Assignment 2_1 - **Jenkins**

Installing Blue Ocean:

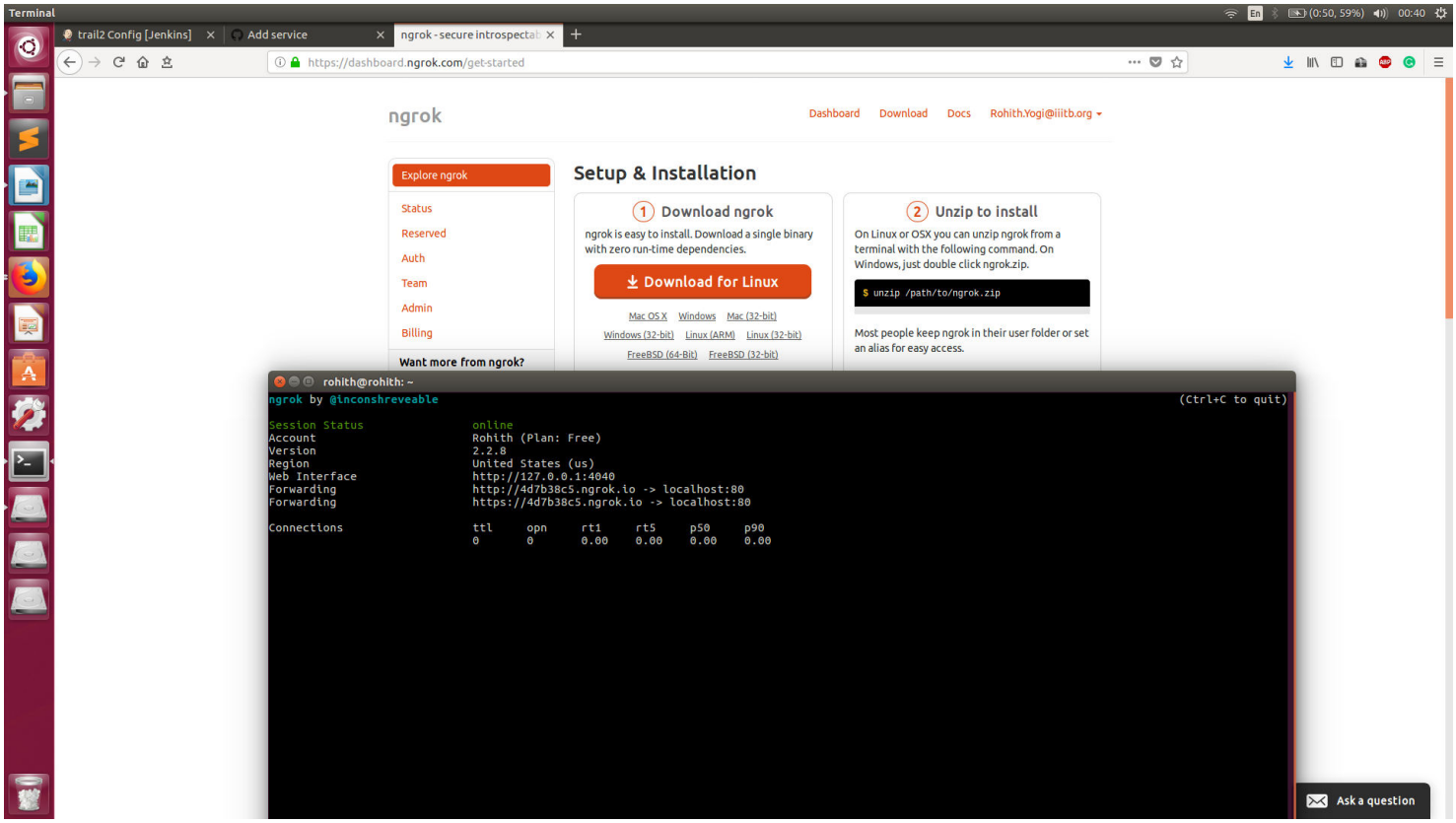
Steps:

i. Go to `Manage Jenkins > Manage Plugins.`
ii. Click on `Available` tab and search for `blue ocean` in the filter.
iii. Select the Blue Ocean `Blue Ocean Aggregator` plugin and install it.
iv. Allow Jenkins to restart.
v. Click on `Open Blue Ocean` option from the left menu to use the Blue Ocean UI.

