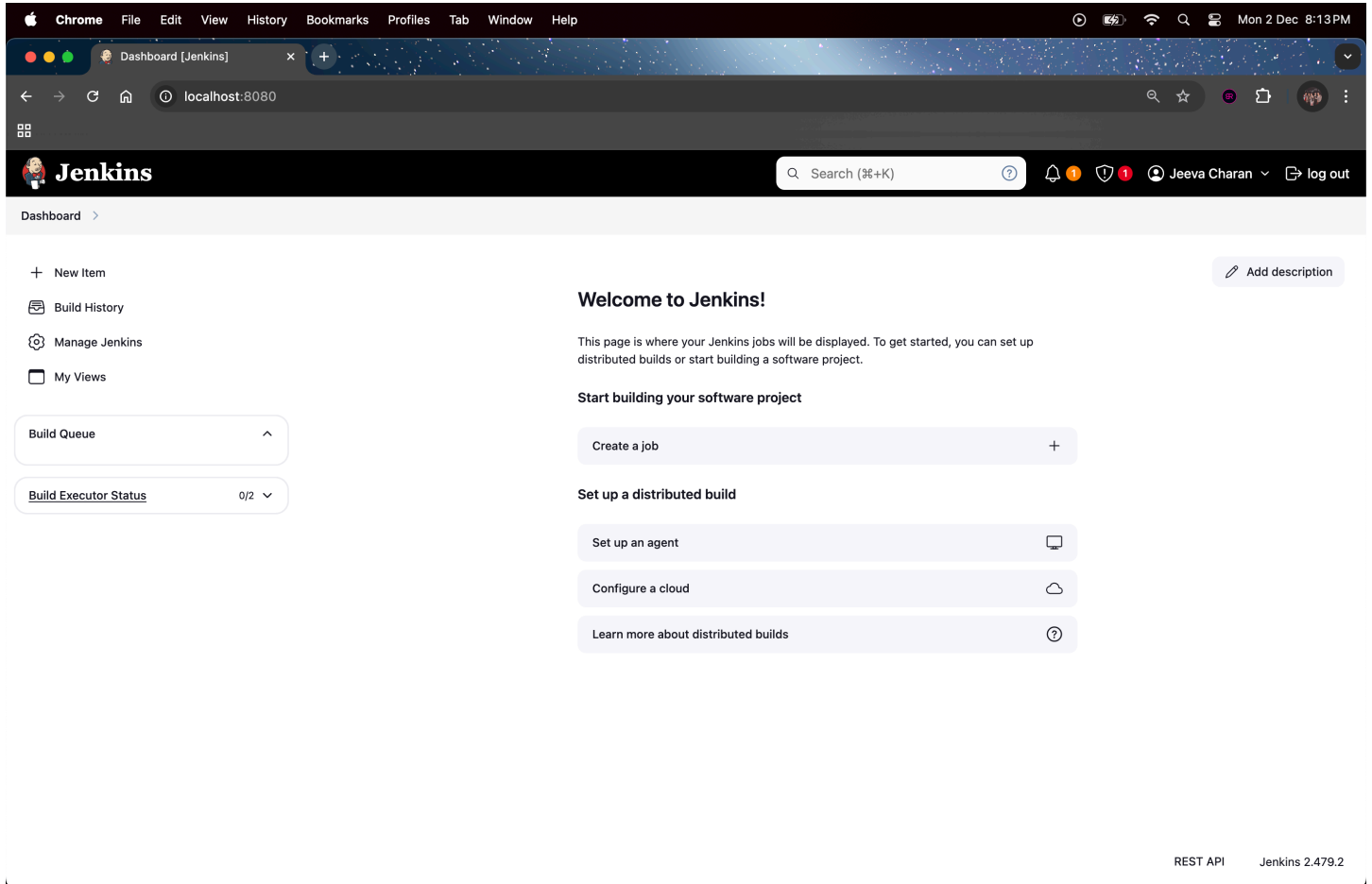


Jenkins Automation

Steps for MavenJava Automation:

Step 1: Open Jenkins (localhost:8080)

