# Repository creation:

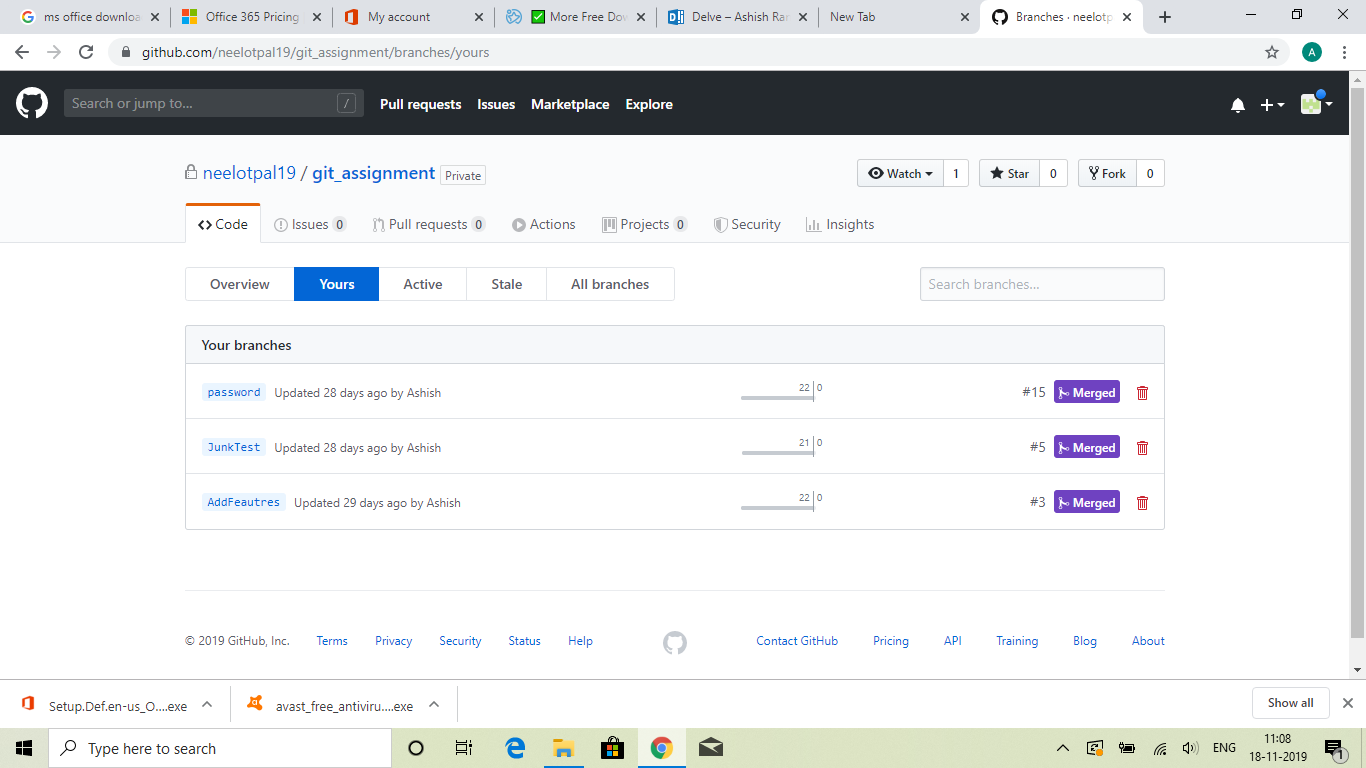
Step 1- Login to github account

