## **What is GitHub?**

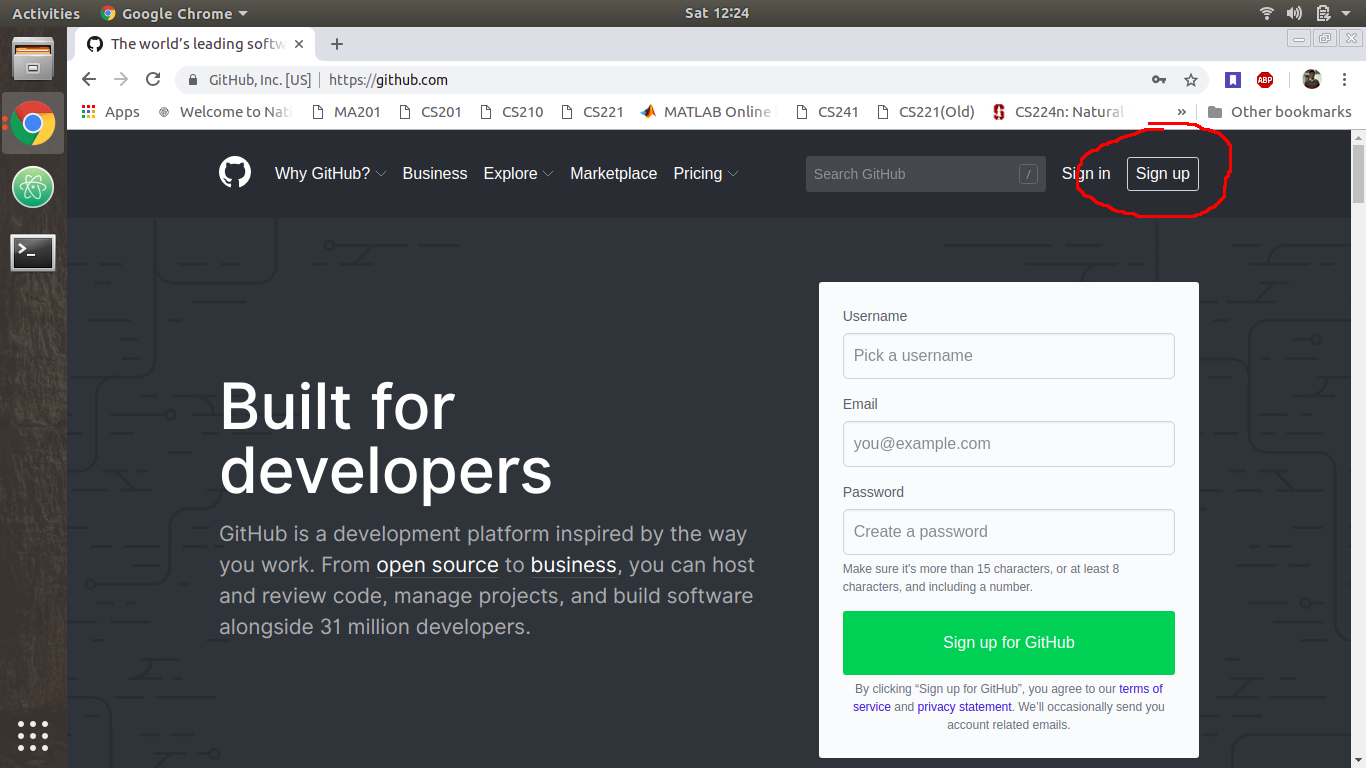
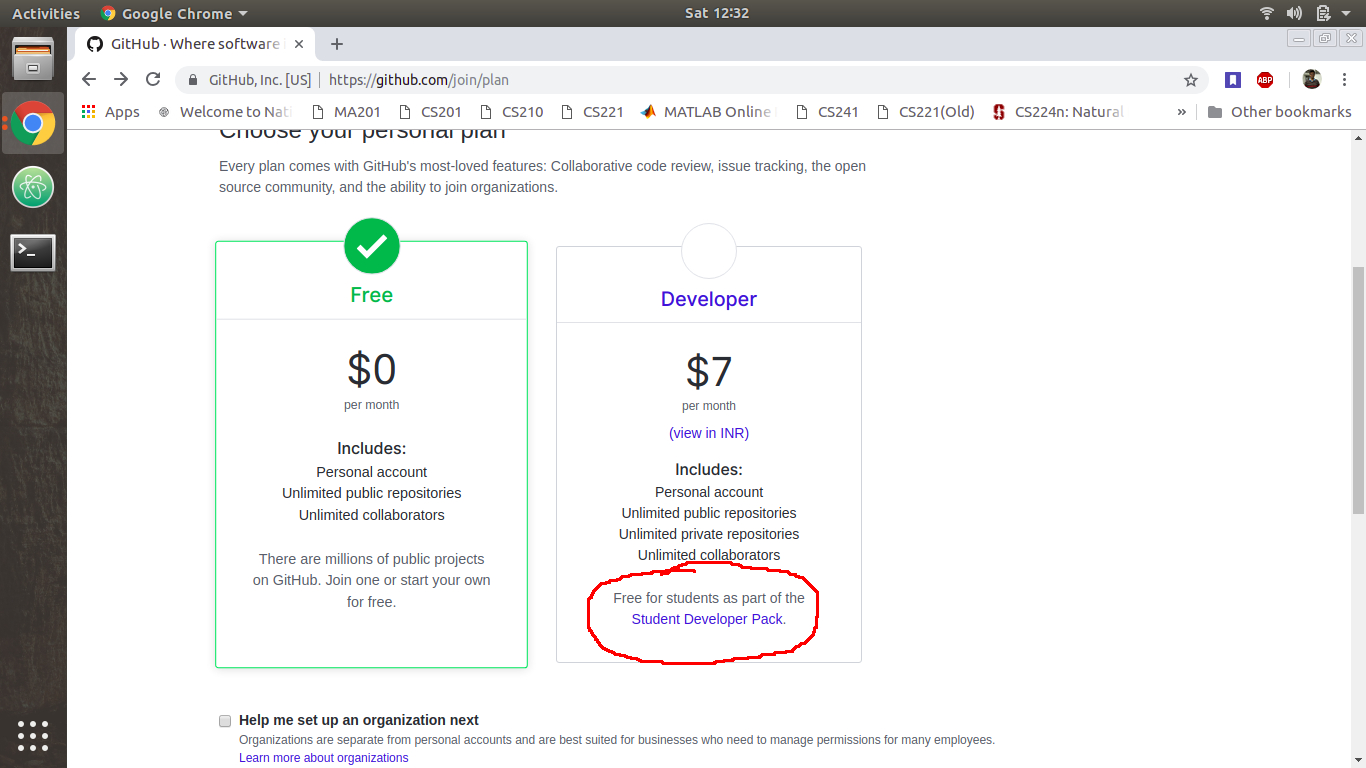