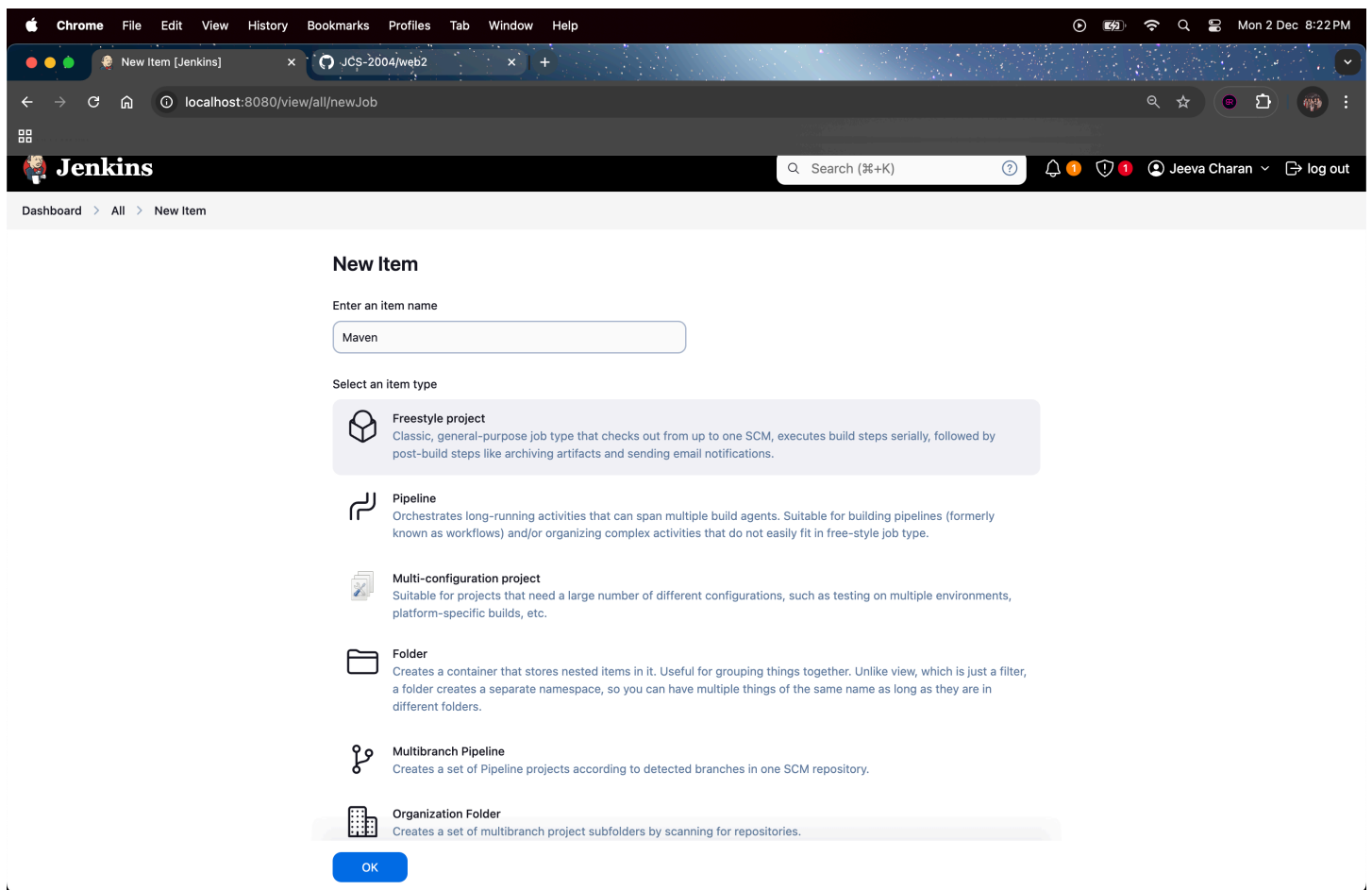
Click on "New Item" (left side menu)



Step 2: Create Freestyle Project (e.g., MavenJava_Build)

Enter project name (e.g., MavenJava_Build)

Click "OK"



Configure the project:

Description: "Java Build demo"

Source Code Management:

Git repository URL: [GitMavenJava repo URL]

Source Code Management

☐ None

☒ Git ?

Repositories ?

Repository URL ?

<https://github.com/penkulamaheshkumar/MavenJavaDemo.git>

Credentials ?

- none -

+ Add

Advanced ▾

Branches to build: */Main or */master

Branches to build ?

Branch Specifier (blank for 'any') ?

Add Branch

Repository browser ?

(Auto)

Additional Behaviours

Add ▾

Build Steps:

Add Build Step -> "Invoke top-level Maven targets"

Maven version: MAVEN_HOME

Goals: clean

Add Build Step -> "Invoke top-level Maven targets"

Maven version: MAVEN_HOME

Goals: install

Build Steps

≡ **Invoke top-level Maven targets** ?

