



# DEVELOPER GUID

Development best practices

Sarbajeet Singh  
Sarbajeet.singh@hcl.com

### Revision History

#	Version	Modified By	Modification Date	Description
1	V0.1	Sarbajeet Singh	08 Feb 23	Initial Draft creation and modification

## Contents

Local Setup.....	3
Git Tools .....	3
GitHub Desktop Client.....	3
GitBash .....	3
IDE Setup.....	3
GitHub .....	4
Account Creation .....	4
Repository Creation .....	5
Clone Repository.....	6
Using GitHub Desktop Client.....	6
Using Git Bash .....	7
Local Changes.....	8
Import Code .....	8
Changes to Remote Repository Using GitHub Desktop Client.....	9
Push changes.....	9
Pull changes .....	10
Changes to Remote Repository Using Git Bash (Command line) .....	10
Add changes.....	10
Commit changes.....	11
Push changes.....	12
Pull changes .....	13
PR Review.....	13
Code Merge.....	15

## Local Setup

### Git Tools

#### GitHub Desktop Client

Install GitHub Client on local laptop from below URL (<https://desktop.github.com/>)

#### GitBash

Install GitBash Client to work on Command Prompt. ([Git - Downloading Package \(git-scm.com\)](#))

### IDE Setup

Download Eclipse IDE or Intelije Idea or Visual studio code from TARMAC ([TARMAC \(hcl.com\)](#))

The screenshot displays the TARMAC Software Asset Management Policy portal. The browser address bar shows the URL: <https://tools.hcl.com/tool/user/toolinventory.aspx?searchtool=Eclipse>. The page header includes the TARMAC logo, a "Software Asset Management Policy" link, and user information for SIDDHARTHA AGRAWAL (52130079) with a "Logout" button. A search bar at the top left contains the text "Raise New License Request". Below the search bar, it indicates "99 result(s) found for Eclipse" and shows filters for "Tools (86)" and "Bundled Tools (13)". The main content area lists four Eclipse-related tools, each with a blue wrench icon, a description, a cost field, an information icon, and a license status box:

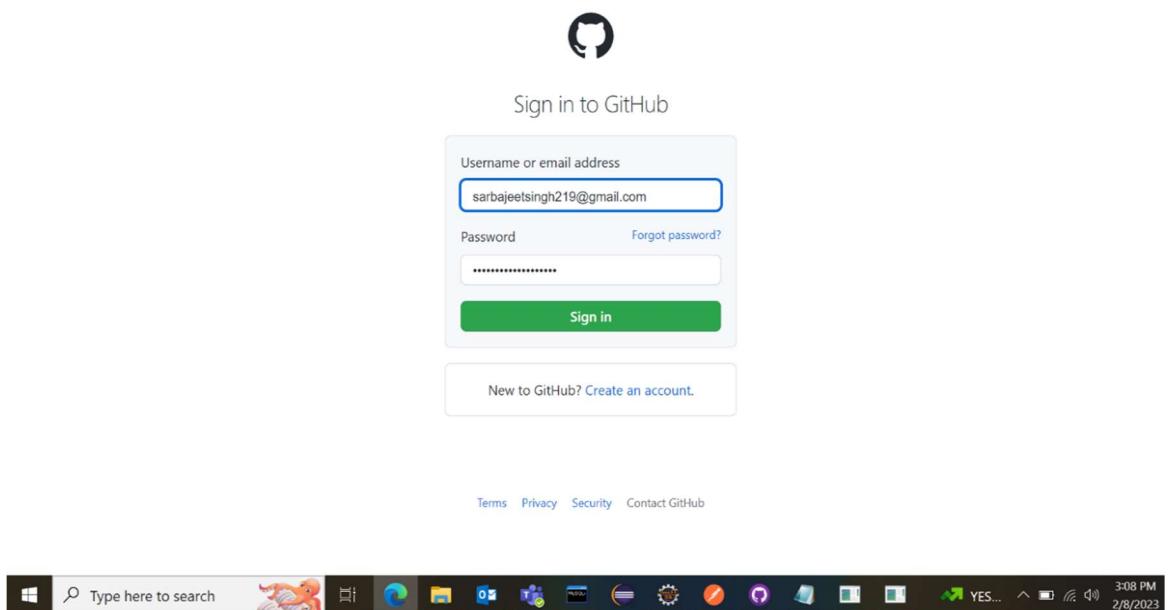
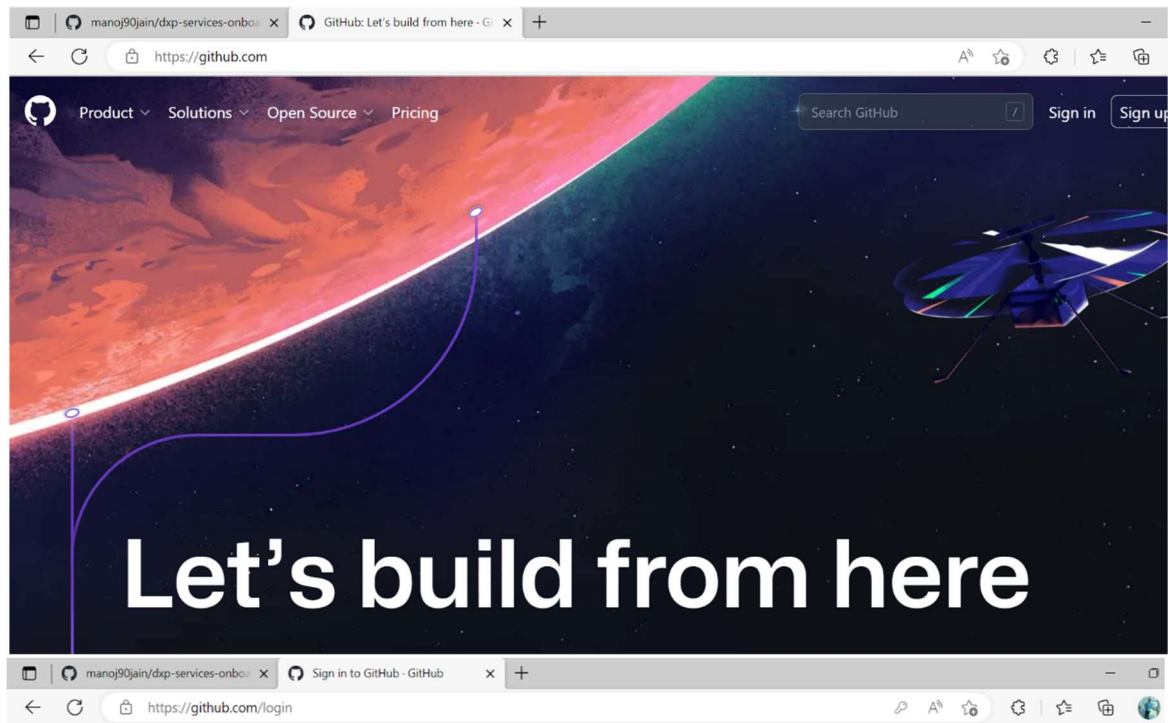
Tool Name	Description	Cost	License Status
EclipseCrossword	Max. Per Unit Per Month Cost (INR):	Information icon	Issuances Restricted
Eclipse for Mobile Developers	Max. Per Unit Per Month Cost (INR):	Information icon	Issuances Restricted
Eclipse for RCP Plug-in	Max. Per Unit Per Month Cost (INR):	Information icon	Issuances Restricted
eclipse plugin for jax-ws webservises	Max. Per Unit Per Month Cost (INR):	Information icon	HCL Inventory: 9974