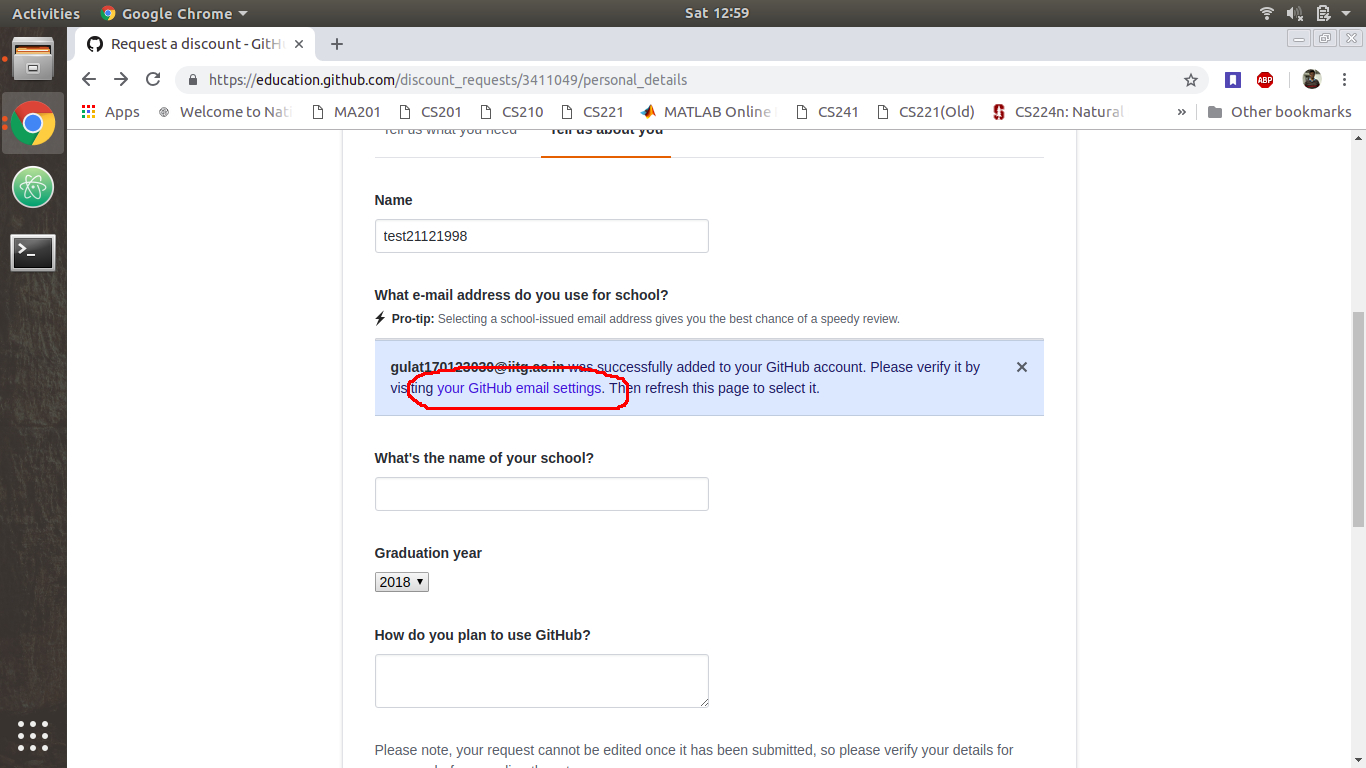
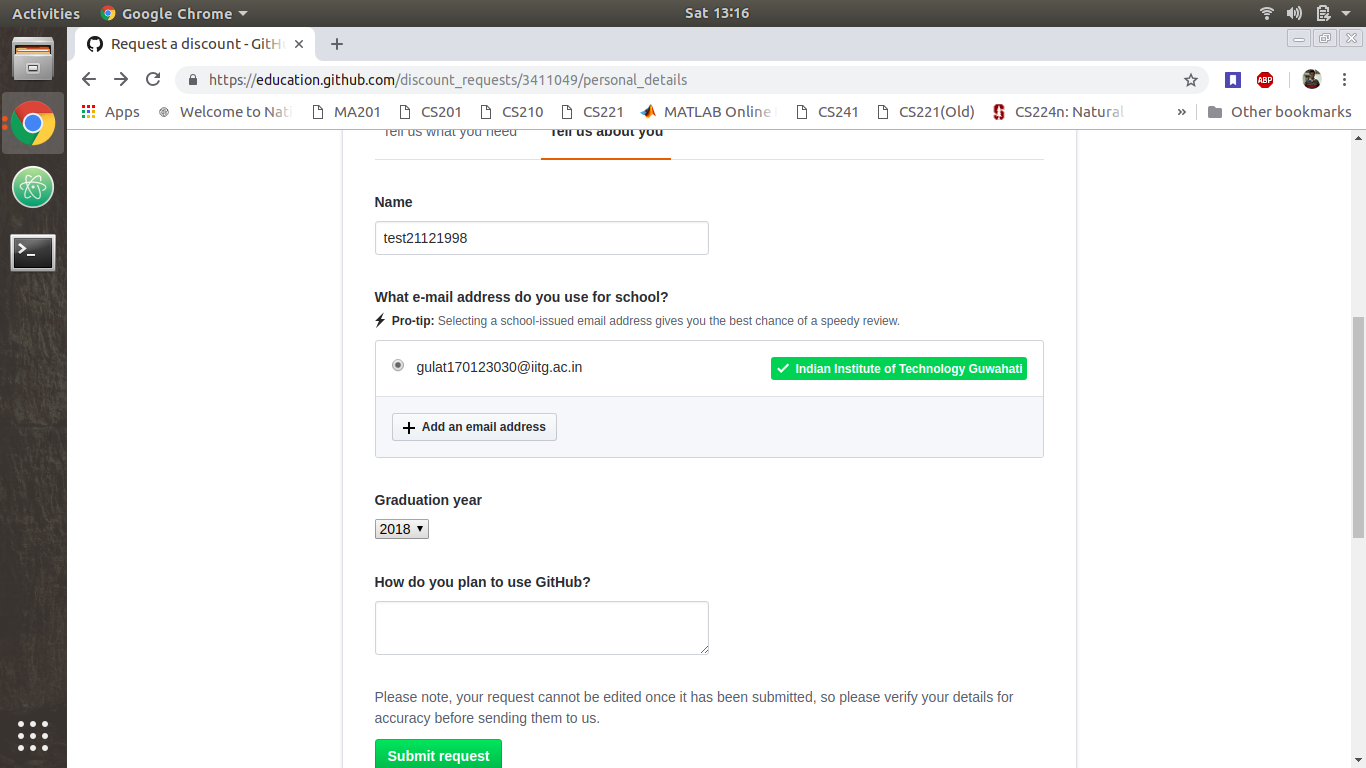
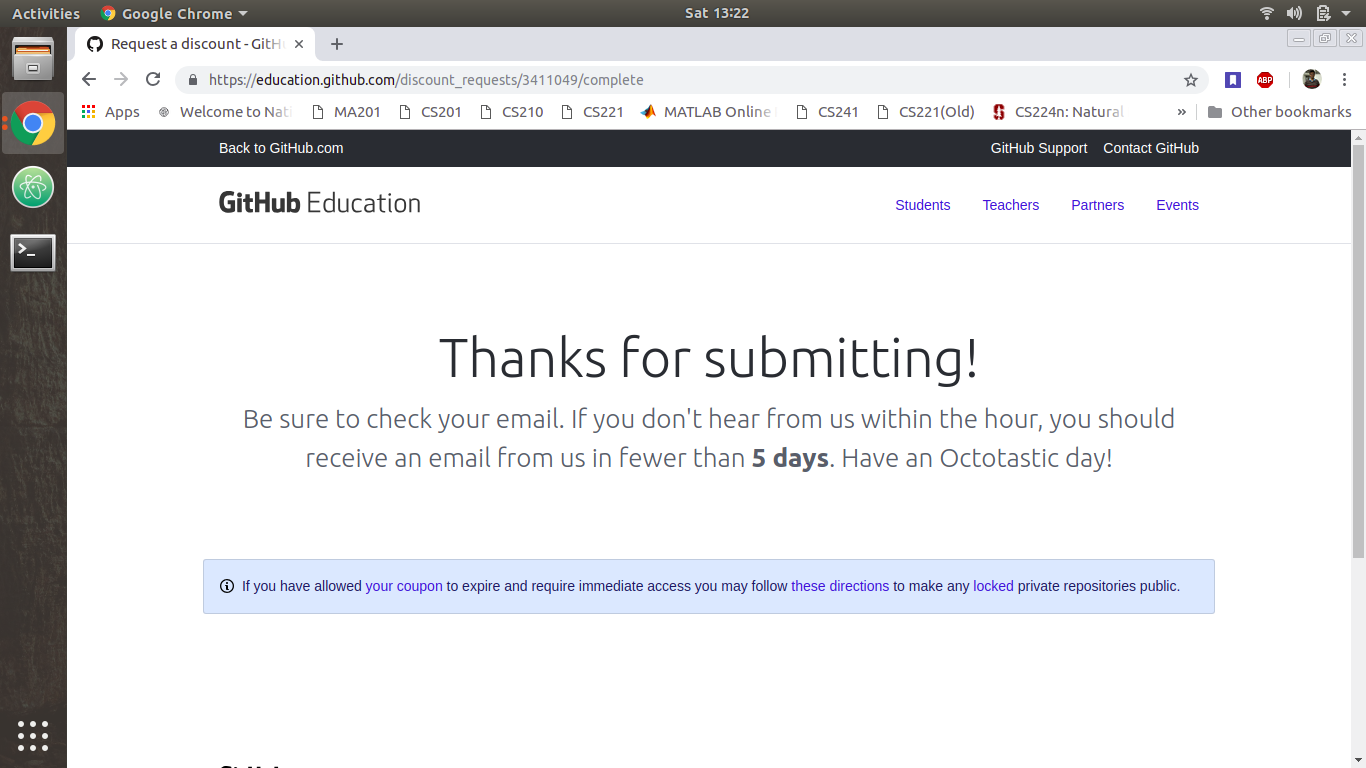
GitHub is a code hosting platform for version control and collaboration. It lets you and others work together on projects from anywhere.

