

JENKINS

JENKINS

- Self-contained, open source automation server
- Automate all sorts of tasks related to building, testing, and delivering or deploying software
- Continuous integration and continuous delivery tool
- Written in Java
- Forked from Hudson
- Platform independent
- Rich set of plugins(Over 1000 plugins)
- Easily configurable
- Easy to create new Jenkins plugin if one is not available

BENEFITS

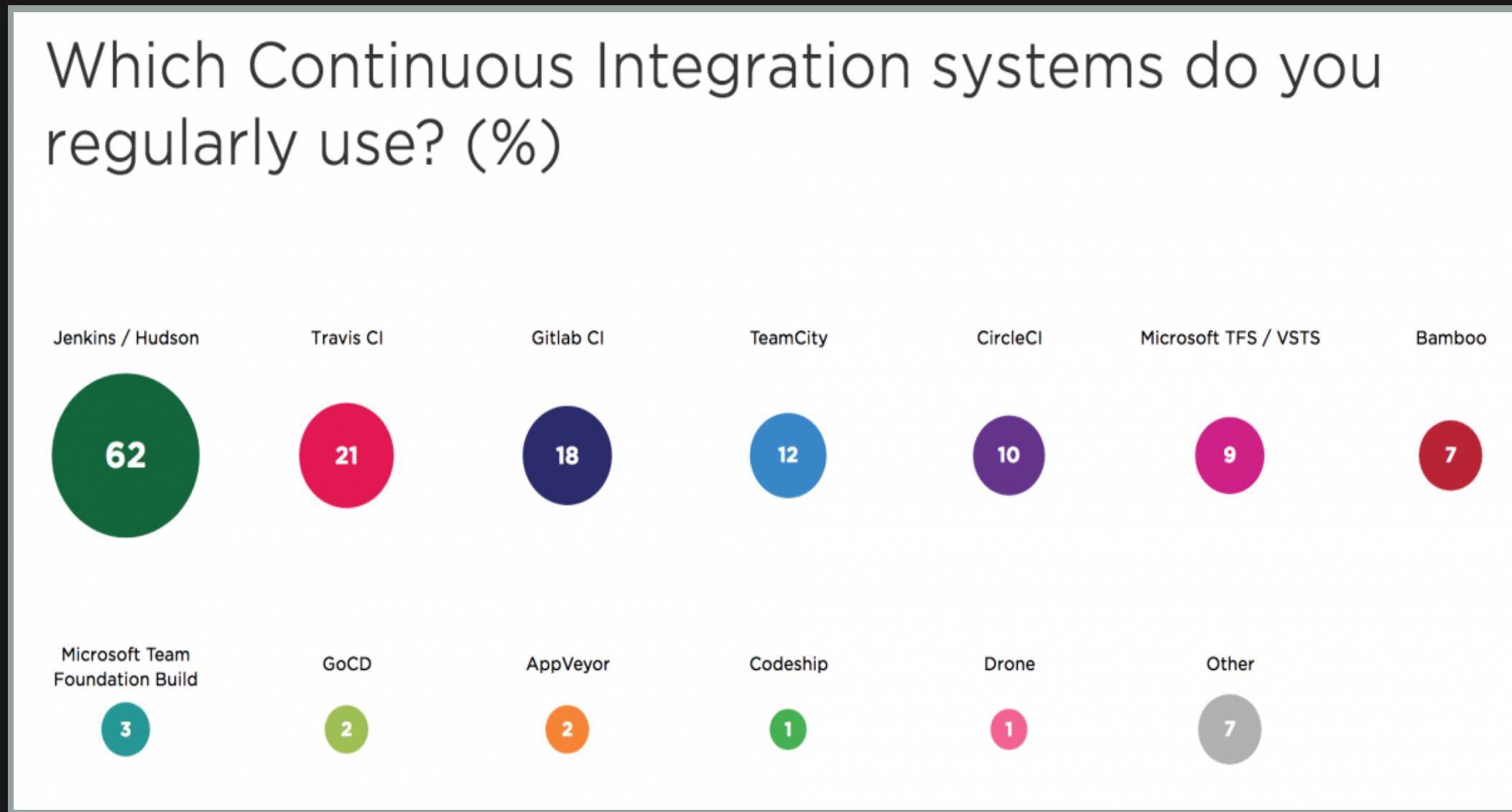
- Developer time is concentrated on work that matters making development faster
- Issues are detected and resolved almost right away which keeps the software in a state - where it can be released at any time safely.
- Complete history maintained
- Write Pipeline once, use many times
- Deployment made easy
- Improves Software development process
- Almost all sorts of plugins(Github, Gitlab, Bitbucket, etc)

JENKINS TYPICAL WORKFLOW

- Code committed in PR
- Build triggered through Webhook
- Pipeline Steps:
 - Get source code from repo
 - Build & Test the code
 - Build & Push Docker Image
 - Generate Report
 - Deploy in Dev/Stage environment(Optional)
 - Continuous Deployment (Optional)
 - Notify teams especially in Failure

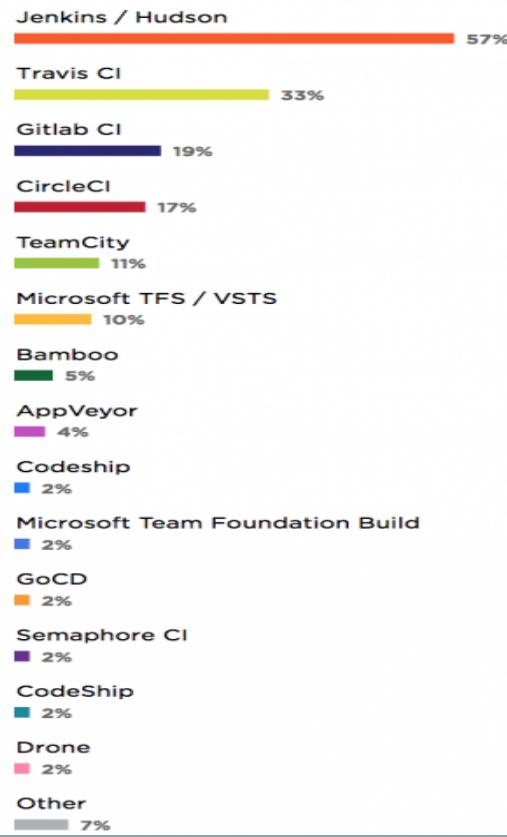
JENKINS ADOPTION

From <https://dzone.com/articles/jenkins-is-showing-the-cicd-way>

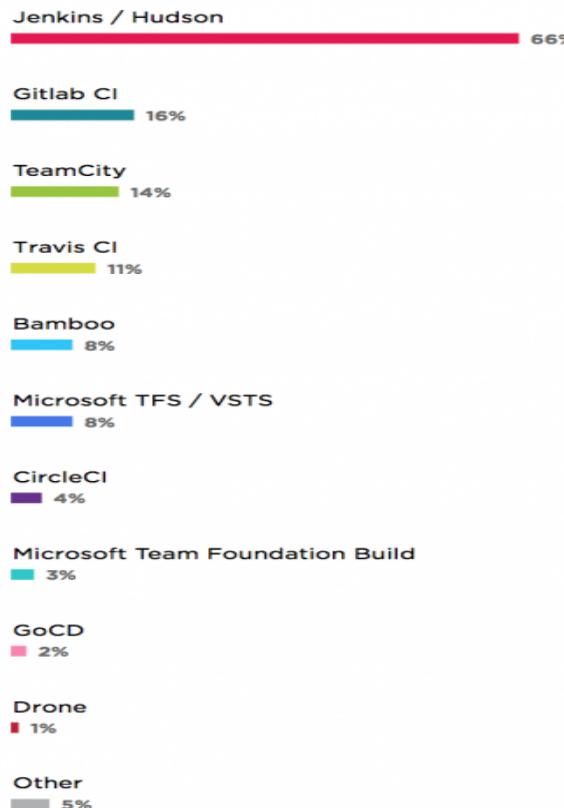


ON-PREMISE VS CLOUD ADOPTION

Which Continuous Integration (CI) system(s) do you regularly use?
Shares among people who use **in cloud** CI.

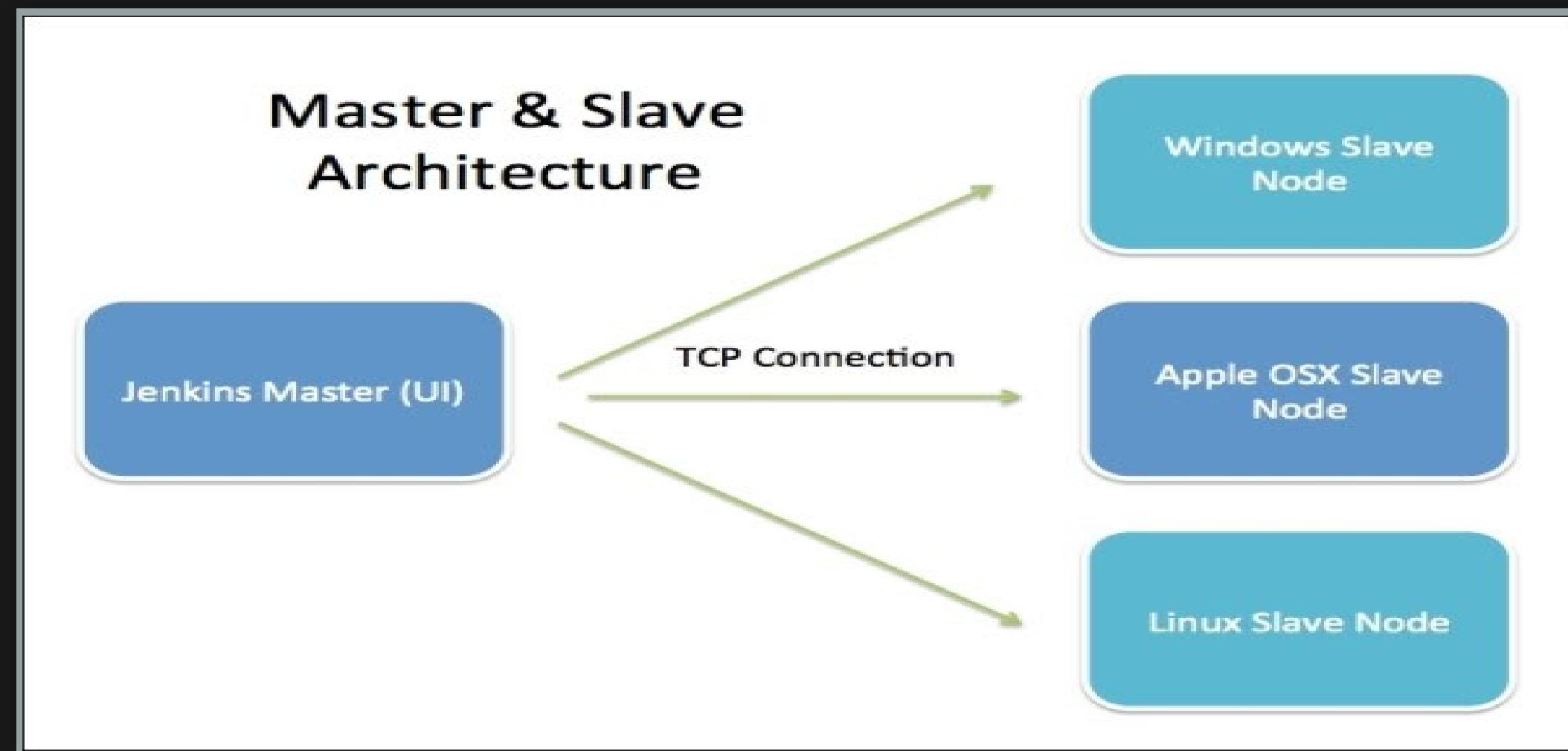


Which Continuous Integration (CI) system(s) do you regularly use?
Shares among people who use **on-premises** CI.



JENKINS ARCHITECTURE

- Master and slave architecture
- Can have single master and multiple slaves with different environment to test app



JENKINS TERMINOLOGIES

- **Job:** A unit of work
- **Master:** central & main unit, Does job scheduling
- **Slave:** Execute one or more jobs
- **Plugin:** Common workflows merged into one
- **View:** Different Jobs can be combined in a single view

LAB - INSTALL JENKINS

- Stand alone application
- Docker
- Package

STAND ALONE APPLICATION

Jenkins comes with an embedded Winstone server; therefore enabling it to run as a standalone application.

Create directory for jenkins stand alone application

```
mkdir jenkins-standalone  
cd jenkins-standalone
```

Download [Jenkins WAR file](#) to jenkins-standalone directory on your machine

To run Jenkins as stand alone application Java is required. Open terminal and check if Java is installed by run the following command

```
java -version
```

If Java is not installed, you can install Java by running the following command

```
sudo apt-get install openjdk-8-jdk -y
```

Run Jenkins

```
java -jar jenkins.war
```

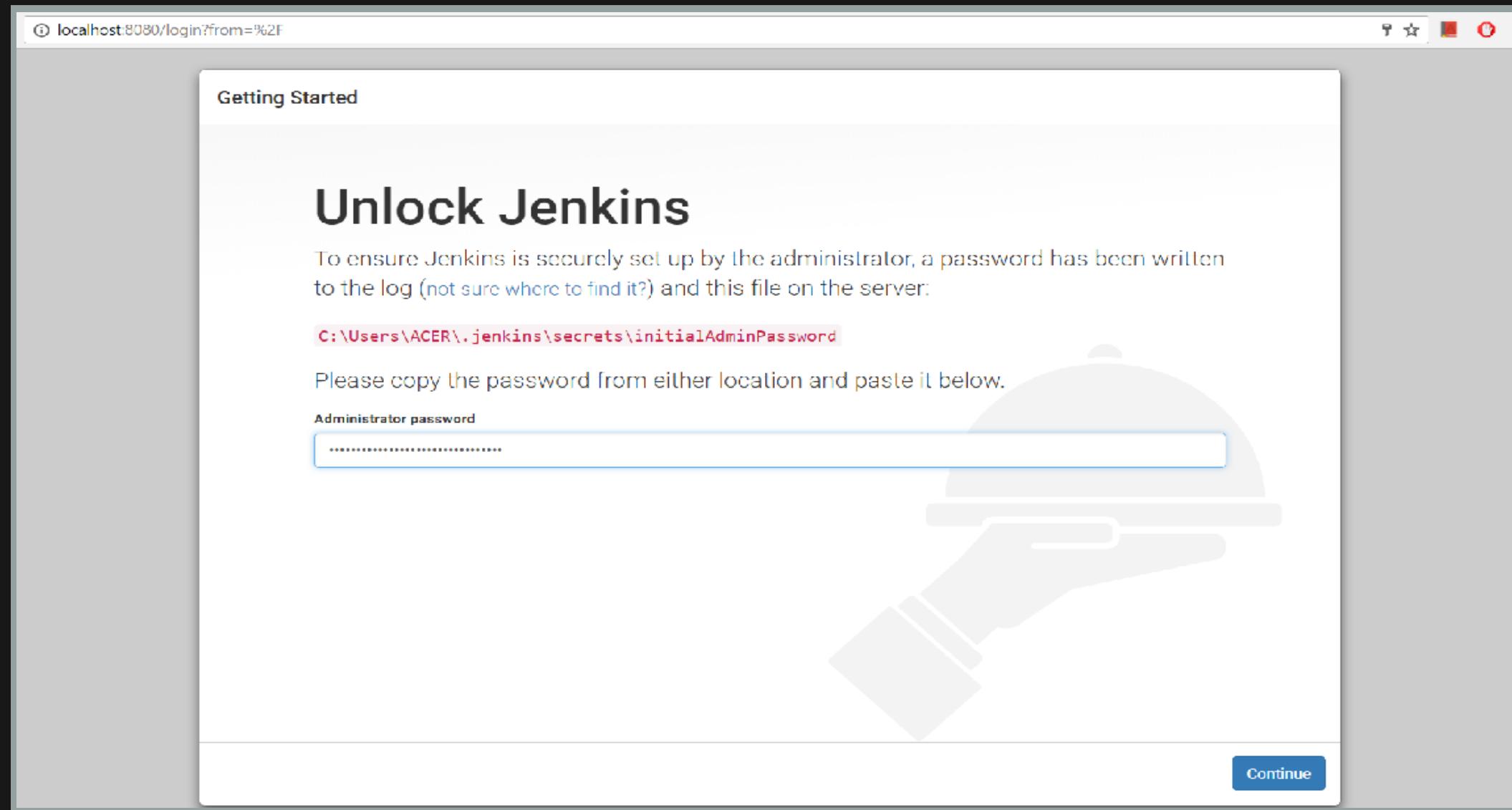
Now you should see the initial setups being run in the command line. The setup will generate a random password for you to use on the initial login. Note this one down.

```
cmd.java -jar jenkins.war

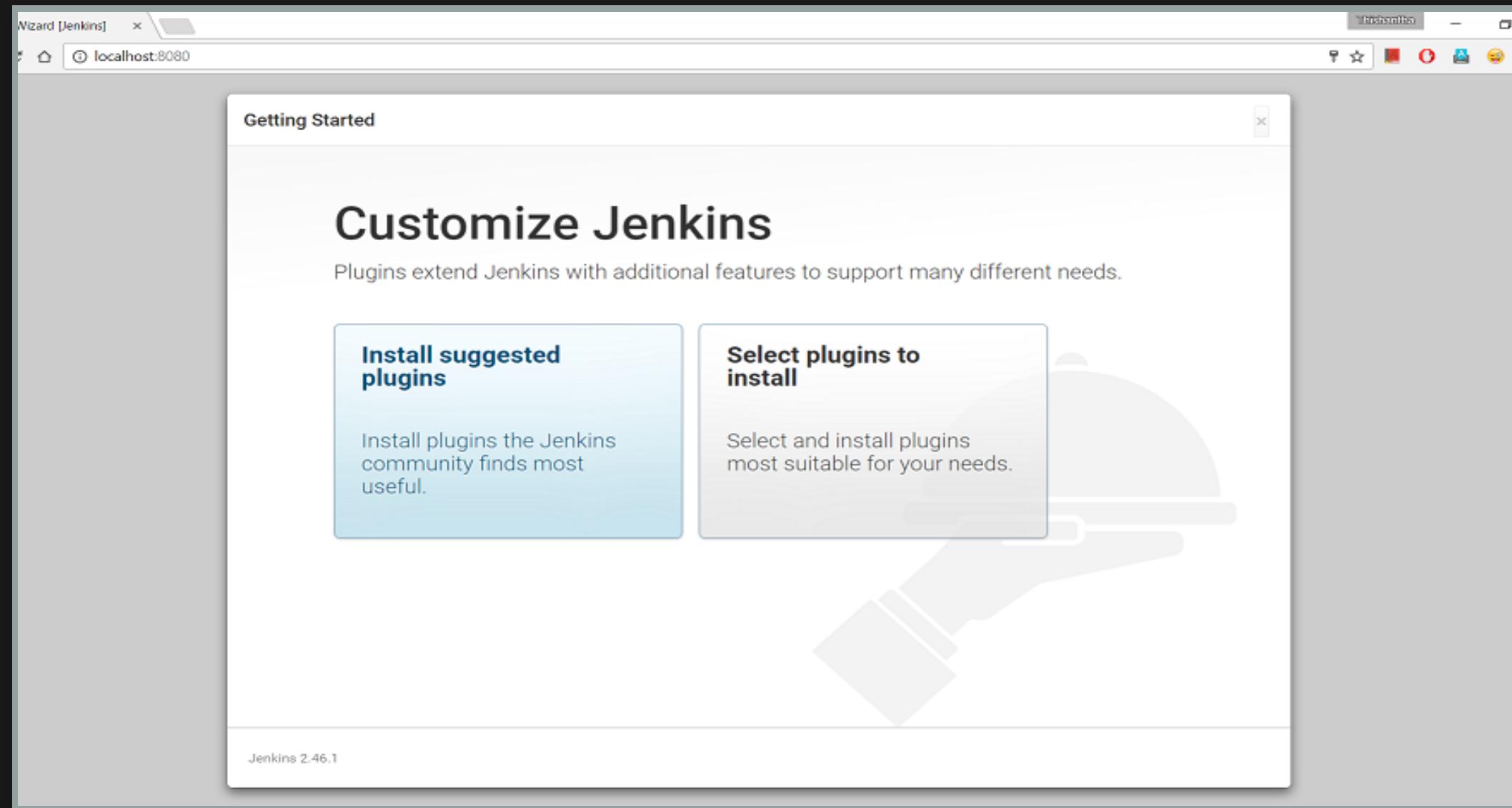
[applicationContext]; startup date [Fri Apr 14 13:48:31 IST 2017]; root of context hierarchy
Apr 14, 2017 1:48:31 PM org.springframework.context.support.AbstractApplicationContext obtainFreshBeanFactory
INFO: Bean factory for application context [org.springframework.web.context.support.StaticWebApplicationContext@3af9c257
]: org.springframework.beans.factory.support.DefaultListableBeanFactory@30483487
Apr 14, 2017 1:48:31 PM org.springframework.beans.factory.support.DefaultListableBeanFactory preInstantiateSingletons
INFO: Pre-instantiating singletons in org.springframework.beans.factory.support.DefaultListableBeanFactory@30483487: defining beans [filter,legacy]; root of factory hierarchy
Apr 14, 2017 1:48:31 PM jenkins.install.SetupWizard init
INFO:

*****
*****
***** Jenkins initial setup is required. An admin user has been created and a password generated.
Please use the following password to proceed to installation:
e415d7bd535f4ba79ac13587fdbd10549
This may also be found at: C:\Users\ACER\.jenkins\secrets\initialAdminPassword
*****
*****
***** Apr 14, 2017 1:48:37 PM hudson.model.UpdateSite updateData
INFO: Obtained the latest update center data file for UpdateSource default
Apr 14, 2017 1:48:37 PM hudson.WebAppMain$3 run
INFO: Jenkins is fully up and running
```

Wait for the console output to print 'Jenkins is fully up and running'. Now go to your browser and type the url: <http://localhost:8080/> and you should see a screen as shown below. Enter the initial administrator password that was generated in the previous step. Input the password and click 'Continue'.



In the next screen you are asked to install necessary plugins for Jenkins. This enables the integration of various platforms with Jenkins. Select 'Install selected plugins'. Wait for the plugin installation to complete



localhost:8080

Getting Started

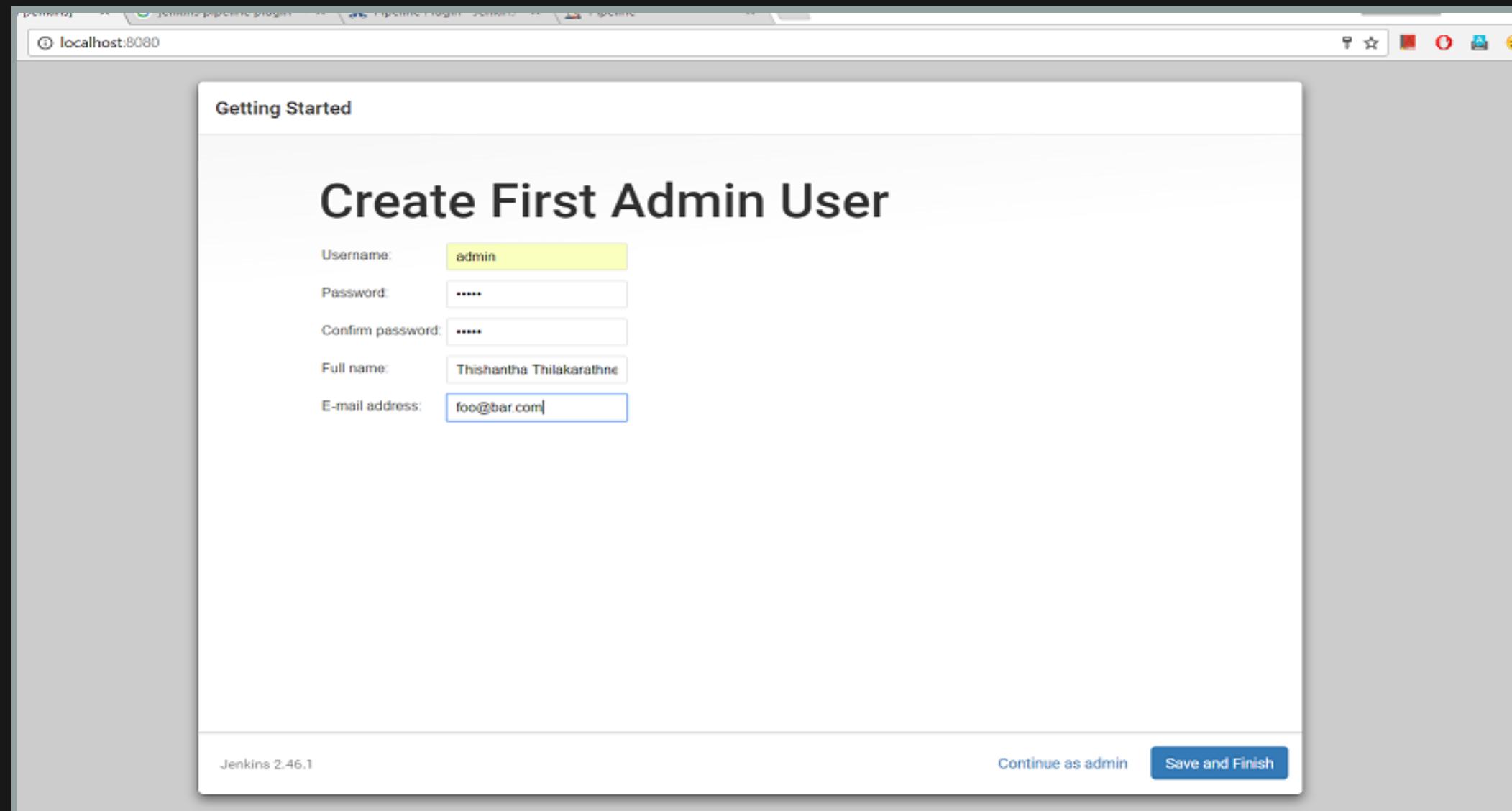
Getting Started

 Folders Plugin	 OWASP Markup Formatter Plugin	 build timeout plugin	 Credentials Binding Plugin	** bouncycastle API Plugin
 Timestamper	 Workspace Cleanup Plugin	 Ant Plugin	 Gradle Plugin	
 Pipeline	 GitHub Organization Folder Plugin	 Pipeline: Stage View Plugin	 Git plugin	
 Subversion Plug-in	 SSH Slaves plugin	 Matrix Authorization Strategy Plugin	 PAM Authentication plugin	
 LDAP Plugin	 Email Extension Plugin	 Mailer Plugin		

** - required dependency

Jenkins 2.46.1

Create an Admin user in the next screen once the plugin installation is complete.



Once finished, click Save and Finish and now proceed to the Jenkins dashboard where you can start creating jobs.



DOCKER

You can create a Jenkins container by running the following command. This will create a container named jenkins.

```
docker run -d -p 8081:8080 --name jenkins jenkins/jenkins
```

You can access Jenkins at <http://localhost:8081>. You can follow the same steps specified in Stand alone application section

PACKAGE

You will install Jenkins through apt by running the following command

```
wget -q -O - https://pkg.jenkins.io/debian/jenkins.io.key | sudo apt-key add -
sudo sh -c 'echo deb https://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list'
sudo apt-get update
sudo apt-get install jenkins -y
```

Check the status of Jenkins by running the command

```
systemctl status jenkins
```

If jenkins is not running you can start Jenkins service by running the command

```
systemctl start jenkins
```

You can access Jenkins at <http://localhost:8080>. You can follow the same steps specified in Stand alone application section.

JENKINS UI



Welcome to Jenkins!

umer

.....|

Keep me signed in

JENKINS UI

The screenshot shows the Jenkins User Interface (UI) homepage. The top navigation bar includes the Jenkins logo, a search bar, and user information (username and log out link). A "ENABLE AUTO REFRESH" button is also present. The left sidebar contains links for "New Item", "People", "Build History", "Manage Jenkins", "My Views", "Lockable Resources", "Credentials", and "New View". The main content area features a "Welcome to Jenkins!" message and a call-to-action button labeled "create new jobs". Below this, there are two expandable sections: "Build Queue" (showing "No builds in the queue.") and "Build Executor Status" (showing "1 Idle" and "2 Idle"). At the bottom, a footer displays the page generation timestamp ("Page generated: Jul 14, 2019 12:27:41 AM PKT"), links to the REST API and Jenkins version ("Jenkins ver. 2.176.1"), and a "Page refresh" button.

Jenkins

search | ? | umer | log out

ENABLE AUTO REFRESH

New Item

People

Build History

Manage Jenkins

My Views

Lockable Resources

Credentials

New View

Welcome to Jenkins!

Please [create new jobs](#) to get started.

Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

Page generated: Jul 14, 2019 12:27:41 AM PKT [REST API](#) Jenkins ver. 2.176.1

JENKINS UI

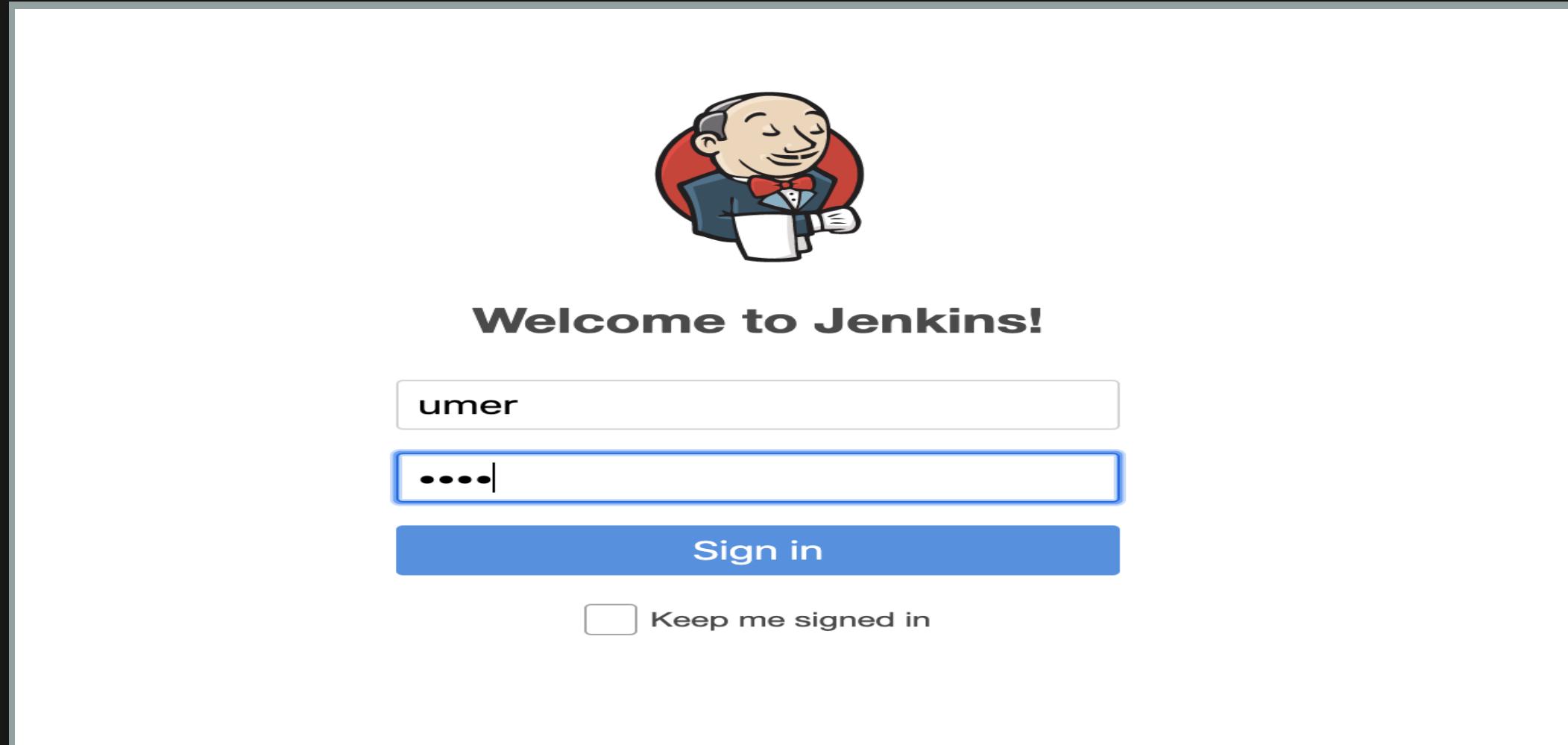
- **New Item:** To create a new job
- **People:** Manages users within Jenkins
- **Build History:** Shows history of builds
- **Manage Jenkins:** Used to configure Jenkins
- **My Views:** Private view for filtering Jenkins jobs
- **Credentials:** Manage global credentials
- **Build Queue:** Builds that have been triggered
- **Build Executor Status:** Status of running builds

LAB - FIRST JENKINS JOB

- Create First Job
- Edit Existing Job
- Run Shell Script
- Parameterized Job

CREATE FIRST JOB

You are going to create your first Jenkins job. Go to your browser and type: <http://localhost:8080/> and enter the admin credentials you created during configuration.



Once logged in, you will see the Jenkins dashboard shown below. Click New Item

The screenshot shows the Jenkins dashboard. At the top, there is a navigation bar with the Jenkins logo, a search bar, and user information (username and log out link). Below the navigation bar is a sidebar containing links: New Item, People, Build History, Manage Jenkins, My Views, Lockable Resources, Credentials, and New View. The main content area features a large "Welcome to Jenkins!" message with a sub-instruction: "Please [create new jobs](#) to get started." Below this message are two collapsed sections: "Build Queue" (showing "No builds in the queue.") and "Build Executor Status" (showing "1 Idle" and "2 Idle"). At the bottom of the page, there is a footer with the text "Page generated: Jul 14, 2019 12:27:41 AM PKT" and links to "REST API" and "Jenkins ver. 2.176.1".

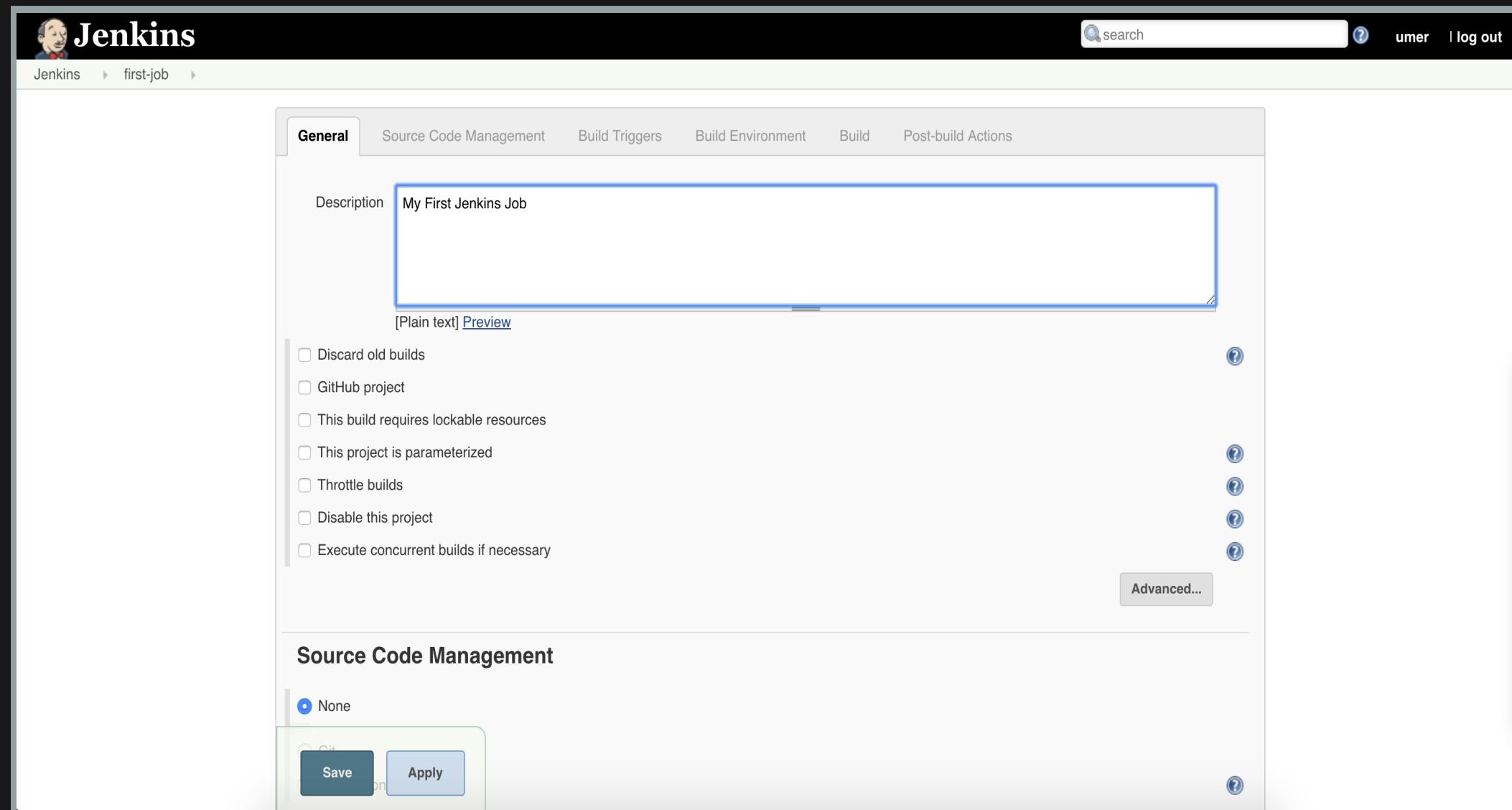
Enter name of your first job i.e. first-job as shown below, select Freestyle Project and click OK

The screenshot shows the Jenkins web interface for creating a new item. At the top, there's a navigation bar with the Jenkins logo, a search bar, and user information. Below it, a large input field is labeled "Enter an item name" and contains the text "first-job". A note below the input field says "» Required field". To the right of the input field, there's a list of project types:

- Freestyle project**: Described as the central feature of Jenkins, combining any SCM with any build system.
- Pipeline**: Orchestrates long-running activities across multiple build agents.
- Multi-configuration project**: Suitable for projects with many configurations, like testing on multiple environments.
- Folder**: Creates a container for grouping items together.
- GitHub Organization**: Scans a GitHub organization for repositories.
- Multibranch Pipeline**: Creates a set of Pipeline projects based on detected branches in one SCM repository.

At the bottom left of the dialog, there's an "OK" button.

Next specify description as show below and click Build. This will take you to the Build Section.



The screenshot shows the Jenkins configuration interface for a job named "first-job". The "General" tab is selected. In the "Description" field, the text "My First Jenkins Job" is entered. Below the description, there are several optional checkboxes: "Discard old builds", "GitHub project", "This build requires lockable resources", "This project is parameterized", "Throttle builds", "Disable this project", and "Execute concurrent builds if necessary". A "Save" button is visible at the bottom left of the configuration panel.

Select Execute shell as shown below.

The screenshot shows the Jenkins job configuration interface for a job named "first-job". The "Build" tab is active. In the "Build" section, a dropdown menu titled "Add build step" is open, showing various options: "Execute Windows batch command", "Execute shell" (which is highlighted), "Invoke Ant", "Invoke Gradle script", "Invoke top-level Maven targets", "Run with timeout", and "Set build status to "pending" on GitHub commit". At the bottom of this menu are two buttons: "Save" and "Apply".

Jenkins first-job

General Source Code Management Build Triggers Build Environment **Build** Post-build Actions

Build Environment

- Delete workspace before build starts
- Use secret text(s) or file(s)
- Abort the build if it's stuck
- Add timestamps to the Console Output
- Inspect build log for published Gradle build scans
- With Ant

Build

Add build step ▾

- Execute Windows batch command
- Execute shell**
- Invoke Ant
- Invoke Gradle script
- Invoke top-level Maven targets
- Run with timeout
- Set build status to "pending" on GitHub commit

Save Apply

Page generated: Jul 14, 2019 1:30:39 AM PKT REST API Jenkins ver. 2.176.1

Type 'echo Hello World' in the Command field and click Save. This job will run this command.

The screenshot shows the Jenkins job configuration interface for a job named "first-job". The "Build" tab is selected. In the "Build" section, there is one build step: "Execute shell". The "Command" field contains the text "echo Hello World". Below the command field, there is a link to "See the list of available environment variables" and an "Advanced..." button. At the bottom of the build step, there is a "Save" button, which is highlighted in green, and an "Apply" button.

Next you will run your first job. Click Build Now

The screenshot shows the Jenkins interface for a project named "first-job".

Left Sidebar: A vertical sidebar with icons and links:

- Back to Dashboard
- Status
- Changes
- Workspace
- Build Now** (highlighted in blue)
- Delete Project
- Configure
- Rename

Top Bar: Includes a search bar, user information, and a "ENABLE AUTO REFRESH" link.

Project Information: The title "Project first-job" is displayed, along with the subtitle "My First Jenkins Job".

Right Side Buttons: "edit description" and "Disable Project".

Content Area: Displays two links:

- Workspace
- Recent Changes

Bottom Left: A "Permalinks" section with a "Build History" tab, a search bar containing "find", and RSS feed links for "RSS for all" and "RSS for failures".

Bottom Right: Page footer with the URL "localhost:8080/job/first-job/build?delay=0sec", and generated time "Page generated: Jul 14, 2019 1:31:50 AM PKT REST API Jenkins ver. 2.176.1".

Once you click Build Now, you will notice a new entry in Build History

The screenshot shows the Jenkins interface for the project "first-job". The top navigation bar includes the Jenkins logo, a search bar, and user information. The left sidebar contains links for Back to Dashboard, Status, Changes, Workspace, Build Now, Delete Project, Configure, and Rename. The main content area is titled "Project first-job" and describes it as "My First Jenkins Job". It features icons for "Workspace" and "Recent Changes". On the right, there are "edit description" and "Disable Project" buttons. A "Permalinks" section on the left displays the "Build History" table, which shows one entry: "#1 Jul 14, 2019 1:32 AM". At the bottom, the URL "localhost:8080/job/first-job/1/" is shown, along with a footer indicating the page was generated on Jul 14, 2019, at 1:31:50 AM PKT, and the Jenkins version is 2.176.1.

Jenkins

search user log out

ENABLE AUTO REFRESH

Back to Dashboard

Status

Changes

Workspace

Build Now

Delete Project

Configure

Rename

Project first-job

My First Jenkins Job

edit description

Disable Project

Workspace

Recent Changes

Build History trend

find

#1 Jul 14, 2019 1:32 AM

RSS for all RSS for failures

Permalinks

localhost:8080/job/first-job/1/

Page generated: Jul 14, 2019 1:31:50 AM PKT REST API Jenkins ver. 2.176.1

Click on the new build to see details as shown below. Next click on Console Output

Jenkins search user log out
Jenkins first-job #1 ENABLE AUTO REFRESH

Back to Project Status Changes Console Output Edit Build Information Delete build '#1'

Build #1 (Jul 14, 2019 1:32:01 AM) Started 15 sec ago Took 0.13 sec add description

No changes.

Started by user umer

localhost:8080/job/first-job/1/console Page generated: Jul 14, 2019 1:32:17 AM PKT REST API Jenkins ver. 2.176.1

You should see Hello World displayed

 Jenkins

Jenkins > first-job > #1

[Back to Project](#) [Status](#) [Changes](#) [Console Output](#) [View as plain text](#) [Edit Build Information](#) [Delete build '#1'](#)

Console Output

```
Started by user umer
Running as SYSTEM
Building in workspace /Users/um255003/.jenkins/workspace/first-job
[first-job] $ /bin/sh -xe /var/folders/rw/w8yx5ssn4y77ff866v7_nnjc0000gn/T/jenkins3002509216377342745.sh
+ echo Hello World
Hello World
Finished: SUCCESS
```

Page generated: Jul 14, 2019 1:32:30 AM PKT [REST API](#) [Jenkins ver. 2.176.1](#)

EDIT EXISTING JOB

You are going to edit your first Jenkins job. Click on the previous job you created

The screenshot shows the Jenkins dashboard with the following details:

- Left Sidebar:** Includes links for "New Item", "People", "Build History", "Manage Jenkins", "My Views", "Lockable Resources", "Credentials", and "New View".
- Job List Header:** Shows filters "All" and "+".
- Job Table:** Displays one job named "first-job" with the following details:
 - Status: S (Stable)
 - Icon: Sun (W)
 - Name: [first-job](#)
 - Last Success: 19 min - #1
 - Last Failure: N/A
 - Last Duration: 0.13 sec
- Legend:** Shows icons for "S" (Stable), "M" (Medium), and "L" (Low).
- RSS Feeds:** Links for "RSS for all", "RSS for failures", and "RSS for just latest builds".
- Build Queue:** Shows "No builds in the queue."
- Build Executor Status:** Shows "1 Idle" and "2 Idle".
- Page Footer:** Includes the URL "localhost:8080/#", page generation time "Page generated: Jul 14, 2019 1:51:56 AM PKT", "REST API", and "Jenkins ver. 2.176.1".

Next click Configure to edit your job

The screenshot shows the Jenkins interface for the 'first-job' project. The top navigation bar includes the Jenkins logo, a search bar, and user information ('user | log out'). Below the header, the breadcrumb navigation shows 'Jenkins > first-job'. On the left, a sidebar lists project actions: Back to Dashboard, Status, Changes, Workspace, Build Now, Delete Project, Configure (which is underlined), and Rename. The main content area is titled 'Project first-job' and describes it as 'My First Jenkins Job'. It features two links: 'Workspace' (with a folder icon) and 'Recent Changes' (with a document icon). A 'Permalinks' section lists four build links: 'Last build (#1), 20 min ago', 'Last stable build (#1), 20 min ago', 'Last successful build (#1), 20 min ago', and 'Last completed build (#1), 20 min ago'. At the bottom of the page, there are RSS feed links for 'RSS for all' and 'RSS for failures'. The footer displays the URL 'localhost:8080/job/first-job/configure' and page statistics: 'Page generated: Jul 14, 2019 1:52:05 AM PKT | REST API | Jenkins ver. 2.176.1'.

localhost:8080/job/first-job/configure

Project first-job

My First Jenkins Job

edit description

Disable Project

Workspace

Recent Changes

Permalinks

- Last build (#1), 20 min ago
- Last stable build (#1), 20 min ago
- Last successful build (#1), 20 min ago
- Last completed build (#1), 20 min ago

RSS for all RSS for failures

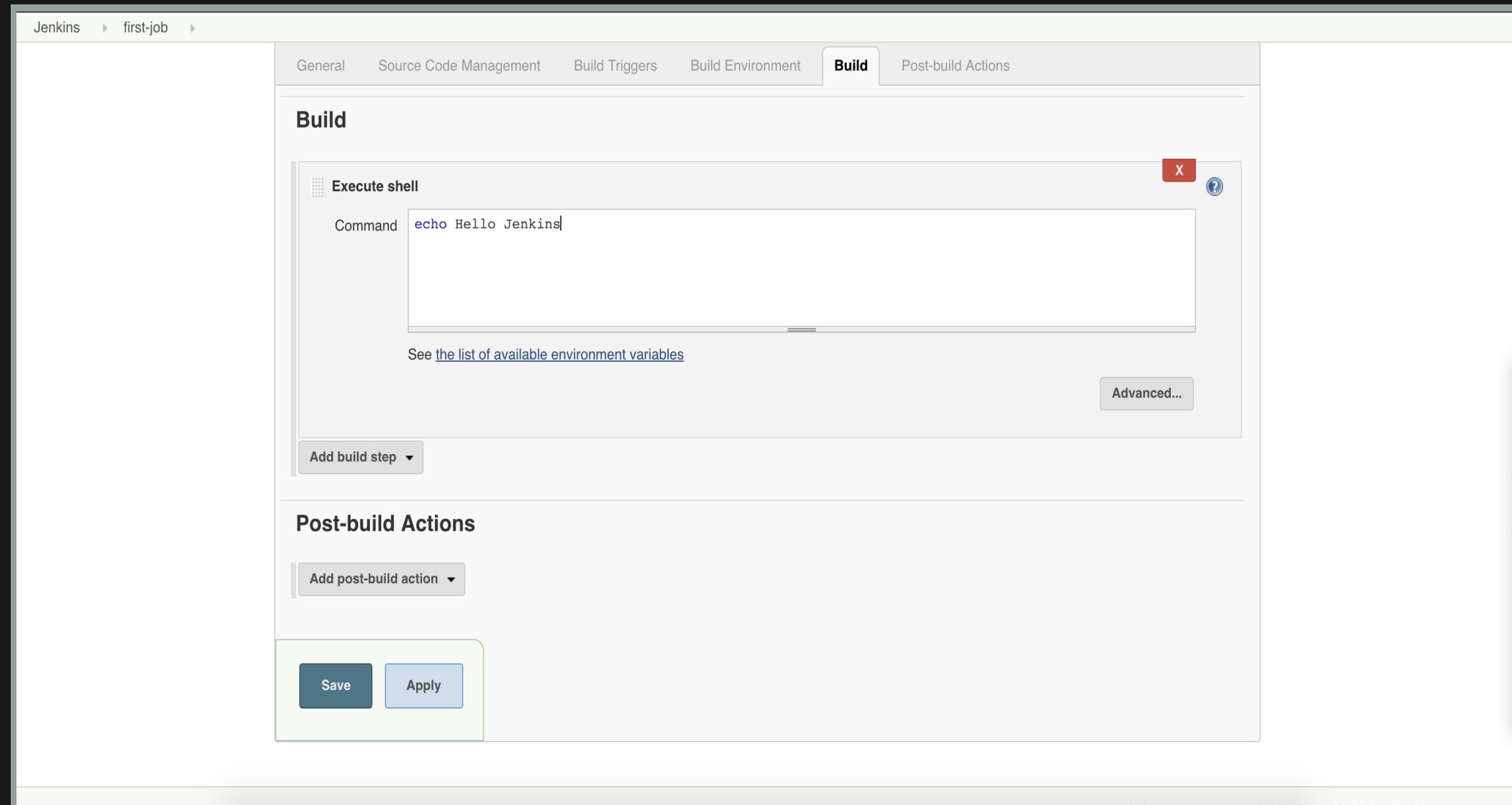
Page generated: Jul 14, 2019 1:52:05 AM PKT | REST API | Jenkins ver. 2.176.1

Click Build

The screenshot shows the Jenkins interface for configuring a new job named "first-job". The "General" tab is selected, displaying the following details:

- Description:** My First Jenkins Job
- Advanced Options:** Discard old builds, GitHub project, This build requires lockable resources, This project is parameterized, Throttle builds, Disable this project, Execute concurrent builds if necessary.
- Source Code Management:** Set to "None".
- Buttons:** Save, Apply.

In the command field replace World with Jenkins and Click Save



Next we are going to run our edited job for that click Build Now

The screenshot shows the Jenkins interface for the 'first-job' project. The top navigation bar includes the Jenkins logo, a search bar, and user information. The left sidebar contains links for Back to Dashboard, Status, Changes, Workspace, Build Now (which is underlined), Delete Project, Configure, and Rename. The main content area is titled 'Project first-job' and describes it as 'My First Jenkins Job'. It features two links: 'Workspace' (with a folder icon) and 'Recent Changes' (with a notebook icon). On the right side, there are buttons for 'edit description' (with a pencil icon) and 'Disable Project'. Below these are 'Permalinks' for 'Build History' (with a sun icon) and 'RSS feeds' for 'RSS for all' and 'RSS for failures'. At the bottom of the page, the URL 'localhost:8080/job/first-job/build?delay=0sec' is shown in a red box, and the footer indicates the page was generated on Jul 14, 2019 at 1:31:50 AM PKT, with links to REST API and Jenkins version 2.176.1.

Jenkins

search

user | log out

ENABLE AUTO REFRESH

Back to Dashboard

Status

Changes

Workspace

Build Now

Delete Project

Configure

Rename

edit description

Disable Project

Workspace

Recent Changes

Build History

trend

find

RSS for all RSS for failures

Permalinks

localhost:8080/job/first-job/build?delay=0sec

Page generated: Jul 14, 2019 1:31:50 AM PKT REST API Jenkins ver. 2.176.1

Once you click Build Now, you will notice a new entry in Build History

The screenshot shows the Jenkins interface for the project "first-job". The top navigation bar includes the Jenkins logo, a search bar, and user information. On the left, a sidebar provides links for Back to Dashboard, Status, Changes, Workspace, Build Now, Delete Project, Configure, and Rename. The main content area is titled "Project first-job" and describes it as "My First Jenkins Job". It features two links: "Workspace" (with a folder icon) and "Recent Changes" (with a document icon). On the right, there are "edit description" and "Disable Project" buttons. A prominent feature is the "Build History" section on the left, which lists the following builds:

#	Date
#2	Jul 14, 2019 1:52 AM
#1	Jul 14, 2019 1:32 AM

Below the build history are RSS feed links: "RSS for all" and "RSS for failures". At the bottom of the page, a footer indicates the page was generated on Jul 14, 2019, at 1:52:48 AM PKT, and provides links to the REST API and Jenkins version 2.176.1.

Click on the new build to see details as shown below. Next click on Console Output

Jenkins search user log out
Jenkins first-job #2 ENABLE AUTO REFRESH

Back to Project Status Changes Console Output Edit Build Information Delete build '#2' Previous Build

Build #2 (Jul 14, 2019 1:52:55 AM) Started 14 sec ago Took 59 ms add description

No changes.

Started by user umer

Page generated: Jul 14, 2019 1:53:10 AM PKT REST API Jenkins ver. 2.176.1

You should see Hello Jenkins displayed

The screenshot shows the Jenkins interface for a build. The left sidebar contains links: Back to Project, Status, Changes, Console Output (which is selected and highlighted in blue), View as plain text, Edit Build Information, Delete build '#2', and Previous Build. The main content area is titled "Console Output". It displays the following log output:

```
Started by user umer
Running as SYSTEM
Building in workspace /Users/um255003/.jenkins/workspace/first-job
[first-job] $ /bin/sh -xe /var/folders/rw/w8yx5ssn4y77ff866v7_nnjc0000gn/T/jenkins8387763687716329987.sh
+ echo Hello Jenkins
Hello Jenkins
Finished: SUCCESS
```

At the bottom of the page, there is a footer bar with the text "Page generated: Jul 14, 2019 1:53:19 AM PKT REST API Jenkins ver. 2.176.1".

RUN SHELL SCRIPT

You are going to create a job to run a shell script. First created a shell script by running the command

```
# create shell script  
vi /tmp/jenkins_script.sh
```

Copy following lines

```
#!/bin/sh  
echo Hello $1 $2
```

Now make the script executable

```
chmod u+x /tmp/jenkins_script.sh
```

Enter the job name, select Freestyle Project and Click OK

The screenshot shows the Jenkins web interface for creating a new item. At the top, the Jenkins logo and the word "Jenkins" are visible, along with a search bar, a user icon, and a "log out" link. Below the header, the URL "Jenkins > All >" is shown. The main content area has a title "Enter an item name" and a text input field containing "shell-job". A note below the input field states "» Required field". Below this, there is a list of project types with icons: "Freestyle project" (selected), "Pipeline", "Multi-configuration project", "Folder", "GitHub Organization", and "Multibranch Pipeline". Each item has a brief description. At the bottom left of the list, there is an "OK" button.

Enter an item name

shell-job

» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

GitHub Organization
Scans a GitHub organization (or user account) for all repositories matching some defined markers.

Multibranch Pipeline
Creates a set of Pipeline projects according to detected branches in one SCM repository.

OK

Enter you the description of your job and click Build to move to that section

The screenshot shows the Jenkins job configuration interface for a job named "shell-job". The "General" tab is selected. The "Description" field contains the text "My First Shell Job". Below the description, there are several checkboxes for build options:

- Discard old builds
- GitHub project
- This build requires lockable resources
- This project is parameterized
- Throttle builds
- Disable this project
- Execute concurrent builds if necessary

At the bottom of the General tab, there is an "Advanced..." button. The "Source Code Management" section is visible below it, showing the "None" option selected. There are "Save" and "Apply" buttons at the bottom of this section.

Select Execute shell in the Add build step drop down as shown below

The screenshot shows the Jenkins job configuration interface for a job named "shell-job". The "Build" tab is selected. In the "Build" section, there is a "Add build step" dropdown menu open. The "Execute shell" option is highlighted with a blue selection bar. Other options visible in the dropdown include "Execute Windows batch command", "Invoke Ant", "Invoke Gradle script", "Invoke top-level Maven targets", "Run with timeout", and "Set build status to "pending" on GitHub commit". Below the dropdown are two buttons: "Save" and "Apply". At the bottom of the page, the URL "localhost:8080/job/shell-job/configure#" is shown in the address bar, along with the page footer which includes the generation time "Page generated: Jul 14, 2019 2:23:02 AM PKT", links to "REST API" and "Jenkins ver. 2.176.1", and a small Jenkins logo.

In the command field type the following commands and Click Save

```
FirstName=FIRSTNAME  
LastName=LAST-NAME  
/tmp/jenkins_script $FirstName $LastName
```

Replace FirstName and LastName with your First name and last name

The screenshot shows the Jenkins job configuration interface for a 'shell-job'. The 'Build' tab is selected. Under the 'Execute shell' section, the 'Command' field contains the following script:

```
FirstName=Umer  
SecondName=Munir  
/tmp/jenkins_script.sh $FirstName $SecondName
```

Below the command field, there is a link to "See the list of available environment variables". At the bottom of the build step section is an "Advanced..." button. Below the build steps, the 'Post-build Actions' section has an "Add post-build action" button. At the bottom of the configuration page are two buttons: 'Save' and 'Apply'. The footer of the page includes the text "Page generated: Jul 14, 2019 2:25:06 AM PKT REST API Jenkins ver. 2.176.1".

Click Build Now to start manual build

The screenshot shows the Jenkins interface for the 'shell-job' project. At the top left is the Jenkins logo and the project name 'shell-job'. The top right features a search bar, user information ('user | log out'), and a 'ENABLE AUTO REFRESH' link. On the left, a sidebar lists project management options: Back to Dashboard, Status, Changes, Workspace, Build Now (highlighted in blue), Delete Project, Configure, and Rename. The main content area is titled 'Project shell-job' and displays the message 'My First Shell Job'. It includes links for 'Workspace' (with an icon of a folder) and 'Recent Changes' (with an icon of a notepad). To the right are buttons for 'edit description' (with a pencil icon) and 'Disable Project'. Below this is a 'Permalinks' section with a 'Build History' tab (selected, showing a 'find' input field and a 'trend' dropdown), an RSS feed link for all builds, and another for failed builds. At the bottom, a footer bar shows the URL 'localhost:8080/job/shell-job/build?delay=0sec', the generation time 'Page generated: Jul 14, 2019 2:24:34 AM PKT', and links for 'REST API' and 'Jenkins ver. 2.176.1'.

Jenkins

search

user | log out

ENABLE AUTO REFRESH

Back to Dashboard

Status

Changes

Workspace

Build Now

Delete Project

Configure

Rename

edit description

Disable Project

Workspace

Recent Changes

Build History trend →

find

RSS for all RSS for failures

Permalinks

Page generated: Jul 14, 2019 2:24:34 AM PKT REST API Jenkins ver. 2.176.1

localhost:8080/job/shell-job/build?delay=0sec

Successfull builds are color coded as blue and failed builds as red

The screenshot shows the Jenkins interface for the 'Project shell-job' project. The top navigation bar includes links for 'Jenkins', 'shell-job', 'search', 'user', 'log out', and 'ENABLE AUTO REFRESH'. On the left, a sidebar lists project management options: 'Back to Dashboard', 'Status', 'Changes', 'Workspace', 'Build Now', 'Delete Project', 'Configure', and 'Rename'. The main content area is titled 'Project shell-job' and displays the message 'My First Shell Job'. It features two links: 'Workspace' (with a folder icon) and 'Recent Changes' (with a notepad icon). A large 'Edit description' button is located in the top right. Below these, the 'Permalinks' section lists four build links: 'Last build (#1), 37 sec ago', 'Last failed build (#1), 37 sec ago', 'Last unsuccessful build (#1), 37 sec ago', and 'Last completed build (#1), 37 sec ago'. At the bottom, there are RSS feed links for 'RSS for all' and 'RSS for failures'. The footer indicates the page was generated on Jul 14, 2019 at 2:25:24 AM PKT, and shows the REST API and Jenkins version information.