Goals

Advanced ▾

≡ **Invoke top-level Maven targets** ?

Goals

Advanced ▾

Add build step ▾

Post-build Actions:

Add Post Build Action -> "Archive the artifacts"

Files to archive: **/*

Add Post Build Action -> "Build other projects"

Projects to build: MavenJava_Test

Trigger: Only if build is stable

Apply and Save

Post-build Actions

Archive the artifacts ?

Files to archive ?

**/*

Advanced ▾

Build other projects ?

Projects to build

MavenJava_Test

❗ No such project 'MavenJava_Test'. Did you mean 'MavenJava_Build'?

☒ Trigger only if build is stable

☐ Trigger even if the build is unstable

☐ Trigger even if the build fails

Add post-build action ▾

Save Apply

Step 3: Create Freestyle Project (e.g., MavenJava_Test)

Chrome File Edit View History Bookmarks Profiles Tab Window Help

localhost:8080/view/all/newJob

Jenkins

Search (⌘+K)

Dashboard > All > New Item

New Item

Enter an item name

MavenJava_Test

Select an item type

Freestyle project

Classic, general-purpose job type that checks out from up to one SCM, executes build steps serially, followed by post-build steps like archiving artifacts and sending email notifications.

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

OK

Enter project name (e.g., MavenJava_Test)

Click "OK"

Configure the project:

Description: "Test demo"

Build Environment:

Check: "Delete the workspace before build starts"

Add Build Step -> "Copy artifacts from another project"

Project name: MavenJava_Build

Build: Stable build only // tick at this

Artifacts to copy: **/*

Build Steps

≡

Copy artifacts from another project

×

Project name ?
MavenJava_Build

Which build ?
Latest successful build

☒ Stable build only

Artifacts to copy ?
**/*

Artifacts not to copy ?

Add Build Step -> "Invoke top-level Maven targets"

Maven version: MAVEN_HOME

Goals: test

Post-build Actions:

Add Post Build Action -> "Archive the artifacts"

Files to archive: **/*

Apply and Save

Invoke top-level Maven targets ?

Maven Version

MAVEN_HOME

Goals

test

Advanced

Add build step

Post-build Actions

Archive the artifacts ?

Files to archive ?

**/*

Save

Apply

Step 4: Create Pipeline View for Maven Java project

Click "+" beside "All" on the dashboard

Enter name: MavenJava_Pipeline

Select "Build pipeline view" // tick here

Chrome

File

Edit

View

History

Bookmarks

Profiles

Tab

Window

Help

Edit View [Jenkins]

Java Archive Downloads - Ja

copy path in mac - Google Se

ChatGPT

localhost:8080/user/jcs/my-views/view/MavenJava_Pipeline/configure

Jenkins

Search (⌘+K)

1

2

Jeeva Charan

log out

Dashboard

Jeeva Charan

My Views

MavenJava_Pipeline

Configure

+ New Item

Build History

Edit View

Delete View

Build Queue

Build Executor Status

Name

MavenJava_Pipeline

Description ?

Plain text

Preview

☐ Filter build queue ?
 ☐ Filter build executors ?

Build Pipeline View Title

Pipeline Flow

Layout

Based on upstream/downstream relationship

This layout mode derives the pipeline structure based on the upstream/downstream trigger relationship between jobs. This is the only out-of-the-box supported layout mode, but is open for extension.

Upstream / downstream config

OK

Apply

create

Pipeline Flow:

