

DevOps

1. Create a Folder with the name Jenkins and create Job inside the folder execute a simple windows batch command to show JDK version in your system. Create a user with your name and assign only create, build, configure jobs permissions under the created Jenkins folder.

a. Create a Folder named Jenkins

Dashboard > All >

Enter an item name

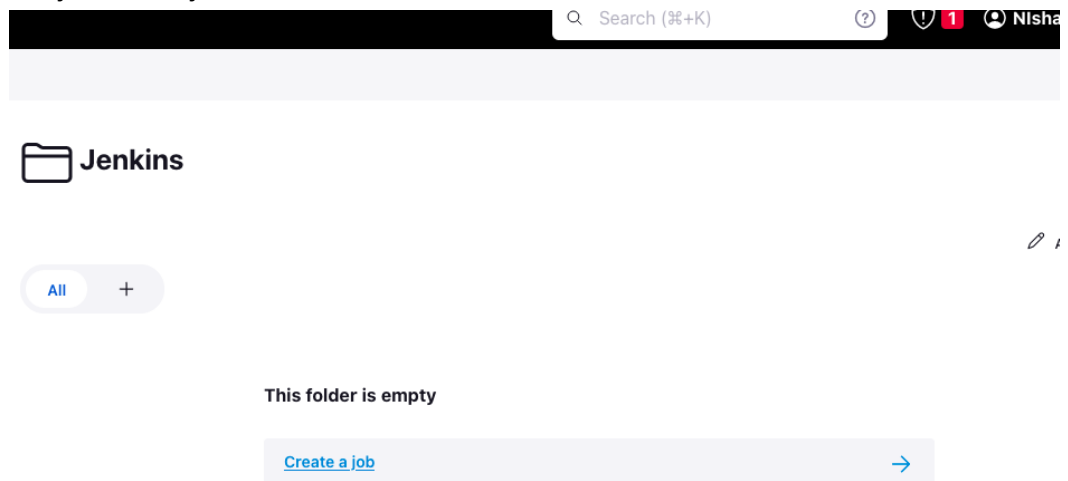
Jenkins

» 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.
- Multibranch Pipeline**
Creates a set of Pipeline projects according to detected branches in one SCM repository.
- Organization Folder**
Creates a set of multibranch project subfolders by scanning for repositories.

OK

B. create a job inside jenkins folder



C. select the Build Step

Dashboard > Jenkins > Jenkins > Configuration

Configure

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

Build Environment

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

Build Steps

Add build step +

Filter

- 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

Google Docs: Online Document | 1ms21mc038 - DevOps - Google | Jenkins Config [Jenkins] | Jenkins Pipeline - javatpoint

localhost:8080/job/Jenkins/job/Jenkins/configure

Dashboard > Jenkins > Jenkins > Configuration

Configure

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

Build Steps

- ☐ Terminate a build if it's stuck
- ☐ With Ant ?

Execute shell ?

Command

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

```
javac --version
```

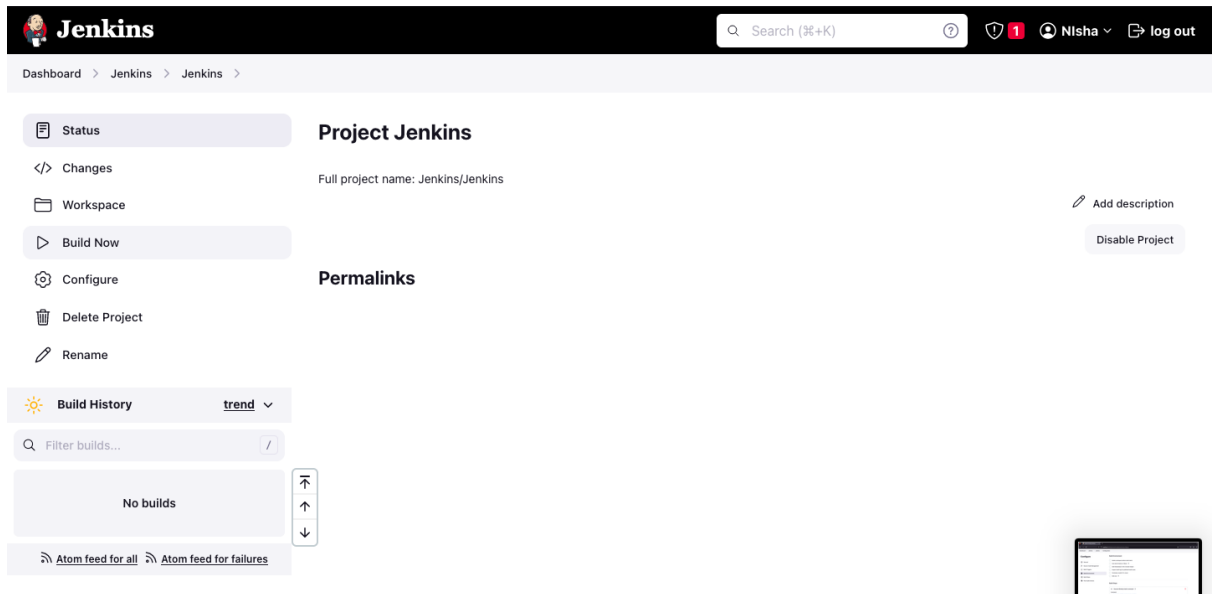
Advanced ▾

Add build step +

Post-build Actions

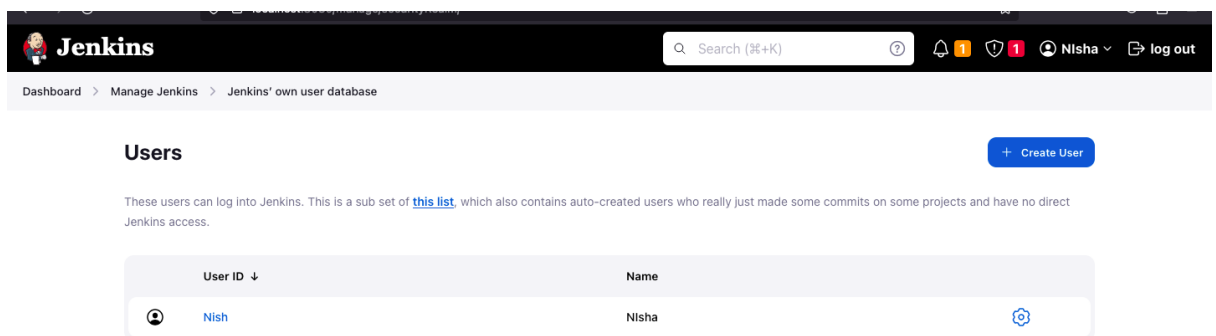
Save **Apply**

d. Build the project



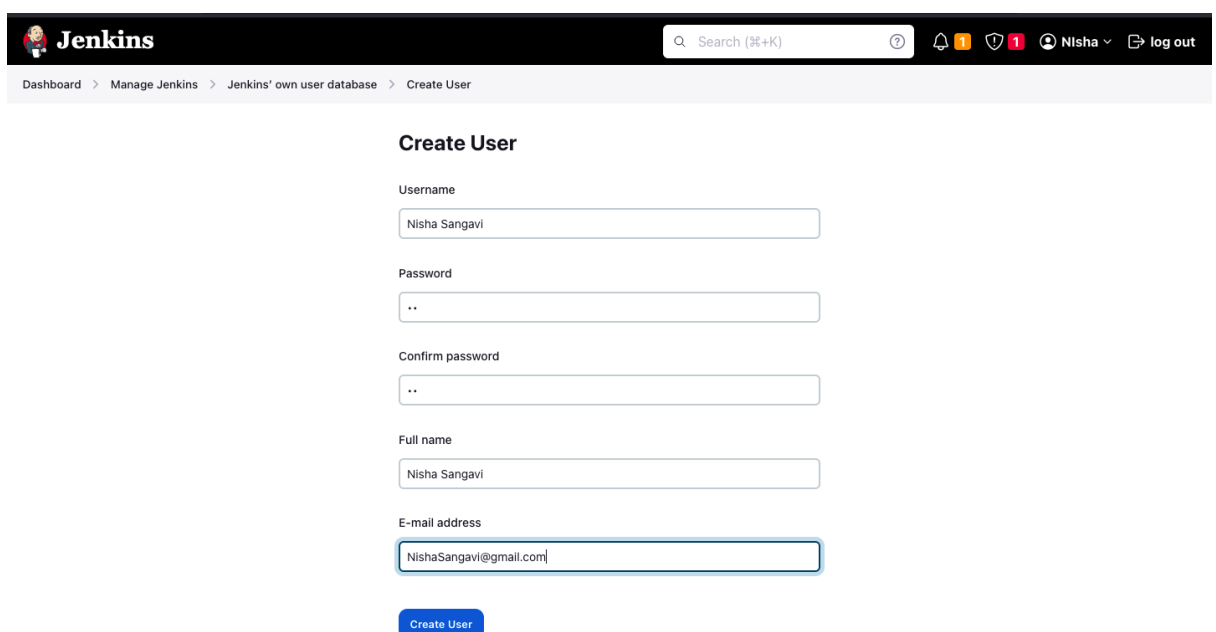
The screenshot shows the Jenkins 'Project Jenkins' page. The left sidebar contains a menu with options: Status, Changes, Workspace, Build Now, Configure, Delete Project, and Rename. The main content area has a 'Project Jenkins' header with the full project name 'Jenkins/Jenkins'. Below this is a 'Permalinks' section. At the bottom, there is a 'Build History' section with a 'trend' dropdown and a search bar. The build history is currently empty, showing 'No builds'. On the right side, there are links for 'Add description' and 'Disable Project'.