Project shell-job

My First Shell Job

edit description

Disable Project

Workspace

Recent Changes

Permalinks

- Last build (#1), 37 sec ago
- Last failed build (#1), 37 sec ago
- Last unsuccessful build (#1), 37 sec ago
- Last completed build (#1), 37 sec ago

RSS for all RSS for failures

Page generated: Jul 14, 2019 2:25:24 AM PKT REST API Jenkins ver. 2.176.1

You can check the Console Output by clicking the down arrow next to latest build

The screenshot shows the Jenkins interface for the 'Project shell-job' project. The top navigation bar includes the Jenkins logo, a search bar, and user information ('user | log out'). Below the header, there's a sidebar with links: 'Back to Dashboard', 'Status', 'Changes', 'Workspace', 'Build Now', 'Delete Project', 'Configure', and 'Rename'. The main content area is titled 'Project shell-job' and displays the message 'My First Shell Job'. On the right side, there are buttons for 'edit description' and 'Disable Project'. Below the title, there are icons for 'Workspace' and 'Recent Changes'. A 'Permalinks' section lists four build links: 'Last build (#1), 37 sec ago', 'Last failed build (#1), 37 sec ago', 'Last unsuccessful build (#1), 37 sec ago', and 'Last completed build (#1), 37 sec ago'. At the bottom left, a 'Build History' panel shows a single build entry for '#2' on Jul 14, 2019 at 2:25 AM. The 'Console Output' link in the dropdown menu is highlighted with a blue background. The footer contains the URL 'localhost:8080/job/shell-job/2/console' and the page generation information 'Page generated: Jul 14, 2019 2:25:24 AM PKT REST API Jenkins ver. 2.176.1'.

localhost:8080/job/shell-job/2/console

Project shell-job

My First Shell Job

edit description

Disable Project

Workspace

Recent Changes

Permalinks

- Last build (#1), 37 sec ago
- Last failed build (#1), 37 sec ago
- Last unsuccessful build (#1), 37 sec ago
- Last completed build (#1), 37 sec ago

Build History

#2 Jul 14, 2019 2:25 AM

Changes

Console Output

Edit Build Information

Delete build '#2'

Page generated: Jul 14, 2019 2:25:24 AM PKT REST API Jenkins ver. 2.176.1

You should see your name in the Console Output section

The screenshot shows the Jenkins interface for a build named "shell-job" (Build #2). The left sidebar contains links for "Back to Project", "Status", "Changes", "Console Output" (which is selected and highlighted in blue), "View as plain text", "Edit Build Information", "Delete build '#2'", and "Previous Build". The main content area is titled "Console Output" and displays the following log output:

```
Started by user umer
Running as SYSTEM
Building in workspace /Users/um255003/.jenkins/workspace/shell-job
[shell-job] $ /bin/sh -xe /var/folders/rw/w8yx5ssn4y77ff866v7_nnjc0000gn/T/jenkins8239412053156803158.sh
+ FirstName=Umer
+ SecondName=Munir
+ /tmp/jenkins_script.sh Umer Munir
Hello Umer Munir
Finished: SUCCESS
```

At the bottom left, it says "localhost:8080". At the bottom right, it says "Page generated: Jul 14, 2019 2:25:57 AM PKT REST API Jenkins ver. 2.176.1".

PARAMETERIZED JOB

You are going to edit your shell job to parameterized job. Go to Jenkins landing page by clicking on Jenkins on the left top corner and click your shell job

The screenshot shows the Jenkins dashboard with the following details:

- Jobs:**
 - first-job:** Last Success: 40 min - #3, Last Failure: N/A
 - shell-job:** Last Success: 22 min - #2, Last Failure: 23 min - #1
- Build Queue:** No builds in the queue.
- Build Executor Status:** 1 Idle, 2 Idle

Page generated: Jul 14, 2019 2:47:49 AM PKT REST API Jenkins ver. 2.176.1

Click configure

Jenkins

shell-job

search user | log out

ENABLE AUTO REFRESH

Back to Dashboard

Status

Changes

Workspace

Build Now

Delete Project

Configure

Rename

Project shell-job

My First Shell Job

edit description

Disable Project

Workspace

Recent Changes

Permalinks

- Last build (#2), 22 min ago
- Last stable build (#2), 22 min ago
- Last successful build (#2), 22 min ago
- Last failed build (#1), 23 min ago
- Last unsuccessful build (#1), 23 min ago
- Last completed build (#2), 22 min ago

Build History trend =

find

#2 Jul 14, 2019 2:25 AM

#1 Jul 14, 2019 2:24 AM

RSS for all RSS for failures

localhost:8080/job/shell-job/configure

Page generated: Jul 14, 2019 2:47:56 AM PKT REST API Jenkins ver. 2.176.1

Check 'This project is parameterized box' as shown below and Click Add Parameter

The screenshot shows the Jenkins configuration interface for a job named "shell-job". The "General" tab is selected. In the "Description" field, the text "My First Shell Job" is entered. Below the description, there is a checkbox labeled "This project is parameterized", which is checked. To the right of this checkbox is a "Save" button. At the bottom of the page, there are "Save" and "Apply" buttons.

Jenkins search user log out

General Source Code Management Build Triggers Build Environment Build Post-build Actions

Description My First Shell Job

[Plain text] Preview

Discard old builds GitHub project This build requires lockable resources This project is parameterized Throttle builds Disable this project Execute concurrent builds if necessary

Add Parameter Advanced...

Source Code Management

Save Apply

Select String Parameter

The screenshot shows the Jenkins job configuration interface for a job named "shell-job". The "General" tab is selected. The "Description" field contains "My First Shell Job". Under the "Advanced" section, the "This project is parameterized" checkbox is checked. A dropdown menu titled "Add Parameter" is open, showing various parameter types: Boolean Parameter, Choice Parameter, Credentials Parameter, File Parameter, List Subversion tags (and more), Multi-line String Parameter, Password Parameter, Run Parameter, and String Parameter. The "String Parameter" option is highlighted with a blue selection bar. At the bottom of the configuration page, there are "Save" and "Apply" buttons.

Jenkins

search user log out

Jenkins shell-job

General Source Code Management Build Triggers Build Environment Build Post-build Actions

Description My First Shell Job

[Plain text] Preview

Discard old builds GitHub project This build requires lockable resources This project is parameterized

Add Parameter ▾

- Boolean Parameter
- Choice Parameter
- Credentials Parameter
- File Parameter
- List Subversion tags (and more)
- Multi-line String Parameter
- Password Parameter
- Run Parameter
- String Parameter

Advanced...

Save Apply

localhost:8080/job/shell-job/configure#

Specify Variable and Default Value as shown below. Replace First_Name and Last_Name with your First name and last name. Click Build to move to that section.

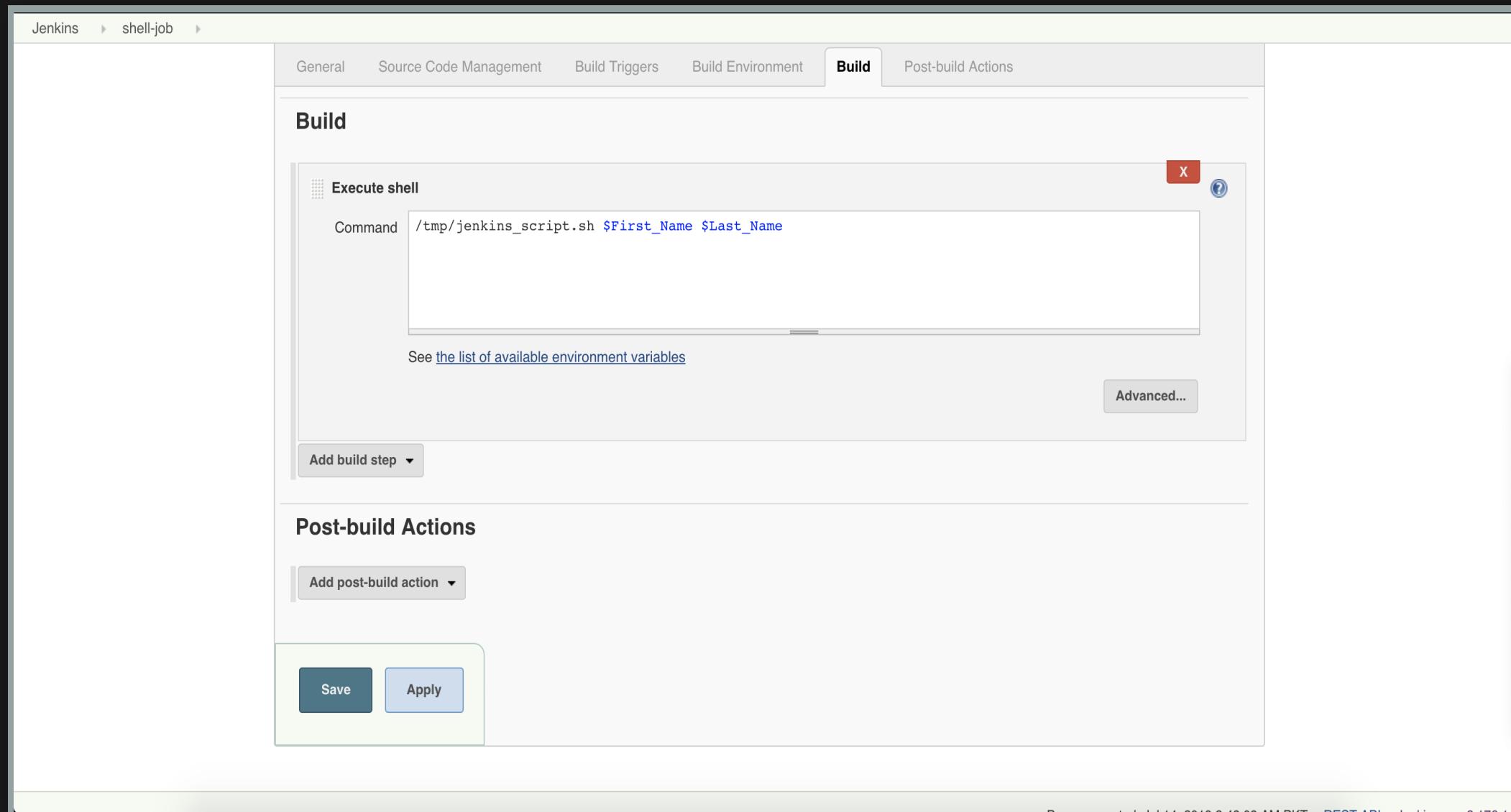
The screenshot shows the Jenkins job configuration interface for a job named "shell-job". The "General" tab is selected. Under the "This project is parameterized" section, two string parameters are defined:

- String Parameter**: Name is "First_Name" and the Default Value is "Umer". A checkbox for "Trim the string" is present but unchecked.
- String Parameter**: Name is "Last_Name" and the Default Value is "Munir".

At the bottom left, there are "Save" and "Apply" buttons.

Replace command field with the following command and Click Save

```
/tmp/jenkins_script.sh $First_Name $Last_Name
```



Click Build with Parameters to run the job.

Jenkins

shell-job

search user log out

ENABLE AUTO REFRESH

Back to Dashboard

Status

Changes

Workspace

[Build with Parameters](#)

Delete Project

Configure

Rename

Project shell-job

My First Shell Job

[edit description](#)

[Disable Project](#)

Workspace

Recent Changes

Permalinks

- [Last build \(#2\), 23 min ago](#)
- [Last stable build \(#2\), 23 min ago](#)
- [Last successful build \(#2\), 23 min ago](#)
- [Last failed build \(#1\), 24 min ago](#)
- [Last unsuccessful build \(#1\), 24 min ago](#)
- [Last completed build \(#2\), 23 min ago](#)

Build History

trend =

find

#2 Jul 14, 2019 2:25 AM

#1 Jul 14, 2019 2:24 AM

RSS for all RSS for failures

localhost:8080/job/shell-job/build?delay=0sec

Page generated: Jul 14, 2019 2:49:18 AM PKT REST API Jenkins ver. 2.176.1

It will prompt for parameters, enter the values and Click Build

The screenshot shows the Jenkins interface for a project named "shell-job". The top navigation bar includes links for "Jenkins", "shell-job", "search", and "umer | log out". On the left, a sidebar lists project actions: Back to Dashboard, Status, Changes, Workspace, Build with Parameters, Delete Project, Configure, and Rename. The main content area is titled "Project shell-job" and displays a message: "This build requires parameters:". Two input fields are present: "First_Name" containing "Umer" and "Last_Name" containing "Munir". A large blue "Build" button is centered below the fields. In the bottom-left corner, there is a "Build History" panel with a search bar and two entries: a successful build #2 from Jul 14, 2019 at 2:25 AM and a failed build #1 from Jul 14, 2019 at 2:24 AM. RSS feed links are provided for both. At the bottom right, a footer note states: "Page generated: Jul 14, 2019 2:49:31 AM PKT Jenkins ver. 2.176.1".

You should see your name in the Console Output section

The screenshot shows the Jenkins interface for a build named "shell-job" (Build #3). The left sidebar contains links for "Back to Project", "Status", "Changes", "Console Output" (which is selected), "View as plain text", "Edit Build Information", "Delete build '#3'", "Parameters", and "Previous Build". The main content area is titled "Console Output" and displays the following log output:

```
Started by user umer
Running as SYSTEM
Building in workspace /Users/um255003/.jenkins/workspace/shell-job
[shell-job] $ /bin/sh -xe /var/folders/rw/w8yx5ssn4y77ff866v7_nnjc0000gn/T/jenkins17560281140238839.sh
+ /tmp/jenkins_script.sh Umer Munir
Hello Umer Munir
Finished: SUCCESS
```

At the bottom of the page, there is a footer bar with the text "Page generated: Jul 14, 2019 2:49:48 AM PKT REST API Jenkins ver. 2.176.1".

GITHUB INTEGRATION

- Integrate Jenkins with GitHub
- Integration built into the default installation
- Number of plugins for integrating with GitHub
- Trigger jobs
 - Polling
 - Service Hook
- Status can be recorded back to GitHub
- Itself located on [GitHub Jenkins](#)
- GitHub Branch Source Plugin

LAB - GITHUB INTEGRATION

Go to Jenkins landing page by clicking on Jenkins on the left top corner and click New Item.

The screenshot shows the Jenkins landing page. On the left, there is a sidebar with various links: New Item, People, Build History, Manage Jenkins, My Views, Lockable Resources, Credentials, and New View. Below these are two collapsed sections: Build Queue (No builds in the queue) and Build Executor Status (1 Idle, 2 Idle). The main content area displays a table of jobs. The table has columns: S (Status), W (Weather), Name (sorted), Last Success, Last Failure, and Last Duration. There are two rows: 'first-job' (Status: Green, Weather: Sun, Last Success: 12 hr - #3, Last Failure: N/A, Last Duration: 56 ms) and 'shell-job' (Status: Green, Weather: Cloudy, Last Success: 11 hr - #3, Last Failure: 12 hr - #1, Last Duration: 88 ms). At the bottom of the main content, there is a legend and three RSS feed links: RSS for all, RSS for failures, and RSS for just latest builds. The footer of the page includes the text "Page generated: Jul 14, 2019 2:40:29 PM PKT" and links to REST API and Jenkins ver. 2.176.1.

S	W	Name ↓	Last Success	Last Failure	Last Duration
		first-job	12 hr - #3	N/A	56 ms
		shell-job	11 hr - #3	12 hr - #1	88 ms

Icon: [S](#) [M](#) [L](#)

Legend [RSS for all](#) [RSS for failures](#) [RSS for just latest builds](#)

Page generated: Jul 14, 2019 2:40:29 PM PKT [REST API](#) [Jenkins ver. 2.176.1](#)

Enter the job name, select Freestyle project and Click OK

The screenshot shows the Jenkins interface for creating a new item. At the top, there's a navigation bar with the Jenkins logo, a search bar, and user information. Below it, the main content area has a title "Enter an item name" and a required input field containing "first-github-job". A list of project types is displayed:

- Freestyle project**: Described as the central feature of Jenkins, combining any SCM with any build system.
- Pipeline**: Orchestrates long-running activities across multiple build agents.
- Multi-configuration project**: Suitable for projects with many configurations, like testing on multiple environments.
- Folder**: Creates a container for grouping items together.
- GitHub Organization**: Scans a GitHub organization for repositories.
- Multibranch Pipeline**: Creates Pipeline projects based on detected branches in an SCM repository.

A blue button labeled "OK" is visible at the bottom left of the list.

Enter job description and click Source Code Management

The screenshot shows the Jenkins job configuration interface for a job named "first-github-job". The "General" tab is selected, displaying the job's description as "My First Job with GitHub Integration". Below the description are several configuration options, each with a help icon (blue question mark) to its right:

- Discard old builds
- GitHub project
- This build requires lockable resources
- This project is parameterized
- Throttle builds
- Disable this project
- Execute concurrent builds if necessary

A "Advanced..." button is located at the bottom right of this section. Below the general settings is the "Source Code Management" section, which currently has "None" selected. There are also "Git" and "Mercurial" options available. At the bottom of the configuration screen are "Save" and "Apply" buttons.

Select Git

Jenkins ➤ first-github-job ➤

General Source Code Management Build Triggers Build Environment Build Post-build Actions

Source Code Management

None Git

Repositories

Repository URL (?)
Please enter Git repository.

Credentials

Branches to build

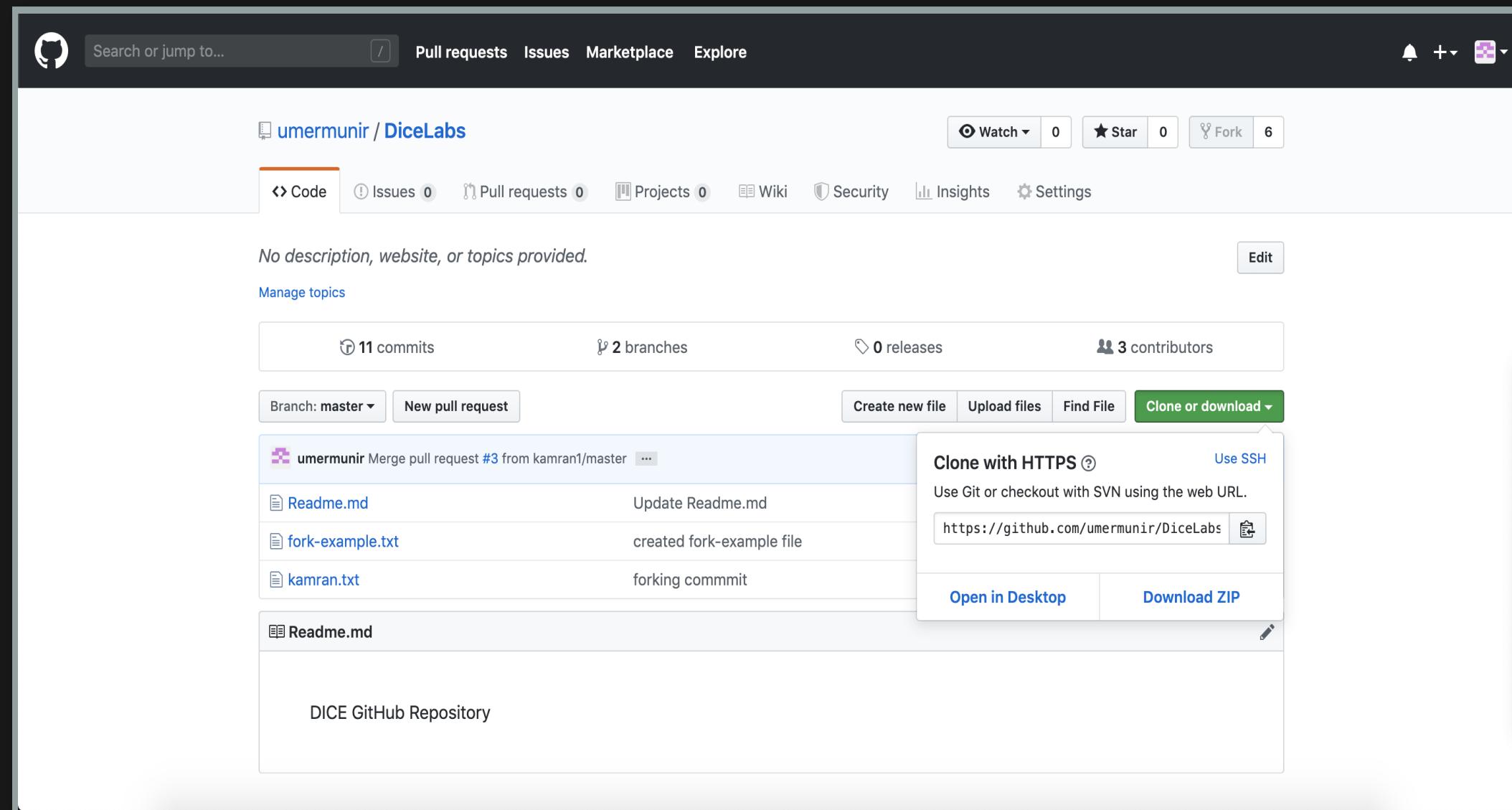
Branch Specifier (blank for 'any') (X) (?)

Repository browser (?)

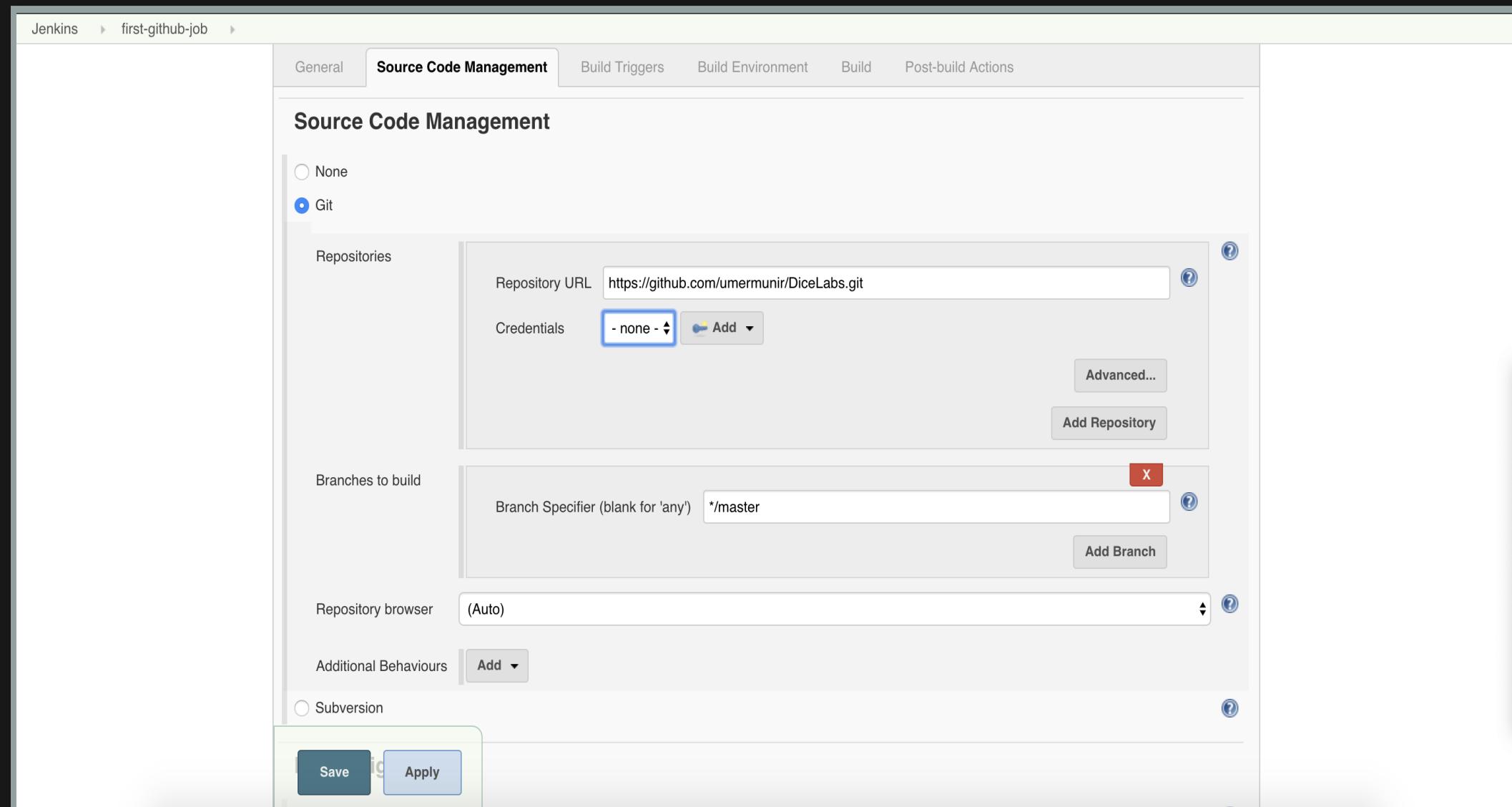
Additional Behaviours

Subversion (?)

Go to the repository page that you have already forked from the DiceLabs repository and copy the link as shown below



Next paste the URL in the Repository URL as shown below



Click Build Trigger to specify the triggers for this job

The screenshot shows the Jenkins configuration interface for a job named "first-github-job". The top navigation bar includes links for Jenkins, first-github-job, General, Source Code Management, Build Triggers (which is selected), Build Environment, Build, and Post-build Actions.

The "Build Triggers" section contains the following configuration:

- Trigger builds remotely (e.g., from scripts) (with a question mark icon)
- Build after other projects are built (with a question mark icon)
- Build periodically (with a question mark icon)
- GitHub hook trigger for GITScm polling (with a question mark icon)
- Poll SCM (with a question mark icon)

The "Build Environment" section contains the following configuration:

- Delete workspace before build starts (with a question mark icon)
- Use secret text(s) or file(s) (with a question mark icon)
- Abort the build if it's stuck (with a question mark icon)
- Add timestamps to the Console Output (with a question mark icon)
- Inspect build log for published Gradle build scans (with a question mark icon)
- With Ant (with a question mark icon)

The "Build" section includes an "Add build step" button.

The "Post-build Actions" section is currently empty, showing an "Add post-build action" button.

At the bottom of the configuration area, there are "Save" and "Apply" buttons.

For this job you are going to select Poll SCM, which means you are going to poll GitHub as per the specified schedule. For this job we want Jenkins to poll GitHub to check for any commits every five minutes

The screenshot shows the Jenkins interface for configuring a job named "first-github-job". The "Build Triggers" tab is selected. Under "Repository browser", "(Auto)" is chosen. In the "Build Triggers" section, the "Poll SCM" option is checked, and the "Schedule" field contains the cron expression "H/5 * * * *". A tooltip below the schedule field states: "Would last have run at Monday, July 15, 2019 11:24:43 PM PKT; would next run at Monday, July 15, 2019 11:29:43 PM PKT." The "Ignore post-commit hooks" checkbox is unchecked. At the bottom, there are "Save" and "Apply" buttons.

Click Build and Select Execute Shell

The screenshot shows the Jenkins job configuration interface for a job named "first-github-job". The "Build" tab is active. In the "Build" section, a dropdown menu titled "Add build step" is open, showing several options: "Execute Windows batch command", "Execute shell" (which is highlighted with a blue selection bar), "Invoke Ant", "Invoke Gradle script", "Invoke top-level Maven targets", "Run with timeout", and "Set build status to "pending" on GitHub commit". At the bottom of this menu are two buttons: "Save" and "Apply". The main configuration area below the tabs contains sections for "Build Environment" and "Post-build Actions", each with several configuration options.

Jenkins first-github-job

General Source Code Management Build Triggers Build Environment **Build** Post-build Actions

Build Environment

- Delete workspace before build starts
- Use secret text(s) or file(s)
- Abort the build if it's stuck
- Add timestamps to the Console Output
- Inspect build log for published Gradle build scans
- With Ant

Build

Add build step ▾

- Execute Windows batch command
- Execute shell**
- Invoke Ant
- Invoke Gradle script
- Invoke top-level Maven targets
- Run with timeout
- Set build status to "pending" on GitHub commit

Save Apply

localhost:8080/job/first-github-job/configure#

Page generated: Jul 14, 2019 2:42:26 PM PKT REST API Jenkins ver. 2.176.1

Specify ls as the command and Click Save

The screenshot shows the Jenkins job configuration interface for a job named "first-github-job". The "Build" tab is selected. In the "Build" section, there is a single build step of type "Execute shell" with the command "ls" entered. The "Save" button at the bottom left is highlighted.

Jenkins first-github-job

General Source Code Management Build Triggers Build Environment Build Post-build Actions

Build Environment

- Delete workspace before build starts
- Use secret text(s) or file(s)
- Abort the build if it's stuck
- Add timestamps to the Console Output
- Inspect build log for published Gradle build scans
- With Ant

Build

Execute shell

Command `ls`

See [the list of available environment variables](#)

Advanced...

Add build step ▾

Save Apply

 Jenkins

Jenkins > first-github-job >

[Back to Dashboard](#)

[Status](#)

[Changes](#)

[Workspace](#)

[Build Now](#)

[Delete Project](#)

[Configure](#)

[Rename](#)

[Build History](#) [trend](#)

find

[RSS for all](#) [RSS for failures](#)

Project first-github-job

My First Job with GitHub Integration

[edit description](#)

[Disable Project](#)

 [Workspace](#)

 [Recent Changes](#)

Permalinks

Page generated: Jul 14, 2019 2:54:37 PM PKT [REST API](#) [Jenkins ver. 2.176.1](#)

Next create a file in the repository that you entered in the Source Code Management section and push that change to GitHub. You should see that it will trigger your job

The screenshot shows the Jenkins interface for the project "first-github-job". The top navigation bar includes a search field, user information, and links for "ENABLE AUTO REFRESH", "Help", and "Log Out". On the left, a sidebar provides links for "Back to Dashboard", "Status", "Changes", "Workspace", "Build Now", "Delete Project", "Configure", and "Rename". The main content area is titled "Project first-github-job" and describes it as "My First Job with GitHub Integration". It features two buttons: "edit description" and "Disable Project". Below these are links for "Workspace" (with a folder icon) and "Recent Changes" (with a notepad icon). A "Permalinks" section lists four build links: "Last build (#1), 2 min 26 sec ago", "Last stable build (#1), 2 min 26 sec ago", "Last successful build (#1), 2 min 26 sec ago", and "Last completed build (#1), 2 min 26 sec ago". At the bottom, there is a "Build History" panel showing a single build entry for "#1" on Jul 14, 2019 at 2:59 PM, with options to view "RSS for all" or "RSS for failures". The footer of the page includes the message "Page generated: Jul 14, 2019 3:01:57 PM PKT", links to "REST API" and "Jenkins ver. 2.176.1", and a "Page refresh" button.

