ASSIGNMENT 2_1 : PRIVATE GIT REPOSITORY WITH JENKINS

About Jenkins:

Jenkins is an open source automation server written in Java. Jenkins helps to automate the non-human part of the software development process, with continuous integration and facilitating technical aspects of continuous delivery.

Blue Ocean:

Blue Ocean is an entirely new, modern and fun way for developers to use Jenkins that has been built from the ground up to help teams of any size approach Continuous Delivery.

Installation steps:

1. Open Jenkins dashboard. Click on **Manage Jenkins.** Select **Manage Plugins** from displayed options.

2. Click on **Available** tab and search for 'Blue Ocean (BlueOcean Aggregator)' plugin.

3. Install the plugin and restart Jenkins (installation without restart can also be chosen).

4. Blue Ocean UI can be used by clicking on **Blue Ocean** option on Jenkins dashboard.

Letting Jenkins access a private GitHub repository:

1. Make a private GitHub repository.
2. Start a freestyle project in Jenkins.
3. Go to **Source Code Management** tab and select **Git**.
4. Copy private repository's URL from its GitHub page and paste it in **Repository URL** text box.
5. Jenkins requires credentials in order to access private repository. To add them, click on **Add** button which is right next to **Credentials** text box. Choose **Jenkins** in drop down menu.

6. Entering details: Choose **Username with Password** under the **Kind** option. Enter your GitHub username and password.

7. After adding your credentials choose the newly added credentials in the dropdown menu that appears next to **Credentials** label.

8. Now, private repository is ready to be used by Jenkins.

Using GIT SCM Poll

1. Open the project for which you intend to use GIT SCM Poll.

2. Go to **Configure -> Build Triggers**

3. Select **GitHub hook trigger for GITScm polling** and save configuration.

4. Install ngrok and run **./ngrok http 8080**

5. Copy **http://...ngrok.io** from terminal to the clipboard. Let us call it <copied_ngrok_url>

6. Go to GitHub repo and open **Settings**.

7. Choose **Integrations & services** from the submenu on the left.

8. Click on **Add service** and choose **Jenkins (GitHub plugin)** from dropdown menu.

9. Add **http://<copied_ngrok_url>/github-webhook** under **Jenkins hook url.**

10. Select the **Active** checkbox and click on **Add Service** button.

11. Now GitHub should make a request to the Jenkins webhook and cause a build (if required) to occur.

Post Build Actions - Extended Email Notification

Configuring email:

1. Select **Manage Jenkins** option from the menu on the left on Jenkins dashboard.

2. Select **Configure Systems** option and go to **Email Notification** section.

3. Add the SMTP server details e.g. **smtp.gmail.com** for a Gmail account.

4. Click on the **`Advanced`** button to enter details of the email account that will be used to send the emails.

5. Select **`Use SMTP Authentication`** if required e.g. Gmail uses a password for authentication and so this must be selected.

6. Enter the account email id under **`User Name`** and password under **`Password`.**

7. Select **`Use SSL`** and specify the **`SMTP port`** (465 in most cases).

8. Save the changes made.

Adding the post build action

1. Go to **Configuration-> Post-build Actions** of the project.

2. Click on **Add post-build action** and choose **Email Notification** from dropdown menu.

3. Enter the email ids of the recipients.

4. The **Advanced** settings can be used for specifying triggers i.e. conditions under which the notification should be sent.

5. Apply and save the changes made.

With this the GitHub integration has successfully been set up with Jenkins.