At the bottom, it says "Showing 1 to 4 of 99 entries" and includes a pagination bar with numbers 1, 2, 3, 4, 5, and 25.

# GitHub

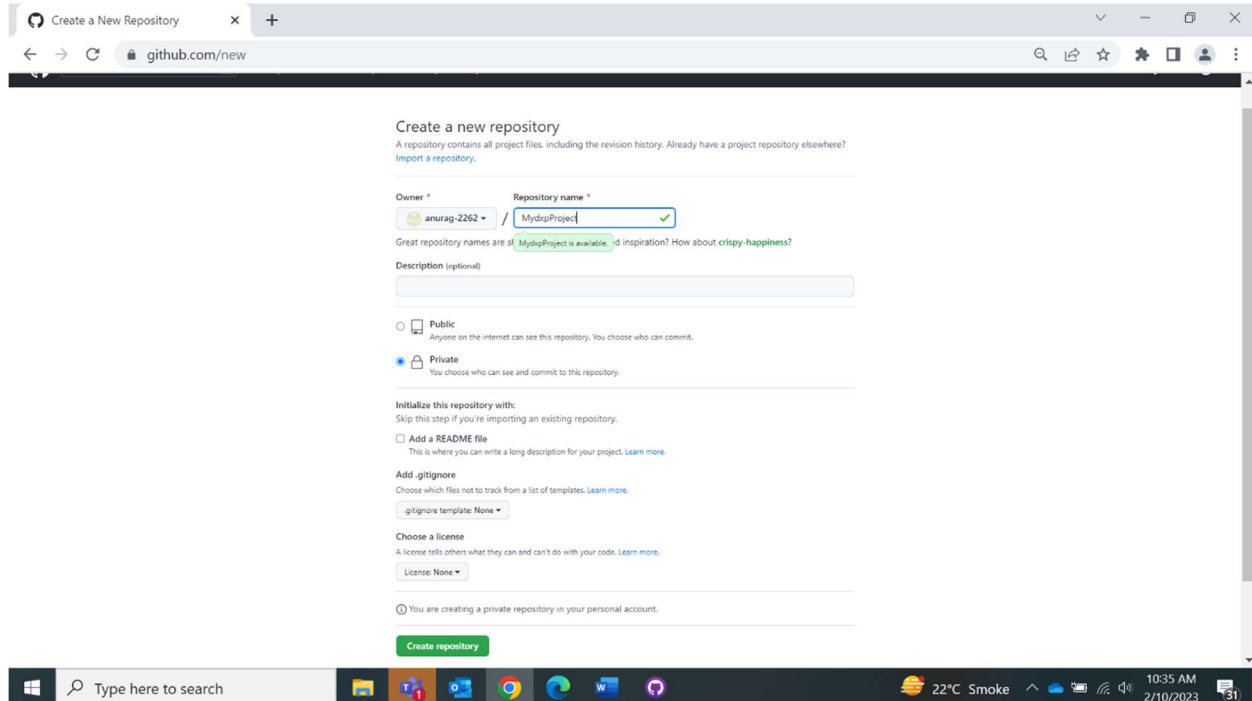
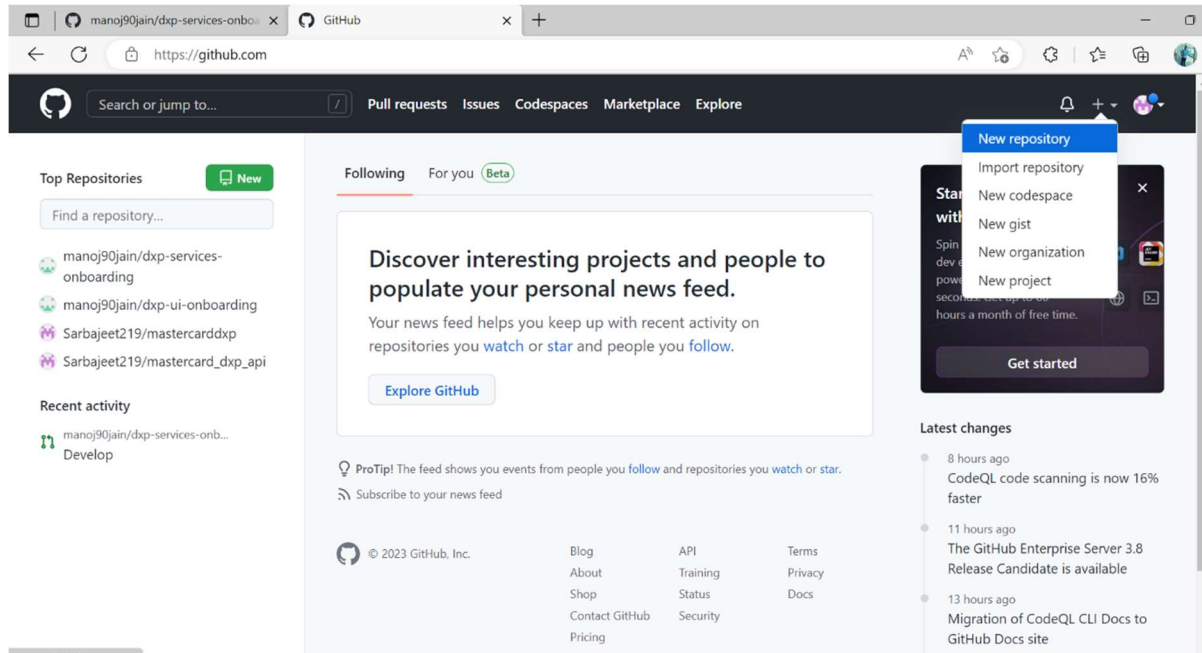
## Account Creation

Go to <https://github.com> and sign up for a new account using your email address and a strong password.



