Jenkins Integration

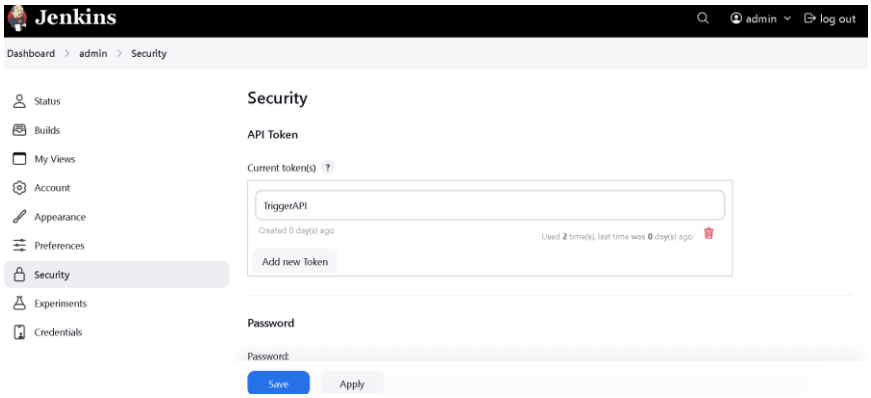
Connectors

1.Jenkins Integration and trigger build

First Go to Jenkins and create the API Token

Click your Name (top right) ➔ security ➔ Add new Token ➔ Save.

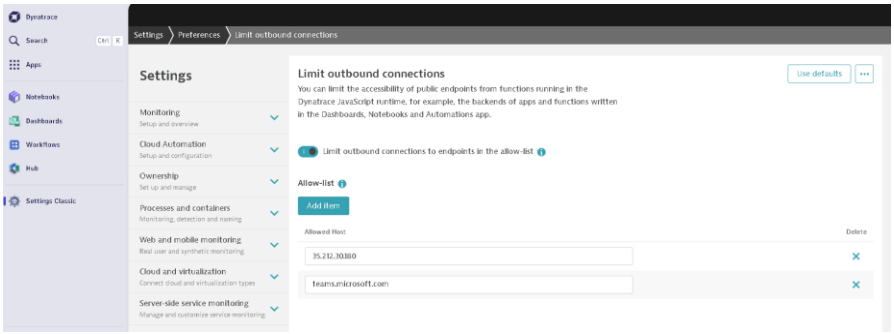
11023af9b726ca61c25a16c1c435778434



Now you have setup the API token . Let's go to Dynatrace Automation workflow.

1. **Allow Jenkins for outbound connections**

    1) Open the **Settings** app and go to **Preferences** > **Limit Outbound Connections**.
    2) Select **Add item** and add the domain of your publicly accessible Jenkins instance.
    3) Select **Save changes**.



2.Grant permissions to Workflows

Some permissions are required by Workflows to run actions on your behalf. Other permissions are required by actions that come bundled with Jenkins Connector itself.

1. Go to **Workflows** and select **Settings** > **Authorization settings**.
2. Select the following permissions besides the general Workflows permission.

- Permissions needed for Jenkins workflow actions:
    - **app-settings:objects:read**

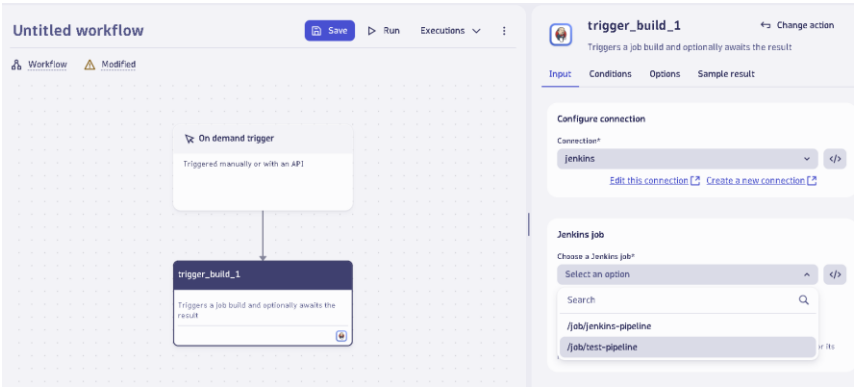
Open the Settings app and go to Dynatrace Apps > Jenkins Connections.