E. Create a User



The screenshot shows the Jenkins 'Users' page. The header includes the Jenkins logo and a search bar. The main content area has a 'Users' section with a '+ Create User' button. Below this, there is a table listing users. The table has two columns: 'User ID' and 'Name'. The first user listed is 'Nish' with the name 'Nisha'. There is a settings icon next to the user name.

User ID ↓	Name
Nish	Nisha



The screenshot shows the Jenkins 'Create User' page. The form includes fields for Username, Password, Confirm password, Full name, and E-mail address. The Username field is filled with 'Nisha Sangavi'. The Password and Confirm password fields are empty. The Full name field is filled with 'Nisha Sangavi'. The E-mail address field is filled with 'NishaSangavi@gmail.com'. There is a 'Create User' button at the bottom.

Username

Nisha Sangavi

Password

..

Confirm password

..

Full name

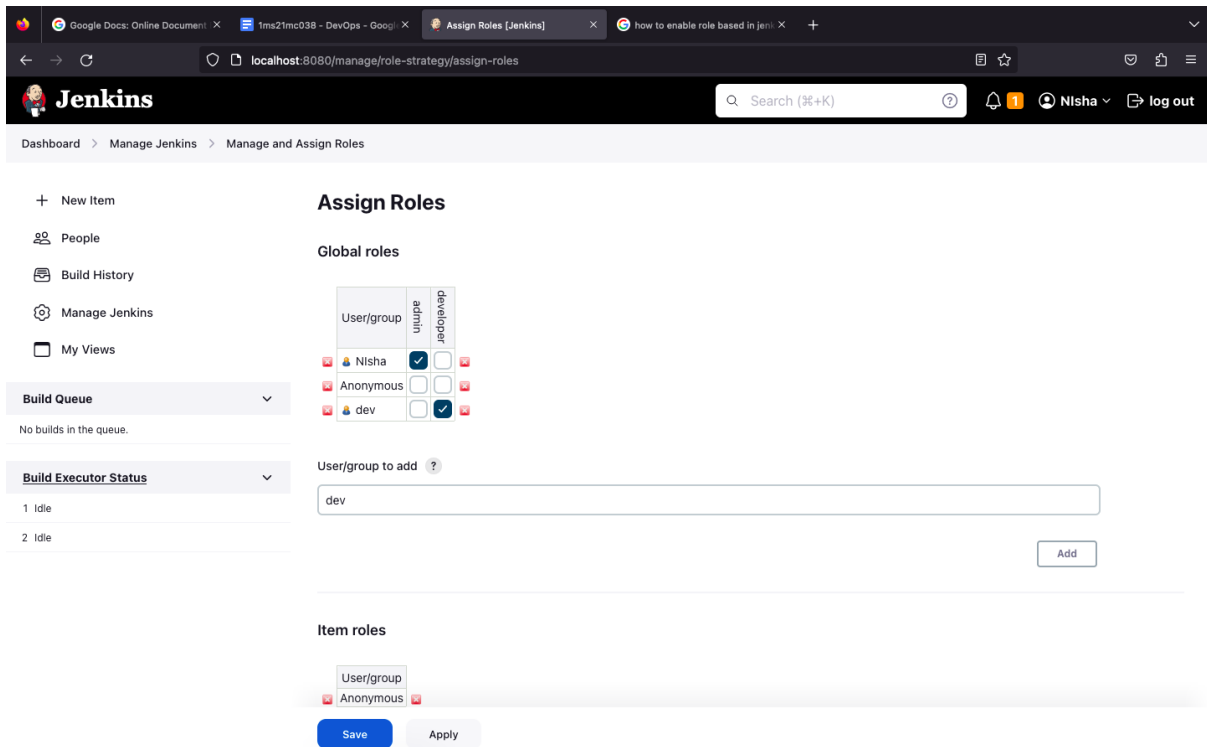
Nisha Sangavi

E-mail address

NishaSangavi@gmail.com

Create User

F. Assign roles



The screenshot shows the Jenkins 'Assign Roles' page. The left sidebar contains navigation links: New Item, People, Build History, Manage Jenkins, and My Views. Below these are two expandable sections: 'Build Queue' (showing 'No builds in the queue.') and 'Build Executor Status' (showing two 'Idle' executors). The main content area is titled 'Assign Roles' and is divided into 'Global roles' and 'Item roles' sections. In the 'Global roles' section, a table lists roles for 'User/group' (admin, developer), 'Nisha', 'Anonymous', and 'dev'. The 'dev' role is selected. Below the table, there is a 'User/group to add' field with 'dev' entered and an 'Add' button. The 'Item roles' section shows a table with 'User/group' and 'Anonymous' roles. At the bottom, there are 'Save' and 'Apply' buttons.

Assign Roles

Global roles

User/group	admin	developer
Nisha	<input checked="" type="checkbox"/>	<input type="checkbox"/>
Anonymous	<input type="checkbox"/>	<input type="checkbox"/>
dev	<input type="checkbox"/>	<input checked="" type="checkbox"/>

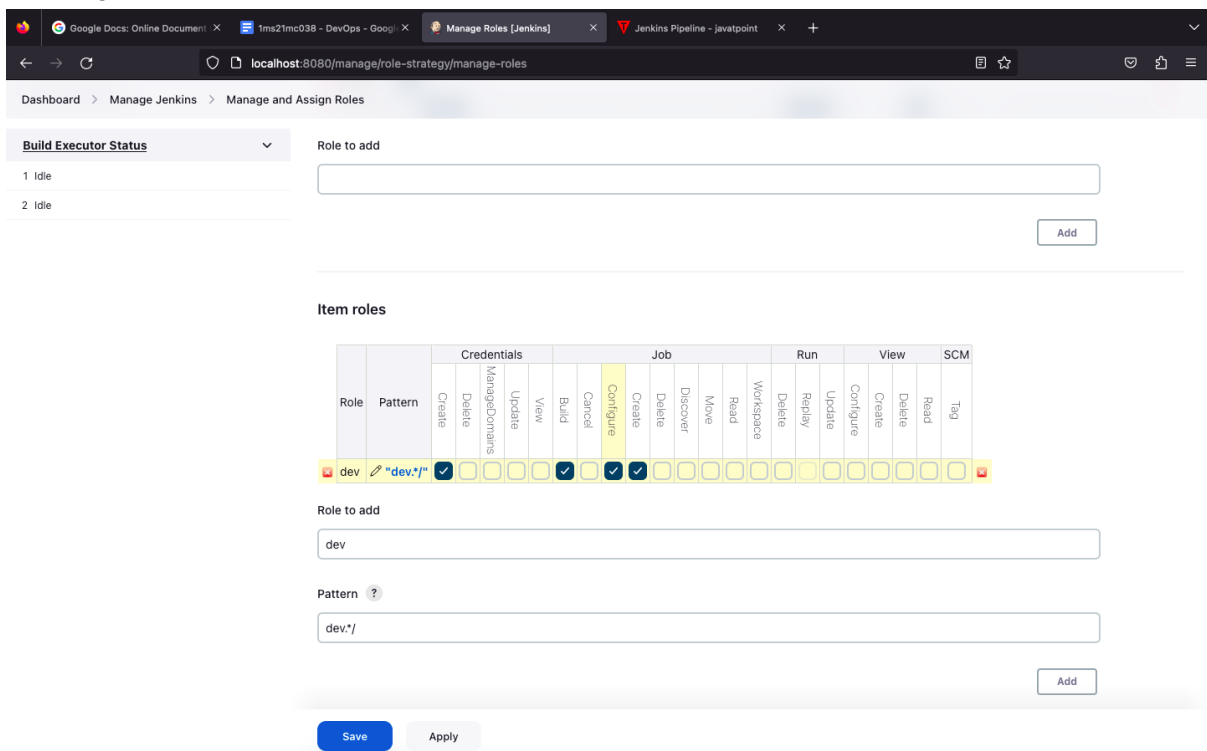
User/group to add ?
dev
Add

Item roles

User/group
Anonymous

Save Apply

G. Assign item roles



The screenshot shows the Jenkins 'Manage Roles' page. The left sidebar is the same as in the previous screenshot. The main content area is titled 'Manage Roles' and has a 'Role to add' field with an 'Add' button. Below this is the 'Item roles' section, which contains a table with columns for Role, Pattern, and various permissions. The 'dev' role is selected, and its pattern is 'dev.*/'. At the bottom, there are 'Save' and 'Apply' buttons.