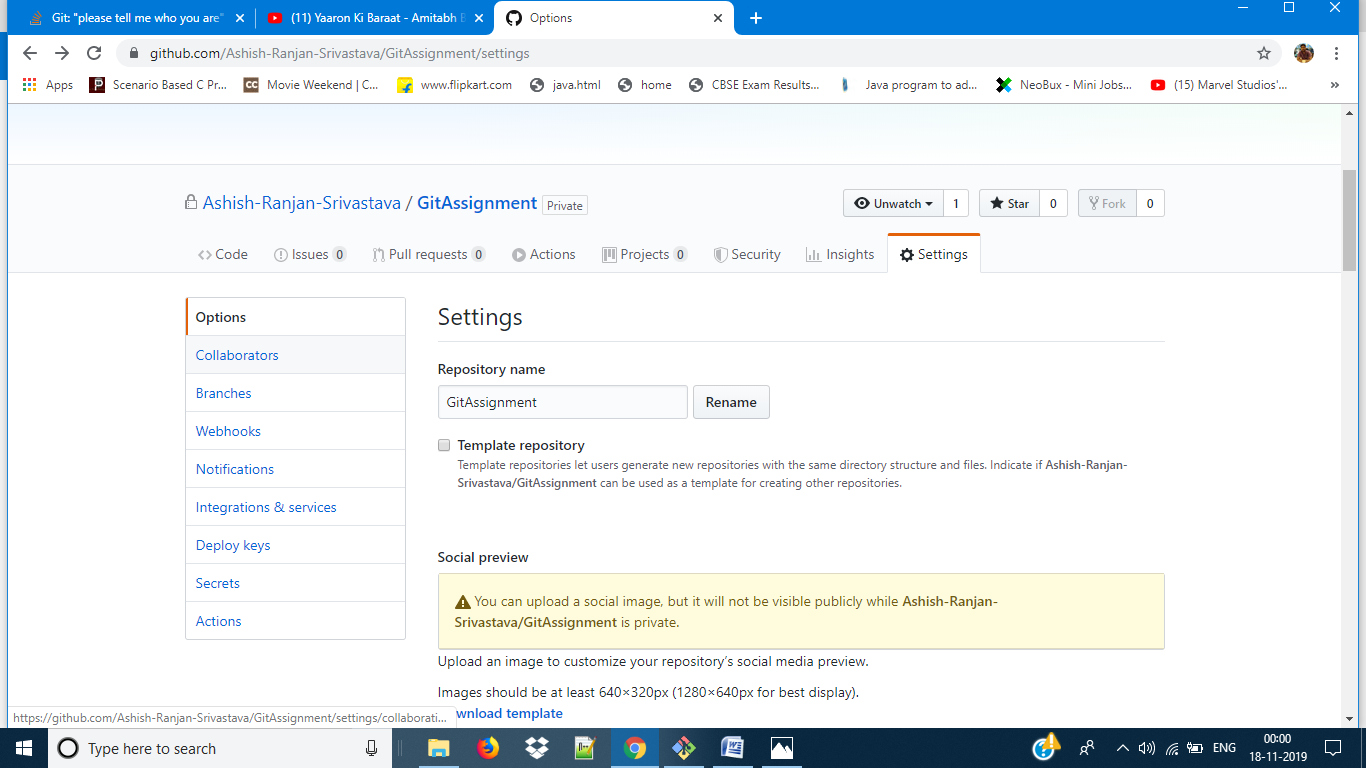
Step 2 – create new repository (let say repository name GitAssignment)

