Lab Title: Implementation of Git

Lab Number: 1

Objective

To understand the basic concepts of Git and practice version control operations such as repository initialization, adding files, committing changes, pushing to remote, branching, merging, pulling updates, and resolving merge conflicts using Git and GitHub.

Theory

Git is a distributed version control system used to manage source code history and collaboration. It helps multiple developers work on the same project without conflicts. Git tracks changes made to files, supports branching for parallel development, and allows merging changes with conflict resolution. GitHub is a popular remote hosting service for Git repositories, enabling easy sharing and collaboration.

Important Concepts:

- **Repository**: The directory where your project is tracked with Git.
- **Commit**: A snapshot of changes saved in Git.
- **Branch**: A separate line of development.
- **Merge**: Combines changes from different branches.
- **Remote**: A repository hosted on a server (e.g., GitHub).

Implementation

1. **Configure Git**

```
git config --global user.name "Your Name"
git config --global user.email "youremail@example.com"
```

2. **Initialize a Repository**

```
mkdir git-lab
cd git-lab
git init
```

3. **Create and Track a File**

```
echo "# Git Lab" > README.md
git add README.md
git commit -m "Initial commit"
```

4. **Create a Remote Repository on GitHub**
    - Go to GitHub and create a new repository named `git-lab`.
5. **Connect Local to Remote and Push**

```
git remote add origin https://github.com/username/git-lab.git
git branch -M main
git push -u origin main
```

6. **Clone Remote Repository (Optional)**

```
git clone https://github.com/username/git-lab.git
```

7. **Create and Switch to New Branch**

```
git checkout -b feature-branch
```

8. **Make Changes and Merge**
    - Modify any file and commit:

```
git add .
git commit -m "Updated README"
```

    - Merge with main:

```
git checkout main
git merge feature-branch
```

9. **Resolve Conflicts (if any)**
    - Manually edit conflicting lines, then:

```
git add <filename>
git commit
```

10. **Pull Updates from Remote**

```
git pull origin main
```

Result

All major Git operations were successfully performed including initialization, committing, pushing, cloning, branching, merging, and resolving conflicts. The code was successfully synchronized between local and remote repositories.

Conclusion

The lab successfully introduced Git as a version control tool. Practical experience was gained in creating repositories, committing changes, working with branches, and handling merge conflicts. This lab demonstrated how Git supports Agile development through efficient collaboration and change tracking.