

Working with Gitlab as a Developer



Table of Contents

Installing Git (One Time Setup)	4
Confirming the Installation	6
Initializing Git (One Time Setup)	7
Importing (Cloning) a Remote Project In your PC.....	8
Day-to-Day Workflow	11
Pull Repository.....	11
Create a Branch.....	11
Staging & Committing the Changes.....	13
Push Changes.....	14
Creating a Merge Request.....	15

Revision History

Date	Updated by	Notes
11/12/2020	Jay Shukla	Complete Version 1.0

Working with Gitlab as a Developer

This documentation will give you **detailed tutorial** about how to efficiently work with **git** & **Gitlab** on daily bases as a Developer, even if you do not know anything about version controlling.

So, let's get Started!

Before we start working with the Gitlab, lets make sure that we have git running in our machine. Open your command prompt and type,

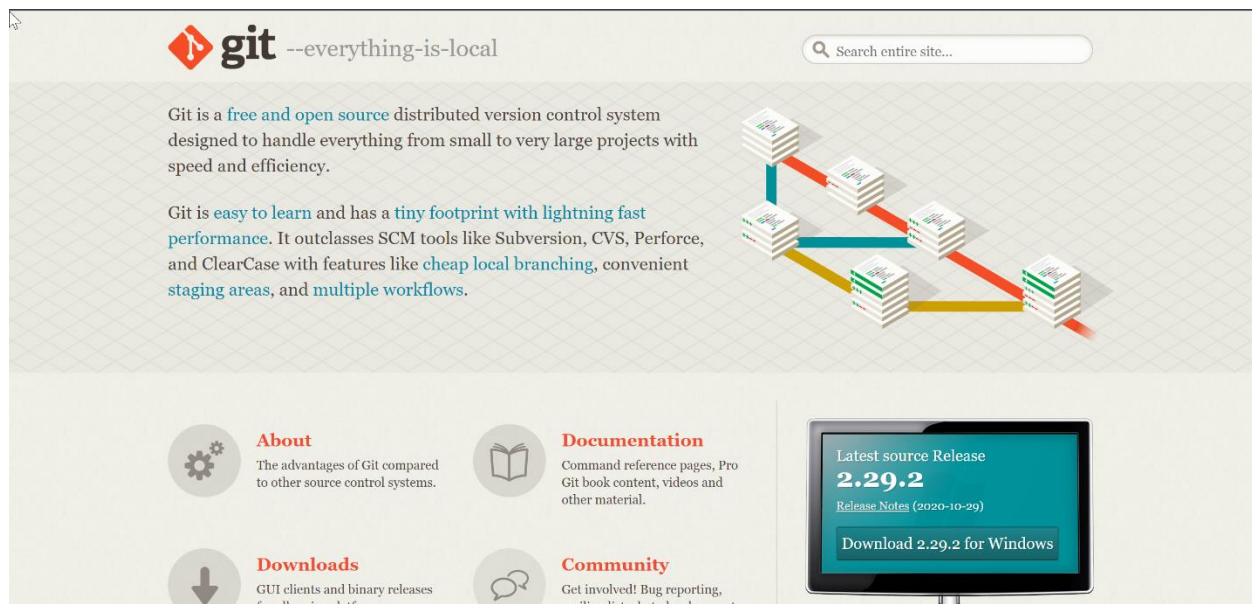
git --version

If it shows the git version then you are good to skip the Installation step, else follow the installation step to get git running on your machine.

Installing Git (One Time Setup)

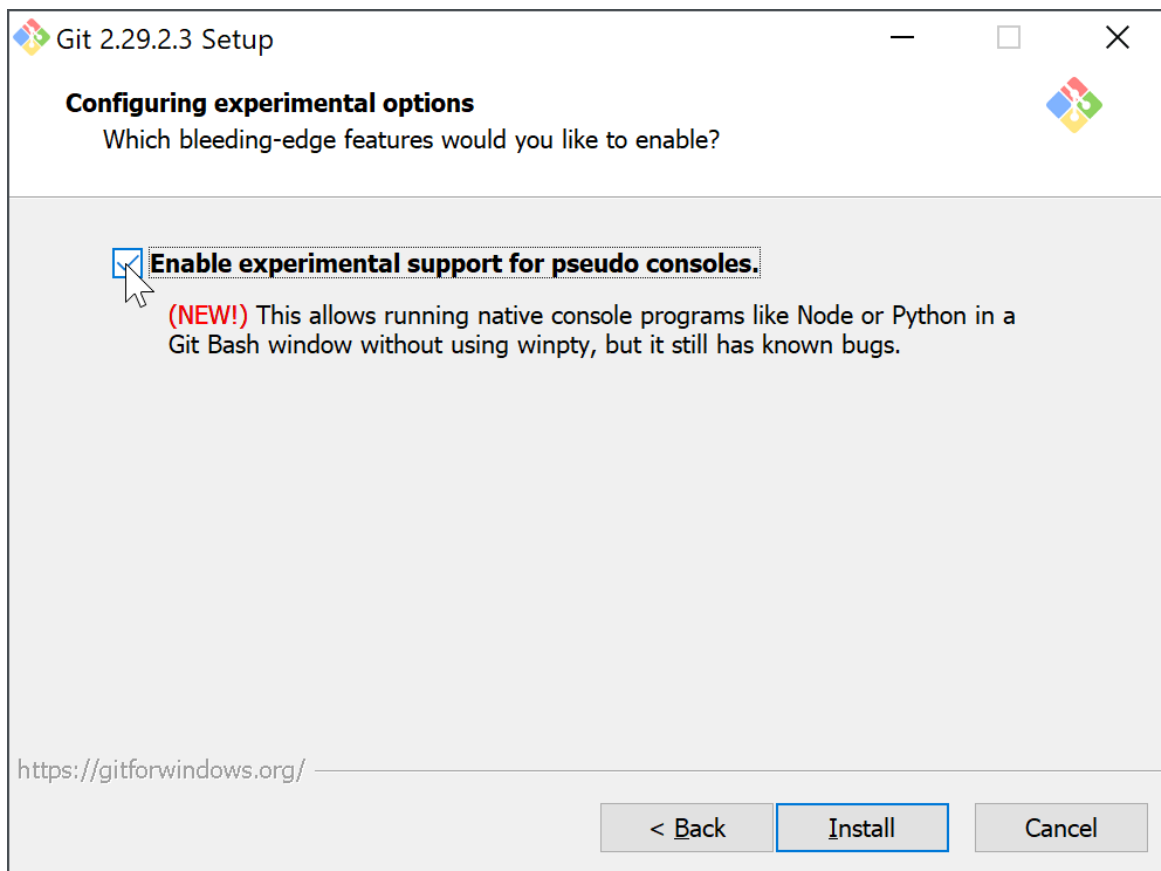
Step – 1: Downloading Git

1. Go to [this website](#).
2. You will see a screen similar to this.
3. Press the *Download* button to install the .exe file.



