**BIOIQ Integration**

**with GitHub**

Contents

[Creation of GitHub account 3](#_Toc29837393)

[Git Setup in Local Machine 6](#_Toc29837394)

[Process setup for export and Import script 10](#_Toc29837395)

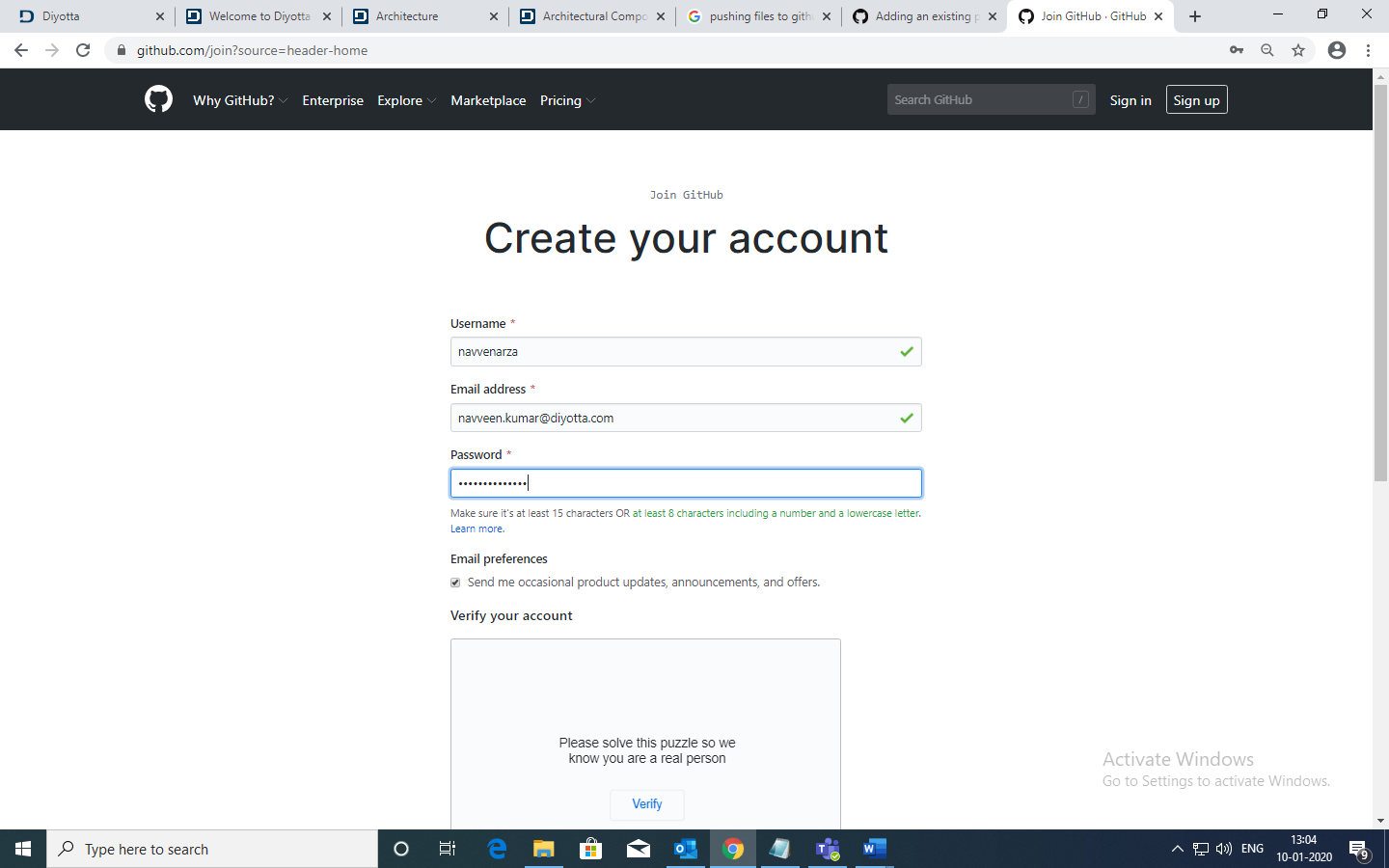
[Export script Configuration 11](#_Toc29837396)

[Import script Configuration 14](#_Toc29837397)