Manage Roles

Role to add
Add

Item roles

Role	Pattern	Credentials	Job	Run	View	SCM
dev	dev.*/*	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>

Role to add
dev
Pattern ?
dev.*/*
Add
Save Apply

H. Build the job

The screenshot shows the Jenkins web interface in a browser. The address bar indicates the URL is `localhost:8080/job/Jenkins/`. The Jenkins logo and name are at the top left. A search bar and user profile (Nisha) are at the top right. The left sidebar contains navigation links: Status, Configure, New Item, Delete Folder, People, Build History, Rename, and Credentials. The main content area shows the Jenkins job overview. It includes a table with columns: S, W, Name, Last Success, Last Failure, and Last Duration. The table has one row for the Jenkins job, which is currently in a 'Success' state. Below the table, there are links for 'Icon: S M L', 'Icon legend', and 'Atom feed' links for all, failures, and latest builds. On the left, there are sections for 'Build Queue' (showing no builds) and 'Build Executor Status' (showing 1 idle and 2 idle executors). At the bottom right, there are links for 'REST API' and 'Jenkins 2.389'.

S	W	Name ↓	Last Success	Last Failure	Last Duration
✓	☁	Jenkins	37 sec #5	1 min 17 sec #4	1.2 sec

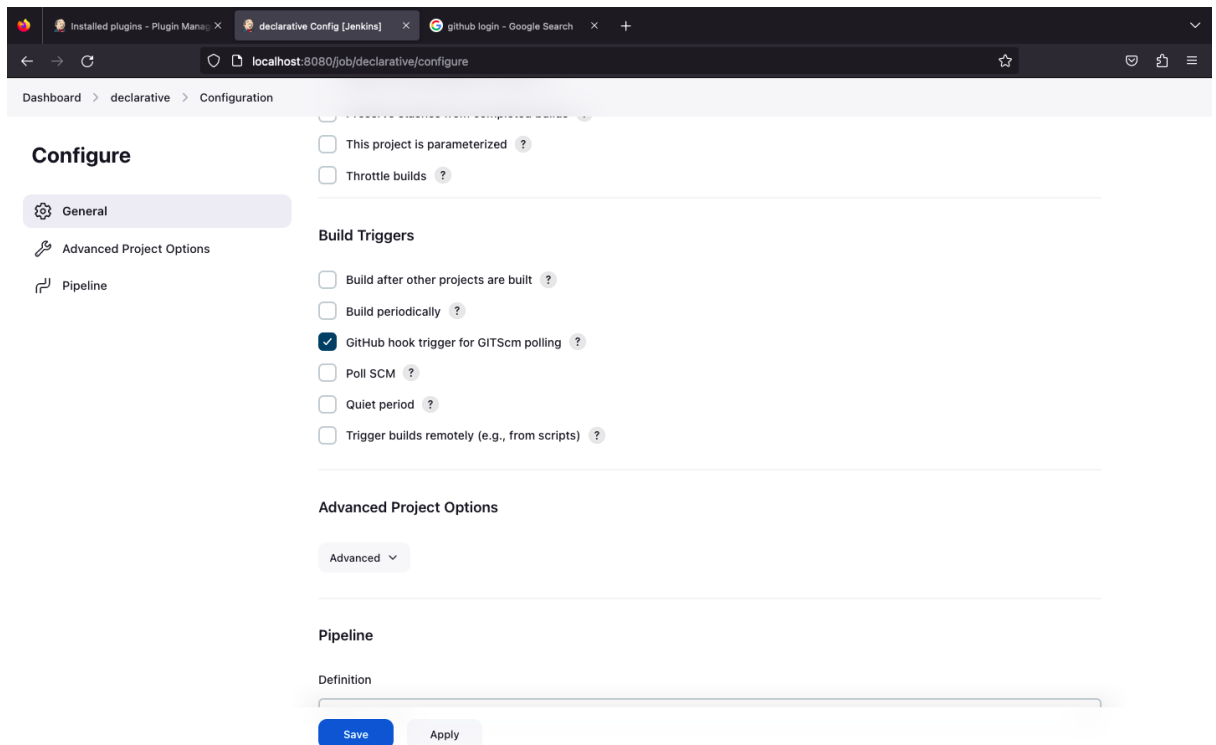
I . Console output

The screenshot shows the Jenkins web interface with the console output for the Jenkins job. The address bar indicates the URL is `localhost:8080/job/Jenkins/#5/console`. The Jenkins logo and name are at the top left. A search bar and user profile (Nisha) are at the top right. The left sidebar contains navigation links: Status, Changes, Console Output, View as plain text, Edit Build Information, Delete build '#5', and Previous Build. The main content area shows the console output for the Jenkins job. It includes a green checkmark icon and the text 'Console Output'. The output text is as follows:

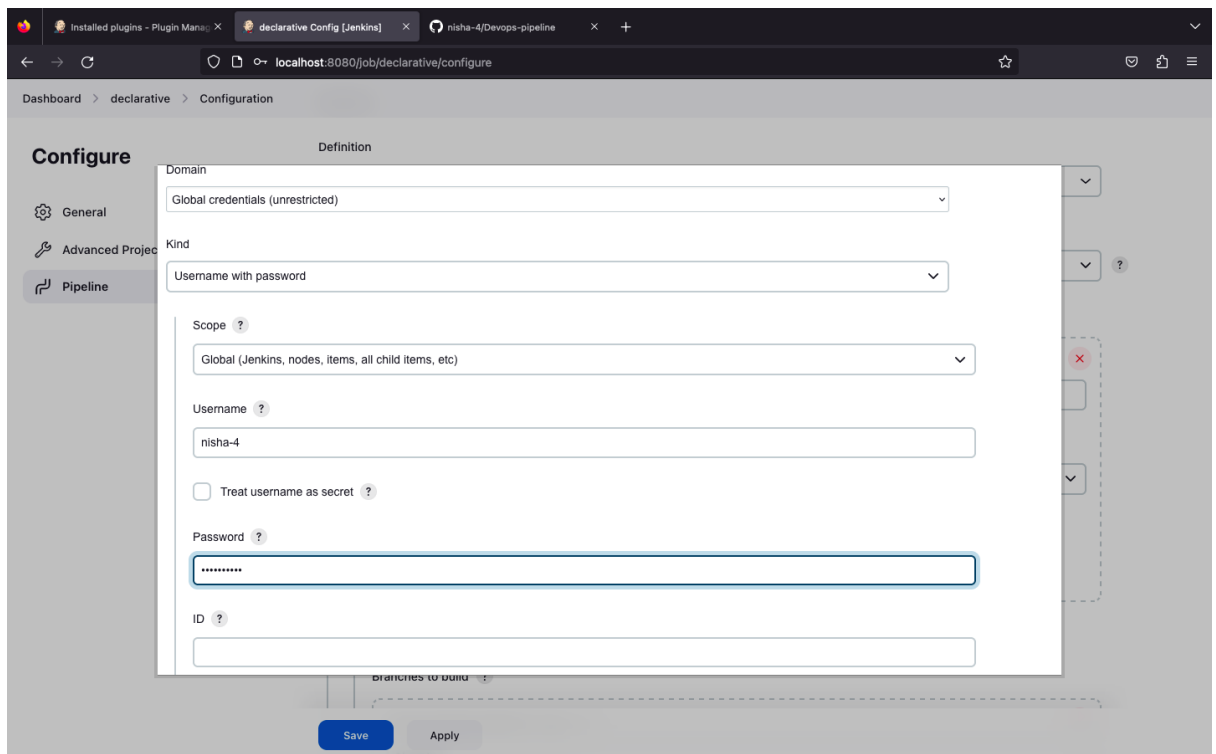
```
Started by user Nisha
Running as SYSTEM
Building in workspace /Users/nish/.jenkins/workspace/Jenkins/Jenkins
[Jenkins] $ /bin/sh -xe /var/folders/jx/hz2sgvhd5lqg73b11_3nh5ym0000gg/T/jenkins2007421358272213291.sh
+ javac --version
javac 17.0.6
Finished: SUCCESS
```

1c,2b. Integrate “Github webhook ” with Jenkins to identify the changes made in github.