Step 3- for P1 (person p1) choose private

Step 4- Go to repository setting for collaboration

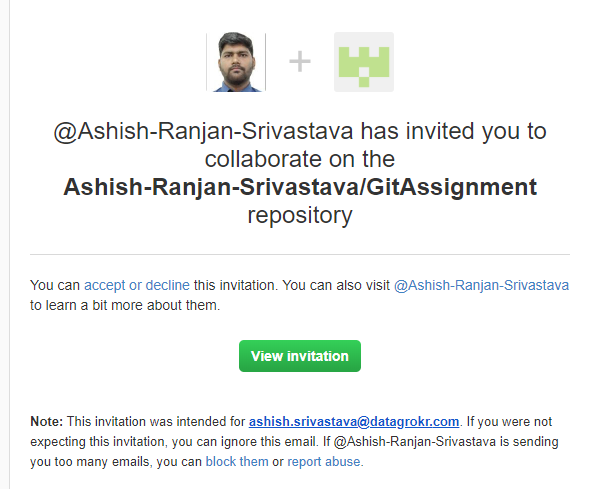
Step5 – invite other person





Step 6 – enter credentials

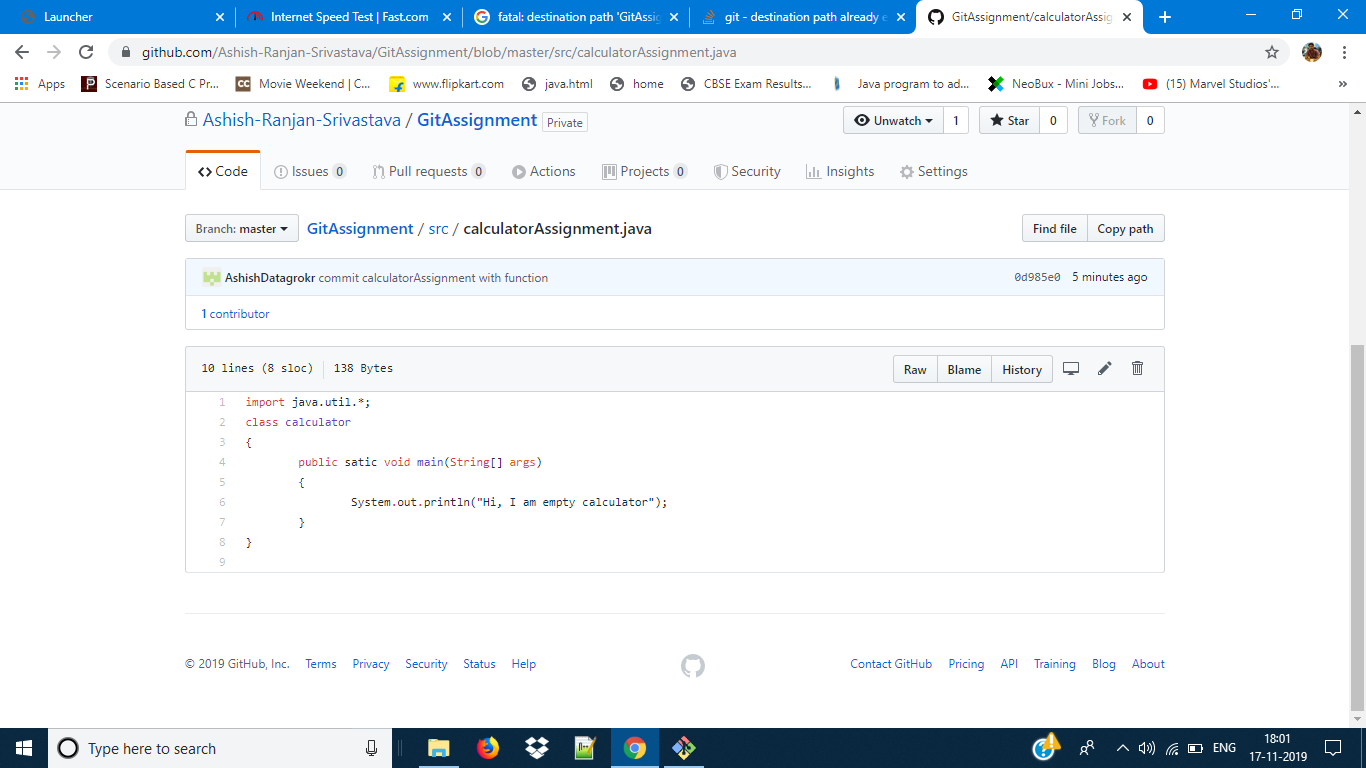
Step 7- person 2 will get collaboration request on email



# Both person must have git bash installed

# Task of person P2

* Clone the repository using “git clone link of github repository”
* Create directory using mkdir directory name
* Open file name calculator.java either by nano or vim command
* Write the functionality and save it
* Before committing we have to add our current directory via “git add . or git add branch Name”
* Now commit the file by using git commit –m “message for description”
* Now push the commit file by git push origin(or master)