This tutorial teaches you GitHub essentials like *repositories, cloning and commits* on **Linux Ubuntu**. You’ll create your own repository, learn how to clone it to your local system (your PC) and push the changes you made to the repository.

## **Step 1. Setting up account on Github**

To complete this tutorial, you will need a GitHub account. Make sure you have an internet connection too!

### **Set up an account on GitHub**

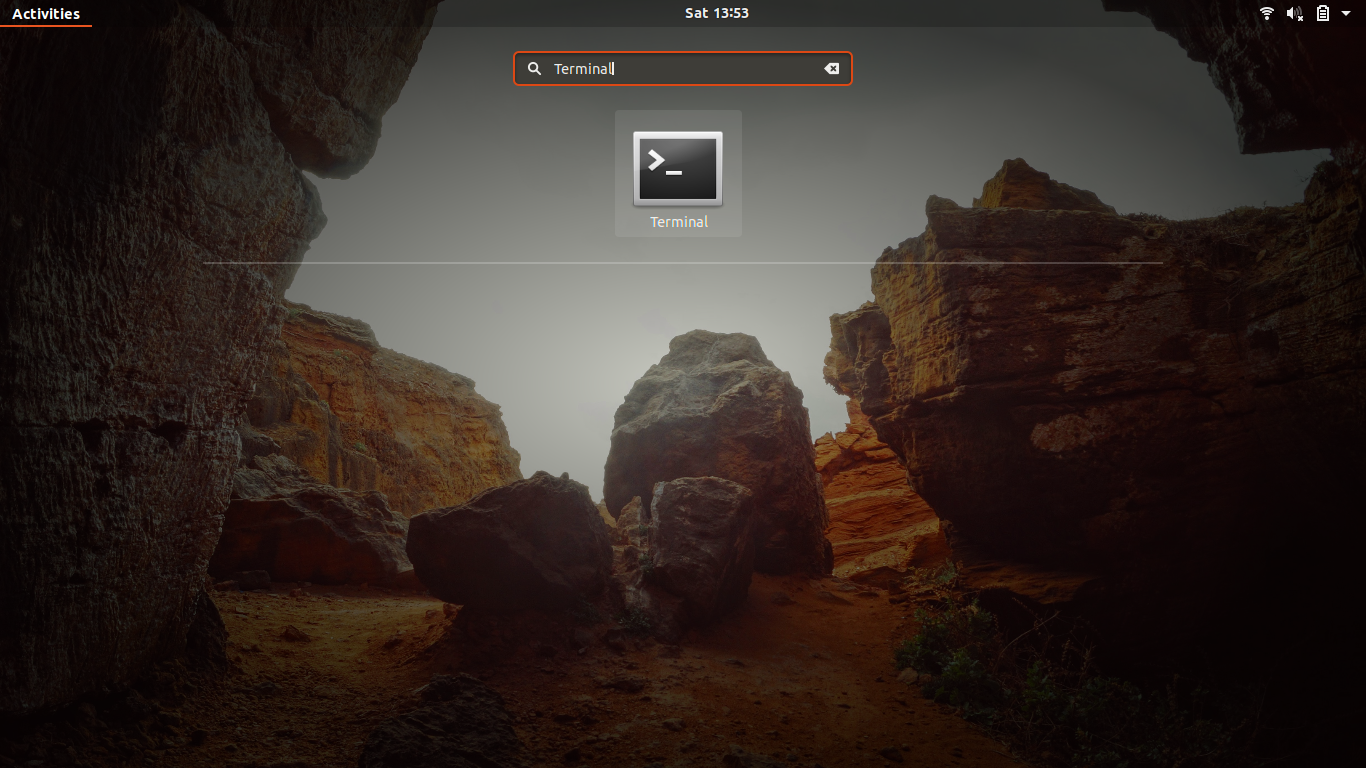
1. Open GitHub website by clicking on the given link: <https://github.com/>
2. Click on **Sign up** button on the upper right corner of the homepage.
3. Fill your valid credentials and solve the puzzle (will not be too hard :P) and click on **Create an account**
4. GitHub provides **Student Developer Pack** to students which allows you to create private repositories. You can also continue with the **Free** plan. The following steps are given to claim the former. Select **Student Developer Pack** under the **Developer** plan.
5. Select **Get your Pack** and then **Yes, I’m a student** when each of the new windows open.
6. Specify your **webmail ID** (e.g. xxxxxxxxx@iitg.ac.in) in the email address field. Verify your webmail ID by logging into your Outlook account in a new tab and clicking on the verification link. If the verification email is not received by your account, click on **your Github email settings.**
7. Click on **Resend** button next to your webmail ID and follow the above step for email verification.
8. Once your email has been verified, go back to the page mentioned in step 6 (and shown in the above screenshot) and refresh the page. Your webmail ID will appear as a button with your college name (IIT Guwahati) mentioned alongside. Select your webmail ID and fill the rest of the credentials, and click on **Submit request.**
9. You will be redirected to a page mentioning the process has been successful. You will also receive a mail about the upgradation of your pack. Go back to the homepage (<https://github.com/>) and log into your account.

## **Step 2. Installing and Setting up Git**

Version control systems are increasingly indispensable in modern software development as versioning allows you to keep track of your software at the source level. You can track changes, revert to previous stages, and branch to create alternate versions of files and directories.

One of the most popular version control systems currently available is Git. Many projects’ files are maintained in a Git repository, and sites like GitHub, GitLab, and Bitbucket help to facilitate software development project sharing and collaboration.