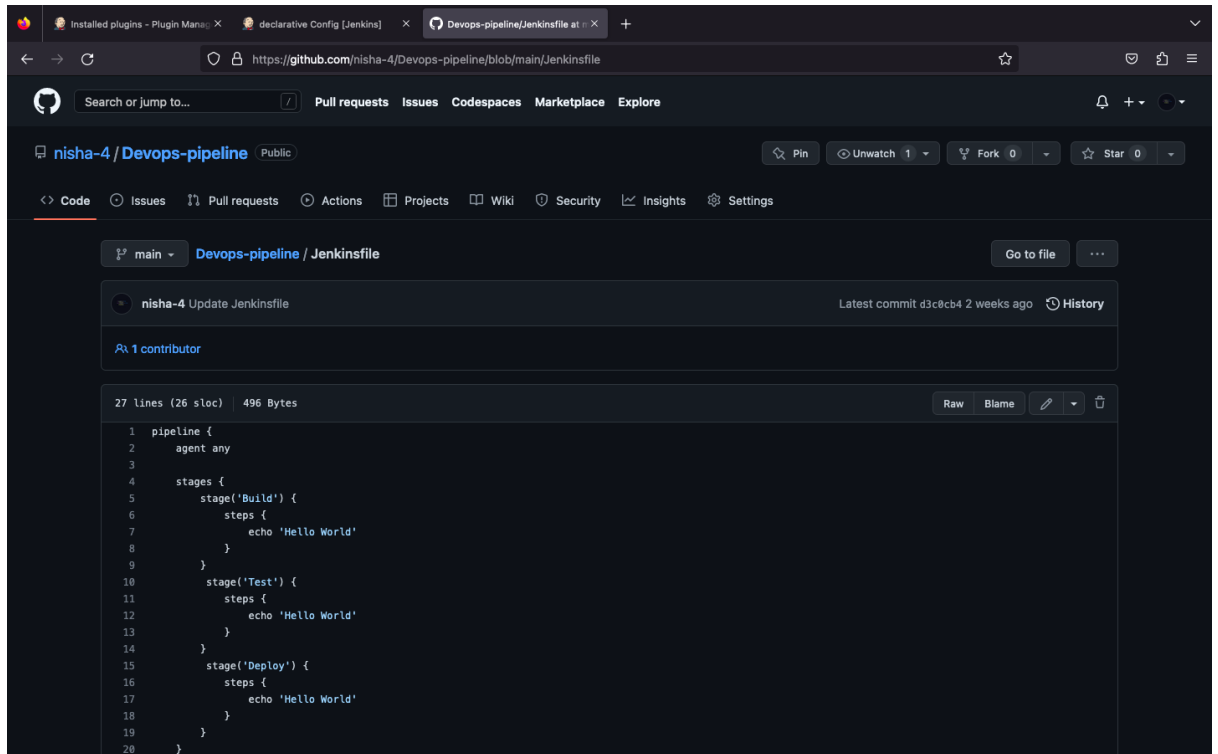
a.Create a pipeline and check for GitHub hook trigger for GITScm polling



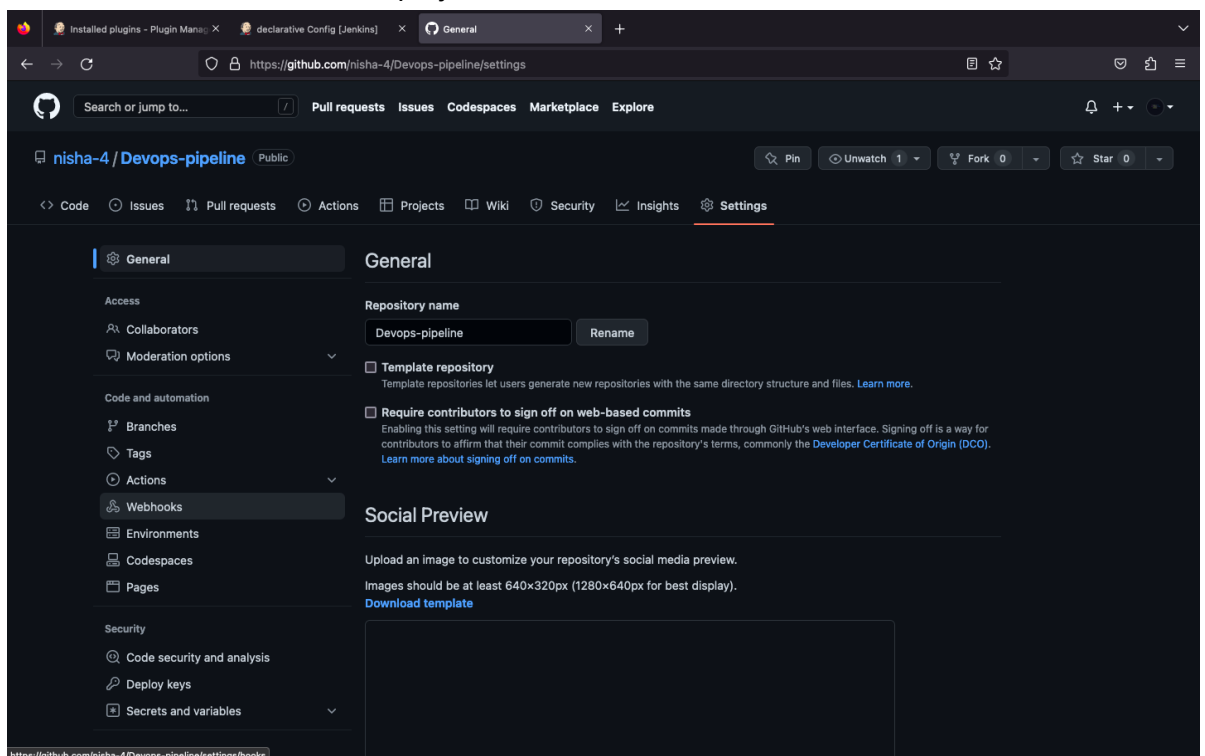
b. Add your credentials of Git



c. Copy the git repository link of Jenkinsfile



D. Select the webhook for the project



E. install ngrok and copy the forwarding link

```
ngrok
Add Single Sign-On to your ngrok dashboard via your Identity Provider: https://ngrok.com/dashSSO

Session Status      online
Session Expires     1 hour, 59 minutes
Terms of Service     https://ngrok.com/tos
Version             3.1.1
Region              India (in)
Latency              -
Web Interface        http://127.0.0.1:4040
Forwarding            https://0e64-171-76-82-146.in.ngrok.io -> http://localhost:8082

Connections
  ttl    opn    rt1    rt5    p50    p90
    0      0     0.00  0.00  0.00  0.00
```

F. Declarative pipeline gets automatically built.

The screenshot shows the Jenkins web interface for a declarative pipeline. The top navigation bar includes the Jenkins logo, a search bar, and user information (Nisha). The main content area is titled "Pipeline declarative" and includes a sidebar with actions like Status, Changes, Build Now, Configure, Delete Pipeline, Move, Full Stage View, Rename, Pipeline Syntax, and GitHub Hook Log. The "Stage View" section displays a table of stages and their average times:

Stage	Average stage times
Declarative: Checkout SCM	1s
Build	438ms
Test	254ms
Deploy	221ms
Declarative: Post Actions	14s

Below the stage view, the "Permalinks" section shows a build history table with columns for build number, time, and status. The first build is #1, dated Feb 2, 2023, 2:50 AM, and is in a successful state. The bottom of the interface shows the REST API and Jenkins version (2.389).


2c. How to run Jenkins job periodically using Jenkins scheduler? Trigger the build automatically with Jenkins scheduler. Write the scheduler pattern automatic triggering of jobs for the following

- Go to Build triggers and check the build periodically by giving the required schedule
- The project will run for the specified schedule automatically.


A . Create a job

Enter an item name


» 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 is a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

OK

Multibranch Pipeline

1. Run Every Minute

Dashboard > job2 > Configuration

Configure

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

Build Triggers

☐ Trigger builds remotely (e.g., from scripts) ?

☐ Build after other projects are built ?

☒ Build periodically ?

Schedule ?