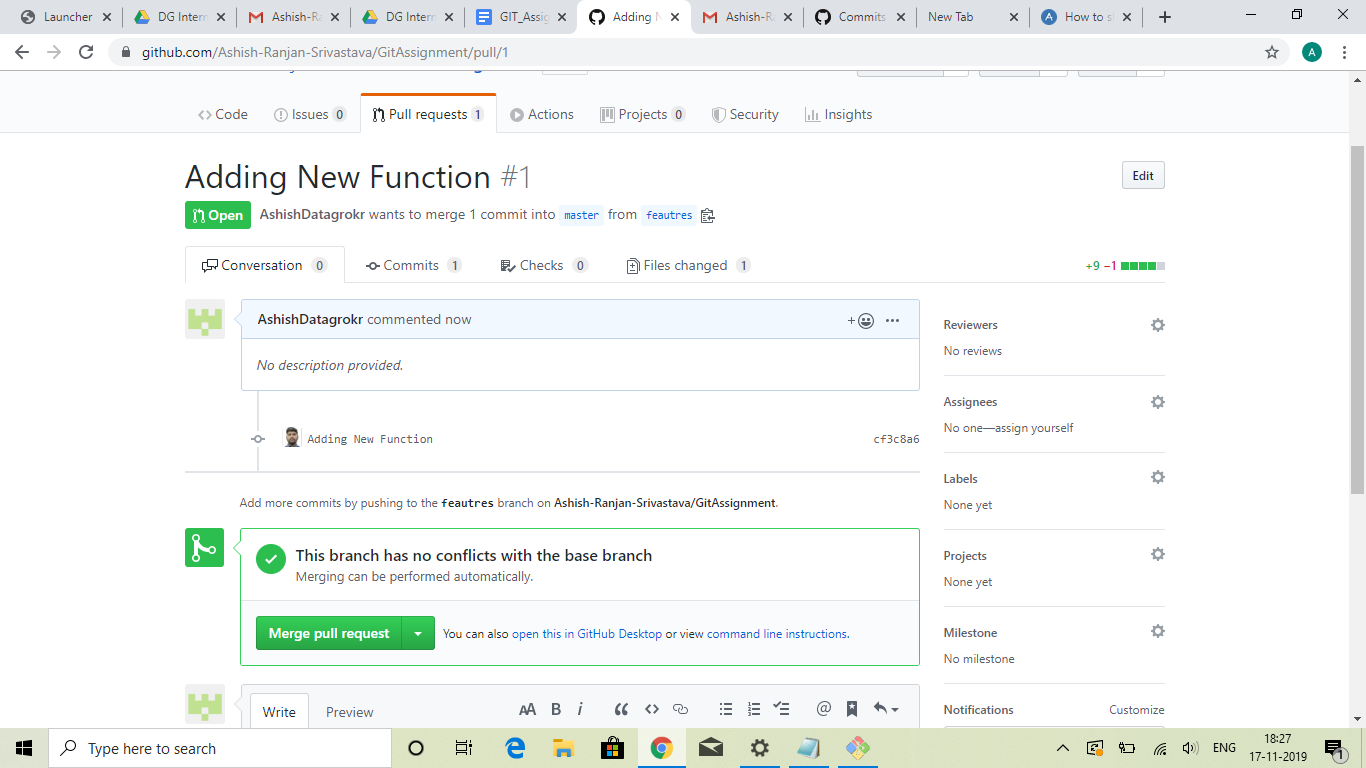


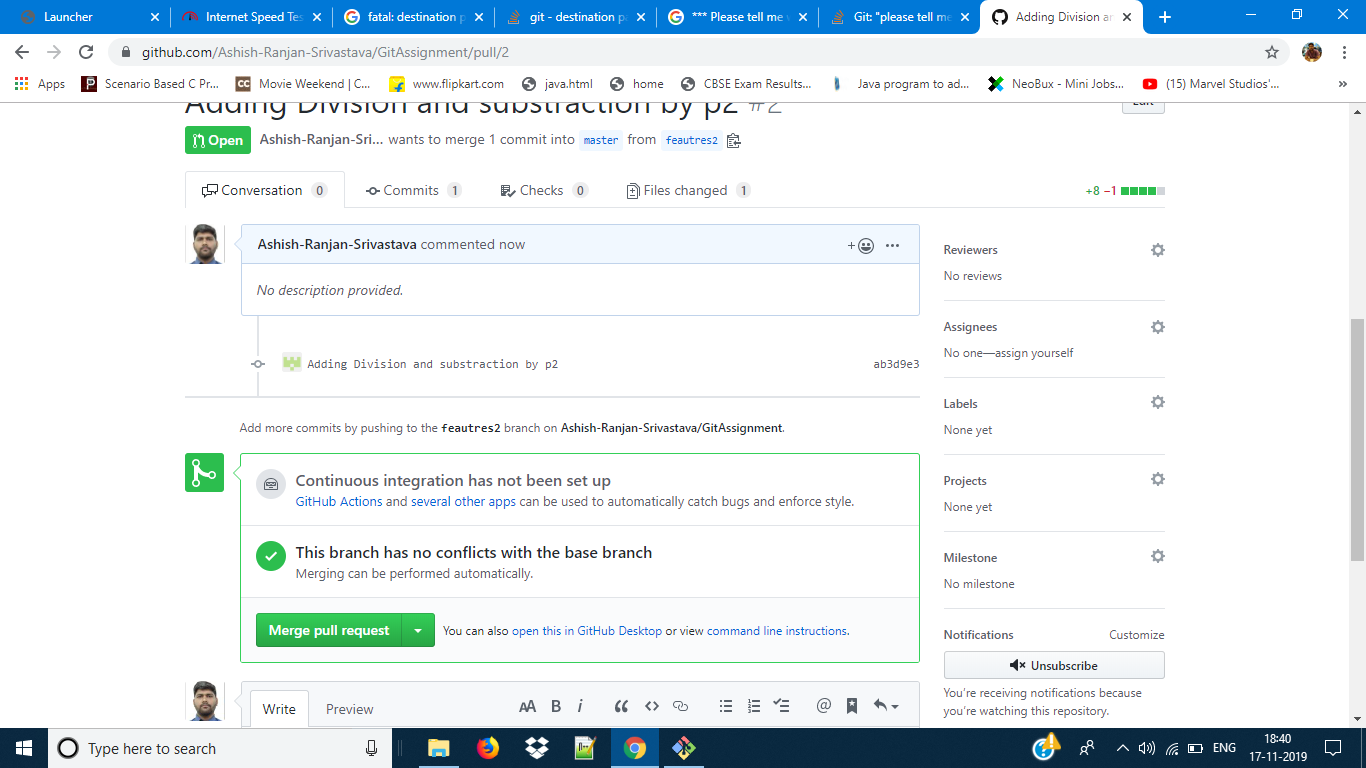
# 1.3 Branch Creation

# Task for person p1

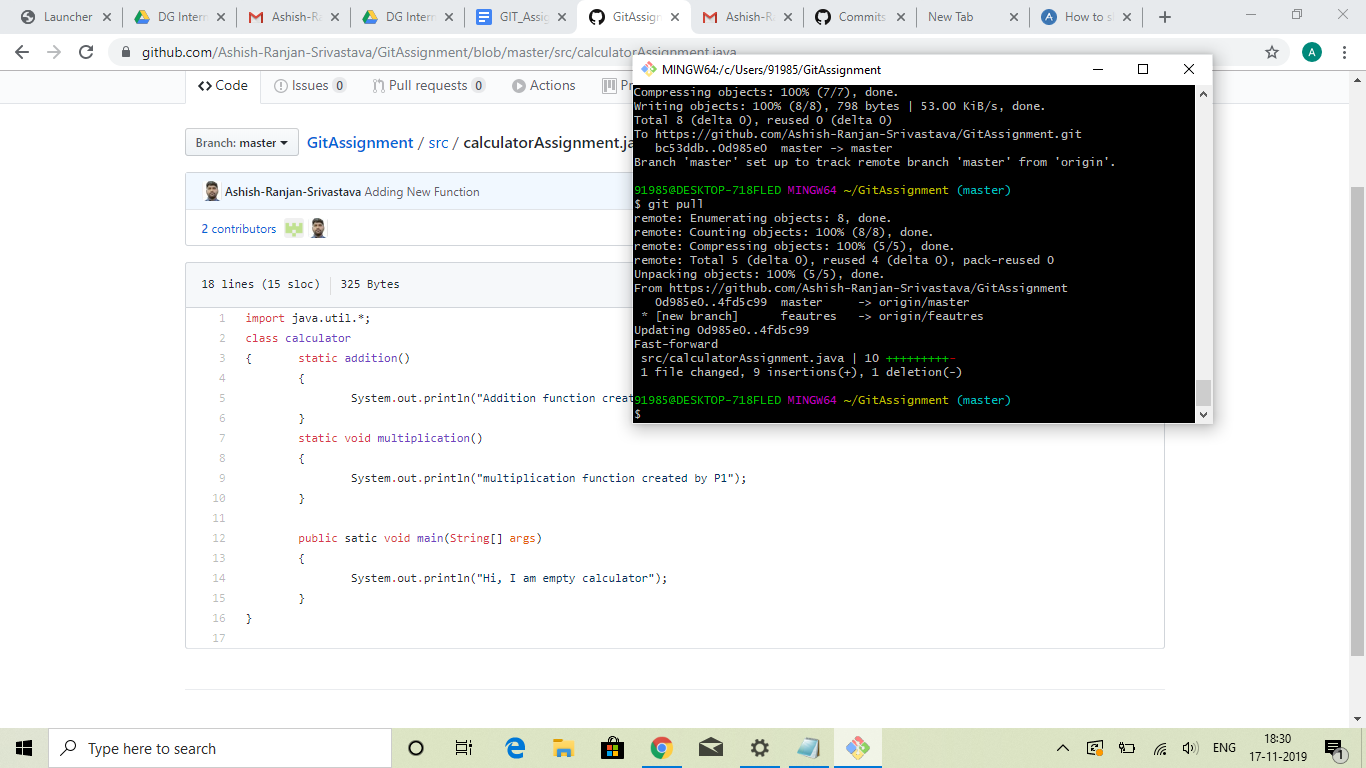
* Clone repository steps already mentioned above
* Now create branch via **git branch branchName** command
* Now switch to newly created branch via **git checkout branchName**
* Go to src directory then calculator.java
* Open through **vim calculator.java command**
* Add functionality Addition and multiplication
* Now push and commit directory

Go to github web repository you will get pull and merge the branches





After pull request and merge changes will be reflected



Now person p2 will add functionality division and substraction step will be same as p1:

• Clone repository steps already mentioned above

• Now create branch via git branch branchName command

• Now switch to newly created branch via git checkout branchName