Jenkins instance URL : http://35.212.30.180:8080/

User* : admin

Password: 11023af9b726ca61c25a16c1c435778434 (your API token copied from jenkins side)

Go Back to Workflow to Trigger the Jenkins Build



Await result:

Only Trigger build

Click SAVE and RUN

==

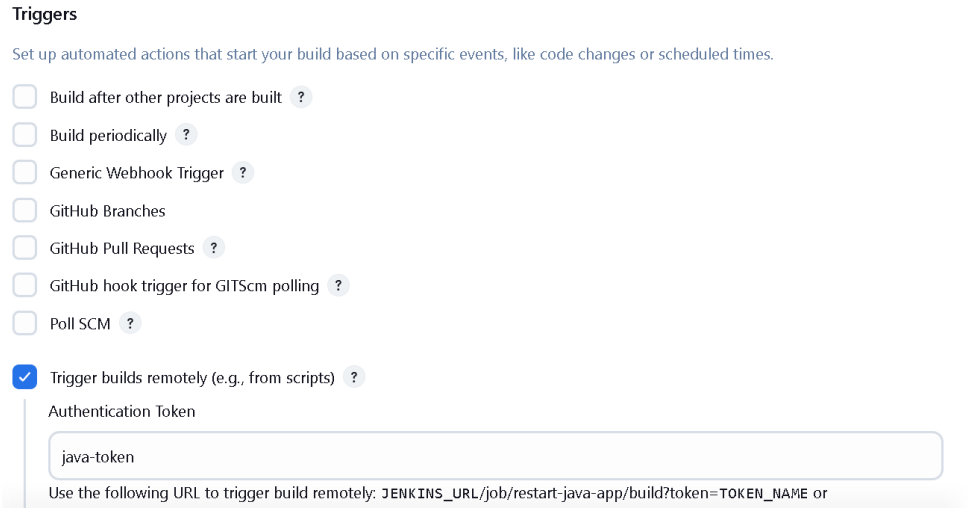
UseCase 1: Trigger the Build directly from Dynatrace

Pre-Requisites:

1. **Connect Jenkins with Dynatrace**

Open Jenkins , Create Pipeline job

This is optional : Plugin :



Pipeline:

```
pipeline {
    agent any

    environment {
        ANSIBLE_PLAYBOOK = "/home/ansible/ansible-scripts/restart_java_app.yml"
        ANSIBLE_INVENTORY = "/home/ansible/ansible-scripts/inventory.ini"
    }

    stages {
        stage('Run Ansible Playbook') {
            steps {
                script {
                    echo "Executing Ansible playbook as ansible user"

                    def ansibleCommand = "sudo -u ansible ansible-playbook -i ${ANSIBLE_INVENTORY} ${ANSIBLE_PLAYBOOK}"

                    sh ansibleCommand
                }
            }
        }
    }

    post {
        success {
            echo "Ansible playbook executed successfully!"
        }
        failure {
            echo "Ansible playbook execution failed!"
            error "Stopping pipeline due to failure"
        }
    }
}
```

