

 VIT Vidyalkar Institute of Technology ACCREDITED A+ BY NAAC	Department of Information Technology
---	--------------------------------------

Semester	T.E. Semester VI – INFT
Subject	DevOps Lab
Laboratory Teacher:	Bushra Shaikh
Laboratory	CC02

Student Name	Sanskriti Mistry
Roll Number	22101A0055
Grade and Subject Teacher's Signature	

Experiment Number	2
Experiment Title	Automating Version Control and Code Collaboration of TE Mini Project with Git
Problem Statement	TechCo's development team is collaborating on a new feature for their application. However, multiple developers are working on the same parts of the codebase, causing merge conflicts, inconsistent commits, and errors in version control. The team struggles to manage branches effectively and maintain a clean history. The current workflow relies heavily on manual processes, making it difficult to track and resolve issues efficiently.
Resources / Apparatus Required	Hardware: Desktop/Laptop Software:
Code:	gitHub Link: https://github.com/Sanskriti2209 1. Initialize a Git Repository and Push It to GitHub 1.1 Initialize a Git Repository Locally

	<ul style="list-style-type: none">· Navigate to your project directory on your local machine (or create a new project folder).· Open your terminal and run the following command to initialize a Git repository: git init <h3>1.2 Add Files to the Repository</h3> <ul style="list-style-type: none">· Add a new file or make changes to existing files. For example, create a README.md file in the project folder: echo "# My Project" > README.md <h3>1.3 Stage and Commit Files</h3> <ul style="list-style-type: none">· Stage the files for commit using: git add .· Commit the staged files with a message: git commit -m "Initial commit" <h3>1.4 Create a GitHub Repository</h3> <ul style="list-style-type: none">· Go to GitHub and create a new repository. You can name it something like "my-project". · Copy the URL of the GitHub repository, e.g., https://github.com/your-username/my-project.git. <h3>1.5 Push to GitHub</h3> <ul style="list-style-type: none">· Set the remote repository URL: git remote add origin https://github.com/your-username/my-project.git· Push the local repository to GitHub: git push -u origin master <h2>2. Create and Merge Multiple Feature Branches</h2> <h3>2.1 Create a New Feature Branch</h3> <ul style="list-style-type: none">· Create a new branch for a feature. For example, create a feature-login branch:
--	--

`git checkout -b feature-login`

2.2 Make Changes in the Feature Branch

· Add a new file or modify an existing file in the feature-login branch.
For example, modify README.md:

```
echo "Login feature" >> README.md
```

2.3 Stage and Commit Changes

· Stage the changes and commit them:

```
git add README.md git commit -m "Add login feature to README"
```

2.4 Push the Feature Branch to GitHub

· Push the feature branch to GitHub:

```
git push origin feature-login
```

2.5 Create Another Feature Branch

· Create another branch for a different feature, for example, feature-registration:

```
git checkout -b feature-registration
```

2.6 Make Changes in the feature-registration Branch

· Modify the same file (README.md), but with content related to registration:

```
echo "Registration feature" >> README.md
```

2.7 Stage and Commit Changes

· Stage and commit the changes:

```
git add README.md git commit -m "Add registration feature to README"
```

2.8 Push the feature-registration Branch to GitHub

· Push the registration branch to GitHub:

```
git push origin feature-registration
```

3. Simulate a Merge Conflict and Resolve It

3.1 Switch to master Branch

- Now switch back to the master branch:

git checkout master

3.2 Merge feature-login into master

- Merge the feature-login branch into the master branch:

git merge feature-login

- Since the changes are in different parts of the file, the merge will succeed without conflict.

3.3 Merge feature-registration into master

- Now, try to merge the feature-registration branch:

git merge feature-registration

- Since the feature-registration branch made changes to the same file (README.md) at the same location as feature-login, Git will raise a merge conflict.

3.4 Resolve the Merge Conflict

- Open the README.md file. You will see something like this:

```
<<<<<< HEAD Login feature ===== Registration feature >>>>>>
feature-registration
```

- Edit the file to resolve the conflict. For example, you might combine both features like this:

markdown

Login feature Registration feature

3.5 Stage the Resolved Conflict

- Once the conflict is resolved, stage the file:

git add README.md

3.6 Commit the Merge

- Commit the merge with a message:

git commit -m "Resolve merge conflict between feature-login and feature-registration"

4. Perform a Pull Request Review and Handle the Integration Process

4.1 Create a Pull Request (PR) on GitHub

- Go to your repository on GitHub.
- You should see a button to create a Pull Request for the branches you've pushed (feature-login and feature-registration).
- Click on "New Pull Request" and select the feature-login branch and compare it with master.
- After reviewing the changes, click "Create Pull Request."
- Add a description of the changes and create the PR.