Step – 2: Setup

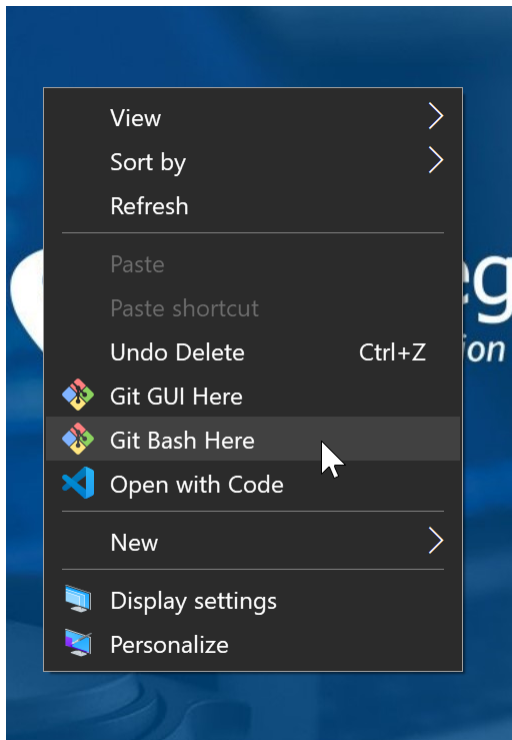
1. Now, simply run the installer.
2. Keep everything on default and click next until you see this page.
3. Tick mark **Enable experimental support for pseudo consoles.**
4. Click install to start the installation.



Confirming the Installation

Step – 1: Open Git bash

- On your desktop, Right click and select “Git Bash Here”



Step – 2: Type “git --version”, You should see some output like this.

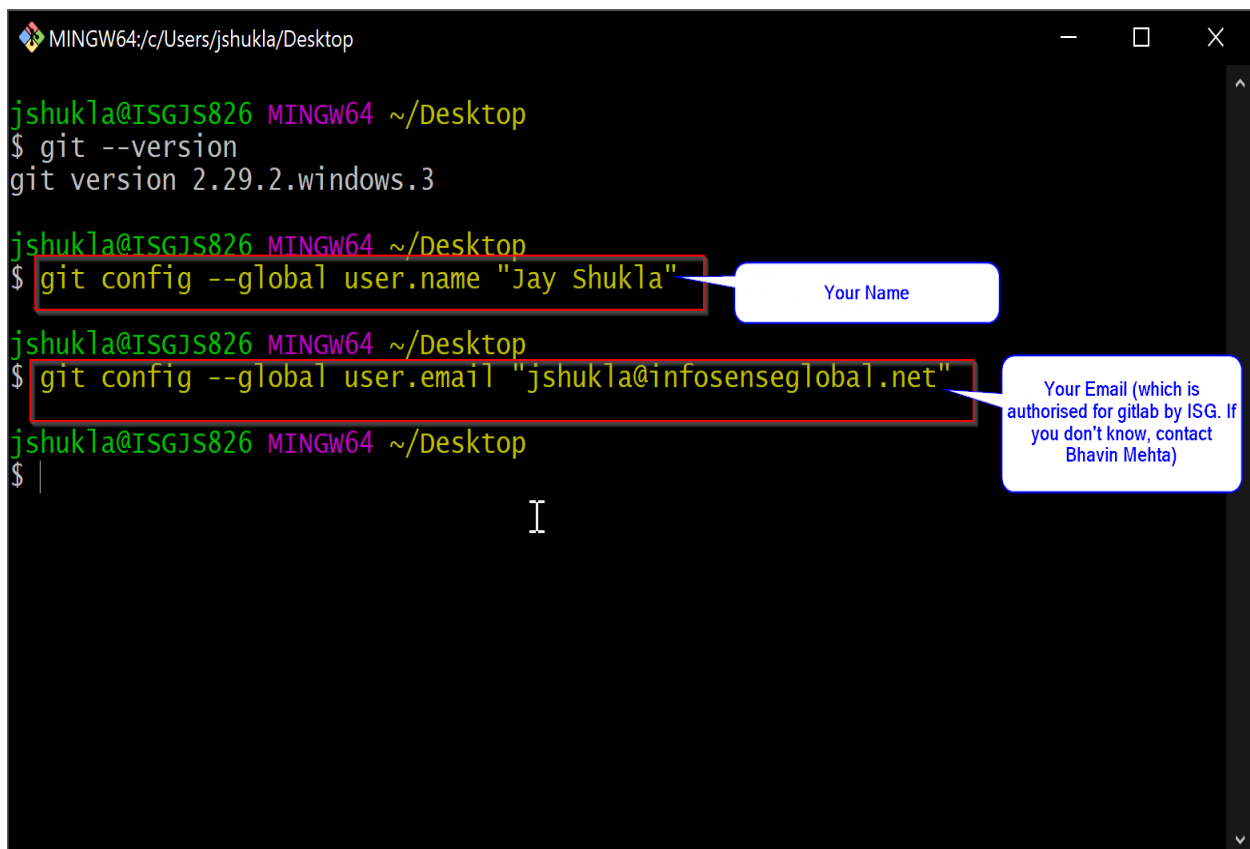
```
MINGW64:/c:/Users/jshukla/Desktop
jshukla@ISGJS826 MINGW64 ~/Desktop
$ git --version
git version 2.29.2.windows.3
jshukla@ISGJS826 MINGW64 ~/Desktop
$ |
```

Initializing Git (One Time Setup)

After the successful installation, you must initialize the **git** by providing your name & email so that **Gitlab** or any other **remote version controlling system** knows who you are.

- To provide your name write,
git config --global user.name "Your Name"
- To provide your email address write,
git config --global user.email yname@xyz.net

NOTE: Make sure your email is authorized to make commits to **IsgGroup** group. If you have any **"you are not authorized to push commits to this repository"** conflicts, then ask Bhavin Mehta to authorize your email.



The screenshot shows a Windows command prompt window titled "MINGW64: c:/Users/jshukla/Desktop". The user is jshukla@ISGJS826. The terminal shows the following commands and output:

```
jshukla@ISGJS826 MINGW64 ~/Desktop
$ git --version
git version 2.29.2.windows.3

jshukla@ISGJS826 MINGW64 ~/Desktop
$ git config --global user.name "Jay Shukla"

jshukla@ISGJS826 MINGW64 ~/Desktop
$ git config --global user.email "jshukla@infosenseglobal.net"

jshukla@ISGJS826 MINGW64 ~/Desktop
$ |
```

Two callout boxes provide additional context:

- A box pointing to the first command says: "Your Name".
- A box pointing to the second command says: "Your Email (which is authorised for gitlab by ISG. If you don't know, contact Bhavin Mehta)".

