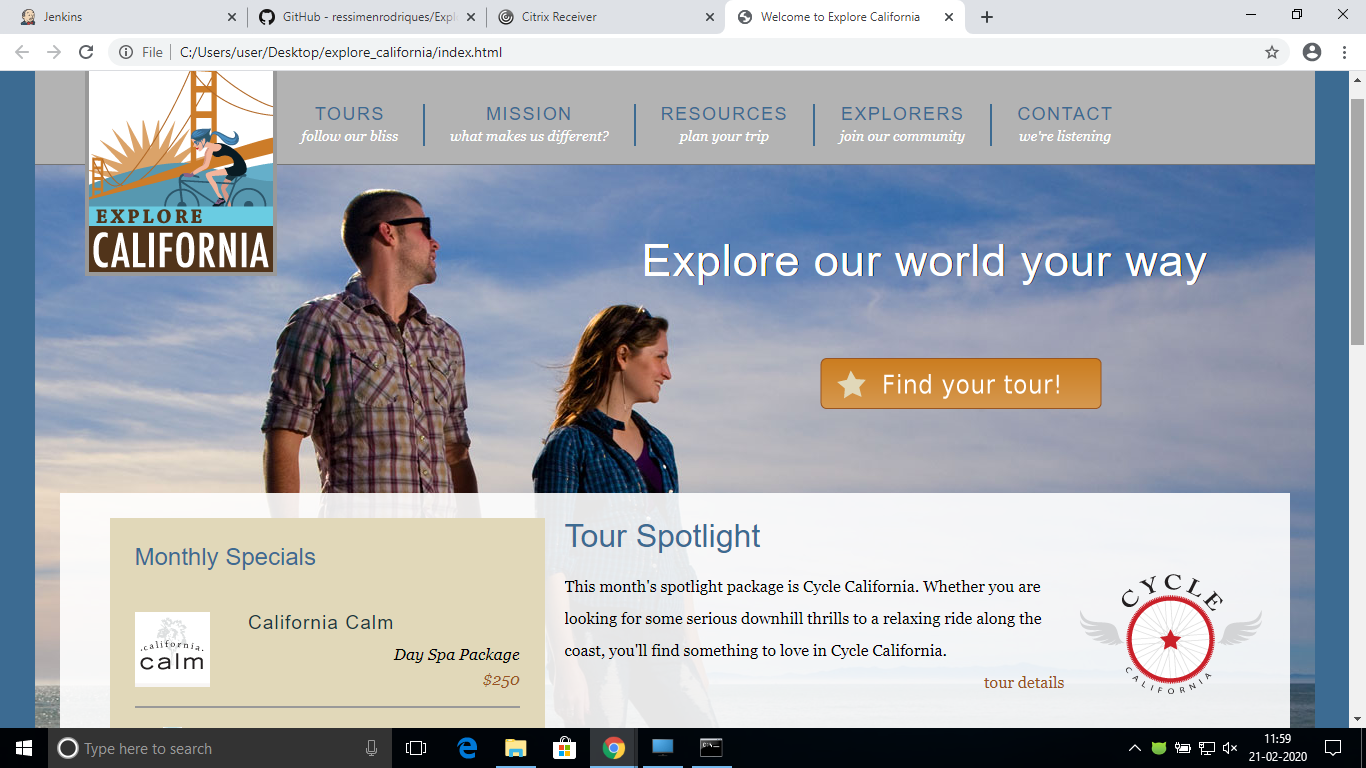
**DevOps Case Study - Explore California**

|  |  |
| --- | --- |
| Document Version | Explore California -DevOps Case Study |
| Date | 20th Feb 2020 |
| Author | Ressimen Rodriques (140636) |
| Revision Number | V1 |
| Revision Date | 20th Feb 2020 |
| Reviewed By |  |
| Approved By |  |

**Contents**

1. About Explore California................................................................................................. 4
2. Scope............................................................................................................................... 4
3. Architecture..................................................................................................................... 4
4. Continuous Integration.................................................................................................... 5
5. CI Integration Process Workflow..................................................................................... 5
6. GIT Introduction & Workflow.......................................................................................... 6
7. Jenkins Introduction........................................................................................................ 8
8. Jenkins Workflow............................................................................................................ 9
9. CatLight Introduction...................................................................................................... 18
10. Continuous Deployment................................................................................................. 19
11. Ansible............................................................................................................................ 19
12. **About Explore California**

Our goal at Explore California is to transform your vacation into an adventure that will educate, inspire, and energize you unlike any other. Our tours are crafted around our central mission, and are designed to engage you in a unique and fulfilling way. All our tours are sensitive to the environment, and will provide you will an opportunity to explore California in your own way.



1. **Scope**

This document provides a detailed use case study on development and operations (DevOps) using Continuous integration and continuous delivery (CI/CD) with guidelines for implementation.

1. **Architecture**

DevOps - CI / CD architecture

Code Repository - GIT

Web Server – Apache TomCat 8.5.50

Build -Jenkins 2.204.2

Deployment - Ansible

Monitoring – Nagios

1. **Continous Integration**

Continuous integration (CI) is a software development practice where developers regularly merge their code changes into a central repository, after which automated builds and tests are run. CI most often refers to the build or integration stage of the software release process and requires both an automation component (e.g., a CI or build service) and a cultural component (e.g., learning to integrate frequently).

The key goals of CI are to find and address bugs more quickly, improve software quality, and reduce the time it takes to validate and release new software updates.

Continuous integration focuses on smaller commits and smaller code changes to integrate. A developer commits code at regular intervals, at minimum once a day. The developer pulls code from the code repository to ensure the code on the local host is merged before pushing to the build server. At this stage the build server runs the various tests and either accepts or rejects the code commit.

The basic challenges of implementing CI include more frequent commits to the common codebase, maintaining a single source code repository, automating builds, and automating testing. Additional challenges include testing in similar environments to production, providing visibility of the process to the team, and allowing developers to easily obtain any version of the application.

1. **CI Integration Process Workflow**

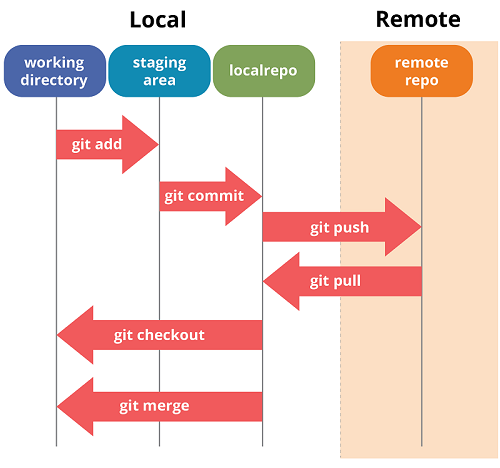
* Develop the web app using HTML, CSS, Bootstrap
* Build the code using GIT
* Install Apache Tomcat to run Jenkins
* Email notifications to the team who made the build success or failed using Jenkins

git.png 

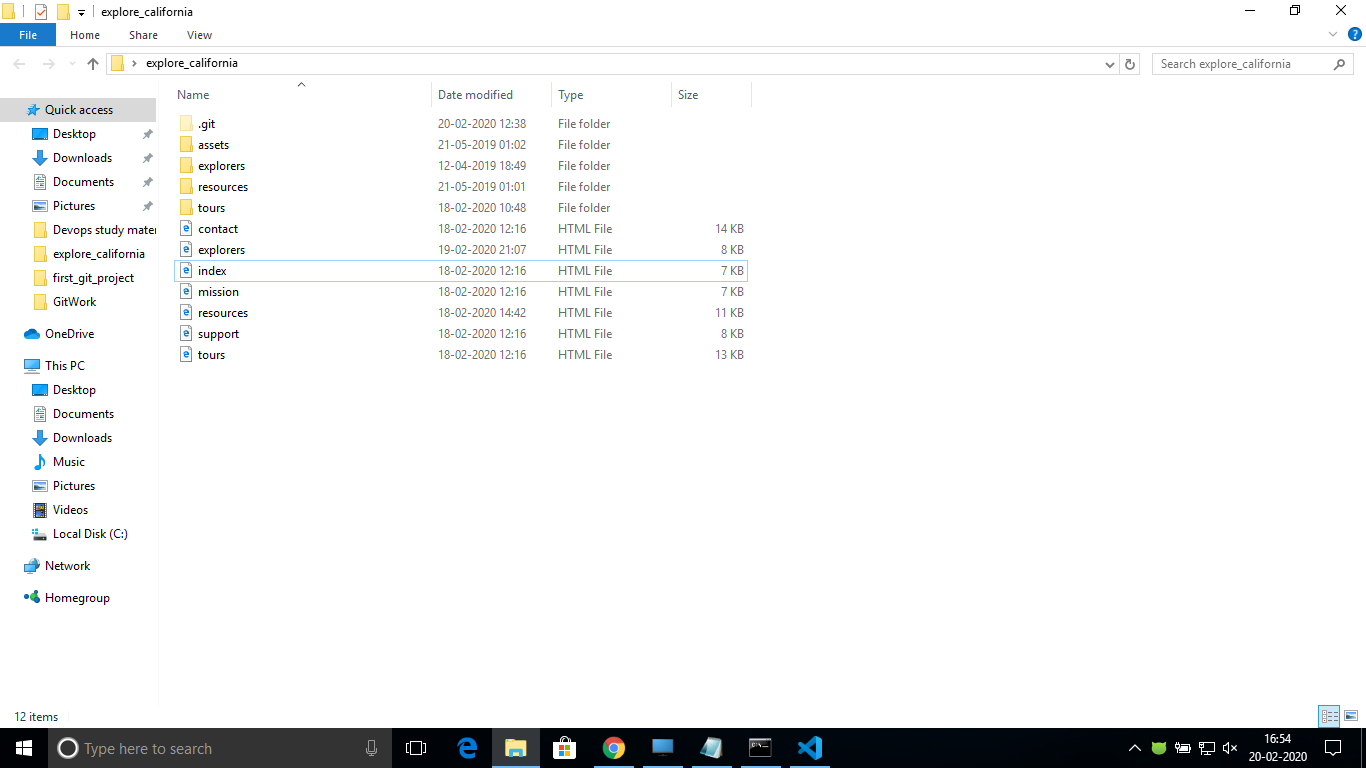
 jenkins.png

1. **GIT Introduction & Workflow:**

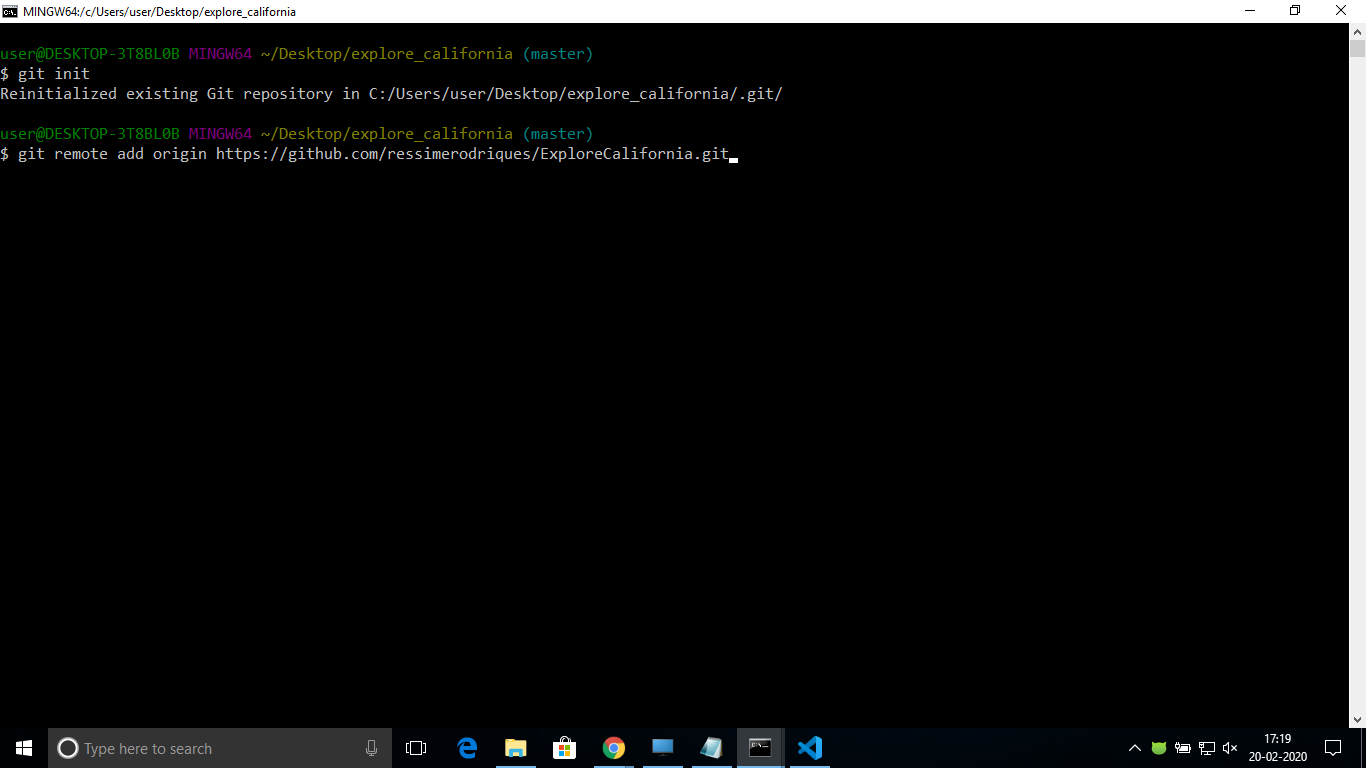
Git is a free and open source distributed version control system designed to handle everything from small to very large projects with speed and efficiency.