restart_java_app.yml

```
---
- name: Restart Java Spring Boot Microservice
  hosts: mytargets
  become: yes
  become_user: root
  tasks:

    - name: Find if Java process is running
      shell: "jps | grep 'memoryleaksimulator-0.0.1-SNAPSHOT.jar' | awk '{print $1}'"
      register: java_pid
      changed_when: false

    - name: Kill the Java process if running
      shell: "kill -9 {{ java_pid.stdout }}"
      when: java_pid.stdout | length > 0

    - name: Start Java Spring Boot microservice
      shell: "nohup java -jar /home/sumanth_suman17/memoryleak/target/memoryleaksimulator-0.0.1-SNAPSHOT.jar 2>&1 &"
      args:
        chdir: "/home/sumanth_suman17/memoryleak/target"
      async: 10
      poll: 0

    - name: Wait for Java application to start
      pause:
        seconds: 5

    - name: Verify Java process is running
      shell: "jps | grep 'memoryleaksimulator-0.0.1-SNAPSHOT.jar'"
      register: java_status
      changed_when: false

    - name: Print success message
      debug:
        msg: "Java service restarted successfully!"
      when: java_status.stdout | length > 0

    - name: Print failure message
      debug:
        msg: "Java service failed to start!"
      when: java_status.stdout | length == 0
```

---

inventory.ini

```
[mytargets]
10.150.0.12 ansible_user=ansible ansible_ssh_private_key_file=~/.ssh/id_rsa
```

ansible-playbook restart_java_app.yml -i inventory.ini

UseCase 2: Trigger the Build from GitHub

Before we configure this from Dynatrace , first we need to make Github to connect Jenkins and push the build

https://github.com/Sumanth17-git/restart-java-ansible

1.Configure Webhook in GitHub

Go to repository→ Settings➔ Webhook

 Open GitHub Repository → Click on Settings.

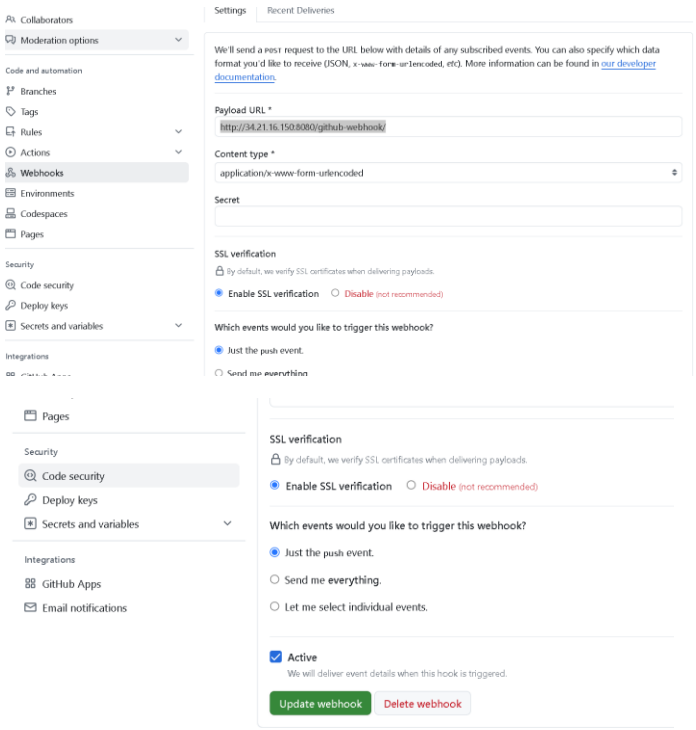
 Go to Webhooks → Click Add Webhook.

 Payload URL: Enter Jenkins Webhook URL

http://your-jenkins-server/github-webhook/

- Content Type: application/json
- Trigger Events: Select "Just the push event".
- Click "Add Webhook".