Importing (Cloning) a Remote Project In your PC

Step - 1: Create New Directory

- **This step is not necessary** but it's good to have some folder where all your git repositories will be stored.
- I usually have a folder named **Gitlab Repos** in my PC where I clone all my repositories.

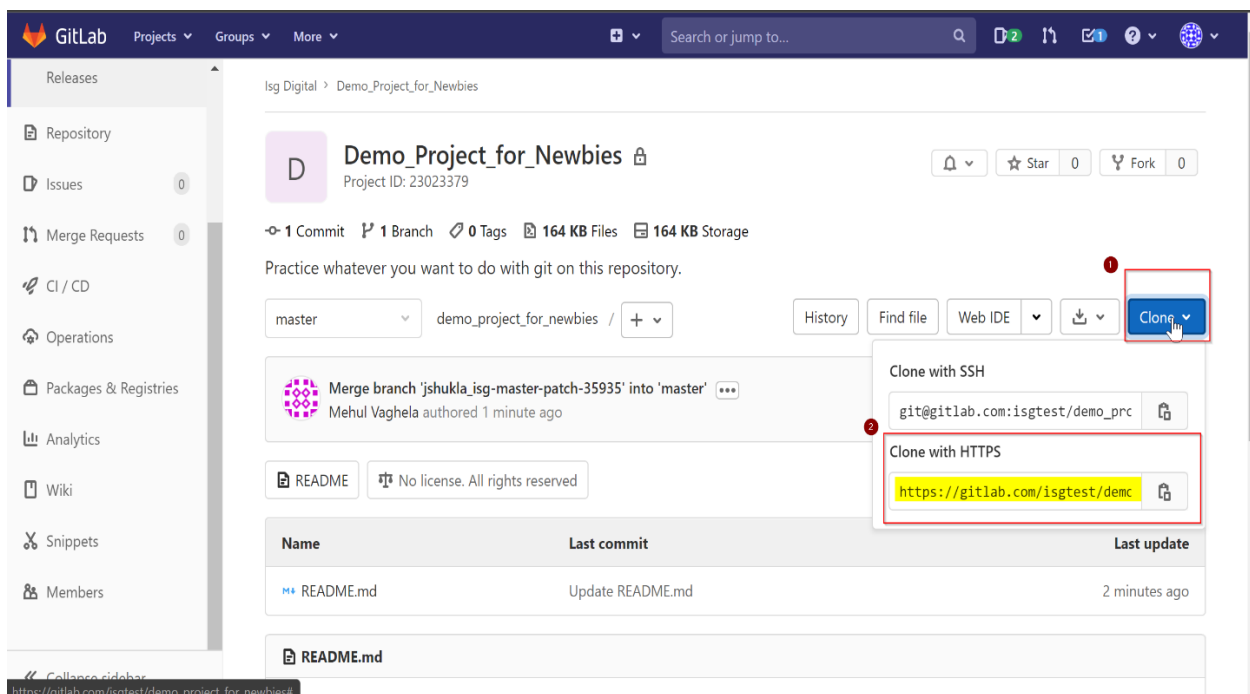


Step – 2: Cloning a Repository

Usually, all our code is stored in a **remote repository**, so first we have to import the code in our pc, this process is called **Cloning a project** and it happens only one time.

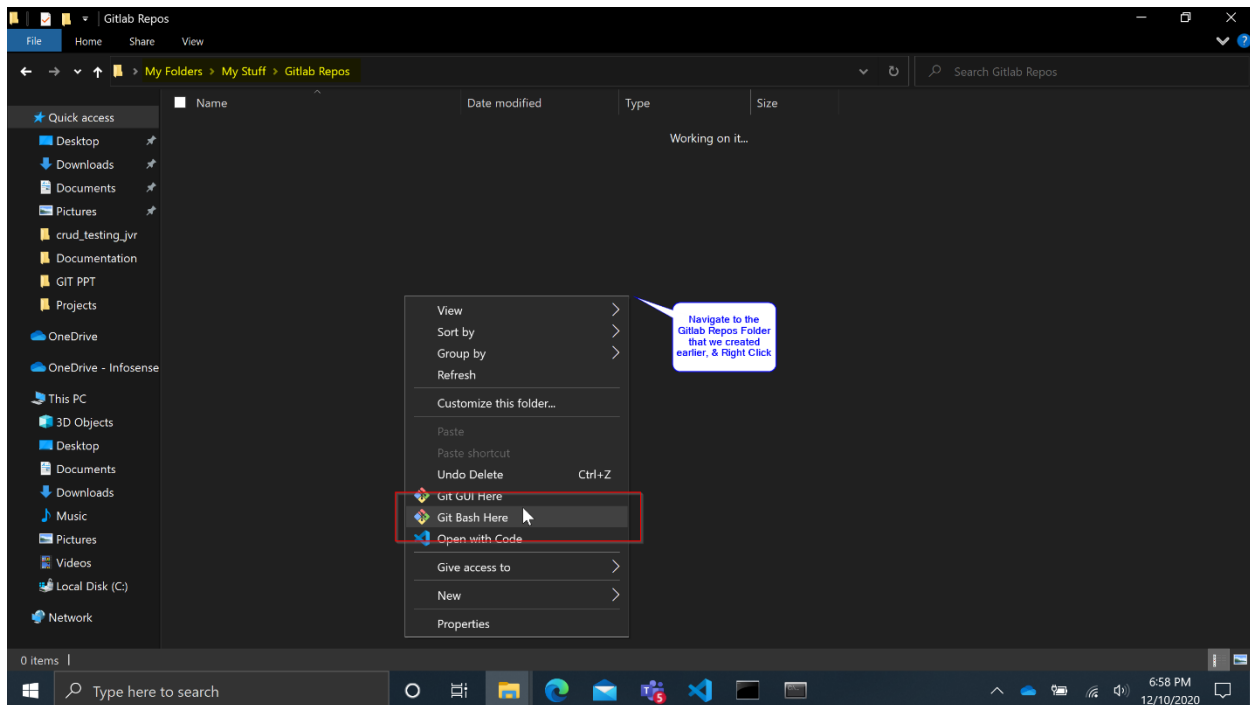
I have created a **practice repository** where you can practice all the things.

- Go to [this link](#).
- Press the clone button and then copy the **https** link.



