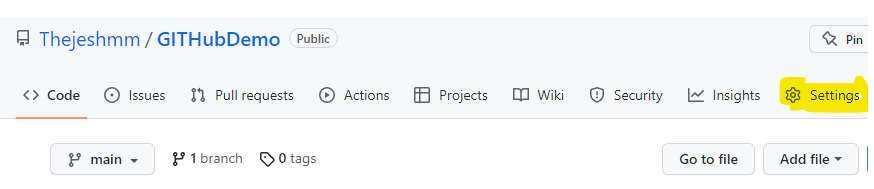
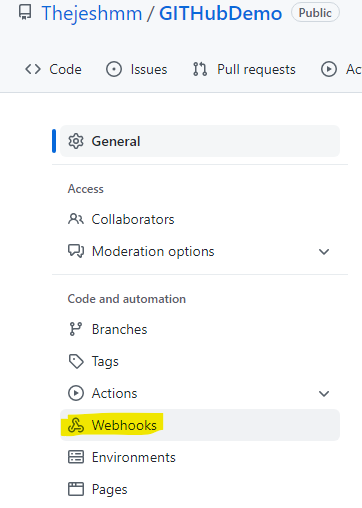
Set Up the Jenkins + GitHub Integration

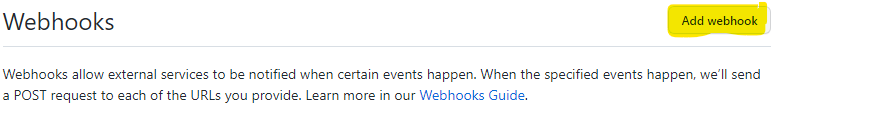
Configuring GitHub:

Step 1: Go to your GitHub repository and click on ‘Settings’

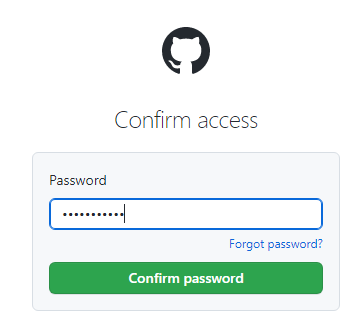


Step2: Click on Webhooks and then click on ‘Add webhook’.





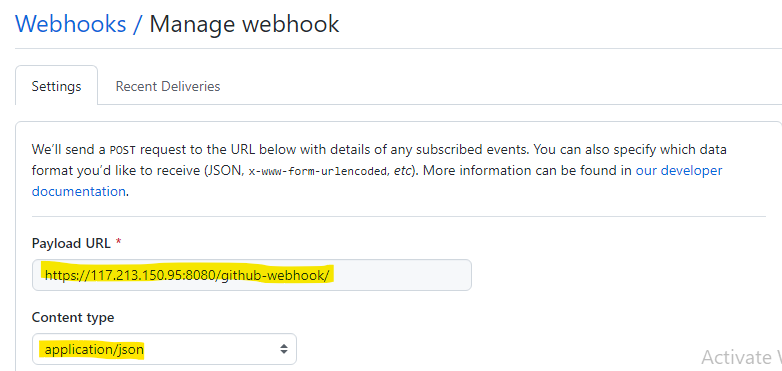
* Provide your GitHub password



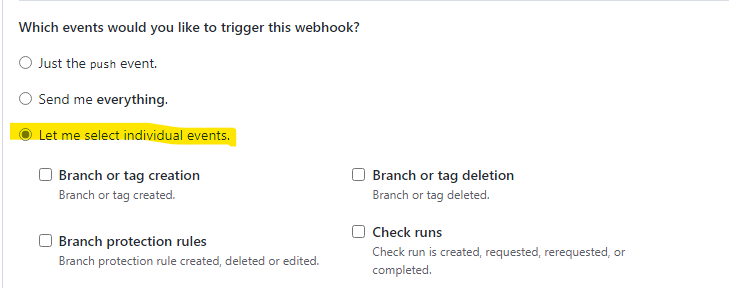
Step 3: In the ‘Payload URL’ field, paste your Jenkins environment URL. At the end of

this URL add /github-webhook/. In the ‘Content type’ select: ‘application/json’ and

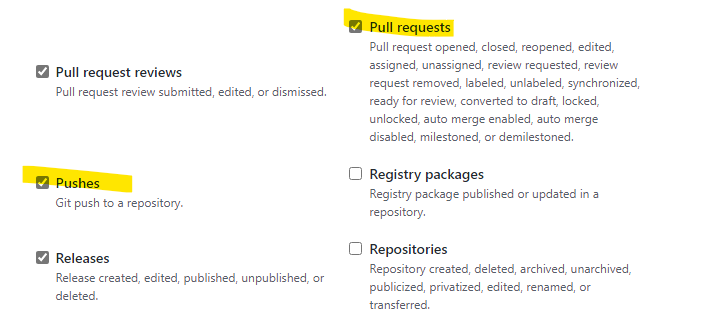
leave the ‘Secret’ field empty



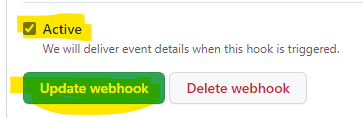
Step 4: In the page ‘Which events would you like to trigger this webhook?’ choose ‘Let me select individual events.’



* Select ‘Pull Requests’ and ‘Pushes’..

