**Setting up CI for a repository on WINDOWS platform**

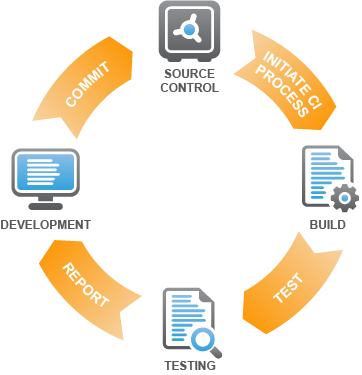


Fig: Continuous Integration (CI)

For setting up a continuous integration (CI) we need to have three major components,

Which are:

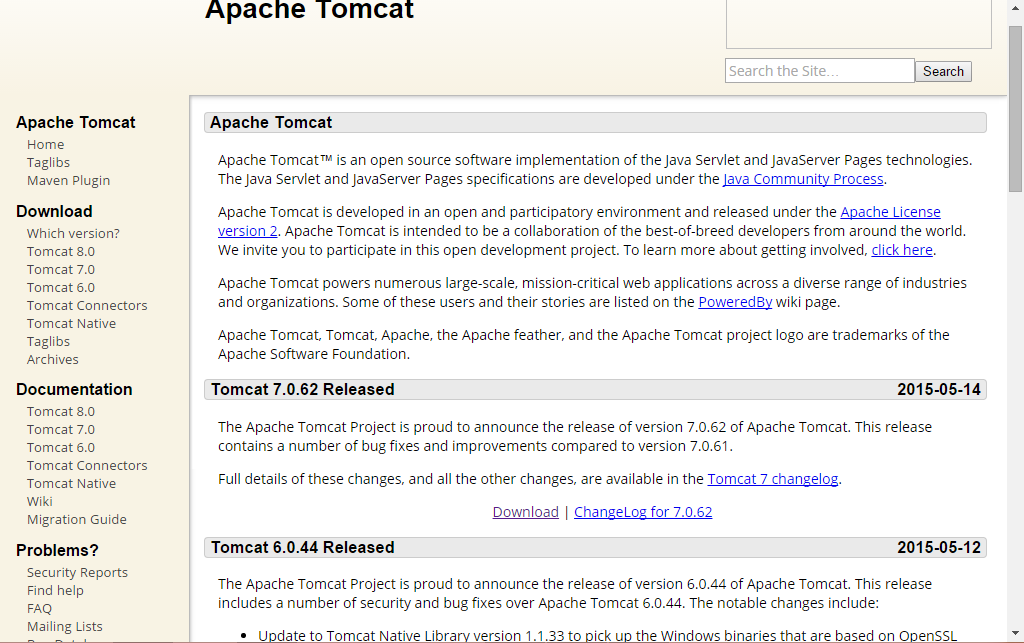
1. a server (Tomcat),
2. software that performs continuous integration (Jenkins) and
3. a repository (github).

To setup continuous integration we need to perform certain steps.

\*Internet required.

1. **Setting up server.**

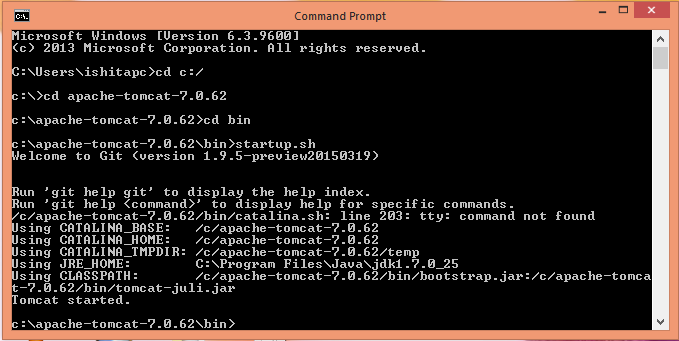
**Step 1.1**: Download Tomcat folder from <http://tomcat.apache.org/> as shown below;



Save the downloaded apache-tomcat-7.0.62 folder to C: / drive.

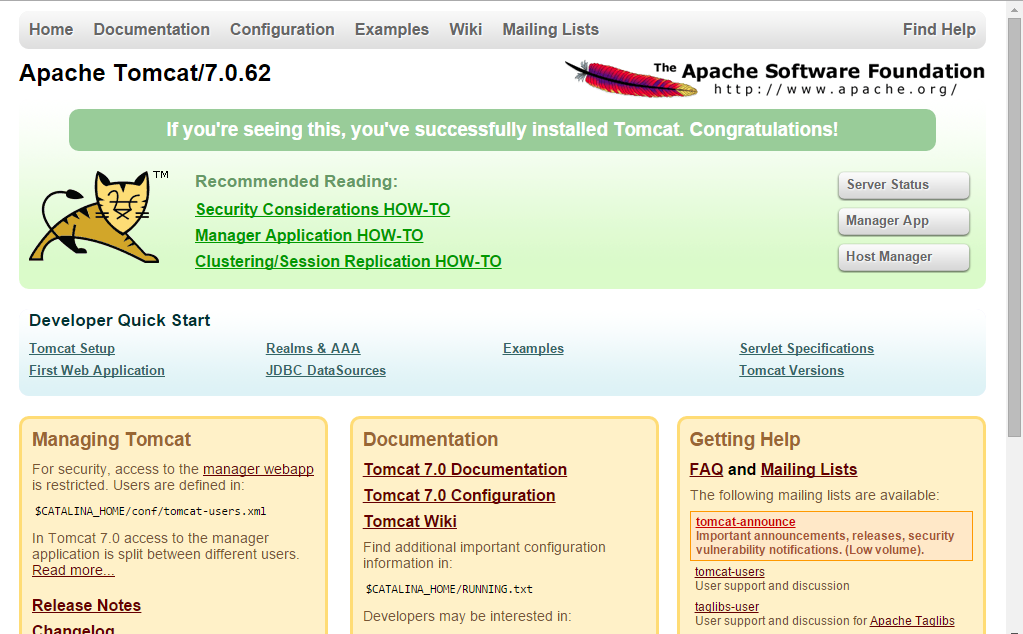
**Step 1.2**: Open command prompt. Go to C:/apache-tomcat-7.0.62/bin folder and run startup file by giving following commands on command prompt:

* *cd c:/*
* *cd apache-tomcat-7.0.62*
* *cd bin*
* *startup.sh*



Now to check if tomcat is working properly, go to a web browser and open <https://localhost:8080>

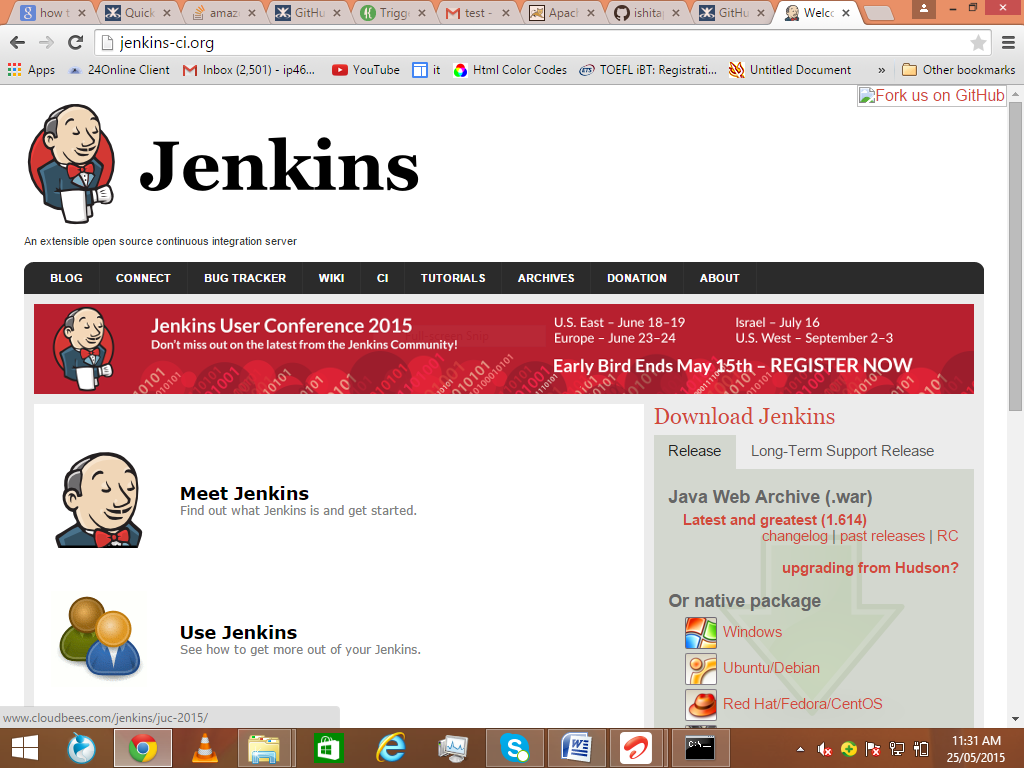
This page should be displayed.



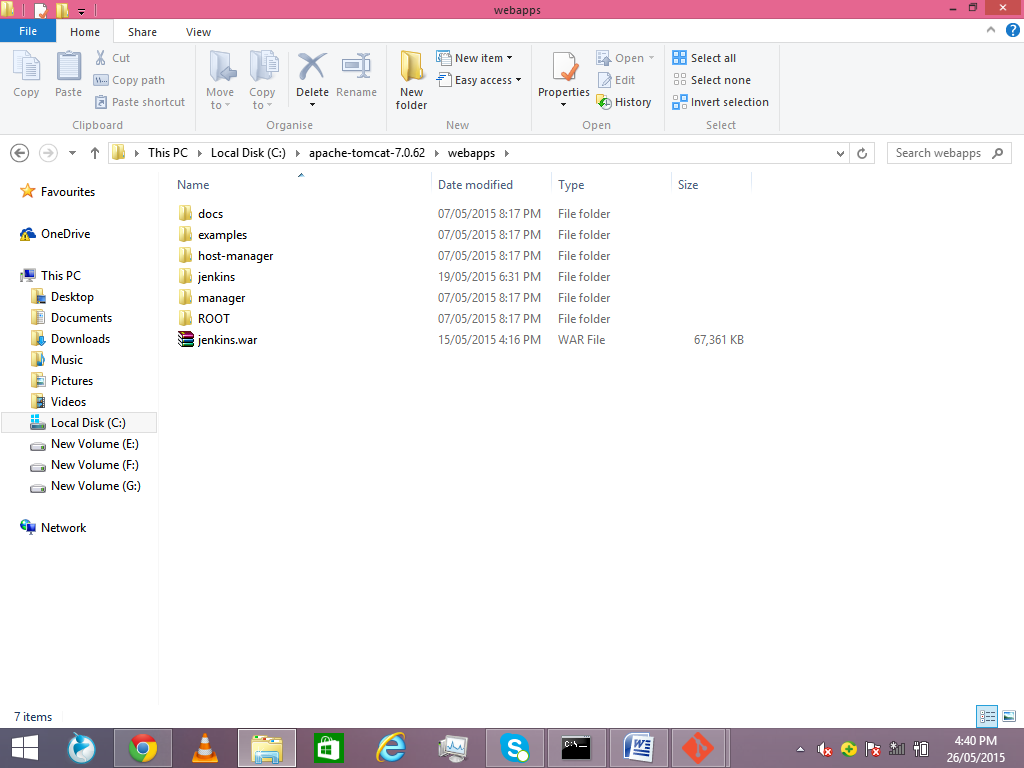
We have successfully started the server from the local host.