Layout: Based on upstream/downstream relationship

Initial job: MavenJava_Build

Apply and Save OK

Step 5: Run the Pipeline and Check Output

Click on the trigger to run the pipeline

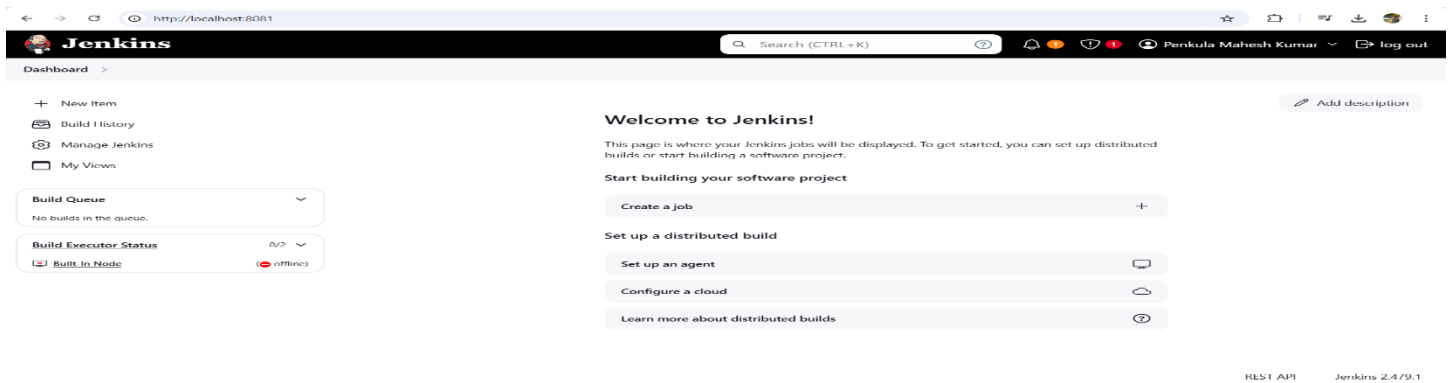
Click on the small black box to open the console to check if the build is success

The screenshot shows the Jenkins web interface. At the top is the Jenkins logo and a search bar. Below the header, the breadcrumb trail reads "Dashboard > MavenJava_Pipeline >". The main section is titled "Build Pipeline" and contains a toolbar with icons for Run, History, Configure, Add Step, Delete, and Manage. The pipeline itself is visualized as a sequence of steps on a light blue background. The first step, "Pipeline #1", is shown in a grey box. The second step, "#1 MavenJava_Build", is a green box indicating a successful build, with details: "28-Nov-2024 10:38:18 am", "3 min 31 sec", and user "mahesh1213". An arrow points from this step to the third step, "#1 MavenJava_Test", which is also a green box indicating success, with details: "28-Nov-2024 10:41:55 am" and "16 sec". At the bottom right, the text "REST API Jenkins 2.479.1" is visible.

Maven Web Automation Steps:

Step 1: Open Jenkins (localhost:8080)

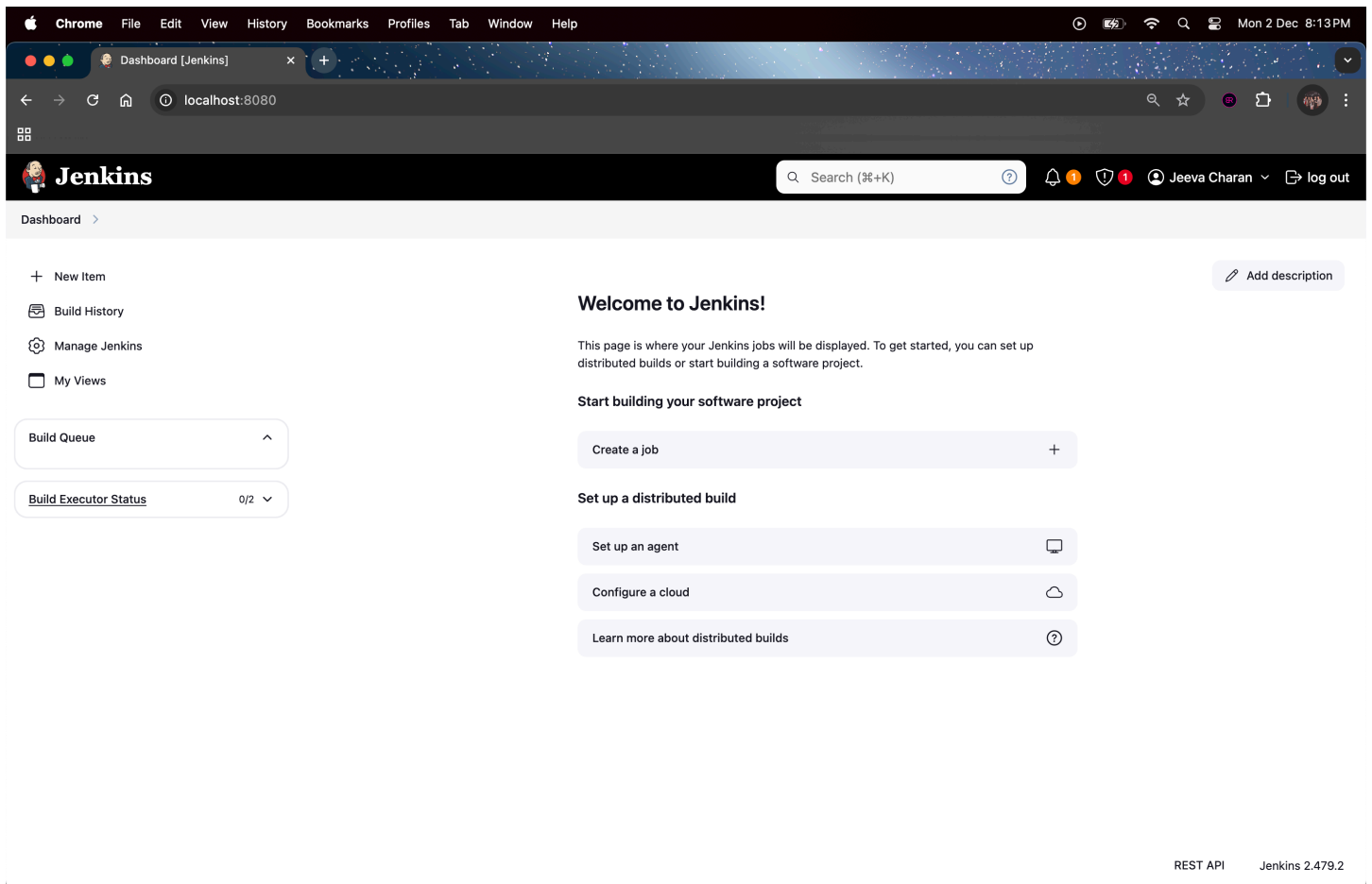
Click on "New Item" (left side menu)