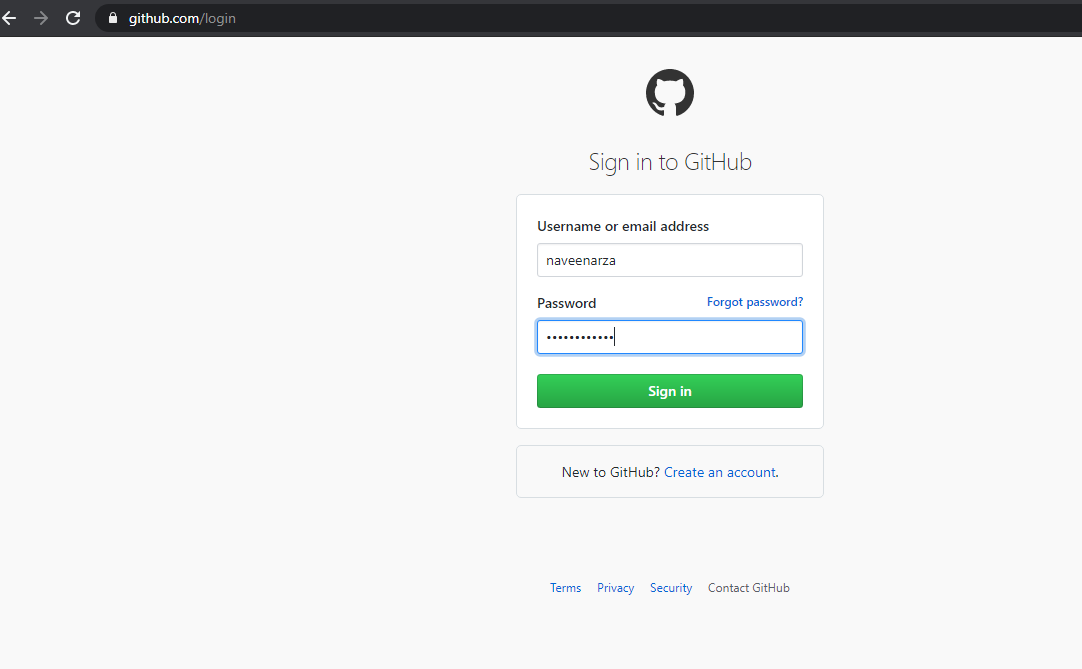
# Creation of GitHub account

**Step 1:** Navigate to<https://github.com/>

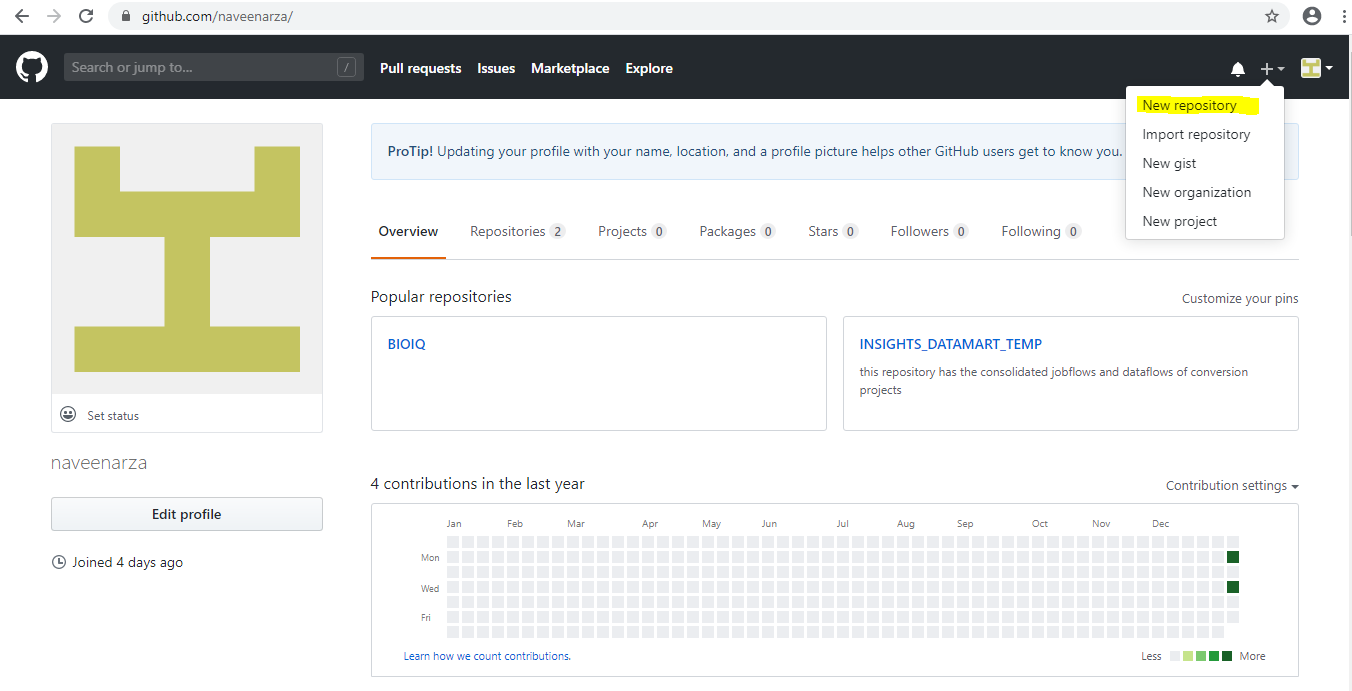
**Step 2:**  Create an account with your email.



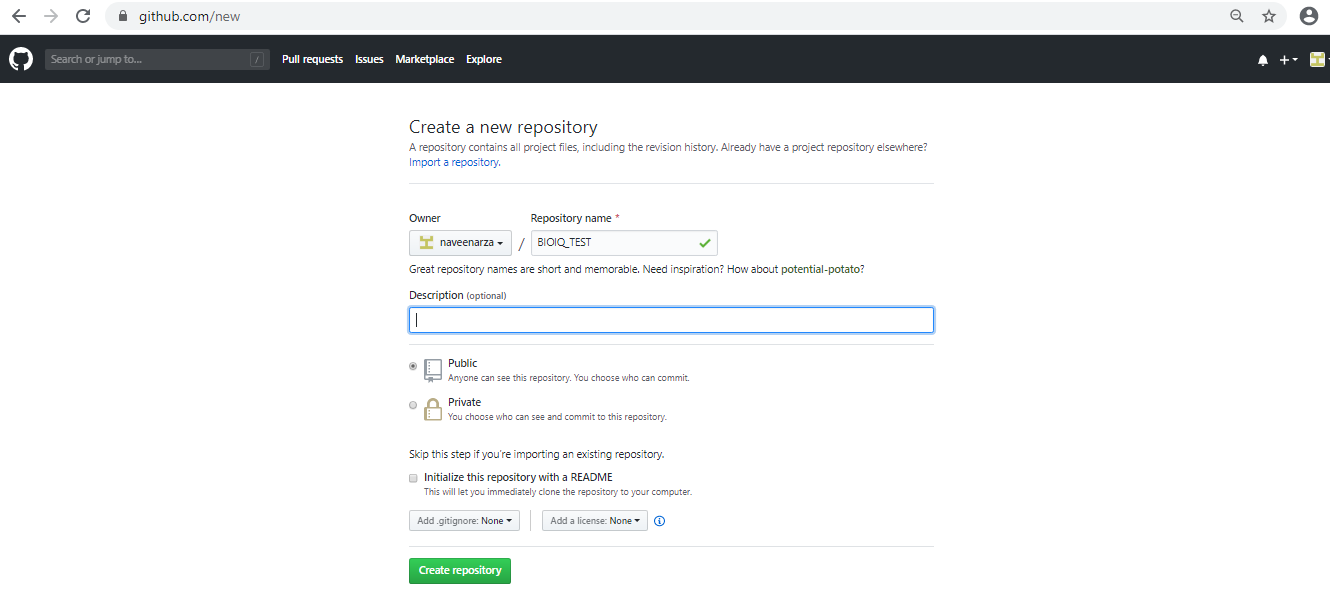
**Step 3:** Login to your GitHub account