Would last have run at Wednesday, 1 February 2023 at 05:53:51 India Standard Time; would next run at Thursday, 2 February 2023 at 05:53:51 India Standard Time.

☐ GitHub hook trigger for GITScm polling ?

☐ Poll SCM ?

Build Environment

☐ Delete workspace before build starts

☐ Use secret text(s) or file(s) ?

☐ Add timestamps to the Console Output

☐ Inspect build log for published build scans

Save

Apply

2. Run Every 15 minute

The screenshot shows the Jenkins configuration page for a job named 'job2'. The left sidebar contains a 'Configure' section with a list of tabs: General, Source Code Management, Build Triggers, Build Environment, Build Steps, and Post-build Actions. The 'Build Triggers' tab is selected. In the 'Build Triggers' section, the 'Build periodically' option is checked, and the 'Schedule' field contains the cron expression 'H/15 * * * *'. A warning message below the field states: 'Do you really mean "every minute" when you say "* * * * *"? Perhaps you meant "H * * * *"' to poll once per hour. Below this, there are checkboxes for 'Trigger builds remotely (e.g., from scripts)', 'Build after other projects are built', 'GitHub hook trigger for GITScm polling', and 'Poll SCM'. At the bottom, there are 'Save' and 'Apply' buttons.

Dashboard > job2 > Configuration

Configure

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

Build Triggers

☐ None

☐ Git ?

☐ Trigger builds remotely (e.g., from scripts) ?

☐ Build after other projects are built ?

☒ Build periodically ?

Schedule ?

H/15 * * * *

⚠ Do you really mean "every minute" when you say "* * * * *"? Perhaps you meant "H * * * *"' to poll once per hour
Would last have run at Thursday, 2 February 2023 at 02:55:16 India Standard Time; would next run at Thursday, 2 February 2023 at 02:55:16 India Standard Time.

☐ GitHub hook trigger for GITScm polling ?

☐ Poll SCM ?

Build Environment

Save Apply

3. Run Every Hour Every Day

The screenshot shows the Jenkins configuration page for a job named 'job2'. The left sidebar contains a 'Configure' section with a list of tabs: General, Source Code Management, Build Triggers, Build Environment, Build Steps, and Post-build Actions. The 'Build Triggers' tab is selected. In the 'Build Triggers' section, the 'Build periodically' option is checked, and the 'Schedule' field contains the cron expression 'H H * * * *'. Below this, there are checkboxes for 'Throttle builds', 'Execute concurrent builds if necessary', 'Trigger builds remotely (e.g., from scripts)', 'Build after other projects are built', and 'Build periodically'. At the bottom, there are 'Save' and 'Apply' buttons.

Dashboard > job2 > Configuration

Configure

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

Build Triggers

☐ Throttle builds ?

☐ Execute concurrent builds if necessary ?

Advanced ▾

Source Code Management

☐ None

☐ Git ?

Build Triggers

☐ Trigger builds remotely (e.g., from scripts) ?

☐ Build after other projects are built ?

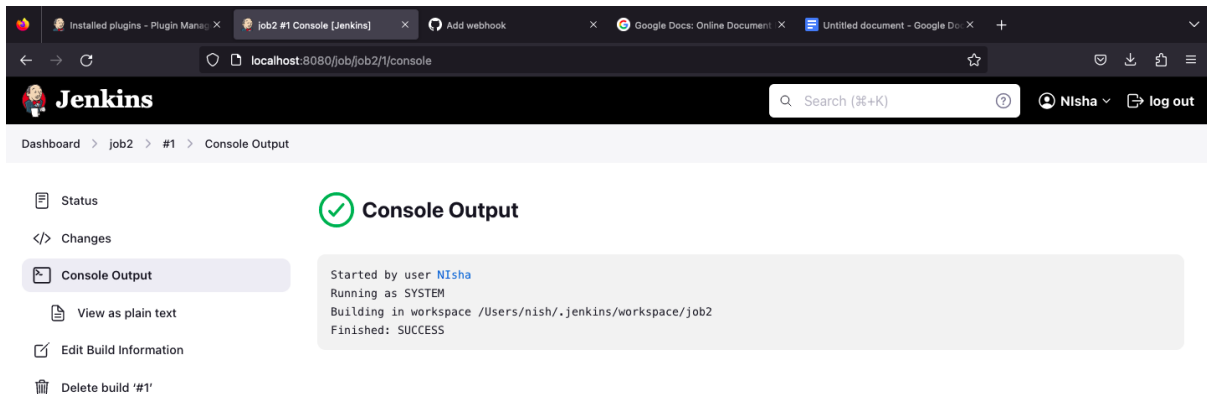
☒ Build periodically ?

Schedule ?

H H * * * *

Save Apply

B. The build is successful

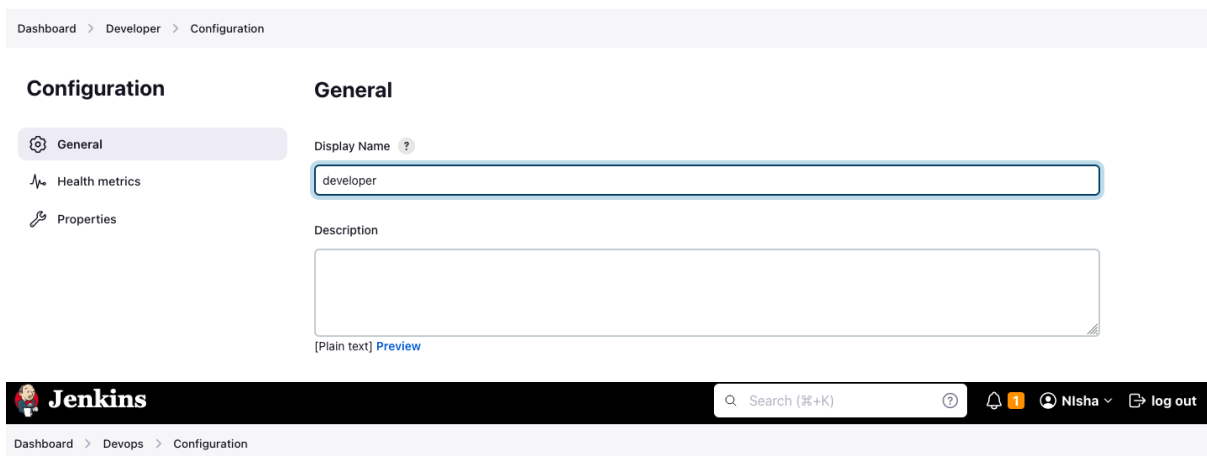


The screenshot shows the Jenkins web interface at localhost:8080/job/job2/1/console. The left sidebar has a 'Console Output' tab selected. The main area shows a green checkmark icon and the text 'Console Output'. Below this, it states: 'Started by user Nisha', 'Running as SYSTEM', 'Building in workspace /Users/nish/.jenkins/workspace/job2', and 'Finished: SUCCESS'. The top navigation bar includes a search bar and a user profile for 'Nisha' with a 'log out' button.

3b,5b . Create multiple users and Use role-based authorization strategy to Assign the permissions to the following roles

- 1. DevOps Engineer: should only create, build, configure, delete jobs under DevOps_Team folder.**
- 2. Developer: should only build, configure, delete jobs under Developer_Team folder.**
- 3. QA Tester: should only build, configure jobs under QA_Team folder. Verify the permissions.**

a. Create the required folders






The screenshot shows the Jenkins Configuration page for a new user named 'developer'. The left sidebar has a 'Configuration' tab selected. The main area has a 'General' section with a 'Display Name' field containing 'developer' and a 'Description' field. The top navigation bar includes a search bar and a user profile for 'Nisha' with a 'log out' button.



The screenshot shows the Jenkins Configuration page for a new user named 'Devops'. The left sidebar has a 'Configuration' tab selected. The main area has a 'General' section with a 'Display Name' field containing 'Devops' and a 'Description' field. The top navigation bar includes a search bar and a user profile for 'Nisha' with a 'log out' button.

