TECHNOLOGY

Computing

Caltech | Center for Technology & Management Education

Jenkins and Docker
Angular Web Apps

Powerd by simplilearn

TECHNOLOGY

Computing

Caltech | Center for Technology & Management Education

Set Up Jenkins

Powerd by simplilearn

You Already Know

Before we begin, let's recall what we have covered till now:

Agile

Git

SQL

Angular

HTML

CSS

JavaScript

Core java

You Already Know

Before we begin, let's recall what we have covered till now:

JDBC

JSP

Servlets

MongoDB

Maven

You Already Know

Before we begin, let's recall what we have covered till now:

JUnit

Spring

Spring Boot

Webservices

Microservices

You Already Know

Before we begin, let's recall what we have covered till now:

Docker

Jenkins

AWS

Recap

Developed Angular Front End

- Created various components and services for the Angular eStore admin and end user.

Developed Java Back End

- Developed the Java back end for the admin and the end-user
- Implemented microservices with Spring Boot

Front-End Back-End Communication

- Used HTTP Client in Angular to communicate with the Java back end
- Implemented communication for both the admin and the end-user projects

A Day in the Life of a Full Stack Developer

As a full stack web developer, our key role is to develop both client and server software.

- Angular and Node can be used to build the front end of the web page.

- Spring Boot, Java, and MySQL or MongoDB can be used to build at the back end.

A Day in the Life of a Full Stack Developer

Bob needs to develop Angular web apps for the admin and the end user.
He brainstorms a bit and finds a solution.

Let me use Jenkins, Docker, and AWS to build CI/CD Pipelines, containerize the apps, and finally host them to AWS EC2 instance.

In this lesson, we will create Jenkins Pipeline for CI/CD for Java back end. Moving ahead, we will create a Dockerfile to build images, run them as containers in Docker, and help Bob complete his task effectively and quickly.

Learning Objectives

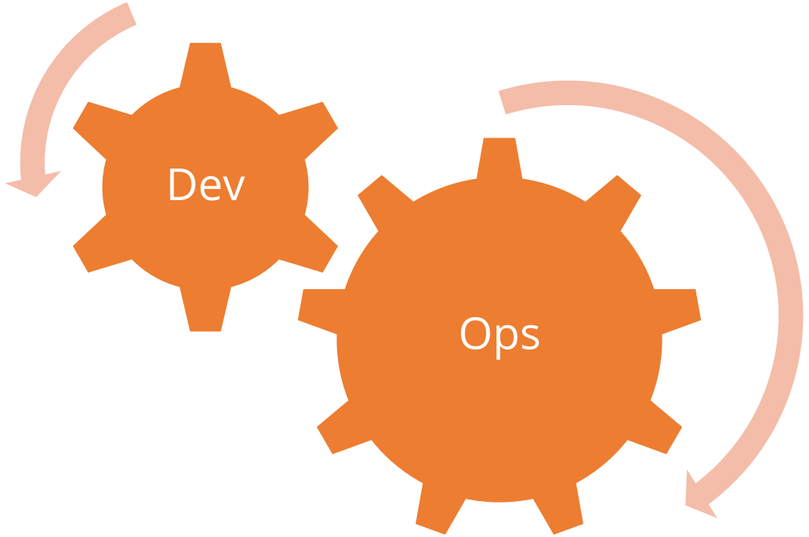
By the end of this lesson, you will be able to:

- Create Jenkins Pipeline
- Implement CI/CD in Jenkins with Git
- Create a Dockerfile for building images
- Integrate Docker in Jenkins to build and release the images as containers

Set Up Jenkins for the Angular Web-Admin Project

Jenkins

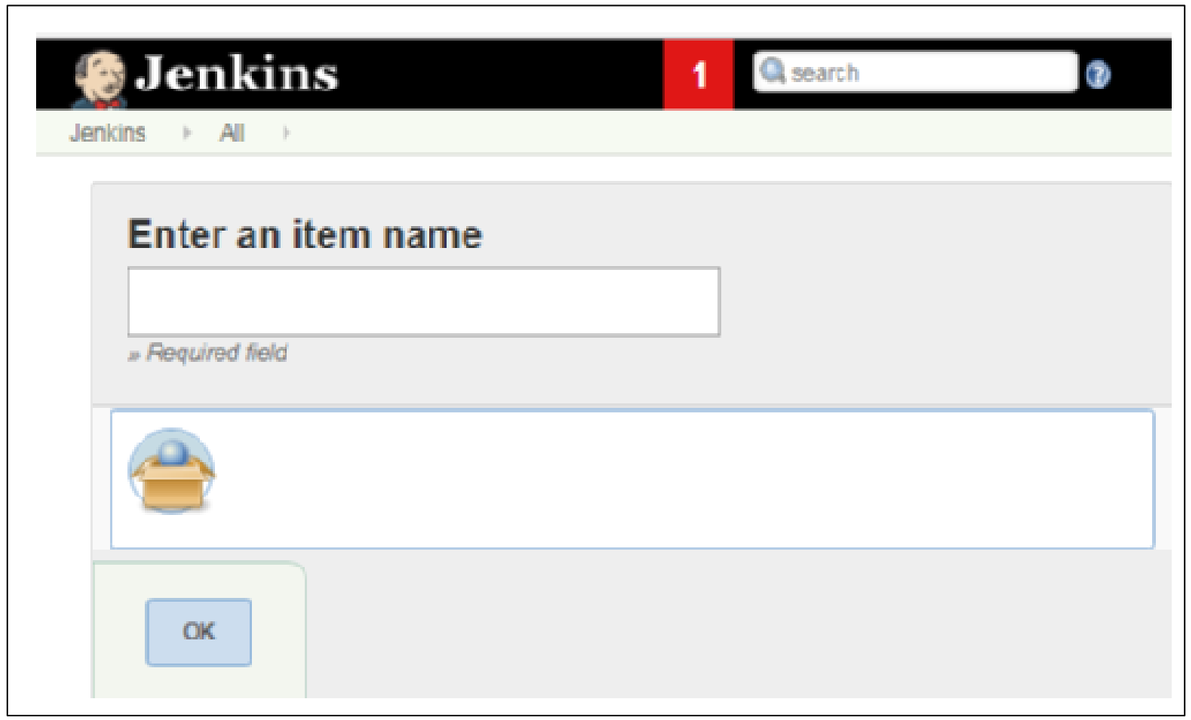
Jenkins is an automation build server used for tasks such as; building, testing, and deploying software.



Jenkins plays an important role in developing continuous integration and continuous delivery pipelines.

