IMT2016129
Ananth Shreekumar

Assignment 2 – Setup Jenkins For GitHub Private Repository

Installing ngrok:

- **ngrok** allows you to expose a web server running on your local machine to the Internet.
- Go to ***https://ngrok.com/download*** and follow the steps mentioned to expose the localhost that Jenkins is using to the Internet. This enables GitHub to communicate with Jenkins in case of any changes to the repository.
- Here, I started a HTTP tunnel on port 8080, which Jenkins is using.

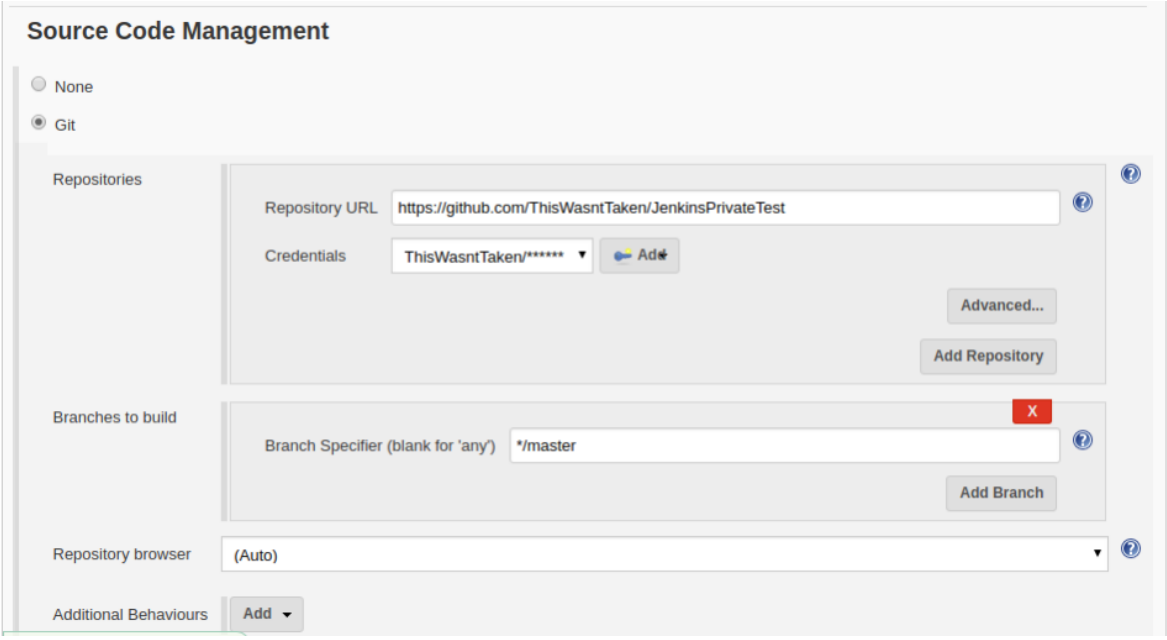
```
ngrok by @inconshreveable

Session Status                online
Account                       ThisWasntTaken (Plan: Free)
Version                       2.2.8
Region                        United States (us)
Web Interface                 http://127.0.0.1:4040
Forwarding                    http://c686ed05.ngrok.io -> localhost:8080
Forwarding                    https://c686ed05.ngrok.io -> localhost:8080

Connections                   ttl     opn     rt1     rt5     p50     p90
                              0       0       0.00    0.00    0.00    0.00
```

Install Blue Ocean:

- Click on **Manage Jenkins** and then on **Manage Plugins.**
- Click on the **Available** tab and search for **Blue Ocean** in the filter. It should appear here if it is not installed.
- Select the **Blue Ocean plugin**, install and restart Jenkins.
- To use the Blue Ocean UI, click on **Open Blue Ocean** on the left menu.