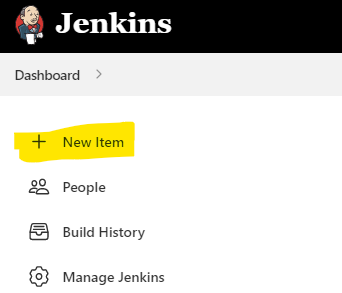


* At the end of this option, make sure that the ‘Active’ option is checked and click on ‘Add webhook’



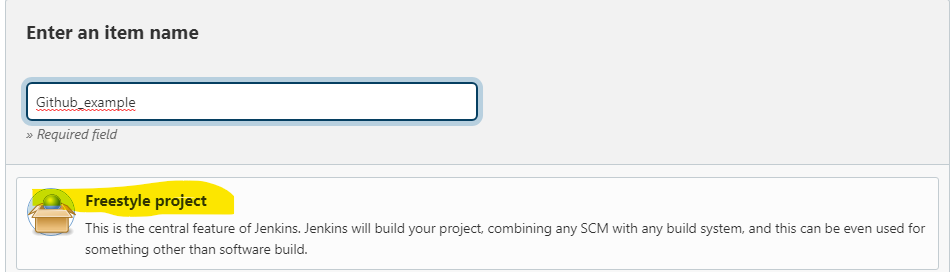
Configuring Jenkins

Step 5: In Jenkins, click on ‘New Item’ to create a new project.

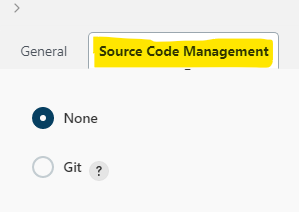


Step 6: Give your project a name, then choose ‘Freestyle project’ and finally, click

on ‘OK’.

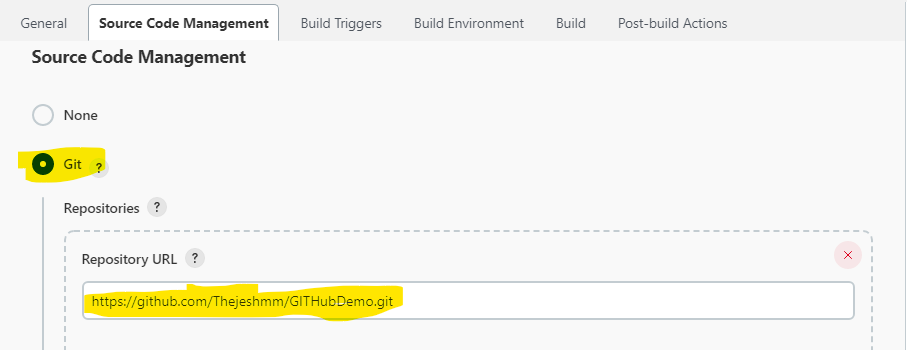


Step 7: Click on the ‘Source Code Management’ tab.



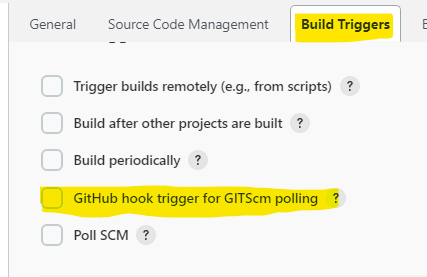
Step 8: Click on Git and paste your GitHub repository URL in the ‘Repository

URL’ field.

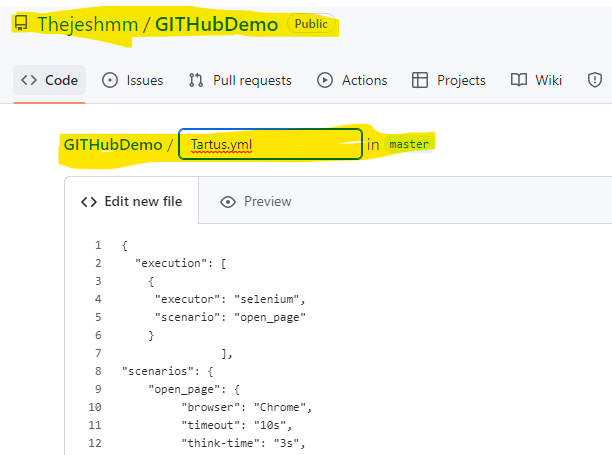


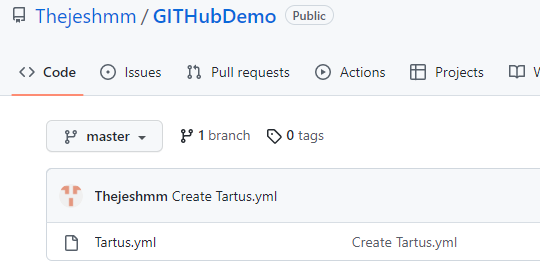
Step 9: Click on the ‘Build Triggers’ tab and then on the ‘GitHub hook trigger for

GITScm polling’. Or, choose the trigger of your choice.



* Create a code in GitHub Repo so the webhook triggers atomically

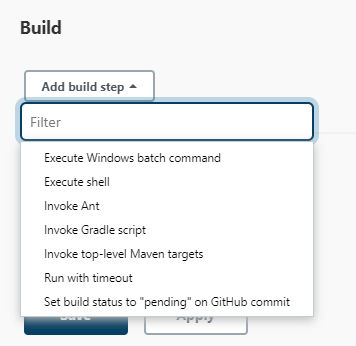




Triggering the Jenkins GitHub Integration With Every Code Commit

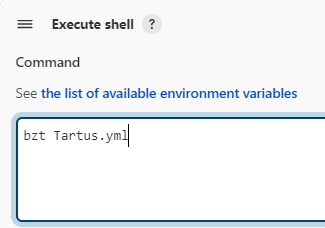
Step 10: Click on the ‘Build’ tab, then click on ‘Add build step’and choose ‘Execute

shell’.



Step 11: To run a Taurus test, simply use the ‘bzt’ command, followed by the name of

your YML file and click on ‘Save’.



Step 12: Go back to your GitHub repository, edit the Taurus script and commit the

changes. We will now see how Jenkins ran the script after the commit.