### **Installing Git**

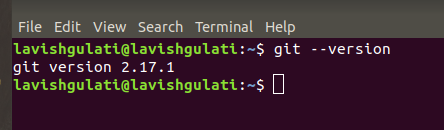
1. Open the terminal by pressing **Ctrl+Alt+T** keys or by going to the **Applications** menu and searching **Terminal** in the Search bar.
2. Type the following commands (without the $ sign) in the Terminal window and specify your user login password when asked for:

*$ sudo apt update*

*$ sudo apt install git*

1. You can confirm that you have installed Git correctly by running the following command:

*$ git --version*

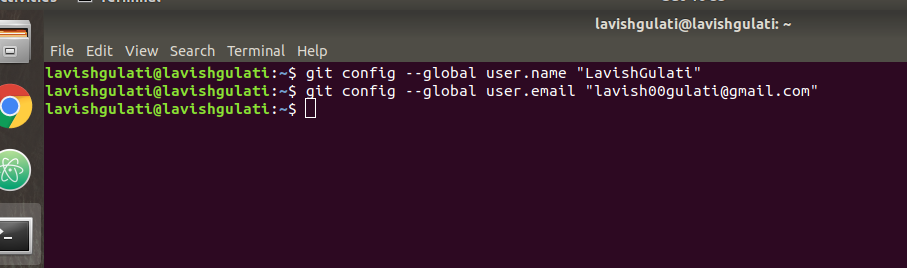
**

### **Setting up Git**

1. You need to specify your username and email ID of your GitHub account to use git. Type the following commands in your Linux terminal:

$ git config --global user.name "your\_GitHub\_user\_name"

$ git config --global user.email "your\_GitHub\_user\_email"



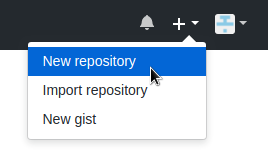
## **Step 3. Creating a Repository**

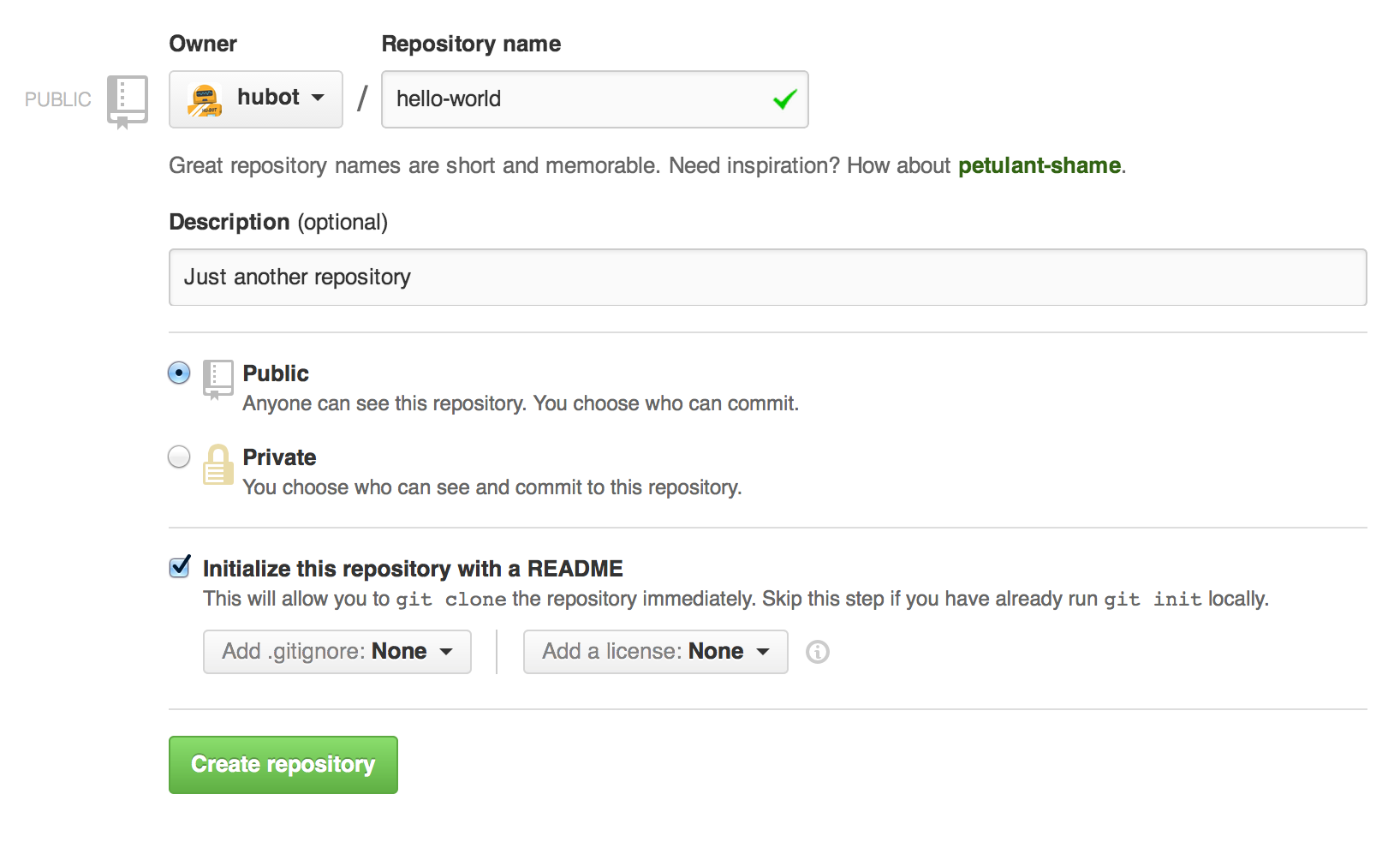
A **repository** is usually used to organize a single project. Repositories can contain folders and files, images, videos, spreadsheets, and data sets – anything your project needs. We recommend including a *README*, or a file with information about your project. GitHub makes it easy to add one at the same time you create your new repository. *It also offers other common options such as a license file.*

Your hello-world repository can be a place where you store ideas, resources, or even share and discuss things with others.

### **Create a new repository**

1. In the upper right corner, next to your avatar or identicon, click on the **+** icon which opens a drop-down menu and then select **New repository**.