## Repository Creation

A repository is a place to store and organize your code projects. To create a repository, click on the "+" sign in the top right corner and select "New repository." Enter a name for your repository, choose if it should be **public or private**, and initialize it with a README file if you like.

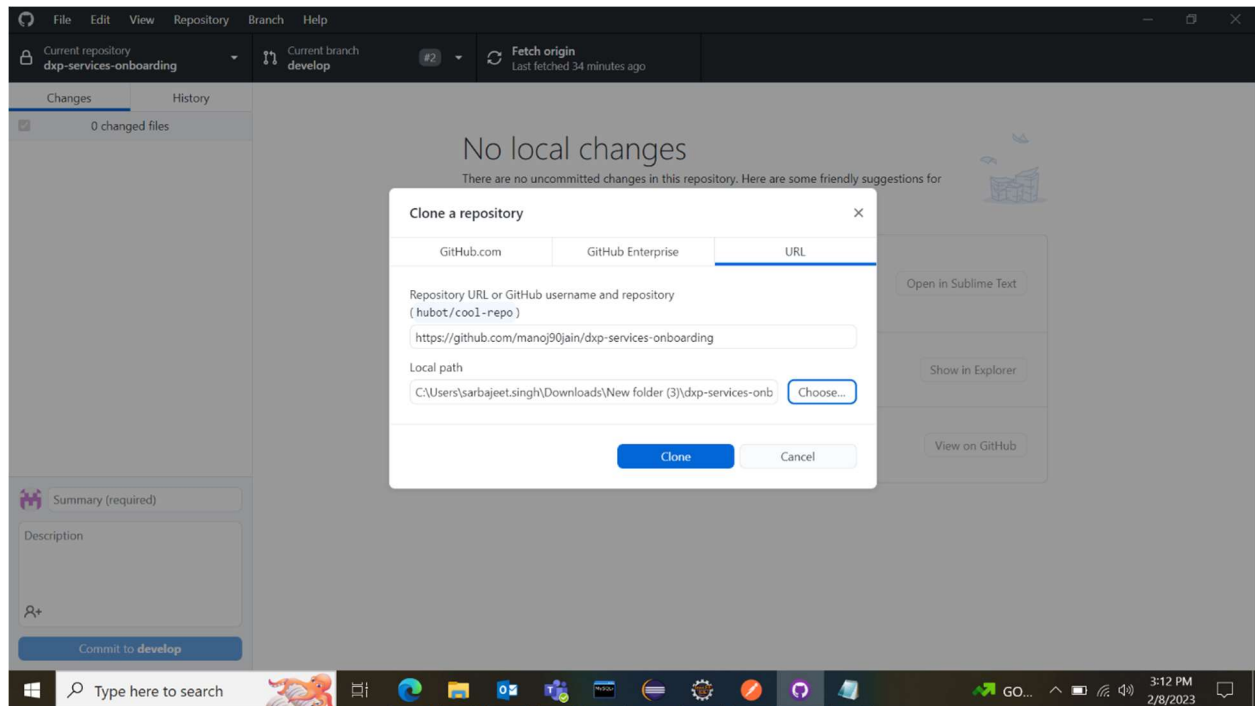


Always select/create private repository for HCL specific development/PoC

## Clone Repository

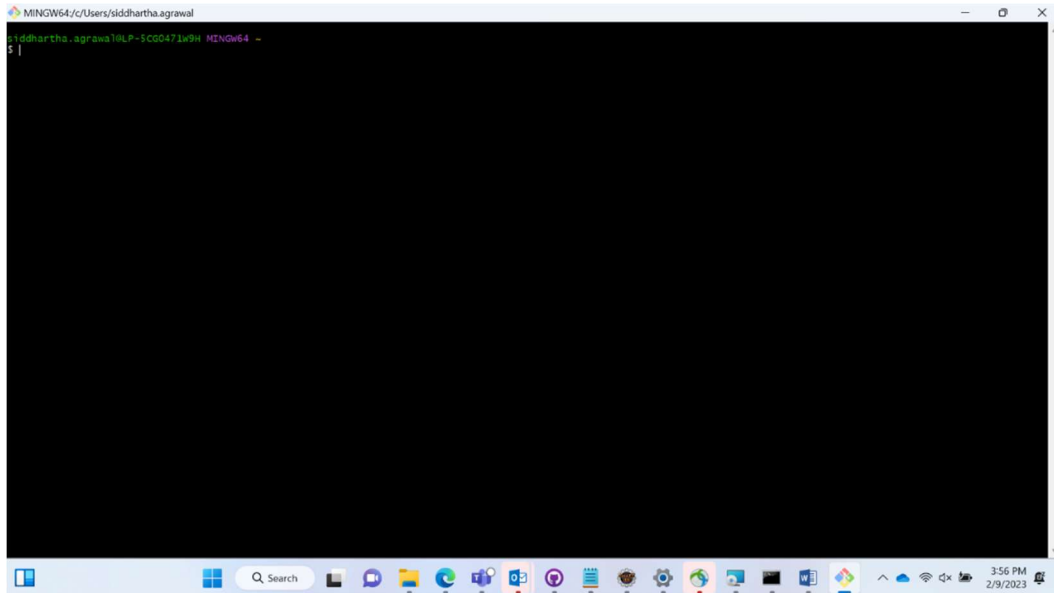
### Using GitHub Desktop Client

We need to clone the repository to our computer to work on it. To clone any repository, go to the repository on GitHub and click on the "Clone or download" button. Copy the URL of the repository. In GitHub client we must first fork the repository in GitHub website then it will show in current repository option of GitHub client. Right click on repository in current repository go to clone repository, select folder, and clone the repository.