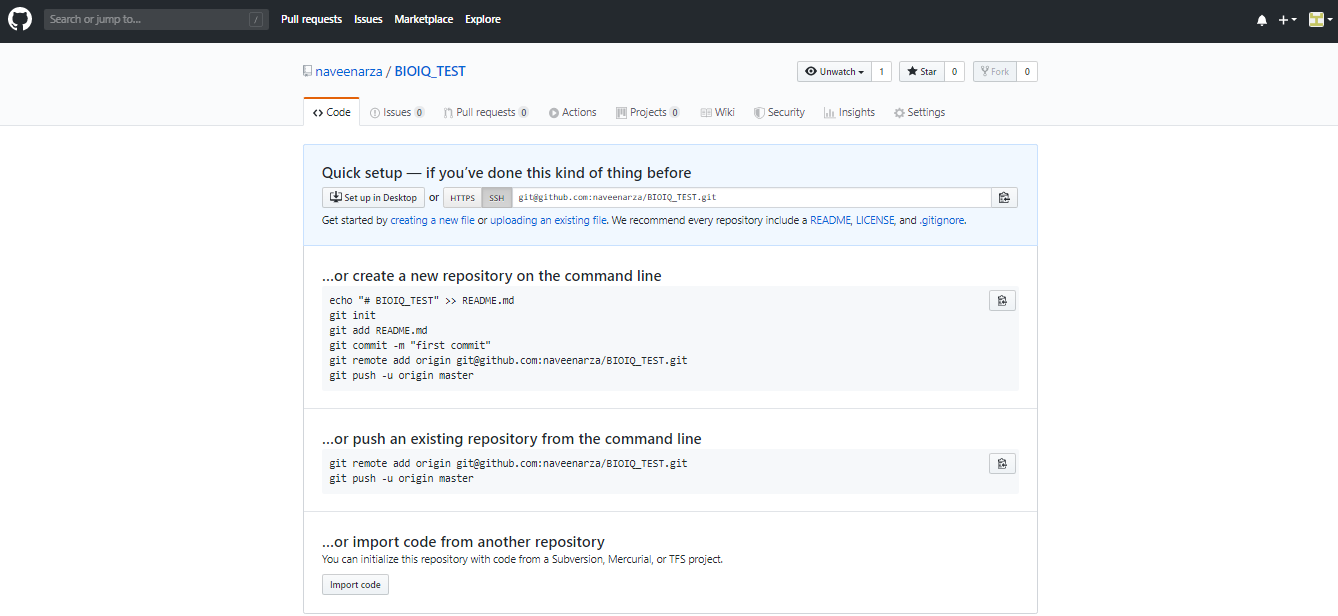
**Step 4:** Create a new repository as shown below



**Step 5:** Enter the name of the repository and click on Create Repository button.



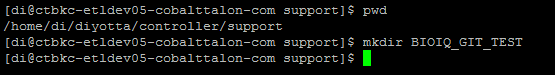
**Step 6:**  It will be navigated to the screen where we can Create, Edit or Upload our code.



# Git Setup in Local Machine

**Step 1:** Create a directory in your local machine where we will place all the code. We will convert this local project to a Git repository in the further steps.

**Syntax: mkdir <directory\_name>**

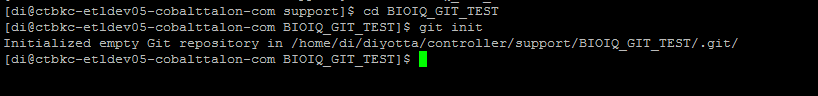


In the above step we have created a directory BIOIQ\_GIT\_TEST in the controller path where all the code that belongs to the project will be placed.

**Step 2:** Change the current working directory to the project directory that we created in the step 1 and initialize the local repository as a git repository.

**Syntax: git init**

* command initializes the local repository as a git repository.

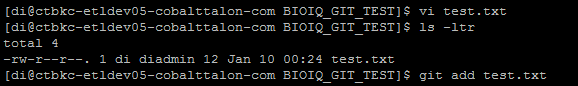


**Note:** If the project directory already exists then proceed from step 2.

**Step 3:** Create/Place the files which needs to be pushed to the remote GitHub repository that we created. Type git add test.txt.

**Syntax: git add** **<filename>**

* command adds the files to the staging area in the local machine.

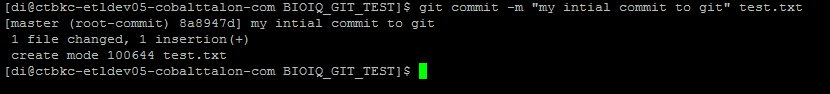


**Note:** To push all the files to the staging area use git add .

**Step 4:** Commit the files that are staged in the local repository.

**Syntax:** **git commit -m “Commit Message” <filename>**

* Commits the tracked changes and prepares them to be pushed to a remote repository.