4.2 Review the Pull Request

- Review the changes in the pull request.
- You can see the changes made and leave comments on specific lines of code if necessary.
- You can ask a team member to review it as well or approve it yourself.

4.3 Merge the Pull Request

- After the PR review, click the "Merge pull request" button.
- Choose "Confirm merge" to integrate the changes from feature-login into master.

4.4 Repeat the Process for feature-registration

- Repeat the same process for the feature-registration branch.
- Create a new PR for feature-registration, review it, and merge it into master.

4.5 Clean Up the Branches

- After successfully merging the feature branches, delete them from GitHub (optional):

git push origin --delete feature-login git push origin --delete feature-registration

5. [Connect GitHub Cloud to Jira](#)

When you connect a GitHub Cloud organization to Jira, your team can link their development activity to Jira issues. This lets you track branches, commits, and pull requests in the context of your Jira issues, on your Jira board, in the releases feature, and more.

5.1 Install the GitHub for Jira app

1. In Jira, select Apps, then select Explore more apps.
2. Search for GitHub for Jira, then select it from the results.
3. Select Get app, then Get it now.

5.2 Connect a GitHub organization

1. After the app is installed, select Get started. If the app is already installed on your Jira site, you can find this section by selecting Apps, then Manage your apps, and then GitHub for Jira.
2. Select Continue.
3. Select GitHub Cloud, then Next.
4. Enter your GitHub username and password, then Sign in.
5. Find the organization you want to connect to Jira, then select Connect.

5.3 Add the Jira app to a new GitHub organization

If no organizations are available to connect to Jira, you'll need to install the Jira app on a new GitHub organization.

1. From step five in the instructions above, Select an organization in GitHub.
2. The GitHub site will open in a new tab and show a list of all the organizations available in your GitHub account. Select the organization you want to connect to Jira.
3. Choose the repositories you want to give Jira access to by selecting either All repositories or Only select repositories.

4. Select Install.

5.4 Connect a new GitHub repository

- If you selected All repositories when you connected a GitHub organization, Jira will be able to access all the repositories in that organization, including any new repositories you create.

- If you selected Only select repositories, any new repositories you create won't be added automatically. Here's how to add them:

1. In Jira, select Apps, then select Manage apps.

2. In the sidebar, under GitHub for Jira, select Configure.

3. Find the organization that contains the repo you want to add and select the three-dot menu (...) to view available actions.

4. Select Configure.

5. A new tab will open with your GitHub organization settings. Under Repository access, select the dropdown Select repositories.

6. Select the repository you want to connect to, then Save.

5.5 Link your development information to Jira issues

To link branches, commits, and pull requests to Jira, your team must include Jira issue keys in their development actions.








1. Find the issue key for the Jira issue you want to link to, for example "JRA-123". You can find the key in several places in Jira:
 - On the board, issue keys appear at the bottom of a card.
 - On the issue's details, issue keys appear in the navigation at the top of the page.

2. Check out a new branch in your repo, using the issue key in the branch name. For example, `git checkout -b JRA-123-<branch-name>`.

3. When committing changes to your branch, use the issue key in your commit message to link those commits to the development panel in your Jira issue.

For example, `git commit -m "JRA-123 <summary of commit>".`

4. When you create a pull request, use the issue key in the pull request title.

	<p>· After you push your branch, you'll see development information in your Jira issue.</p>
Output:	<div><h2>Add Login feature to README #7</h2><p>Merged Sanskruti2209 merged 1 commit into <code>main</code> from <code>feature-login</code> 1 minute ago</p><p>Conversation 2 · Commits 1 · Checks 0</p><p>Sanskruti2209 commented 6 minutes ago</p><p>new pull request - feature login</p><p></p><p> Sanskruti2209 Add Login feature to README</p><p>Sanskruti2209 commented 3 minutes ago</p><p>Sanskruti2209 left a comment</p><p>perfect</p><p></p><p>Sanskruti2209 commented 2 minutes ago</p></div> <div><p>Merged Add Login feature to README #1</p><p>Sanskruti2209 merged 1 commit into <code>main</code> from <code>feature-login</code> 1 minute ago</p><p></p><p>Sanskruti2209 commented 2 minutes ago View reviewed code</p><p>Sanskruti2209 left a comment Owner Author</p><p>completed</p><p></p><p>Sanskruti2209 merged commit <code>f9457a1</code> into <code>main</code> 1 minute ago</p><p>Pull request successfully merged and closed Delete branch</p><p>You're all set—the <code>feature-login</code> branch can be safely deleted.</p><p> Add a comment</p><p> Try the new merge editor</p></div>