1. **Configuring Jenkins**

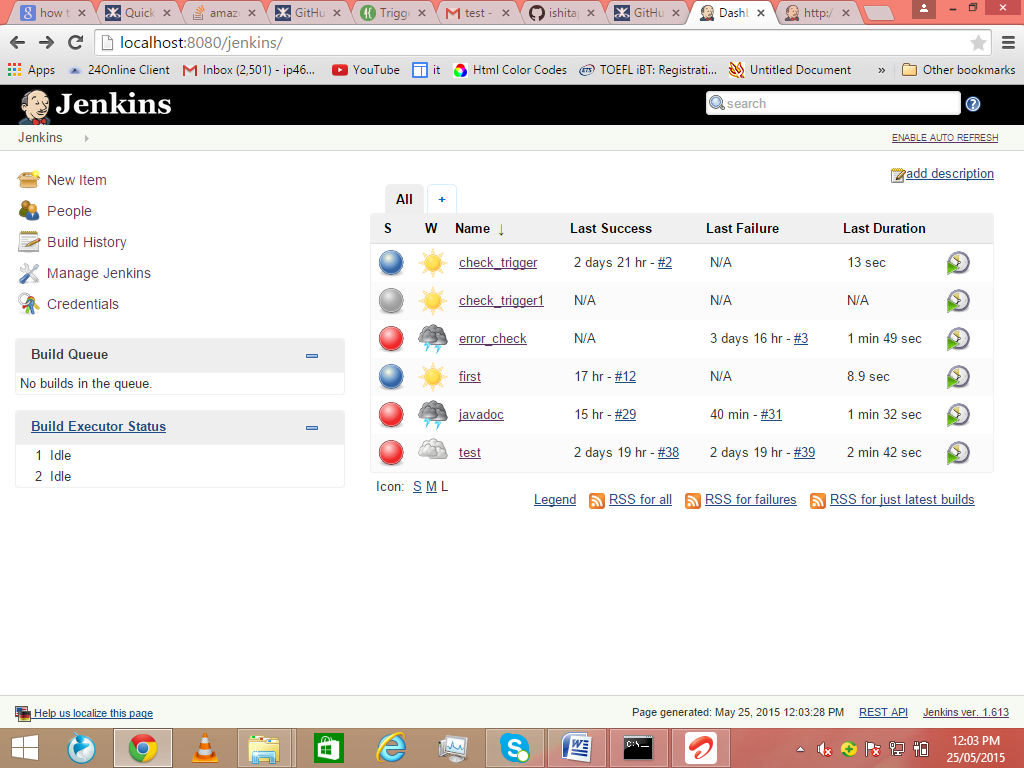
**Step 2.1**: Now we need to download jenkins.war file from <http://jenkins-ci.org/>



Sava the downloaded jenkins.war file in tomcat’s webapp folder.



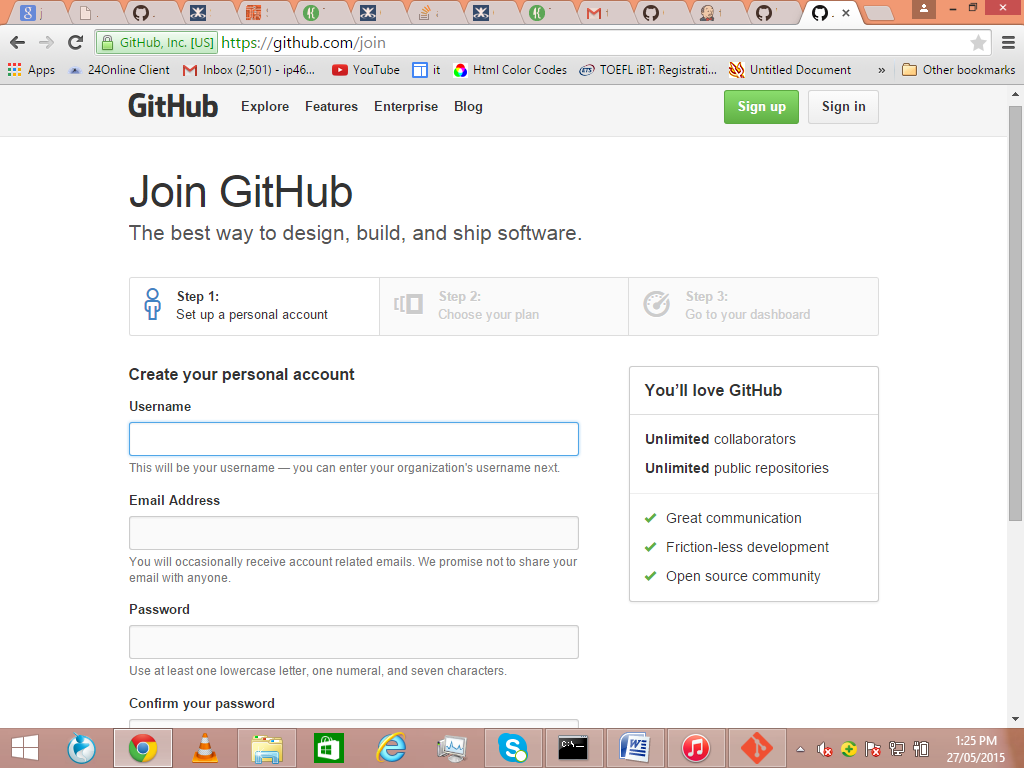
**Step 2.2**: Now open a web browser. Open <https://localhost:8080/jenkins>



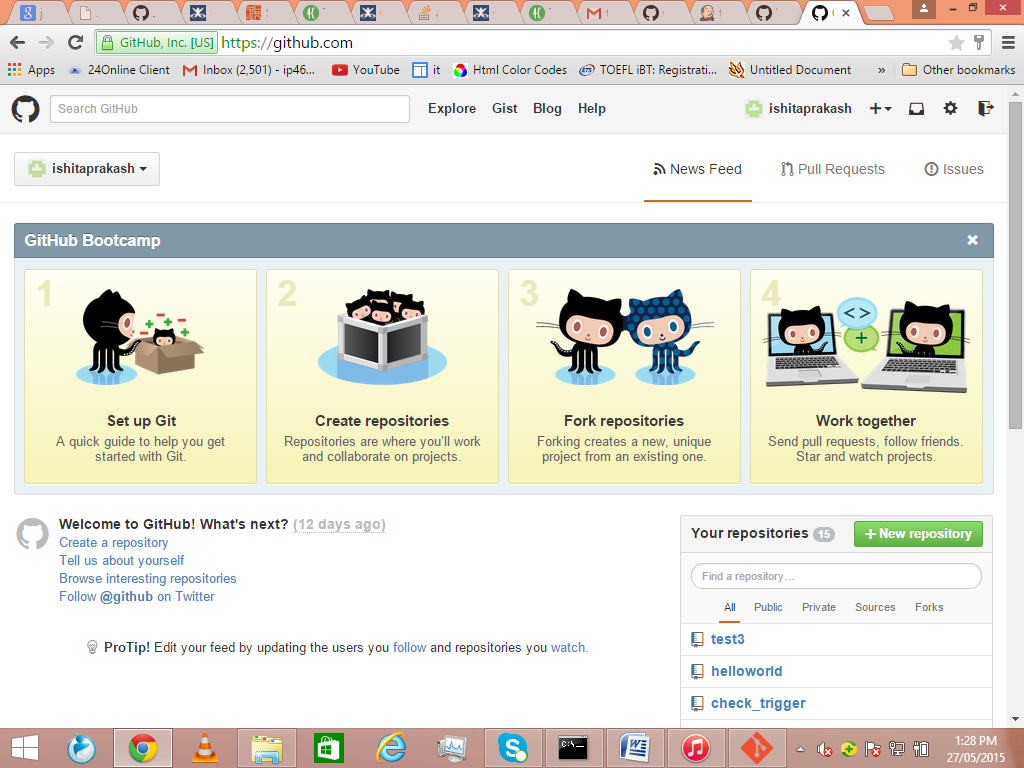
*Note->Make sure you have already started the server through command line as stated earlier.*

1. **Configuring Git**

**Step 3.1**: Now we need to **create a profile on git**. Go to <https://github.com>



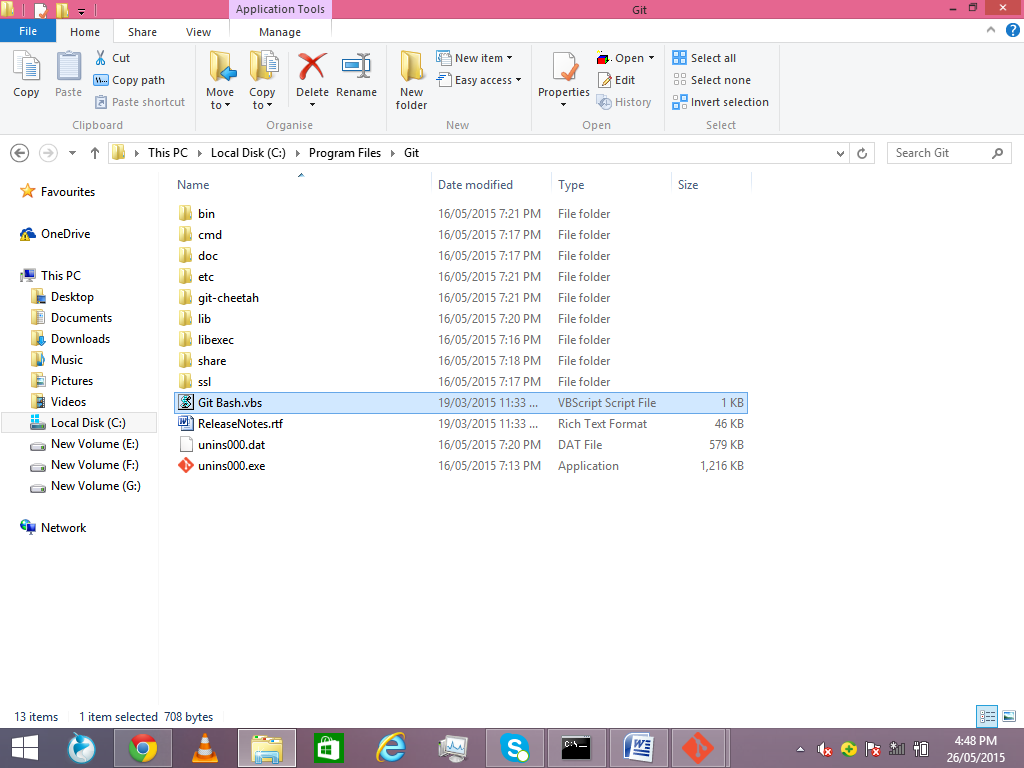
When you successfully create a profile, you will see:



We need to download Git on our system to enable Git bash for running Git commands.

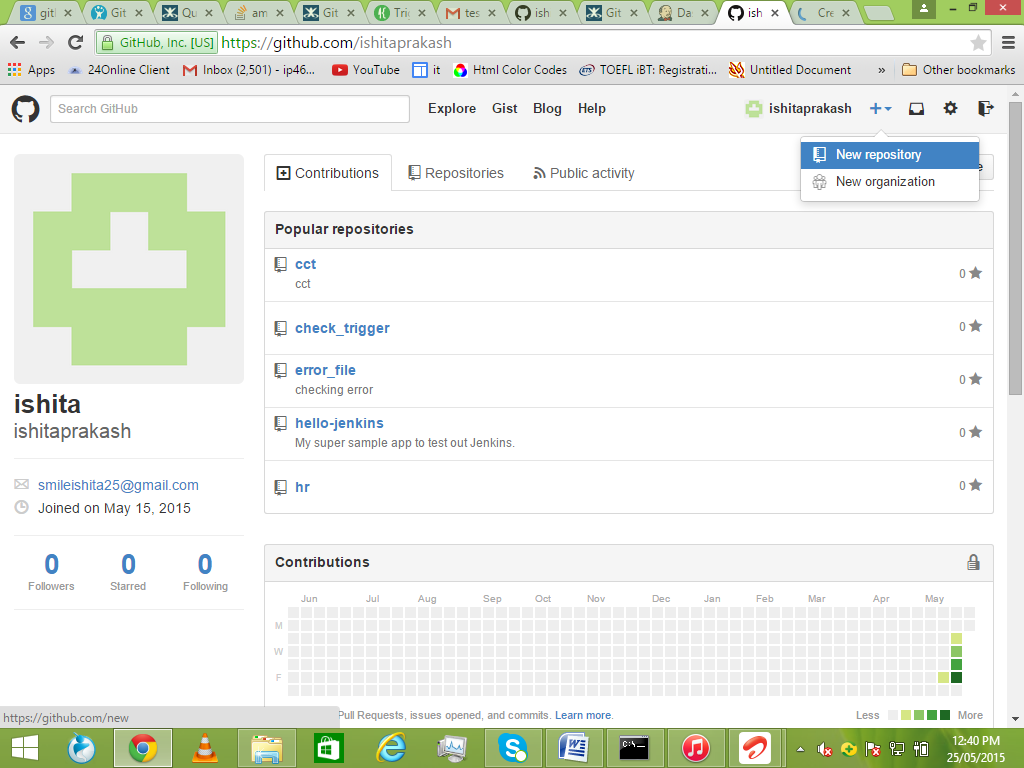
Download GitHub from <http://github-for-windows.en.softonic.com/>

Save the Git folder on your system.