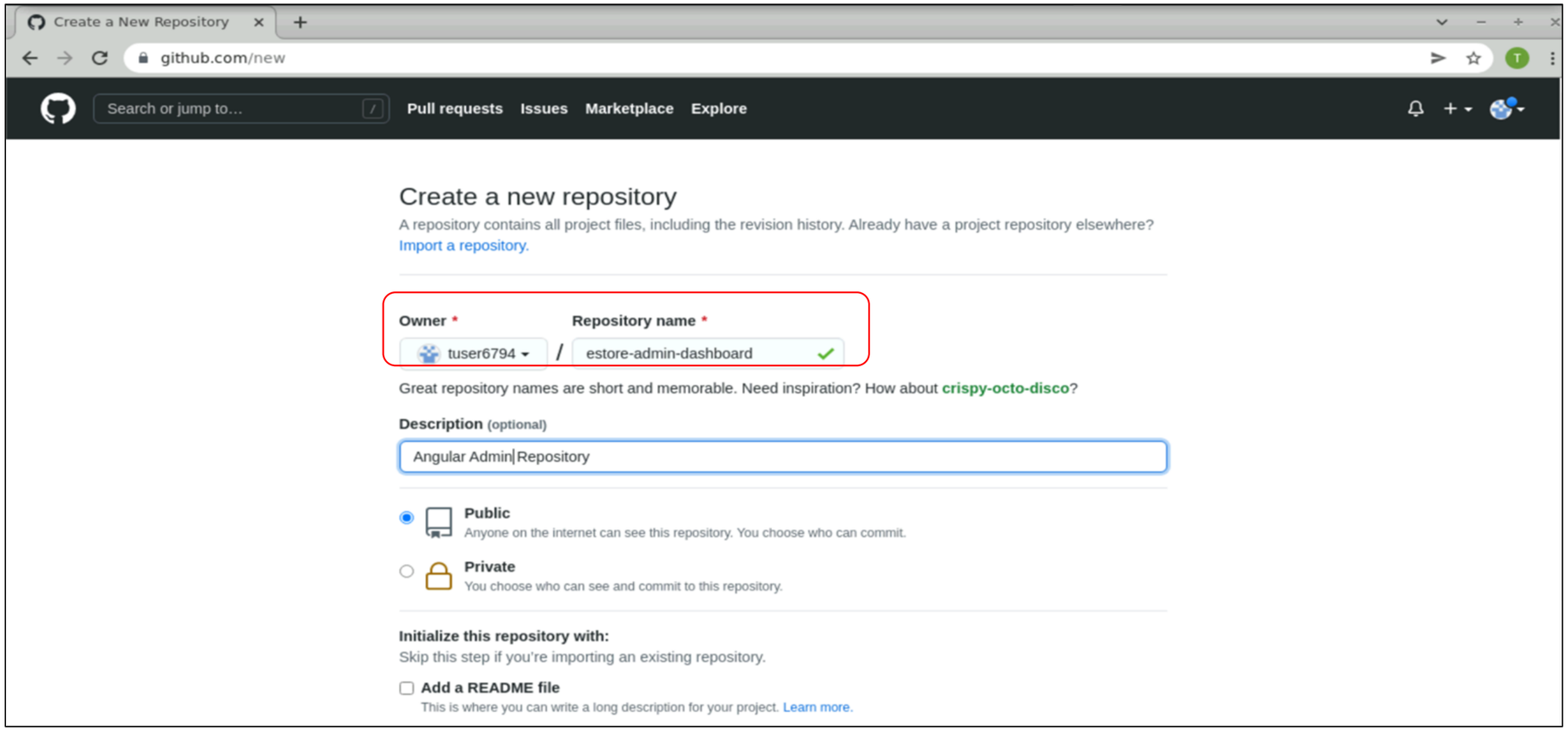
Setting Up Jenkins Environment

Jenkins environment is needed to create the NodeJS project using Jenkinsfile for the build.



Create Git Repository on GitHub for the Web-Admin Project

Create a new repository on your GitHub account which will be used by Jenkins to sync the code in SCM.



Sync the Project on GitHub for the Web-Admin Project

Push the code on GitHub for the web-admin project.

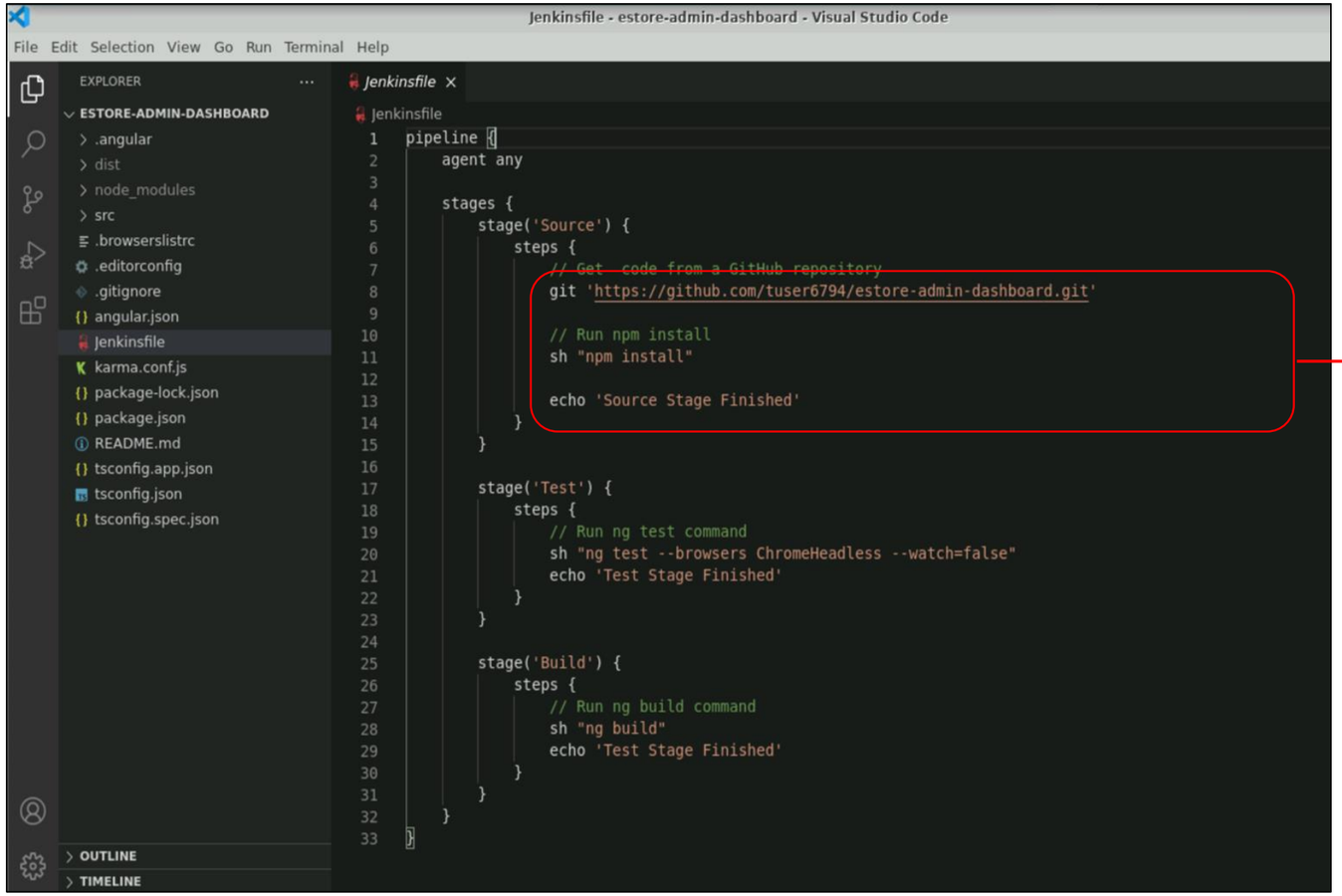
erishantgmail@ip-172-31-84-97: ~/Downloads/estore-admin-dashboard