Step – 3: Open Git Bash

- Navigate to the “Gitlab Repos” folder we created earlier (or to any place where you want to store your repo)
- Right click and open git bash

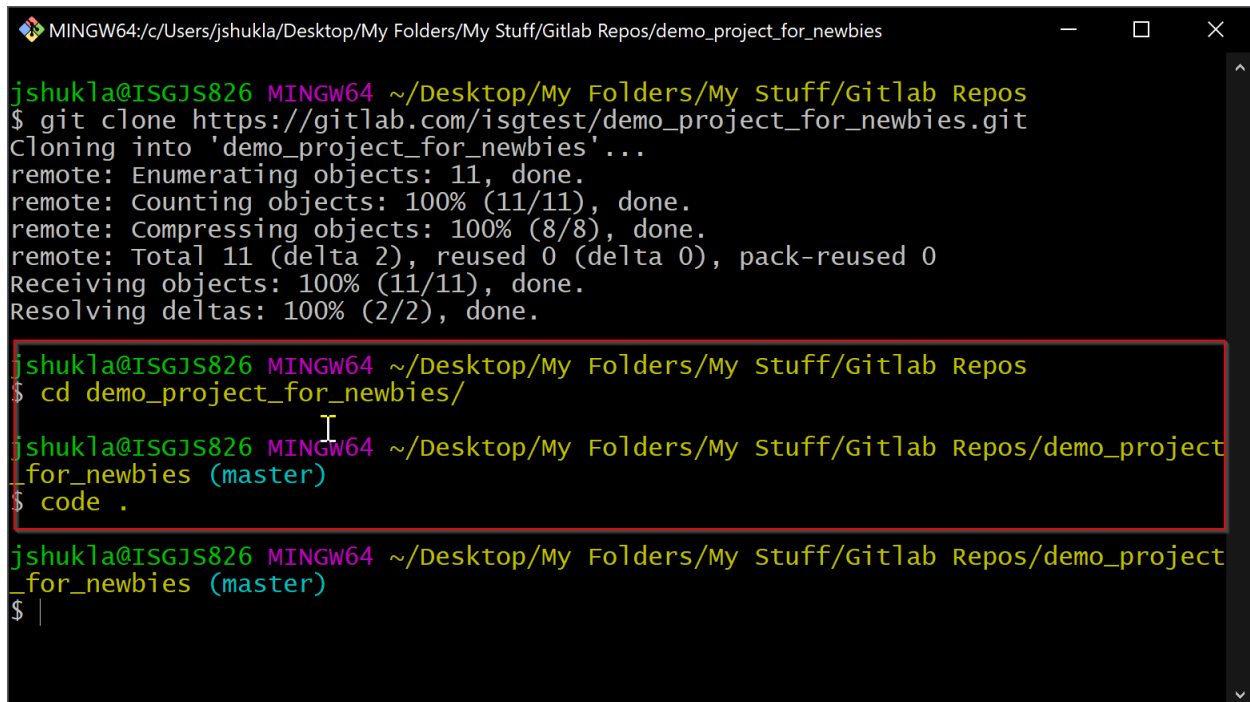


Step – 4: Clone the Repo

- Type “git clone https_link_you_copied” and press enter, a new Repository will be created.
- Now you can open that repository in your **vs code**.



```
MINGW64:/c/Users/jshukla/Desktop/My Folders/My Stuff/Gitlab Repos
jshukla@ISGJS826 MINGW64 ~/Desktop/My Folders/My Stuff/Gitlab Repos
$ git clone https://gitlab.com/isgtest/demo_project_for_newbies.git
```



```
MINGW64:/c/Users/jshukla/Desktop/My Folders/My Stuff/Gitlab Repos/demo_project_for_newbies
jshukla@ISGJS826 MINGW64 ~/Desktop/My Folders/My Stuff/Gitlab Repos
$ git clone https://gitlab.com/isgtest/demo_project_for_newbies.git
Cloning into 'demo_project_for_newbies'...
remote: Enumerating objects: 11, done.
remote: Counting objects: 100% (11/11), done.
remote: Compressing objects: 100% (8/8), done.
remote: Total 11 (delta 2), reused 0 (delta 0), pack-reused 0
Receiving objects: 100% (11/11), done.
Resolving deltas: 100% (2/2), done.
jshukla@ISGJS826 MINGW64 ~/Desktop/My Folders/My Stuff/Gitlab Repos
$ cd demo_project_for_newbies/
jshukla@ISGJS826 MINGW64 ~/Desktop/My Folders/My Stuff/Gitlab Repos/demo_project_for_newbies (master)
$ code .
jshukla@ISGJS826 MINGW64 ~/Desktop/My Folders/My Stuff/Gitlab Repos/demo_project_for_newbies (master)
$ |
```