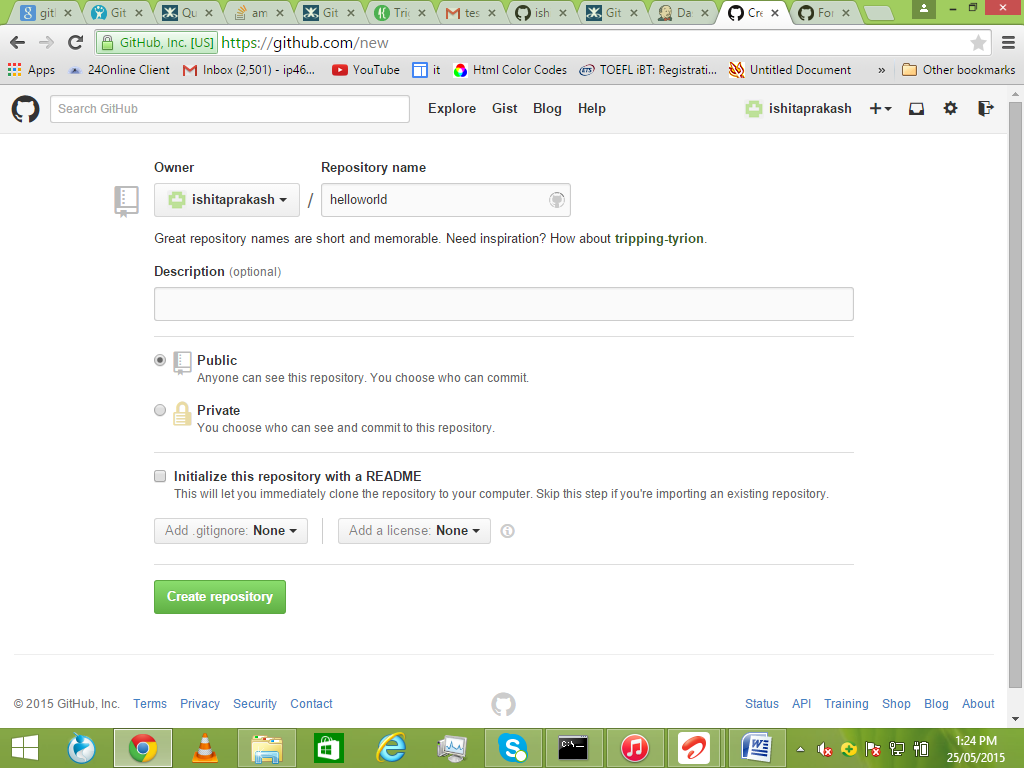


Git Bash is the application you open when we need to run Git commands.

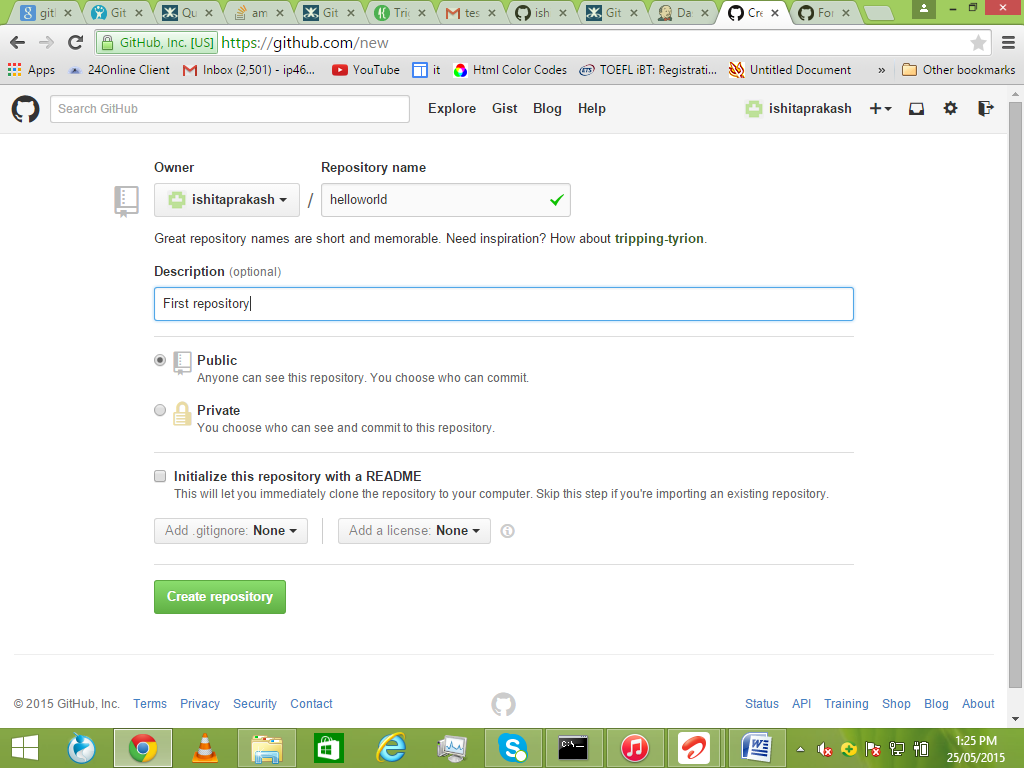
Now you are ready to **create repositories on Git Hub**.



**Step 3.2**: Create a name for your repository.



Optionally, but recommended, add a description of your repository.

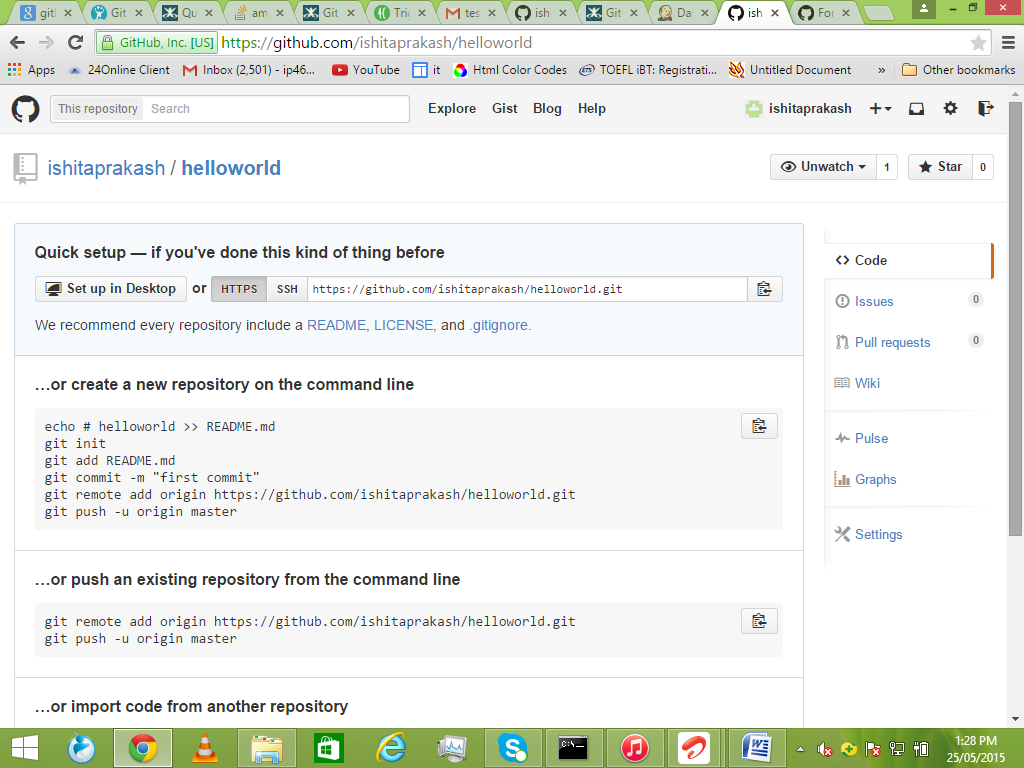


Choose between creating a public or private repository.

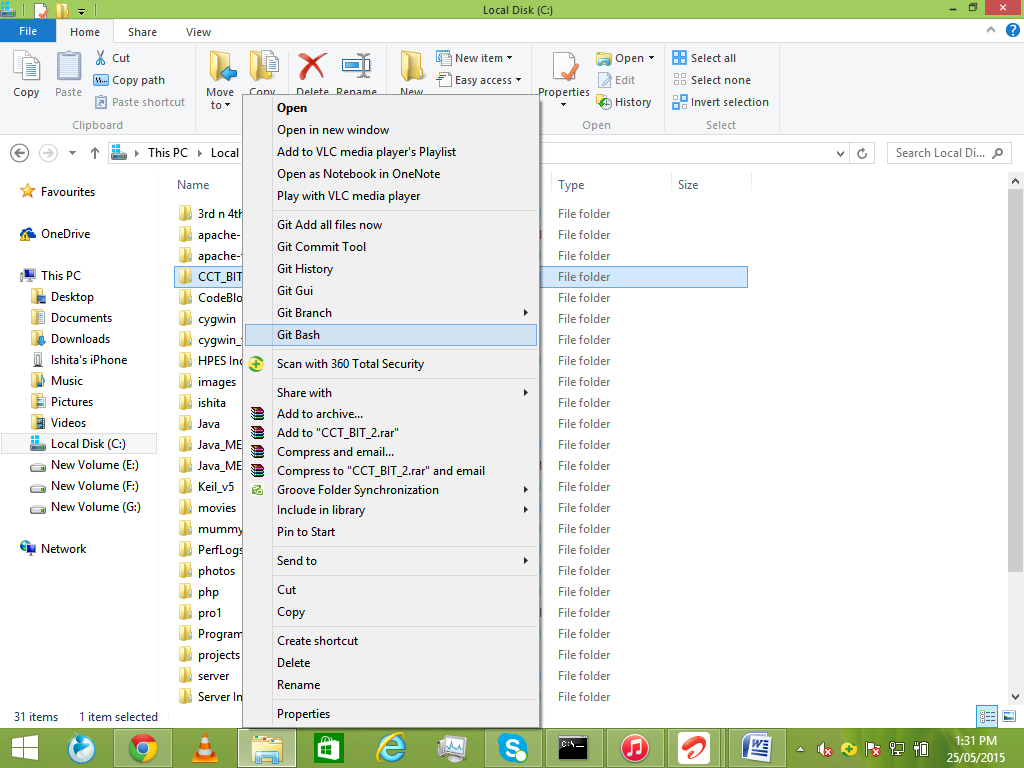
*Note->*

* *Public repositories are a great choice for getting started! They're visible to any user on GitHub, so you can benefit from a collaborative community.*
* *Private repositories require a little more setup. They're only available to you, the repository owner, as well as any collaborators you choose to share with.*

Click Create repository.