• Go to src directory then calculator.java

• Open through vim calculator.java command

• Add functionality Addition and multiplication

• Now push and commit directory

Go to github web repository you will get pull and merge the branches

1.4

Q1> Find how to push your files for staging and commit using one command? Q2> Now create new file and commit your changes using single command you found and note what happened and why it happened?

Ans**:- git config –global alias .name ‘!git add – Al && git commit ‘**

**git commit name –m”message”**

**[Ques]: 2**

**S1. Create branch from master.**

**S2. Create file junk.py/.js/.java S3. Commit you changes and push your local branch to remote**

S1: Create branch from master- **git branch junkTest**

S2: Create file “junk”-  **vim junk.py**

S3: Add and commit  **git add junk.py and git commit –m “message”**

Push to remote – **git push origin junkTest**

Git revert HEAD -2 (number of commit you want to remove) will remove junk.py both form remote and local.

**[Ques]: 3**

S1: Create branch from master- git branch squash

S2:Create file “main” file- **vim main.py**

S3:Commit a file git commit –m “Create file main.py”

S4: Create empty file- **echo >>abc.py**

S5:Commit file- git ac –m “Add empty file abc.py”

Push: **git push –u credential**

# Using squash commit which combine the group of commit and do an another commit to hide the previous commit.

Git rebase –i HEAD ~2

Set the peak

Select the steps you want to squash by s

Q-3

**Git pull**- It is used to update the local machine repository, if any changes occur in remote repository.

**Git merge**- It is used to merge the development branch to master branch.

Q-4

**Git pull**- when multiple user commit their changes on a single repository sequence is not maintained.

**Git Pull** --rebase- It gives the clean log and commits are synchronised .