harshada05044 / myproject

Issues

Pull requests

Actions

Projects

Wiki

Security

Insights

myproject

Public

Watch 1

create_user

had recent pushes 4 minutes ago

Compare & pull request

main

3 Branches

0 Tags

Go to file

Add file

Code

harshada05044

Merge pull request #2 from harshada05044/feature-registration

494c8ca · yesterday

8 Commits

README.md

Merge branch 'main' into feature-registration

yesterday

README

Login feature Registration feature

Manage access

Add people

Select all

Type

Find a collaborator...

harshada05044
Collaborator




submitted successfully.

Add registration feature to README #2


Merged Sanskruti2209 merged 2 commits into `main` from `feature-registration` now


Conversation **1** Commits **2** Checks **0** Files changed **1**







Sanskriti2209 commented 2 minutes ago Owner ...


new pull request2 - feature Registration





Sanskriti2209 and others added 2 commits 22 minutes ago


-   Add registration feature to README 0f8d694 None
-   Merge branch 'main' into feature-registration Verified 7e4a7f7




Sanskriti2209 commented now View reviewed changes

Sanskriti2209 left a comment Owner Author ...

resolved merge conflicts




Merged Add registration feature to README #2
Sanskruti2209 merged 2 commits into `main` from `feature-registration` now




Sanskriti2209 commented 1 minute ago View reviewed changes


Sanskriti2209 left a comment Owner Author ...

resolved merge conflicts







Sanskriti2209 merged commit `b32ba1b` into `main` now Revert














Pull request successfully merged and closed Delete branch

You're all set—the `feature-registrati...` branch can be safely deleted.

 [Try the new merge experience](#)



Add a comment

Write Preview H B I           

Add your comment here...

```

MINGW64/c/GitDemo
ACER@LAPTOP-2GKL12QM MINGW64 ~
$ cd C:\GitDemo

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo
$ git init
Initialized empty Git repository in C:/GitDemo/.git/

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (master)
$ echo "# My Project" > README.md

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (master)
$ git add .
warning: in the working copy of 'README.md', LF will be replaced by CRLF the next time you commit

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (master)
$ git commit -m [initial commit]
error: pathspec '[initial commit]' did not match any file(s) known to git

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (master)
$ git commit -m "Initial commit"
[master (root-commit) e76b8c5] Initial commit
1 file changed, 1 insertion(+)
create mode 100644 README.md

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (master)
$ git branch -M main
bash: '$\E[200~git': command not found

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (master)
$ git branch -M main
git remote add origin https://github.com/Sanskriti2209/Anomaly-Detection.git
git push -u origin main

```

```

MINGW64/c/GitDemo
ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (main)
$ git checkout -b feature-login
Switched to a new branch 'feature-login'

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (feature-login)
$ echo "Login feature" >> README.md

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (feature-login)
$ git add README.md git commit -m "Add login feature to README"
error: unknown switch 'm'
usage: git add [<options>] [--] <pathspec>...

    -n, --[no-]dry-run      dry run
    -v, --[no-]verbose     be verbose

    -i, --[no-]interactive
                           interactive picking
    -p, --[no-]patch        select hunks interactively
    -e, --[no-]edit         edit current diff and apply
    -f, --[no-]force        allow adding otherwise ignored files
    -u, --[no-]update        update tracked files
    --[no-]renormalize       renormalize EOL of tracked files (implies -u)
    -N, --[no-]intent-to-add
                           record only the fact that the path will be added later
    -A, --[no-]all          add changes from all tracked and untracked files
    --[no-]ignore-removal   ignore paths removed in the working tree (same as --no-al

)
    --[no-]refresh          don't add, only refresh the index
    --[no-]ignore-errors    just skip files which cannot be added because of errors
    --[no-]ignore-missing   check if - even missing - files are ignored in dry run
    --[no-]sparse           allow updating entries outside of the sparse-checkout cone
    --[no-]chmod (+|-)x     override the executable bit of the listed files
    --[no-]pathspec-from-file <file>
                           read pathspec from file