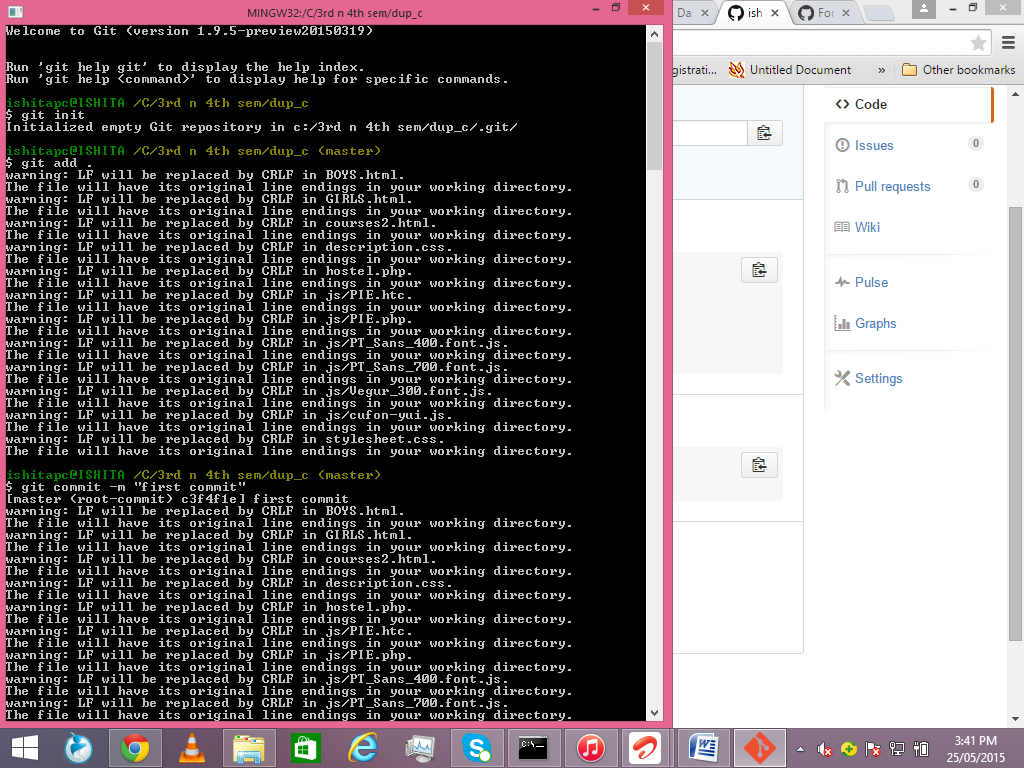


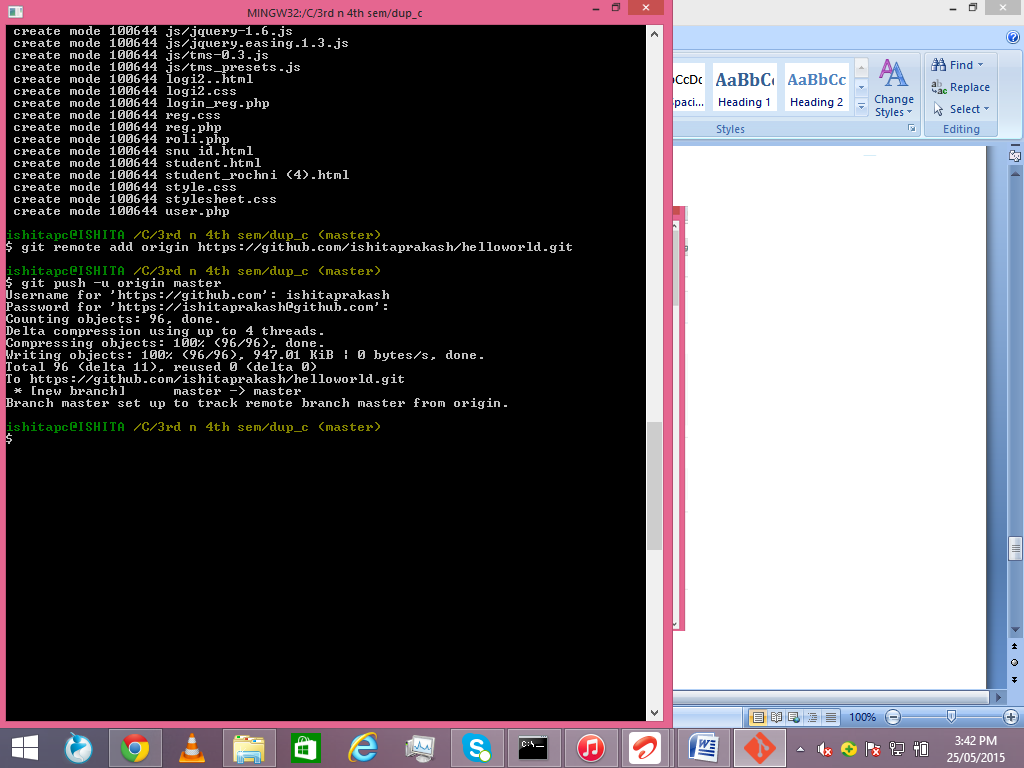
**Step 3.3**: Now we need to **push our project on Git**. Right click on the project and click on Git bash.



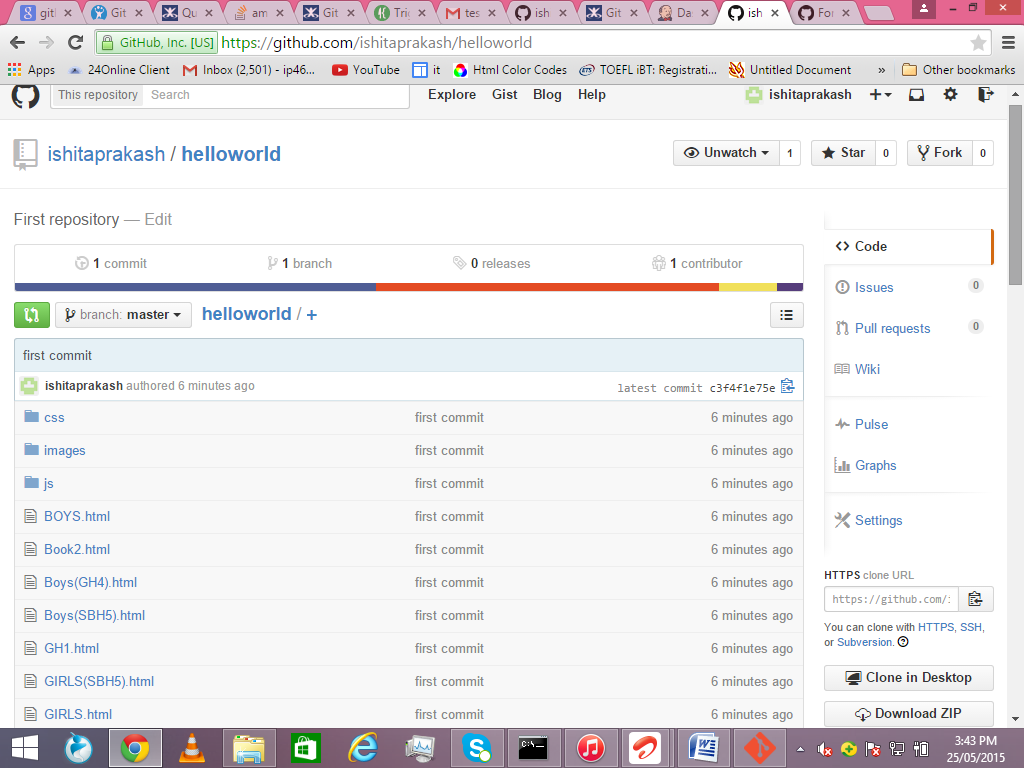
*Note-> The commands are:*

* *git init*
* *git add .*
* *git commit –m “message”*
* *git remote add origin* [*https://github.com/<username>/<repository*](https://github.com/%3cusername%3e/%3crepository) *name>*
* *git push –u origin master*



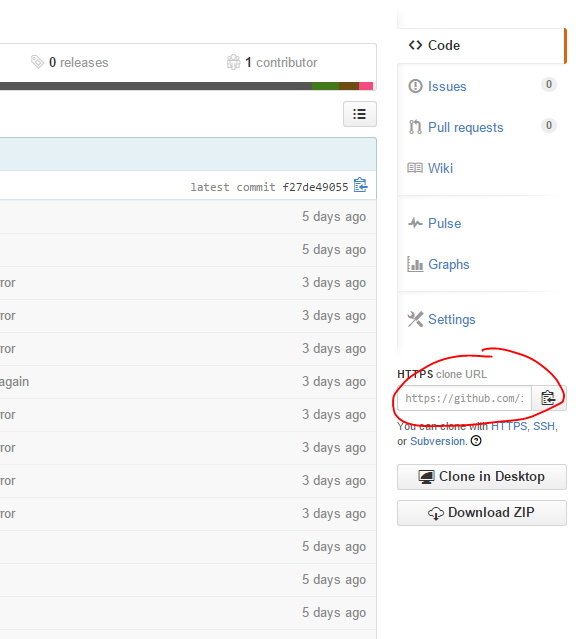


Once you reload git page you will see this. All the files added to the repository.