Configuration

-  General
-  Health metrics
-  Properties


General

Display Name 





Description

[Plain text] [Preview](#)


b. Create manage roles


 **Jenkins**


Search (⌘+K)


   Nisha  log out


Dashboard > Manage Jenkins > Manage and Assign Roles

 New Item

 People

 Build History

 Manage Jenkins

 My Views

Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

Manage Roles


Global roles

Role	Overall	Credentials		Agent				Job				Run		View		SCM													
	Administer	Read	Create	ManageDomains	Delete	Update	View	Build	Configure	Connect	Create	Delete	Disconnect	Provision	Cancel	Configure	Create	Delete	Discover	Move	Read	Workspace	Delete	Reply	Update	Configure	Create	Delete	Tag
admin	<input checked="" type="checkbox"/>																												
developer	<input type="checkbox"/>	<input checked="" type="checkbox"/>																											
devops	<input type="checkbox"/>	<input checked="" type="checkbox"/>																											
qa	<input type="checkbox"/>	<input checked="" type="checkbox"/>																											





Role to add

Add




C . Assign roles

 **Jenkins**


Search (⌘+K)

   Nisha  log out

Dashboard > Manage Jenkins > Manage and Assign Roles

Role	Pattern	Credentials		Job				Run		View		SCM																	
		Create	Delete	ManageDomains	Update	View	Build	Cancel	Configure	Create	Delete	Discover	Move	Read	Workspace	Delete	Reply	Update	Configure	Create	Delete	Tag							
DevOps	 *dev.*																												
Developer	 *devL.*																												
qa1	 *qa.*																												

Role to add

Pattern 

Add

Agent roles

	Credentials	Agent
<div>Save</div>	<div>Apply</div>	

D. created users list

Users [Jenkins]

job2 Config [Jenkins]

Add webhook

Google Docs: Online Document

Untitled document - Google D

localhost:8080/securityRealm/

Jenkins

Search (⌘+K)

Nisha

log out

Dashboard > Jenkins' own user database

Users

Create User

These users can log into Jenkins. This is a sub set of [this list](#), which also contains auto-created users who really just made some commits on some projects and have no direct Jenkins access.

User ID ↓	Name		
admin	admin		
dev	dev		
dev1	dev1		
Nish	Nisha		
NishaSangavi	Nisha Sangavi		
qa	qa		

Jenkins 2.389

E. the final output

Jenkins

Search (CTRL+K)

Developer

log out

Dashboard > Developer >

Status

Configure

People

Build History

Project Relationship

Check File Fingerprint

Rename

Developer

Folder name: Developer1 demo

All

Add description

Build Queue

No builds in the queue.

Build Executor Status

1 Idle

2 Idle

Dashboard > qat >

Status

Configure

People

Build History

Project Relationship

Check File Fingerprint

Rename

qat

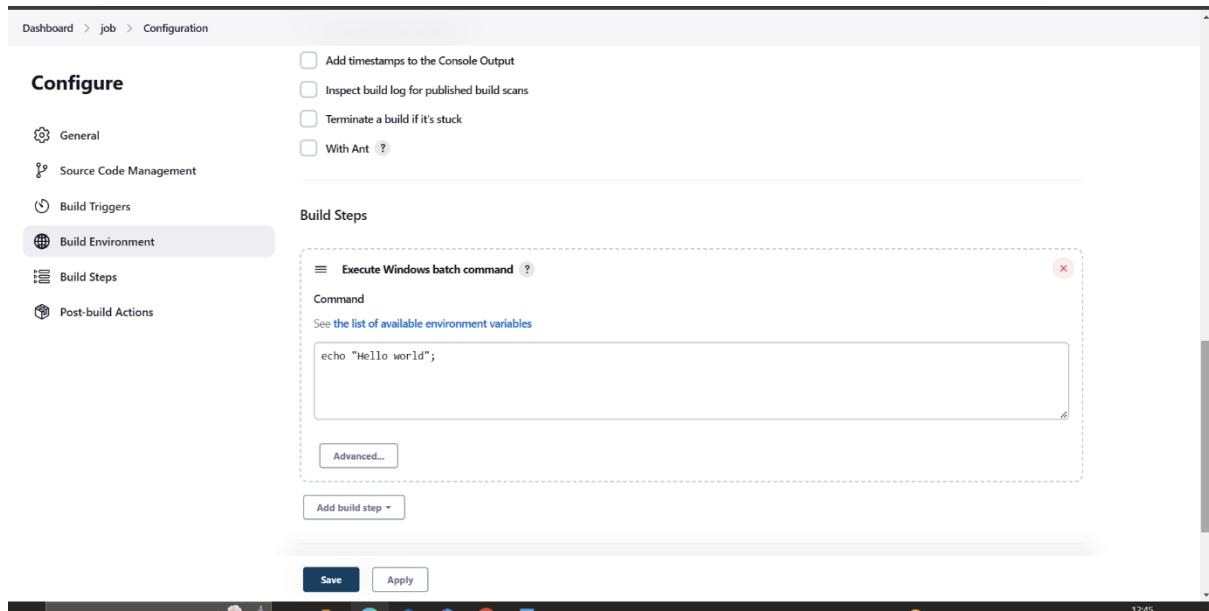
Folder name: QAT demo

All

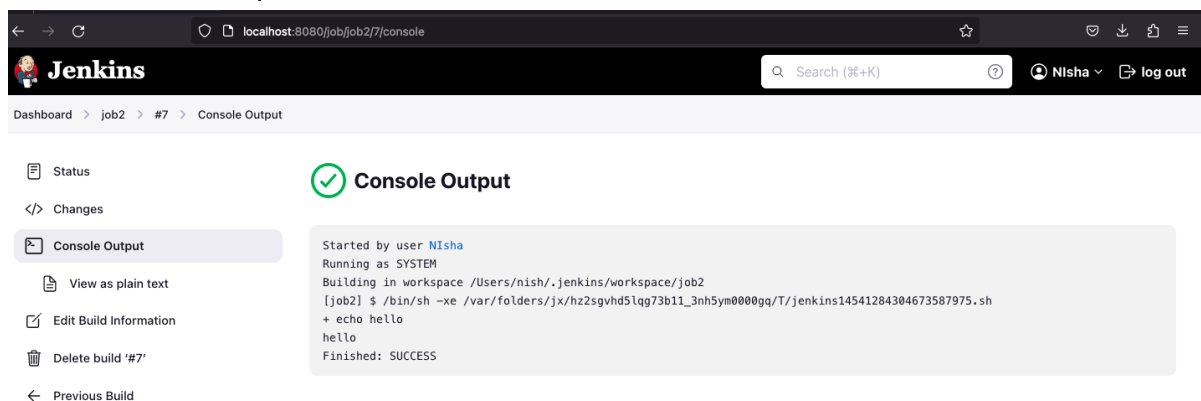
Add description

3c. Create a simple job to display simple hello world message using windows batch command. And demonstrate how to trigger Jenkins build automatically for the created job using scheduler. Implement Poll SCM as per scheduler.