Installing Ngrok:

Steps:

i. Go to **`https://ngrok.com/download`**
ii. Follow the series of steps given.
iii. To start a HTTP tunnel on port 80, run this in terminal:
```
$./ngrok http 80
```
iv. we get



Building a repository in GitHub (Private):

Steps:

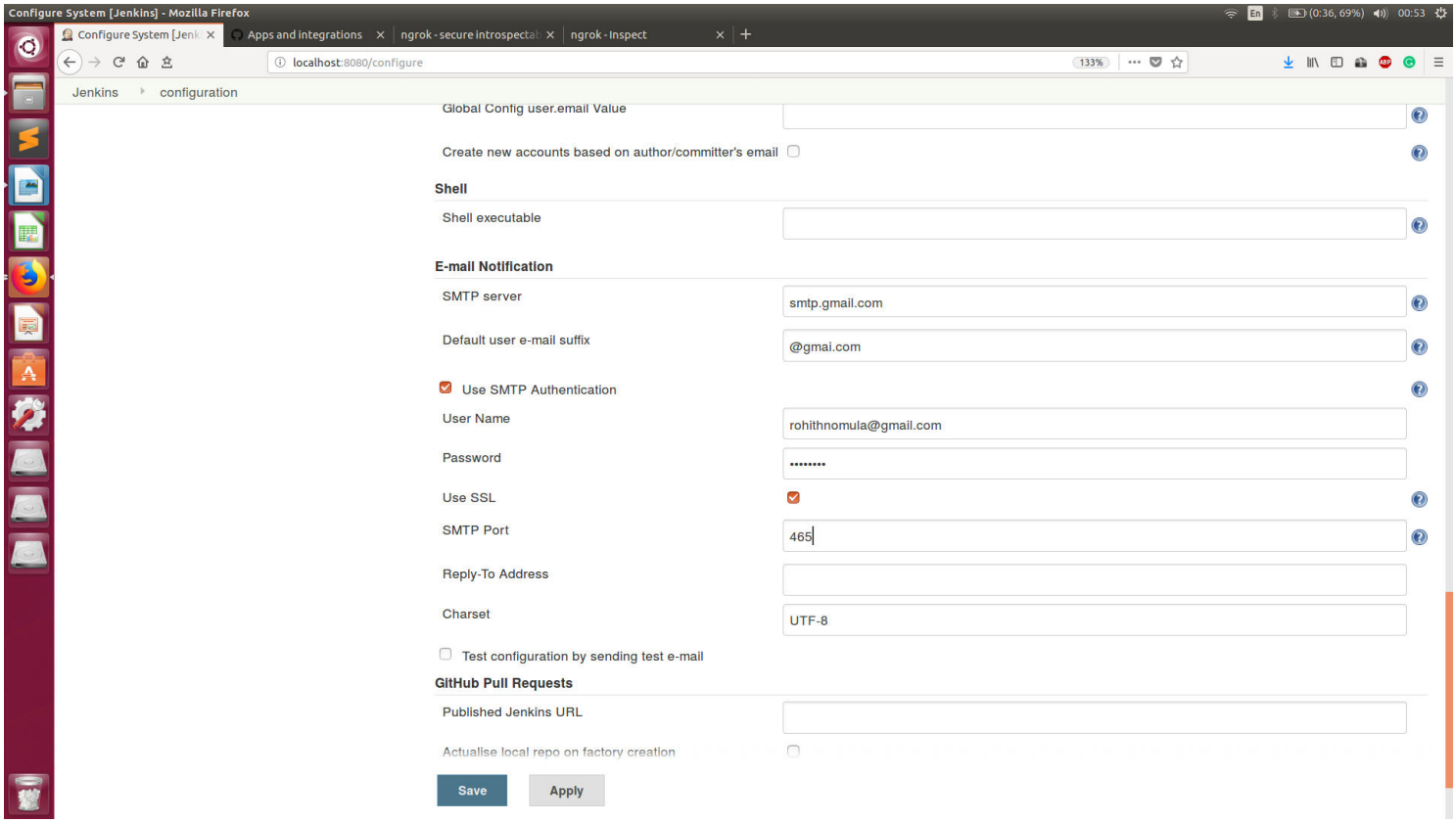
i. Create a new repository by clicking in + option on top right of GitHub home page.
ii. Under it type in name of the repository and click on private option.
iii. Click on new item in Jenkins dashboard page.
iv. Type in the project name and click on `freestyle project` and `Ok`.
v. Add the private repository's URL in the `Repository URL` text box under the GitHub project.
vi. Since the repository is private, Jenkins will not be able to access the repo without credentials. Click on `Add` button next to `Credential` text box.
vii. Choose `Jenkins` as the Credential Provider in the dropdown menu. A popup should appear asking for the details.
viii. Choose `Username with Password` under the `Kind` option.
ix. Enter GitHub username and password of the GitHub account which has the above private repository under their respective text boxes.