Step 2: Create Freestyle Project (e.g., MavenWeb_Build)

Enter project name (e.g., MavenWeb_Build)

Click "OK"



Configure the project:

Description: "Web Build demo"

Source Code Management:

Git repository URL: [GitMavenWeb repo URL]

Branches to build: */Main or master

Source Code Management

☐ None

☒ Git ?

Repositories ?

Repository URL ?

https://github.com/penkulamaheshkumar/MavenWebDemo.git

Credentials ?

- none -

+ Add

Advanced ▾

Build Steps:

Add Build Step -> "Invoke top-level Maven targets"

Maven version: MAVEN_HOME

Goals: clean

Add Build Step -> "Invoke top-level Maven targets"

Maven version: MAVEN_HOME

Goals: install

Build Steps

The first screenshot shows the 'Invoke top-level Maven targets' configuration with 'Maven Version' set to 'MAVEN_HOME' and 'Goals' set to 'clean'. The second screenshot shows the same configuration but with 'Goals' set to 'install'.

Post-build Actions:

Add Post Build Action -> "Archive the artifacts"

Files to archive: **/*

Add Post Build Action -> "Build other projects"

Projects to build: MavenWeb_Test

Trigger: Only if build is stable

Apply and Save

Post-build Actions

The first screenshot shows the 'Archive the artifacts' action with 'Files to archive' set to '**/*'. The second screenshot shows the 'Build other projects' action with 'Projects to build' set to 'MavenWeb_Test' and the trigger set to 'Trigger only if build is stable'. Below the second screenshot are 'Save' and 'Apply' buttons.

Step 3: Create Freestyle Project (e.g., MavenWeb_Test)

Enter project name (e.g., MavenWeb_Test)

Click "OK"

Configure the project:

Description: "Test demo"

Build Environment:

Check: "Delete the workspace before build starts"



Add Build Step -> "Copy artifacts from another project"


Project name: MavenWeb_Build


Build: Stable build only


Artifacts to copy: **/*

Build Steps


 **Copy artifacts from another project** 


Project name 

Which build 

Latest successful build 

☒ Stable build only




Artifacts to copy 

Artifacts not to copy 


Add Build Step -> "Invoke top-level Maven targets"

Maven version: MAVEN_HOME


Goals: test


 **Invoke top-level Maven targets**  

Maven Version

MAVEN_HOME 

Goals

test 

Advanced 

Post-build Actions:

Add Post Build Action -> "Archive the artifacts"

Files to archive: **/*

Add Post Build Action -> "Build other projects"

Projects to build: MavenWeb_Deploy

Apply and Save

Post-build Actions

Archive the artifacts ?

Files to archive ?

**/*

Advanced ▾

Build other projects ?