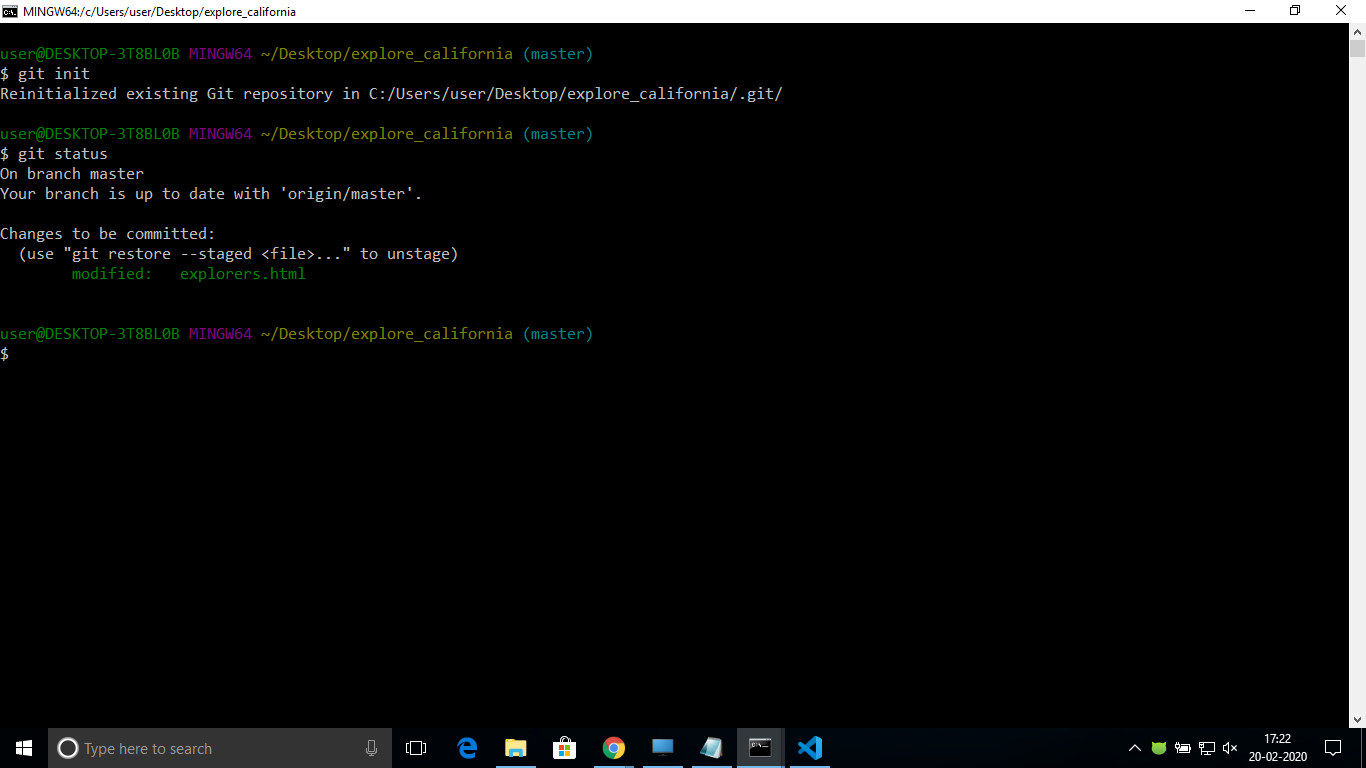
Download the latest git from git-scm.com and install the software. Go to the project folder and right click and select **Git bash here.** Git will be initialised and cmd prompt will open.

****

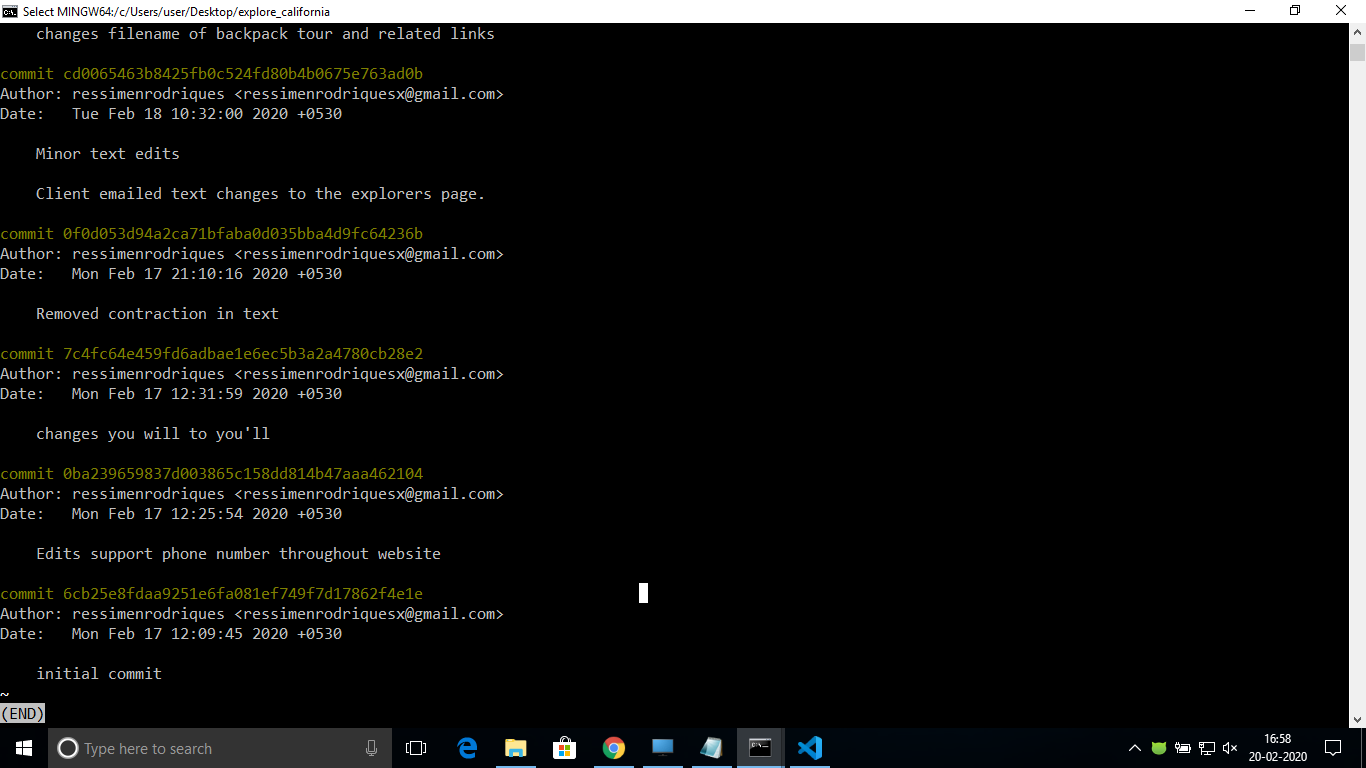
Initializing GIT repository as per the below screenshot

****

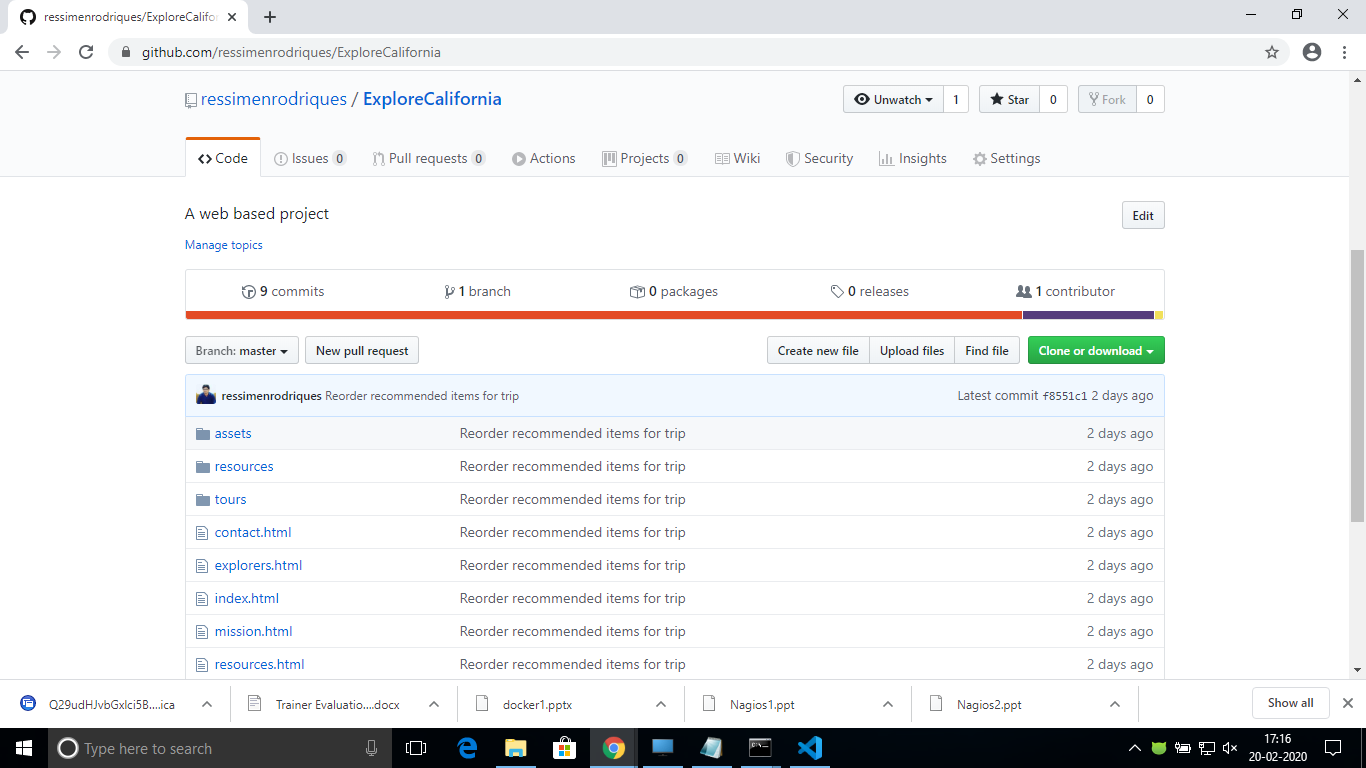
Committingchanges in GIT

****

Git Log to see the commits as shown in the below screenshot.

****

Log in to github.com and create a repository and add the remote repository to the project.