1. Name your repository **Hello-World**.
2. Write a short description(optional). Choose whether you want to make your repository **Public** or **Private** (if you are a beginner in using GitHub, choose **Public**).
3. Select **Initialize this repository with a README** (optional).
4. Click **Create Repository.**

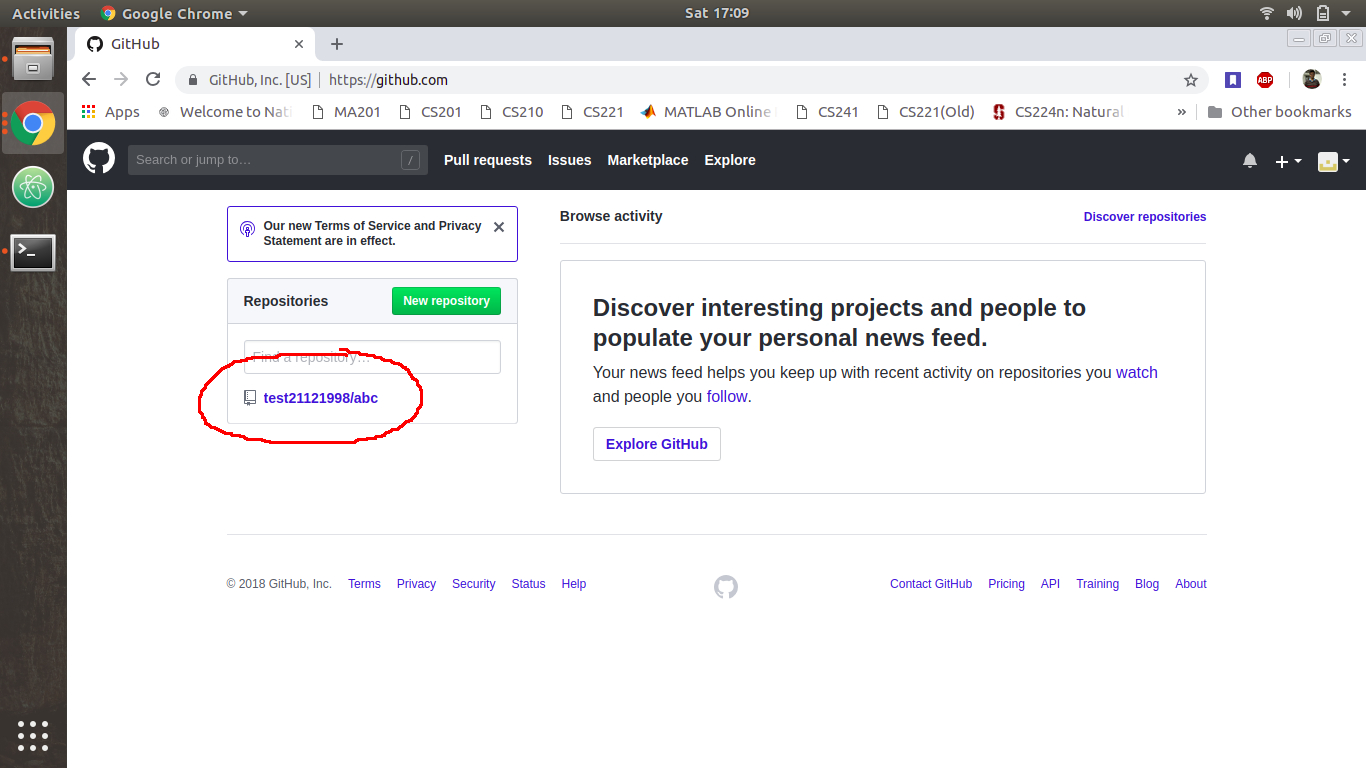
## **Step 4. Cloning your Repository**

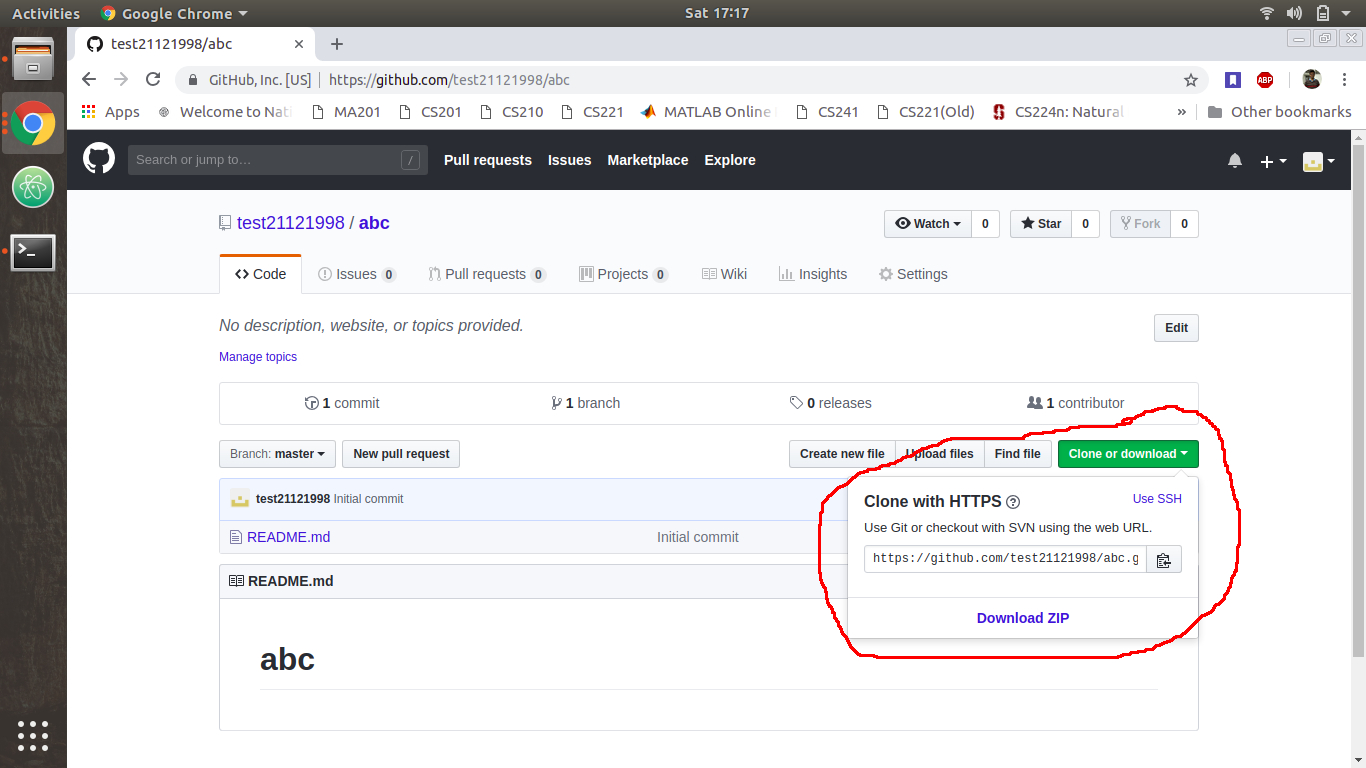
Cloning a repository means that you’ll copy all of the files and directories on the GitHub server onto your machine so you can work with them.