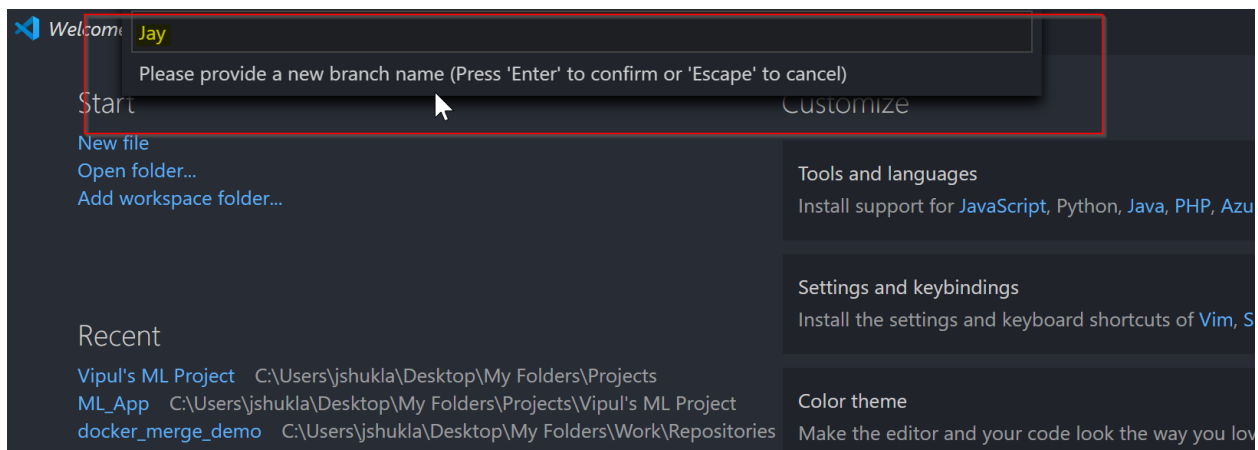
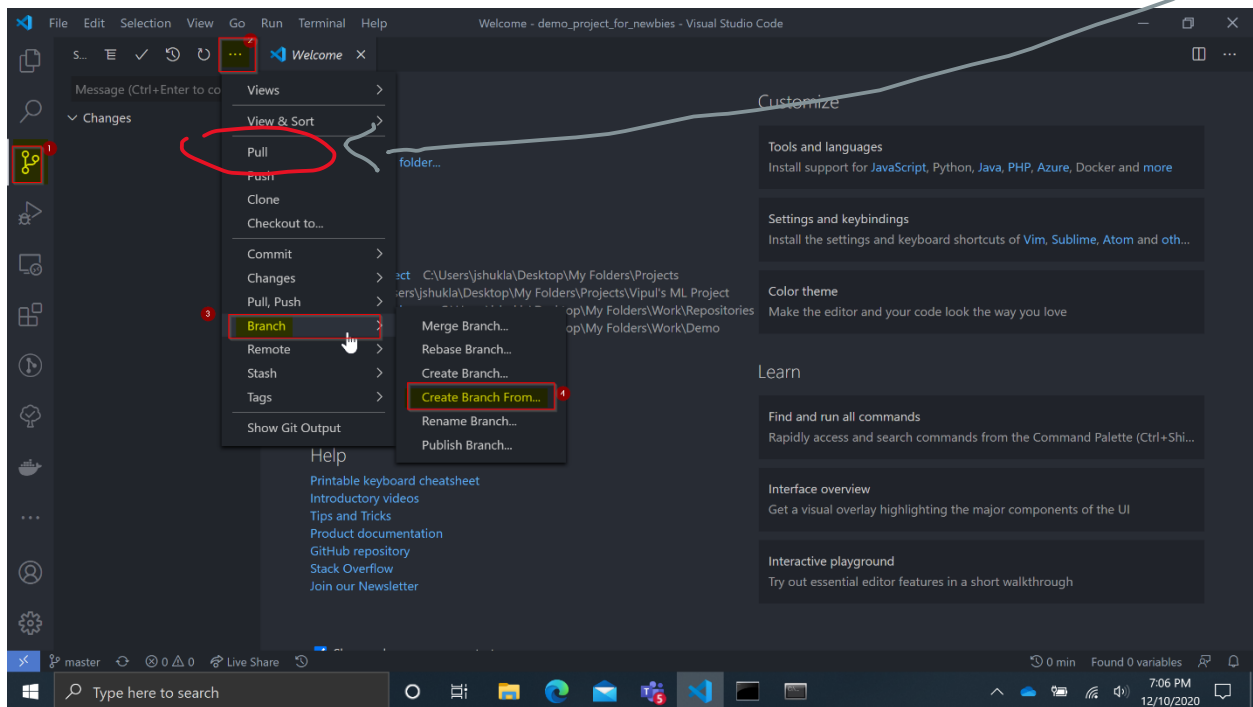
Day-to-Day Workflow

This is the most important section of this tutorial and you will encounter these tasks on the daily bases. So, let's get started.

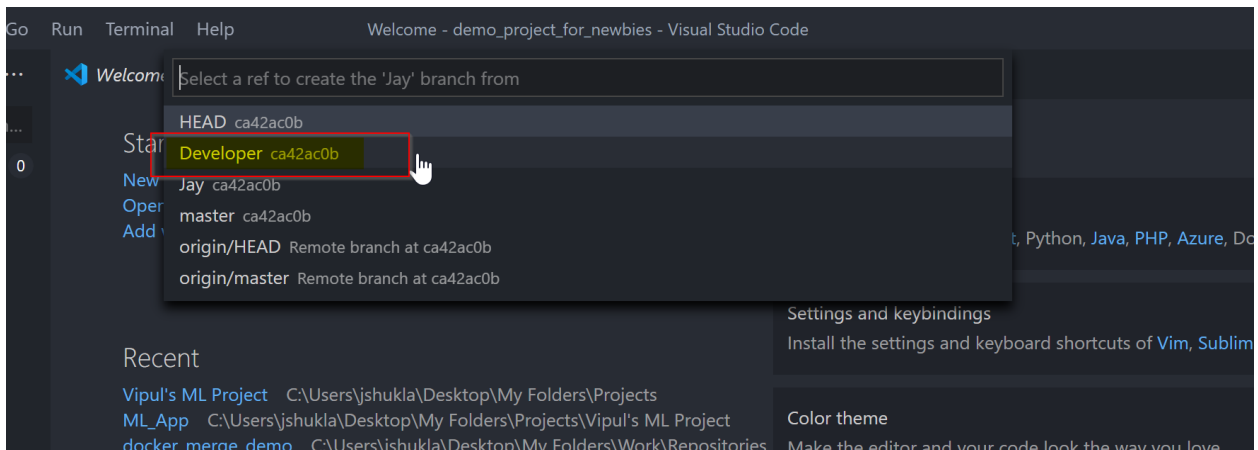
Step - 0: Remember before doing anythin always do a pull. Press 3rd button in the dropdown.

Step – 1: Creating a Branch IF IT NOT ALREADY EXISTS

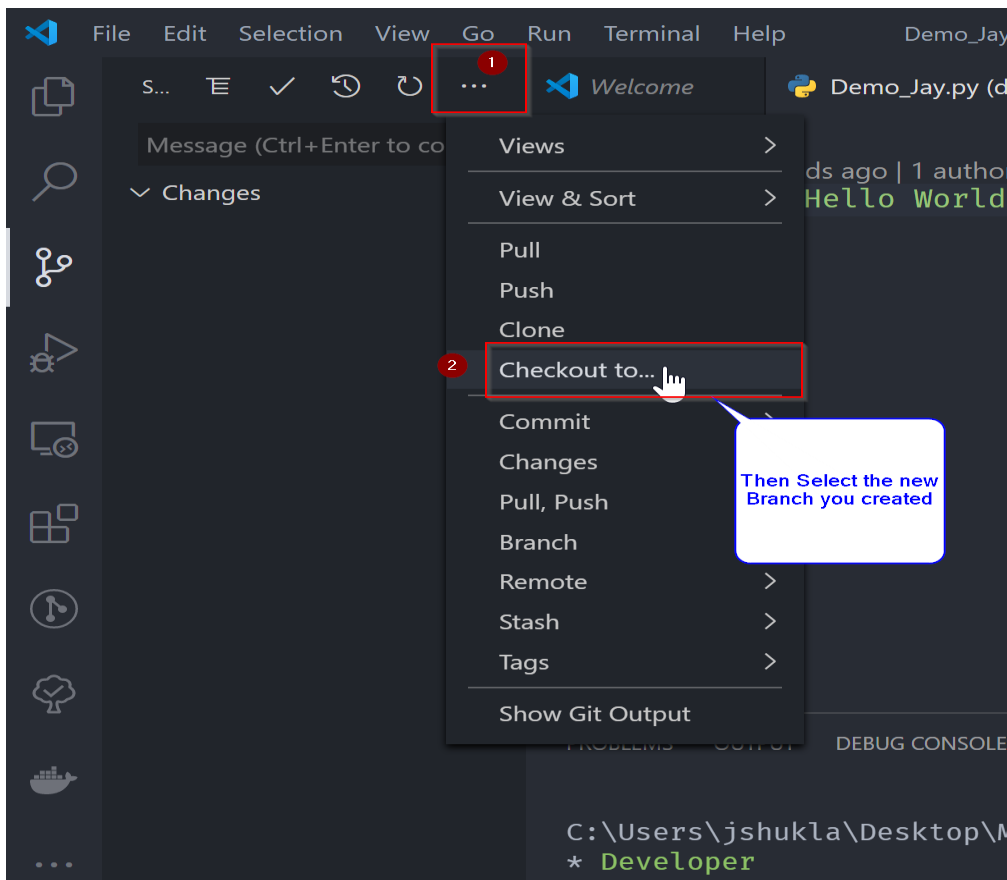
- If The Branch With Your Name Does Not Exist Then, First You Will Need To Create A Branch Of Your Name Before Making Changes To the Repository.
- To create a new branch, follow the steps sown in the below figure.