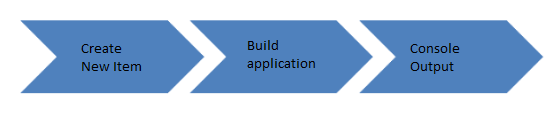
****

1. **Jenkins Introduction:**

Jenkins is a free and open source automation server. Jenkins helps to automate the non-human part of the software development process, with continuous integration and facilitating technical aspects of continuous delivery. It is a server-based system that runs in servlet containers such as Apache Tomcat.

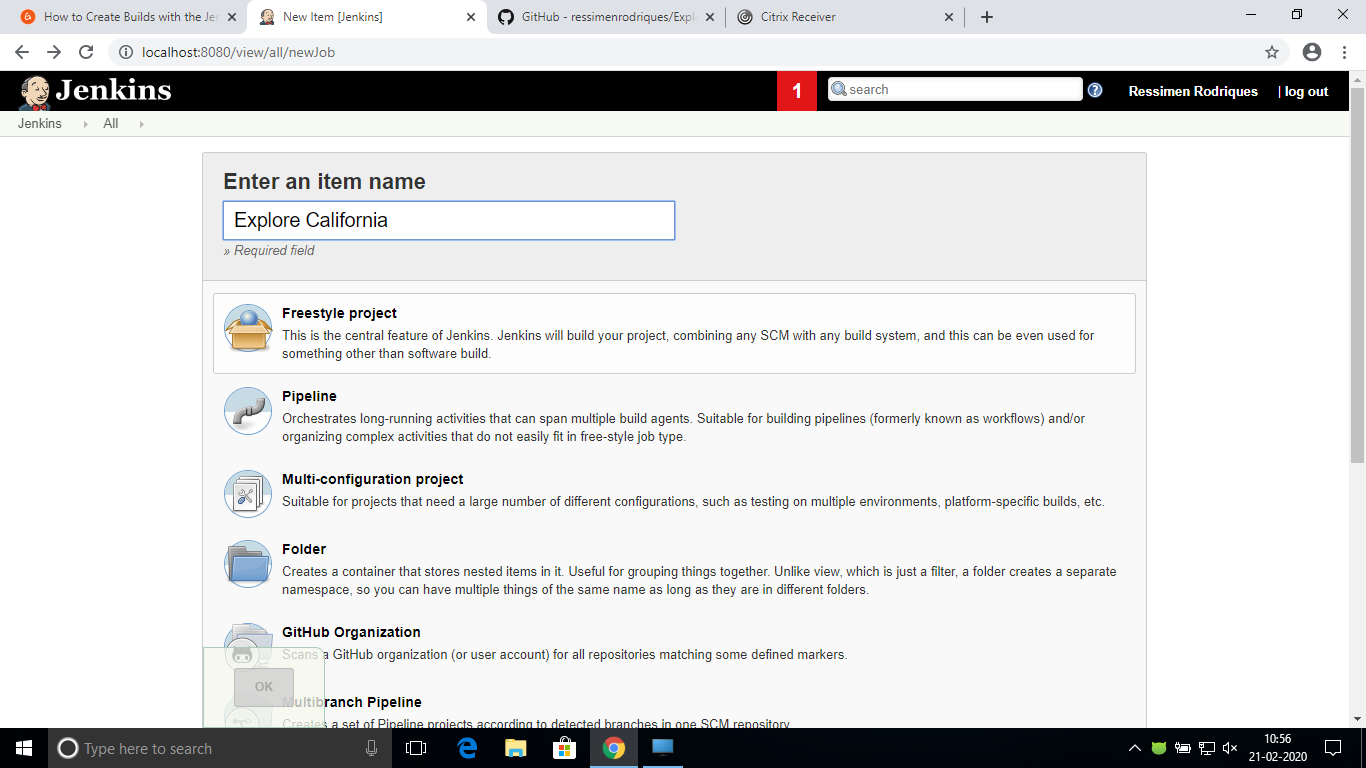
It supports version control tools, including AccuRev, CVS, Subversion, Git, Mercurial, Perforce, TD/OMS, ClearCase and RTC, and can execute Apache Ant, Apache Maven and sbt based projects as well as arbitrary shell scripts and Windows batch commands. The creator of Jenkins is Kohsuke Kawaguchi. Released under the MIT License, Jenkins is free software.

1. **Jenkins Workflow:**

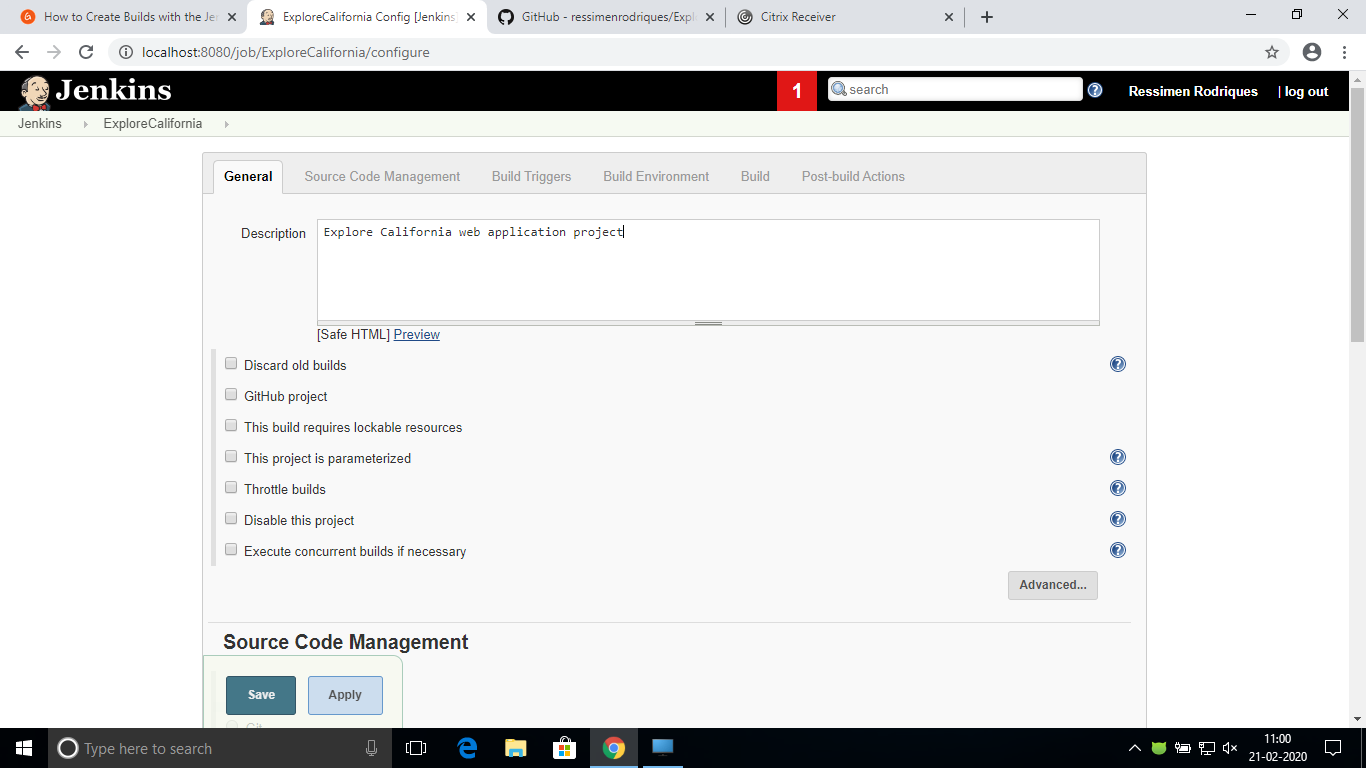
****

Log on to your Jenkins dashboard by visiting your Jenkins installation path

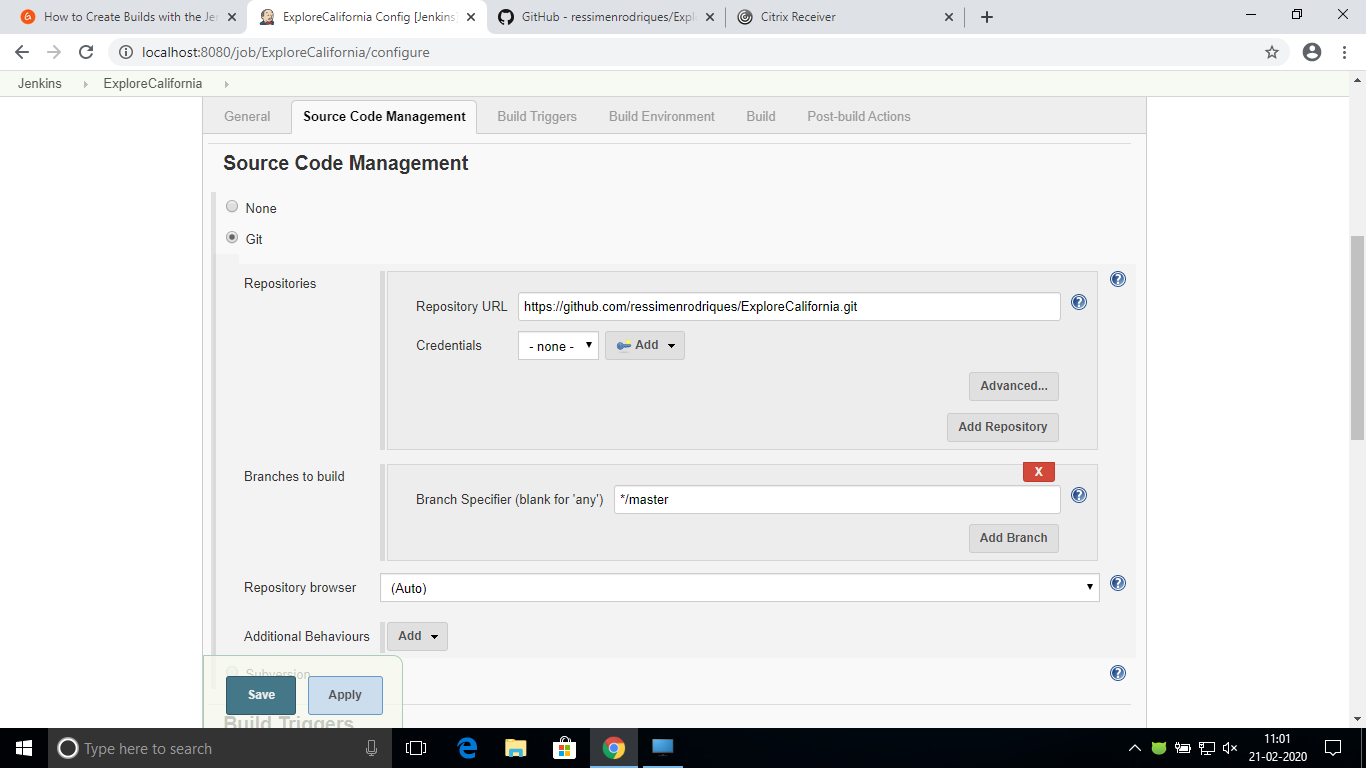
Click on New Item at the top left corner of your dashboard. Enter the name of the project and click on freestyle project.