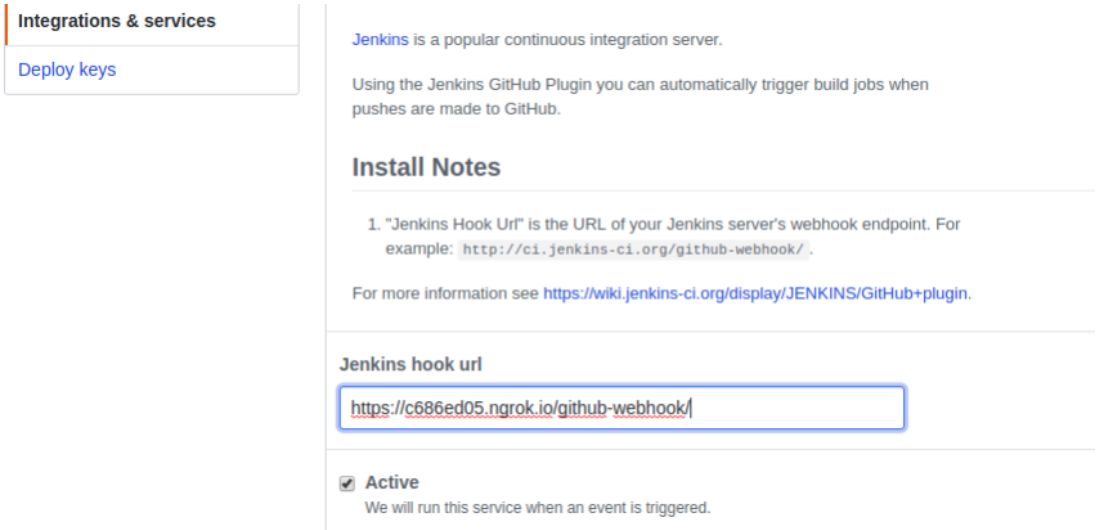
Make a private GitHub Repository:

- Click on **New Item** in the Jenkins dashboard. Enter the project name, select **Freestyle Project** and click on OK.
- Now, configure the Project as needed. Under **Source Code Management** select **Git**.
- If a private repository already exists to be used, skip this step. Otherwise, create a private repository in GitHub.
- Enter the private repository's URL in the **Repository URL** text box.
- Since the repository is private, GitHub login credentials will have to be provided. Click on **Add Credentials**, choose **Jenkins** from the dropdown menu, and provide the Username and Password of the GitHub account which has the above private repository.
- Choose the added credentials from the dropdown menu next to the **Credentials** field. Finally, it should look like the picture shown below.



Configuring the GitSCM Poll Build Trigger:

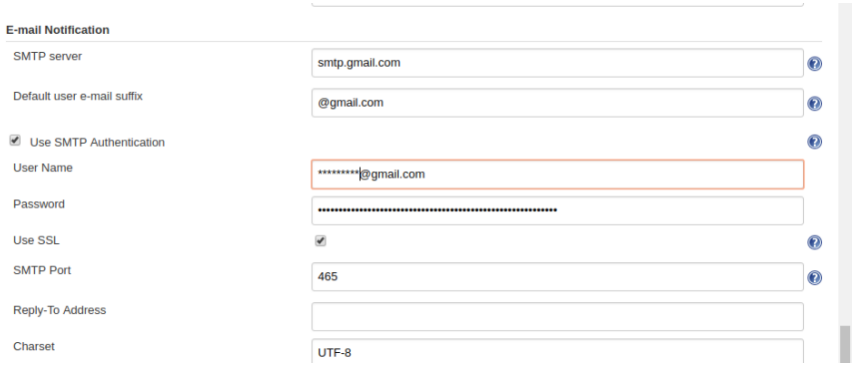
- Under **Build Triggers** in the Jenkins **Configuration** for the project, select **GitHub hook trigger for GITScm polling**.
- Go to the GitHub private repository. We will have to setup the GitSCM Polling logic here.
- Click on the Settings tab and select **Integrations & Services**.
- Click on **Add Service** and select **Jenkins (Git Plugin)** from the dropdown menu. This will prompt for the GitHub password.
- Provide the password. Now, enter the **Jenkins hook url** as shown below.



- Notice that the address provided by **ngrok** previously has been used here.
- Select **Active** to activate the service and click on Add Service.
- Now, GitHub can make requests to Jenkins to build whenever there has been a commit.

Post Build Action – Email Notifications:

- On the Jenkins dashboard, click on **Manage Jenkins** and then on **Configure System**.
- Scroll down to **E-Mail Notification** and click on **Advanced**.



- Enter the details as shown above and save. However, since gmail has a very high degree of security, you might have to allow less-secure apps to access the account in your gmail settings.
- Now, go to the project configurations. Scroll down to **Post-Build Actions**.
- Give the Email Addresses of the recipients and select **Send e-mail for every unstable build** and save.

Now, the GitHub Integration has been setup with Jenkins, and for every unstable build, an email will be sent to the recipients.