Project first-github-job

My First Job with GitHub Integration

edit description

Disable Project

Workspace

Recent Changes

Permalinks

- Last build (#1), 2 min 26 sec ago
- Last stable build (#1), 2 min 26 sec ago
- Last successful build (#1), 2 min 26 sec ago
- Last completed build (#1), 2 min 26 sec ago

Build History

#1 Jul 14, 2019 2:59 PM

RSS for all RSS for failures

Page generated: Jul 14, 2019 3:01:57 PM PKT REST API Jenkins ver. 2.176.1

Check the Output Console of build, you should see logs as shown below

Jenkins > first-github-job > #1

Console Output

Status
Changes
Console Output
View as plain text
Edit Build Information
Delete build '#1'
Git Build Data
No Tags

```
Started by timer
Running as SYSTEM
Building in workspace /Users/um255003/.jenkins/workspace/first-github-job
No credentials specified
Cloning the remote Git repository
Cloning repository https://github.com/umermunir/DiceLabs.git
> git init /Users/um255003/.jenkins/workspace/first-github-job # timeout=10
Fetching upstream changes from https://github.com/umermunir/DiceLabs.git
> git --version # timeout=10
> git fetch --tags --progress https://github.com/umermunir/DiceLabs.git +refs/heads/*\:refs/remotes/origin/*
> git config remote.origin.url https://github.com/umermunir/DiceLabs.git # timeout=10
> git config --add remote.origin.fetch +refs/heads/*\:refs/remotes/origin/* # timeout=10
> git config remote.origin.url https://github.com/umermunir/DiceLabs.git # timeout=10
Fetching upstream changes from https://github.com/umermunir/DiceLabs.git
> git fetch --tags --progress https://github.com/umermunir/DiceLabs.git +refs/heads/*\:refs/remotes/origin/*
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision 6d6c4732f3b21580c2b333b0dd5da25d9d5a2215 (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 6d6c4732f3b21580c2b333b0dd5da25d9d5a2215
Commit message: "Create jenkins-test.txt"
First time build. Skipping changelog.
[first-github-job] $ /bin/sh -xe /var/folders/rw/w8yx5ssn4y77ff866v7_nnjc0000gn/T/jenkins859425953946515190.sh
+ ls
Readme.md
fork-example.txt
jenkins-test.txt
kamran.txt
Finished: SUCCESS
```

Page generated: Jul 14, 2019 3:02:10 PM PKT [REST API](#) Jenkins ver. 2.176.1

GITHUB WEBHOOKS

We need to add a service to call the Jenkins Github webhook on a push, to do this go to settings -> integrations & Services and add a new service. The Jenkins Github plugin service should be in the list of available services.

The screenshot shows the GitHub settings page for the repository "marcbest / medium-jenkins-git-tut". The top navigation bar includes "Watch" (0), "Star" (0), and "Fork" (0) buttons. Below the navigation bar are links for "Code", "Issues 0", "Pull requests 0", "Projects 0", "Wiki", "Pulse", "Graphs", and "Settings". The "Settings" link is highlighted with a red box. On the left, a sidebar menu lists "Options", "Collaborators", "Branches", "Webhooks", "Integrations & services" (which is highlighted with a red box), and "Deploy keys". The main content area is titled "Installed integrations" and contains a message about integrating with commercial, open source, and homegrown tools, with links to "Browse our directory" and "build your own". Below this is a section titled "Services" with a "Add service" button. A modal window titled "Available Services" is open, showing a list with "github" at the top and "Jenkins (GitHub plugin)" highlighted with a red box. At the bottom of the page, there are links for "Contact GitHub", "API", "Training", "Shop", "Blog", and "About".

Enter the URL of your Jenkins instance followed by /github-webhook/

Services / Manage Jenkins (GitHub plugin) Test service

Jenkins is a popular continuous integration server.
Using the Jenkins GitHub Plugin you can automatically trigger build jobs when pushes are made to GitHub.

Install Notes

1. "Jenkins Hook Url" is the URL of your Jenkins server's webhook endpoint. For example: `http://ci.jenkins-ci.org/github-webhook/`.

For more information see <https://wiki.jenkins-ci.org/display/JENKINS/GitHub+plugin>.

Jenkins hook url

Active
We will run this service when an event is triggered.

Update service Delete service

We need to give the Jenkins user access to our repository by adding a deploy key in the Github settings. For that you will create an SSH key

```
jenkins@ip:/home/ubuntu$ ssh-keygen
```

Copy the public key so that it can be added to Github

```
jenkins@ip:/home/ubuntu$ cat ~/.ssh/id_rsa.pub
```

Add the key copied in the previous step to Github. To do this head to the repository settings -> Deploy keys

The screenshot shows the GitHub repository settings page for 'marcbest / medium-jenkins-git-tut'. The 'Deploy keys' tab is selected in the sidebar. A red box highlights the main input area where a new deploy key is being added. The 'Title' field contains 'Jenkins' and the 'Key' field contains a long SSH RSA key. Below the key, there is a checkbox for 'Allow write access' which is unchecked. A green 'Add key' button is at the bottom.

marcbest / medium-jenkins-git-tut

Watch 0 Star 0 Fork 0

Code Issues 0 Pull requests 0 Projects 0 Wiki Pulse Graphs Settings

Add deploy key

Options

Collaborators

Branches

Webhooks

Integrations & services

Deploy keys

Deploy keys

There are no deploy keys for this repository

Title

Jenkins

Key

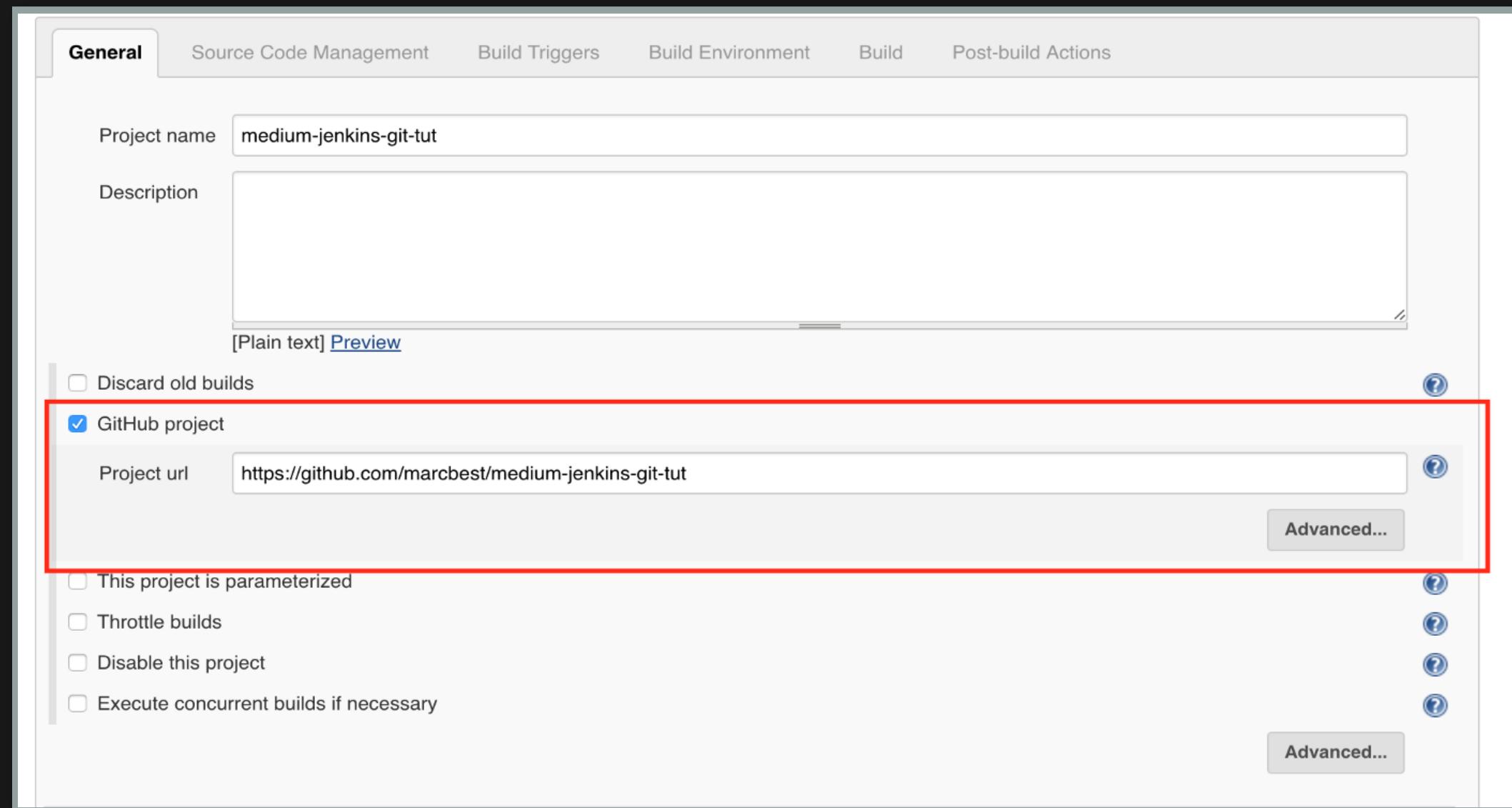
```
ssh-rsa AAAAB3NzaC1yc2EAAAQABAAQCoD1zZ6nhwzXyQF1p7wiHM1dBDAS1IFycfFgCGuh0R4dHGbObfvI3wU3ddPk3FnB5Khzkwc+ZEouJBZvp0wO/ZU7QtVPBJa0N22sJZ1ijzNcdKu6wS51+aQizKdc+4w6nGoSf3L/q2kpg6Yb47YfaiwTPZsS0MOfBCuel9IYw+O9+rVrd4U8D3cDr/5yghbJJuvzsrHtvx9VYmRTk8nofRFom2IDbALykJcvOAtl7ofz7UVF0dkqcufoWnD8FrDqPr/iuM8EcnuEhCo6fnALElxPegvr0OMwLnUkpxQIBDalqfaRL2DApH5fpqf0jXiZs2vCsYCtDJ71b0p/p ienkins@ip-172-31-22-109
```

Allow write access

Can this key be used to push to this repository? Deploy keys always have pull access.

Add key

In the General section of the job configuration check the Github project tick box and enter the URL to the repository



Next update the Source Code Management section, first set the repository URL (note the format `git@github.com:{YOUR_REPO}`).

Source Code Management

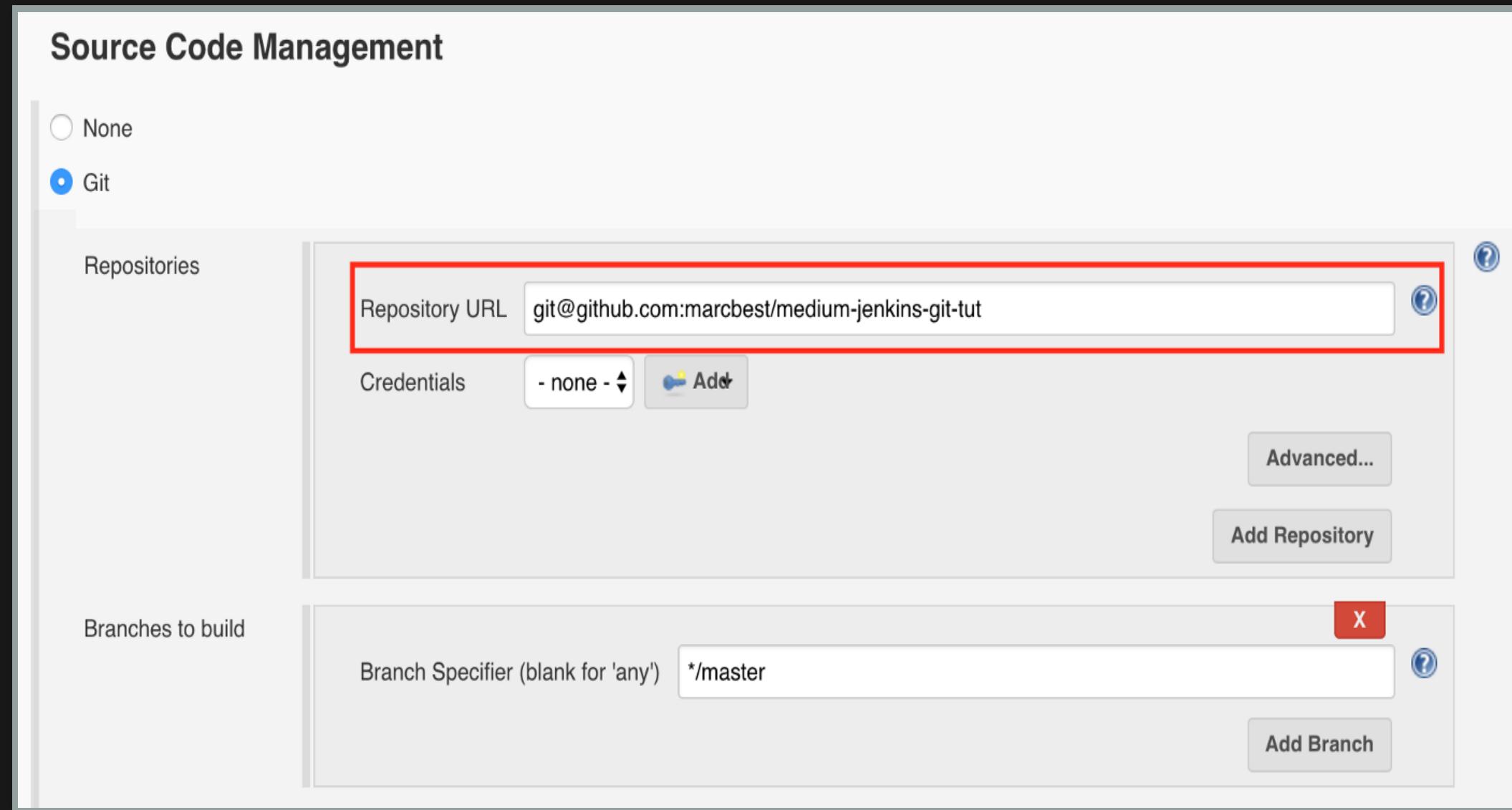
None
 Git

Repositories

Repository URL	git@github.com:marcbest/medium-jenkins-git-tut	?
Credentials	- none -	Add
Advanced...		
Add Repository		

Branches to build

Branch Specifier (blank for 'any')	*/master	X	?
Add Branch			



The last step is to tell Jenkins to build when the Github hook is called

Build Triggers

- Trigger builds remotely (e.g., from scripts) ?
- Build after other projects are built ?
- Build periodically ?
- GitHub Branches
- GitHub Pull Requests ?
- GitHub hook trigger for GITScm polling ?
- Poll SCM ?

POST BUILD ACTION

- Allows user to run tasks after build
- Rich set of Plugins
 - Email Notification
 - Slack Notification
 - Build Other Project
 - Archive Artifacts
 - Build Status on GitHub

LAB - EMAIL

In this Lab you are going to create a job to send email notification when the build fails. Create New Item, Enter the job name, select Freestyle Project and Select OK

The screenshot shows the Jenkins interface for creating a new item. At the top, there's a navigation bar with the Jenkins logo, a search bar, and user information. Below it, the main content area has a title "Enter an item name" and a text input field containing "post-action-job". A note below the input says "» Required field". The page lists several project types with icons:

- Freestyle project**: This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.
- Pipeline**: Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
- Multi-configuration project**: Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
- Folder**: Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.
- GitHub Organization**: Scans a GitHub organization (or user account) for all repositories matching some defined markers.
- Branch Pipeline**: Creates a set of Pipeline projects according to detected branches in one SCM repository.

A prominent blue "OK" button is located at the bottom left of the list of project types.

Enter Job Description and Click Source Code Management

The screenshot shows the Jenkins job configuration interface for a job named "post-action-job".

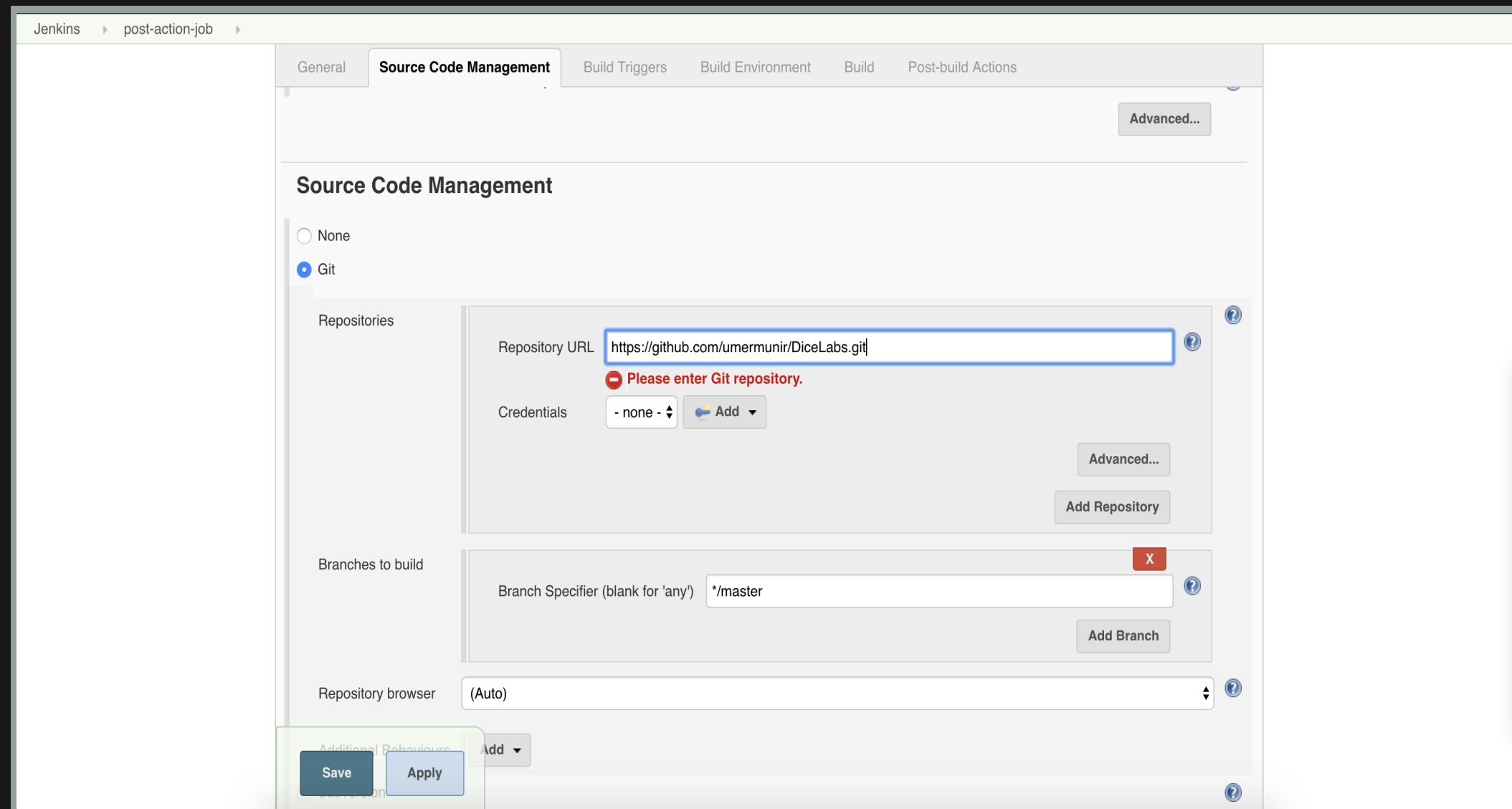
General Tab:

- Description:** Post Action Job
- Post-build Actions:** Discard old builds, GitHub project, This build requires lockable resources, This project is parameterized, Throttle builds, Disable this project, Execute concurrent builds if necessary.
- Advanced...** button

Source Code Management Tab:

- Source Code Management:** None selected (radio button is checked)
- Buttons:** Save, Apply

Enter the URL of repository that you forked from DiceLabs repository and Click Build Triggers



Select Poll SCM and specify the schedule as shown below and Click Build

The screenshot shows the Jenkins job configuration interface for a job named "post-action-job". The "Build Triggers" tab is selected, displaying various trigger options. The "Poll SCM" option is checked, and its schedule is set to "H/5 * * * *". A note below the schedule indicates the last run was at July 16, 2019, 12:46:41 AM PKT, and the next run is scheduled for July 16, 2019, 12:51:41 AM PKT. The "Build Environment" tab is also visible, containing options like "Delete workspace before build starts". At the bottom, there are "Save" and "Apply" buttons.

Jenkins > post-action-job >

General Source Code Management **Build Triggers** Build Environment Build Post-build Actions

Additional Behaviours **Add ▾**

Subversion ?

Build Triggers

Trigger builds remotely (e.g., from scripts) ?
 Build after other projects are built ?
 Build periodically ?
 GitHub hook trigger for GITScm polling ?
 Poll SCM ?

Schedule **H/5 * * * *** ?

Would last have run at Tuesday, July 16, 2019 12:46:41 AM PKT; would next run at Tuesday, July 16, 2019 12:51:41 AM PKT.

Ignore post-commit hooks ?

Build Environment

Delete workspace before build starts ?

Save **Apply**

Enter a incorrect command as shown below so that we can have a failed build

The screenshot shows the Jenkins job configuration for 'post-action-job'. The 'Build Environment' tab is selected. Under the 'Build' section, there is a 'Execute shell' step with the command 'las'. This command is likely a typo for 'ls' (list directory contents), which would result in a failure. The 'Post-build Actions' section is also visible at the bottom.

Jenkins > post-action-job >

General Source Code Management Build Triggers **Build Environment** Build Post-build Actions

Delete workspace before build starts Use secret text(s) or file(s) Abort the build if it's stuck Add timestamps to the Console Output Inspect build log for published Gradle build scans With Ant

Build

Execute shell

Command `las`

See [the list of available environment variables](#)

Post-build Actions

E-mail Notification

Save Apply

In the Post-build Actions select Email Notification, enter your email address and select 'Send email for every unstable build' as shown below

The screenshot shows the Jenkins job configuration interface for a job named "post-action-job". The "Build" tab is selected. Below it, the "Post-build Actions" section is open, displaying the "E-mail Notification" action. The "Recipients" field contains "umer.munir@live.com". The "Send e-mail for every unstable build" checkbox is checked, while the "Send separate e-mails to individuals who broke the build" checkbox is unchecked. At the bottom of the "Post-build Actions" section, there are "Save" and "Apply" buttons.

Jenkins > post-action-job >

General Source Code Management Build Triggers Build Environment **Build** Post-build Actions

See [the list of available environment variables](#)

Add build step ▾

Post-build Actions

E-mail Notification

Recipients umer.munir@live.com

Whitespace-separated list of recipient addresses. May reference build parameters like \$PARAM. E-mail will be sent when a build fails, becomes unstable or returns to stable.

Send e-mail for every unstable build

Send separate e-mails to individuals who broke the build

Add post-build action ▾

Save Apply

Page generated: Jul 16, 2019 12:46:40 AM PKT [REST API](#) [Jenkins ver. 2.176.1](#)

Next you are going to configure email that will be used by Jenkins to send Email Notification, for that click on Manage Jenkins and Configure System

The screenshot shows the Jenkins Manage Jenkins interface. On the left, there's a sidebar with various management links: New Item, People, Build History, Manage Jenkins (which is selected and highlighted in purple), My Views, Lockable Resources, Credentials, and New View. Below this are two collapsed sections: Build Queue (No builds in the queue) and Build Executor Status (1 Idle, 2 Idle). At the bottom left is a URL bar showing 'localhost:8080/configure'. The main content area is titled 'Manage Jenkins' and contains several configuration options: 'Configure System' (selected), 'Configure Global Security', 'Configure Credentials', 'Global Tool Configuration', 'Reload Configuration from Disk', 'Manage Plugins', 'System Information', and 'System Log'. The 'Configure System' section has a sub-description: 'Configure global settings and paths.' The 'Configure Global Security' section has a sub-description: 'Secure Jenkins; define who is allowed to access/use the system.' The 'Configure Credentials' section has a sub-description: 'Configure the credential providers and types.' The 'Global Tool Configuration' section has a sub-description: 'Configure tools, their locations and automatic installers.' The 'Reload Configuration from Disk' section has a sub-description: 'Discard all the loaded data in memory and reload everything from file system. Useful when you modified config files directly on disk.' The 'Manage Plugins' section has a sub-description: 'Add, remove, disable or enable plugins that can extend the functionality of Jenkins.' The 'System Information' section has a sub-description: 'Displays various environmental information to assist trouble-shooting.' The 'System Log' section is at the bottom.

Go to email Notification section and enter details as shown below and click Save

E-mail Notification

SMTP server	smtp.gmail.com
Default user e-mail suffix	
<input checked="" type="checkbox"/> Use SMTP Authentication	
User Name	umer.munirrr@gmail.com
Password
Use SSL	<input checked="" type="checkbox"/>
SMTP Port	465
Reply-To Address	
Charset	UTF-8
<input type="checkbox"/> Test configuration by sending test e-mail	
<input type="button" value="Save"/> <input type="button" value="Apply"/>	

Next go to job and click build now. Next click on the failed build

The screenshot shows the Jenkins interface for the 'post-action-job' project. The top navigation bar includes the Jenkins logo, a search bar, and user information ('user | log out'). A link to 'ENABLE AUTO REFRESH' is also present.

The main content area is titled 'Project post-action-job' and describes it as a 'Post Action Job'. On the left, a sidebar provides links for 'Back to Dashboard', 'Status', 'Changes', 'Workspace', 'Build Now' (which is underlined to indicate it's active), 'Delete Project', 'Configure', 'Git Polling Log', and 'Rename'. On the right, there are buttons for 'edit description' (with a pencil icon) and 'Disable Project' (in a dark blue box).

In the center, there are two icons: 'Workspace' (a folder icon) and 'Recent Changes' (a document with a pencil icon). Below these are 'Permalinks' with a list of four links:

- [Last build \(#3\), 3 min 14 sec ago](#)
- [Last failed build \(#3\), 3 min 14 sec ago](#)
- [Last unsuccessful build \(#3\), 3 min 14 sec ago](#)
- [Last completed build \(#3\), 3 min 14 sec ago](#)

To the left of the permalinks is a 'Build History' section with a 'trend' dropdown set to 'trend'. It contains a search bar with 'find' and three build entries:

#	Date
3	Jul 16, 2019 1:00 AM
2	Jul 16, 2019 12:55 AM
1	Jul 16, 2019 12:51 AM

At the bottom of the history section are links for 'RSS for all' and 'RSS for failures'.

The footer of the page displays the URL 'localhost:8080/job/post-action-job/build?delay=0sec' and the page generation time 'Page generated: Jul 16, 2019 1:04:09 AM PKT REST API Jenkins ver. 2.176.1'.

You will notice that you got an error while sending an email because Gmail account needs to be configured to send email from an untrusted source which in this case is Jenkins

```
Jenkins  ▶ post-action-job  ▶ #3

[post-action-job] $ /bin/sh -xe /var/folders/rw/w8yx5ssn4y77ff866v7_nnjc0000gn/T/jenkins8526146985632271005.sh
+ las
/var/folders/rw/w8yx5ssn4y77ff866v7_nnjc0000gn/T/jenkins8526146985632271005.sh: line 2: las: command not found
Build step 'Execute shell' marked build as failure
Sending e-mails to: umer.munir@live.com
ERROR: 535-5.7.8 Username and Password not accepted. Learn more at
535 5.7.8 https://support.google.com/mail/?p=BadCredentials w67sm21148427wma.24 - gsmtp

javax.mail.AuthenticationFailedException: 535-5.7.8 Username and Password not accepted. Learn more at
535 5.7.8 https://support.google.com/mail/?p=BadCredentials w67sm21148427wma.24 - gsmtp

        at com.sun.mail.smtp.SMTPTransport$Authenticator.authenticate(SMTPTransport.java:809)
        at com.sun.mail.smtp.SMTPTransport.authenticate(SMTPTransport.java:752)
        at com.sun.mail.smtp.SMTPTransport.protocolConnect(SMTPTransport.java:669)
        at javax.mail.Service.connect(Service.java:317)
        at javax.mail.Service.connect(Service.java:176)
        at javax.mail.Service.connect(Service.java:125)
        at javax.mail.Transport.send0(Transport.java:194)
        at javax.mail.Transport.send(Transport.java:124)
        at hudson.tasks.Mailer.perform(Mailer.java:131)
        at hudson.tasks.Mailer.perform(Mailer.java:173)
        at hudson.tasks.Mailer.perform(Mailer.java:136)
        at hudson.tasks.BuildStepMonitor$1.perform(BuildStepMonitor.java:20)
        at hudson.model.AbstractBuild$AbstractBuildExecution.perform(AbstractBuild.java:741)
        at hudson.model.AbstractBuild$AbstractBuildExecution.performAllBuildSteps(AbstractBuild.java:690)
        at hudson.model.Build$BuildExecution.post2(Build.java:186)
        at hudson.model.AbstractBuild$AbstractBuildExecution.post(AbstractBuild.java:635)
        at hudson.model.Run.execute(Run.java:1843)
        at hudson.model.FreeStyleBuild.run(FreeStyleBuild.java:43)
        at hudson.model.ResourceController.execute(ResourceController.java:97)
        at hudson.model.Executor.run(Executor.java:429)
Finished: FAILURE

Page generated: Jul 16, 2019 1:04:22 AM PKT  REST API  Jenkins ver. 2.176.1
```

JENKINS JOB DSL

- Plugin that allows you to define jobs in programmatic form
- DSL stands for Domain Specific Language
 - Groovy as scripting language
- Easier to manage jobs
 - Use UI if you have less jobs to maintain
 - When number of jobs grow use DSL
- Easier to restore
- Can use with Version Control
- Job Structure
 - Parameters
 - SCM
 - Triggers
 - Build Steps
 - Post-build actions

LAB - JOB DSL

In this Lab, you are going to create your job using a DSL job. First we are going to install Job DSL plugin. Go to the landing page and Click Manage Jenkins

The screenshot shows the Jenkins dashboard with the following details:

- Left Sidebar:** Includes links for New Item, People, Build History, Manage Jenkins, My Views, Lockable Resources, Credentials, and New View.
- Top Bar:** Features a search bar, user information (username), and log out options. There is also an "ENABLE AUTO REFRESH" link.
- Job List:** A table displaying three jobs:

S	W	Name ↓	Last Success	Last Failure	Last Duration
		first-github-job	13 min - #23	13 hr - #22	3.3 sec
		first-job	1 day 21 hr - #3	N/A	56 ms
		shell-job	1 day 20 hr - #3	1 day 21 hr - #1	88 ms
- Bottom Status:** Shows the Build Queue (No builds in the queue) and Build Executor Status (1 Idle, 2 Idle).
- Page Footer:** Includes a timestamp (Page generated: Jul 15, 2019 11:37:56 PM PKT), REST API link, and Jenkins version (Jenkins ver. 2.176.1).

Next, click Manage Plugins

Screenshot of the Jenkins Manage Jenkins page.

The left sidebar shows navigation links:

- New Item
- People
- Build History
- Manage Jenkins** (selected)
- My Views
- Lockable Resources
- Credentials
- New View

Below the sidebar are two collapsed sections:

- Build Queue**: No builds in the queue.
- Build Executor Status**: 1 Idle, 2 Idle.

The main content area is titled "Manage Jenkins" and contains several configuration options:

- Configure System**: Configure global settings and paths.
- Configure Global Security**: Secure Jenkins; define who is allowed to access/use the system.
- Configure Credentials**: Configure the credential providers and types.
- Global Tool Configuration**: Configure tools, their locations and automatic installers.
- Reload Configuration from Disk**: Discard all the loaded data in memory and reload everything from file system. Useful when you modified config files directly on disk.
- Manage Plugins** (highlighted with a box): Add, remove, disable or enable plugins that can extend the functionality of Jenkins.
- System Information**: Displays various environmental information to assist trouble-shooting.
- System Log**

The bottom left corner shows the URL: `localhost:8080/pluginManager`.