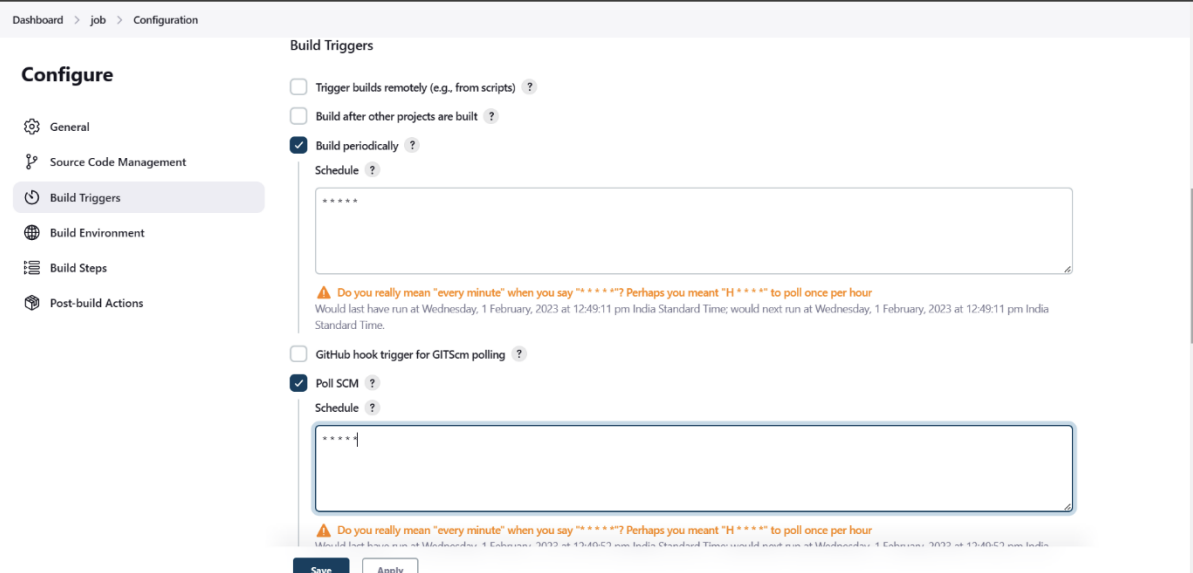
a. Create a job



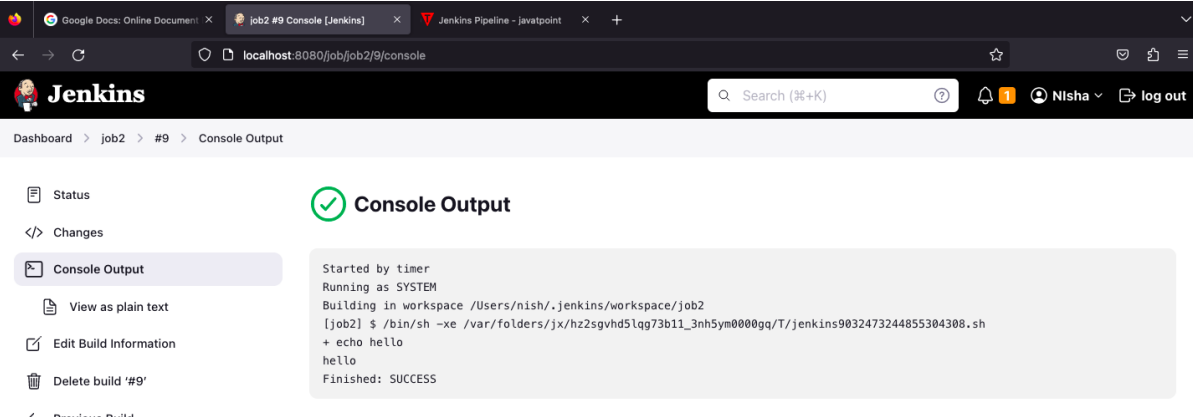
b. Console output



c. Enter the build schedule



The screenshot shows the Jenkins Configuration page for a job. The 'Build Triggers' section is active. The 'Build periodically' checkbox is checked, and the schedule field contains '****'. A warning message below the field states: 'Do you really mean "every minute" when you say "****"? Perhaps you meant "H ****" to poll once per hour'. The 'Poll SCM' checkbox is also checked, and its schedule field also contains '****' with the same warning message. The 'Save' and 'Apply' buttons are at the bottom.

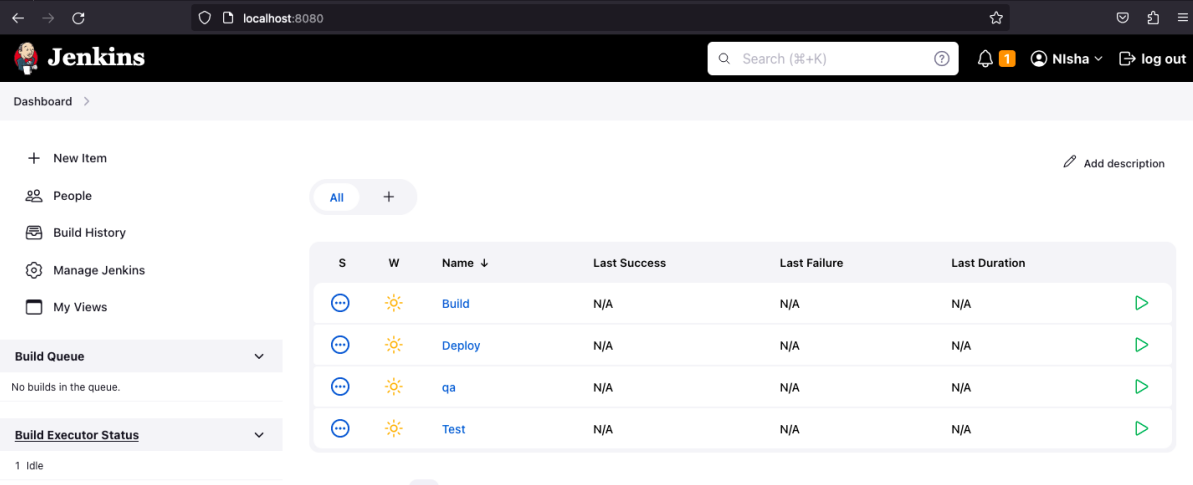


The screenshot shows the Jenkins Console Output page for a job. The 'Console Output' tab is selected. The output text is as follows:

```
Started by timer
Running as SYSTEM
Building in workspace /Users/nish/.jenkins/workspace/job2
[job2] $ /bin/sh -xe /var/folders/jx/hz2sgvhd5lqg73b11_3nh5ym0000gq/T/jenkins9032473244855304308.sh
+ echo hello
hello
Finished: SUCCESS
```

4b. Create a pipeline with at least 4 stages using plugin and demonstrate how to trigger the 3 rd and 4th stage of the created pipeline manually.

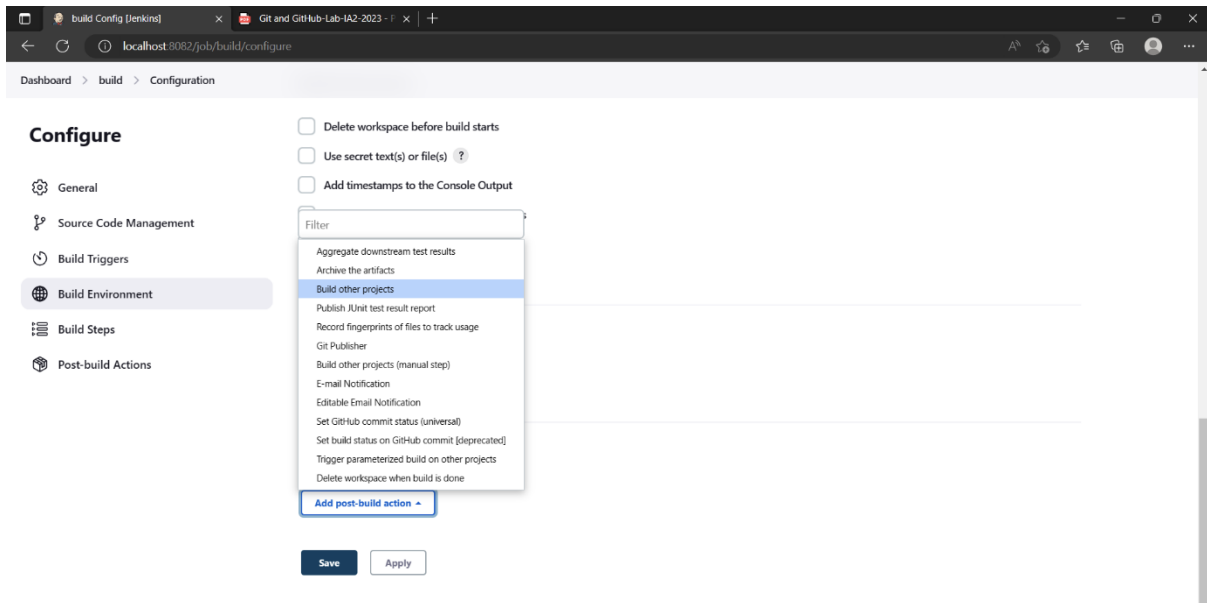
a. Create 4 jobs



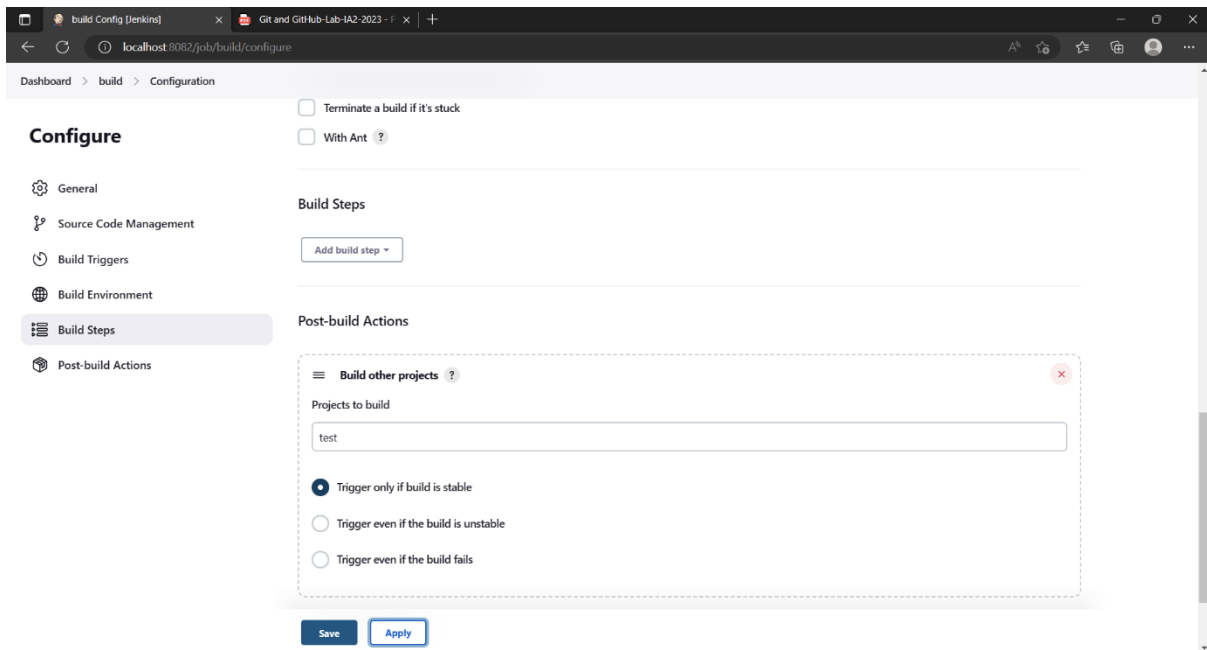
The screenshot shows the Jenkins Dashboard. The 'Build Queue' section shows 'No builds in the queue.' The 'Build Executor Status' section shows '1 Idle' and '2 Idle' executors. The 'All' tab is selected, showing a table of builds.