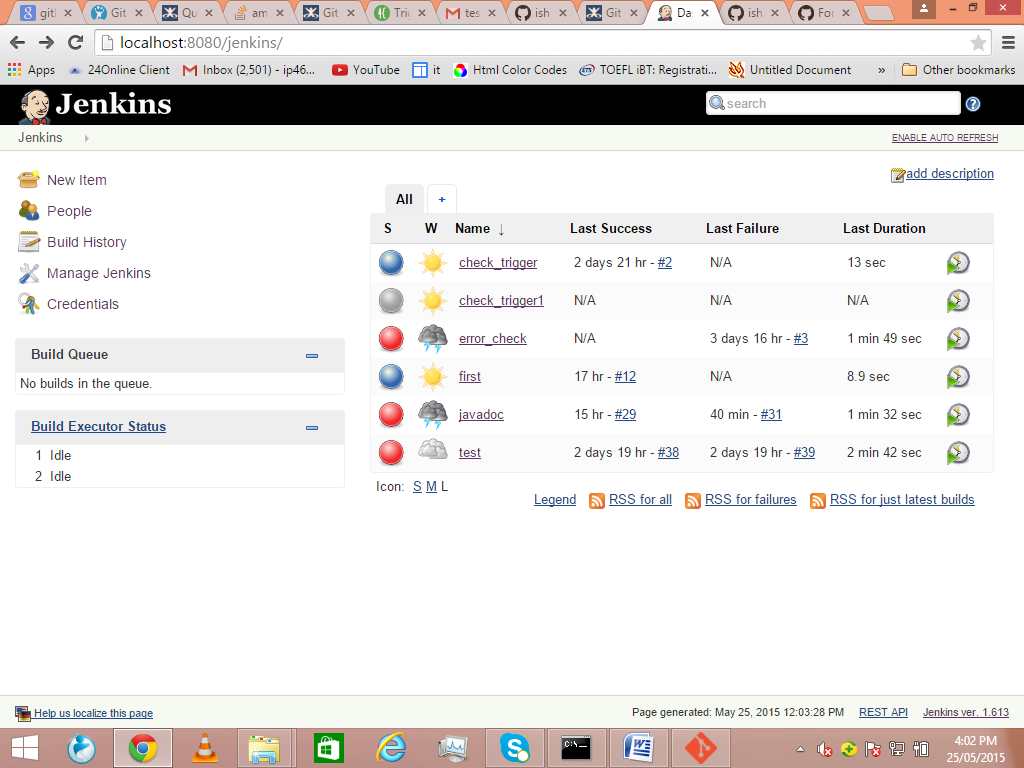


1. **Configuring Jenkins for Git**

**Step 4.1**: Now we need to tell Jenkins the **URL of your repository** so that it can access it. Copy this URL.



**Step 4.2**: Now we need to **create new job in Jenkins**.

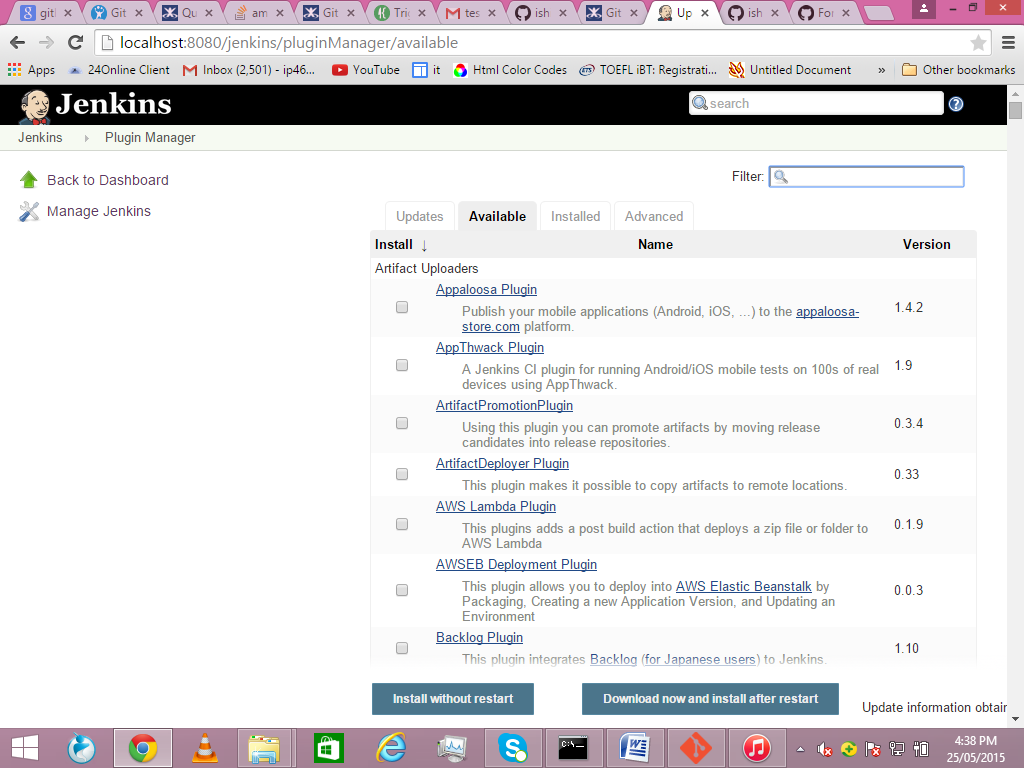


**Step 4.3:** First we need to add git plugin.

Go to Manage Jenkins.

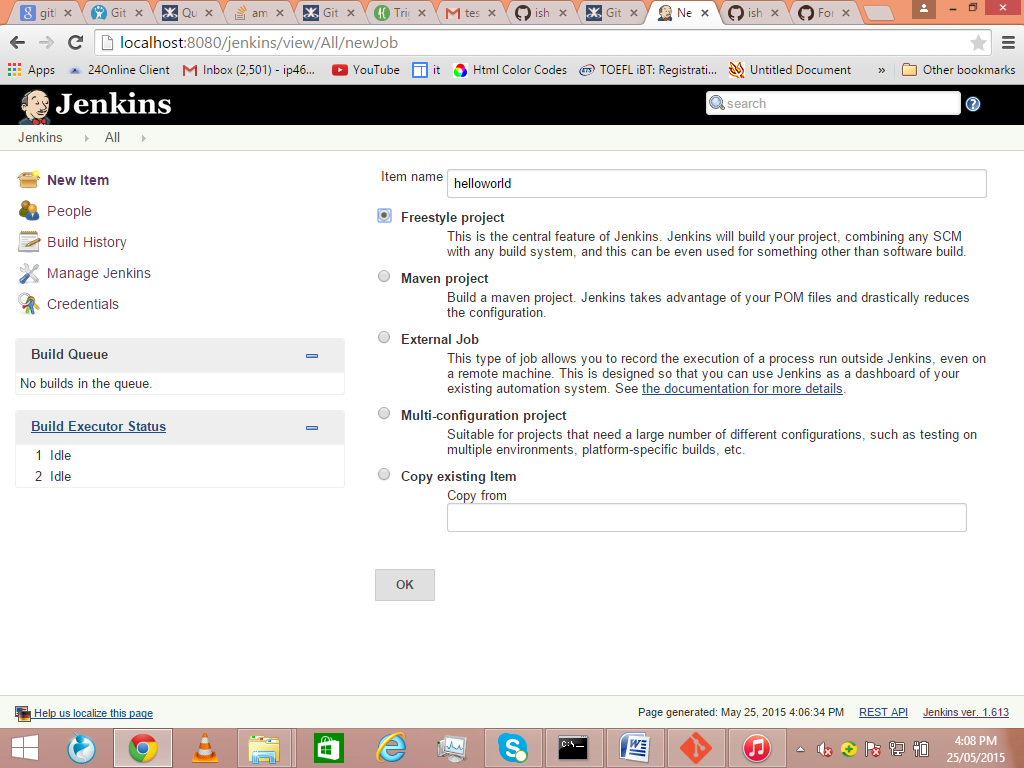


Click manage plugins.



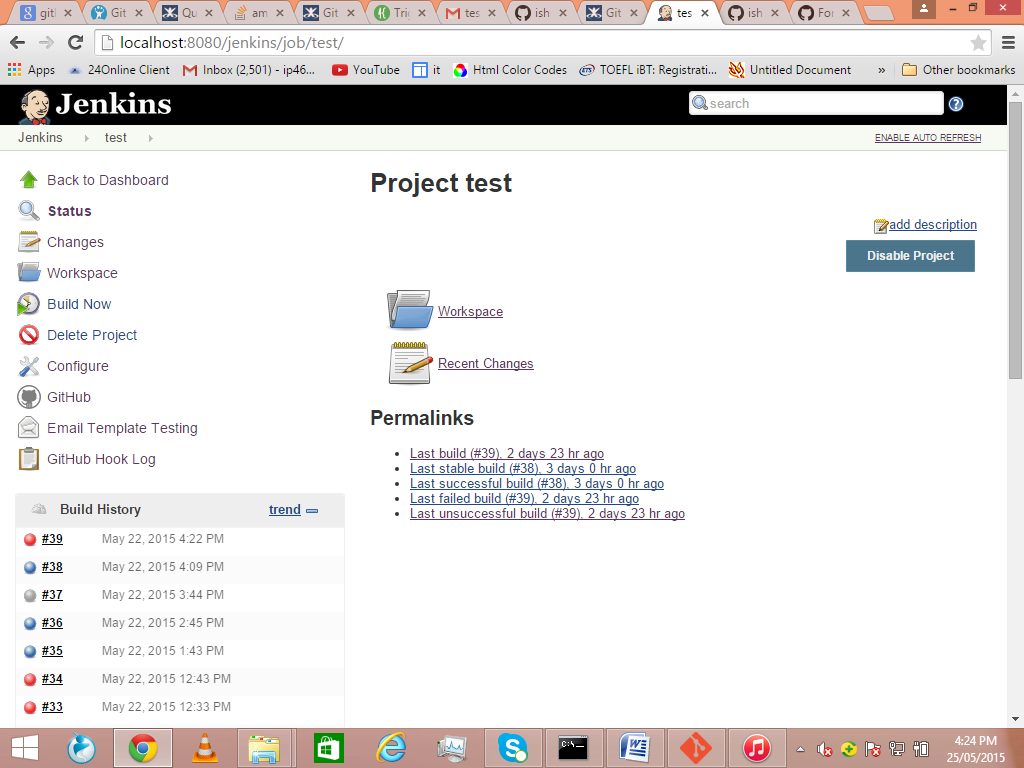
Search GIT Plugin on filter and install.

**Step 4.4:** Click on new item on the home page of Jenkins.

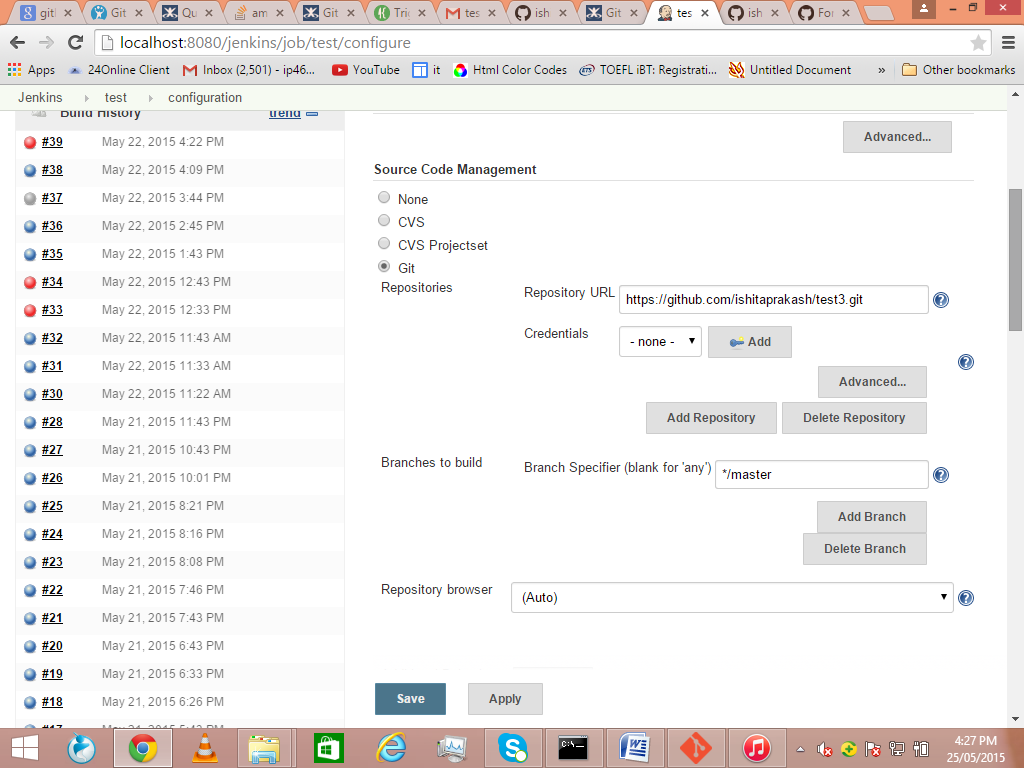


Name the project.

Select the type of project you want to create and click OK.