****

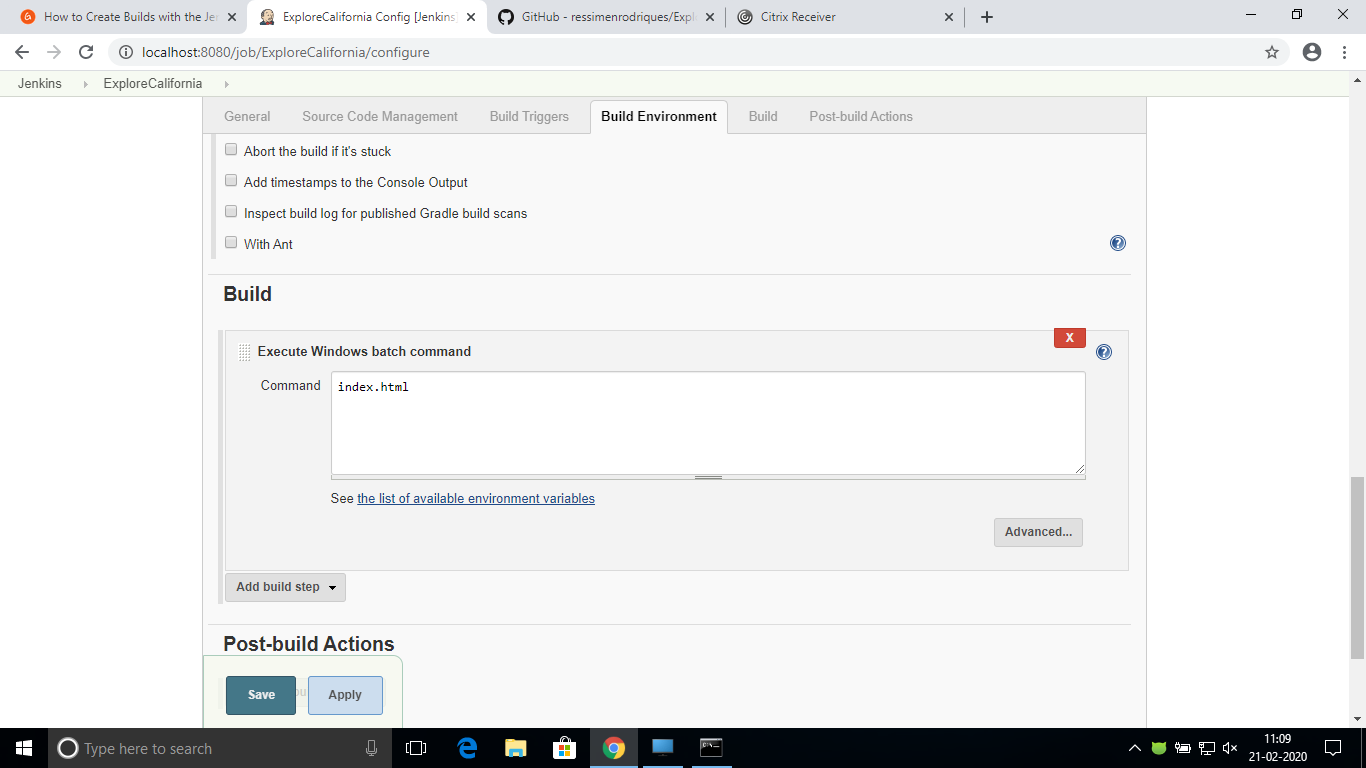
After the project is created go to the project 🡪Configure you will see the below screen. Add a description about the project.

****

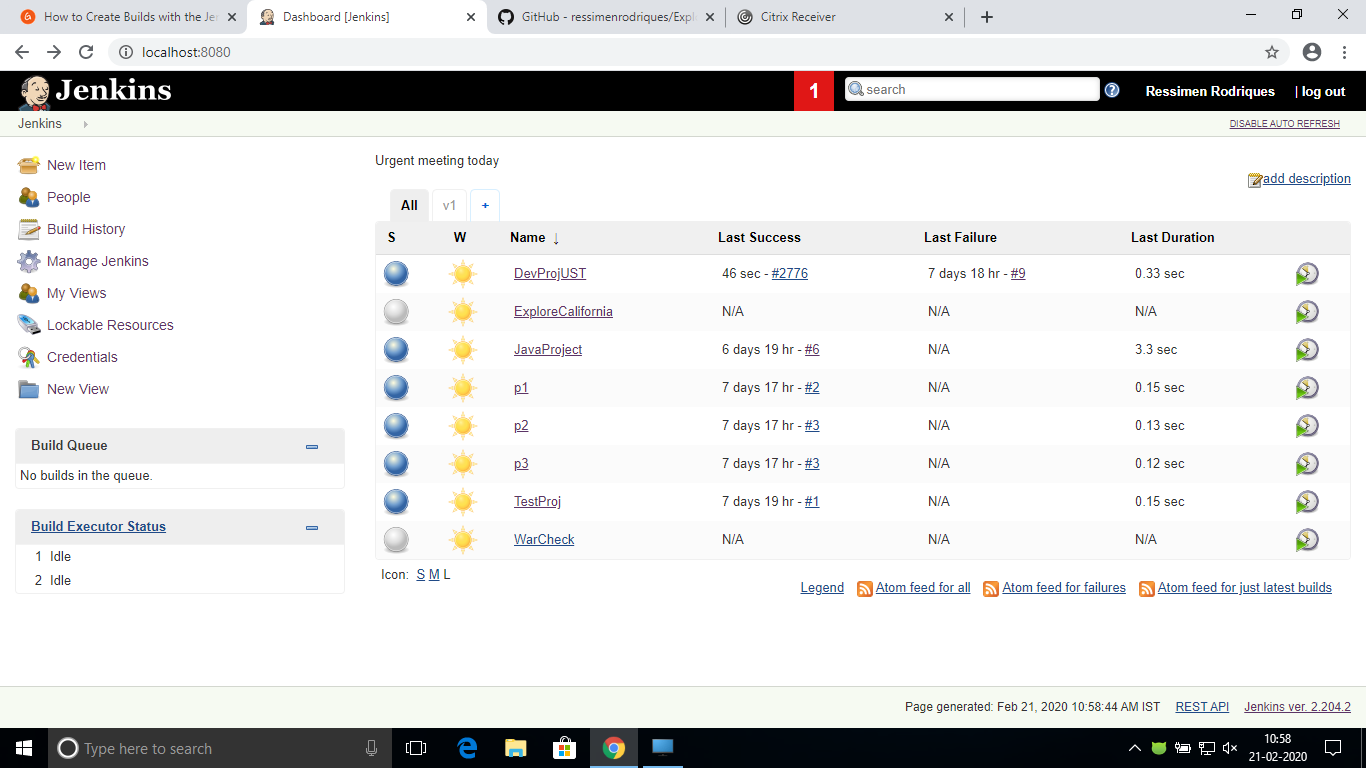
Under the Source Code Management tab provide the Repository Url of the project.

****

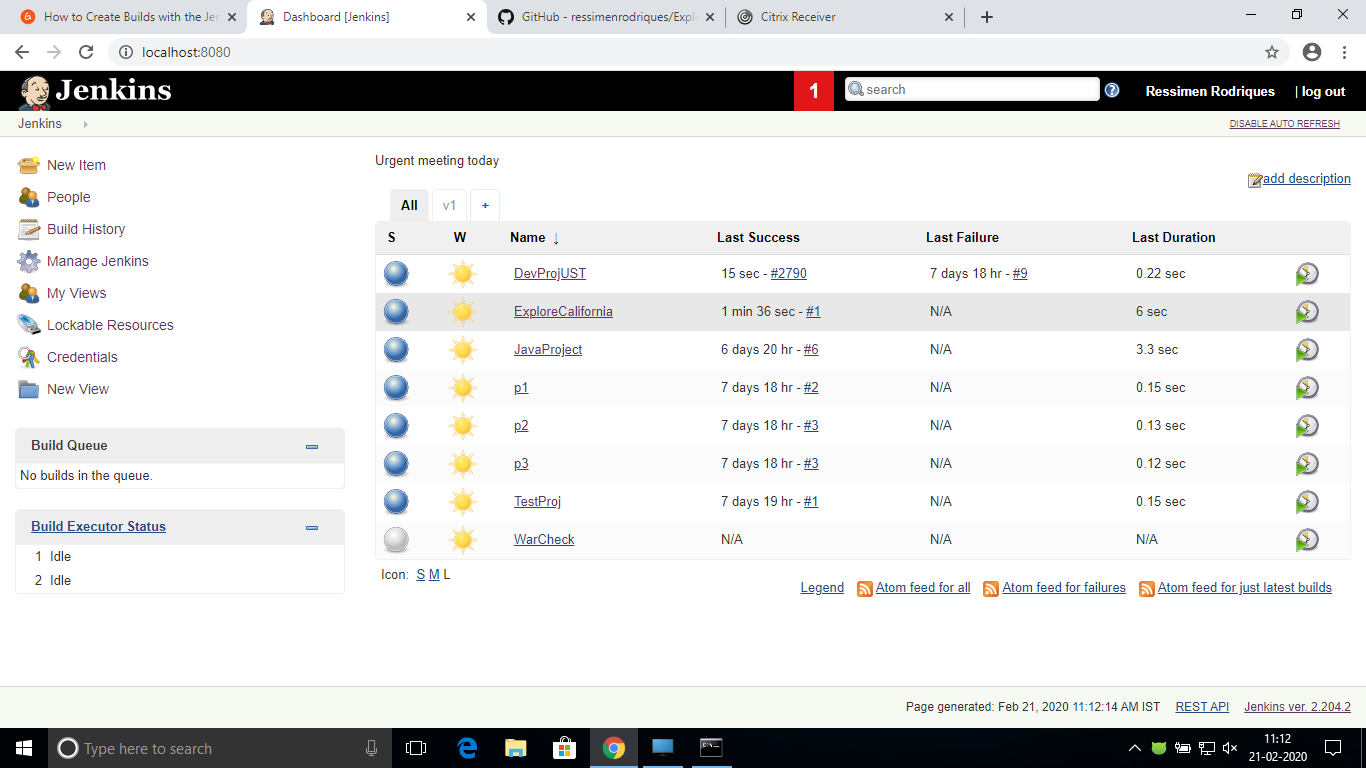
In the Build Environment tab add the desired command under Build

****

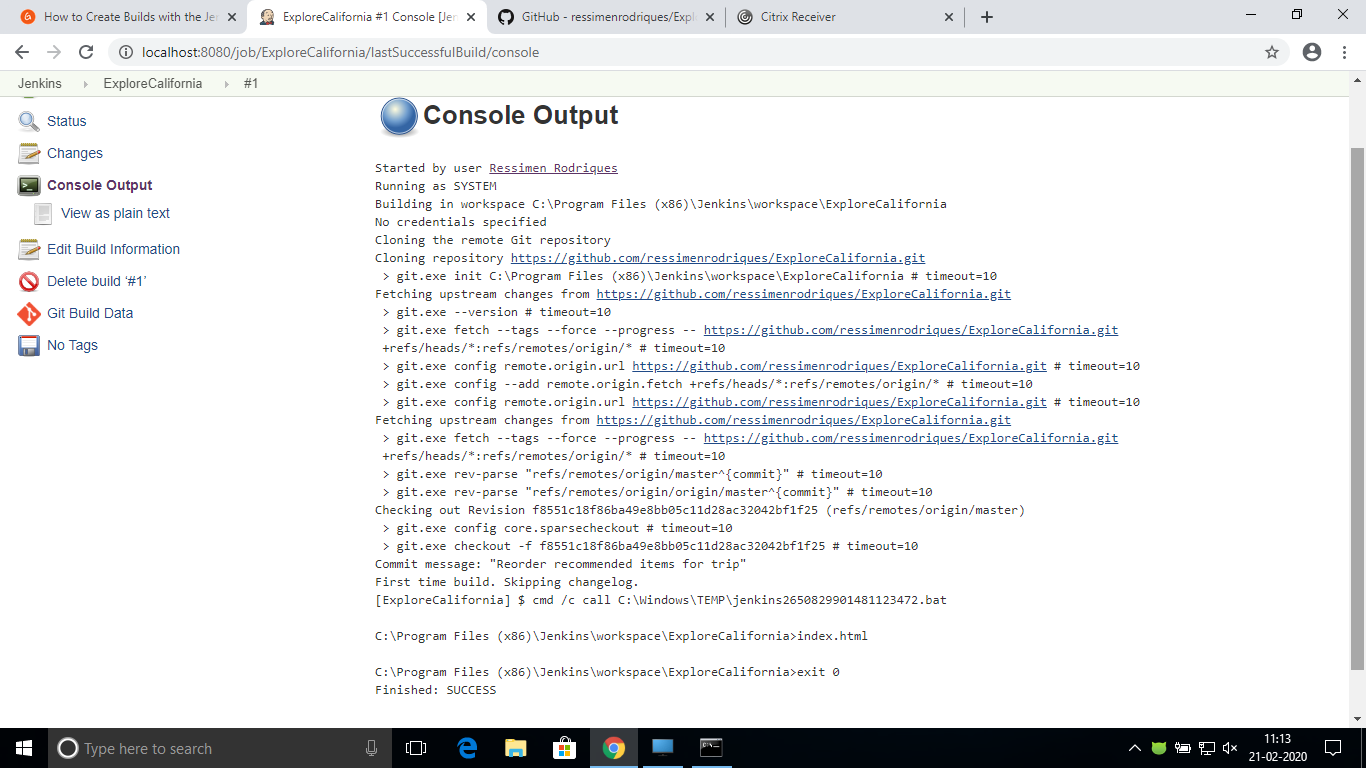
Build the project after configuring.