File Edit View Search Terminal Help

```
erishantgmail@ip-172-31-84-97:~/Downloads/estore-admin-dashboard$ git push -u origin maste
r
Enumerating objects: 82, done.
Counting objects: 100% (82/82), done.
Delta compression using up to 4 threads
Compressing objects: 100% (77/77), done.
Writing objects: 100% (82/82), 213.56 KiB | 7.36 MiB/s, done.
Total 82 (delta 14), reused 0 (delta 0)
remote: Resolving deltas: 100% (14/14), done.
To https://github.com/tuser6794/estore-admin-dashboard.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
erishantgmail@ip-172-31-84-97:~/Downloads/estore-admin-dashboard$
```

Configure Jenkins Pipeline Stages

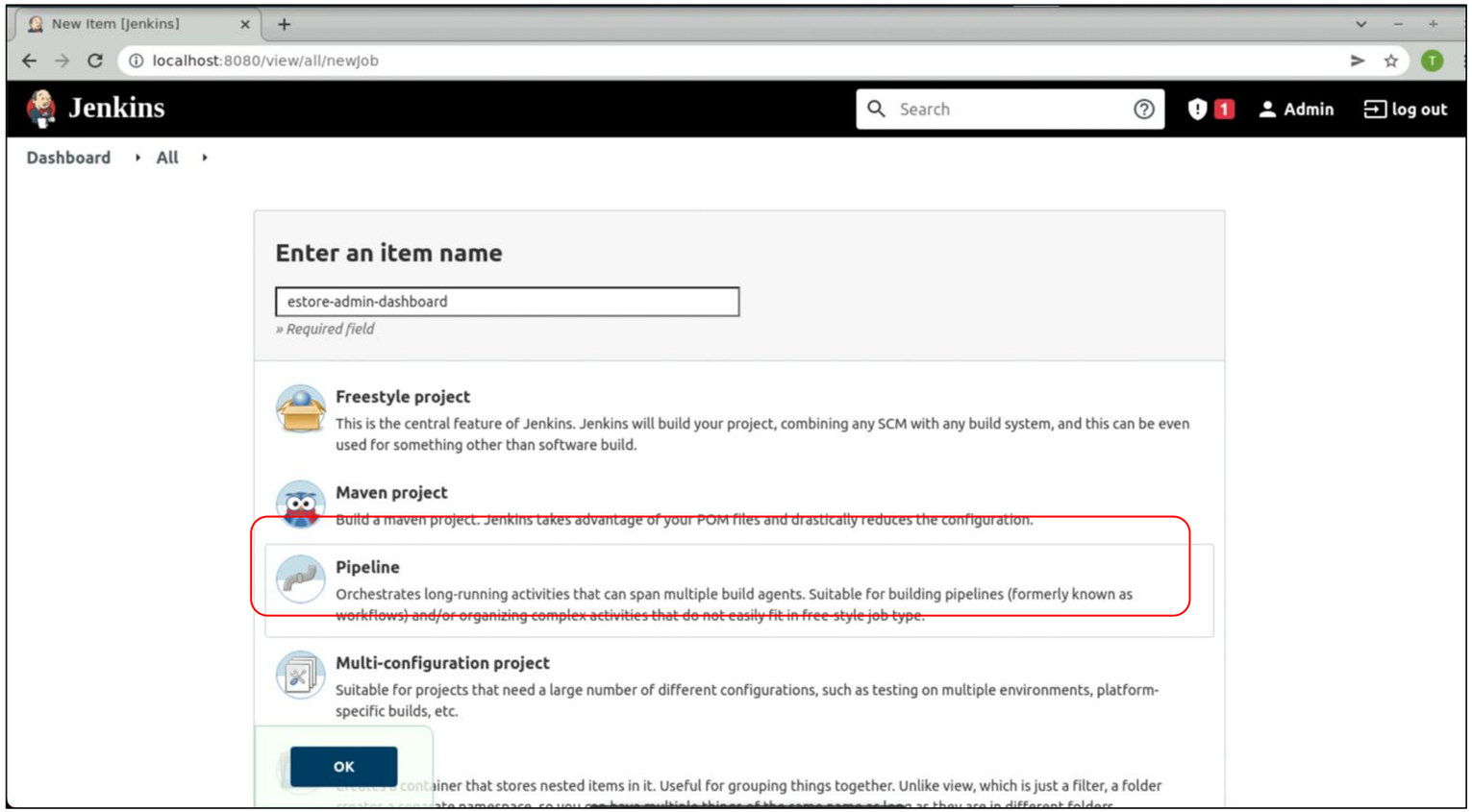
Create a Jenkinsfile for Jenkins to build the **project** as a Pipeline in Jenkins.



Create Jenkinsfile in the root directory of the project and push it to GitHub so that Jenkins can use it for Pipeline.

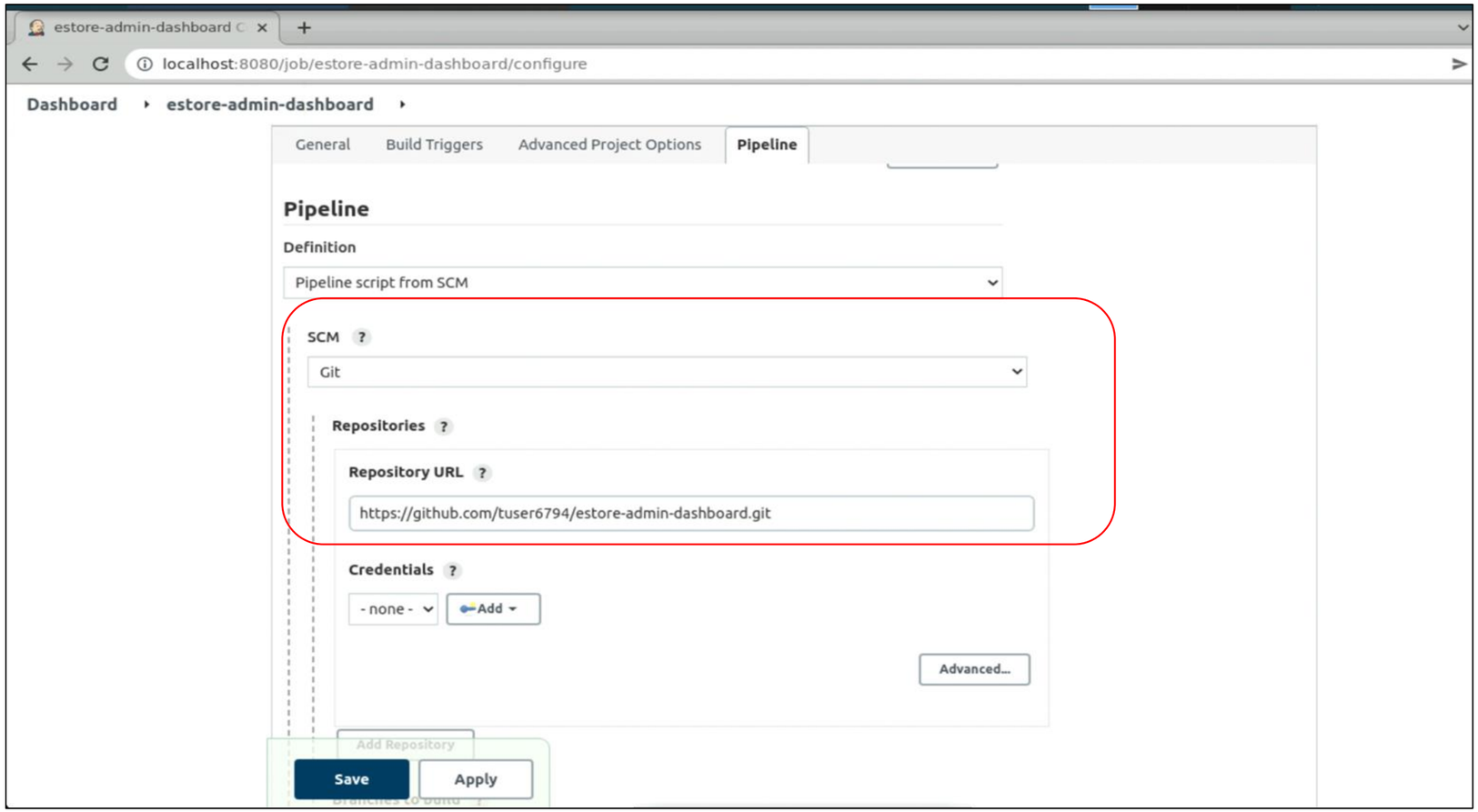
Create a Jenkins Pipeline Project for the Web-Admin Project