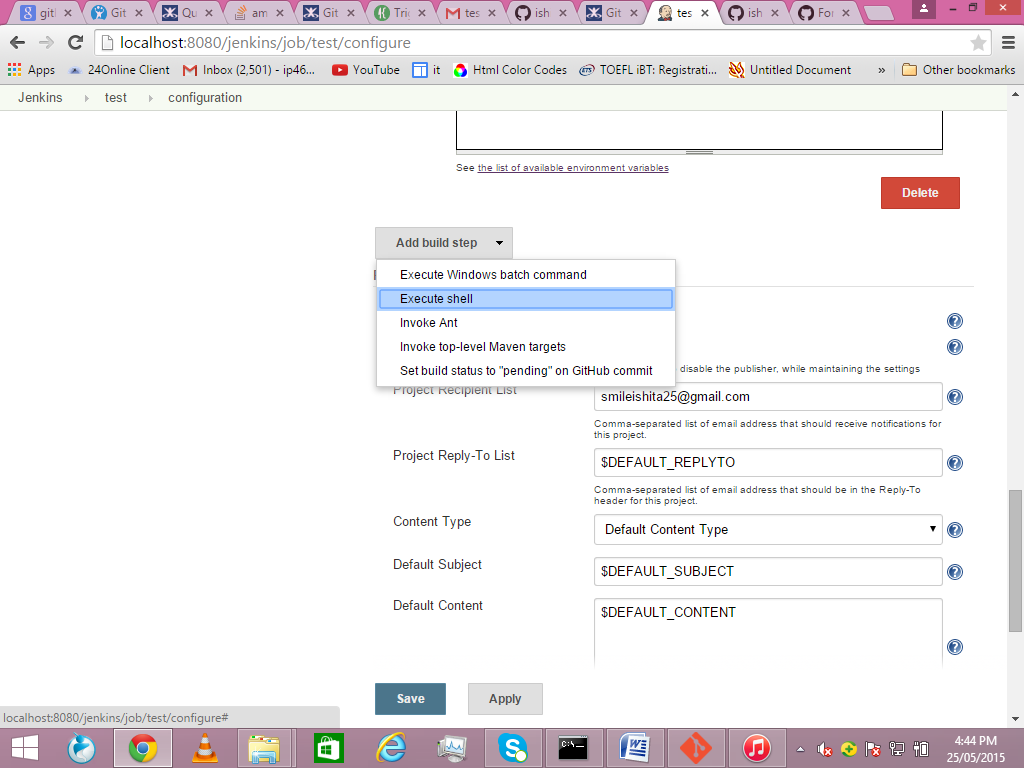


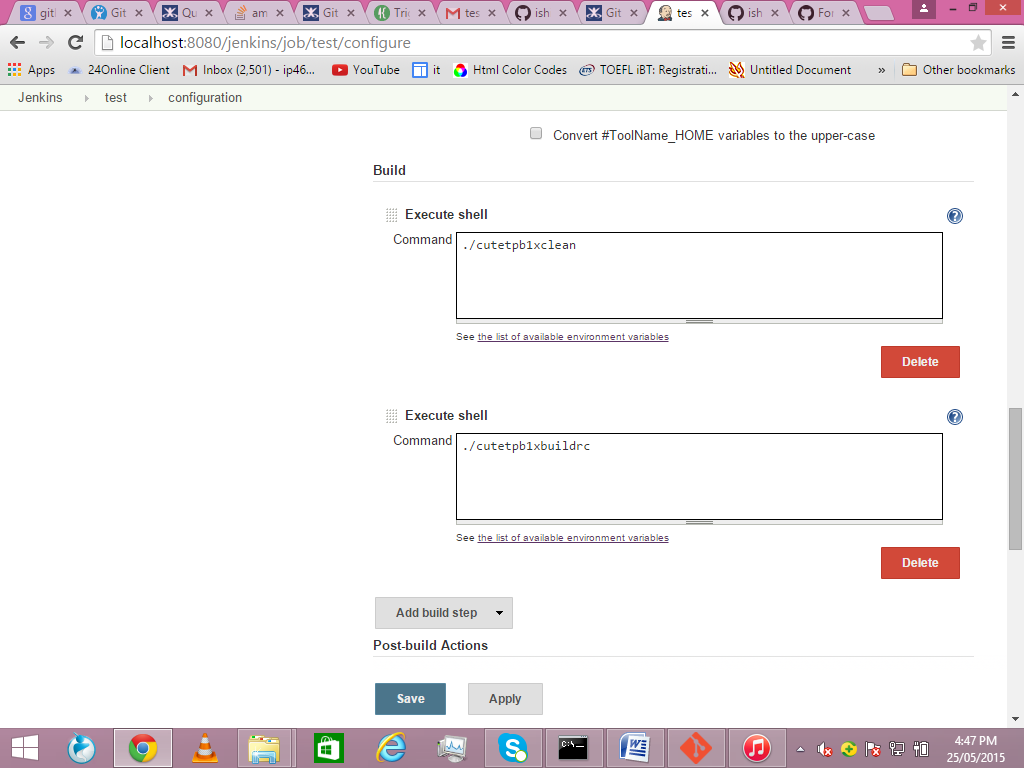
**Step 4.5:** Press configure on the left column.



Select the git option in source code management. Paste the URL of git repository.

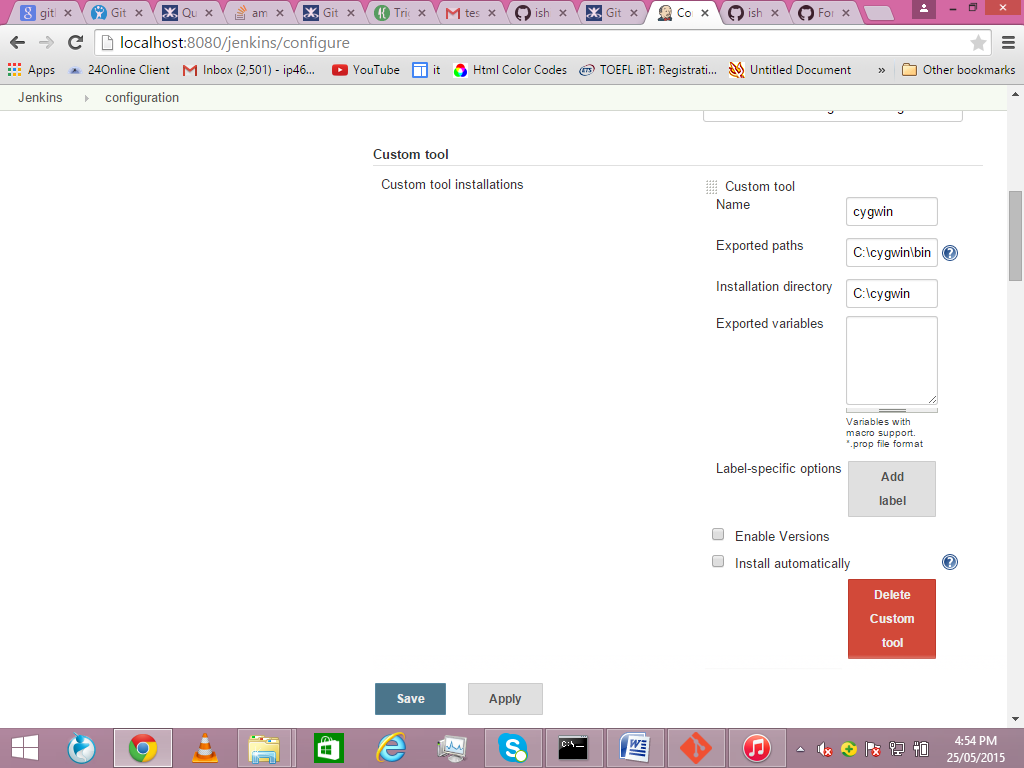
Now add build step as required by your project. Here in this project we need to add shell commands.





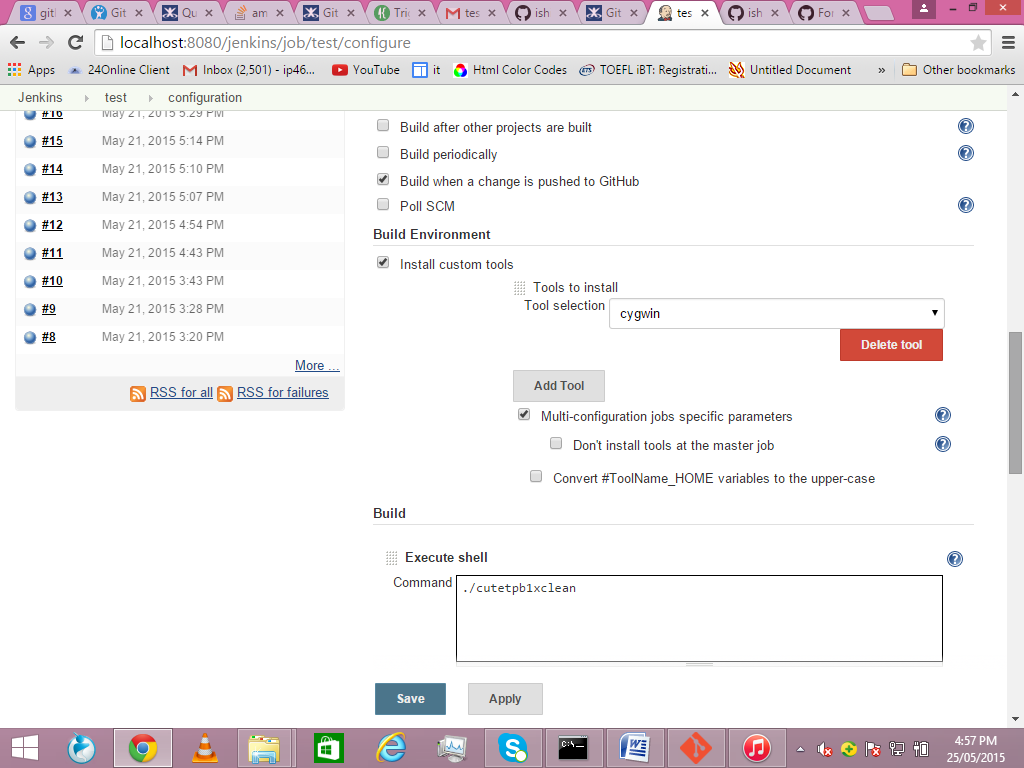
**Step 4.6**: Now we need a tool for compilation. You can automatically install the tool from manage plugins or you can add a tool from your system through custom tools.

For custom tool firstly you need to install custom tool plugin from manage plugins.

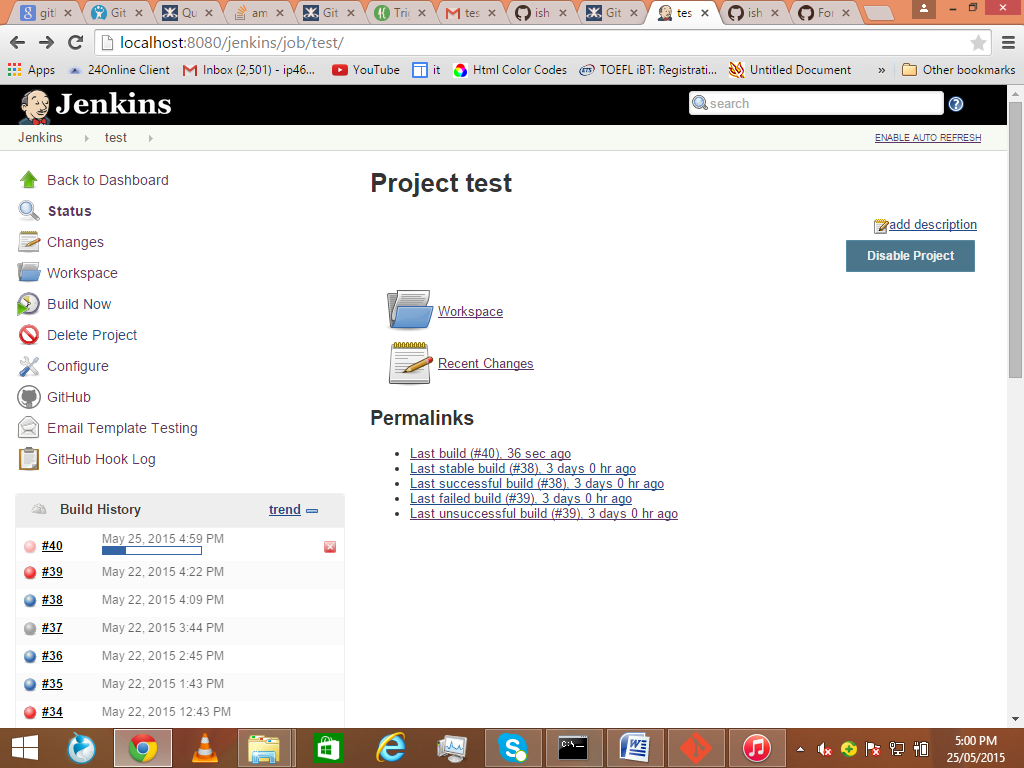


In global configurations add the tool and give the path where the tool is installed in your system.