Projects to build

MavenWeb_Deploy

☒ Trigger only if build is stable

☐ Trigger even if the build is unstable

☐ Trigger even if the build fails

Add post-build action ▾

Save Apply

Step 4: Create Freestyle Project (e.g., MavenWeb_Deploy)

Enter project name (e.g., MavenWeb_Deploy)

Click "OK"

Configure the project:

Description: "Web Code Deployment"

Build Environment:

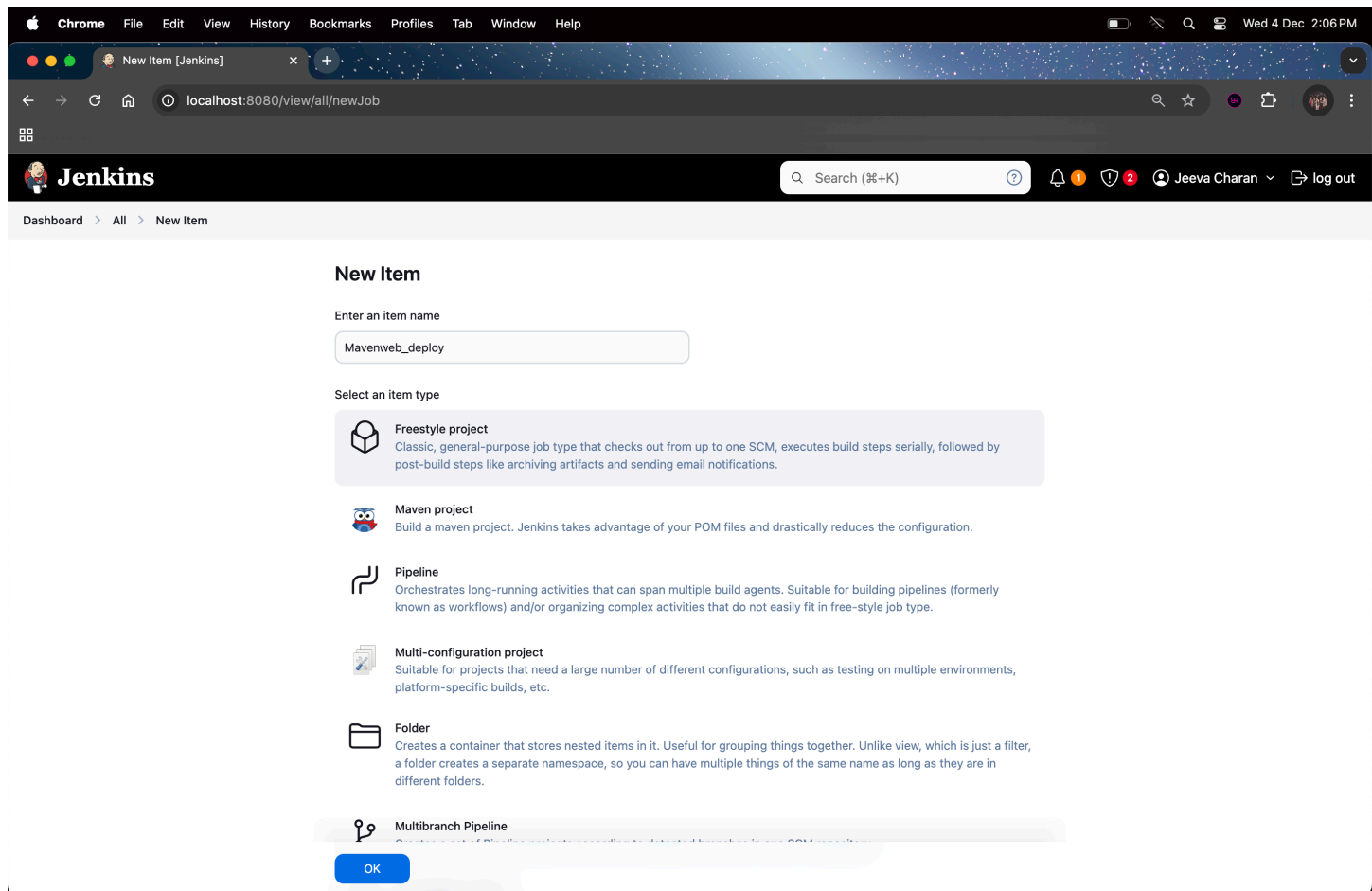
Check: "Delete the workspace before build starts"

Add Build Step -> "Copy artifacts from another project"

Project name: MavenWeb_Test

Build: Stable build only

Artifacts to copy: **/*



Build Steps

Copy artifacts from another project

Project name ?
MavenWeb_Test

No such project 'MavenWeb_Test'. Did you mean 'MavenJava_Test'?