**Note:** To commit all the files in the local repository use (git commit -m “Commit Message”) without any filename

**Step 5:** Adding Remote GIT URL to the local repository. This can be done either using **SSH** or **HTTPS** URL**.**

**SSH URL:** git@github.com:naveenarza/BIOIQ\_TEST.git

**HTTPS URL:** <https://github.com/navvenarza/BIOIQ_TEST>

**Syntax:** git remote add origin <git URL(SSH/HTTPS)>

**Command:** gitremote add origin [git@github.com:naveenarza/BIOIQ\_TEST.git](mailto:git@github.com:naveenarza/BIOIQ_TEST.git)



**Note:** if you set up ssh, then you won’t have to type your password every time you push code to GitHub. If you use the https, then GitHub prompts for password every time you.

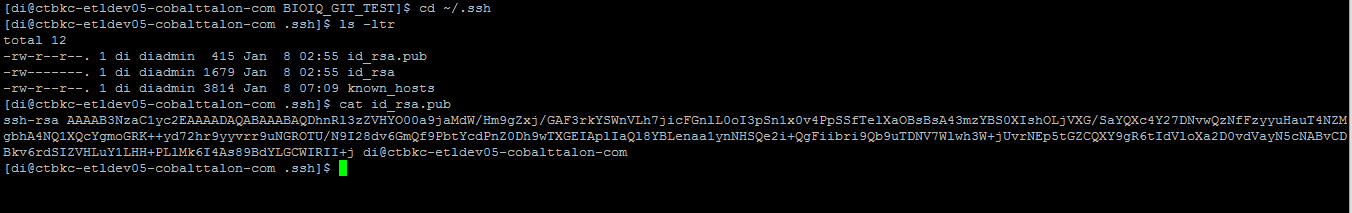
**Generation of new SSH Key:**

1. Run the below command with your git account email id.

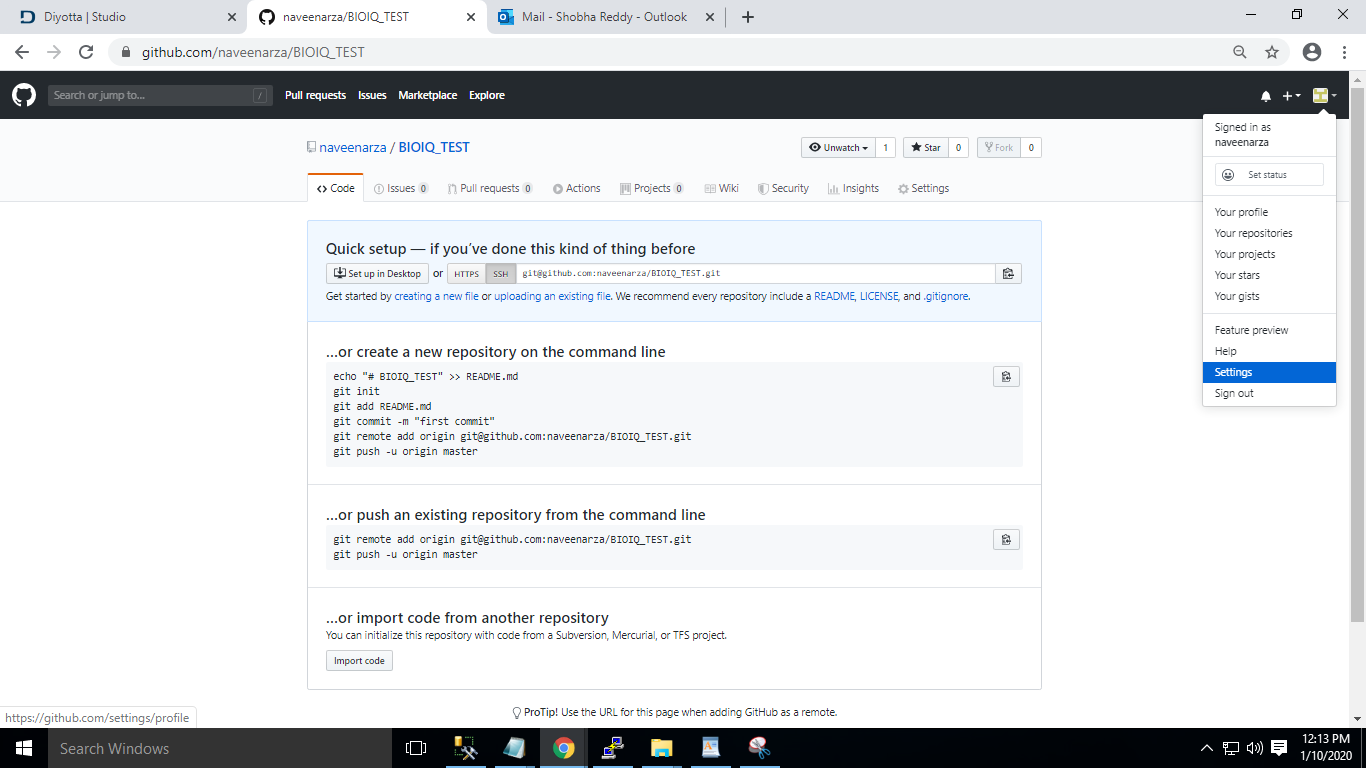
ssh-keygen -C "naveen.kumar@diyotta.com" -t rsa

1. When you're prompted to "Enter a file in which to save the key," press Enter. This accepts the default file location
2. Type command cd to check the public and private key which are generated in Step 1