- Select Developer Branch. (You are creating a branch from the Developer branch, so you will have a **copy** of all the files in the Developer branch, so that whenever you make changes in your branch the *Developer branch stays unchanged.*)



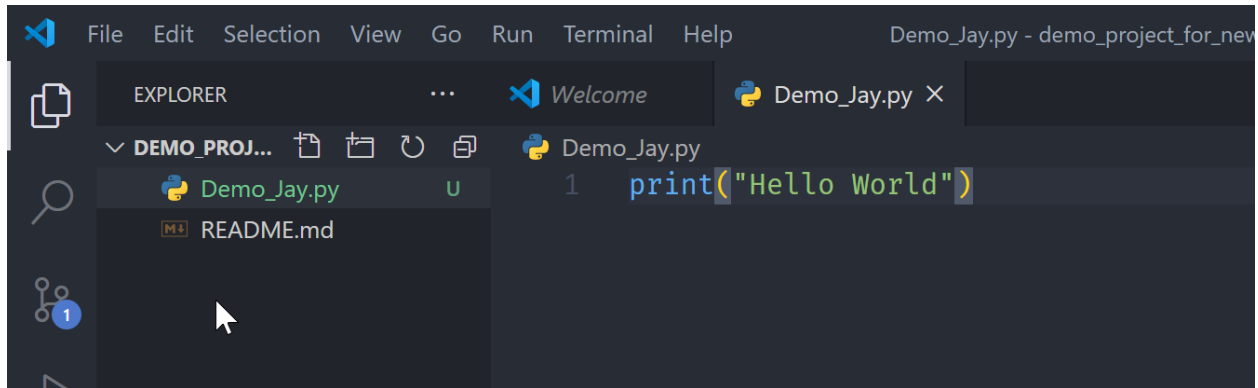
- Then now get into (Checkout to) the branch that you just created, to make changes in that branch.



NOTE: In case you want to see *how many branches are there* then press **CTRL+SHIFT+ `** to open a terminal and then type **git branch**.

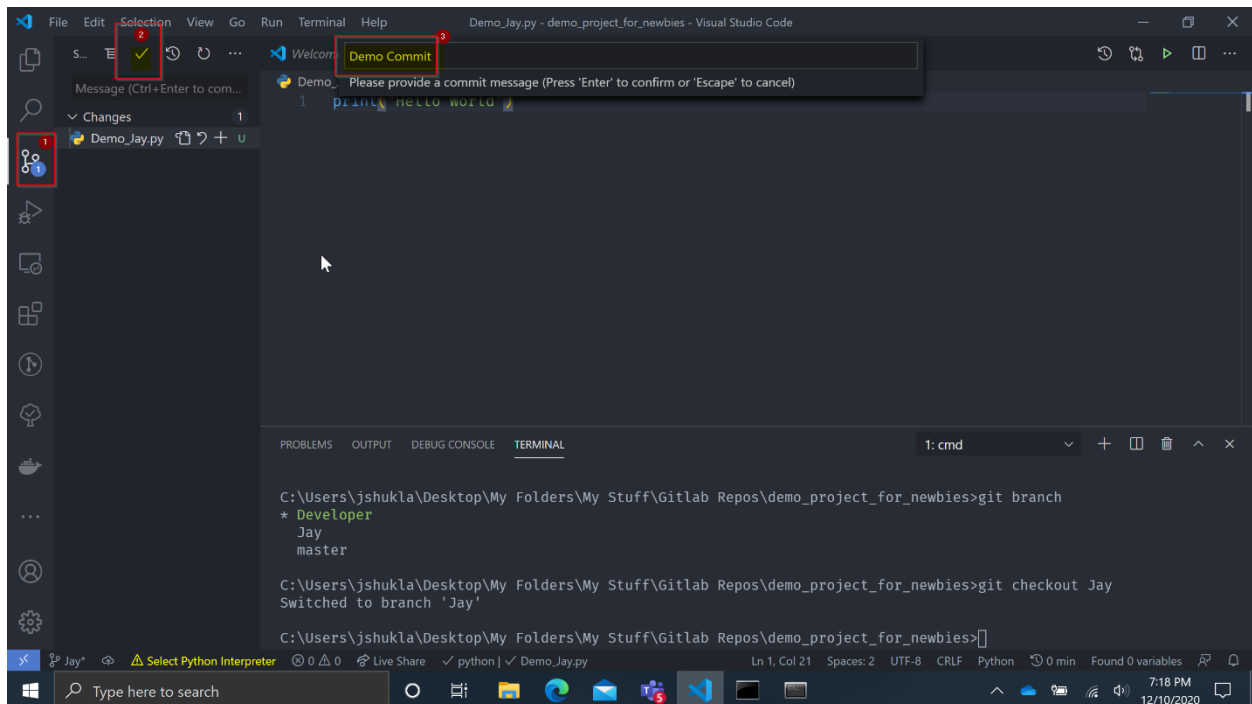
Step – 2: Edit the Files

Now that you are in your branch, you are free to make whatever changes you want to make in the repository. YEY!