Create a new project in Jenkins of type Pipeline.



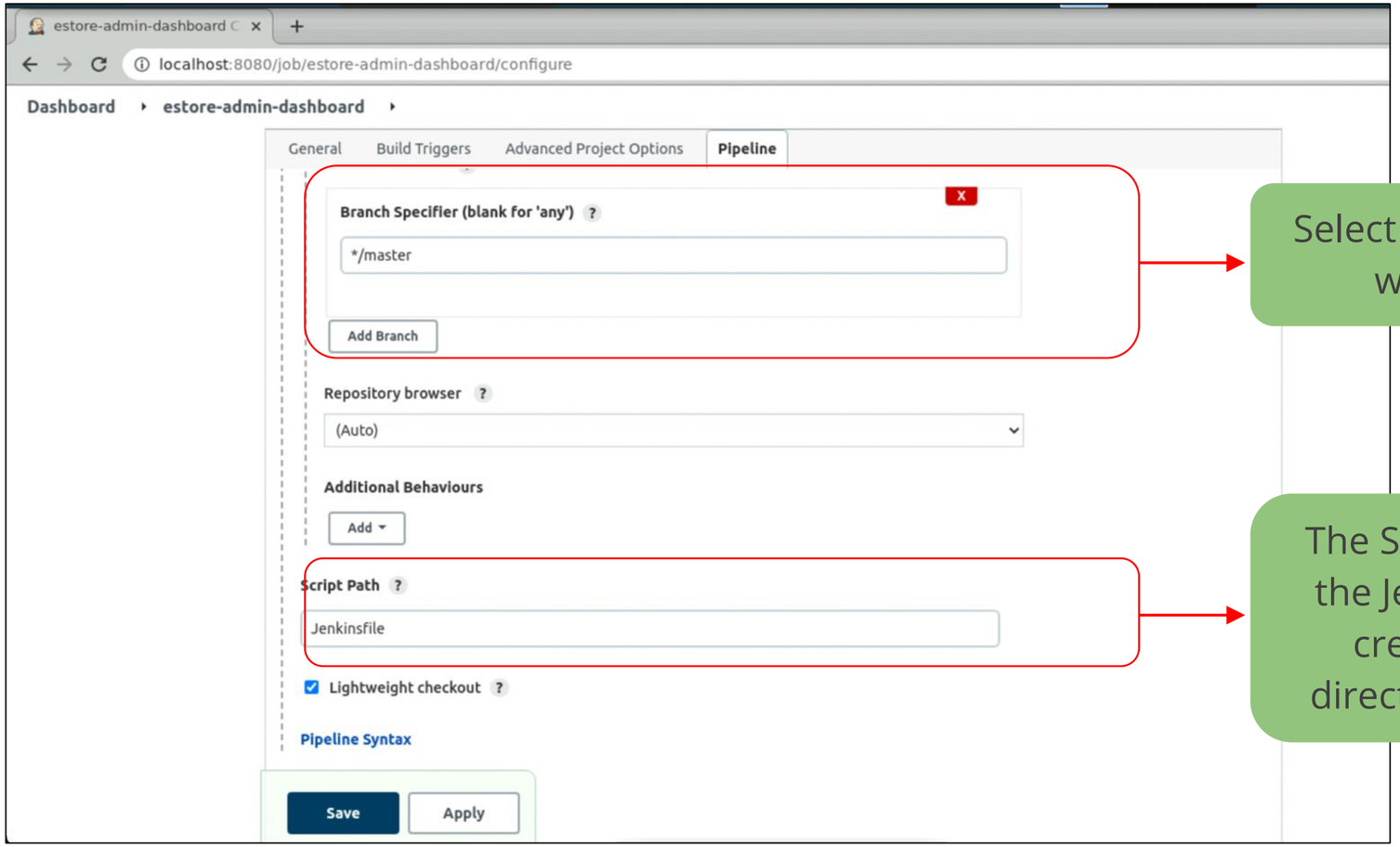
Configure Jenkins Pipeline SCM for the Web-Admin Project

Configure the Jenkins Pipeline project by passing the GitHub repository URL.



Configure Jenkinsfile in Jenkins for the Web-Admin Project

From the branch specifier, **select** the branch.