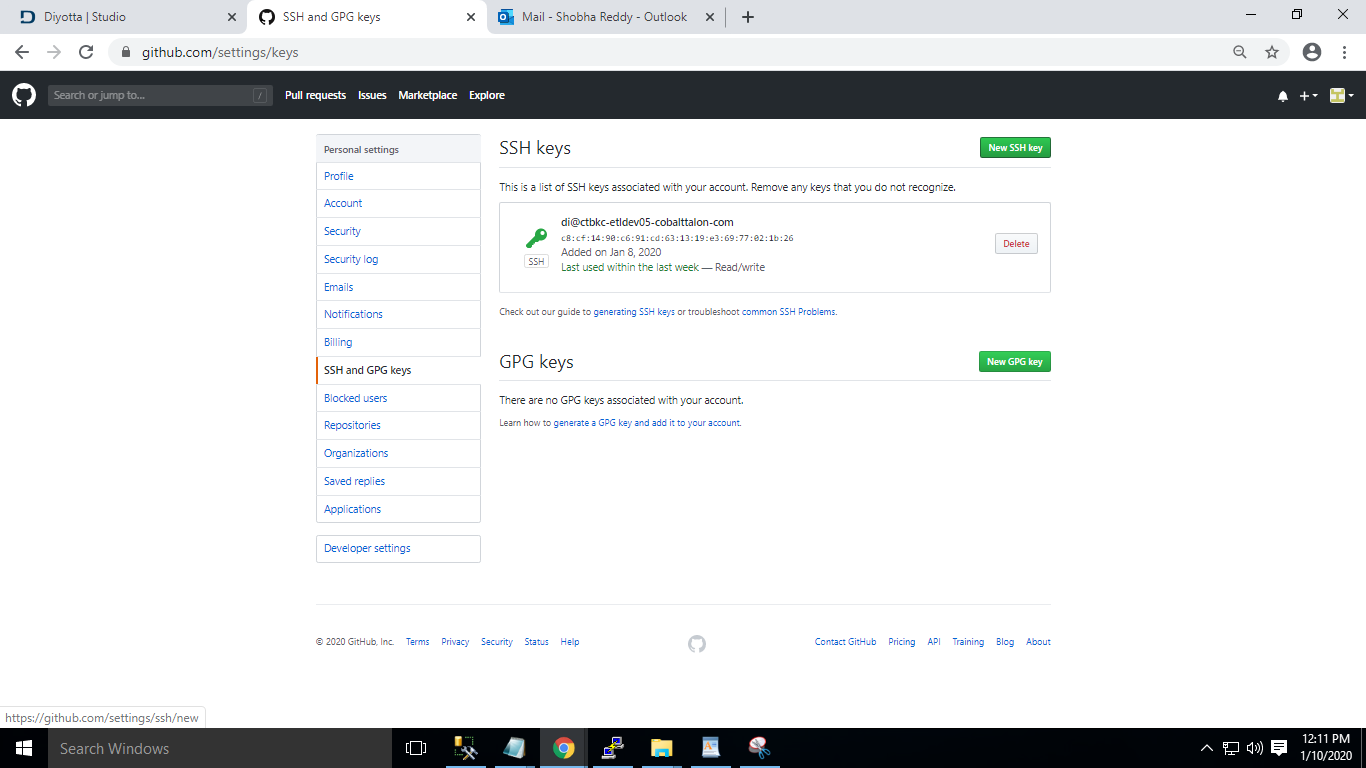
cd ~/.ssh



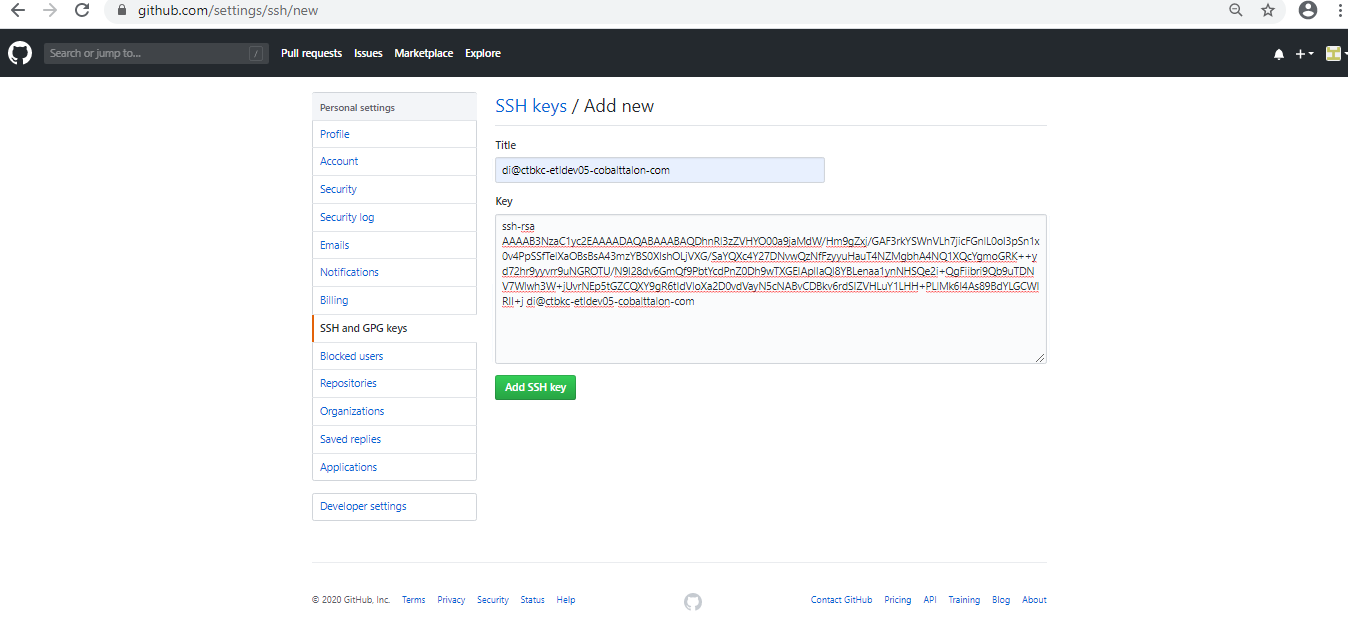
1. Copy the public key which is present in id\_rsa.pub and paste it in the SSH key section in GitHub repository. Follow the below steps to add the SSH Key to the repository.
2. Go to the settings as shown in the below screenshot.