Click Available and search for 'dsl' as shown below

The screenshot shows the Jenkins Plugin Manager interface. At the top, there is a navigation bar with the Jenkins logo, a search bar containing 'search', and user information ('user | log out'). Below the navigation bar, the page title is 'Plugin Manager'.

On the left side, there are two links: 'Back to Dashboard' and 'Manage Jenkins'. The main content area has four tabs at the top: 'Updates' (disabled), 'Available' (selected), 'Installed', and 'Advanced'. A search bar on the right is set to 'Filter: dsl'.

The central part of the screen displays a table of available plugins. The columns are 'Name' and 'Version'. The table is sorted by name. Each plugin entry includes a checkbox for installation and a brief description.

Install ↓	Name	Version
<input type="checkbox"/>	Job DSL This plugin allows Jobs and Views to be defined via DSLs	1.74
<input type="checkbox"/>	OpenShift Client This plugin provides Jenkins pipeline DSL interactions for OpenShift.	1.0.31
<input type="checkbox"/>	Alauda DevOps Pipeline This plugin provides Jenkins pipeline DSL interactions for kubernetes.	2.0.3
<input type="checkbox"/>	Alauda Pipeline This plugin provides Jenkins pipeline DSL interactions for Alauda.io	1.4.3
<input type="checkbox"/>	BTC DSL for Pipeline This provides simple build DSLs for Jenkins Pipelines.	2.4.0
<input type="checkbox"/>	JX Pipelines This provides simple build DSLs for Jenkins Pipelines.	1.0.15
<input type="checkbox"/>	Simple Build DSL for Pipeline This provides simple build DSLs for Jenkins Pipelines.	0.2
<input type="checkbox"/>	XML Job to Job DSL Use this plugin to convert your jobs into DSL Groovy scripts	0.1.10

At the bottom of the page, there are three buttons: 'Install without restart', 'Download now and install after restart', and 'Check now'. A status message 'Update information obtained: 15 min ago' is also present.

Select 'Job DSL' and click 'Download now and install after restart'

The screenshot shows the Jenkins Plugin Manager interface. At the top, there's a navigation bar with the Jenkins logo, a search bar, and user information. Below it, the 'Plugin Manager' page has a sidebar with links to 'Back to Dashboard' and 'Manage Jenkins'. The main area has tabs for 'Updates', 'Available' (which is selected), 'Installed', and 'Advanced'. A search bar at the top right is set to 'Filter: dsl'. The main content is a table listing available plugins:

Install	Name	Version
<input checked="" type="checkbox"/>	Job DSL This plugin allows Jobs and Views to be defined via DSLs	1.74
<input type="checkbox"/>	OpenShift Client This plugin provides Jenkins pipeline DSL interactions for OpenShift.	1.0.31
<input type="checkbox"/>	Alauda DevOps Pipeline This plugin provides Jenkins pipeline DSL interactions for kubernetes.	2.0.3
<input type="checkbox"/>	Alauda Pipeline This plugin provides Jenkins pipeline DSL interactions for Alauda.io	1.4.3
<input type="checkbox"/>	BTC DSL for Pipeline This provides simple build DSLs for Jenkins Pipelines.	2.4.0
<input type="checkbox"/>	JX Pipelines This provides simple build DSLs for Jenkins Pipelines.	1.0.15
<input type="checkbox"/>	Simple Build DSL for Pipeline This provides simple build DSLs for Jenkins Pipelines.	0.2
<input type="checkbox"/>	XML Job to Job DSL Use this plugin to convert your jobs into DSL Groovy scripts	0.1.10

At the bottom, there are three buttons: 'Install without restart', 'Download now and install after restart' (which is highlighted in blue), and 'Check now'. A status message 'Update information obtained: 15 min ago' is also present.

Wait for the plugin to installed and jenkins to be restarted

The screenshot shows the Jenkins Update Center interface. At the top, there's a navigation bar with the Jenkins logo, a search bar, and user information. Below the navigation, the main content area has a title "Installing Plugins/Upgrades". On the left, there's a sidebar with links to "Back to Dashboard", "Manage Jenkins", and "Manage Plugins". The main content area includes a "Preparation" section with a bulleted list: "Checking internet connectivity", "Checking update center connectivity", and "Success". Below that is a "Job DSL" section with a progress bar labeled "Installing". At the bottom, there are two green arrow icons with text: one pointing right to "Go back to the top page" and another below it to "Restart Jenkins when installation is complete and no jobs are running". A footer at the bottom right indicates the page was generated on July 15, 2019, at 11:42:36 PM PKT, and provides links to the REST API and Jenkins version information.

Jenkins

search

user | log out

ENABLE AUTO REFRESH

Back to Dashboard

Manage Jenkins

Manage Plugins

Installing Plugins/Upgrades

Preparation

- Checking internet connectivity
- Checking update center connectivity
- Success

Job DSL

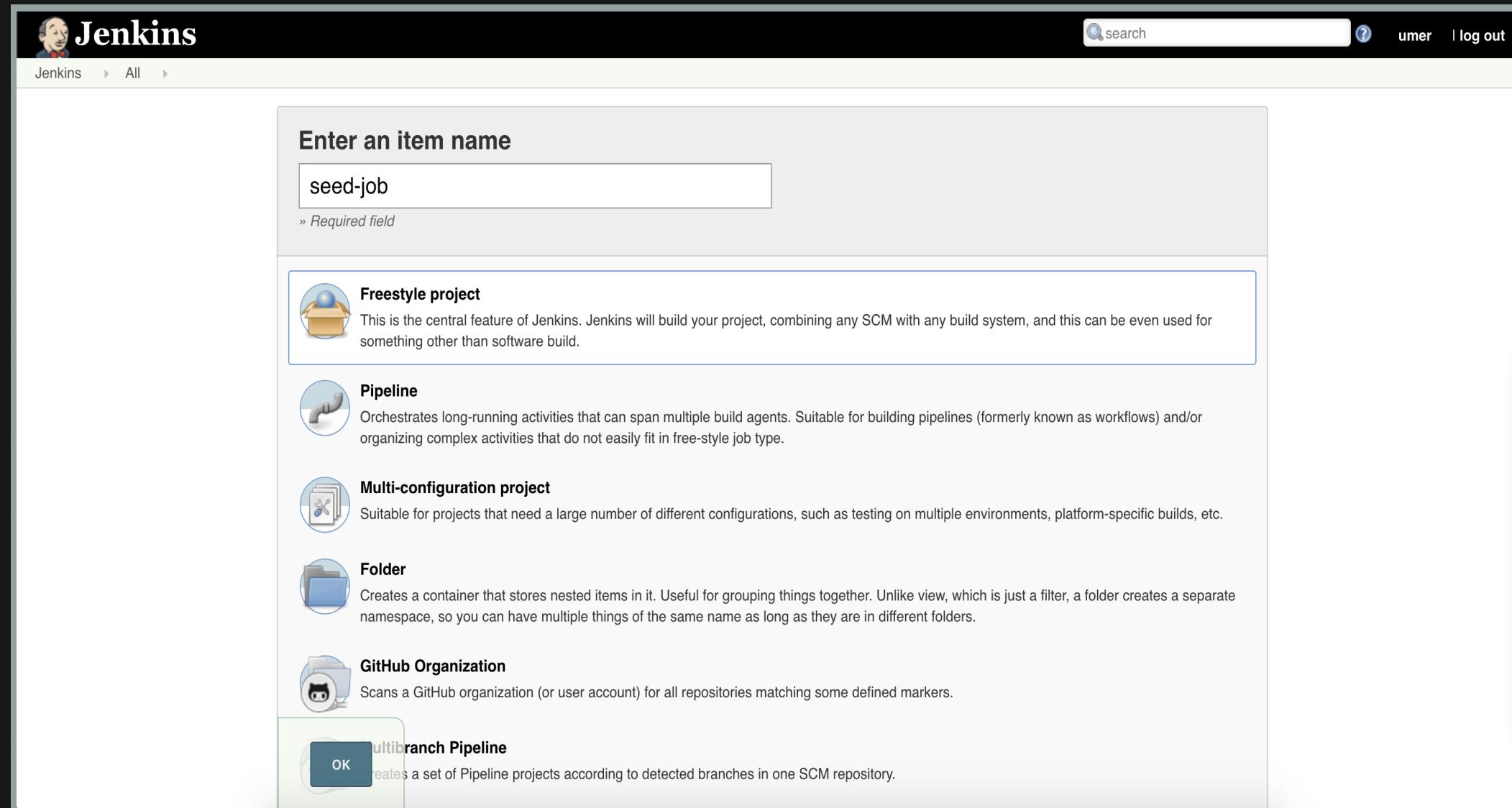
Installing

Go back to the top page
(you can start using the installed plugins right away)

Restart Jenkins when installation is complete and no jobs are running

Page generated: Jul 15, 2019 11:42:36 PM PKT [REST API](#) [Jenkins ver. 2.176.1](#)

Once restarted and create new item as shown below



The screenshot shows the Jenkins web interface for creating a new item. At the top, there is a navigation bar with the Jenkins logo, a search bar, and user information. Below the navigation bar, the main content area has a title "Enter an item name" and a required input field containing "seed-job". A message below the input field states "» Required field".

The page lists several project types:

- Freestyle project**: This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.
- Pipeline**: Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
- Multi-configuration project**: Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
- Folder**: Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.
- GitHub Organization**: Scans a GitHub organization (or user account) for all repositories matching some defined markers.
- Multibranch Pipeline**: Creates a set of Pipeline projects according to detected branches in one SCM repository.

A blue "OK" button is visible at the bottom left of the list of project types.

Enter job description and Click Build

The screenshot shows the Jenkins job configuration interface for a job named "seed-job". The "Build" tab is selected in the top navigation bar, which also includes tabs for General, Source Code Management, Build Triggers, Build Environment, Post-build Actions, and Help.

The main content area is divided into three sections:

- Build Environment**: Contains several checkboxes for build-related actions:
 - Delete workspace before build starts
 - Use secret text(s) or file(s)
 - Abort the build if it's stuck
 - Add timestamps to the Console Output
 - Inspect build log for published Gradle build scans
 - With Ant
- Build**: Contains a button labeled "Add build step ▾".
- Post-build Actions**: Contains a button labeled "Add post-build action ▾".

At the bottom of the configuration panel, there are two buttons: "Save" and "Apply".

The footer of the page displays the following information: "Page generated: Jul 16, 2019 12:23:34 AM PKT" and links to "REST API" and "Jenkins ver. 2.176.1".

Select 'Process Job DSLs'

The screenshot shows the Jenkins job configuration interface for a job named "seed-job". The "Build" tab is active. In the "Build" section, under "Add build step", the "Process Job DSLs" option is highlighted with a blue selection bar.

Build Environment

- Delete workspace before build starts
- Use secret text(s) or file(s)
- Abort the build if it's stuck
- Add timestamps to the Console Output
- Inspect build log for published Gradle build scans
- With Ant

Build

Add build step ▾

- Execute Windows batch command
- Execute shell
- Invoke Ant
- Invoke Gradle script
- Invoke top-level Maven targets
- Process Job DSLs**
- Run with timeout
- Set build status to "pending" on GitHub commit

localhost:8080/job/seed-job/configure#

Page generated: Jul 16, 2019 12:23:34 AM PKT [REST API](#) [Jenkins ver. 2.176.1](#)

Enter the DSL job

```
def gitUrl = "https://github.com/umermunir/DiceLabs.git"

job('test-job') {
    description "My First DSL Job with GitHub Integration"
    scm {
        git {
            remote {
                url(gitUrl)
            }
            branch('*/*')
        }
    }
    triggers {
        scm('H/5 * * * *')
    }
    steps {
        shell "ls"
    }
}
```

Click Save

The screenshot shows the Jenkins interface for configuring a job named "seed-job". The "Build" tab is selected. A modal window titled "Process Job DSLs" is open, containing a Groovy script for defining a Jenkins job.

```
1 def gitUrl = "https://github.com/umermunir/DiceLabs.git"
2
3 job('test-job') {
4     description "My First DSL Job with GitHub Integration"
5     scm {
6         git {
7             remote {
8                 url(gitUrl)
9             }
10            branch('*/*')
11        }
12    }
13    triggers {
14        scm('H/5 * * * *')
15    }
16    steps {
17        shell "ls"
18    }
19}
```

The modal also contains options for "Look on Filesystem" and "Use Groovy Sandbox". At the bottom, there are "Save" and "Apply" buttons, along with checkboxes for "Ignore changes", "Fail if seed collision", and "Action for existing jobs and views managed by another seed job".

Click Build Now to run the job

Jenkins

seed-job

search user log out

ENABLE AUTO REFRESH

Back to Dashboard

Status

Changes

Workspace

Build Now

Delete Project

Configure

Job DSL API Reference

Rename

Build History trend

find

#1 Jul 16, 2019 12:27 AM

RSS for all RSS for failures

Project seed-job

Workspace

Recent Changes

Generated Items:

- test-job

add description

Disable Project

Permalinks

- Last build (#1), 8.6 sec ago
- Last stable build (#1), 8.6 sec ago ▾
- Last successful build (#1), 8.6 sec ago
- Last completed build (#1), 8.6 sec ago

localhost:8080/job/seed-job/lastStableBuild/

Page generated: Jul 16, 2019 12:27:13 AM PKT REST API Jenkins ver. 2.176.1

Check Console Output to verify if the new job has been created

 Jenkins

Jenkins > seed-job > #1

[Back to Project](#) [Status](#) [Changes](#) [Console Output](#) [View as plain text](#) [Edit Build Information](#) [Delete build '#1'](#)

Console Output

```
Started by user umer
Running as SYSTEM
Building in workspace /Users/um255003/.jenkins/workspace/seed-job
Processing provided DSL script
Added items:
    GeneratedJob{name='test-job'}
Finished: SUCCESS
```

Page generated: Jul 16, 2019 12:27:24 AM PKT [REST API](#) [Jenkins ver. 2.176.1](#)

Check if the job has been created by going to the landing page

Jenkins

search | ? user | log out

New Item People Build History Manage Jenkins My Views Lockable Resources Credentials New View

All +

S	W	Name ↓	Last Success	Last Failure	Last Duration
		first-github-job	1 hr 19 min - #23	14 hr - #22	3.3 sec
		first-job	1 day 22 hr - #3	N/A	56 ms
		seed-job	1 min 15 sec - #6	2 min 54 sec - #5	0.17 sec
		shell-job	1 day 21 hr - #3	1 day 22 hr - #1	88 ms
		test-job ▾	N/A	N/A	N/A

Build Queue

No builds in the queue.

Build Executor Status

1 Idle
2 Idle

localhost:8080/job/test-job/

Page generated: Jul 16, 2019 12:43:41 AM PKT REST API Jenkins ver. 2.176.1

Click on newly created job to check configuration

The screenshot shows the Jenkins job configuration interface for a job named "test-job".

General Tab:

- Description:** My First DSL Job with GitHub Integration
- Advanced Options:** Discard old builds, GitHub project, This build requires lockable resources, This project is parameterized, Throttle builds, Disable this project, Execute concurrent builds if necessary.
- Buttons:** Advanced...

Source Code Management Tab:

- Source Code Management Type:** Git (selected)
- Repository URL:** https://github.com/umermunir/DiceLabs.git
- Buttons:** Save, Apply

Jenkins ➤ test-job ➤

General Source Code Management Build Triggers **Build Environment** Build Post-build Actions

Use secret text(s) or file(s) ?

Abort the build if it's stuck ?

Add timestamps to the Console Output ?

Inspect build log for published Gradle build scans ?

With Ant ?

Build

Execute shell X ?

Command `ls`

See [the list of available environment variables](#) Advanced...

Add build step ▾

Post-build Actions

Save Apply

JENKINS PIPELINE

- Set of plugins which support implementing and integrating continuous delivery pipelines into Jenkins
- Is a job type to perform sequence of steps
 - Build
 - Test
 - Deploy
- A Pipeline can be created
 - Through UI
 - In SCM
- Types
 - Scripted Pipeline
 - Declarative Pipeline

SCRIPTED PIPELINE

Follows a more imperative programming model built with Groovy.

```
node {
    stage ('Build') {
        //...
    }
    stage ('Test') {
        //...
    }
    stage ('Deploy') {
        //...
    }
}
```

DECLARATIVE PIPELINE

Presents a more simplified and opinionated syntax on top of the Pipeline sub-systems

```
pipeline {  
    agent any  
    stages {  
        stage('Build') {  
            steps {  
                //..  
            }  
        }  
        stage ('Test') {  
            steps {  
                //..  
            }  
        }  
    }  
}
```

LAB - FIRST PIPELINE

In this Lab, you are going to create a Jenkins Pipeline Job without any SCM integration. Go to Jenkins landing page by clicking on Jenkins on the left top corner and click New Item.

The screenshot shows the Jenkins dashboard with the following details:

- Left Sidebar:** Includes links for "New Item", "People", "Build History", "Manage Jenkins", "My Views", "Lockable Resources", "Credentials", and "New View".
- Top Bar:** Features a search bar, user information, and "ENABLE AUTO REFRESH" options.
- Job List:** A table displays one job:

S	W	Name ↓	Last Success	Last Failure	Last Duration
		first-job	15 min - #3	N/A	56 ms
- Legend:** Shows RSS icons for "RSS for all", "RSS for failures", and "RSS for just latest builds".
- Build Queue:** Displays "No builds in the queue."
- Build Executor Status:** Shows "1 Idle" and "2 Idle".
- Page Footer:** Includes "Page generated: Jul 14, 2019 2:22:31 AM PKT", "REST API", and "Jenkins ver. 2.176.1".

Enter the name of the job and select Pipeline and Click OK

Jenkins

2

search

umer | log out

Jenkins All

Enter an item name

first-pipeline-job

» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

GitHub Organization
Scans a GitHub organization (or user account) for all repositories matching some defined markers.

Multibranch Pipeline
Creates a set of Pipeline projects according to detected branches in one SCM repository.

OK

Enter the description and Click Pipeline.

The screenshot shows the Jenkins configuration interface for a job named "first-pipeline-job".

General Tab:

- Description:** First Pipeline Job
- Buttons:** [Plain text] [Preview](#)
- Checkboxes:**
 - Discard old builds
 - Do not allow concurrent builds
 - Do not allow the pipeline to resume if the master restarts
 - GitHub project
 - Pipeline speed/durability override
 - Preserve stashes from completed builds
 - This project is parameterized
 - Throttle builds
- Build Triggers:**
 - Build after other projects are built
 - Build periodically
 - Scm polling
 - GIT
- Buttons:** Save, Apply

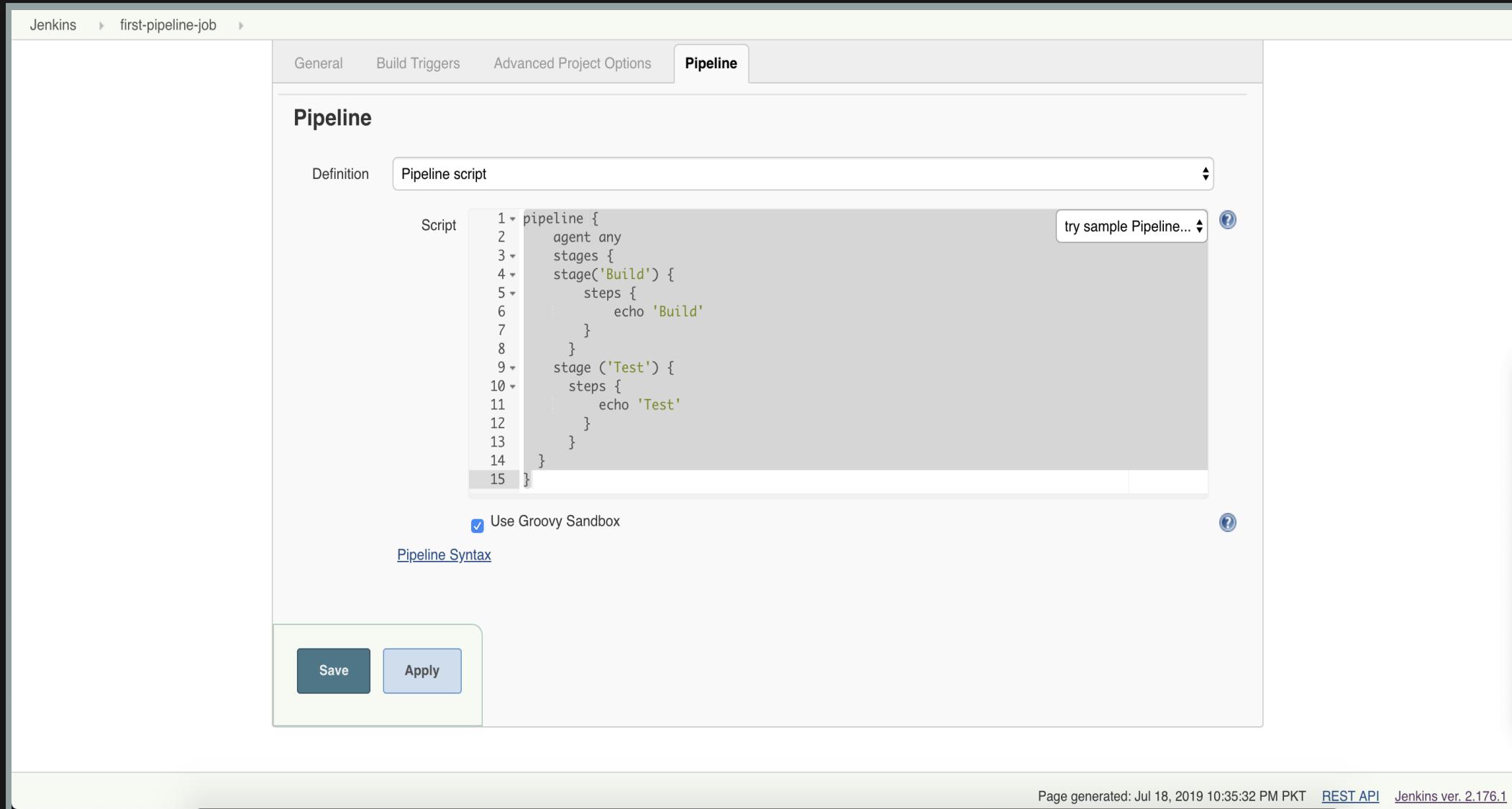
Select Pipeline script in the Definition drop box.

The screenshot shows the Jenkins interface for defining a new pipeline job. The top navigation bar includes links for Jenkins, first-pipeline-job, General, Build Triggers, Advanced Project Options, and Pipeline. The Pipeline tab is currently active. In the main content area, under the Pipeline section, the 'Definition' dropdown is set to 'Pipeline script'. Below this, there is a large text area labeled 'Script' containing the number '1'. To the right of this text area is a button labeled 'try sample Pipeline...' with a question mark icon. At the bottom of the script area, there is a checked checkbox labeled 'Use Groovy Sandbox' and a link labeled 'Pipeline Syntax'. At the very bottom of the page, there are two buttons: 'Save' and 'Apply', both enclosed in a light green rounded rectangle. The footer of the page displays the generation time 'Page generated: Jul 18, 2019 10:35:32 PM PKT', a 'REST API' link, and the Jenkins version 'Jenkins ver. 2.176.1'.

Enter the below script in the Script section

```
pipeline {
    agent any
    stages {
        stage('Build') {
            steps {
                echo 'Build'
            }
        }
        stage ('Test') {
            steps {
                echo 'Test'
            }
        }
    }
}
```

Click Save



The screenshot shows the Jenkins Pipeline configuration page for a job named "first-pipeline-job". The "Pipeline" tab is selected. The "Definition" dropdown is set to "Pipeline script". The "Script" area contains the following Groovy code:

```
1 pipeline {  
2     agent any  
3     stages {  
4         stage('Build') {  
5             steps {  
6                 echo 'Build'  
7             }  
8         }  
9         stage ('Test') {  
10            steps {  
11                echo 'Test'  
12            }  
13        }  
14    }  
15 }
```

A "try sample Pipeline..." button is visible next to the script editor. A checkbox labeled "Use Groovy Sandbox" is checked. Below the script editor, there are "Pipeline Syntax" and "Pipeline Help" links. At the bottom, there are "Save" and "Apply" buttons.

Page generated: Jul 18, 2019 10:35:32 PM PKT [REST API](#) [Jenkins ver. 2.176.1](#)

Click Build to run the Pipeline Job

Jenkins

first-pipeline-job

2

search

user | log out

ENABLE AUTO REFRESH

Back to Dashboard

Status

Changes

Build Now

Delete Pipeline

Configure

Full Stage View

Rename

Pipeline Syntax

Recent Changes

Stage View

No data available. This Pipeline has not yet run.

Build History

trend

find

RSS for all RSS for failures

Permalinks

localhost:8080/job/first-pipeline-job/build?delay=0sec

Page generated: Jul 18, 2019 10:38:02 PM PKT REST API Jenkins ver. 2.176.1

The screenshot shows the Jenkins interface for a pipeline job named "Pipeline first-pipeline-job". The top navigation bar includes the Jenkins logo, user information, and a search bar. The left sidebar contains links for managing the project, such as "Back to Dashboard", "Status", "Changes", "Build Now", "Delete Pipeline", "Configure", "Full Stage View", "Rename", and "Pipeline Syntax". The main content area features the pipeline's name, a "Recent Changes" section with a notebook icon, and a "Stage View" section which displays a message stating "No data available. This Pipeline has not yet run." Below this, there is a "Build History" section with a search bar and buttons for "RSS for all" and "RSS for failures". At the bottom of the page, there are links for "Permalinks" and the Jenkins URL ("localhost:8080/job/first-pipeline-job/build?delay=0sec"). The footer provides page generation details and links to the REST API and Jenkins version.

You can see the Build and Test stages in the screenshot below

Jenkins

first-pipeline-job

2

search

user | log out

ENABLE AUTO REFRESH

Back to Dashboard

Status

Changes

Build Now

Delete Pipeline

Configure

Full Stage View

Rename

Pipeline Syntax

Build History trend —

find

#1 Jul 18, 2019 10:38 PM

RSS for all RSS for failures

Pipeline first-pipeline-job

First Pipeline Job

Recent Changes

edit description

Disable Project

Stage View

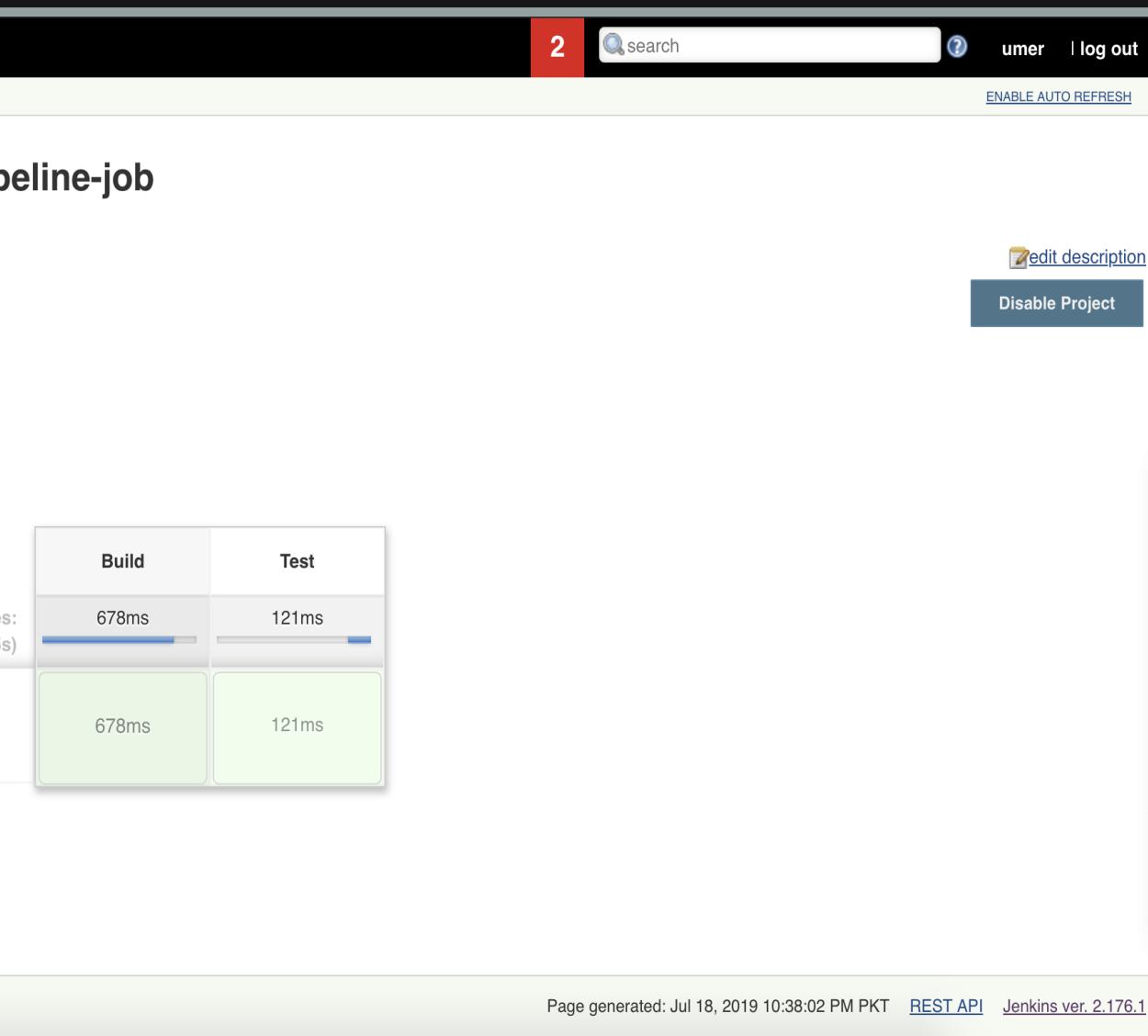
Average stage times:
(Average full run time: ~5s)

Build	Test
678ms	121ms
678ms	121ms

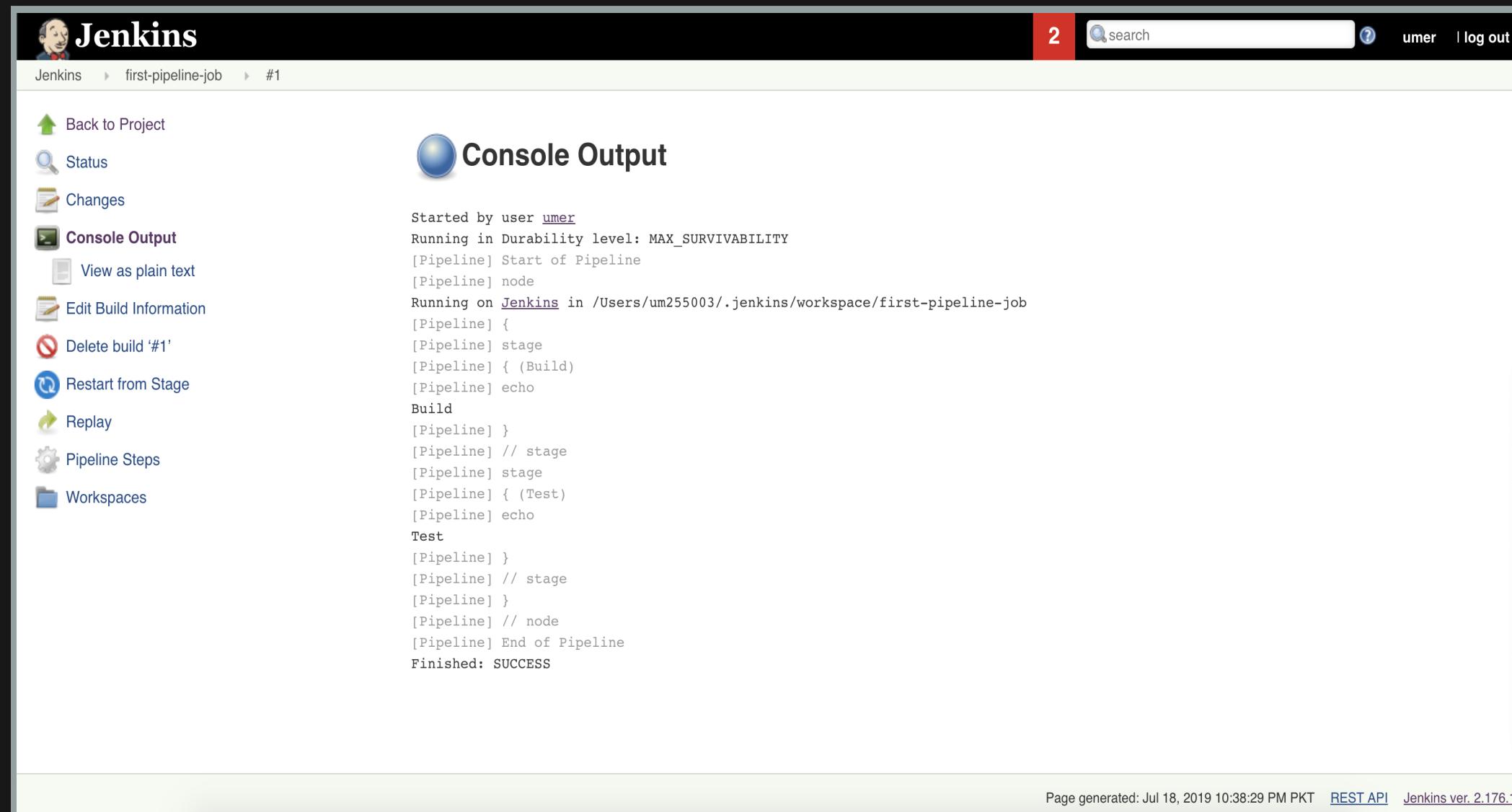
#1 Jul 18 22:38 No Changes

Permalinks

Page generated: Jul 18, 2019 10:38:02 PM PKT REST API Jenkins ver. 2.176.1



Check Console Output, you should see the logs as shown below



The screenshot shows the Jenkins interface for a pipeline job. The left sidebar has a 'Console Output' section selected. The main content area is titled 'Console Output' and displays the following log output:

```
Started by user umer
Running in Durability level: MAX_SURVIVABILITY
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /Users/um255003/.jenkins/workspace/first-pipeline-job
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Build)
[Pipeline] echo
Build
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Test)
[Pipeline] echo
Test
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

At the bottom of the page, there is a footer bar with the text "Page generated: Jul 18, 2019 10:38:29 PM PKT REST API Jenkins ver. 2.176.1".

JENKINSFILE

- File that contains the definition of Jenkins Pipeline and checked into source control
- Pipeline as code
- Benefits
 - Version control
 - Code review on the Pipeline
 - Audit trail for the Pipeline
 - Single source of truth for the Pipeline

LAB - JENKINSFILE

In this Lab, you are going to create a Jenkinsfile, push to the GitHub repository and then create a Jenkins job to run the steps defined in the Jenkinsfile. First create a Jenkinsfile at the root of repository that was forked from DiceLabs with the following script. Name of the file should be Jenkinsfile. Push this file to the remote repository.