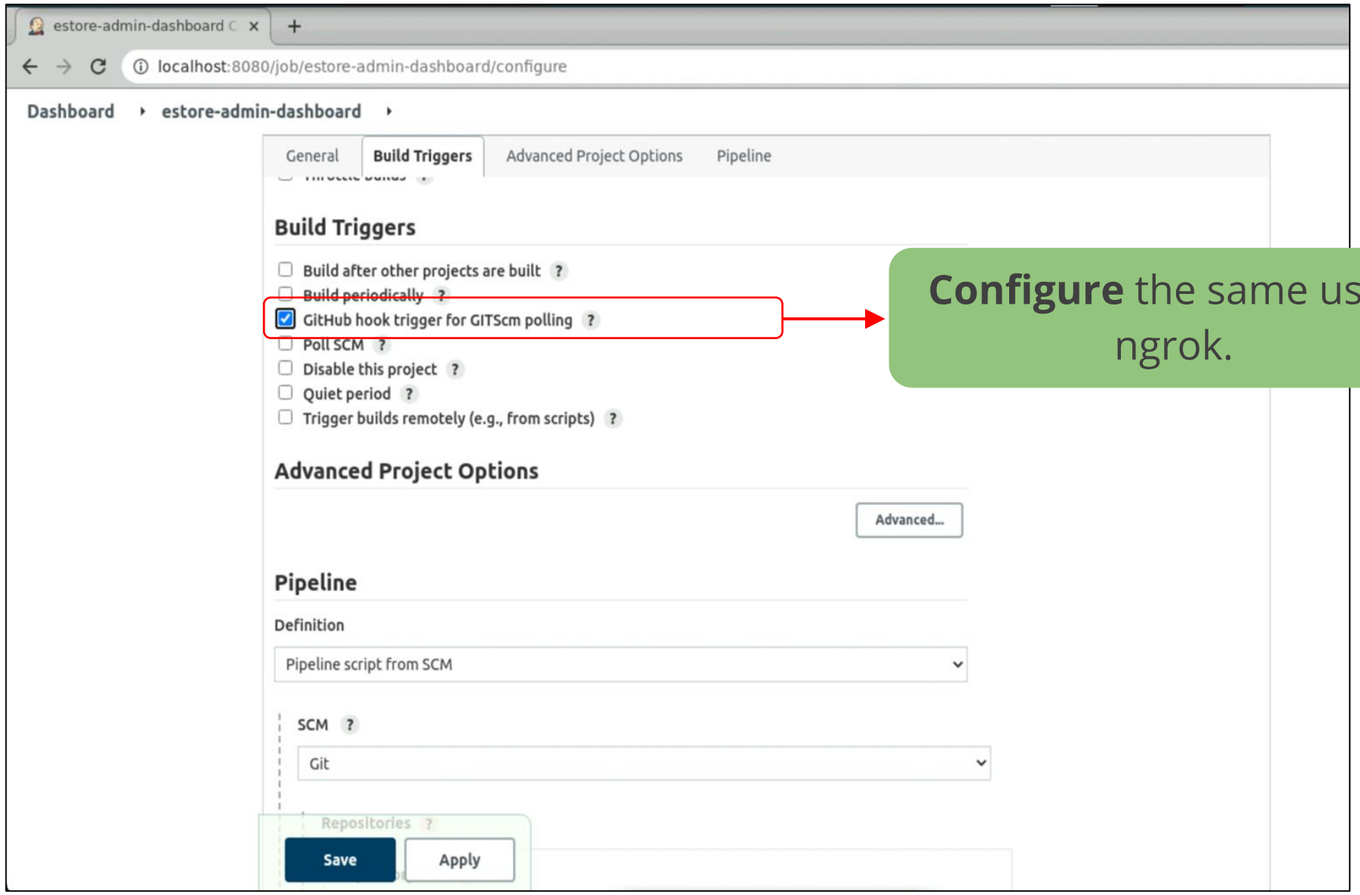


Select the default branch, which is master.

The Script Path contains the Jenkinsfile, which is created in the root directory of the project.

Optional: GitHub Hook Trigger for the Web-Admin Project

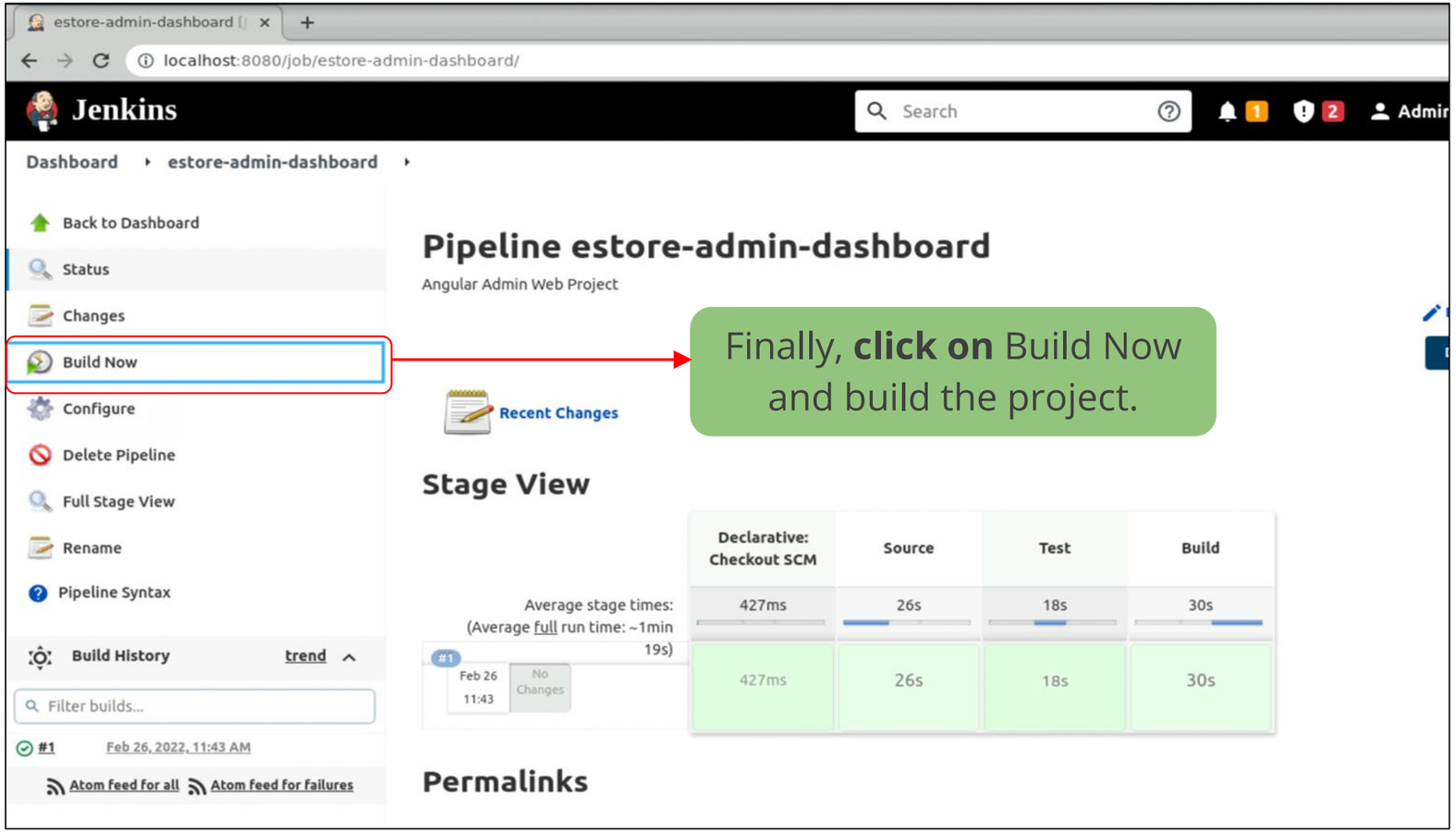
Configure **GitHub trigger** for GITScm Polling. This option will work when Jenkins is running with a proper URL instead of the localhost.



Configure the same using ngrok.