S	W	Name ↓	Last Success	Last Failure	Last Duration
...	☀	Build	N/A	N/A	N/A
...	☀	Deploy	N/A	N/A	N/A
...	☀	qa	N/A	N/A	N/A
...	☀	Test	N/A	N/A	N/A

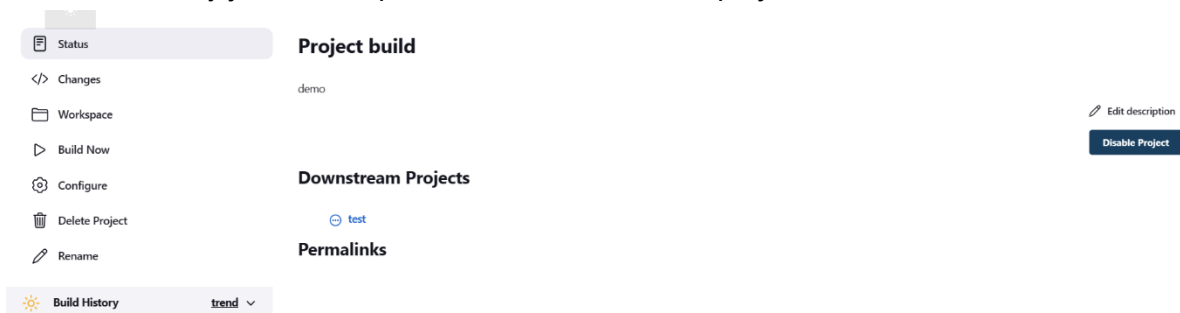
b. Create the filter



c. Enter the post build project trigger



d. For every job select upstream and downstream project



Dashboard > pipe >

+ New Item

People

Build History

Edit View

Delete View

Project Relationship

Check File Fingerprint

Manage Jenkins

My Views

Restart Safely

Name

pipe

Description ?

a demc

[Plain text] Preview

☐ Filter build queue
 ☐ Filter build executors

Build Pipeline View Title

Build Queue

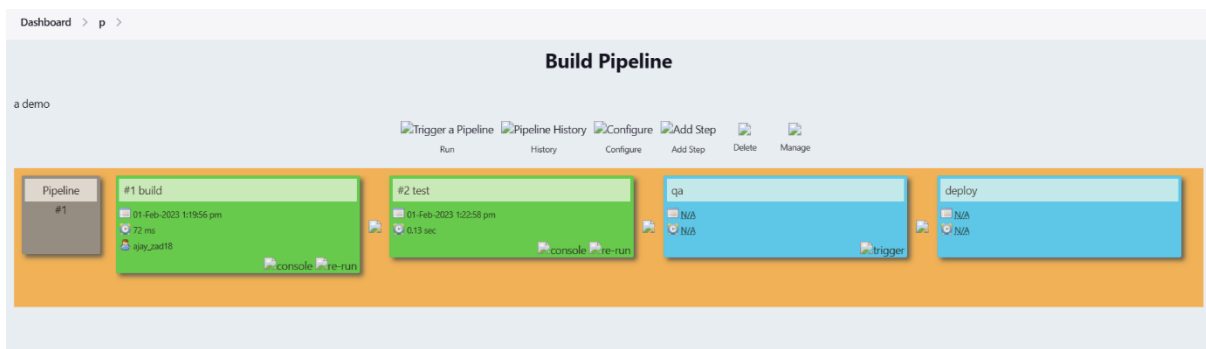
No builds in the queue.

Build Executor Status

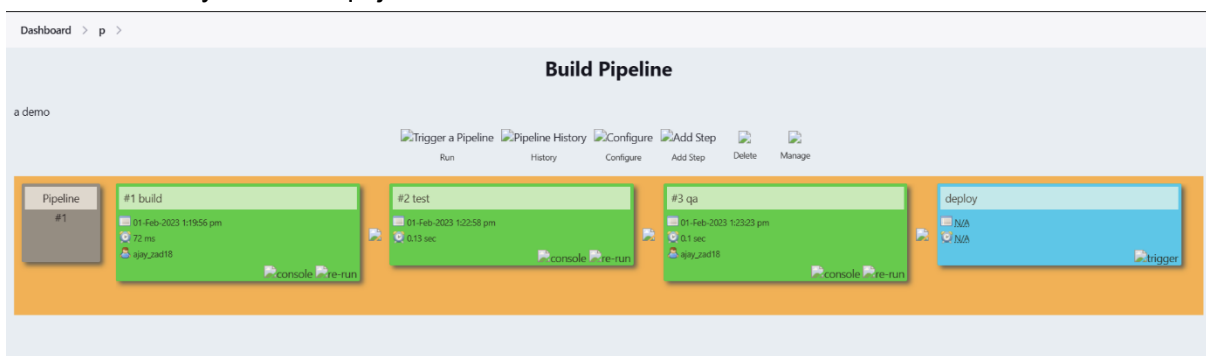
1 idle

OK Apply

e. The final output



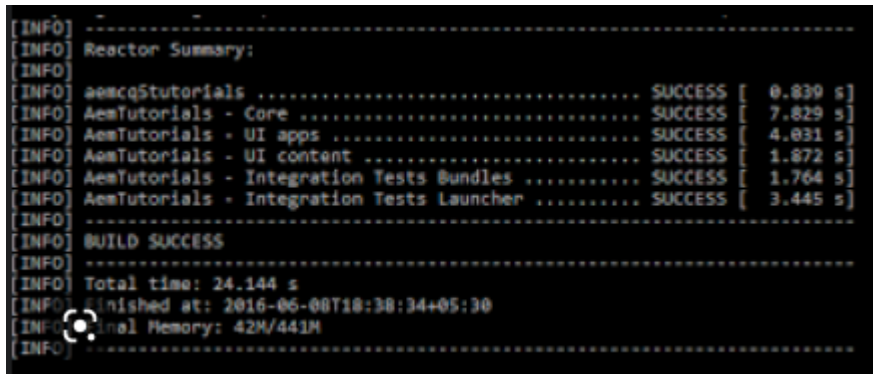
f. Manually click the qa job



5c. Build a maven project project display execute Hello world program by creating pom.xml

```
mvn archetype:generate -DgroupId=com.MSRIT.app -DartifactId=Nisha  
-DarchetypeArtifactId=maven-archetype-quickstart -DarchetypeVersion=1.4  
-DinteractiveMode=false
```

Cd Nisha



```
[INFO] -----  
[INFO] Reactor Summary:  
[INFO] aasmcqtutorials ..... SUCCESS [ 0.839 s]  
[INFO] AasmTutorials - Core ..... SUCCESS [ 7.829 s]  
[INFO] AasmTutorials - UI apps ..... SUCCESS [ 4.031 s]  
[INFO] AasmTutorials - UI content ..... SUCCESS [ 1.872 s]  
[INFO] AasmTutorials - Integration Tests Bundles ..... SUCCESS [ 1.764 s]  
[INFO] AasmTutorials - Integration Tests Launcher ..... SUCCESS [ 3.445 s]  
[INFO] -----  
[INFO] BUILD SUCCESS  
[INFO] -----  
[INFO] Total time: 24.144 s  
[INFO] Finished at: 2016-06-08T10:38:34+05:30  
[INFO] Final Memory: 42M/441M  
[INFO] -----
```

```
java -cp target/Nisha-1.0-SNAPSHOT.jar com.MSRIT.app.App
```

Output: Hello world