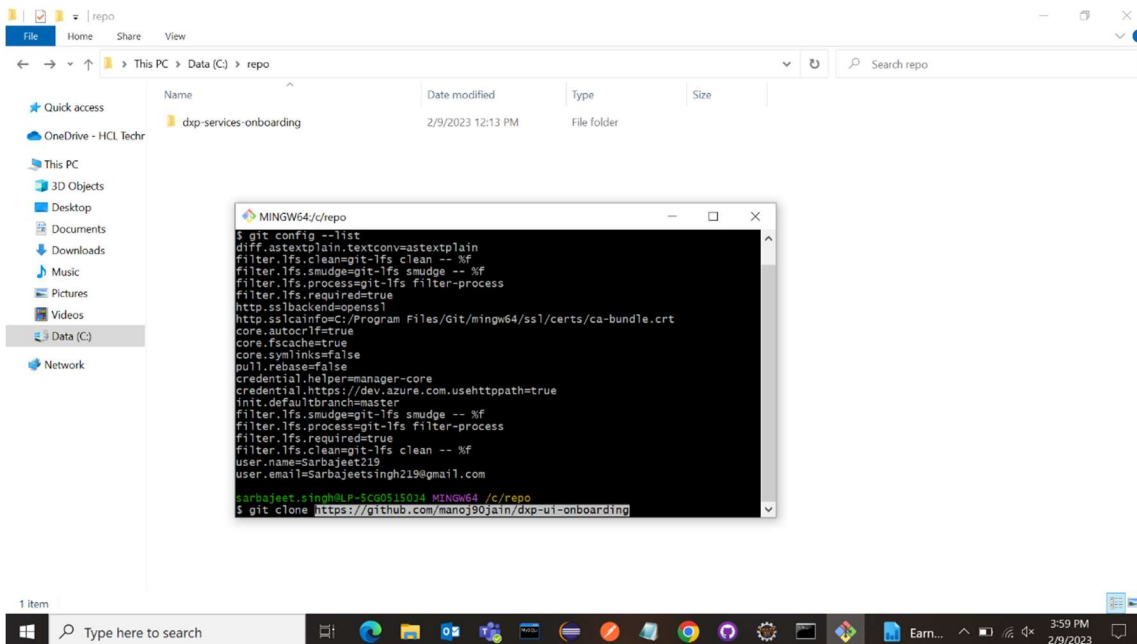


## Using Git Bash

Open Gitbash installed into local machine.



Open a terminal or command prompt and navigate to the directory where you want to clone the repository. Use the following command to clone the repository:

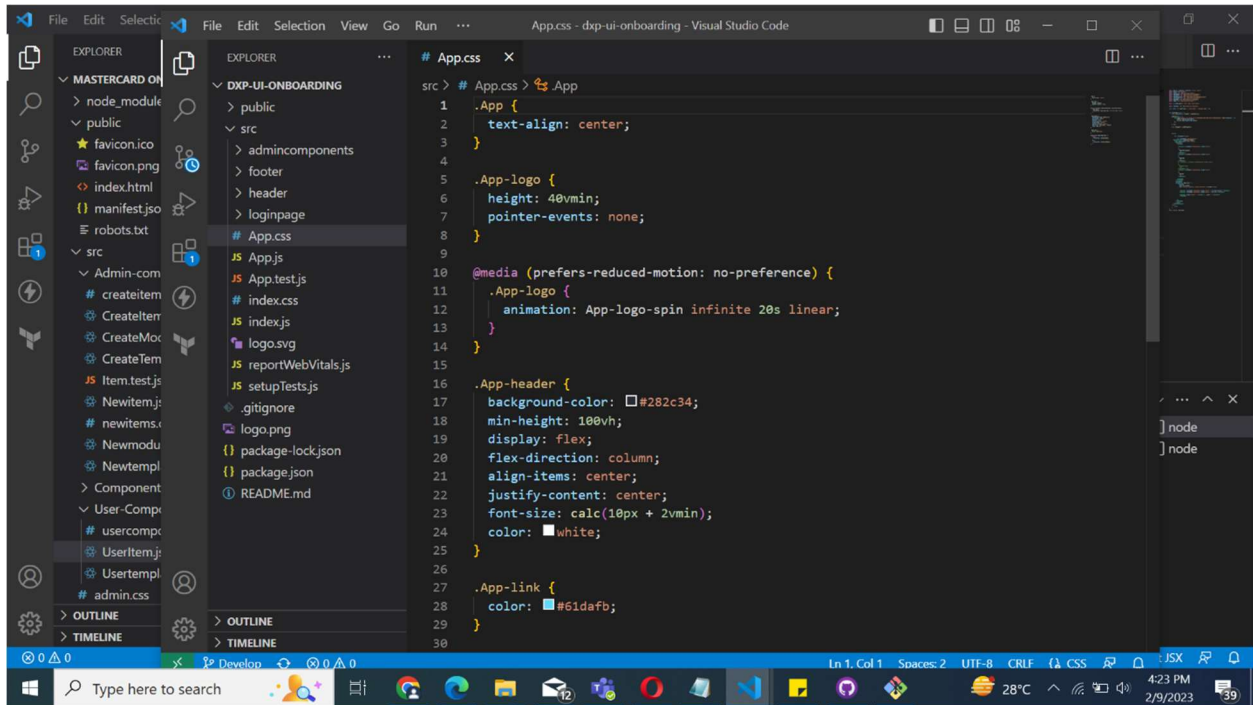




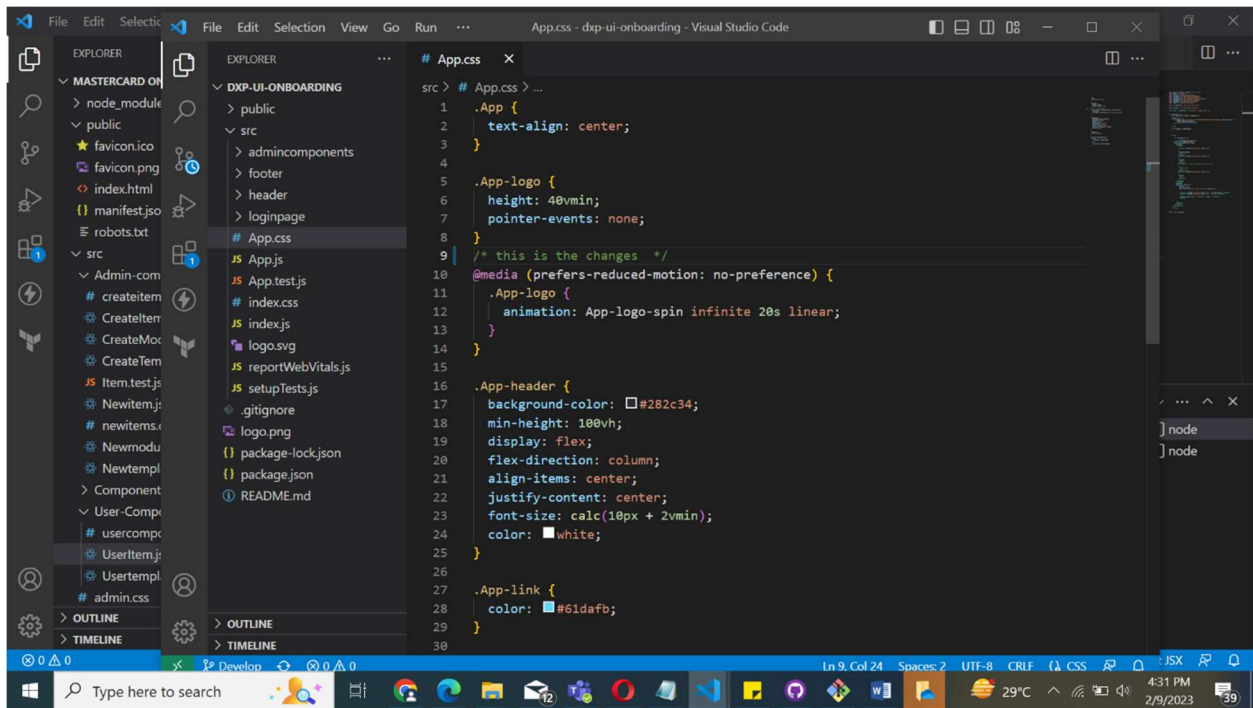
## Local Changes

## Import Code

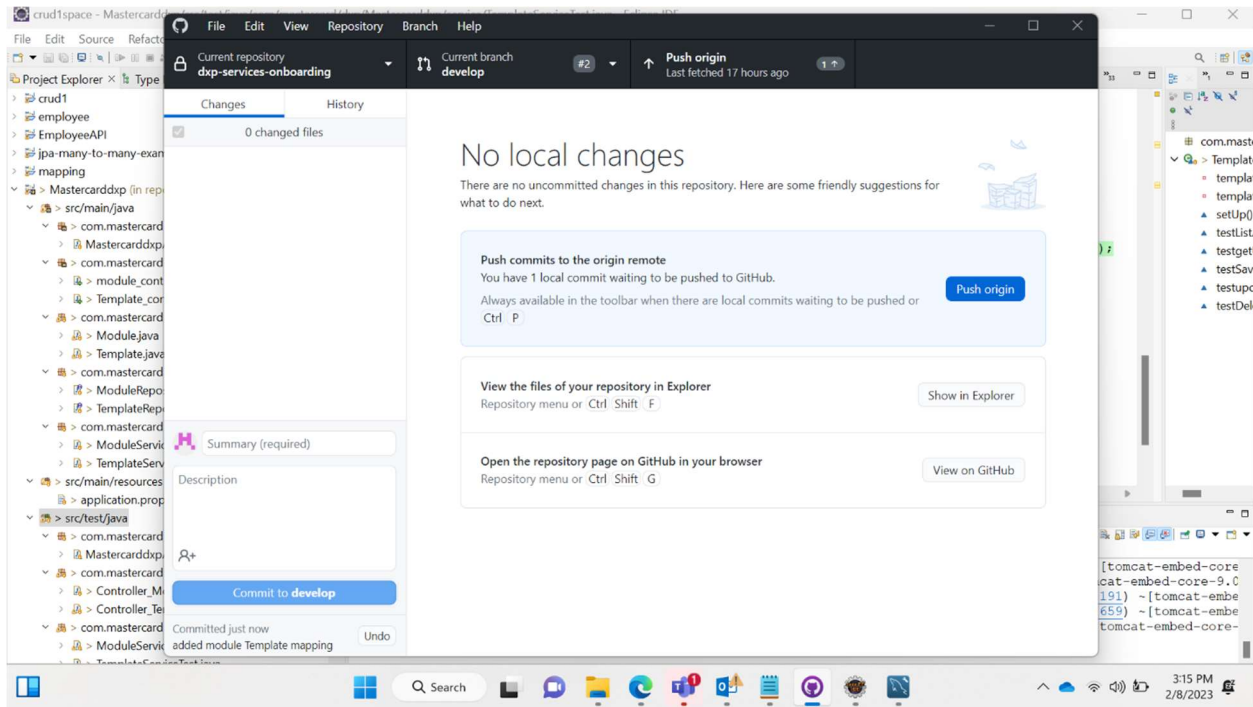
After successful clone import the code to IDE



Make the changes in code :