Run the Jenkins Pipeline Project for the Web-Admin Project

Notice the stages appearing as mentioned in the Jenkinsfile.
Source > Test > Build

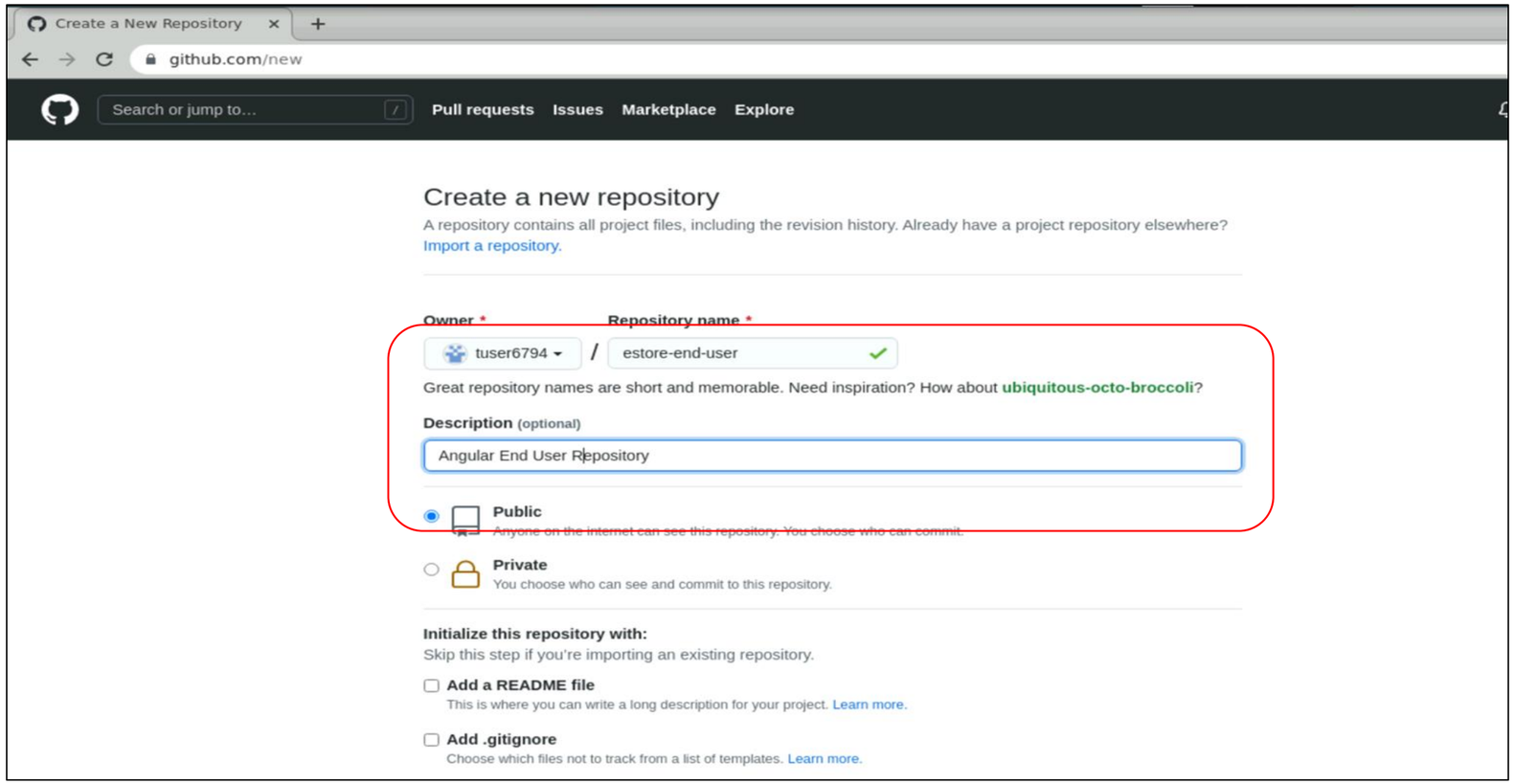


Finally, **click on** Build Now and build the project.

Set Up Jenkins for the Angular End-User Project

Create Git Repository on GitHub for the End-User Project

Create a **new repository** on your GitHub account which will be used by Jenkins to sync the code in SCM.



Sync the Project on GitHub for the End-User Project

Push the code on GitHub for the end-user project.

erishantgmail@ip-172-31-84-97: ~/Downloads/estore-end-user

File Edit View Search Terminal Help

```
erishantgmail@ip-172-31-84-97:~/Downloads/estore-end-user$ git push -u origin master
Enumerating objects: 87, done.
Counting objects: 100% (87/87), done.
Delta compression using up to 4 threads
Compressing objects: 100% (83/83), done.
Writing objects: 100% (87/87), 212.60 KiB | 7.59 MiB/s, done.
Total 87 (delta 15), reused 0 (delta 0)
remote: Resolving deltas: 100% (15/15), done.
To https://github.com/tuser6794/estore-end-user.git
 * [new branch]      master -> master
Branch 'master' set up to track remote branch 'master' from 'origin'.
erishantgmail@ip-172-31-84-97:~/Downloads/estore-end-user$
```

Configure Jenkins Pipeline Stages for the End-User Project

Create a Jenkinsfile for Jenkins to build project as pipeline in Jenkins.

Jenkinsfile - estore-end-user - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER

ESTORE-END-USER

- .angular
- .vscode
- dist
- node_modules
- src
- .browserslistrc
- .editorconfig
- .gitignore
- angular.json
- Jenkinsfile U
- karma.conf.js
- package-lock.json
- package.json
- README.md
- tsconfig.app.json
- tsconfig.json
- tsconfig.spec.json

