -m "Implemented login page"
```

3. Push Branch to Remote



```bash
git push origin feature/login-page
```

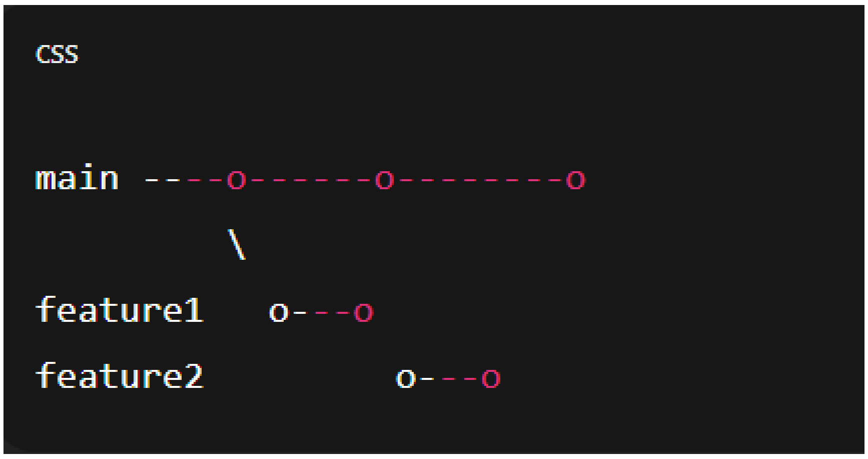
4. Create Pull Request

- Open PR on GitHub to merge feature into main or develop.

5.Merge PR after Review

- Resolve any conflicts.
- Merge and pull latest changes locally.

Git Branching Diagram:



```css
main ---o------o--------o
         \
feature1   o--o
feature2         o--o
```

* Observations

- Git tracks all code changes, making it easy to revert or review history.
- Branching allows multiple developers to work on features simultaneously without conflicts.
- Pull Requests ensure code is reviewed before merging, improving code quality.
- Regular git pull prevents merge conflicts and keeps local repo updated.
- Collaboration through GitHub/GitLab enhances teamwork and project management.

ASSESSMENT

Rubrics	Full Mark	Marks Obtained	Remarks
Concept	10		
Planning and Execution/ Practical Simulation/ Programming	10		
Result and Interpretation	10		
Record of Applied and Action Learning	10		
Viva	10		
Total	**50**		

Signature of the Student:

Name :

Regn. No. :

Signature of the Faculty:

Page No............

*****Asapplicableaccordingtotheexperiment.**
Twosheetsperexperiment(10-20)tobeused.