Step – 3: Staging and Committing the Changes

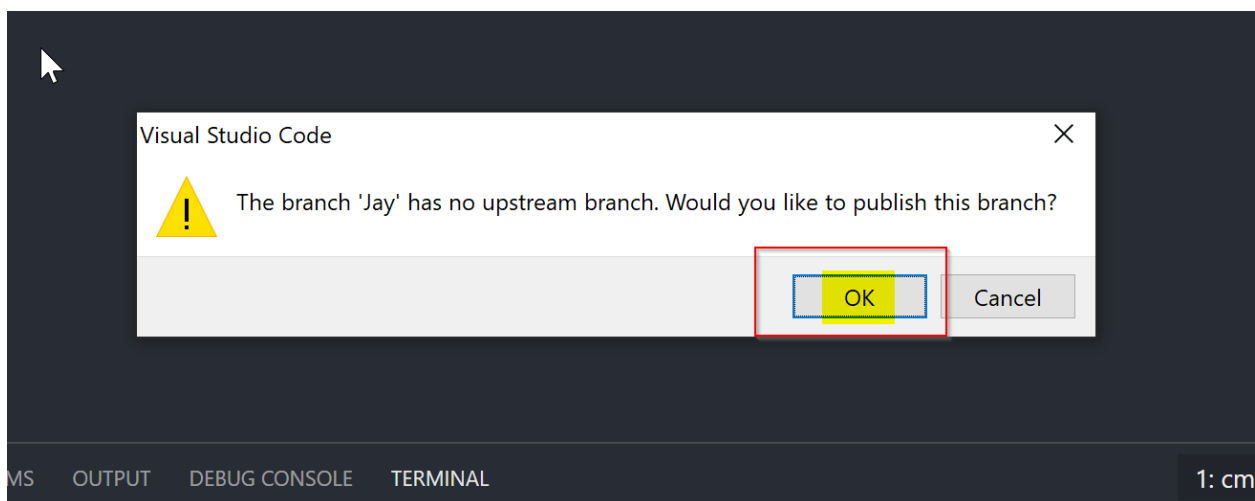
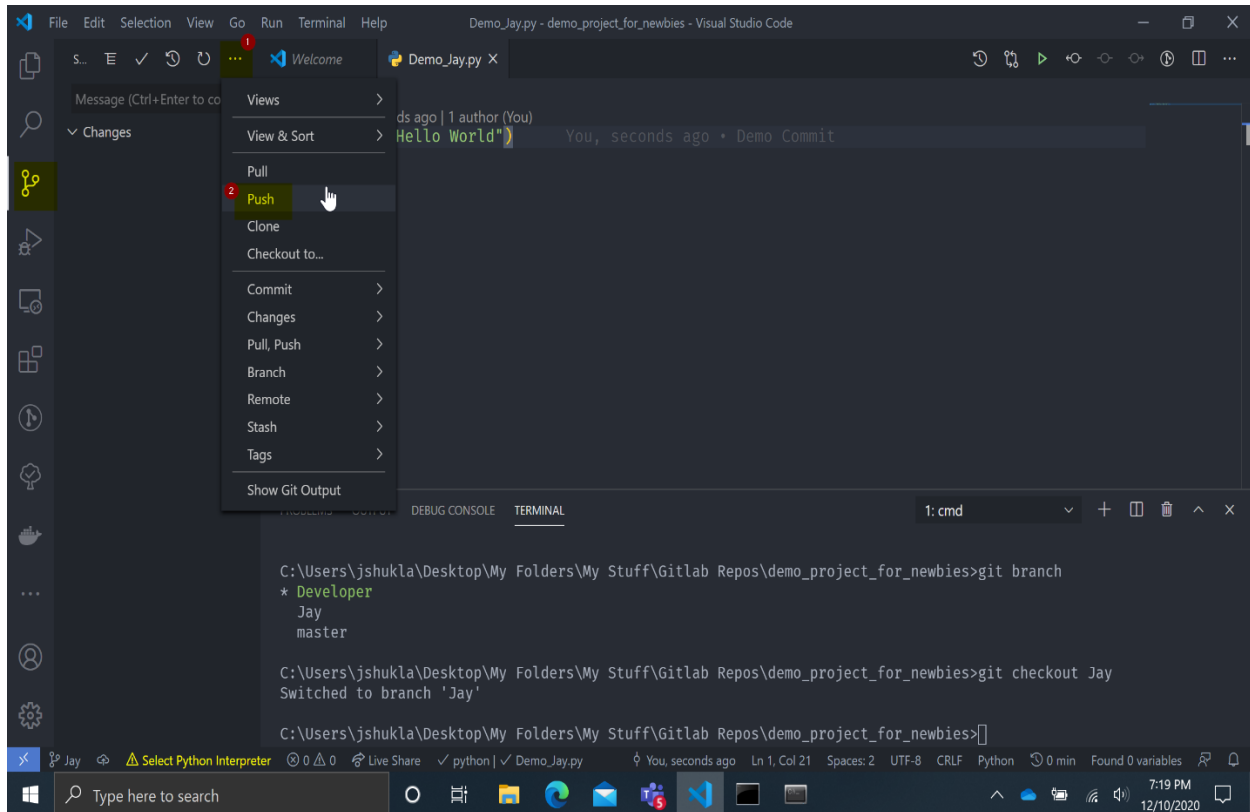
- Follow the steps in below figure to stage and **Commit** your changes.
- **NOTE:** If some pop-up tells you that “The changes are not staged, stage & commit them directly?” then press “Yes, do it every time”
- Finally Type the commit message in the pop-up, this will be the **name of your commit**.



Step – 4: Push Changes to the Remote Repository

Now that the changes are *securely stored* in your *local machine*, we would like to push the changes to the **remote** repository so that your Team can see them.

- Follow the steps in below figures to push your changes to the remote repository.
- **NOTE: If Some conflict happens here then you will have to pull the Jay branch and then push again. Just open terminal and type `git pull branch_name`.**