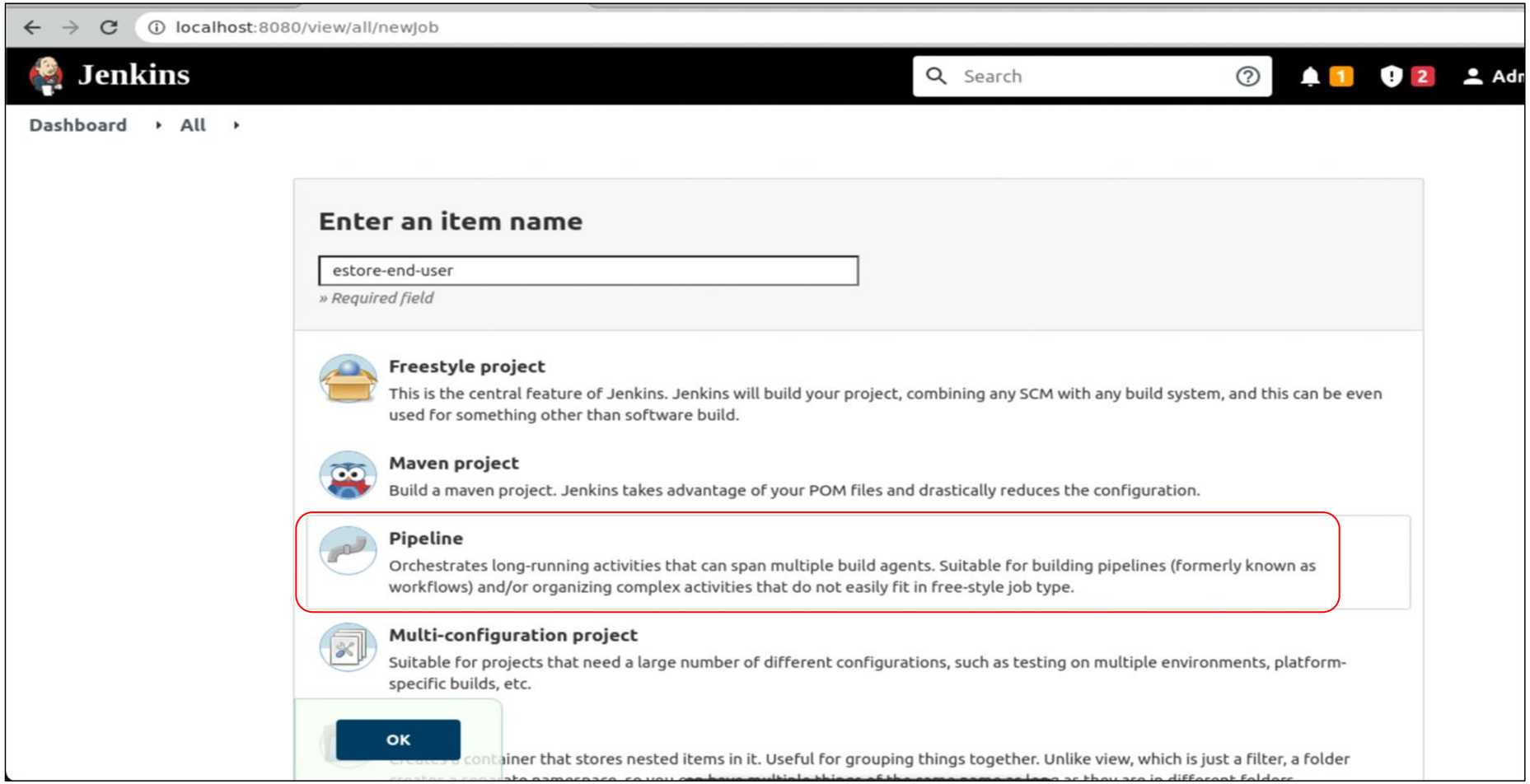
Jenkinsfile U

```
pipeline {
    agent any

    stages {
        stage('Source') {
            steps {
                // Get  code from a GitHub repository
                git 'https://github.com/tuser6794/estore-end-user.git'

                // Run npm install
                sh "npm install"

                echo 'Source Stage Finished'
            }
        }

        stage('Test') {
            steps {
                // Run ng test command
                sh "ng test --browsers ChromeHeadless --watch=false"
                echo 'Test Stage Finished'
            }
        }

        stage('Build') {
            steps {
                // Run ng build command
                sh "ng build"
                echo 'Test Stage Finished'
            }
        }
    }
}
```

Create Jenkinsfile in root directory of the project and push it to GitHub so that Jenkins project can use it for Pipeline.

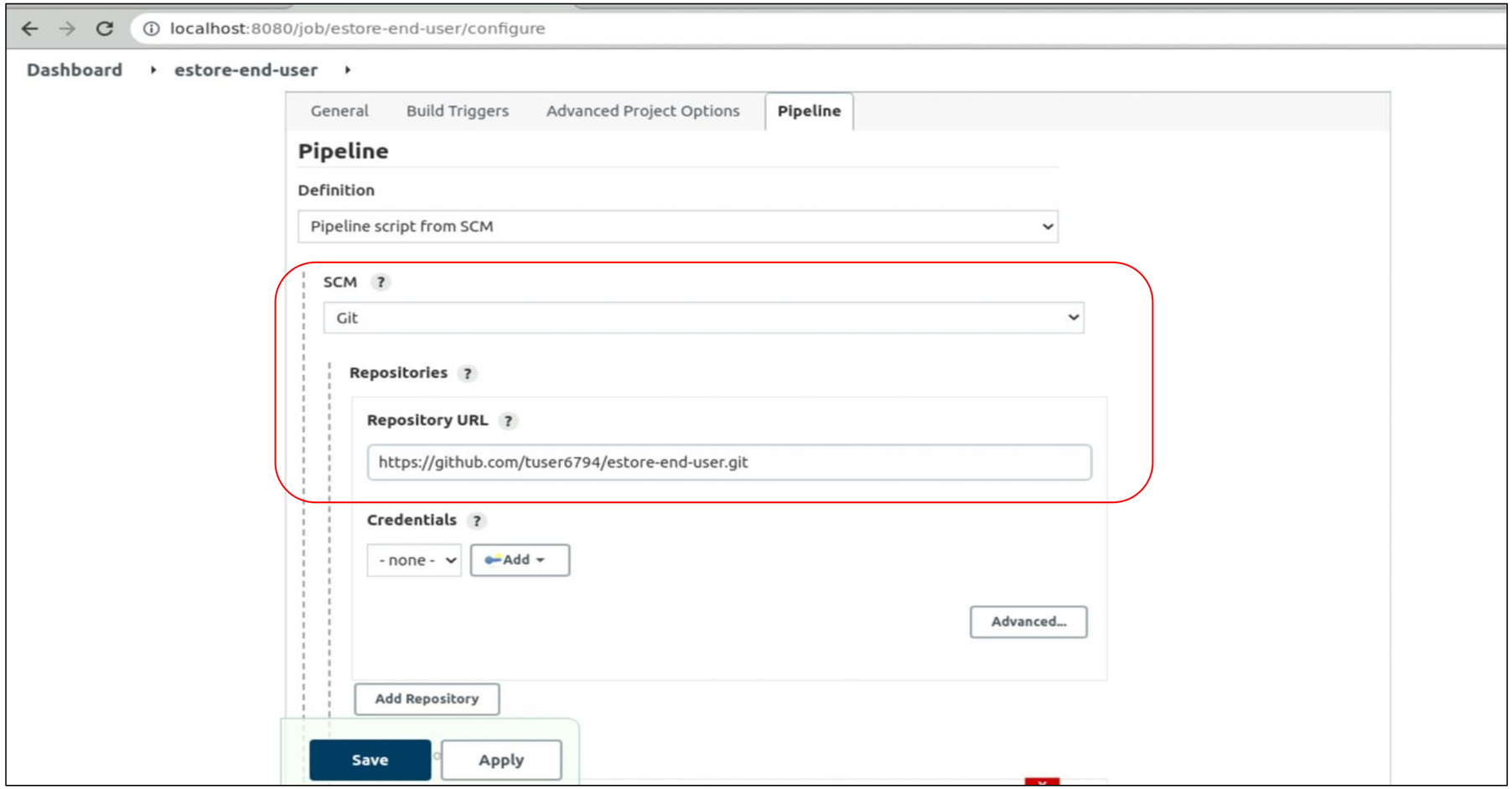
Create a Jenkins Pipeline Project for the End-User Project

Create a new project in Jenkins of type Pipeline.