****

The dashboard should lists the build as successful

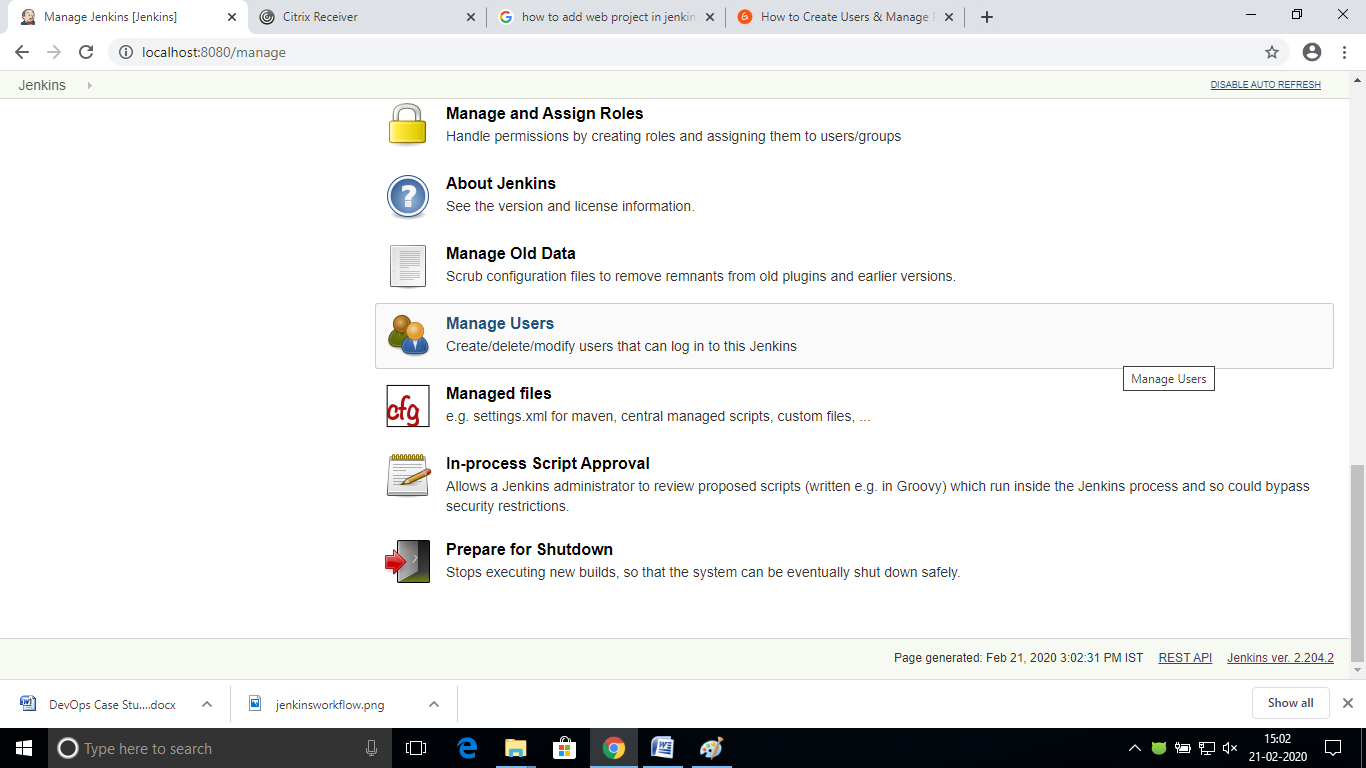
****

**Console Output:**

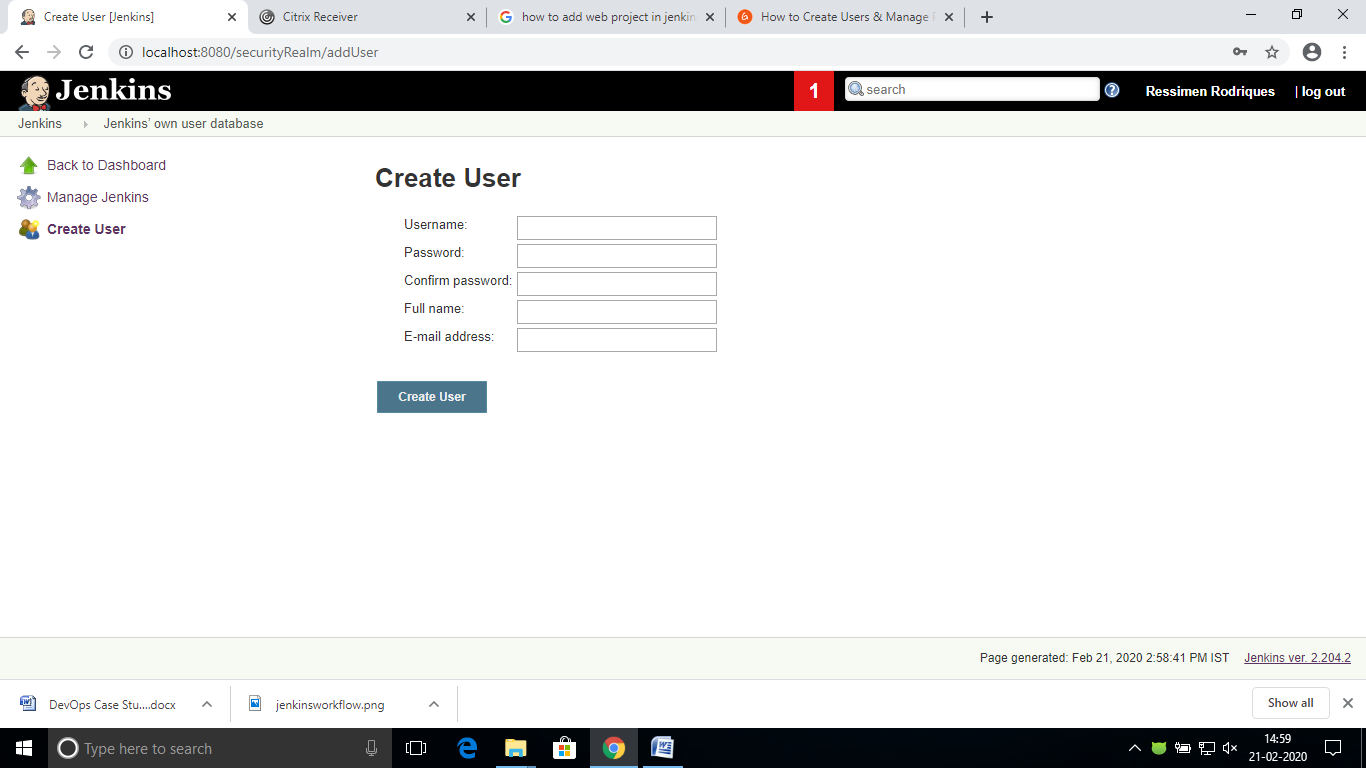
****

**Creating New Users**

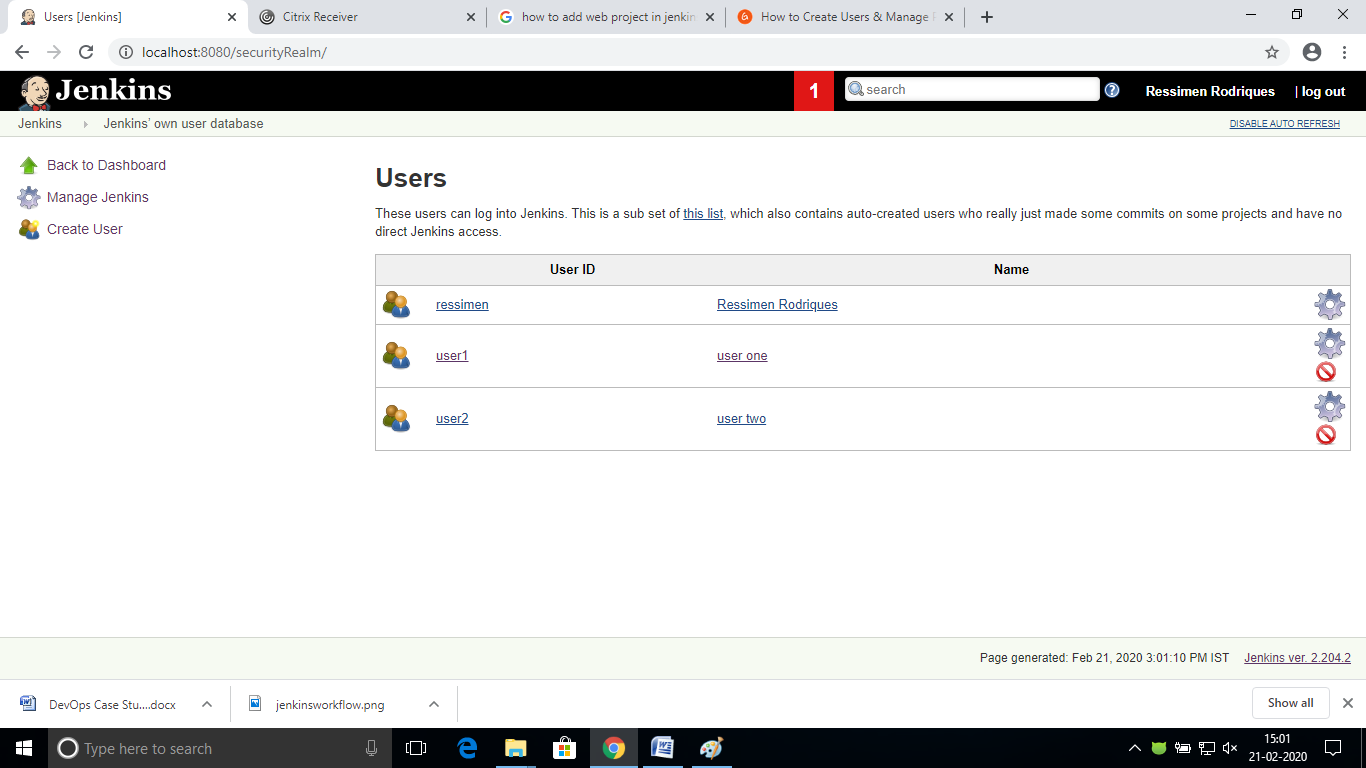
Go to manage jenkins and click on Manage Users.