Now you need to clone the project that you created on GitHub to your workspace. To do this, go to the folder in which you want to clone your repository.

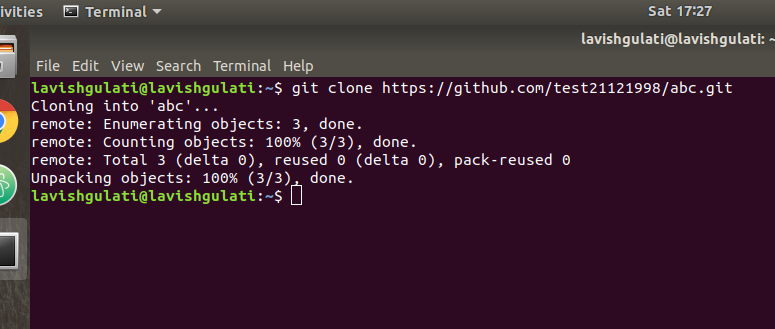
### **Cloning your Repository**

1. Go to your project page (repository) on GitHub by searching its name in the **Repositories** panel on the left of the screen and clicking it



1. There will be a *green* button **Clone or download** containing the link of the repositoryon the right side of the screen. Clicking on that will open a drop-down dialogue box. Copy the link given in the box.
2. Now run the *git clone* command on the terminal window and pass the link that you copied with the command as shown.

$ git clone https://github.com/test21121998/abc.git

1. Your repository will be cloned as shown below. You can access the repository folder which was created in your workspace when you cloned it.

## **Step 5. Committing the changes**

Once you finish a change to your project, you should **commit** the change to your remote repository (the one on GitHub’s servers).

To **commit** something is to tell Git that you are putting your changes in the queue to be pushed (sent) to your remote repository.

Imagine that you just created an HTML page and added some titles and text to it. You have the first version of this document now, so you should commit it.

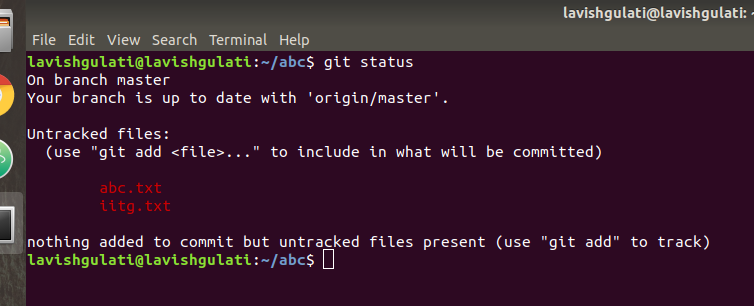
To do this, run some commands so that Git understands that we want to send our changes do the remote repo.

**Make sure you are inside the repository whose changes you want to commit to the remote server.**

### **Committing the changes**

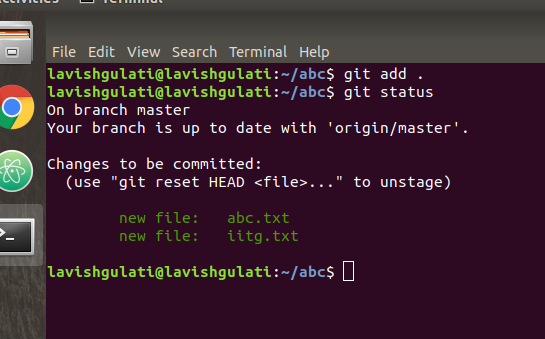
1. With the *git status* command, you can see which changed files you will commit to the server. The files shown in *red* are the files in which changes are made (compared to the last commit, if any) but not yet staged for the commit.

$ git status



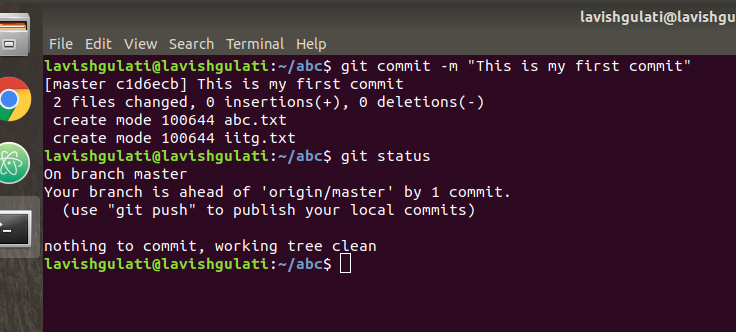
1. Run *git add file\_name* to tell Git to stage the file. Alternatively, you can run *git add --all* or *git add .* (dot),both of them having minor differences, to send all the files that you made some changes to. Run *git status* to observe the changes to your working tree. The files shown in *green* are the files which are staged for the new commit.

$ git add .