Which build ?
Latest successful build

☒ Stable build only

Artifacts to copy ?
**/*

Artifacts not to copy ?

Post-build Actions:

Add Post Build Action -> "Deploy WAR/EAR to a container"

WAR/EAR File: **/*.war

Context path: Webpath

Add container -> Tomcat 9.x remote

Credentials: Username: admin, Password: 1234

Tomcat URL: https://localhost:8085/

Apply and Save

Deploy war/ear to a container

WAR/EAR files ?
**/*.war

Context path ?
Webpath

Containers

Tomcat 9.x Remote

Credentials
- none -

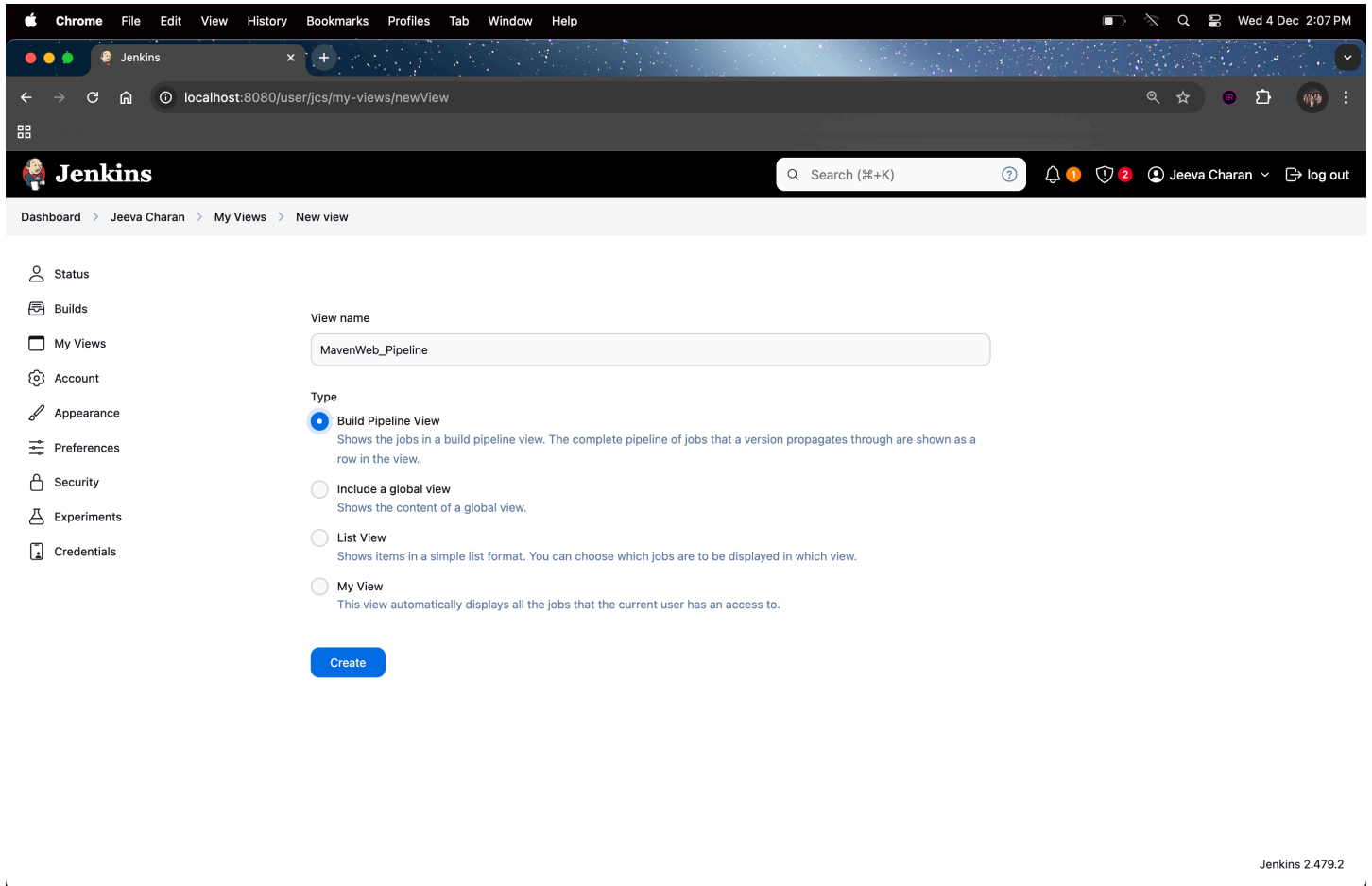
+ Add

Tomcat URL ?
https://localhost:8081/

Step 5: Create Pipeline View for MavenWeb

Click "+" beside "All" on the dashboard

Enter name: MavenWeb_Pipeline



Select "Build pipeline view"