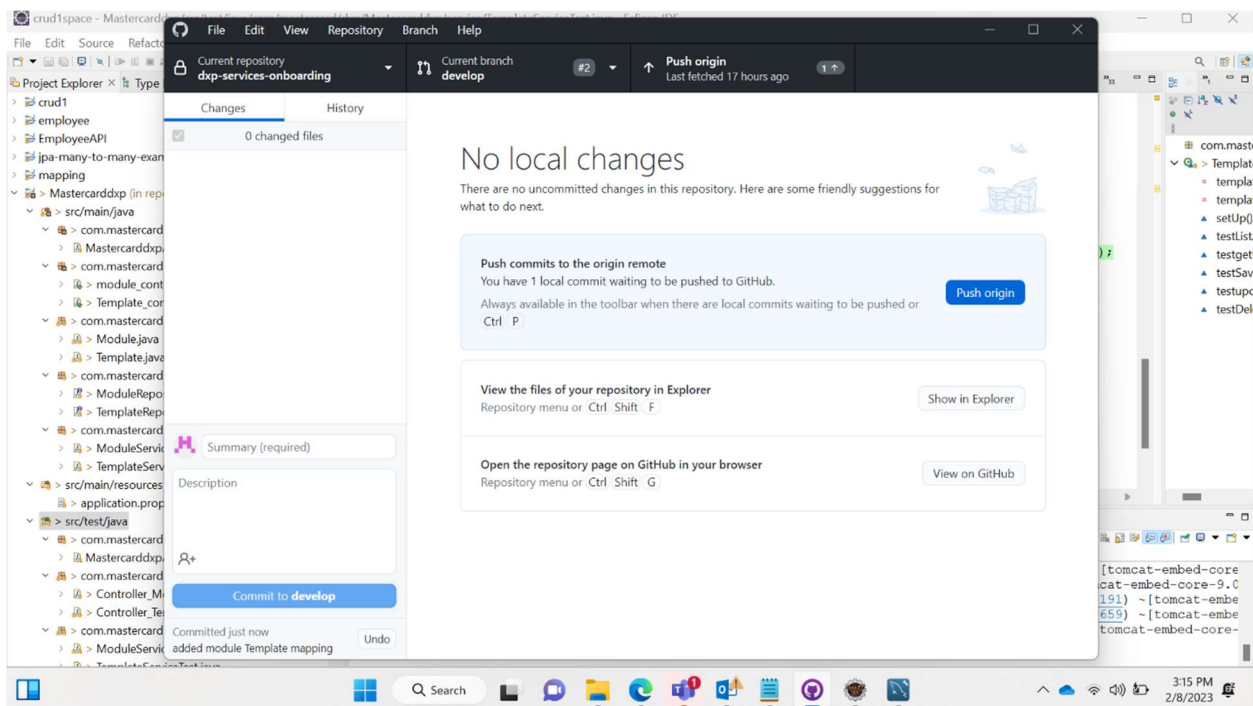


## Changes to Remote Repository Using GitHub Desktop Client



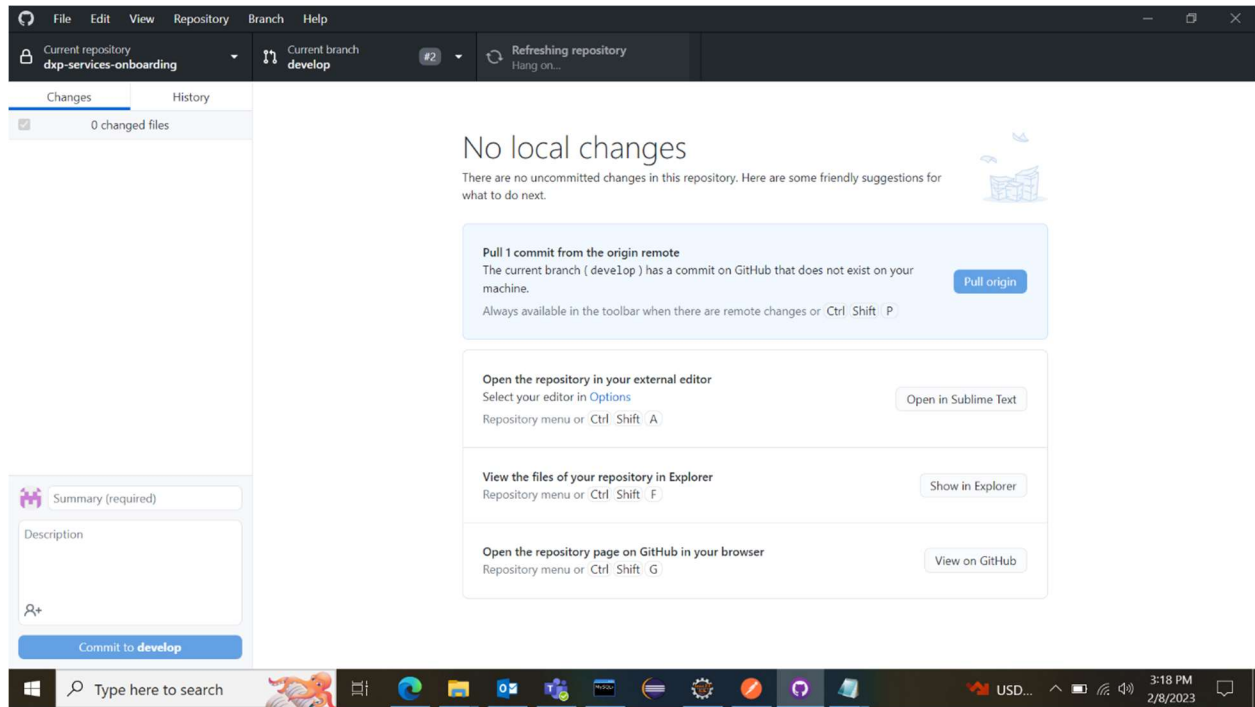
### Push changes

Push the changes to the remote repository in GitHub Client. After committing we can see push origin option in GitHub client.



## Pull changes

In GitHub client we can see the pull origin option.



## Changes to Remote Repository Using Git Bash (Command line)

### Add changes

Once you have made changes to the code, you need to stage the changes for commit. Use the following command to stage the changes:

```
$ git add
```

```
MINGW64/c/repo/dxp-ui-onboarding
rani.gupta@LP-5CG131BKX8 MINGW64 /c/repo/dxp-ui-onboarding (main)
$ git branch
* main

rani.gupta@LP-5CG131BKX8 MINGW64 /c/repo/dxp-ui-onboarding (main)
$ git checkout Develop
Switched to a new branch 'Develop'
branch 'Develop' set up to track 'origin/Develop'.

rani.gupta@LP-5CG131BKX8 MINGW64 /c/repo/dxp-ui-onboarding (Develop)
git: 'status' is not a git command. See 'git --help'.

The most similar command is
    status

rani.gupta@LP-5CG131BKX8 MINGW64 /c/repo/dxp-ui-onboarding (Develop)
$ git status
On branch Develop
Your branch is up to date with 'origin/Develop'.

nothing to commit, working tree clean

rani.gupta@LP-5CG131BKX8 MINGW64 /c/repo/dxp-ui-onboarding (Develop)
$ git status
On branch Develop
Your branch is up to date with 'origin/Develop'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   src/App.css

no changes added to commit (use "git add" and/or "git commit -a")

rani.gupta@LP-5CG131BKX8 MINGW64 /c/repo/dxp-ui-onboarding (Develop)
$ git add src/App.css

rani.gupta@LP-5CG131BKX8 MINGW64 /c/repo/dxp-ui-onboarding (Develop)
$ git status
On branch Develop
Your branch is up to date with 'origin/Develop'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   src/App.css

rani.gupta@LP-5CG131BKX8 MINGW64 /c/repo/dxp-ui-onboarding (Develop)
$
```

## Commit changes

Commit the changes to the repository using the following command:

```
$ git commit -m "Your commit message"
```