1. Navigate to SSH and GPG Keys section and click on New SSH Key button.

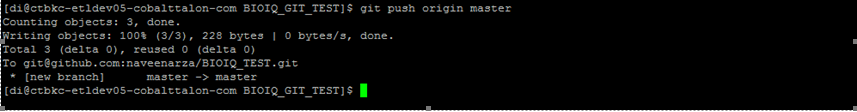


1. Paste the copied public key as shown below and click on Add SSH key.

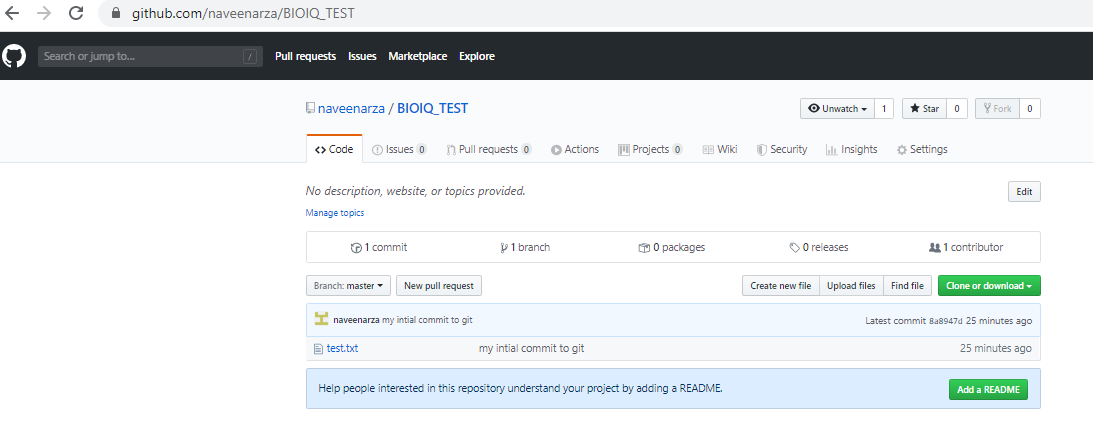


**Step 6:** Pushing the changes from local to the remote repository.

**Syntax: git push origin master**



**Step 7:** Validate whether the code is pushed to the remote Git repository.



# Process setup for export and Import script

**Prerequisites:**

Place the files in the folders mentioned below.

**diy\_export.sh Location:**

/home/di/diyotta/controller/support/git\_scripts/

**diy\_import.sh Location:**

/home/di/diyotta/controller/support/git\_scripts/

**Git Location:**

/home/di/diyotta/controller/support/BIOIQ\_GIT/

**Env.properties file location:**

/home/di/diyotta/controller/support/conf/

**diy\_functions.sh location:**

/home/di/diyotta/controller/support/lib/migration/

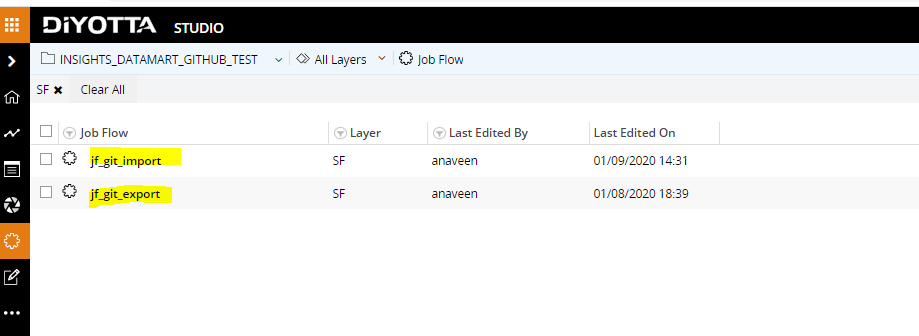
**Note 1:** Update the git location in **env.properties** file and make sure that all the paths mentioned in the properties file exists in your setup.

**Note 2:** Update the location in the final while loop end with thetemp location (/home/di/diyotta/controller/support/tmp ) in **diy\_export.sh**

**Below are the jobflows which will run the Import and Export scripts. We have to run/schedule the jobs to catchup the changes every time.**

Import the attached json files to the environment you are going to automate the export and import the GitHub process.





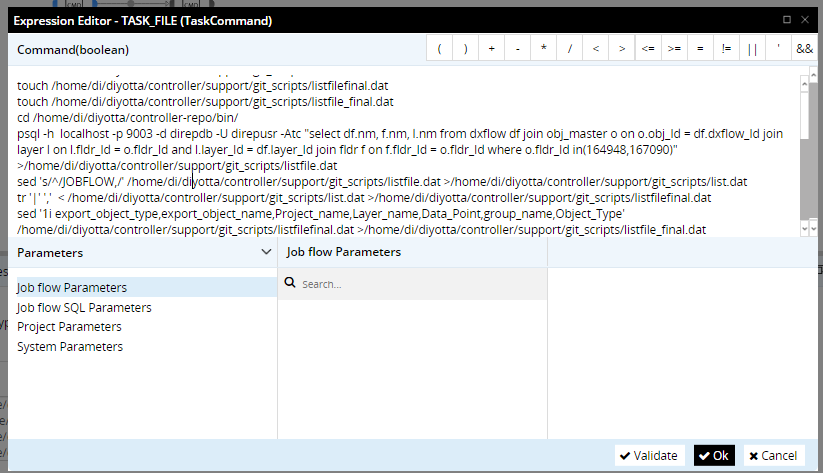
# Export script Configuration

The export script will get all json files of the jobflow from Diyotta metadata and pushes them to the remote repository.