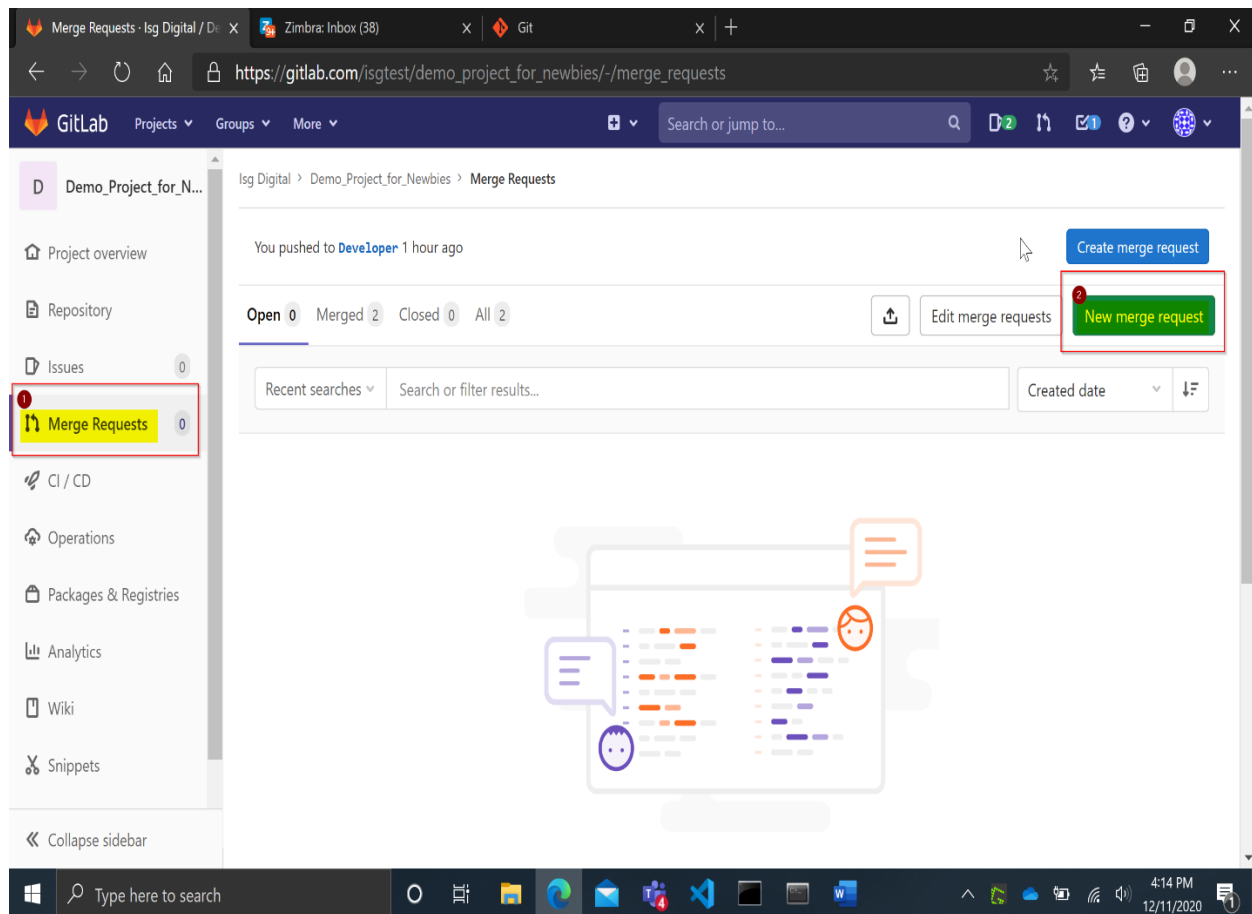
Step – 5: Making Changes Again

If you want to make some changes again, then **You Don't Have to Create a New Branch Now**. Just make the changes again and **repeat steps 0,3 & 4** to commit and push new changes.

Step – 6: Merge your Changes in Developer branch

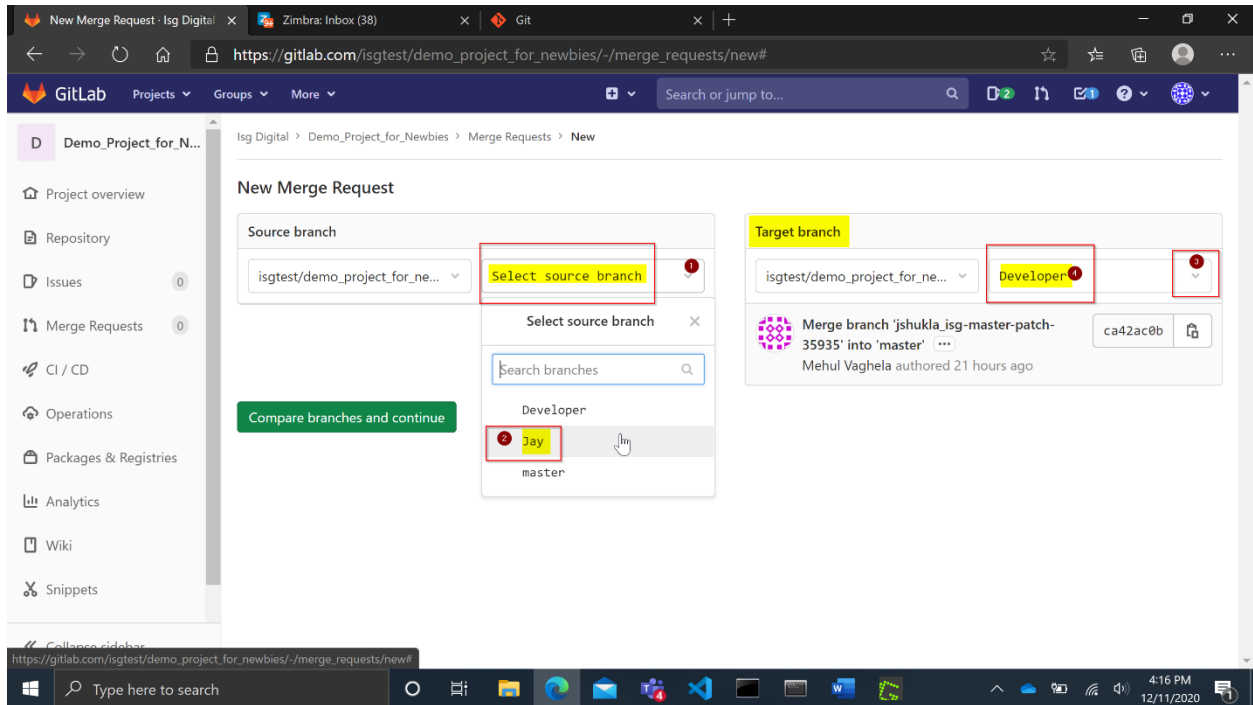
You will need to **merge** your Named branch into Developer branch to merge your changes with the changes done by your team.

- Go to [the Gitlab Repository](#) .
- Follow Below Steps to Create a merge request.



- Then on next page they will ask you to select branches.

- In our case the source branch will be **your branch** and the target branch will be the **Developer** branch.
- **Carefully Select Developer as A target branch, If you don't see a developer branch then that branch must have been deleted. In that case contact some one who is handling the Gitlab Project.**



- Now on next page simply press **Submit merge request**.

Approval rules	Approvers	No. approvals required
	Any eligible user ?	0
Tip: add a CODEOWNERS to automatically add approvers based on file paths and file types.		
Merge options	<input checked="" type="checkbox"/> Delete source branch when merge request is accepted. <input type="checkbox"/> Squash commits when merge request is accepted.	
<div>Submit merge request</div>		
Commits 2	Changes 1	
11 Dec, 2020 1 commit		

Congrats, now you know everything you need to know for working with Gitlab!