**Never commit to the master/Main branch.** Commit your changes to developer or another branch. In GitHub client open your repository where at left hand side we can see all the changes we have made in code. Write description and enter on commit button below.

```
MINGW64/c/repo/dxp-ui-onboarding
$ git checkout Develop
Switched to a new branch 'Develop'
branch 'Develop' set up to track 'origin/Develop'.

rani.gupta@LP-5CG131BKX8 MINGW64 /c/repo/dxp-ui-onboarding (Develop)
git: 'status' is not a git command. See 'git --help'.

The most similar command is
    status

rani.gupta@LP-5CG131BKX8 MINGW64 /c/repo/dxp-ui-onboarding (Develop)
$ git status
On branch Develop
Your branch is up to date with 'origin/Develop'.

nothing to commit, working tree clean

rani.gupta@LP-5CG131BKX8 MINGW64 /c/repo/dxp-ui-onboarding (Develop)
$ git status
On branch Develop
Your branch is up to date with 'origin/Develop'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   src/App.css

no changes added to commit (use "git add" and/or "git commit -a")

rani.gupta@LP-5CG131BKX8 MINGW64 /c/repo/dxp-ui-onboarding (Develop)
$ git add src/App.css

rani.gupta@LP-5CG131BKX8 MINGW64 /c/repo/dxp-ui-onboarding (Develop)
$ git status
On branch Develop
Your branch is up to date with 'origin/Develop'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   src/App.css

rani.gupta@LP-5CG131BKX8 MINGW64 /c/repo/dxp-ui-onboarding (Develop)
$ git commit -m "The changes i made in App.css file"
[Develop 7ded150] The changes i made in App.css file
1 file changed, 1 insertion(+), 1 deletion(-)

rani.gupta@LP-5CG131BKX8 MINGW64 /c/repo/dxp-ui-onboarding (Develop)
$
```

## Push changes

Push the changes to the remote repository using the following

**\$ git push**

```
MINGW64/c/repo/dxp-ui-onboarding
On branch Develop
Your branch is up to date with 'origin/Develop'.

nothing to commit, working tree clean

rani.gupta@LP-5CG131BKX8 MINGW64 /c/repo/dxp-ui-onboarding (Develop)
$ git status
On branch Develop
Your branch is up to date with 'origin/Develop'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   src/App.css

no changes added to commit (use "git add" and/or "git commit -a")

rani.gupta@LP-5CG131BKX8 MINGW64 /c/repo/dxp-ui-onboarding (Develop)
$ git add src/App.css

rani.gupta@LP-5CG131BKX8 MINGW64 /c/repo/dxp-ui-onboarding (Develop)
$ git status
On branch Develop
Your branch is up to date with 'origin/Develop'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   src/App.css

rani.gupta@LP-5CG131BKX8 MINGW64 /c/repo/dxp-ui-onboarding (Develop)
$ git commit -m "The changes i made in App.css file"
[Develop 7ded150] The changes i made in App.css file
1 file changed, 1 insertion(+), 1 deletion(-)

rani.gupta@LP-5CG131BKX8 MINGW64 /c/repo/dxp-ui-onboarding (Develop)
$ git push
Enumerating objects: 7, done.
Counting objects: 100% (7/7), done.
Delta compression using up to 12 threads
Compressing objects: 100% (4/4), done.
Writing objects: 100% (4/4), 408 bytes | 57.00 KiB/s, done.
Total 4 (delta 3), reused 0 (delta 0), pack-reused 0
remote: Resolving deltas: 100% (3/3), completed with 3 local objects.
To https://github.com/manoj90jain/dxp-ui-onboarding.git
   bfb0777..7ded150  Develop -> Develop

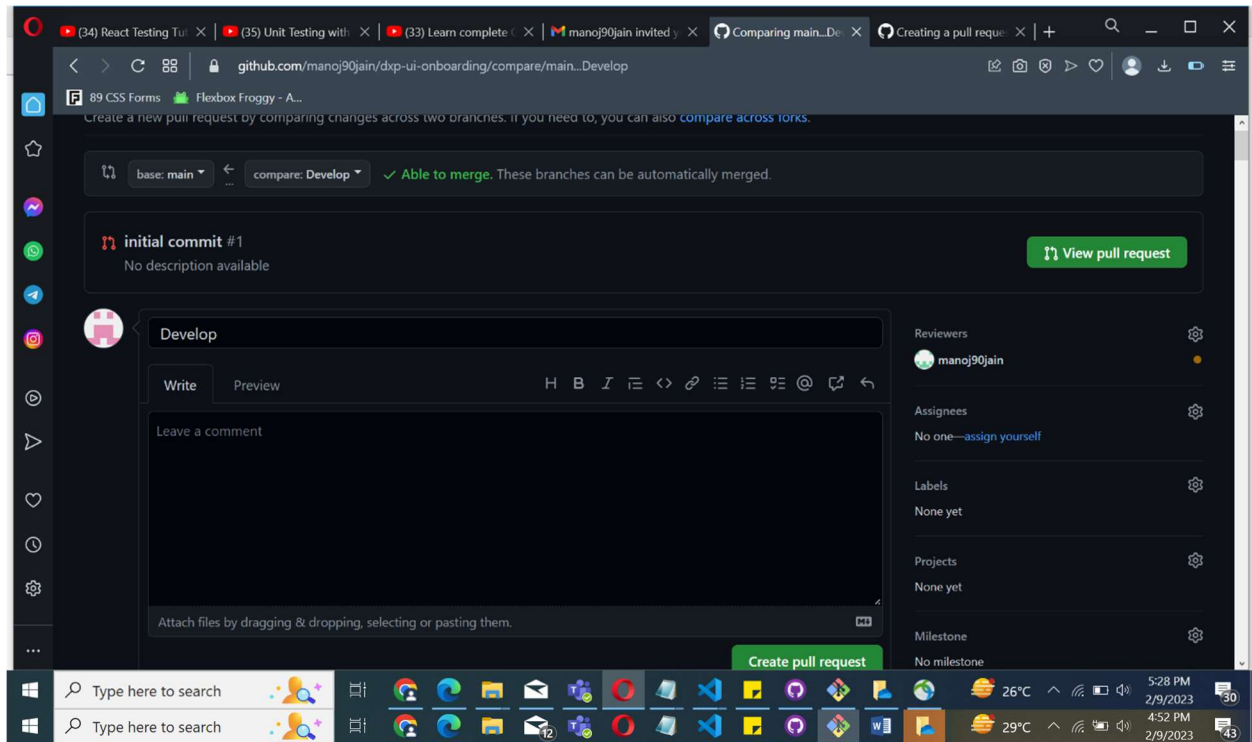
rani.gupta@LP-5CG131BKX8 MINGW64 /c/repo/dxp-ui-onboarding (Develop)
$
```