****

Click On Create User

****

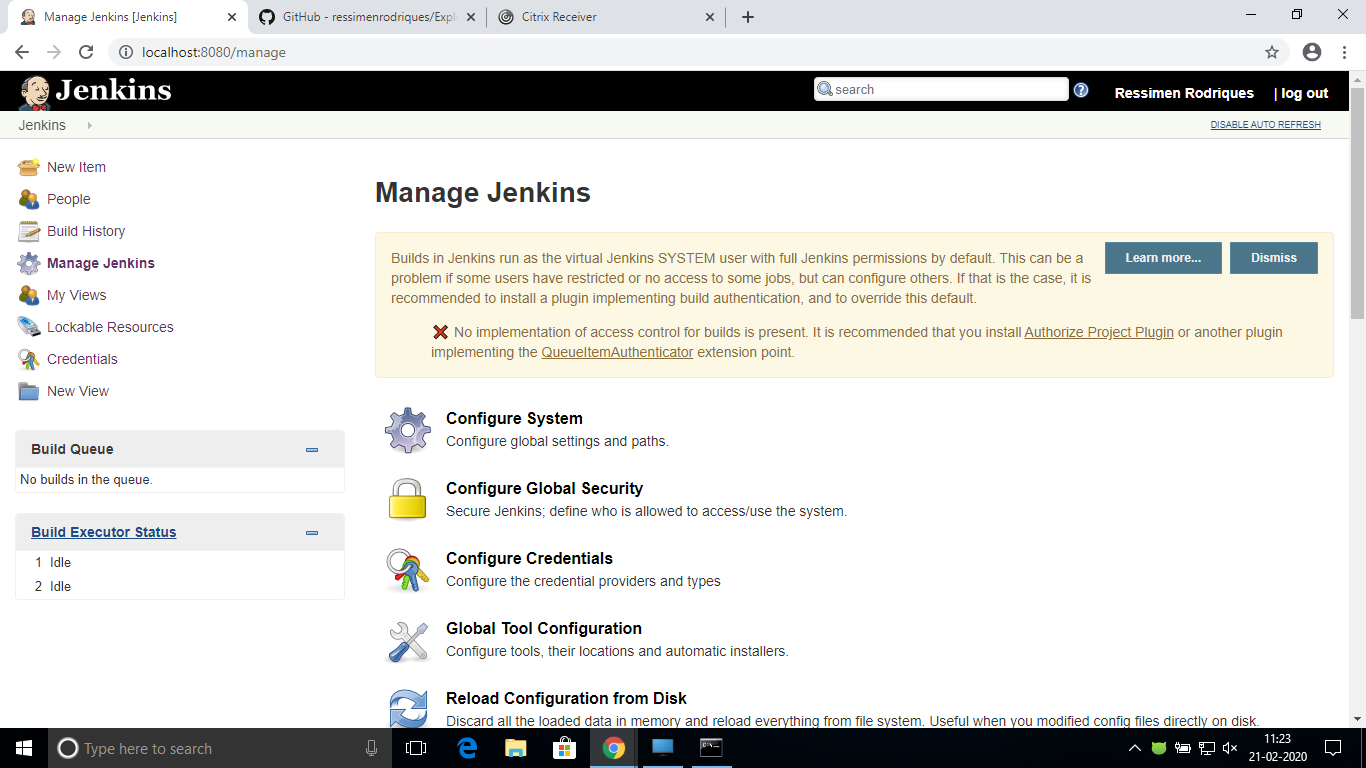
Newly created users will be listed as shown in the below screenshot.

****

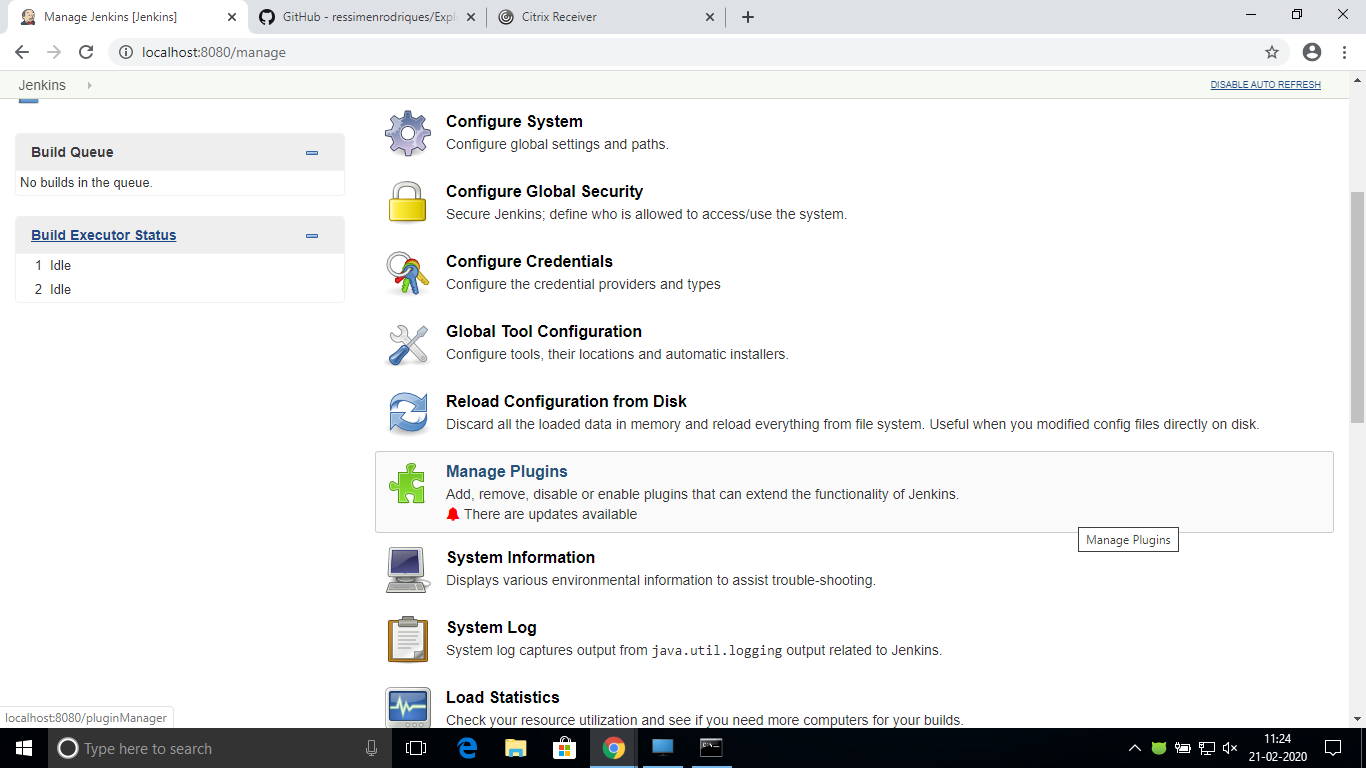
**Creating Roles and Assigning Roles:**

The **Role Strategy Plugin**enables you to assign different roles and privileges to different users**.** You will first need to install the plugin in your Jenkins manage environment.

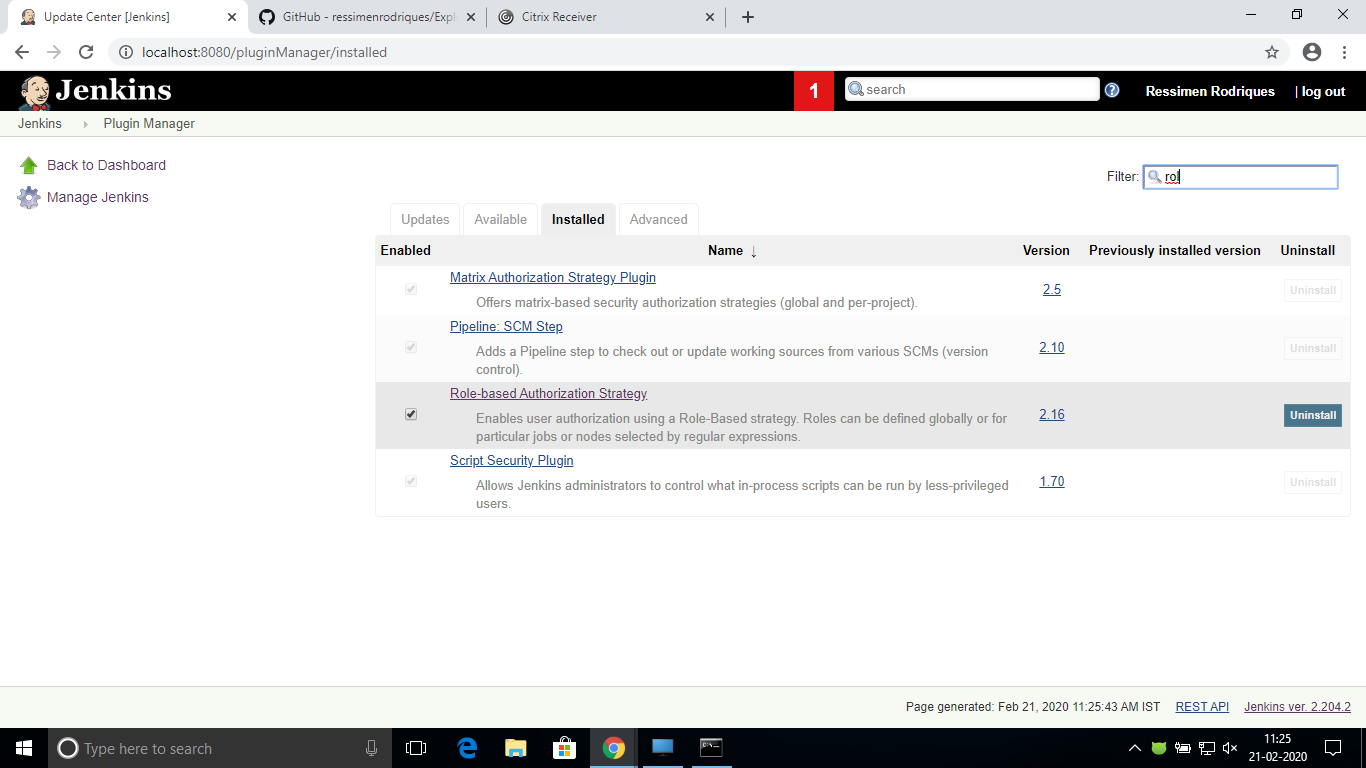
Go to Manage jenkins and select manage plugins