```
pipeline {
    agent any
    stages {
        stage('Build') {
            steps {
                echo 'Build'
            }
        }
        stage ('Test') {
            steps {
                echo 'Test'
            }
        }
    }
}
```

Go to the Jenkins landing page. Click New Item

The screenshot shows the Jenkins web interface. On the left, a sidebar lists various navigation options: New Item, People, Build History, Manage Jenkins, My Views, Lockable Resources, Credentials, and New View. Below these are two expandable sections: Build Queue (showing 'No builds in the queue.') and Build Executor Status (showing '1 Idle' and '2 Idle'). In the center, there is a search bar and a user menu. A table displays a single job entry:

S	W	Name ↓	Last Success	Last Failure	Last Duration
		first-job	15 min - #3	N/A	56 ms

Below the table, there is a legend and links for RSS feeds: RSS for all, RSS for failures, and RSS for just latest builds. At the bottom, a footer indicates the page was generated on July 14, 2019, at 2:22:31 AM PKT, and shows the REST API and Jenkins version information.

Page generated: Jul 14, 2019 2:22:31 AM PKT [REST API](#) Jenkins ver. 2.176.1

Enter job details, select Pipeline and Click OK

The screenshot shows the Jenkins web interface for creating a new item. At the top, there is a navigation bar with the Jenkins logo, a search bar, and user information. Below the navigation bar, the main content area has a title "Enter an item name" and a required input field containing "first-jenkinsfile-job". Below this, there is a list of project types:

- Freestyle project**: Described as the central feature of Jenkins, combining any SCM with any build system.
- Pipeline**: Described as orchestrating long-running activities across multiple build agents, suitable for building pipelines and organizing complex activities.
- Multi-configuration project**: Suitable for projects with many configurations, such as testing on multiple environments.
- Folder**: Creates a container for grouping items together.
- GitHub Organization**: Scans a GitHub organization for repositories matching defined markers.
- Multibranch Pipeline**: Creates a set of Pipeline projects based on detected branches in one SCM repository.

A large blue "OK" button is visible at the bottom left of the list.

Enter description and click Build Triggers

Jenkins first-jenkinsfile-job

General Build Triggers Advanced Project Options Pipeline

Description My First Jenkinsfile Job

[Plain text] [Preview](#)

Discard old builds [?](#)

Do not allow concurrent builds [?](#)

Do not allow the pipeline to resume if the master restarts [?](#)

GitHub project [?](#)

Pipeline speed/durability override [?](#)

Preserve stashes from completed builds [?](#)

This project is parameterized [?](#)

Throttle builds [?](#)

Build Triggers

Build after other projects are built [?](#)

Build periodically [?](#)

GitHub hook trigger for GITScm polling [?](#)

Poll SCM [?](#)

Disable this project [?](#)

[Save](#) [Apply](#) e.g., from scripts)

Select Poll SCM and enter schedule as shown below. Next click Pipeline

The screenshot shows the Jenkins configuration interface for a project named "first-jenkinsfile-job". The "Build Triggers" tab is selected. Under the "Build Triggers" section, the "Poll SCM" option is checked, and the schedule is set to "H/5 * * * *". A note below the schedule indicates the last run was at Friday, July 19, 2019 12:19:35 AM PKT, and the next run is scheduled for Friday, July 19, 2019 12:24:35 AM PKT. Other trigger options like "Build after other projects are built" and "GitHub hook trigger for GITScm polling" are also listed. The "Advanced Project Options" section contains an "Advanced..." button. At the bottom, there are "Save" and "Apply" buttons.

Jenkins > first-jenkinsfile-job >

General Build Triggers Advanced Project Options Pipeline

Build Triggers

Build after other projects are built ?
 Build periodically ?
 GitHub hook trigger for GITScm polling ?
 Poll SCM ?

Schedule **H/5 * * * ***

Would last have run at Friday, July 19, 2019 12:19:35 AM PKT; would next run at Friday, July 19, 2019 12:24:35 AM PKT.

Ignore post-commit hooks ?
 Disable this project ?
 Quiet period ?
 Trigger builds remotely (e.g., from scripts) ?

Advanced Project Options

Advanced...

Save Apply

Select 'Pipeline script from SCM'

The screenshot shows the Jenkins Pipeline configuration page for a job named "first-jenkinsfile-job". The "Pipeline" tab is selected in the top navigation bar. Under the "Definition" section, a dropdown menu is open, showing two options: "Pipeline script" (which is checked) and "Pipeline script from SCM". Below the dropdown, there is a "Script" input field containing the number "1" and a "try sample Pipeline..." button. A checkbox labeled "Use Groovy Sandbox" is checked. At the bottom of the form are "Save" and "Apply" buttons. The footer of the page displays the generation time "Page generated: Jul 19, 2019 12:21:25 AM PKT", the REST API link, and the Jenkins version "Jenkins ver. 2.176.1".

In the SCM, select Git

The screenshot shows the Jenkins Pipeline configuration page for a job named "first-jenkinsfile-job". The "Advanced Project Options" tab is selected. In the "Pipeline" section, under "Definition", the "Pipeline script from SCM" dropdown is set to "Pipeline script from SCM". The "SCM" dropdown menu is open, showing options: "None" (selected), "Git" (highlighted with a blue selection bar), and "Subversion". The "Script Path" field is set to "Jenkinsfile". The "Lightweight checkout" checkbox is checked. At the bottom left, there are "Save" and "Apply" buttons. At the bottom right, the footer displays: "Page generated: Jul 19, 2019 12:21:25 AM PKT REST API Jenkins ver. 2.176.1".

Enter the URL of repository that you forked from DiceLabs. Click Save

The screenshot shows the Jenkins job configuration interface for a job named "first-jenkinsfile-job". The "Advanced Project Options" tab is selected. The "Definition" is set to "Pipeline script from SCM". The "SCM" provider is "Git". Under "Repositories", there is one entry with the "Repository URL" set to "https://github.com/umermunir/". Under "Branches to build", there is one entry with the "Branch Specifier" set to "*/*master". The "Repository browser" is set to "(Auto)". Under "Additional Behaviours", there is an "Add" button. The "Script Path" is set to "Jenkinsfile". The "Lightweight checkout" checkbox is checked. At the bottom left, there are "Save" and "Apply" buttons, and a link to "Pipeline Syntax".

Jenkins first-jenkinsfile-job

General Build Triggers Advanced Project Options Pipeline

Definition Pipeline script from SCM

SCM Git

Repositories

Repository URL https://github.com/umermunir/

Credentials - none - Add Advanced... Add Repository

Branches to build

Branch Specifier (blank for 'any') */master

Add Branch

Repository browser (Auto)

Additional Behaviours Add

Script Path Jenkinsfile

Lightweight checkout

Pipeline Syntax

Save Apply

Next create a new file and push to remote repository and wait for the job to be triggered

The screenshot shows the Jenkins interface for a pipeline named "Pipeline first-jenkinsfile-job". The pipeline has one stage, "Declarative: Checkout SCM", which took 7s. The "Build" stage took 163ms and the "Test" stage took 102ms. The pipeline was last run on Jul 19, 2019 at 12:29 AM.

Pipeline first-jenkinsfile-job

My First Jenkinsfile Job

Recent Changes

Stage View

Declarative: Checkout SCM	Build	Test
7s	163ms	102ms

Average stage times:
(Average full run time: ~12s)

#1 Jul 19, 2019 12:29 AM No Changes

Declarative: Checkout SCM Build Test

7s 163ms 102ms

Permalinks

- [Last build \(#1\), 14 sec ago](#)
- [Last stable build \(#1\), 14 sec ago](#)
- [Last successful build \(#1\), 14 sec ago](#)
- [Last completed build \(#1\), 14 sec ago](#)

Check Console Output

Jenkins > first-jenkinsfile-job > #1

Workspaces

```
> git config remote.origin.url https://github.com/umermunir/DiceLabs.git # timeout=10
Fetching upstream changes from https://github.com/umermunir/DiceLabs.git
> git fetch --tags --progress https://github.com/umermunir/DiceLabs.git +refs/heads/*:refs/remotes/origin/*
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
> git rev-parse refs/remotes/origin/origin/master^{commit} # timeout=10
Checking out Revision 7496ac5ec6c3bc4fa6d33d38502ba83a9db0fe8c (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f 7496ac5ec6c3bc4fa6d33d38502ba83a9db0fe8c
Commit message: "test jenkins"
First time build. Skipping changelog.
[Pipeline]
[Pipeline] // stage
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Build)
[Pipeline] echo
Build
[Pipeline]
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Test)
[Pipeline] echo
Test
[Pipeline]
[Pipeline] // stage
[Pipeline] }
[Pipeline] // withEnv
[Pipeline]
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Page generated: Jul 19, 2019 12:29:41 AM PKT [REST API](#) [Jenkins ver. 2.176.1](#)

CONTINUOUS INTEGRATION

- Merge code changes from different developers into a central repository
- Automated builds and tests are run
- Key goals are to fail fast and find and address bugs quicker
- Benefits Developers most, Less merge conflicts

LAB - CONTINOUS INTERGRATION

In this Lab, you are going to fork a simple java application, clone it, create a jenkins file in that, push to remote repository and then create a CI job. First create a new job

Jenkins 2 search user log out

Enter an item name

ci-job

» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

GitHub Organization
Scans a GitHub organization (or user account) for all repositories matching some defined markers.

Multibranch Pipeline
Creates a set of Pipeline projects according to detected branches in one SCM repository.

Enter description and Click Build Triggers

The screenshot shows the Jenkins configuration interface for a job named "ci-job". The top navigation bar includes the Jenkins logo, the job name "ci-job", a user icon, and a "log out" link. A red notification badge with the number "2" is visible in the top right corner.

The main content area has a header with tabs: "General" (selected), "Build Triggers", "Advanced Project Options", and "Pipeline".

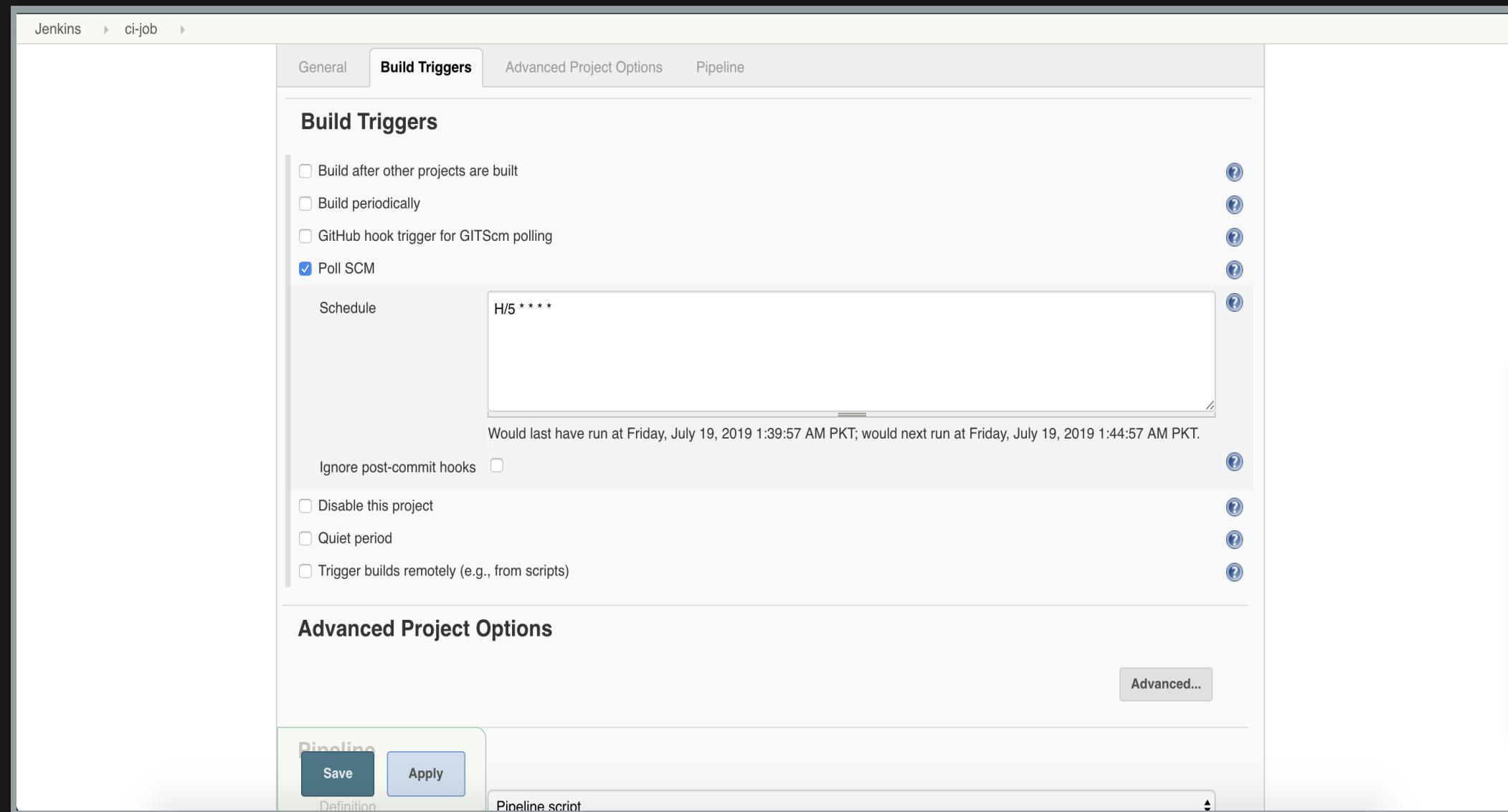
General Tab:

- Description:** First Continuous Integration Job
- Buttons:** [Plain text] [Preview](#)
- Checkboxes:**
 - Discard old builds
 - Do not allow concurrent builds
 - Do not allow the pipeline to resume if the master restarts
 - GitHub project
 - Pipeline speed/durability override
 - Preserve stashes from completed builds
 - This project is parameterized
 - Throttle builds
- Help Links:** Each checkbox has a question mark icon to its right.

Build Triggers Tab:

- Checkboxes:**
 - Build after other projects are built
 - Build periodically
- Buttons:** Save, Apply, SCM polling
- Help Links:** Each checkbox has a question mark icon to its right.

Select Poll SCM, specify the schedule and Click Pipeline



The screenshot shows the Jenkins configuration interface for a project named "ci-job". The "Build Triggers" tab is selected. Under "Build Triggers", the "Poll SCM" option is checked, and the schedule is set to "H/5 * * * *". A note below the schedule indicates the last run was at Friday, July 19, 2019 1:39:57 AM PKT, and the next run is scheduled for Friday, July 19, 2019 1:44:57 AM PKT. Other trigger options like "Build after other projects are built", "Build periodically", and "GitHub hook trigger for GITScm polling" are available but not selected. In the "Advanced Project Options" section, there is a "Pipeline" tab selected, showing a "Pipeline script" field which is currently empty. Below the pipeline script field are "Save" and "Apply" buttons. There is also an "Advanced..." button for more options.

Select 'Pipeline script from SCM'

The screenshot shows the Jenkins Pipeline configuration page for a job named "ci-job". The "Pipeline" tab is selected. In the "Definition" section, a dropdown menu is open, showing two options: "Pipeline script" and "Pipeline script from SCM". The "Pipeline script from SCM" option is highlighted with a blue selection bar. Below the dropdown, there is a "Script" input field containing the number "1" and a "try sample Pipeline..." button. A checkbox labeled "Use Groovy Sandbox" is checked. At the bottom of the configuration area, there are "Save" and "Apply" buttons.

Jenkins > ci-job >

General Build Triggers Advanced Project Options Pipeline

Pipeline

Definition

✓ Pipeline script
Pipeline script from SCM

Script 1 try sample Pipeline... ?

Use Groovy Sandbox ?

[Pipeline Syntax](#)

Save Apply

Page generated: Jul 19, 2019 1:40:22 AM PKT [REST API](#) [Jenkins ver. 2.176.1](#)

In the SCM, select Git

The screenshot shows the Jenkins Pipeline configuration page for a job named "ci-job". The "Advanced Project Options" tab is selected. In the "Pipeline" section, under "Definition", the "Pipeline script from SCM" dropdown is open, showing three options: "None", "Git", and "Subversion". The "Git" option is highlighted with a blue selection bar. Other fields in the pipeline section include "Script Path" set to "Jenkinsfile" and "Lightweight checkout" checked. Below the pipeline section is a "Pipeline Syntax" section with "Save" and "Apply" buttons. At the bottom of the page, the footer displays "Page generated: Jul 19, 2019 1:40:22 AM PKT REST API Jenkins ver. 2.176.1".

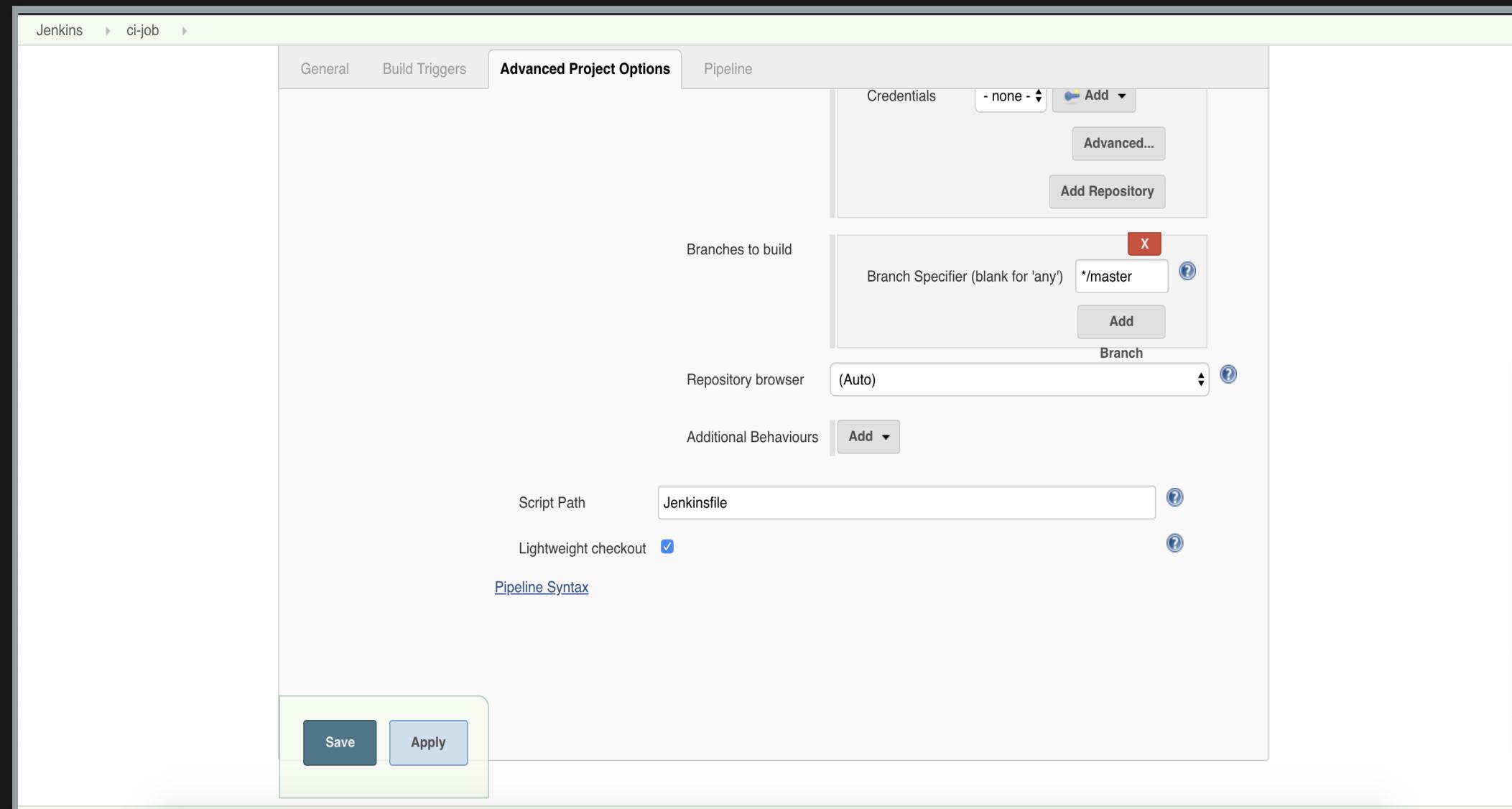
Fork the [DICE Java Repo](#), clone it on your system. Create a Jenkinsfile in the root folder of the cloned repository and push it to remote repository

```
pipeline {
    agent {
        docker {
            image 'maven:3-alpine'
        }
    }
    stages {
        stage('Build') {
            steps {
                sh 'mvn -B -DskipTests clean package'
            }
        }
        stage('Test') {
            steps {
                sh 'java -jar target/hello-world-1.jar'
            }
        }
    }
}
```

Copy the URL of forked repository as shown below

The screenshot shows a GitHub repository page for the user 'umermunir' with the repository name 'dice-hello-world-java'. The page includes a navigation bar with links for Pull requests, Issues, Marketplace, and Explore. Below the navigation bar, there are buttons for Watch (0), Star (0), and Fork (0). The main content area displays the repository's details: No description, website, or topics provided. It shows 4 commits, 1 branch, 0 releases, and 1 contributor. A 'Clone or download' button is highlighted in green. A dropdown menu is open, showing options for 'Clone with HTTPS' (with a copy icon) and 'Use SSH'. The HTTPS URL is displayed as <https://github.com/umermunir/dice-hel>. Other options in the dropdown include 'Open in Desktop' and 'Download ZIP'. The repository's files listed on the left include Jenkinsfile, README.md, and pom.xml.

Enter the Repository URL in the Pipeline section and Click Save



Now create a new file in the forked repository and push it to remote repository that should trigger the build

Jenkins 2 user log out search ENABLE AUTO REFRESH

Back to Dashboard Status Changes Build Now Delete Pipeline Configure Full Stage View Rename Pipeline Syntax Git Polling Log

Pipeline ci-job

First Continuous Integration Job

Recent Changes

Stage View

Average stage times:
(Average full run time: ~3min 59s)

Declarative: Checkout SCM	Build	Test
3s	3min 45s	674ms
3s	17s	939ms
4s	7min 14s	409ms

#2 Jul 19, 2019 1:54 AM 1 commit
#1 Jul 19, 2019 1:42 AM No Changes

RSS for all RSS for failures

Permalinks

edit description Disable Project

The screenshot shows the Jenkins Pipeline ci-job dashboard. On the left, there's a sidebar with various project management and configuration links. The main area features a title 'Pipeline ci-job' and a subtitle 'First Continuous Integration Job'. Below the title is a 'Recent Changes' section with a note about average stage times. A 'Stage View' section displays a grid of execution times for different stages. To the right of the stage view is a table comparing declarative checkout, build, and test times. At the bottom, there are two recent build entries: one from July 19 at 1:54 AM with one commit, and another from July 19 at 1:42 AM with no changes. At the very bottom, there are RSS feed links for all and for failures.

Check Console Output

Jenkins > ci-job > #2

```
[INFO]
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ hello-world ---
[INFO] Tests are skipped.
[INFO]
[INFO] --- maven-jar-plugin:3.0.2:jar (default-jar) @ hello-world ---
[INFO] Building jar: /Users/um255003/.jenkins/workspace/ci-job/target/hello-world-1.jar
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 13.268 s
[INFO] Finished at: 2019-07-18T20:54:35Z
[INFO] Final Memory: 16M/151M
[INFO] -----
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Test)
[Pipeline] sh
+ java -jar target/hello-world-1.jar
Hello World!
[Pipeline] }
[Pipeline] // stage
[Pipeline] }
$ docker stop --time=1 3ebe94b02e8c1653703878f881d4003e8ebac957ef4ad38d85bc6a4cc96cce06
$ docker rm -f 3ebe94b02e8c1653703878f881d4003e8ebac957ef4ad38d85bc6a4cc96cce06
[Pipeline] // withDockerContainer
[Pipeline] }
[Pipeline] // withEnv
[Pipeline] }
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS
```

Page generated: Jul 19, 2019 1:55:52 AM PKT [REST API](#) [Jenkins ver. 2.176.1](#)

CONTINUOUS INSPECTION

- Can be used for Continuous Inspection
 - Static Code Analysis
 - Measure Code Quality
 - Identify non-compliant code
 - Fix code quality issue
 - Used to generate reports
- Plugins
 - Warnings Next Generation
 - SonarQube

CONTINUOUS DELIVERY

- CI stage is approved
- A small build cycle for short sprints for releasing small features
- Code changes are automatically built & tested
- Can be deployed to a test environment
- Can use branching strategy (other than master)
- Mindset to always have a deployment-ready build artifact.

LAB - CONTINUOUS DELIVERY

- Docker
- Python

DOCKER

In this Lab, you are going to fork a repository, clone it, create a jenkins file, push to remote repository and then create CI/CD job. You will use the jenkins pipeline to create docker image, test it and then push to docker hub. First you are going to add dockerhub by clicking Credentials

The screenshot shows the Jenkins dashboard with a single job listed:

S	W	Name ↓	Last Success	Last Failure	Last Duration
		first-job	15 min - #3	N/A	56 ms

Job details:

- Icon: [S](#) [M](#) [L](#)
- Legend: [RSS for all](#) [RSS for failures](#) [RSS for just latest builds](#)

Build Queue:

No builds in the queue.

Build Executor Status:

1 Idle
2 Idle

Page generated: Jul 14, 2019 2:22:31 AM PKT [REST API](#) [Jenkins ver. 2.176.1](#)

Click System

Jenkins

2 search user log out

Jenkins > Credentials

New Item People Build History Project Relationship Check File Fingerprint Manage Jenkins My Views Lockable Resources Credentials System New View

Credentials

T P Store ↓ Domain ID Name

T	P	Store ↓	Domain	ID	Name
⭐	🏡	Jenkins	(global)	dockerhub	umermunirrr/***** (dockerhub)

Icon: S M L

Stores scoped to Jenkins

P Store ↓ Domains

P	Store ↓	Domains
🏡	Jenkins	!(global)

Build Queue

No builds in the queue.

Build Executor Status

1 Idle
2 ci-cd-job #2 (Publish)

Click Global credentials

Screenshot of the Jenkins System configuration page.

The left sidebar shows navigation links:

- New Item
- People
- Build History
- Project Relationship
- Check File Fingerprint
- Manage Jenkins
- My Views
- Lockable Resources
- Credentials
- System
 - Add domain
- New View

The main content area is titled "System". It displays a table with one row:

Domain	Description
 Global credentials (unrestricted) ▾	Credentials that should be available irrespective of domain specification to requirements matching.