## Pull changes

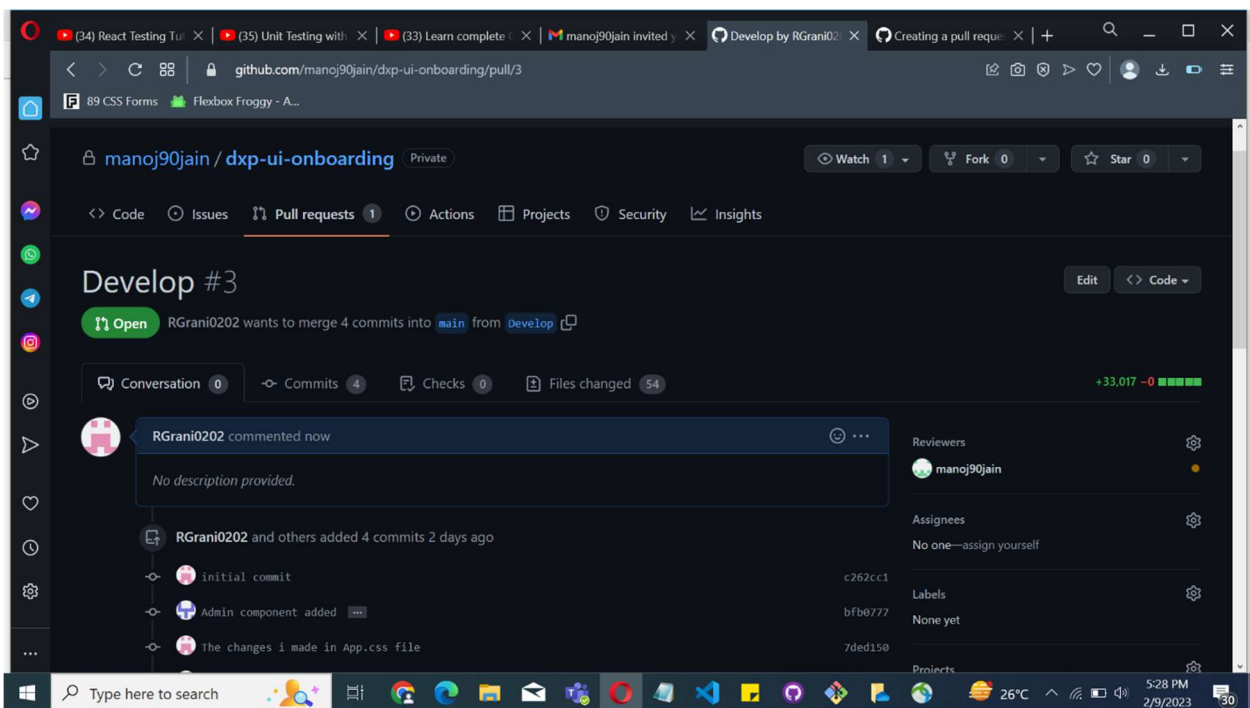
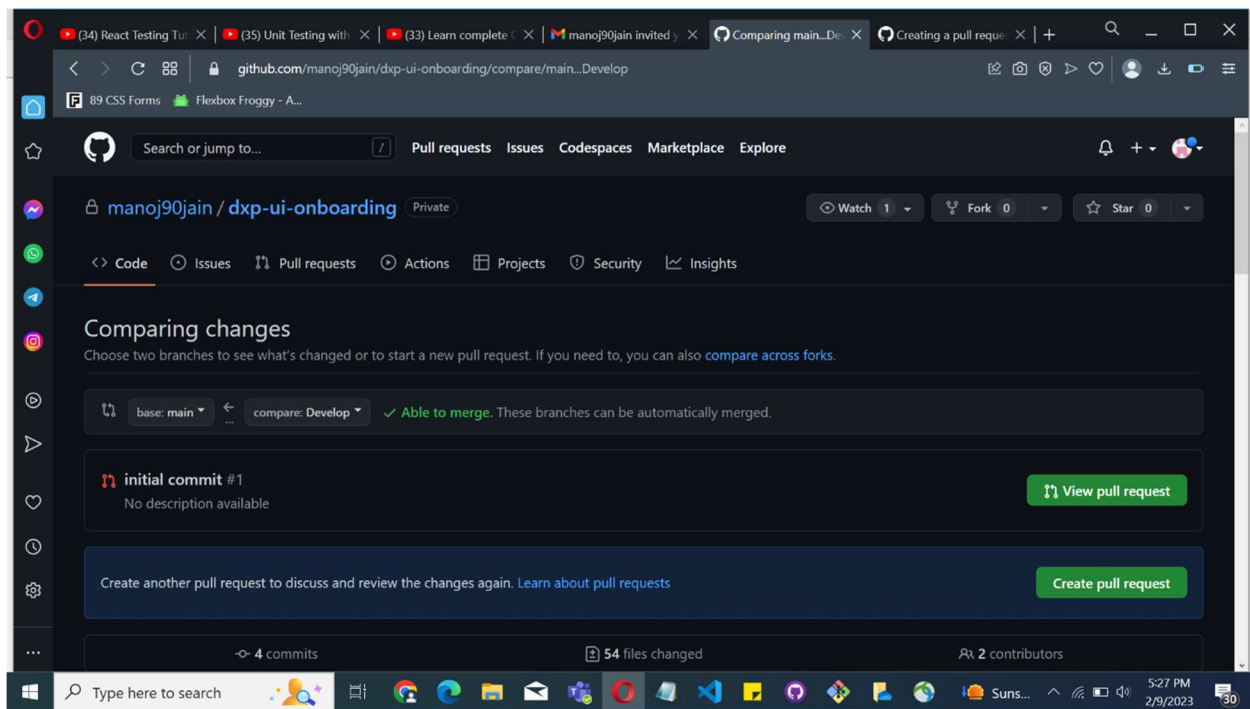
If someone else has made changes to the repository, you can pull the changes to your local repository using the following

```
$ git pull
```

## PR Review







## Code Merge

Merge branches: When you are done working on a branch, you can merge it with the master branch using the following command:

```
git checkout master git merge branch-name
```

Resolve conflicts: If there are conflicts between the branches, you will need to resolve them before you can merge the branches. Use a text editor to open the conflicted files and resolve the conflicts by choosing which changes you want to keep. Push changes: Once the conflicts have been resolved, push the changes to the remote repository using the following command: `git push`