x. After adding your credentials choose the newly added credentials in the dropdown menu that appears next to `Credentials` label.
xi. Now the private repository can be used with Jenkins.

Using Git SCM Poll:

Steps:

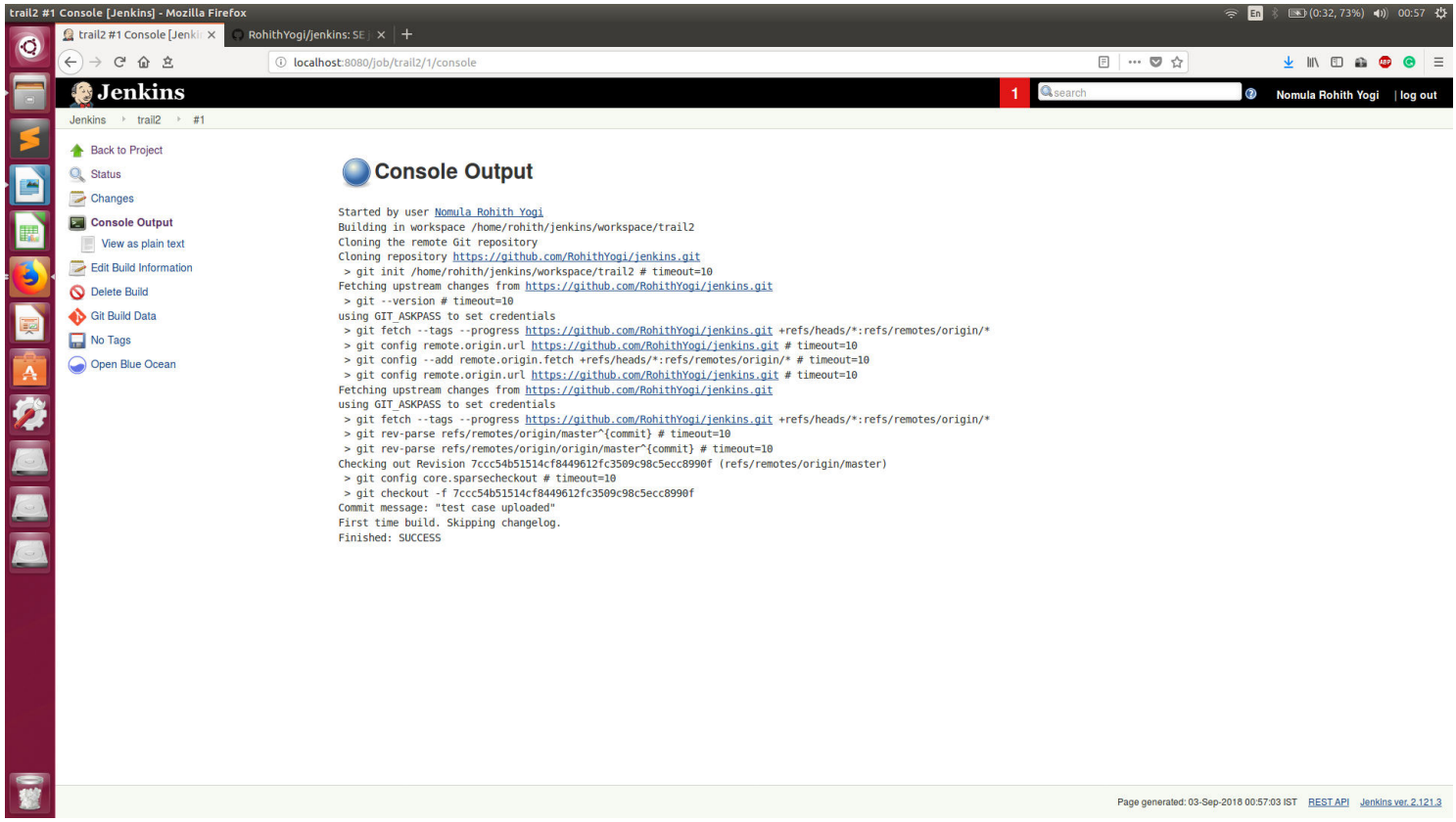
i. Go to `Configuration` page of project.
ii. Under Build Triggers in the Jenkins Configuration for the project, select GitHub hook trigger for GITScm polling.
iii. Go to GitHub private repository.
iv. Click on the Settings tab and select Integrations & Services.
v. Click on Add Service and select Jenkins (Git Plugin) from the dropdown menu.
vi. Provide the password for GitHub.
vii. Add `http://<public-ip or URL>/github-webhook` under `Jenkins hook url`. `Like given below`
viii. Select Active and click on Add Service.
ix. This will make sure that GitHub can make requests to Jenkins to build if there is a requirement for it.