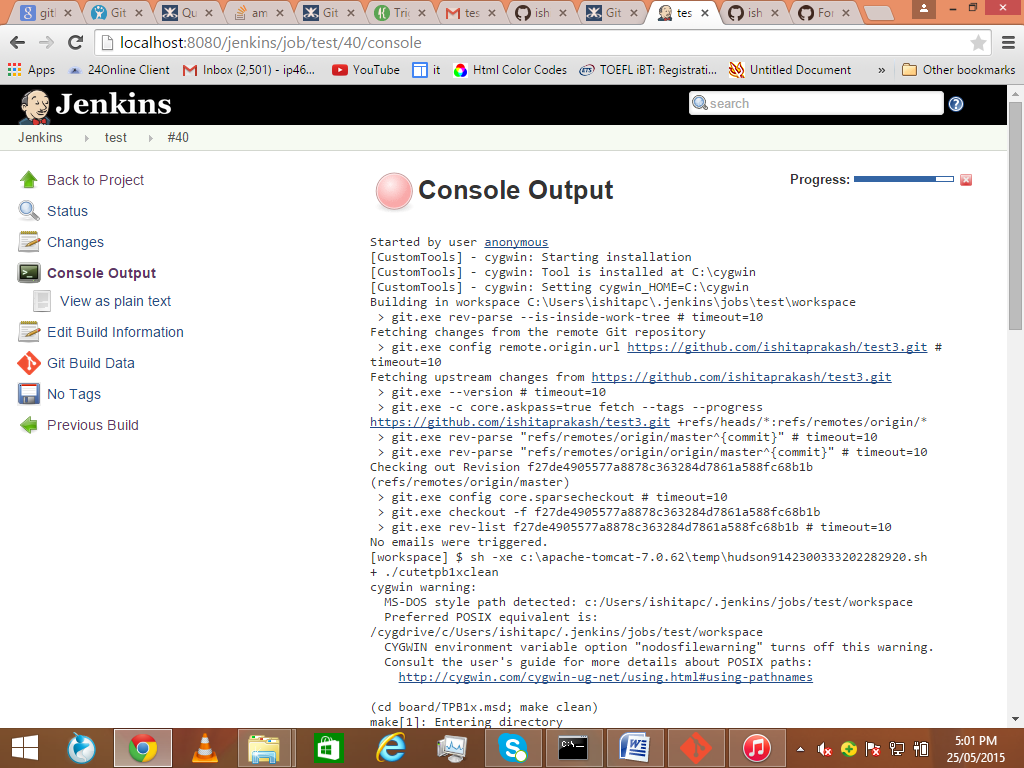
Now in the project configuration select custom tool.



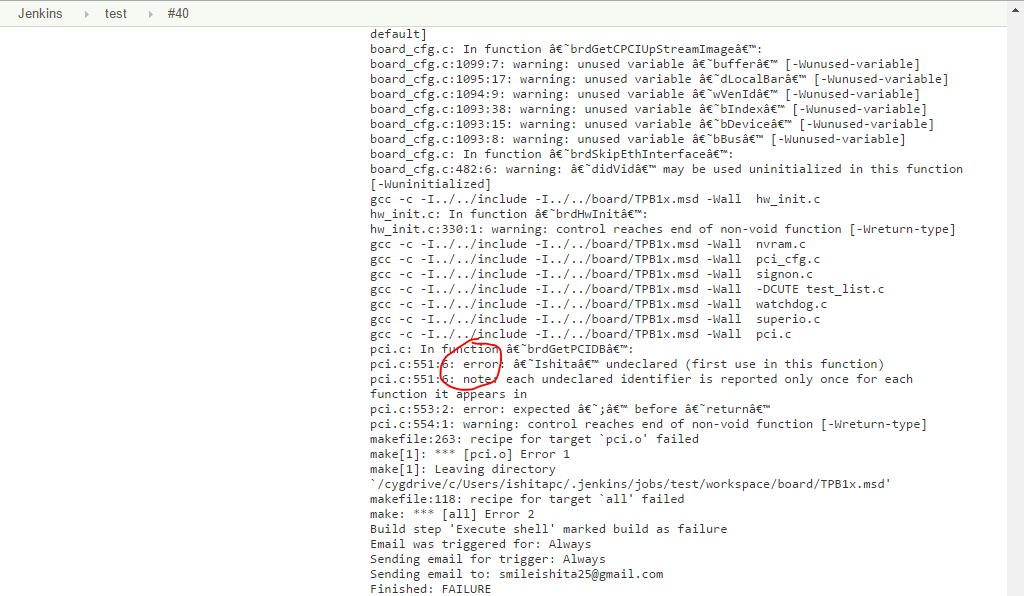
Now save and click build now.



You can press on the build symbol and see the console output.



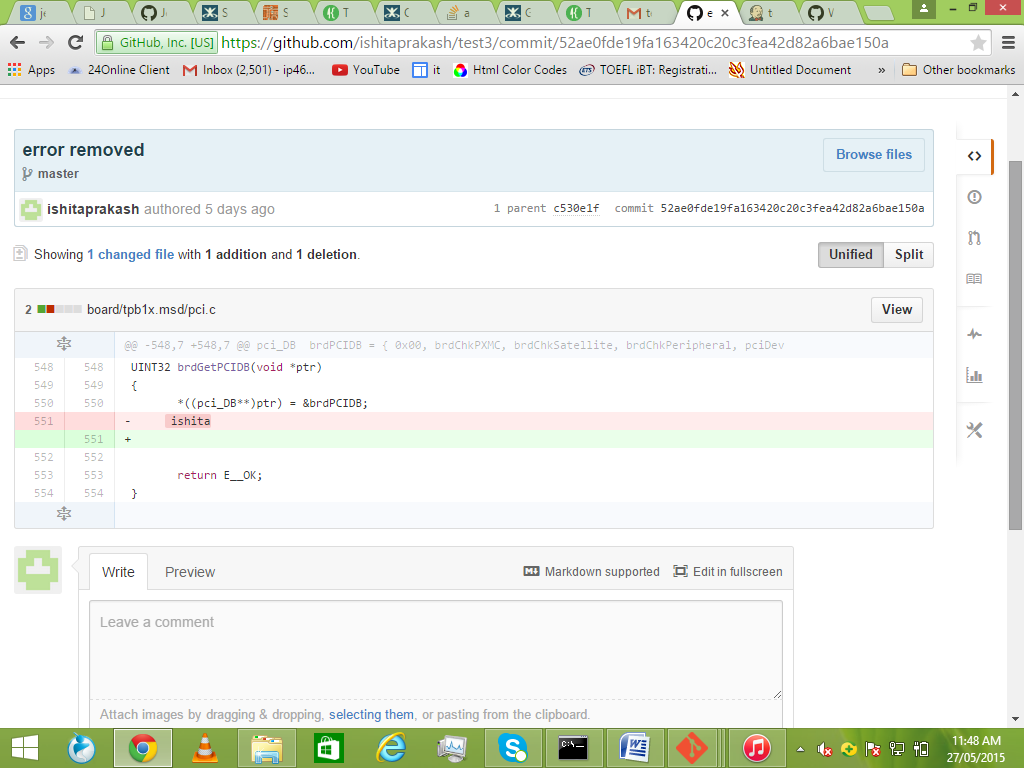
If there is an error , it will be reflected in the console.

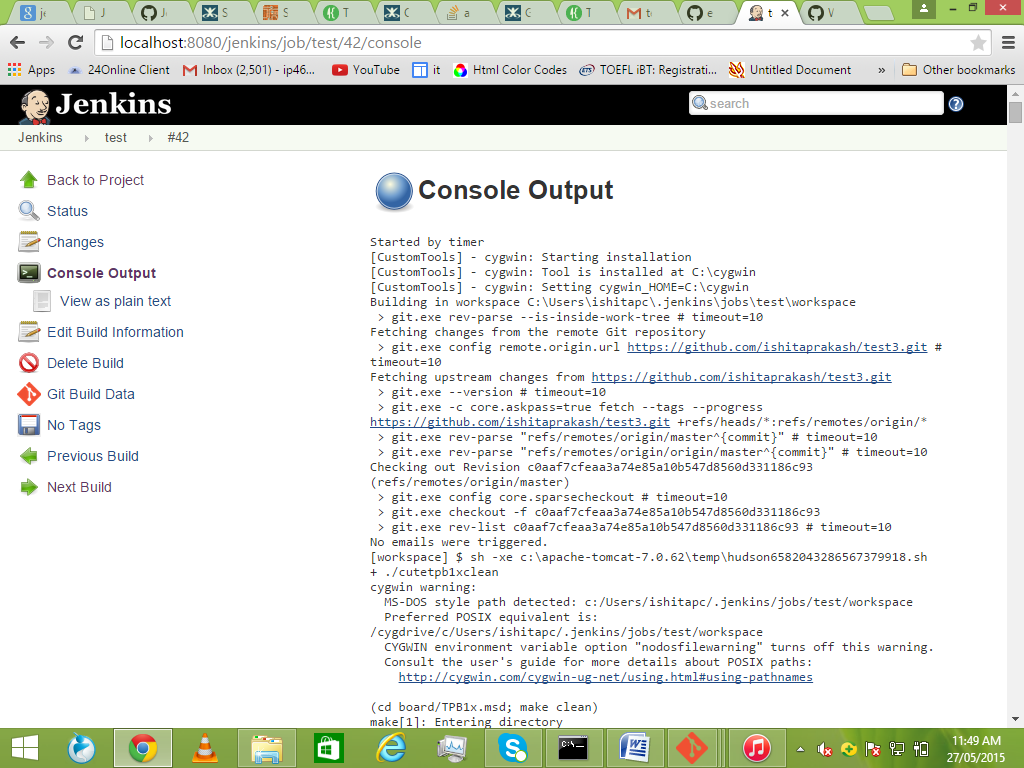


**Step 4.7**: When you remove the error, commit again on Git and build again.

*Note-> for making a new commit, same commands have to be used as before on Git Bash. (Use a different name for remote repo like origin1 instead of origin).*