Below the table, it says "Icon: S M L".

At the bottom of the page, the URL is shown as "localhost:8080/credentials/store/system/domain/_".

Click Add Credentials

Screenshot of the Jenkins Global credentials (unrestricted) page.

The page title is "Global credentials (unrestricted)".

The table shows one credential entry:

	Name	Kind	Description
	umermunirr/***** (dockerhub)	Username with password	dockerhub 

Icon: [S](#) [M](#) [L](#)

Page generated: Jul 19, 2019 3:31:30 AM PKT [REST API](#) [Jenkins ver. 2.176.1](#)

Specify your dockerhub username, password and credential id as shown below. You will be using this credential id in Jenkins file

The screenshot shows the Jenkins Global credentials (unrestricted) configuration page. The URL is `http://Jenkins:8080/credentials/store/system/domain/_/umermunirrr/***** (dockerhub)`. The page displays a single credential entry for DockerHub:

- Scope:** Global (Jenkins, nodes, items, all child items, etc)
- Username:** umermunirrr
- Password:** (redacted)
- ID:** dockerhub
- Description:** dockerhub

On the left sidebar, there are links for Back to Global credentials (unrestricted), Update, Delete, and Move. A "Save" button is located at the bottom left of the form.

At the bottom right of the page, the footer text reads: "Page generated: Jul 19, 2019 3:31:42 AM PKT REST API Jenkins ver. 2.176.1".

Go to landing page, click New Item and Enter job details

The screenshot shows the Jenkins web interface. At the top, there is a navigation bar with the Jenkins logo, a user icon, and a search bar. A red notification badge with the number '2' is visible. Below the navigation bar, the URL 'Jenkins > All' is shown. The main content area has a light gray background and contains a form for creating a new item. The form has a title 'Enter an item name' and a required input field containing the text 'ci-cd-job'. Below the input field, a note says '» Required field'. There are several options listed as cards:

- Freestyle project**: This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.
- Pipeline**: Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
- Multi-configuration project**: Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
- Folder**: Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.
- GitHub Organization**: Scans a GitHub organization (or user account) for all repositories matching some defined markers.
- Multibranch Pipeline**: Creates a set of Pipeline projects according to detected branches in one SCM repository.

At the bottom left of the card list, there is a blue 'OK' button.

Enter job description and Click Build Triggers

The screenshot shows the Jenkins General configuration page for a job named "ci-cd-job". The "General" tab is selected. The "Description" field contains the text "My First CI/CD Job". Below the description are several configuration options, each with a help icon (blue question mark) to its right:

- Discard old builds
- Do not allow concurrent builds
- Do not allow the pipeline to resume if the master restarts
- GitHub project
- Pipeline speed/durability override
- Preserve stashes from completed builds
- This project is parameterized
- Throttle builds

Below these options is a section titled "Build Triggers" with the following triggers listed:

- Build after other projects are built
- Build periodically
 - Scm polling
 - Scm polling
 - Scm polling
 - Scm polling

At the bottom of the page are two buttons: "Save" and "Apply".

Select Poll SCM and specify schedule as shown below

The screenshot shows the Jenkins job configuration interface for a project named "ci-cc-job". The "Build Triggers" tab is selected. Under the "Build Triggers" section, the "Poll SCM" option is checked. The "Schedule" field contains the cron expression "H/5 * * * *". A note below the schedule states: "Would last have run at Friday, July 19, 2019 3:24:41 AM PKT; would next run at Friday, July 19, 2019 3:29:41 AM PKT." Below the schedule, there are several other trigger options: "Build after other projects are built", "Build periodically", "GitHub hook trigger for GITScm polling", "Ignore post-commit hooks" (unchecked), "Disable this project" (unchecked), "Quiet period" (unchecked), and "Trigger builds remotely (e.g., from scripts)" (unchecked). At the bottom of the page, there is an "Advanced..." button, a "Pipeline" tab, and a "Save" button.

Select Pipeline

Jenkins > ci-cd-job >

General Build Triggers Advanced Project Options **Pipeline**

Pipeline

Definition Pipeline script

Script 1 try sample Pipeline... ?

Use Groovy Sandbox ?

[Pipeline Syntax](#)

Save **Apply**

Page generated: Jul 19, 2019 3:25:49 AM PKT [REST API](#) Jenkins ver. 2.176.1

The screenshot shows the Jenkins Pipeline configuration interface for a job named 'ci-cd-job'. The 'Pipeline' tab is active. In the 'Definition' section, there is a 'Pipeline script' input field containing a single character '1'. To the right of the input field is a 'try sample Pipeline...' button with a question mark icon. Below the input field is a 'Script' editor window. Underneath the editor, there is a checked checkbox labeled 'Use Groovy Sandbox' with a question mark icon next to it. A link to 'Pipeline Syntax' is also present. At the bottom of the configuration panel are two buttons: 'Save' and 'Apply'.

In the Definition select 'Pipeline script from SCM' and select 'Git' in SCM

The screenshot shows the Jenkins Pipeline configuration page for a job named "ci-cd-job". The "Pipeline" tab is selected. In the "Definition" section, a dropdown menu is open, showing two options: "Pipeline script" (which has a checked checkbox) and "Pipeline script from SCM". The "Pipeline script from SCM" option is highlighted with a blue selection bar. Below the dropdown, there is a "Script" input field containing the number "1" and a "try sample Pipeline..." button. A "Use Groovy Sandbox" checkbox is checked. There is also a link to "Pipeline Syntax". At the bottom of the configuration area, there are "Save" and "Apply" buttons.

Fork DICE Docker Img repository, clone it, create a Jenkins file below and then push it.

```
pipeline {
    agent any
    environment {
        registryCredential = 'dockerhub'
    }
    stages {
        stage('Build') {
            steps {
                sh 'docker build -t umermunirrr/test-node-app .'
            }
        }
        stage('Test') {
            steps {
                sh 'docker container rm -f node || true'
                sh 'docker container run -p 8001:8080 --name node -d umermunirrr/test-node-app'
                sh 'curl -I http://localhost:8001'
            }
        }
        stage('Publish') {
            steps{
                script {
                    docker.withRegistry( '', registryCredential ) {
                        sh 'docker push umermunirrr/test-node-app:latest'
                    }
                }
            }
        }
    }
}
```

Copy the URL of forked repository, specify that in the Repository URL and click Save

The screenshot shows the Jenkins Pipeline configuration page for a job named "ci-cc-job". The "Advanced Project Options" tab is selected. Under the "Pipeline" section, the "Definition" is set to "Pipeline script from SCM". The "SCM" provider is "Git". In the "Repositories" section, there is one repository defined with the URL "munir/dice-hello-world-nodejs.git". A red error message "Please enter Git repository." is displayed next to the URL field. The "Branches to build" section shows a branch specifier of "*/*master". The "Repository browser" is set to "(Auto)". The "Additional Behaviours" section has an "Add" button. At the bottom left, there are "Save" and "Apply" buttons.

Jenkins > ci-cc-job >

General Build Triggers Advanced Project Options Pipeline

Pipeline

Definition Pipeline script from SCM

SCM Git

Repositories

Repository URL munir/dice-hello-world-nodejs.git Please enter Git repository.

Credentials - none - Add

Advanced... Add Repository

Branches to build

Branch Specifier (blank for 'any') */master

Add Branch

Repository browser (Auto)

Additional Behaviours Add

Save Apply

Create a new file in the forked repository and wait for it to trigger the build

The screenshot shows the Jenkins interface for a pipeline job named "Pipeline ci-cd-job". The top navigation bar includes the Jenkins logo, a user icon, a search bar, and links for "2", "Help", "User", "Log Out", and "Enable Auto Refresh". The left sidebar contains links for "Back to Dashboard", "Status", "Changes", "Build Now", "Delete Pipeline", "Configure", "Full Stage View", "Rename", "Pipeline Syntax", and "Polling Log". The main content area displays the job's name, "Pipeline ci-cd-job", and its description, "My First CI/CD Job". It features a "Recent Changes" section with a notebook icon and a "Stage View" section with a message: "No data available. This Pipeline has not yet run." A "Permalinks" section at the bottom includes a search bar with "find" and RSS feed links for "RSS for all" and "RSS for failures". The footer indicates the page was generated on Jul 19, 2019, at 3:28:26 AM PKT, and shows the REST API and Jenkins version information.

Jenkins

2

search

User | log out

ENABLE AUTO REFRESH

Back to Dashboard

Status

Changes

Build Now

Delete Pipeline

Configure

Full Stage View

Rename

Pipeline Syntax

Polling Log

Recent Changes

Stage View

No data available. This Pipeline has not yet run.

Build History

trend

find

RSS for all RSS for failures

Permalinks

Page generated: Jul 19, 2019 3:28:26 AM PKT REST API Jenkins ver. 2.176.1

 Jenkins

Jenkins ci-cd-job

2 search umer | log out

ENABLE AUTO REFRESH

Back to Dashboard Status Changes Build Now Delete Pipeline Configure Full Stage View Rename Pipeline Syntax Git Polling Log

Recent Changes

Pipeline ci-cd-job

My First CI/CD Job

[Edit description](#) [Disable Project](#)

Stage View

Average stage times:
(Average full run time: ~1min 44s)

	Build	Test	Publish
#2	1s	2s	46s
#1	4s	877ms	1min 33s
	3s	2s	2s failed
			95ms failed

Successive: Checkout SCM Logs

find #2 Jul 19, 2019 3:29 AM #1 Jul 19, 2019 3:28 AM RSS for all RSS for failures

Permalinks

Check Console Output

```
Jenkins ▶ ci-cd-job ▶ #2

15210a41d4ee: Waiting
e2a8a00a83b2: Waiting
7bff69ee8682: Pushed
76f4ec530ela: Pushed
452bed668936: Pushed
a0822d54fb0d: Pushed
f8313885dabb: Pushed
787062cd94fb: Mounted from library/node
b27cb3611f76: Mounted from library/node
29fbe5747f9a: Mounted from library/node
533be996568a: Mounted from library/node
a4e797bc3f15: Mounted from library/node
392f356944ff: Mounted from library/node
15210a41d4ee: Mounted from library/node
e2a8a00a83b2: Mounted from library/node
latest: digest: sha256:88382b2cef5a656abed73f8bdb2d499f87cd5300e743e2f4949fd43e93b30da6 size: 3049
[Pipeline]
[Pipeline] // withDockerRegistry
[Pipeline]
[Pipeline] // withEnv
[Pipeline]
[Pipeline] // script
[Pipeline]
[Pipeline] // stage
[Pipeline]
[Pipeline] // withEnv
[Pipeline]
[Pipeline] // withEnv
[Pipeline]
[Pipeline] // node
[Pipeline] End of Pipeline
Finished: SUCCESS

Page generated: Jul 19, 2019 3:32:03 AM PKT REST API Jenkins ver. 2.176.1
```

PYTHON

In this Lab, you are going to fork a repository, clone it, create a jenkins file, push to remote repository and then create CI/CD job. You will use the jenkins pipeline to compile python script and then create artifact. First Create a New Job

The screenshot shows the Jenkins dashboard with the following elements:

- Header:** Includes the Jenkins logo, a red notification badge with the number "2", a search bar, and user information.
- Left Sidebar:** A vertical menu with icons and links:
 - New Item
 - People
 - Build History
 - Project Relationship
 - Check File Fingerprint
 - Manage Jenkins
 - My Views
 - Lockable Resources
 - Credentials
 - New View
- Welcome Message:** "Welcome to Jenkins!" with a sub-instruction: "Please [create new jobs](#) to get started."
- Build Queue:** A section titled "Build Queue" with the message "No builds in the queue."
- Build Executor Status:** A section titled "Build Executor Status" showing "1 Idle" and "2 Idle".
- Page Footer:** Includes the URL "localhost:8080/newJob", page generation time "Page generated: Jul 19, 2019 11:23:15 AM PKT", and links to "REST API" and "Jenkins ver. 2.176.1".

Enter job name, select Pipeline and Click OK as shown below

The screenshot shows the Jenkins interface for creating a new job. In the top left, the Jenkins logo and the word "Jenkins" are visible. The top right features a user icon with the number "2", a search bar, and links for "Help", "User", and "log out". The main area has a light gray background with a white input field. The input field contains the text "ci-cd-python" and has a blue border, indicating it is a required field. Below the input field, the text "» Required field" is displayed in a smaller font. A list of project types is shown in cards:

- Freestyle project**: Represented by a blue icon of a house-like build box. Description: "This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build."
- Pipeline**: Represented by a blue icon of a hand holding a gear. Description: "Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type."
- Multi-configuration project**: Represented by a blue icon of a document with a gear. Description: "Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc."
- Folder**: Represented by a blue icon of a folder. Description: "Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders."
- GitHub Organization**: Represented by a blue icon of the GitHub logo. Description: "Scans a GitHub organization (or user account) for all repositories matching some defined markers."
- Multibranch Pipeline**: Represented by a blue icon of a branch. Description: "Creates a set of Pipeline projects according to detected branches in one SCM repository."

A blue button labeled "OK" is located at the bottom left of the card for "Multibranch Pipeline".

Enter job description and Click Build Triggers

The screenshot shows the Jenkins General configuration page for a job named "ci-cd-python". The "General" tab is selected. In the "Description" field, the text "My First Python CI/CD" is entered. Below the description, there are several build-related checkboxes:

- Discard old builds
- Do not allow concurrent builds
- Do not allow the pipeline to resume if the master restarts
- GitHub project
- Pipeline speed/durability override
- Preserve stashes from completed builds
- This project is parameterized
- Throttle builds

Below these checkboxes is a section titled "Build Triggers" containing two options:

- Build after other projects are built
- Build periodically

At the bottom of the configuration page are two buttons: "Save" and "Apply". The "Apply" button is highlighted with a blue border.

Select Poll SCM, specify schedule as shown below

The screenshot shows the Jenkins project configuration interface for a project named "ci-cd-python". The "Build Triggers" tab is selected. Under the "Build Triggers" section, the "Poll SCM" option is checked, and the "Schedule" field contains the value "H/5 * * * *". A blue rectangular box highlights this schedule entry. Below the schedule, a note states: "No schedules so will only run due to SCM changes if triggered by a post-commit hook". The "Advanced Project Options" section is also visible at the bottom.

Jenkins > ci-cd-python >

General Build Triggers Advanced Project Options Pipeline

Build Triggers

Build after other projects are built ?
 Build periodically ?
 GitHub hook trigger for GITScm polling ?
 Poll SCM ?

Schedule ?
H/5 * * * *

No schedules so will only run due to SCM changes if triggered by a post-commit hook ?

Ignore post-commit hooks
 Disable this project ?
 Quiet period ?
 Trigger builds remotely (e.g., from scripts) ?

Advanced Project Options

Advanced...

Pipeline Save Apply Definition Pipeline script

Select 'Pipeline script from SCM'

The screenshot shows the Jenkins Pipeline configuration page for a project named "ci-cd-python". The "Pipeline" tab is selected in the top navigation bar. In the "Definition" section, a dropdown menu is open, showing two options: "Pipeline script" (which is checked) and "Pipeline script from SCM". Below the dropdown, there is a "Script" input field containing the number "1" and a "try sample Pipeline..." button. A checkbox labeled "Use Groovy Sandbox" is checked. There is also a link to "Pipeline Syntax". At the bottom of the form, there are "Save" and "Apply" buttons.

Jenkins > ci-cd-python >

General Build Triggers Advanced Project Options Pipeline

Pipeline

Definition

✓ Pipeline script
Pipeline script from SCM

Script 1 try sample Pipeline... ?

Use Groovy Sandbox ?

[Pipeline Syntax](#)

Save Apply

Page generated: Jul 19, 2019 11:23:53 AM PKT [REST API](#) Jenkins ver. 2.176.1

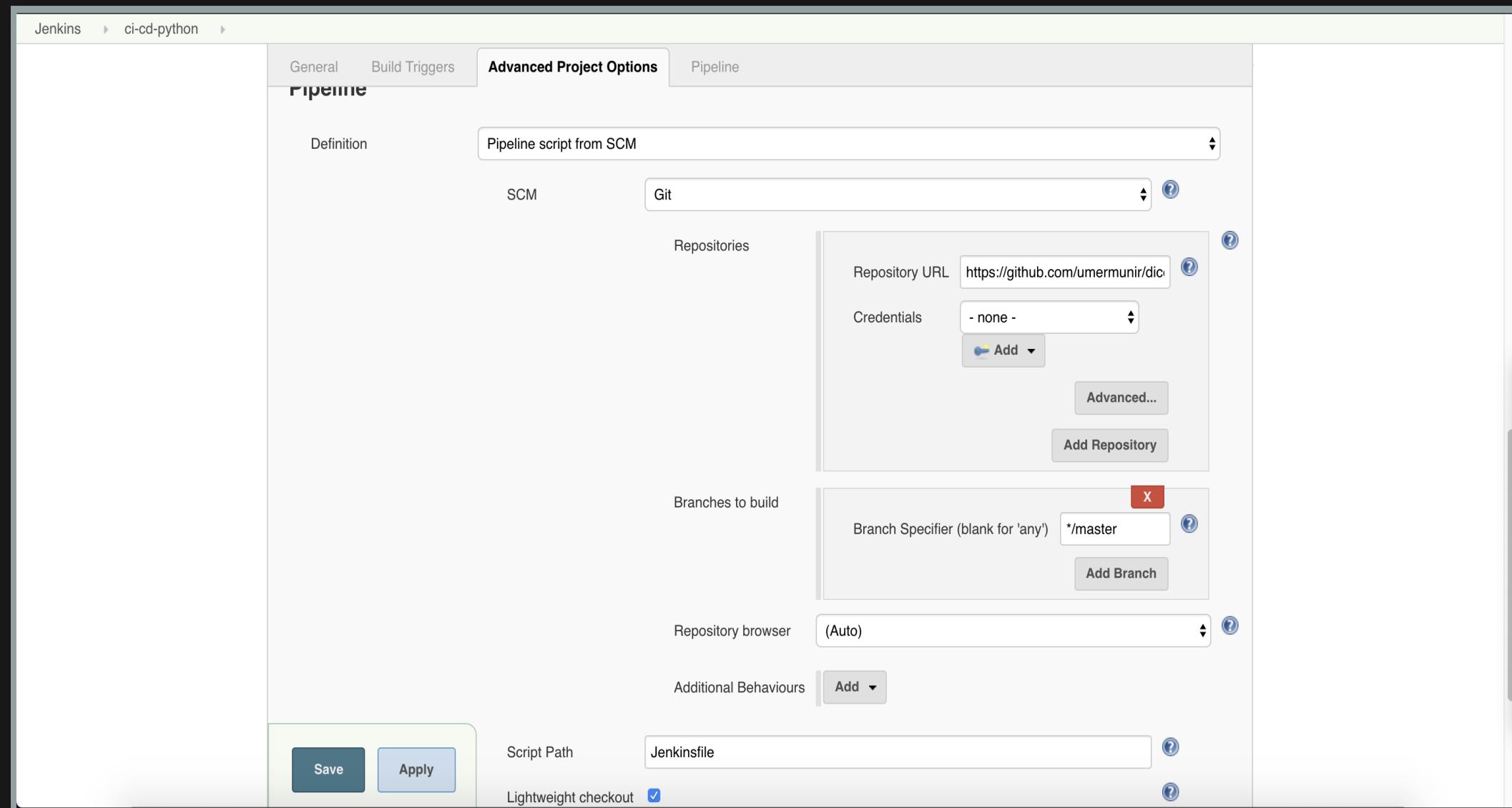
Select 'Git'

The screenshot shows the Jenkins Pipeline configuration page for a project named "ci-cd-python". The "Advanced Project Options" tab is selected. In the "Pipeline" section, under "Definition", the "Pipeline script from SCM" dropdown is set to "Pipeline script from SCM". The "SCM" dropdown menu is open, showing options: "None" (selected), "Git" (highlighted with a blue selection bar), "Subversion", and "Jenkinsfile". The "Script Path" field is set to "Jenkinsfile". The "Lightweight checkout" checkbox is checked. Below the form, there are "Save" and "Apply" buttons. At the bottom of the page, the footer includes the text "Page generated: Jul 19, 2019 11:23:53 AM PKT REST API Jenkins ver. 2.176.1".

Fork [DICE Python](#), clone it and then create a JenkinsFile below and push to repository

```
pipeline {
    agent none
    stages {
        stage('Build') {
            agent { docker { image 'python:2-alpine' } }
            steps {
                sh 'python -m py_compile sources/add2vals.py sources/calc.py'
            }
        }
        stage('Test') {
            agent { docker { image 'qnib/pytest' } }
            steps {
                sh 'py.test --verbose --junit-xml test-reports/results.xml sources/test_calc.py'
            }
        }
        stage('Deliver') {
            agent { docker { image 'cdrx/pyinstaller-linux:python2' } }
            steps {
                sh 'pyinstaller --onefile sources/add2vals.py'
            }
            post {
                success {
                    archiveArtifacts 'dist/add2vals'
                }
            }
        }
    }
}
```

Enter your Repository URL and Click Save



Make a change in local repository and push to remote repository to trigger the job

Jenkins ➤ ci-cd-python ➤ [ENABLE AUTO REFRESH](#)

[Back to Dashboard](#) [Status](#) [Changes](#) [Build Now](#) [Delete Pipeline](#) [Configure](#) [Full Stage View](#) [Rename](#) [Pipeline Syntax](#) [Polling Log](#)

Pipeline ci-cd-python
My First Python CI/CD

[Recent Changes](#) [Edit description](#) [Disable Project](#)

Stage View

Average stage times:

Build	Test	Deliver
24s	51s	31s

#1 Jul 19, 2019 11:29 AM No Changes

24s 51s

Build History [trend](#)

find Jul 19, 2019 11:29 AM

[RSS for all](#) [RSS for failures](#)

Permalinks

- [Last build \(#1\), 2.6 sec ago](#)

Page generated: Jul 19, 2019 11:29:43 AM PKT [REST API](#) Jenkins ver. 2.176.1

Click the Build and you should see the details as shown below.

Jenkins

2 search user log out

Jenkins ci-cd-python #1

ENABLE AUTO REFRESH

Back to Project Status Changes Console Output Edit Build Information Delete build '#1' Polling Log Git Build Data No Tags Docker Fingerprints Restart from Stage Replay Pipeline Steps Workspaces

Build #1 (Jul 19, 2019 11:29:40 AM)

Started 9 min 45 sec ago Took 2 min 10 sec

Build Artifacts add2vals 3.39 MB view

Started by an SCM change

Revision: a578760b3f8b3194f6b6e21167793f352053d58e

- refs/remotes/origin/master

Page generated: Jul 19, 2019 11:39:25 AM PKT REST API Jenkins ver. 2.176.1

TRIGGER JOBS FROM BASH SCRIPT

- REST API
- Jenkins CLI
- Trigger Job with Email

REST API

In this Lab, you are going to generate a token and use that to trigger job remotely using REST API. Go to the landing Page and Click the arrow next to your user

The screenshot shows the Jenkins dashboard with a single job listed:

S	W	Name ↓	Last Success	Last Failure	Last Duration
		first-job	N/A	N/A	

Legend: [S](#) [M](#) [L](#) | [RSS for all](#) [RSS for failures](#) [RSS for just latest builds](#)

Build Queue: No builds in the queue.

Build Executor Status: 1 Idle, 2 Idle

Page generated: Jul 19, 2019 2:21:44 PM PKT | [REST API](#) | Jenkins ver. 2.176.1

Select Configure

Jenkins

2

search

umer | log out

New Item

People

Build History

Project Relationship

Check File Fingerprint

Manage Jenkins

My Views

Lockable Resources

Credentials

New View

All +

S	W	Name ↓	Last Success	Last Failure	Last Duration
		first-job	N/A	N/A	N/A

Icon: S M L

Legend RSS for all RSS for failures RSS for just latest builds

Build Queue

No builds in the queue.

Build Executor Status

1 Idle
2 Idle

localhost:8080/user/umer/configure

Page generated: Jul 19, 2019 2:21:56 PM PKT REST API Jenkins ver. 2.176.1

The screenshot shows the Jenkins dashboard. A context menu is open over the 'Configure' link in the top right corner, listing options: Builds, Configure (which is selected), My Views, and Credentials. On the left sidebar, there are links for New Item, People, Build History, Project Relationship, Check File Fingerprint, Manage Jenkins, My Views, Lockable Resources, Credentials, and New View. Below the sidebar, there's a table showing a single job named 'first-job' with status information. Under 'Build Queue', it says 'No builds in the queue.' Under 'Build Executor Status', it shows '1 Idle' and '2 Idle'. At the bottom, the URL 'localhost:8080/user/umer/configure' is shown in a red box, along with page generation details.

Click Add Token

The screenshot shows the Jenkins user profile page for the user 'umer'. The left sidebar contains links for People, Status, Builds, Configure, My Views, and Credentials. The main content area shows the user's full name as 'umer' and an empty description field. Under the 'API Token' section, it states 'There are no registered tokens for this user.' and features a prominent 'Add new Token' button. The 'Credentials' section notes that credentials are only available to the user. The 'E-mail' section shows the email address 'umer@live.com'. The 'Extended Email Job Watching' section indicates 'No configuration available'. At the bottom are 'Save' and 'Apply' buttons.

Jenkins

2 search umer | log out

Jenkins > umer

People

Status

Builds

Configure

My Views

Credentials

Full Name
umer

Description

API Token

Current token(s)
There are no registered tokens for this user.

Add new Token

Credentials

Credentials are only available to the user they belong to

E-mail

E-mail address
umer@live.com

Your e-mail address, like joe.chin@sun.com

Extended Email Job Watching

No configuration available

Save Apply

Specify name of the token

The screenshot shows the Jenkins user profile page for the user 'umer'. The left sidebar includes links for People, Status, Builds, Configure, My Views, and Credentials. The main content area has tabs for Full Name, Description, API Token, Credentials, E-mail, and Extended Email Job Watching. The 'API Token' tab is active, showing a list of current tokens with 'umer-token' selected. A modal dialog is open, displaying the token value 'umer-token' in a text input field, with 'Generate' and 'Cancel' buttons below it. The 'Add new Token' button is also visible. The 'Credentials' tab shows a note about e-mail availability. The 'E-mail' tab shows an e-mail address 'umer@live.com'. At the bottom are 'Save' and 'Apply' buttons.

Jenkins

2 search umer log out

umer

People Status Builds Configure My Views Credentials

Full Name: umer

Description:

API Token

Current token(s): There are no registered tokens for this user.

umer-token Generate Cancel

Add new Token

Credentials

Credentials are only available to the user they belong to

E-mail

E-mail address: umer@live.com

Your e-mail address, like joe.chin@sun.com

Extended Email Job Watching

Save Apply

Copy token generated

The screenshot shows the Jenkins user profile page for the user 'umer'. The left sidebar includes links for People, Status, Builds, Configure, My Views, and Credentials. The main content area displays the user's full name ('umer') and an empty description field. Under the 'API Token' section, there is a list of current tokens: 'umer-token' (with the value '11b7f5a47cf9a9a6fa3f7f2c931a86bfdd'). A warning message states: '⚠ Copy this token now, because it cannot be recovered in the future.' Below this is a button labeled 'Add new Token'. The 'Credentials' section notes that credentials are only available to the user. The 'E-mail' section shows the email address 'umer@live.com' with a placeholder 'Your e-mail address, like joe.chin@sun.com'. The 'Extended Email Job Watching' section indicates 'No configuration available'. At the bottom are 'Save' and 'Apply' buttons.

Jenkins

2 search umer | log out

Jenkins > umer

People Status Builds Configure My Views Credentials

Full Name: umer

Description:

API Token

Current token(s):

umer-token 11b7f5a47cf9a9a6fa3f7f2c931a86bfdd

⚠ Copy this token now, because it cannot be recovered in the future.

Add new Token

Credentials

Credentials are only available to the user they belong to

E-mail

E-mail address: umer@live.com

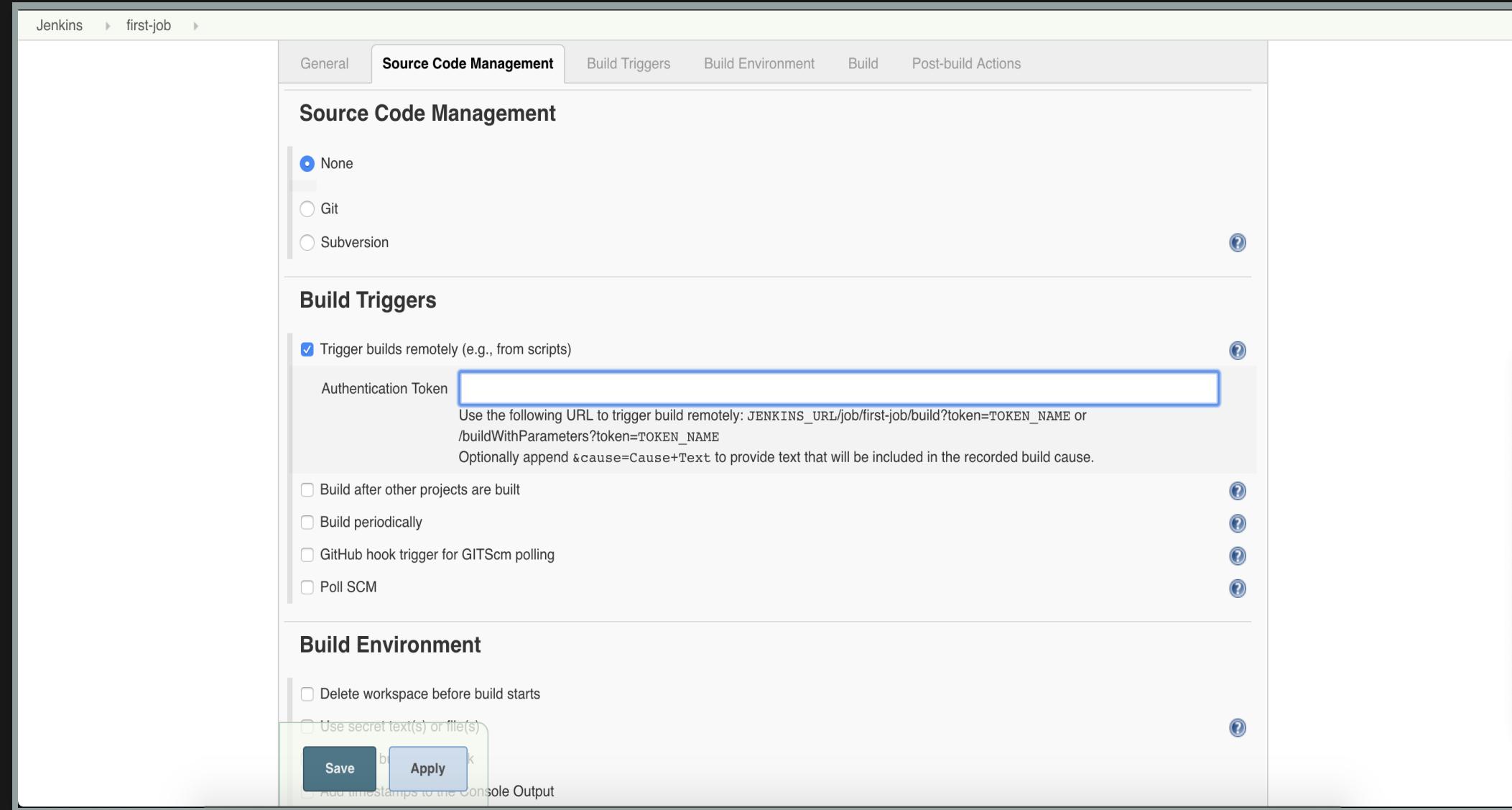
Your e-mail address, like joe.chin@sun.com

Extended Email Job Watching

No configuration available

Save Apply

Next go to the job you want to trigger and in the Build Triggers section, select 'Trigger builds remotely' and paste the copied token and Click Save.