****

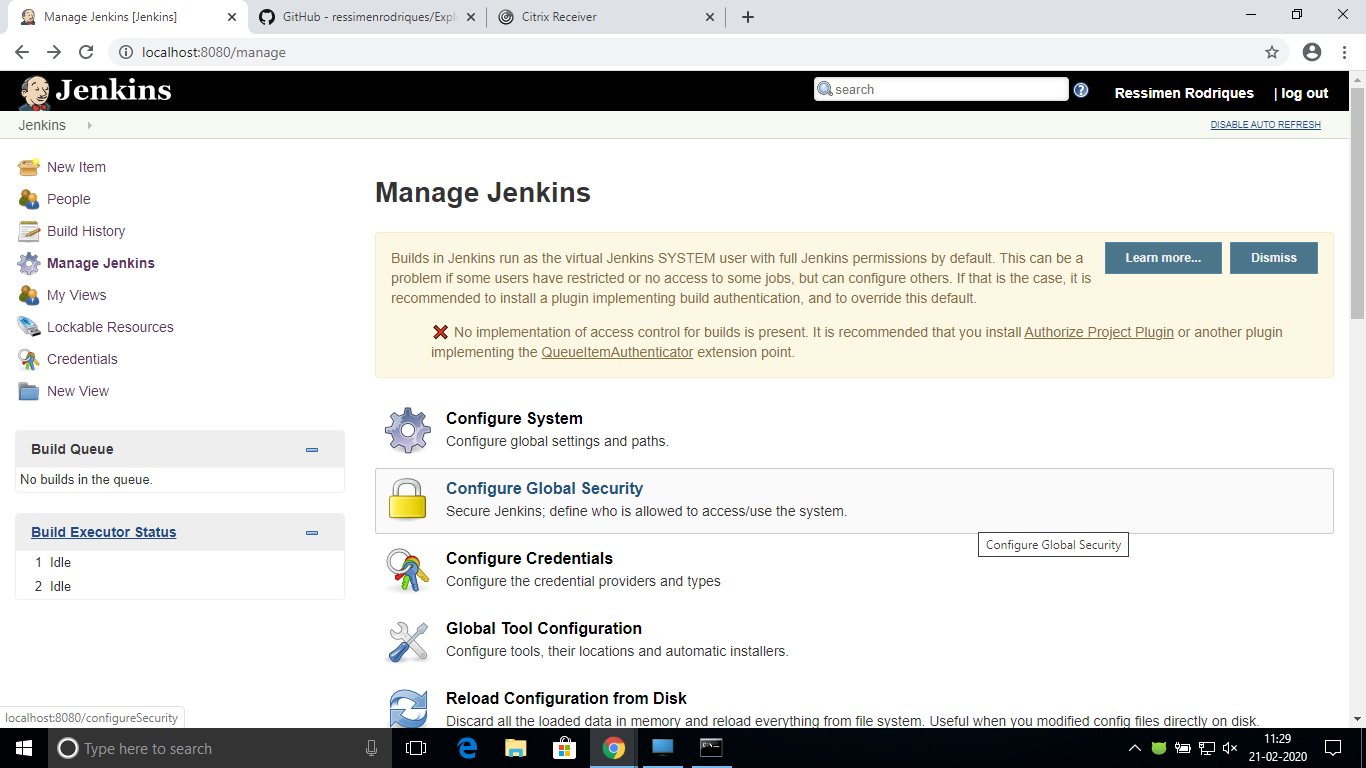
**Manage Plugins**

****

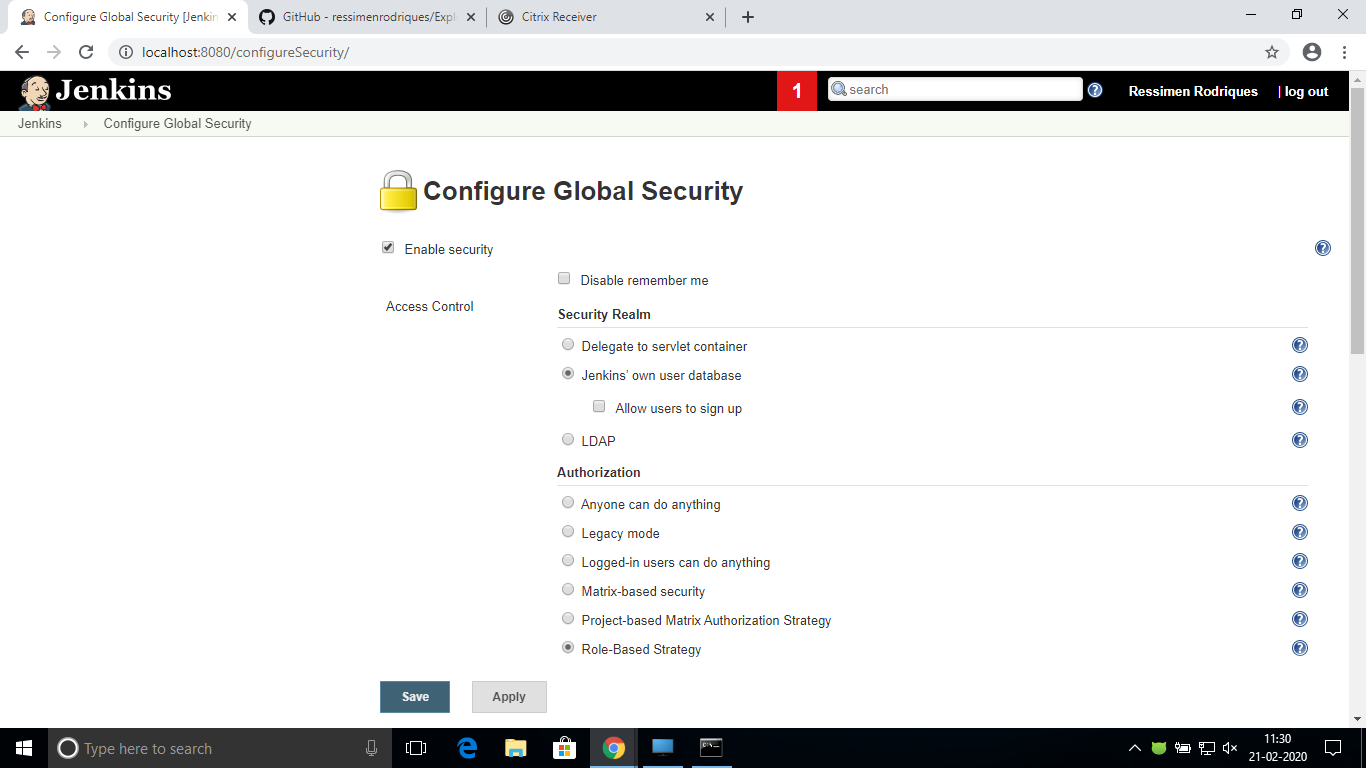
Filter using **Role based strategy plugin**

****

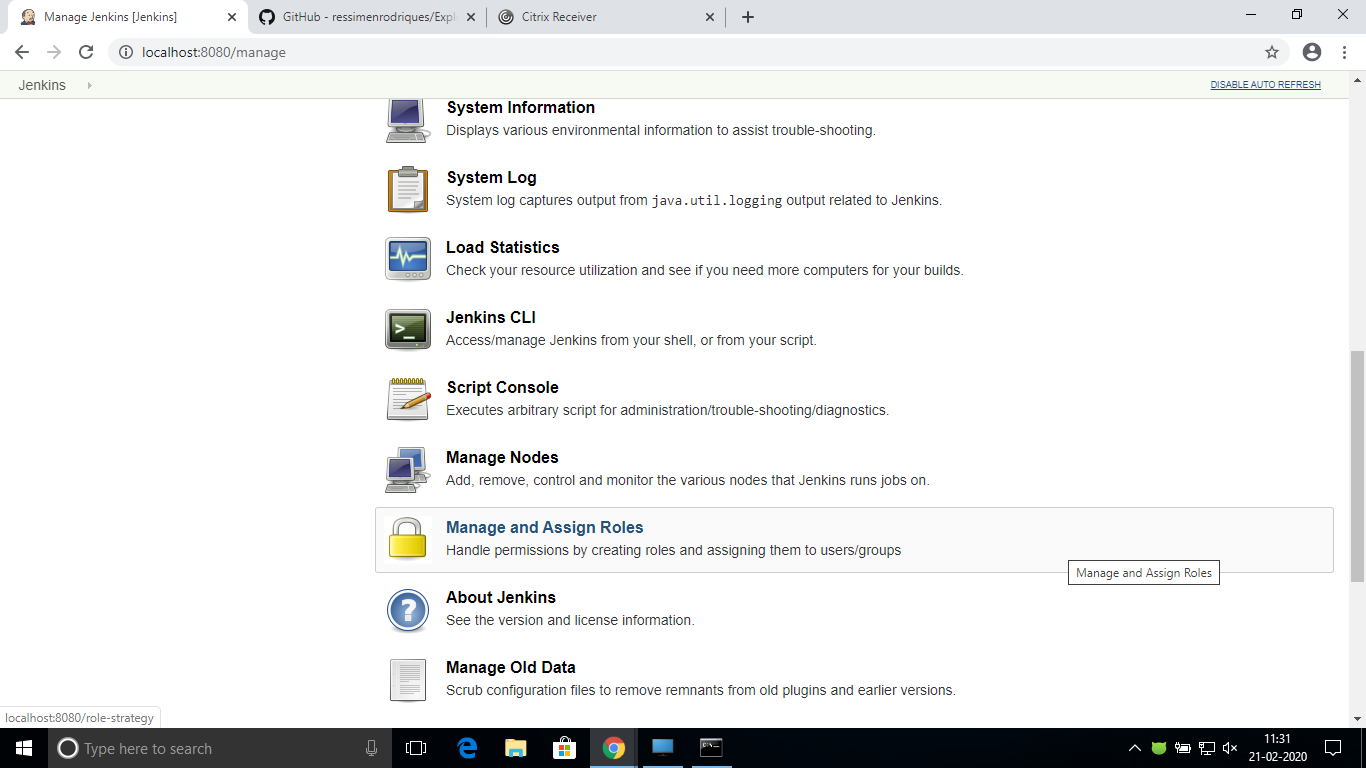
**Configure Global Security**

****

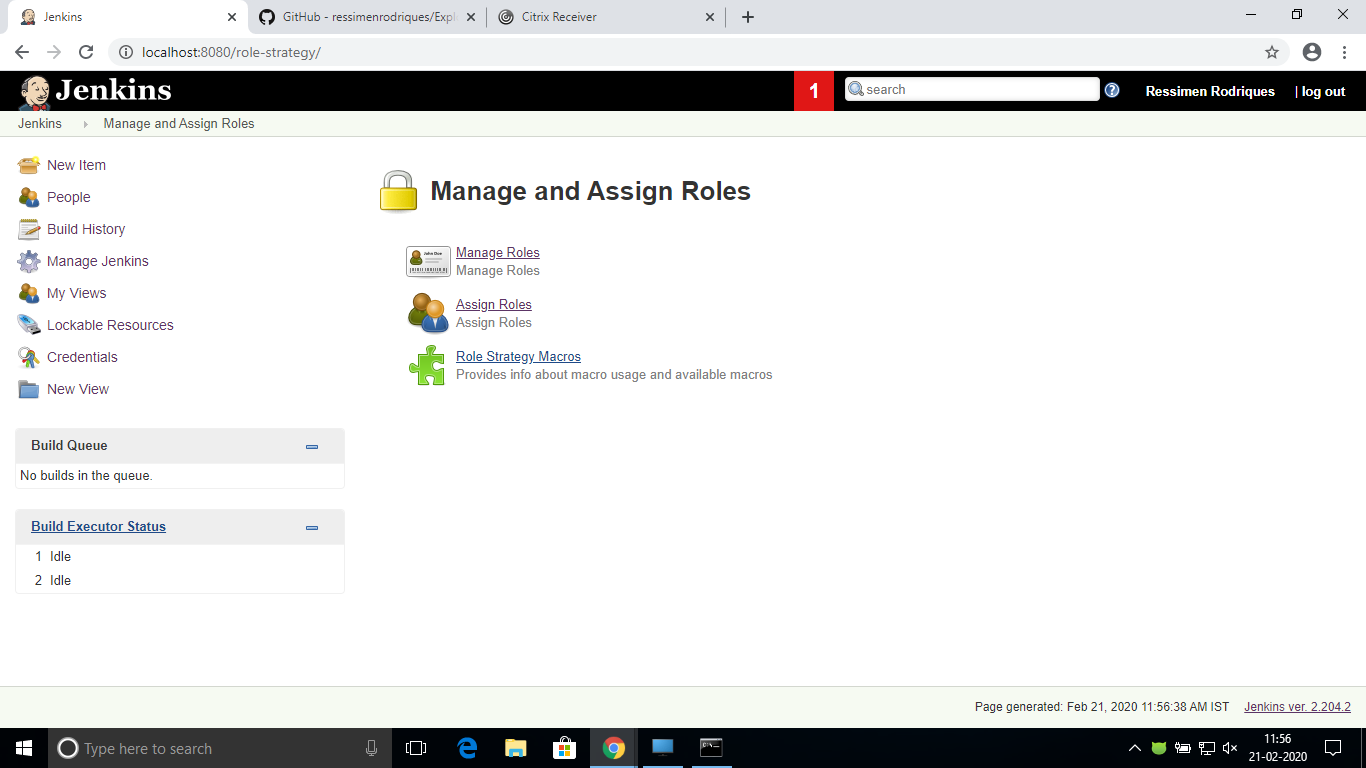
Go to Manage Jenkins -> Configure Global Security -> Under Authorization, select Role Based Strategy. Click on Save.

****

Go to manage and assign roles after installing the strategy plugin.