1. In the above example, two files: iitg.txt and abc.txt were created and the *git status* command was run to see what was changed. Then the files were added with *git add* and *git status* was run again to see which files were added to the Git workspace.
2. With that you can now **commit** the changes. Just run the *git commit* command, just like *git commit -m “commit\_message”*. Remember to include a descriptive message of what was added to he commit. Run *git status* to observe the changes.

$ git commit -m “commit\_message”

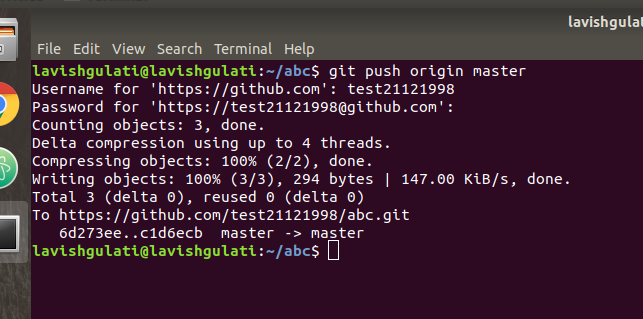


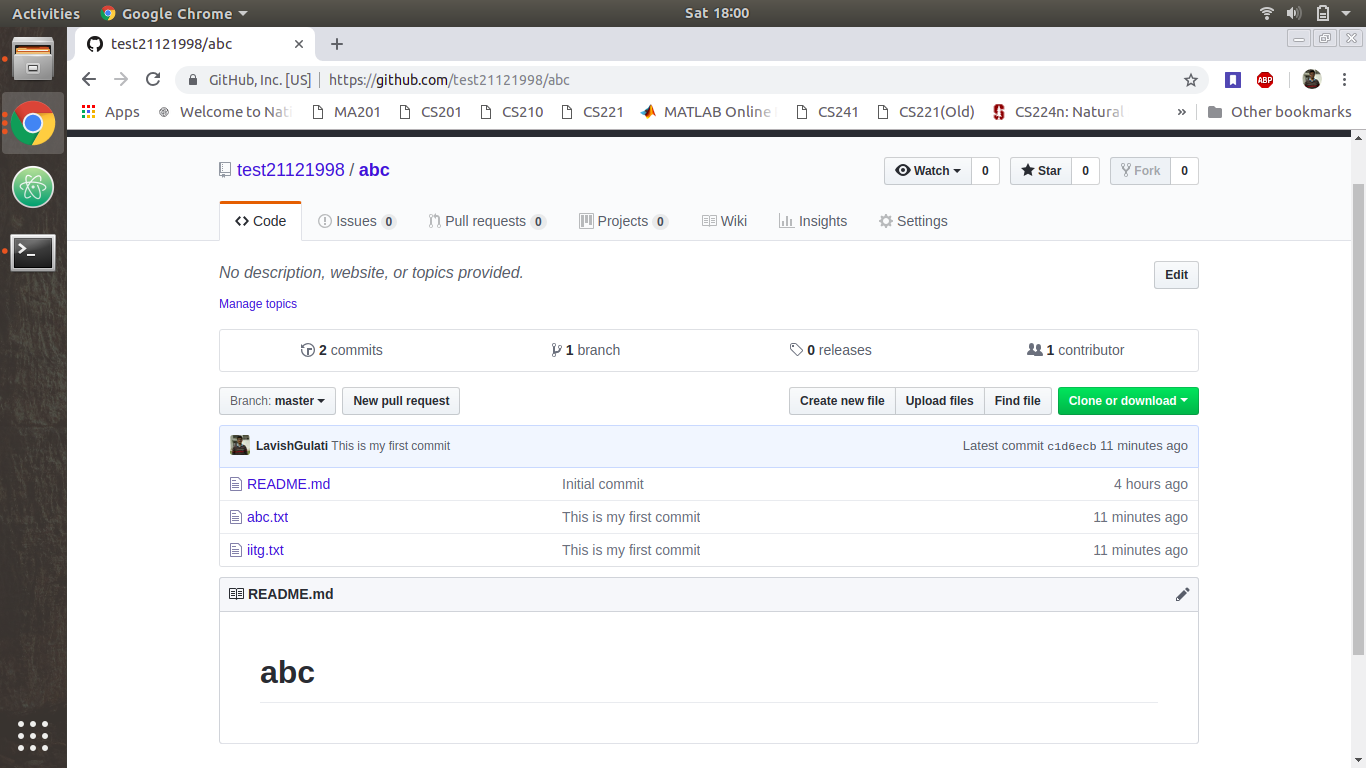
## **Step 6. Pushing the changes**

After you’ve made and merged all the changes, you can now send them to your remote repository on GitHub.

### **Pushing the changes**

1. You will use *git push origin master* to do this. You can also just use *git push*. It’ll have the same result. But when you push changes for first time on your workspace, you need to do *git push origin master* so that Git will know that your workspace is the origin of the push. Specify your username and password of your GitHub account when asked.

$ git push origin master

1. Now your commit will appear on your GitHub repository’s page.

## **Conclusion**

In this tutorial, you learned how to create a project on GitHub so that you can track your progress every time you study something new. This will help you get to know the command line (Terminal), Git commands, and GitHub. Besides that, it’ll help you create a nice portfolio that you can show in job interviews.

Practicing like this will also help you better understand how to use **Git with remote repositories** (the repositories hosted on some platform like **GitHub**). You’ll also level up your knowledge and skills on the Terminal.

Author: Lavish Gulati, Web Development Coordinator, Coding Club, IIT Guwahati

Sources: <https://medium.freecodecamp.org/how-you-can-learn-git-and-github-while-youre-learning-to-code-7a592ea287ba>

<https://guides.github.com/activities/hello-world/>

In case of any suggestions or improvements, mail to [lavish00gulati@gmail.com](mailto:lavish00gulati@gmail.com), or ping me :P