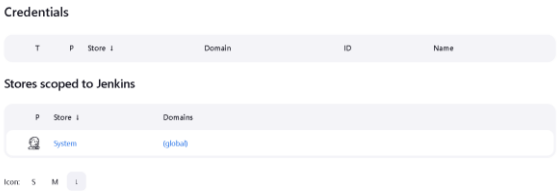
http://34.48.3.231:8080//github-webhook/ (Your Jenkins Host URL)



Step2: Setup GITHUB Connection in Jenkins

Create GIT-Hub Credentials in Jenkins

Click Manage Jenkins ➔ Credentials ➔ Click Global



Once it is setup , next step is to configure the Git Credentials on Jenkins

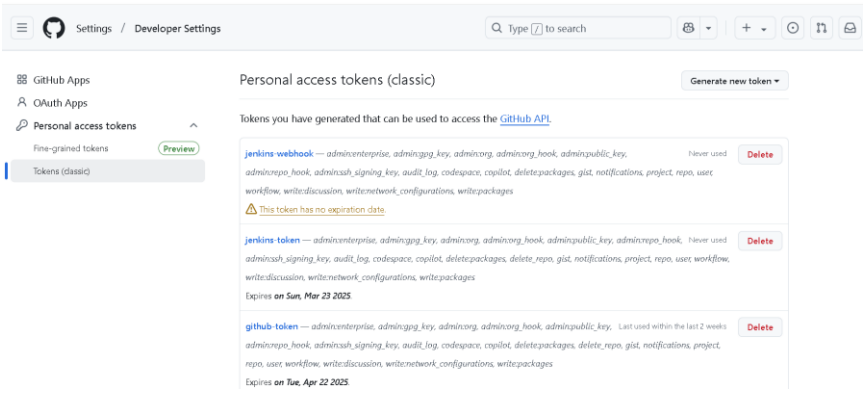
How to Add GitHub Credentials in Jenkins

To allow Jenkins to pull code from a **private GitHub repository**, you need to store your **GitHub credentials** in Jenkins securely.

Step 1: Generate a Personal Access Token (PAT) in GitHub

1. **Go to GitHub** and navigate to **Settings** → **Developer settings** → **Personal access tokens**.
2. Click **Generate new token** (or **Fine-grained tokens** for new GitHub versions).
3. Select **Expiration** (Choose "No Expiration" if you don't want it to expire).
4. Select **Scopes**:
    - ✅ repo (Full control over private repositories)
    - ✅ admin:repo_hook (For webhooks)

5. Click **Generate token**.
6. Copy the **token** and save it somewhere safe.



Step 2: Add GitHub Credentials to Jenkins

1. **Login to Jenkins** and go to **Manage Jenkins → Manage Credentials**.
2. Click **(global) > Add Credentials**.
3. Select:
    - **Kind**: **Username with password**
    - **Username**: Your GitHub username (Sumanth17-git)
    - **Password**: Paste the **GitHub PAT** token
    - **ID**: (Optional) Set an identifier like github-token
    - **Description**: "GitHub Credentials for Jenkins"

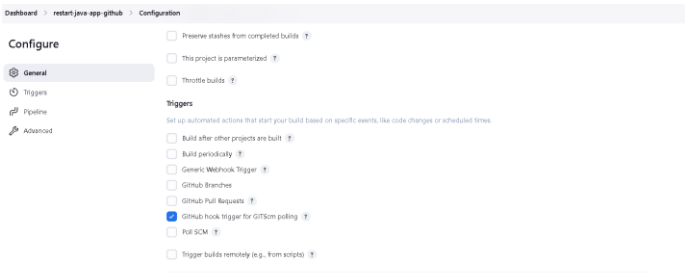
4. Click **Save**.

All set , lets update/create the Jenkinsfile in your Github Repo

credentialsId: 'github-token'