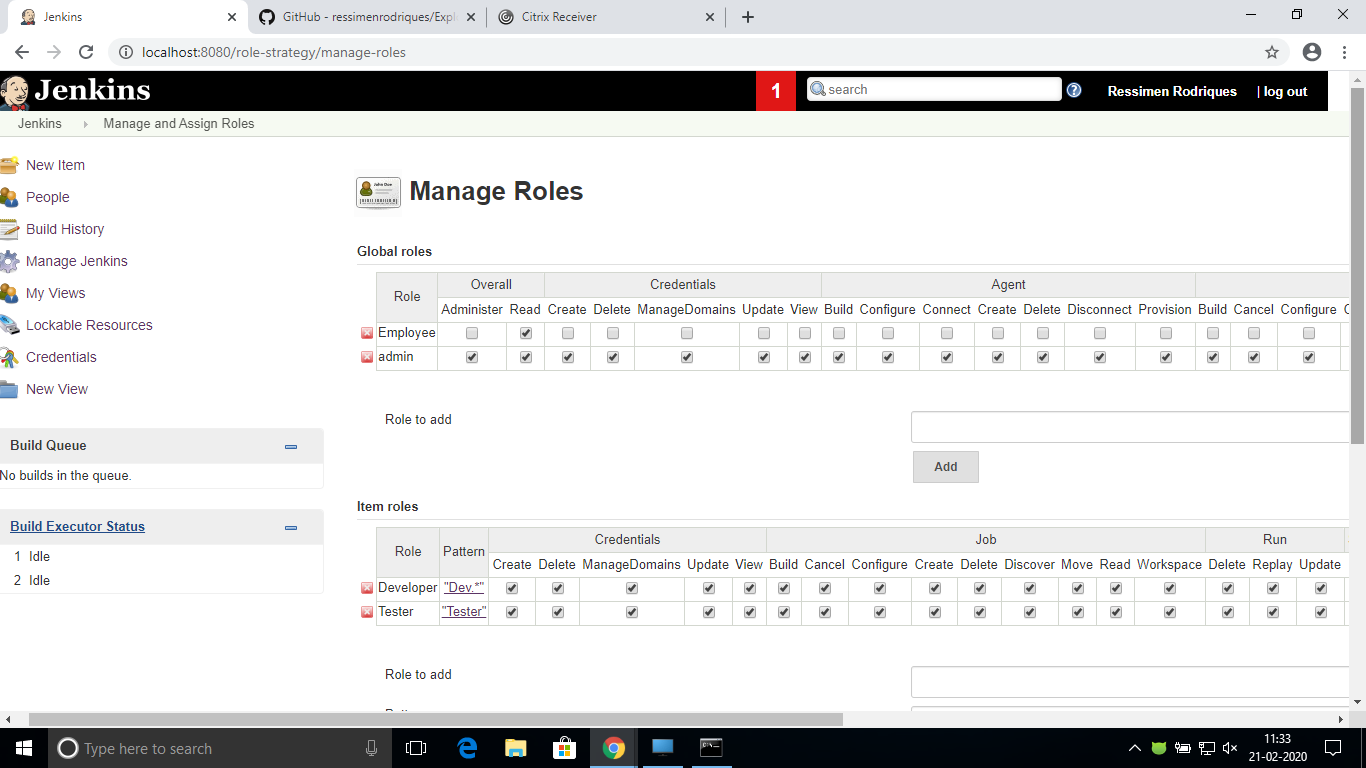
****

Click on Manage Roles to add new roles based on your organization.

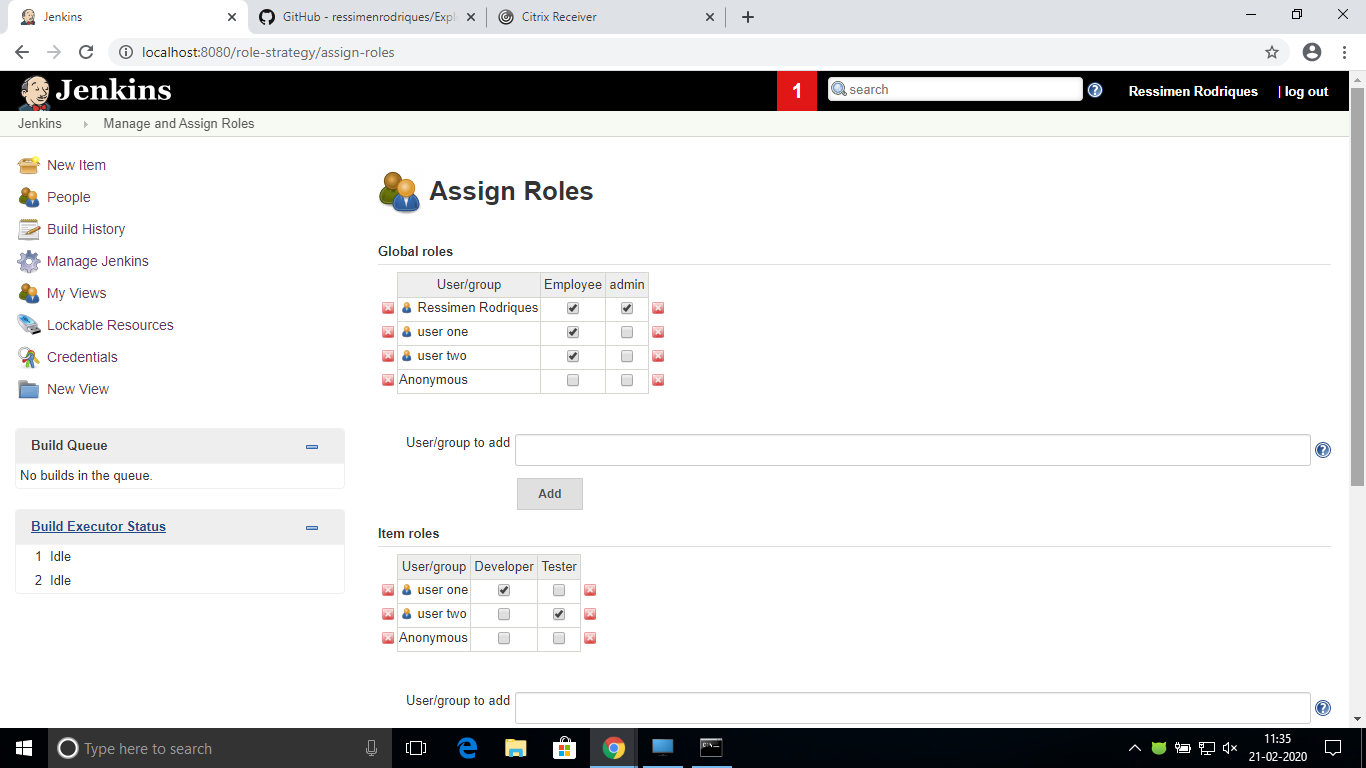
****

To create a new role called "developer”, type "developer" under "role".

Click on "Add" to create a new role. Now, select the permissions you want to assign to the "Developer" role and click save

****

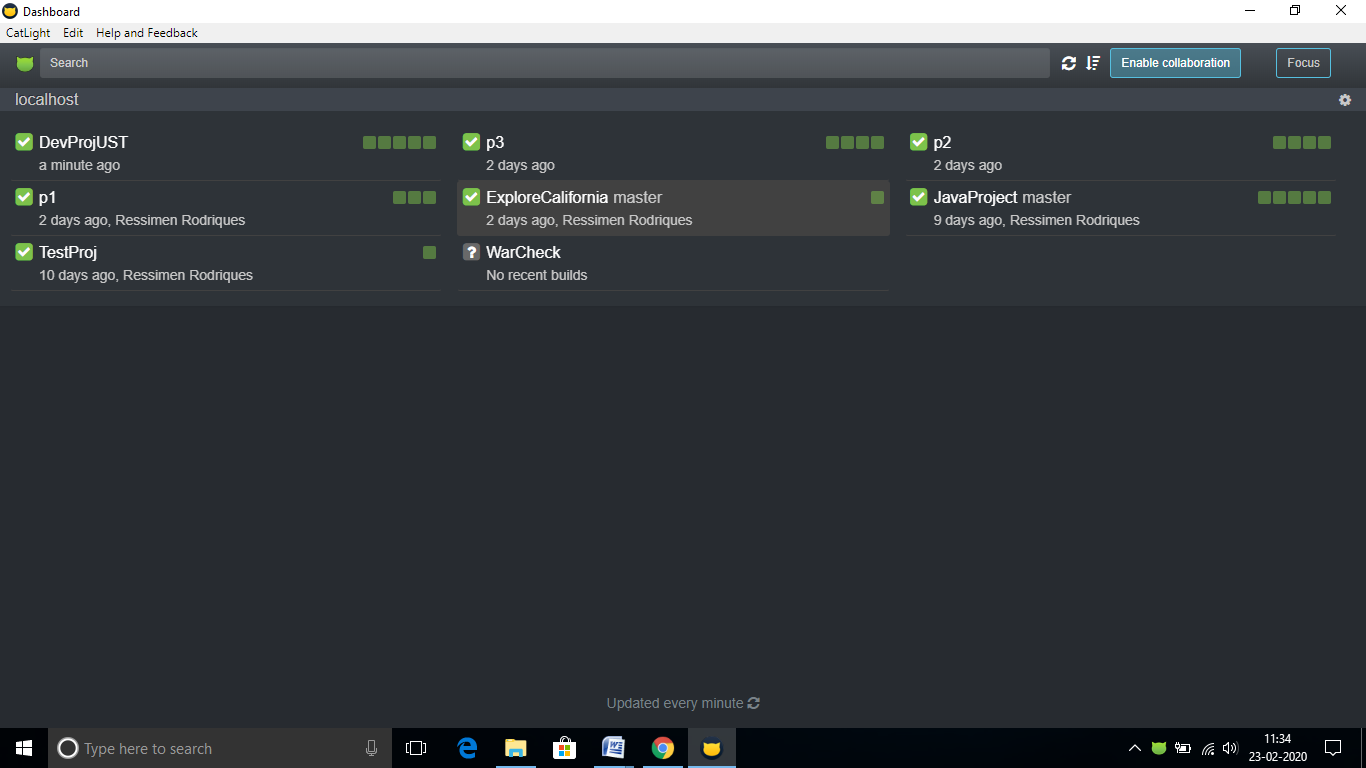
After creating roles, assign them to specific users. Go to Manage Jenkins🡪 Manage and Assign Roles🡪Assign Roles

****

1. **CatLight Introduction:**

CatLight replaces the large volume of emails that no-one reads about test or build failures. CatLight checks the continous delivery server, bug and task manager.

Catlight sends notification when a build breaks down, new important bug appears, or list of tasks changes. Catlight dashboard shows the build history and highlights the latest changes.

****

1. **Continuous Deployment:**

Continuous deployment is a strategy for software releases wherein any code commit that passes the automated testing phase is automatically released into the production environment, making changes that are visible to the software's users.

Continuous deployment pipelines use similar tools to those in continuous delivery, with an enhanced emphasis on code testing prior to and after deployment into production.

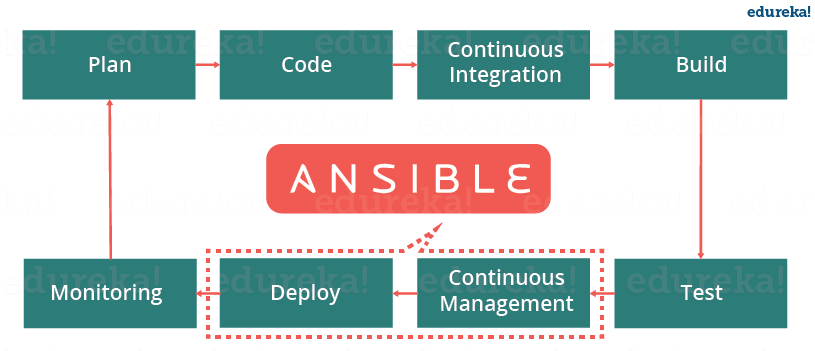
During development, version control and build automation, along with specialized tools, such as the project management software Apache Maven, ensure smooth delivery of code using continuous integration pipeline software, such as Jenkins.

Unit tests and functional tests put code into as many execution scenarios as possible to predict its behavior in production. Unit testing frameworks include NUnit, TestNG and RSpec, among many others.

For continuous deployment, IT automation and configuration management tools, such as Puppet and Ansible, handle code deployment and hosting resource configuration. Integration and acceptance tests can be set up in tools such as Cucumber and Calabash.

1. **Ansible:**

Ansible is a universal language, unraveling the mystery of how work gets done. Turns tough tasks into repeatable playbooks. It is well known that in DevOps development and operations work is integrated. This integration is very important for modern test-driven applications. Hence, Ansible integrates this by providing a stable environment to both development and operations resulting in a smooth delivery pipeline.



When developers begin to think of infrastructure as part of their application i.e as Infrastructure as code (IaC), stability and performance become normative. Infrastructure as Code is the process of managing and provisioning computing infrastructure and their configuration through machine-processable definition files, rather than physical hardware configuration or the use of interactive configuration tools. This is where Ansible automation plays a major role and stands out among its peers.

In a Continuous Delivery pipeline, Sysadmins work tightly with developers, development velocity is improved, and more time is spent doing activities like performance tuning, experimenting, and getting things done, and less time is spent fixing problems.