1.**Task\_FILE in jf\_git\_export jonflow :**

Modify the Metadata details with respective environment.

Psql -h localhost -p direpdb -U direpusr -ATC “ select df.nm,f.nm,l.nm from dxflow df join obj\_master o on o. obj\_id = df.dxflow\_id join layer l on l.fldr\_id = o.fldr\_id and l.layer\_id = df.layer\_id join fldr f on f. fldr\_id=o.fldr\_id where o.fldr\_id in ($FS\_Project\_id) “



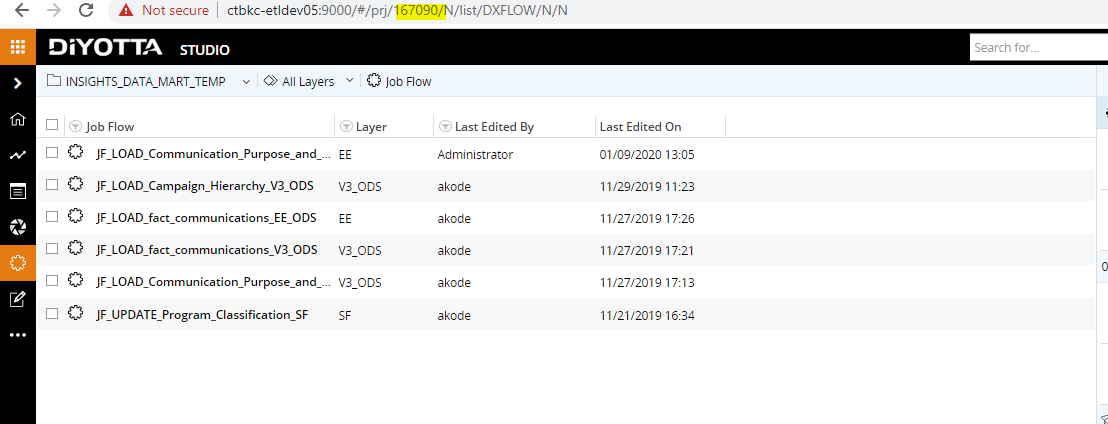
**2. Parametrized project\_ID , branch name and provide them accordingly.**

**Project\_ID :** This is unique for each project and we can get the project\_id as shown in below note.

**A screenshot of a cell phone

Description automatically generated**

**Note: We can get the project\_ID from the url as shown in below screenshot which is highlighted.**



**3.CMD\_GIT in jf\_git\_export jobflow :**

The Export script requires 6 inputs:

**1.Listfinal.dat**: This will create automatically and contains the jobflows list which we want to push the jobflows

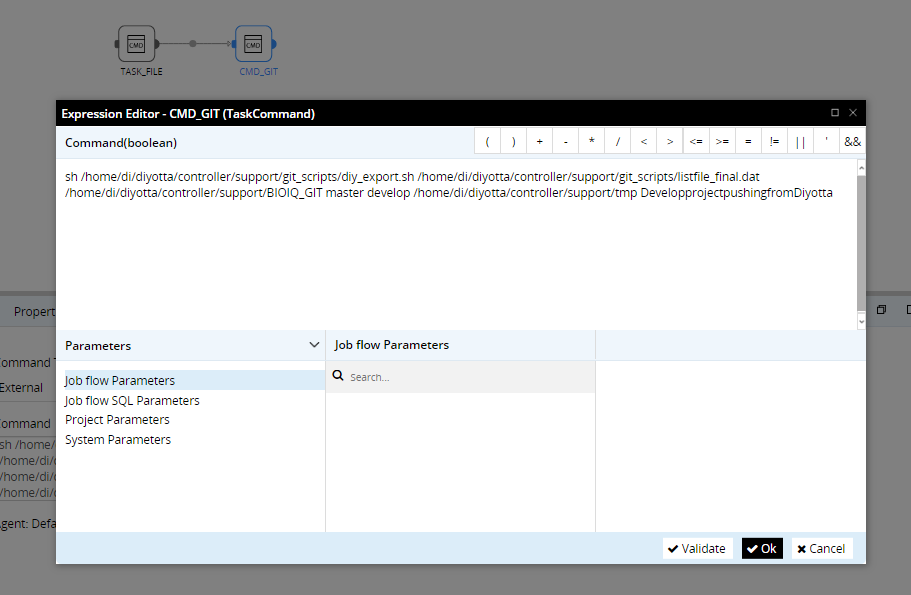
**2.Git\_Location**: local git repository location.

**3.Git main Branch**: Master will be the main branch .

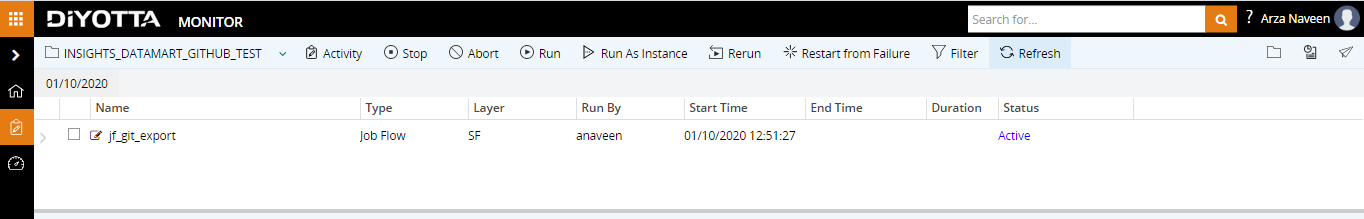
**4.Release Branch:** Name of the Branch You want to export the jobflows.