Run the following command from your terminal

```
curl -u username:token http://localhost:8080/job/first-job/build?token=token
```

where "username" is your Jenkins user, "token" is the token you generated earlier and first-job is the job you want to trigger

You should see the triggered build as shown below

The screenshot shows the Jenkins interface for the project "first-job". The top navigation bar includes the Jenkins logo, a user icon, a search bar, and links for "ENABLE AUTO REFRESH", "Help", "User", and "log out". A red notification badge with the number "2" is visible in the top right corner. The main content area is titled "Project first-job" and displays the message "My First Job". On the left, a sidebar lists project actions: "Back to Dashboard", "Status", "Changes", "Workspace", "Build Now", "Delete Project", "Configure", and "Rename". To the right of the title, there are "edit description" and "Disable Project" buttons. Below the title, there are two links: "Workspace" (with a folder icon) and "Recent Changes" (with a notepad icon). A "Permalinks" section contains a "Build History" table with one entry: "#1 Jul 19, 2019 2:34 PM". At the bottom, there are RSS feed links for "RSS for all" and "RSS for failures". The footer of the page includes the text "Page generated: Jul 19, 2019 2:34:13 PM PKT REST API Jenkins ver. 2.176.1".

JENKINS CLI

First you are going to download jenkins-cli by running following command

```
wget http://localhost:8080/jnlpJars/jenkins-cli.jar
```

where localhost:8080 is the URL of Jenkins running on your system

Trigger your job by running the following command. Replace "username" with your user in Jenkins and "password" with your password.

```
java -jar jenkins-cli.jar -s "http://localhost:8080" -auth username:password build "first-job"
```

Where localhost:8080 is the Jenkins URL and "first-job" is the job you want to trigger

You should see the job being triggered as shown below

The screenshot shows the Jenkins interface for the project "first-job". The top navigation bar includes the Jenkins logo, a user icon, a search bar, and links for "ENABLE AUTO REFRESH", "Help", and "log out". The left sidebar contains links for "Back to Dashboard", "Status", "Changes", "Workspace", "Build Now", "Delete Project", "Configure", and "Rename". The main content area is titled "Project first-job" and "My First Job". It features a "Workspace" section with a folder icon and a "Recent Changes" section with a document icon. On the right, there are buttons for "edit description" and "Disable Project". A "Permalinks" section lists four build links: "Last build (#2), 5.7 sec ago", "Last stable build (#2), 5.7 sec ago", "Last successful build (#2), 5.7 sec ago", and "Last completed build (#2), 5.7 sec ago". At the bottom, there are RSS feed links for "RSS for all" and "RSS for failures". The footer indicates the page was generated on Jul 19, 2019, at 3:04:21 PM PKT, and provides links for the REST API and Jenkins version 2.176.1.

Project first-job

My First Job

Workspace

Recent Changes

Permalinks

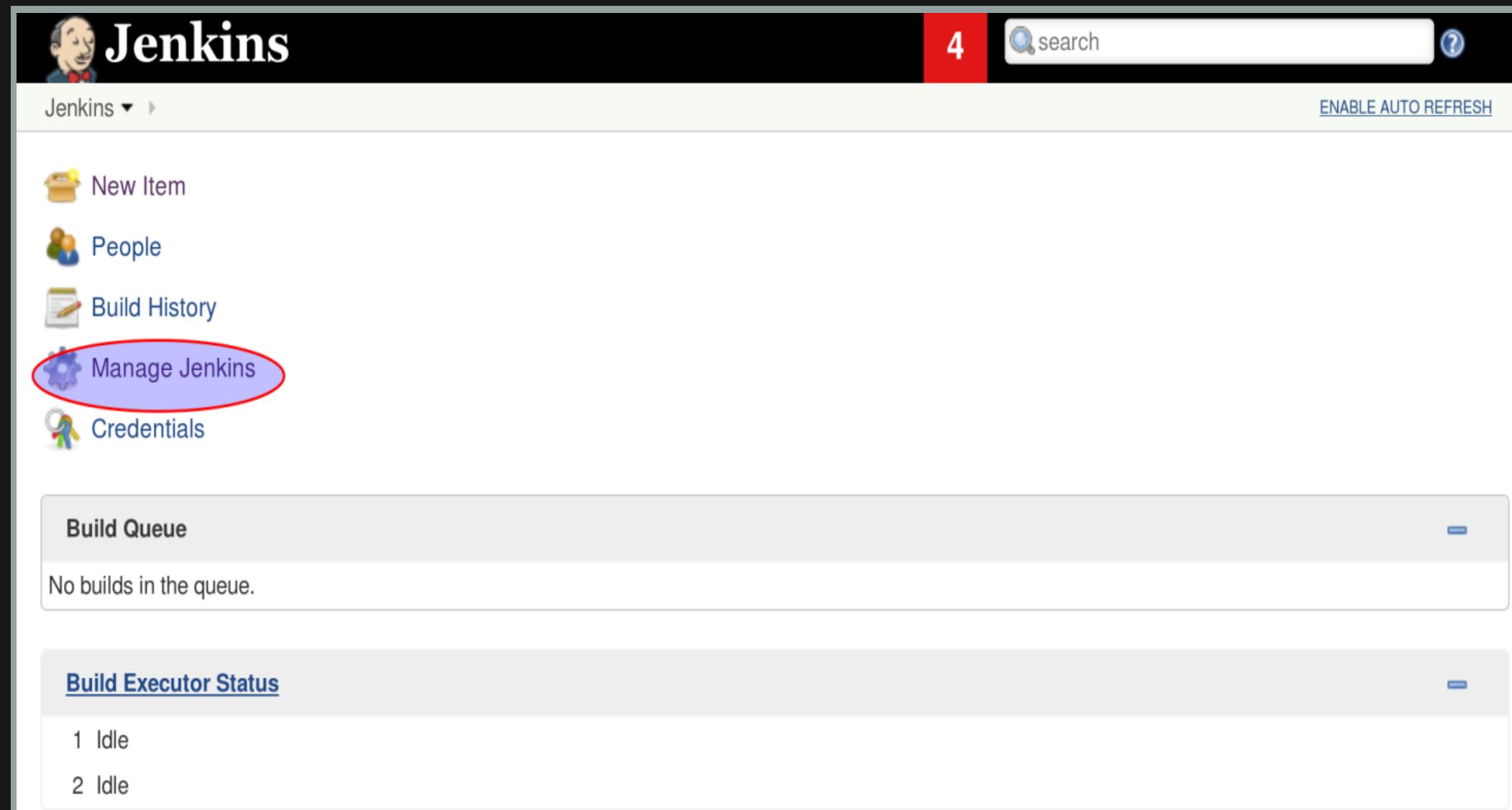
- Last build (#2), 5.7 sec ago
- Last stable build (#2), 5.7 sec ago
- Last successful build (#2), 5.7 sec ago
- Last completed build (#2), 5.7 sec ago

RSS for all RSS for failures

Page generated: Jul 19, 2019 3:04:21 PM PKT REST API Jenkins ver. 2.176.1

TRIGGER JOB WITH EMAIL

On your Jenkins home page, click Manage Jenkins.

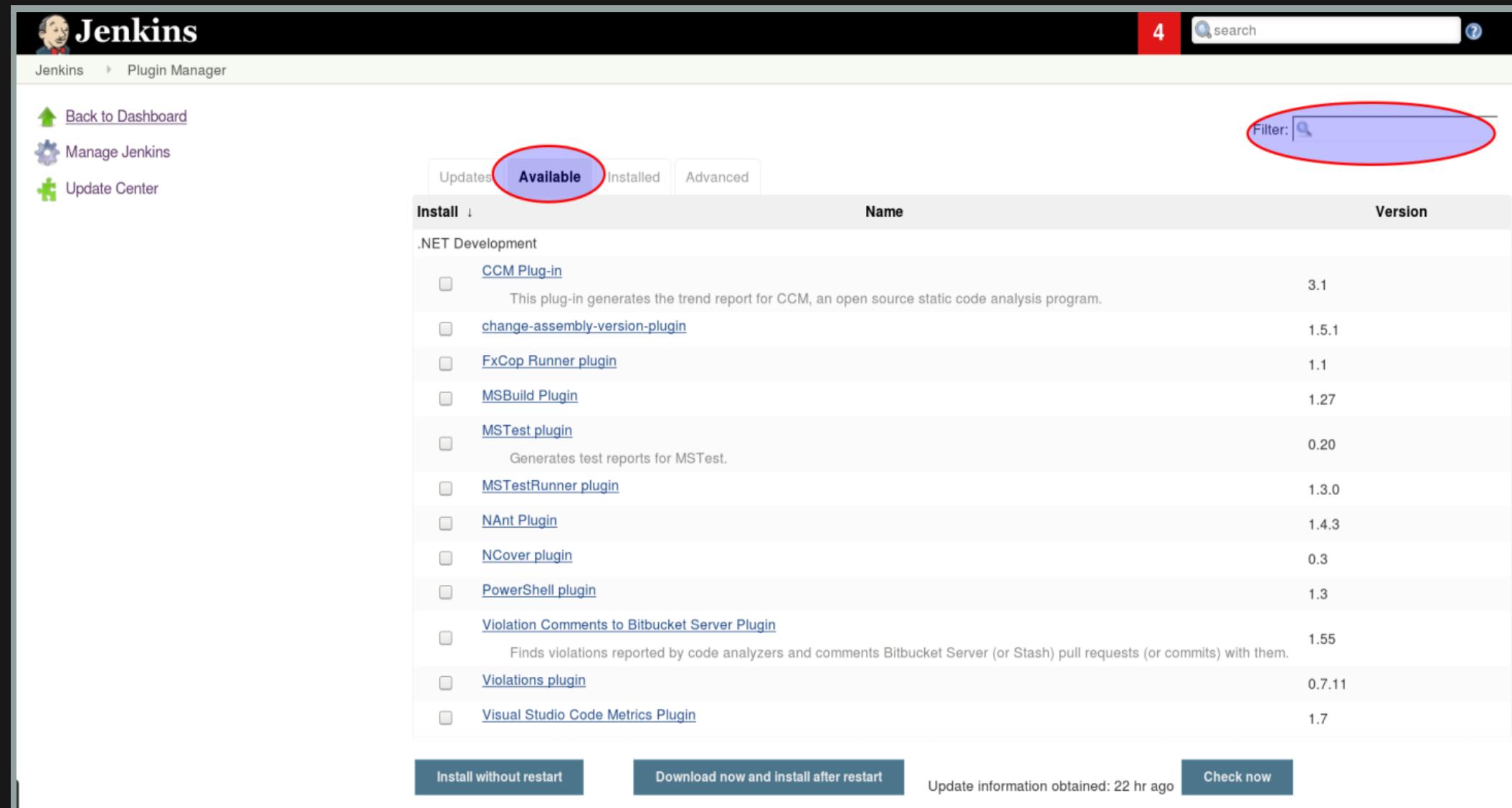


The screenshot shows the Jenkins home page. At the top, there is a navigation bar with the Jenkins logo, a search bar, and a help icon. A red notification badge with the number '4' is visible. Below the navigation bar, there is a dropdown menu labeled 'Jenkins ▾' and a link 'ENABLE AUTO REFRESH'. On the left side, there is a sidebar with several links: 'New Item', 'People', 'Build History', 'Manage Jenkins' (which is highlighted with a red oval), and 'Credentials'. The main content area contains two sections: 'Build Queue' (which displays 'No builds in the queue.') and 'Build Executor Status' (which shows '1 Idle' and '2 Idle').

Click on Go to plugin manager button

The screenshot shows the Jenkins Manage Jenkins interface. On the left, there's a sidebar with links like New Item, People, Build History, Manage Jenkins (which is selected), and Credentials. Below that are two sections: Build Queue (empty) and Build Executor Status (1 Idle, 2 Idle). The main area is titled "Manage Jenkins". It displays several warning messages: one about old data format, one about a new Jenkins version, and one about unsecured access. It also lists "Warnings have been published for the following currently installed components:" which include Script Security Plugin 1.29.1 (with a note about multiple sandbox bypasses) and Pipeline: Input Step 2.7 (with a note about users interacting with input steps). At the bottom, there's a "Go to plugin manager" button, which is circled in red to indicate it's the target of the click. Other buttons in this section include "Configure which of these warnings are shown", "Manage", "Dismiss", "Setup Security", and "Dismiss".

Select the Available tag and using the Filter search box, search for Poll Mailbox Trigger Plugin, select it and choose Install without restart.

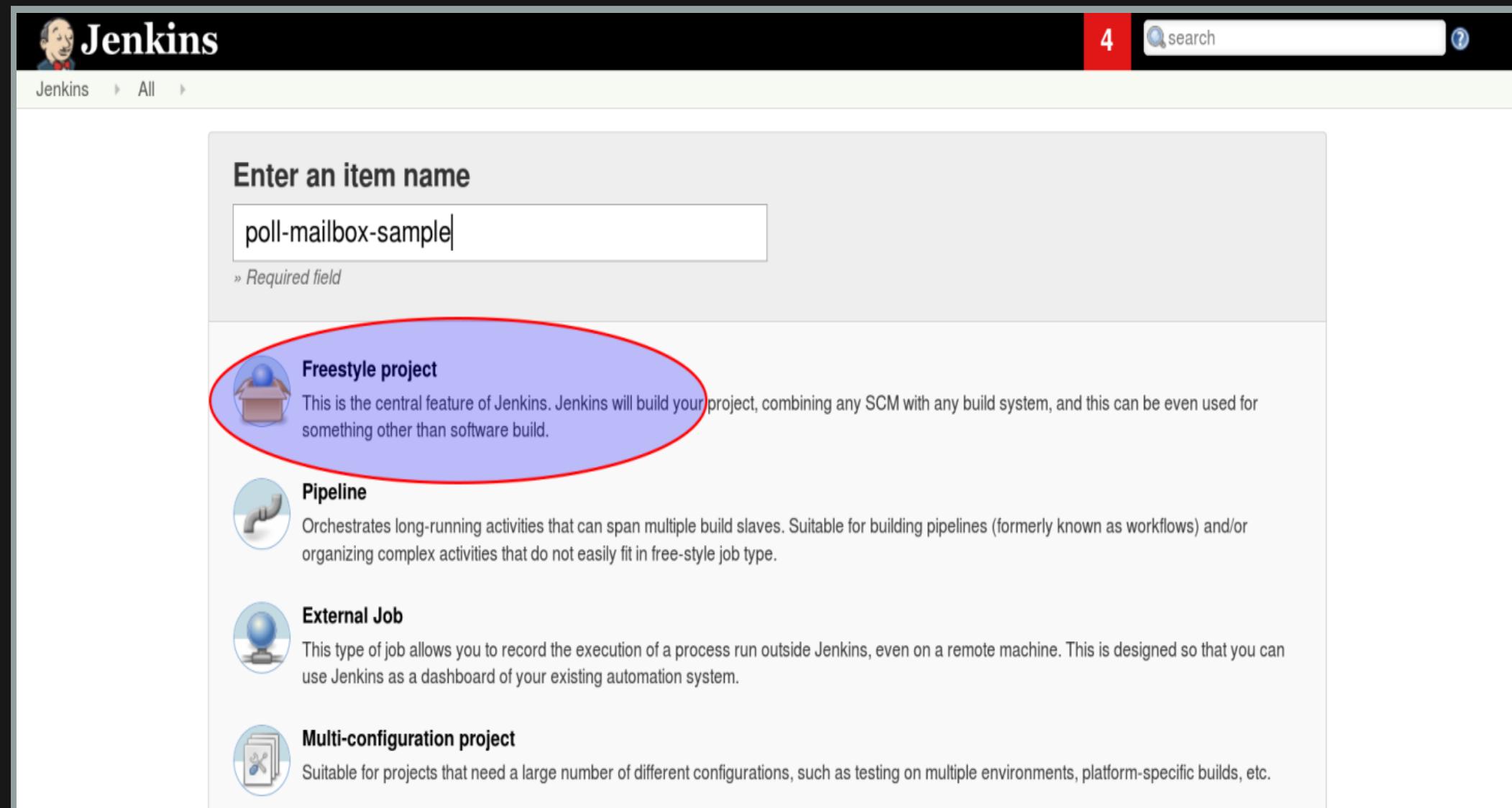


The screenshot shows the Jenkins Plugin Manager interface. At the top, there is a navigation bar with a Jenkins logo, a 'Back to Dashboard' link, and a 'Plugin Manager' link. To the right of the navigation bar is a red box containing the number '4'. Further to the right is a search bar with a magnifying glass icon and a question mark icon. Below the navigation bar, there are four tabs: 'Updates' (disabled), 'Available' (selected and highlighted with a red oval), 'Installed', and 'Advanced'. On the left side, there is a sidebar with links: 'Back to Dashboard', 'Manage Jenkins', and 'Update Center'. The main content area is titled 'Install' and shows a list of '.NET Development' plugins. The table has three columns: 'Name', 'Version', and 'Description'. The 'Name' column lists the plugin names, the 'Version' column lists their versions, and the 'Description' column provides a brief description of each plugin. At the bottom of the page, there are three buttons: 'Install without restart' (disabled), 'Download now and install after restart', and 'Check now'. A status message 'Update information obtained: 22 hr ago' is displayed between the 'Download now...' button and the 'Check now' button.

Install	Name	Version
CCM Plug-in	3.1	This plug-in generates the trend report for CCM, an open source static code analysis program.
change-assembly-version-plugin	1.5.1	
FxCop Runner plugin	1.1	
MSBuild Plugin	1.27	
MSTest plugin	0.20	Generates test reports for MSTest.
MSTestRunner plugin	1.3.0	
NAnt Plugin	1.4.3	
NCover plugin	0.3	
PowerShell plugin	1.3	
Violation Comments to Bitbucket Server Plugin	1.55	Finds violations reported by code analyzers and comments Bitbucket Server (or Stash) pull requests (or commits) with them.
Violations plugin	0.7.11	
Visual Studio Code Metrics Plugin	1.7	

Install without restart Download now and install after restart Update information obtained: 22 hr ago Check now

On the Jenkins homepage, select New item, and then create a new freestyle project as shown below



The screenshot shows the Jenkins interface for creating a new item. At the top, there is a navigation bar with the Jenkins logo, a search bar containing 'search', and a help icon. Below the navigation bar, the URL 'Jenkins > All' is visible. The main content area has a title 'Enter an item name' and a text input field containing 'poll-mailbox-sample'. A note below the input field states '» Required field'. Below this, there are four options: 'Freestyle project' (with a house icon), 'Pipeline' (with a pipe icon), 'External Job' (with a globe icon), and 'Multi-configuration project' (with a gear and document icon). The 'Freestyle project' option is highlighted with a large red oval.

Enter an item name

poll-mailbox-sample

» Required field

Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.

Pipeline
Orchestrates long-running activities that can span multiple build slaves. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.

External Job
This type of job allows you to record the execution of a process run outside Jenkins, even on a remote machine. This is designed so that you can use Jenkins as a dashboard of your existing automation system.

Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.

Enter description and Click Build Triggers

Jenkins

4 search ?

Jenkins > poll-mailbox-sample >

General Source Code Management Build Triggers Build Environment Build Post-build Actions

Project name poll-mailbox-sample

Description This is a sample job to demo the poll-mailbox-trigger-plugin

[Plain text] [Preview](#)

Discard old builds [?](#)

GitHub project [?](#)

This project is parameterized [?](#)

Throttle builds [?](#)

Disable this project [?](#)

Execute concurrent builds if necessary [?](#)

Select Poll Mailbox Trigger. If you want to use a GMAIL account, enter GMAIL email address in the username and the password as shown below. We need some advanced configuration as well so click on that

Build Triggers

- Build after other projects are built ?
- Build periodically ?
- GitHub hook trigger for GITScm polling ?
- Poll Mailbox Trigger ?

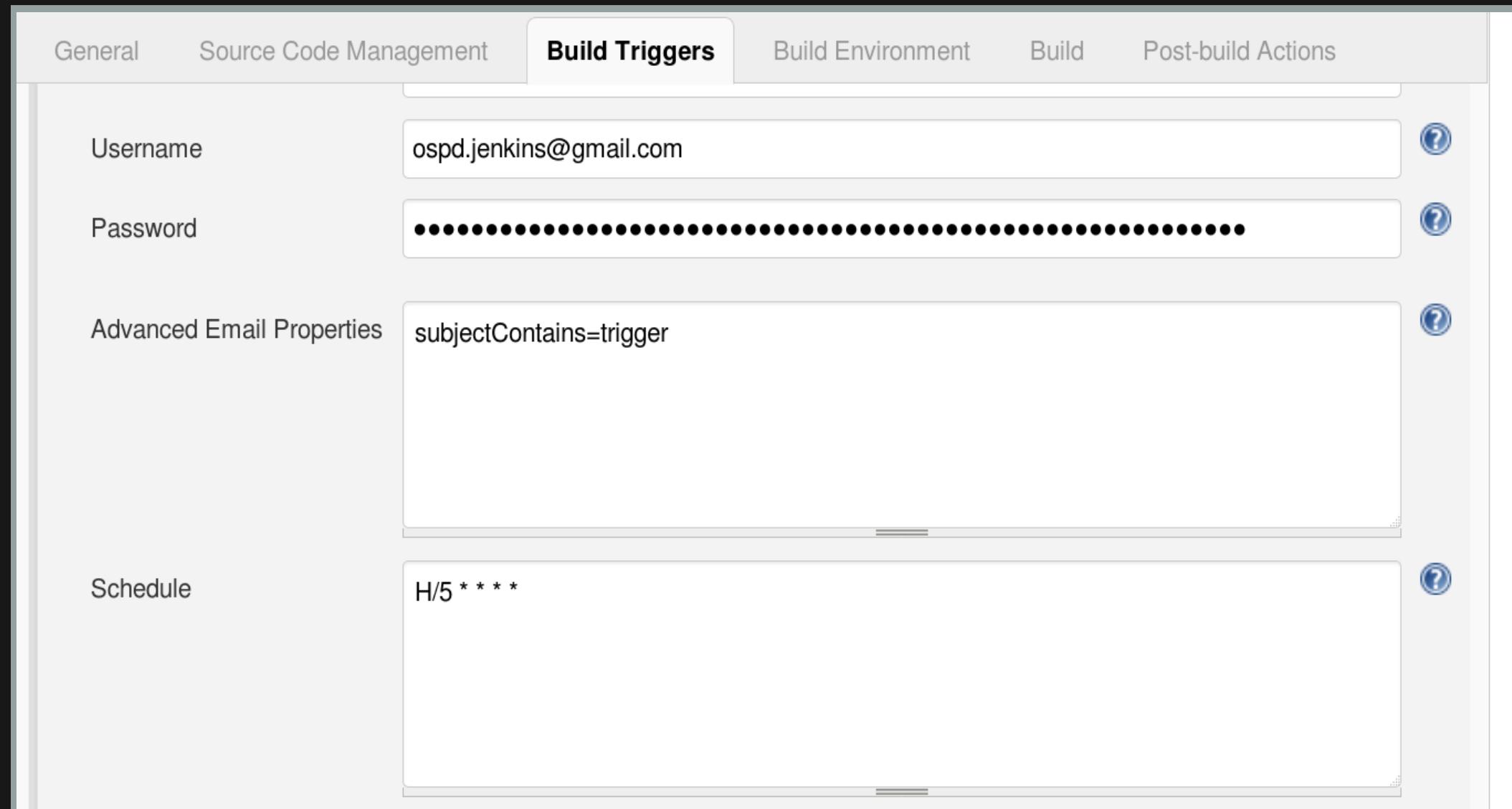
Host ?

Username ?

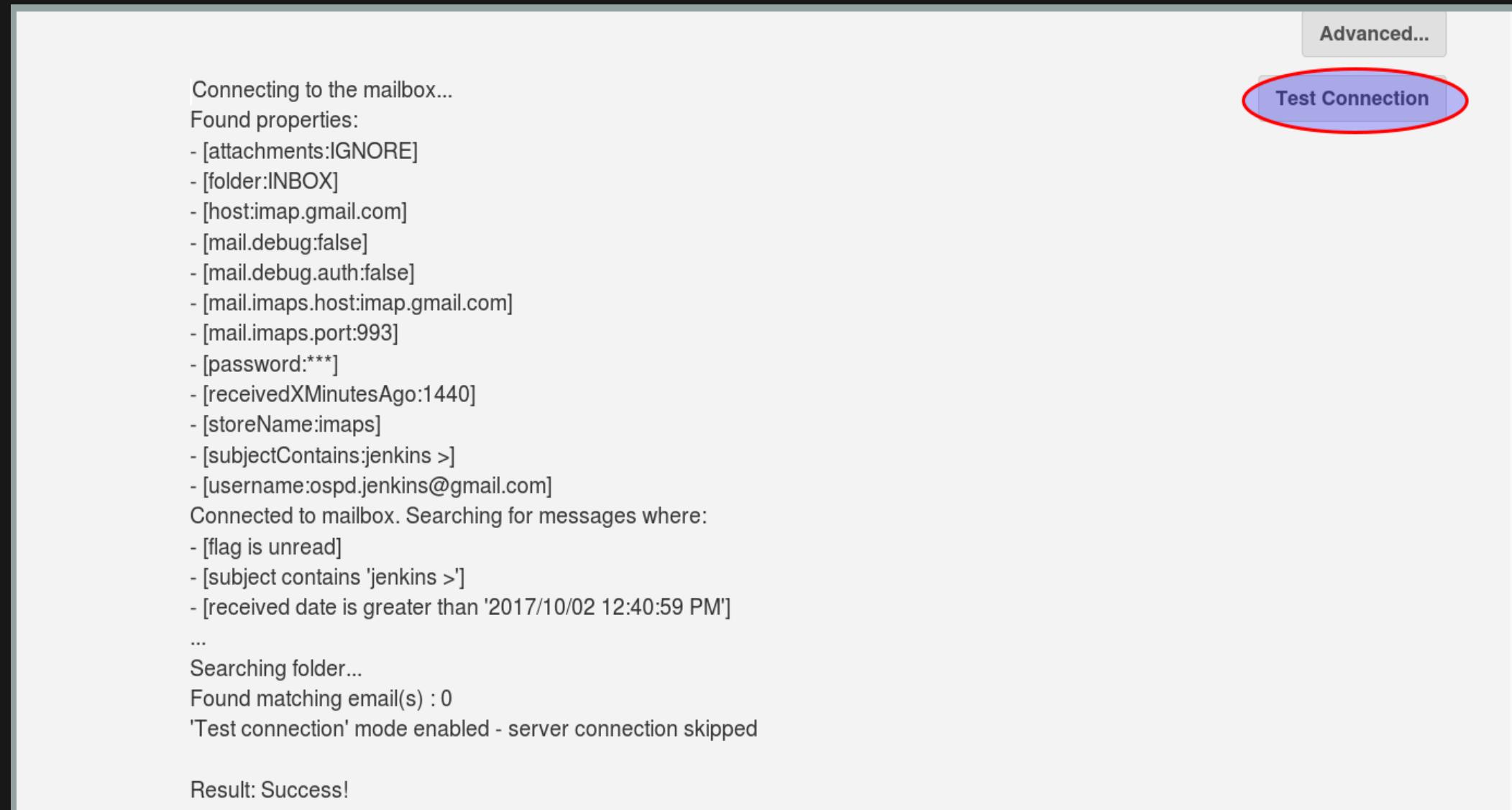
Password ?

Advanced...

We can use the `subjectContains` property to set the string in the subject line that the job would need to be present for the job to be triggered. We can also configure the interval at which the plugin scans the inbox for unread messages containing the `subjectContains` property in the subject line.



After configuring everything, hit on the Test Connection button to verify



Click on Build section and the command as shown below

The screenshot shows the Jenkins job configuration interface for a job named "first-job". The "Build" tab is selected in the top navigation bar. Below it, the "Build Environment" section contains several checkboxes:

- Delete workspace before build starts
- Use secret text(s) or file(s)
- Abort the build if it's stuck
- Add timestamps to the Console Output
- Inspect build log for published Gradle build scans
- With Ant

The "Build" section contains a single step configuration:

Execute shell

Command: `echo Hello World`

Below the command input field, there is a link: "See [the list of available environment variables](#)". To the right of the command input field is an "Advanced..." button.

At the bottom of the build configuration area, there is a "Save" button and an "Apply" button.

Send an email with the subject line "trigger", this should trigger your job

The screenshot shows the Jenkins interface for a job named "poll-mailbox-sample". The left sidebar lists options like Back to Project, Status, Changes, and Console Output. The main area is titled "Console Output" and displays the build log. A red box highlights the first line of the log: "[PollMailboxTrigger] Job was triggered by email sent from [REDACTED]@redhat.com (log)". The log continues with the build process, including the command executed and the output "Hello Sai".

Jenkins

4 search

Jenkins poll-mailbox-sample #1

Back to Project

Status

Changes

Console Output

View as plain text

Edit Build Information

Delete Build

Parameters

Console Output

```
[PollMailboxTrigger] Job was triggered by email sent from [REDACTED]@redhat.com (log)
Building in workspace /var/lib/jenkins/workspace/poll-mailbox-sample
[poll-mailbox-sample] $ /bin/sh -xe /tmp/jenkins5905295343002967673.sh
+ echo 'Hello Sai'
Hello Sai
Finished: SUCCESS
```

JENKINS SECURITY

- Keep Jenkins shielded from the internet using firewall rules
- You will need to whitelist github IP for push requests
- Keep Jenkins up-to-date
 - Use LTS (Long Term Support) edition
- Always keep the plugins up to date
- Configure authentication/authorization
 - Administrator can decrypt credentials
- In larger systems, don't build on the master
- Limit project names to a sane (e.g. alphanumeric) character set
- Avoid scheduling all jobs to start at the same time
- Archive unused jobs before removing them