Post Build Actions (e-mail Notification):

Steps:

i. Go to Jenkins Dashboard.
ii. Click on `Manage Jenkins > Configure Systems` and scroll down to `E-mail Notification` section.
iii. Click on the `Advanced` button to configure the mail account that will be used to send the mails.
iv. Select `Use SMTP Authentication` if required.
v. Enter the Credentials.
vi. Select `Use SSL` and specify the `SMTP port`.
vii. Select `Allow sending to unregistered users`.
viii. Go to `Configuration` page of project and scroll down to `Post-build Actions`.
ix. Click on `Add post-build action` and `choose Editable Email Notification` from dropdown menu.
x. Post build action for extended email notification is now complete.
xi. If we commit any changes in GitHub we get this upon building in jenkins


trail2 #1 Console [Jenkins] - Mozilla Firefox
trail2 #1 Console [Jenki
RohithYogi/jenkins: SE
localhost:8080/job/trail2/1/console
Jenkins
search
Nomula Rohith Yogi
| log out
Jenkins
trail2
#1
Back to Project
Status
Changes
Console Output
View as plain text
Edit Build Information
Delete Build
Git Build Data
No Tags
Open Blue Ocean
Console Output
Started by user Nomula Rohith Yogi
Building in workspace /home/rohith/jenkins/workspace/trail2
Cloning the remote Git repository
Cloning repository https://github.com/RohithYogi/jenkins.git
> git init /home/rohith/jenkins/workspace/trail2 # timeout=10
Fetching upstream changes from https://github.com/RohithYogi/jenkins.git
> git --version # timeout=10
using GIT_ASKPASS to set credentials
> git fetch --tags --progress https://github.com/RohithYogi/jenkins.git +refs/heads/*:refs/remotes/origin/*
> git config remote.origin.url https://github.com/RohithYogi/jenkins.git # timeout=10
> git config --add remote.origin.fetch +refs/heads/*:refs/remotes/origin/* # timeout=10
> git config remote.origin.url https://github.com/RohithYogi/jenkins.git # timeout=10
Fetching upstream changes from https://github.com/RohithYogi/jenkins.git
using GIT_ASKPASS to set credentials
> git fetch --tags --progress https://github.com/RohithYogi/jenkins.git +refs/heads/*:refs/remotes/origin/*
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
> git rev-parse refs/remotes/origin/origin/master^{commit} # timeout=10
Checking out Revision 7ccc54b51514cf8449612fc3509c98c5ecc8990f (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 7ccc54b51514cf8449612fc3509c98c5ecc8990f
Commit message: "test case uploaded"
First time build. Skipping changelog.
Finished: SUCCESS
Page generated: 03-Sep-2018 00:57:03 IST
REST API
Jenkins ver. 2.121.3