*Git indicates the file where changes have been made.*



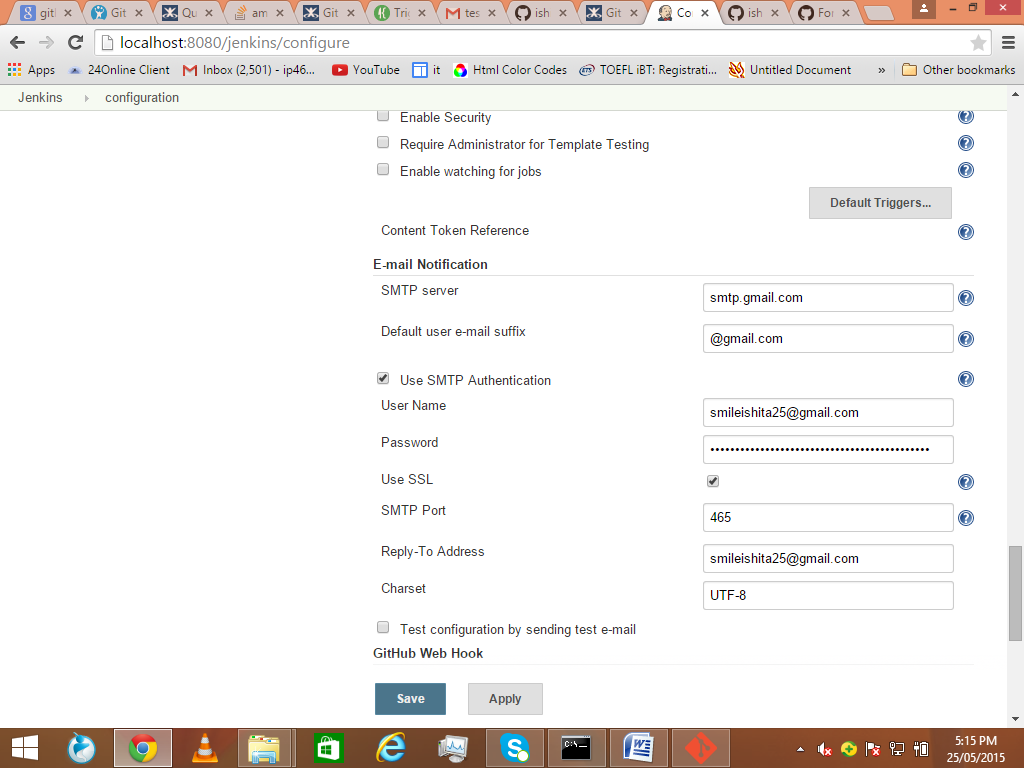


*Note: Blue colour indicates successful build.*

**Step 4.7**: Now if we **want E-mail notification on every build**. First we need to install Email-ext plugin.



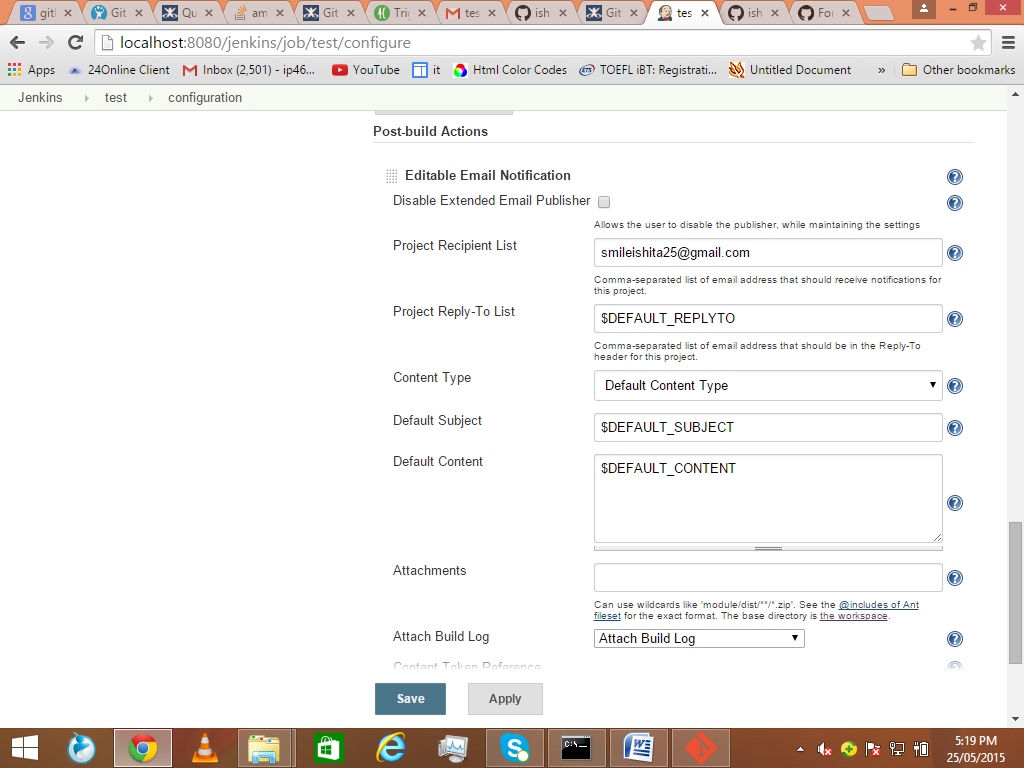
Now go to global configurations and provide the details of E-mail Notification.



Now go to project configurations.

Add a post-build action.

And choose editable e-mail notifications.

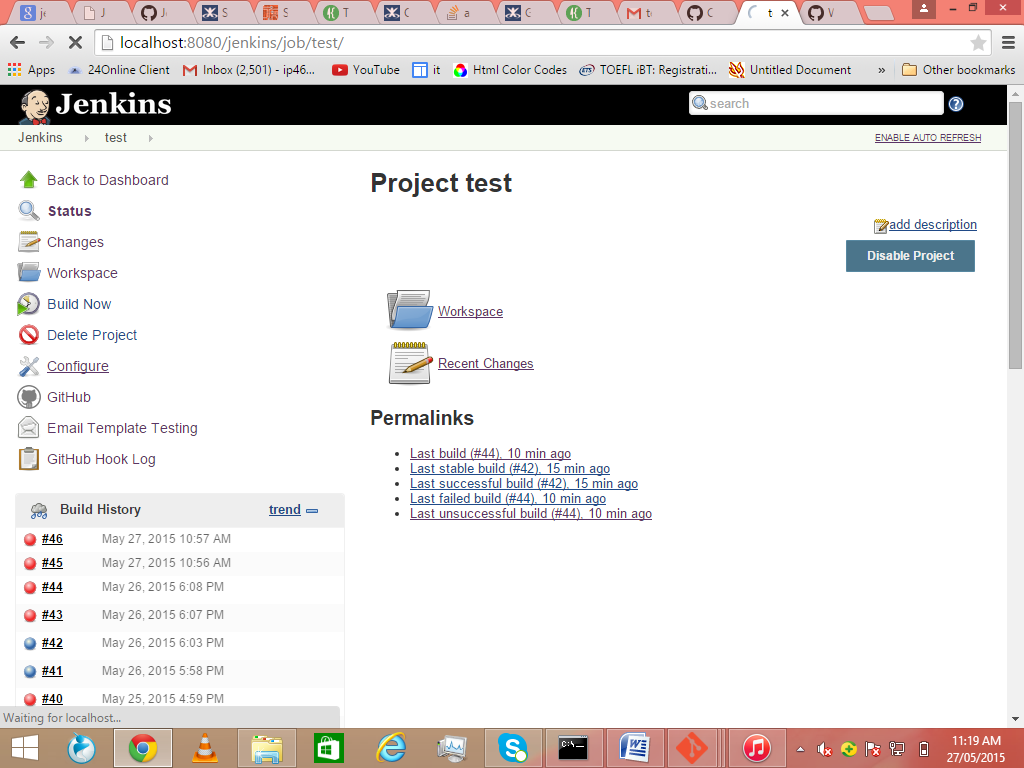


Add your e-mail address. In advanced settings select always.

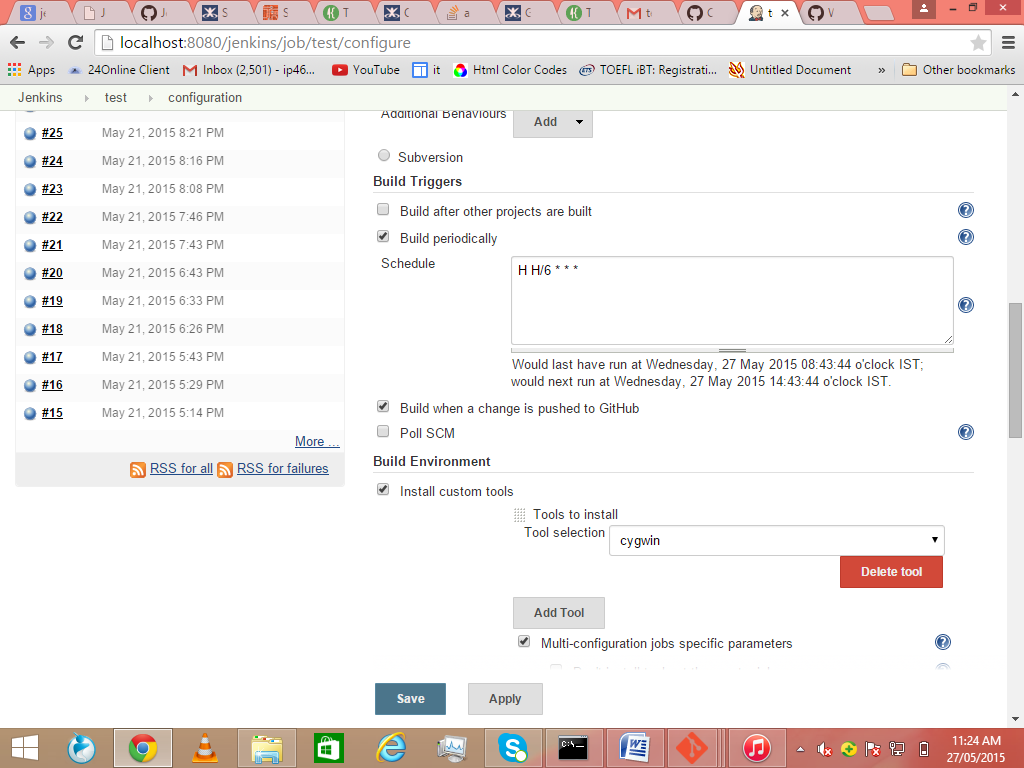
Save and build the project again. You will get a build notification on your mail.

1. **Triggering periodic builds in Jenkins.**

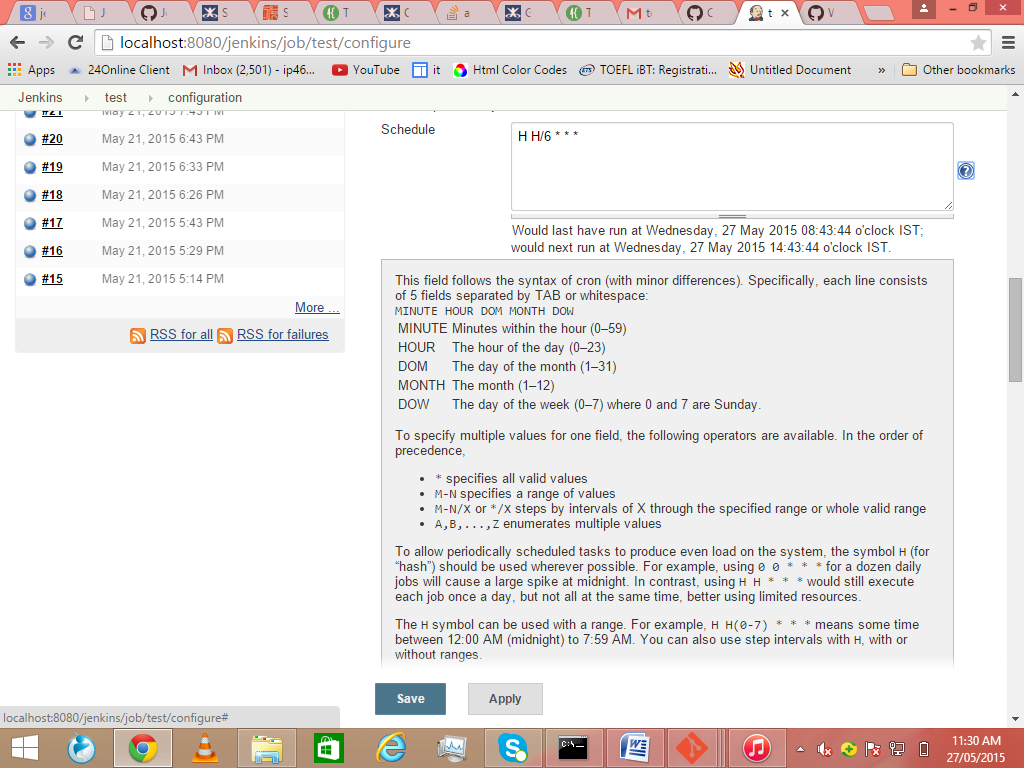
**Step 5.1**: Go to project configuration.



**Step 5.2**: Scroll down to build triggers and select build periodically. And give a schedule according to your requirement.



*Note: You can understand the format of giving schedule by clicking on ‘?’ symbol.*

**

Save and now your project will build periodically, each time checking out files from repository.