Pipeline Flow:

Layout: Based on upstream/downstream relationship

Initial job: MavenWeb_Build

Apply and Save

Pipeline Flow

Layout

Based on upstream/downstream relationship

This layout mode derives the pipeline structure based on the upstream/downstream trigger relationship between jobs. This is the only out-of-the-box supported layout mode, but is open for extension.

Upstream / downstream config

Select Initial Job ?

MavenWeb_Build

Trigger Options

Build Cards

Standard build card

Use the default build cards

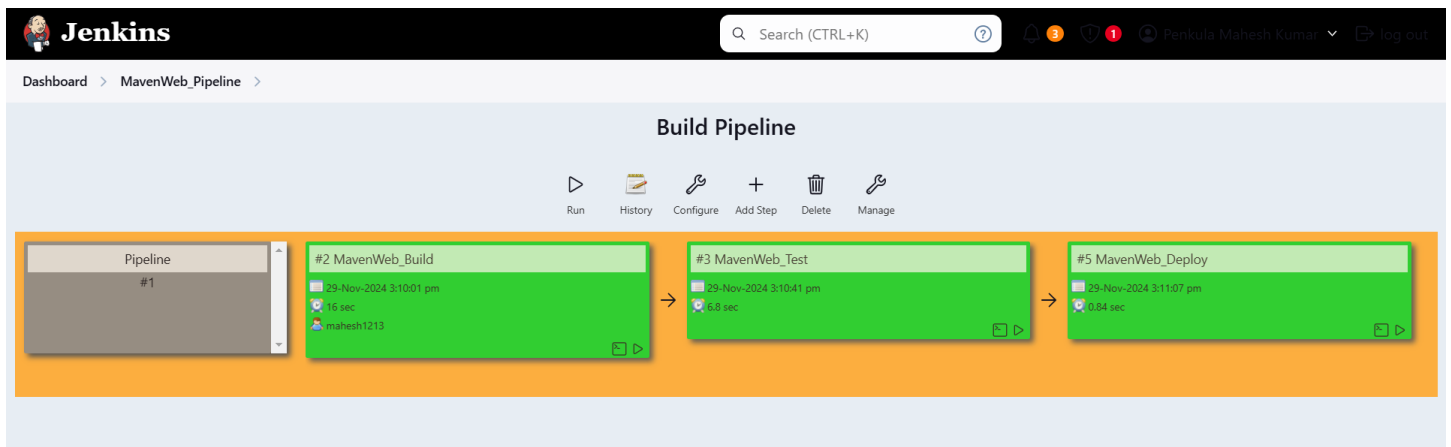
Restrict triggers to most recent successful builds ?

OK

Apply

Step 6: Run the Pipeline and Check Output

Click on the trigger “**RUN**” to run the pipeline



The image shows the Jenkins web interface for a pipeline named 'MavenWeb_Pipeline'. The top navigation bar includes the Jenkins logo, a search bar, and user information. The main content area displays the 'Build Pipeline' view, which shows a sequence of three build steps: #1 Pipeline, #2 MavenWeb_Build, #3 MavenWeb_Test, and #5 MavenWeb_Deploy. Each step is represented by a green card with a play button icon. The cards are connected by arrows, indicating the flow of the pipeline. The first card, #1 Pipeline, is currently selected and expanded, showing its details. The other cards show their status, timestamp, duration, and the user who triggered the build.

Jenkins Search (CTRL+K) Penkula Mahesh Kumar log out

Dashboard > MavenWeb_Pipeline >

Build Pipeline

Run History Configure Add Step Delete Manage

Pipeline #1

#2 MavenWeb_Build
29-Nov-2024 3:10:01 pm
16 sec
mahesh1213

#3 MavenWeb_Test
29-Nov-2024 3:10:41 pm
6.8 sec

#5 MavenWeb_Deploy
29-Nov-2024 3:11:07 pm
0.04 sec