```

```

MINGW64/c/GitDemo
remote:      https://github.com/Sanskriti2209/Anomaly-Detection/pull/new/feature-login
remote:
To https://github.com/Sanskriti2209/Anomaly-Detection.git
* [new branch]      feature-login -> feature-login

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (feature-login)
$ git checkout -b feature-registration
Switched to a new branch 'feature-registration'

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (feature-registration)
$ echo "Registration feature" >> README.md

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (feature-registration)
$ git add README.md git commit -m "Add registration feature to README"
error: unknown switch 'm'
usage: git add [<options>] [--] <pathspec>...

    -n, --[no-]dry-run      dry run
    -v, --[no-]verbose      be verbose

    -i, --[no-]interactive  interactive picking
    -p, --[no-]patch        select hunks interactively
    -e, --[no-]edit         edit current diff and apply
    -f, --[no-]force        allow adding otherwise ignored files
    -u, --[no-]update        update tracked files
    --[no-]renormalize       renormalize EOL of tracked files (implies -u)
    -N, --[no-]intent-to-add record only the fact that the path will be added later
    -A, --[no-]all          add changes from all tracked and untracked files
    --[no-]ignore-removal   ignore paths removed in the working tree (same as --no-all)
    --[no-]refresh          don't add, only refresh the index

```

```

MINGW64/c/GitDemo
ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (feature-registration)
$ git push origin feature-registration
Total 0 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
remote:
remote: Create a pull request for 'feature-registration' on GitHub by visiting:
remote:      https://github.com/Sanskriti2209/Anomaly-Detection/pull/new/feature-registration
remote:
To https://github.com/Sanskriti2209/Anomaly-Detection.git
* [new branch]      feature-registration -> feature-registration

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (feature-registration)
$ git checkout master
error: pathspec 'master' did not match any file(s) known to git

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (feature-registration)
$ git merge feature-login
Already up to date.

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (feature-registration)
$ git merge feature-registration
Already up to date.

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (feature-registration)
$ git checkout -b feature-login
fatal: a branch named 'feature-login' already exists

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (feature-registration)
$ git checkout feature-login
Switched to branch 'feature-login'
   README.md
ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (feature-login)

```

```

MINGW64:/c/GitDemo

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (feature-login)
$ git checkout feature-registration
Switched to branch 'feature-registration'

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (feature-registration)
$ echo "Registration feature" >> README.md

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (feature-registration)
$ git add .
warning: in the working copy of 'README.md', LF will be replaced by CRLF the next time Git touches it

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (feature-registration)
$ git commit -m "Add registration feature to README"
[feature-registration 0f8d694] Add registration feature to README
1 file changed, 1 insertion(+)

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (feature-registration)
$ git commit -m "Add registration feature to README"
On branch feature-registration
nothing to commit, working tree clean

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (feature-registration)
$ git push origin feature-registration
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Writing objects: 100% (3/3), 299 bytes | 299.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Sanskriti2209/Anomaly-Detection.git
e76b8c5..0f8d694 feature-registration -> feature-registration

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (feature-registration)
$ git checkout main

```

```

MINGW64:/c/GitDemo

Writing objects: 100% (3/3), 299 bytes | 299.00 KiB/s, done.
Total 3 (delta 0), reused 0 (delta 0), pack-reused 0 (from 0)
To https://github.com/Sanskriti2209/Anomaly-Detection.git
e76b8c5..0f8d694 feature-registration -> feature-registration

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (feature-registration)
$ git checkout main
Switched to branch 'main'
Your branch is up to date with 'origin/main'.

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (main)
$ git merge feature-login
Updating e76b8c5..ca7b3e0
Fast-forward
 4 README.md | 3 +++
1 file changed, 3 insertions(+)

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (main)
$ git merge feature-registration
Auto-merging README.md
CONFLICT (content): Merge conflict in README.md
Automatic merge failed; fix conflicts and then commit the result.

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (main|MERGING)
$ git add README.md

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (main|MERGING)
$ git commit -m "Resolve merge conflict between feature-login and feature-registration"
[main cd42dc3] Resolve merge conflict between feature-login and feature-registration

ACER@LAPTOP-2GKL12QM MINGW64 /c/GitDemo (main)
$ git push origin --delete feature-login git push origin --delete feature-registration

```