Jenkins 3

Task 1: Automating Builds in Jenkins Using Build Triggers:

Steps:

1. Create a new freestyle Job (name = tesla-dev).
2. Copy configuration from previous Job (tesla app).
3. In your Jenkins job configuration, go to the "Build Triggers" section, Select
   a. **Build Periodically**.

      Build Periodically in Jenkins is a build trigger option that allows you to schedule the execution of your job at specified intervals. It's useful for tasks like nightly builds, database backup, system update.

      - To run this job every minute = * * * * *
      - To run this job every hour = H * * * *
      - To run this job every mid-night = 0 0 * * *
      - To run this job every 2am = 0 2 * * *

   b. **Poll SCM**.

      Poll SCM is a Jenkins feature that checks your source code repository for changes at scheduled intervals. If it finds any new commits, it triggers a build. This keeps your Jenkins job automatically in sync with your code changes, ensuring up-to-date and continuous integration.

      - To poll for changes every minute: * * * * *
      - To poll for changes every hour: H * * * *

c. **GitHub-Webhook**.

GitHub Webhook is a Jenkins feature that allows you to configure webhooks in your GitHub repository, triggering Jenkins builds automatically whenever there’s a change in the code.

- Navigate to your GitHub repository (maven-web-app).
- Go to **Settings** > **Webhooks** > **Add webhook**.
- Set the Payload URL to **http://your-jenkins-url/github-webhook/**.
- Set the Content type to **application/json**.
- Select the events that should trigger the webhook (**Just the Push event**).
- Save the webhook.

d. **Build after other projects are built**.

Build after other projects are built is a Jenkins feature that allows you to configure a job to automatically trigger a build when another specified project or projects have completed their builds.

- Create a new Freestyle project (tesla-uat).
- In your Jenkins job configuration, locate the "Build Triggers" section.
- Check the box for "Build after other projects are built."
- Enter the names of the projects in the "Projects to watch" (tesla dev).
- In build step, Add build step to execute shell = echo “Testing completed”
- Return to tesla dev and build the project.

With this setup:

Whenever tesla dev completes their builds successfully, Jenkins will automatically trigger the build of tesla-uat.

e. **Build other projects in Post-build Actions**.

Build other projects is a post-build action in Jenkins that allows you to trigger the builds of other specified projects after the current job completes its build, regardless of the build result (success, failure, unstable).

- Create new Freestyle project (tesla-prod).

  -Click configuration, source code management.

  -Select git and in the Repository URL,
https://github.com/fewaitconsulting/maven-web-app

  - Click build steps and select **Invoke top level maven.**

  -Choose Maven Version and Goal = clean deploy.
  Save.
- Open the previous **tesla-uat project** from the Dashboard.
- In your Jenkins job configuration, go to the "Post-build Actions" section.
- Click on "Add post-build action" and select "Build other projects."
- Enter the names of the projects in the "Projects to build" field (tesla-prod).
- Build tesla-dev.

With this setup:

Whenever tesla dev completes their builds successfully, Jenkins will automatically trigger the build of tesla-uat After tesla-uat completes its build, Jenkins will trigger the builds of tesla-prod as post-build actions.

tesla-dev ➔ tesla-uat ➔ tesla-prod.

Task 2: Types of Jenkins Jobs:

Maven Project.

A Maven Project in Jenkins is a type of job specifically designed for building and managing projects using Apache Maven, a popular build automation and project management tool. Maven simplifies the build process by managing dependencies, compiling source code, running tests, and packaging the application.

Here are the steps to create a Maven Project in Jenkins:

We first install maven project plugin.

-Click on Dashboard.

-Click on Manage Jenkins.

-Click on Manage Plugins, Available.

-Search and install Maven Integration.

- Go back to Dashboard.

1. **Create a New Job:**
   - Click on "New Item" on the Jenkins dashboard.
   - Enter a name for your project (paypal-app) and select "Maven Project.", click OK.
2. **Configure General Settings:**
   - Specify the project name and description.
   - Choose the appropriate project parameters like Discard Old Builds, GitHub project, etc.
3. **Source Code Management:**
   - Select your version control system (Git).
   - Provide the repository URL (https://github.com/LandmakTechnology/maven-web-application).
4. **Build:**

- Root POM = pom.xml.
- Add Maven goals to execute (clean package sonar:sonar).

5. **Post-build Actions:**
   - Select run only if build succeeds.

   First Post build action:
   - Choose the "Invoke top-level Maven targets" option.
   - Specify the Maven version to use.
   - Add Maven goals to execute (deploy).

   Add another Post build action:
   - Choose the " Deploy war/ear to a container”.
   - Add WAR/EAR files (target/*war).
   - Add Container Tomcat version (Tomcat 9.x Remote)
   - Add Credentials.
   - Add URL.

6. **Save and Build:**
   - Save your configuration and click on "Build Now" to start a build.

Task 3: Discard Old Builds.

The Discard Old Builds feature in Jenkins allows you to control the number of builds to keep for a particular job, helping manage disk space and resources. This is essential for preventing an excessive accumulation of old build artefacts and logs.

We can check the Builds/jobs in the Jenkins server as shown below

-Ls jobs/tesla-app/builds/

Here's how to configure "Discard Old Builds" in Jenkins:

1. **Navigate to Job Configuration:**

- Open the configuration page of **tesla-app**.

2. **Enable "Discard Old Builds":**

   - Check the box for "Discard Old Builds."

3. **Specify Strategy:**

   - Choose a strategy for discarding old builds. There are two common strategies:

     - **Log Rotation:**

       - Set the maximum number of days to keep builds (2).
       - Set the maximum number of builds to keep (2).

4. **Save Configuration:**

   - Save your configuration changes.

Build Job and check for results.

Task 4: Add Timestamps in Jenkins Console Output:

The Add Timestamps feature in Jenkins enhances the console output of your build by adding timestamps to each line. This is helpful for tracking the timing of each step during the build process and diagnosing issues.

Here's how you can add timestamps to the Jenkins console output:

1. **Navigate to Job Configuration:**

   - Open the configuration page of your Jenkins job.

2. **Add Timestamps:**

   - Scroll down to the "Build Environment" section.

3. **Check "Add timestamps to the Console Output":**

   - Check the box for "Add timestamps to the Console Output."

4. **Save Configuration:**

   - Save your configuration changes.