**5.Temporary location**: Temporary location for storing intermediate results.

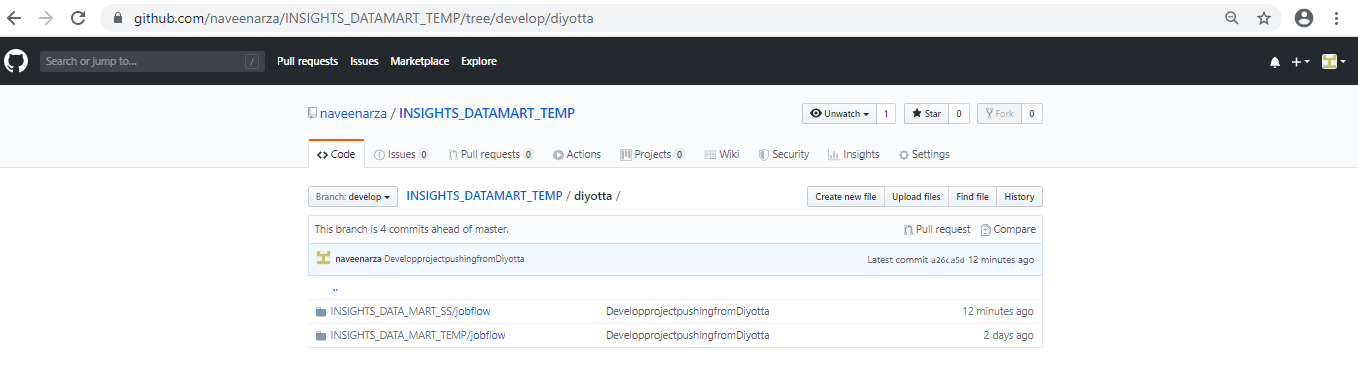
**6.Commit Message**: Message that we want to print for pushing the files.



**4.Run the jobflow Jf\_git\_export.**



**5.Validate Git hub remote repository once after the jobflow succeeded.**



# Import script Configuration

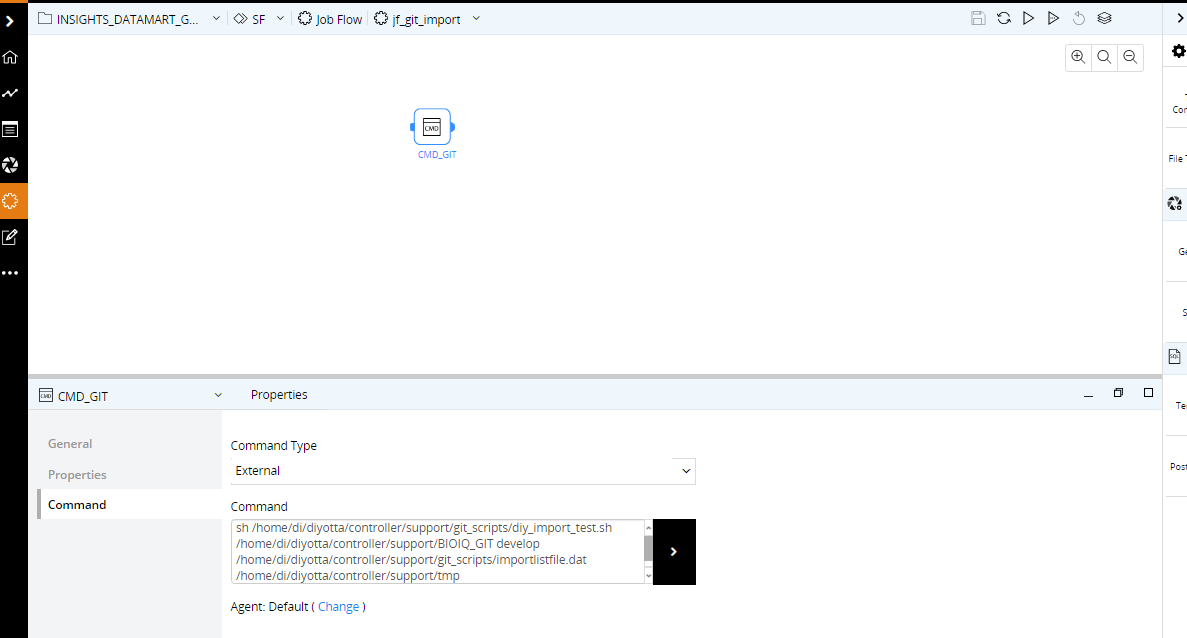
The Import script will get json file from remote repository and imports the jobflows to the specified project in Diyotta suite.

**1.We need to setup the Local Git repository in the Environment where we want to migrate the code**.

**2.CMD\_GIT in jf\_git\_import jobflow :** Modify the paths as per the local setup.

The Import script requires 5 inputs:

* **1.Git\_Location**: local git repository location
* **2. Branch\_name :** Name of the Branch You want to get the files.
* **3.importListfile.dat**: we need to specify the list of jobflows which we want to import
* **4.Temporary location**: Temporary location for storing intermediate results.
* **5.Project Name:** Name of the project for importing the jobflows.

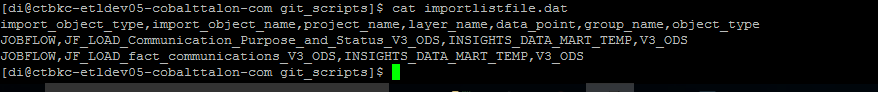


3. **Project\_name and the branch name should be passed as input .**

A screenshot of a cell phone

Description automatically generated

**Note**: Below is the format for placing the Jobflows list in **importListfile.dat**



4.**Run the Jobflow Jf\_git\_import.**

5.**Validate whether the jobflows present in the listfile.dat are imported to the respective project folder.**