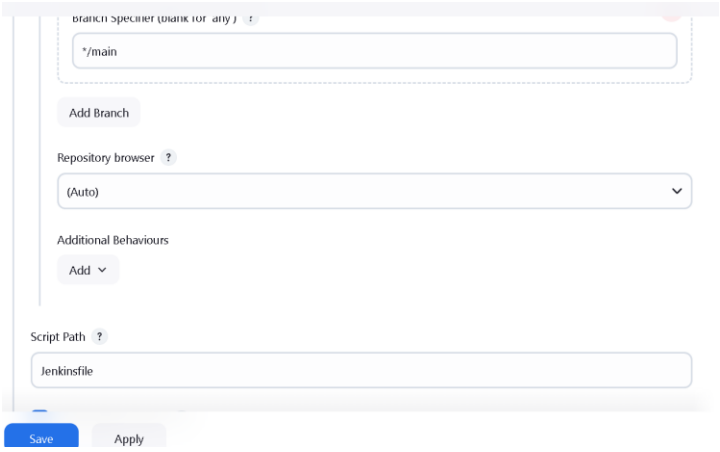
```
stages {
    stage('Checkout Code') {
        steps {
            script {
                git branch: 'main',
                    credentialsId: 'dynatrace-github',  // Use the ID you set in Jenkins
                    url: 'https://github.com/Sumanth17-git/restart-java-ansible.git'
            }
        }
```

Create the Jenkins Pipeline



Provide these details

Update the branch as **main**

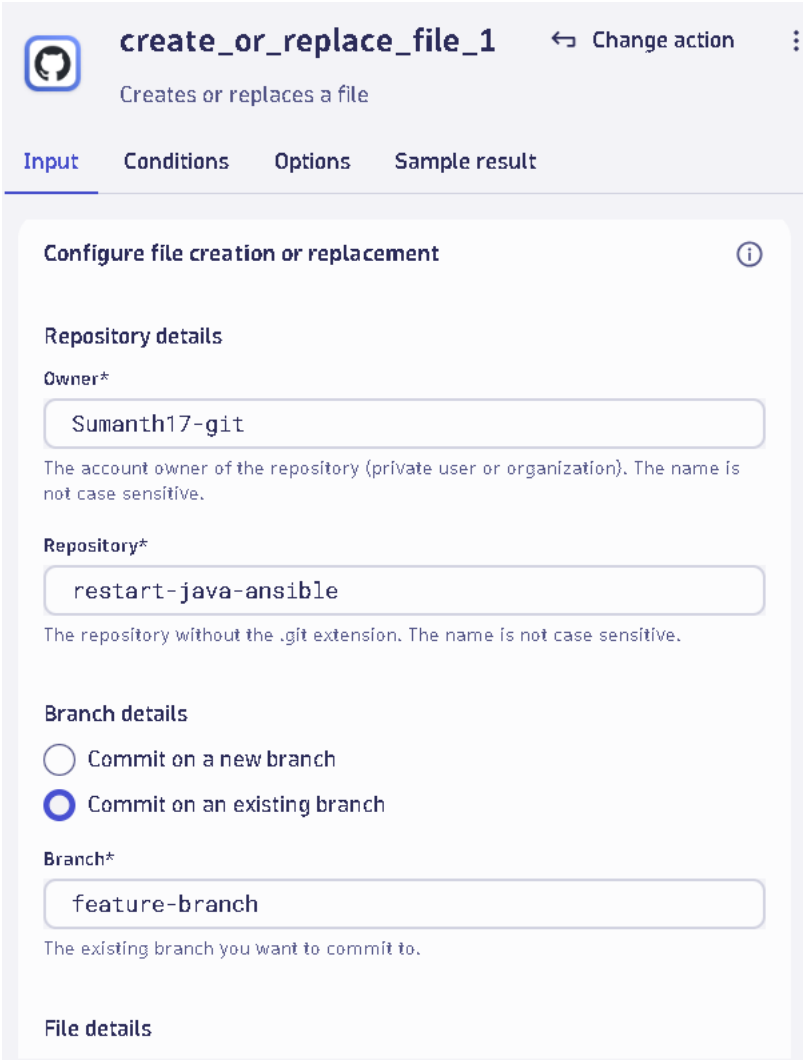


Script path : **Jenkinsfile**

Click Apply And SAVE

If everything works fine , when you make some changes in Github repo and commit to Main branch . this jenkins will get trigger automatically.

Update the GitHub Task



Update the pull request