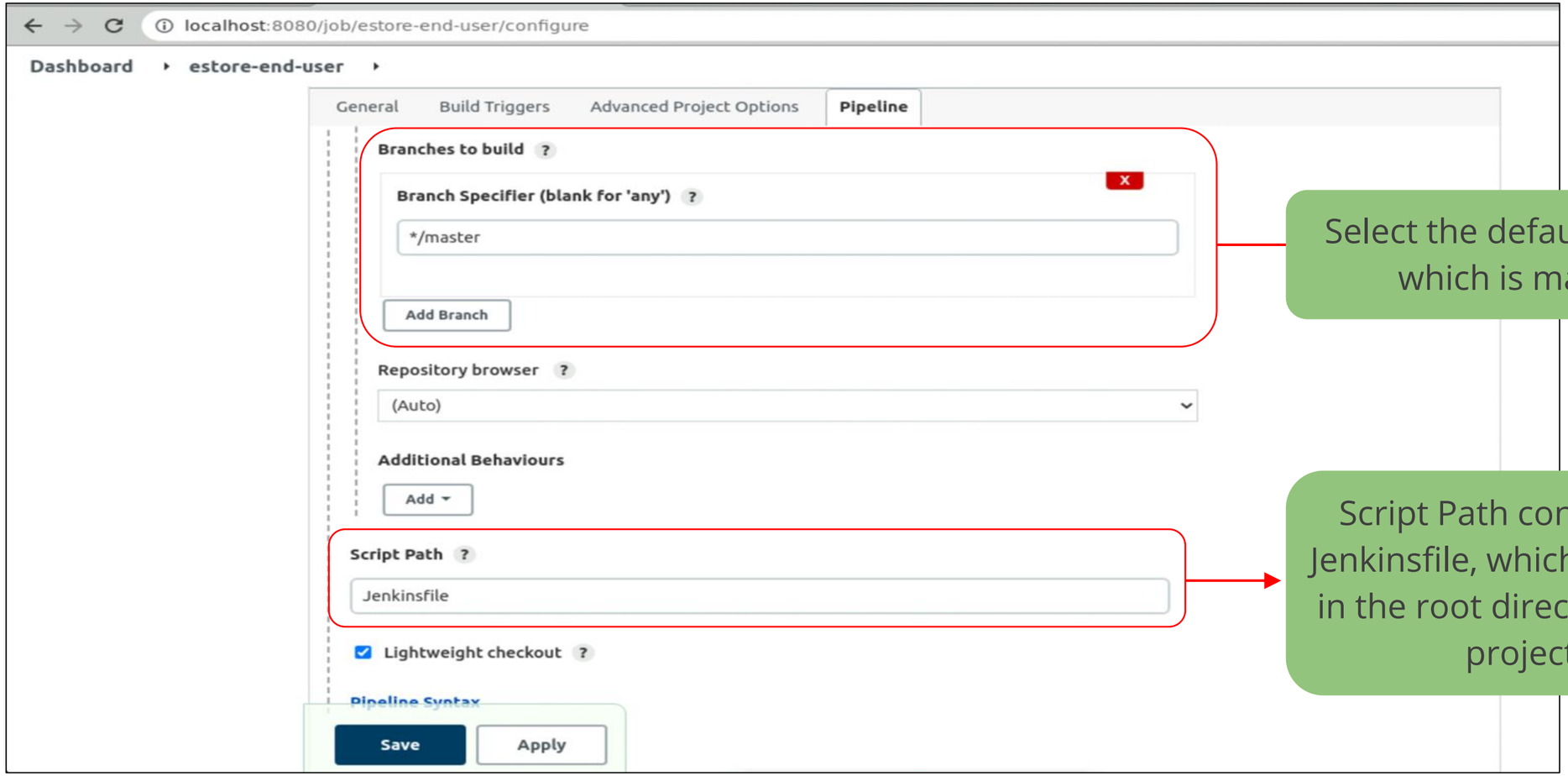
Configure Jenkins Pipeline SCM for the End-User Project

Configure the Jenkins Pipeline Project by passing the GitHub Repository URL.



Configure Jenkinsfile in Jenkins for the End-User Project

From the branch specifier, select the branch.

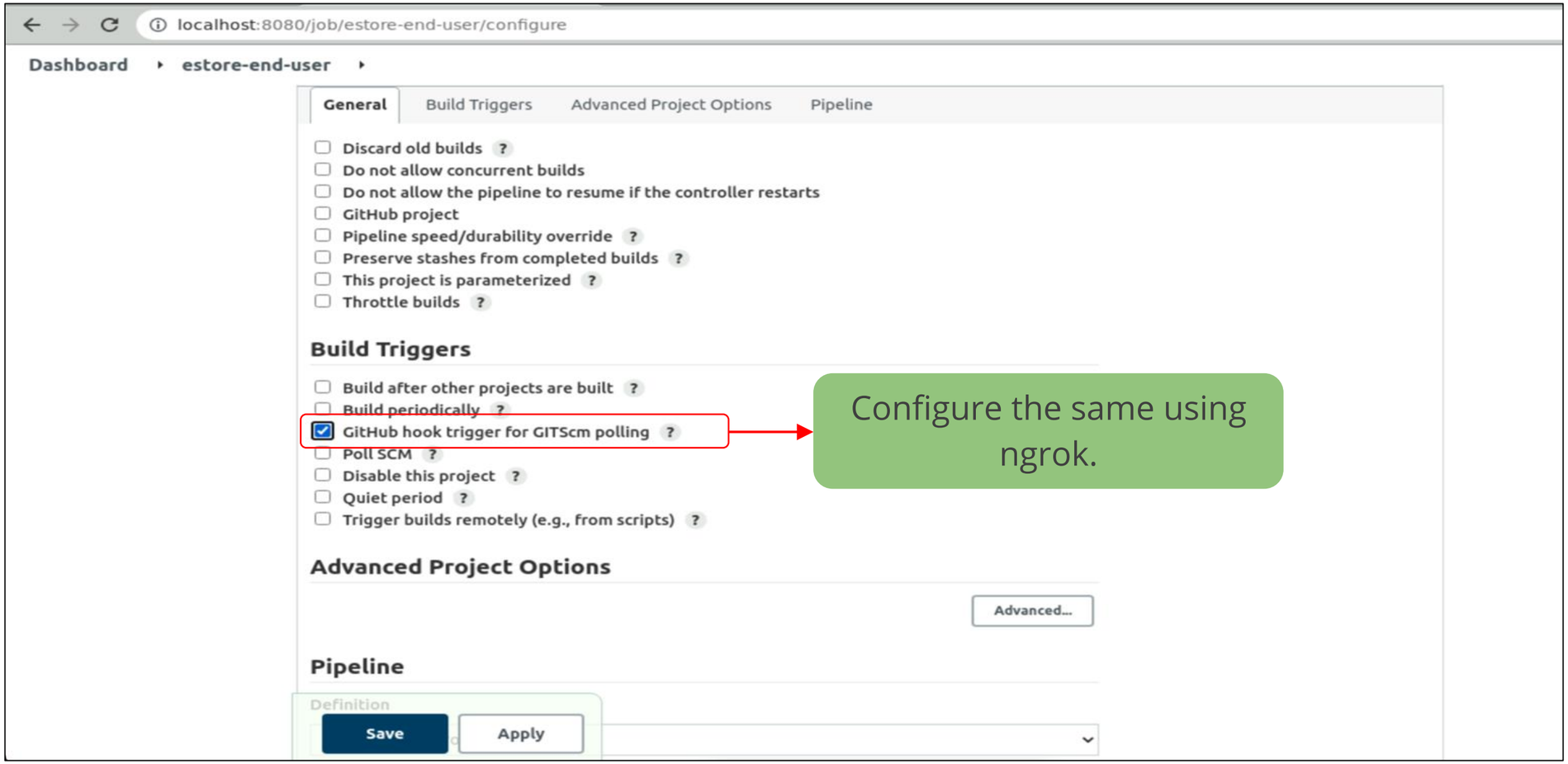


Select the default branch, which is master.

Script Path contains the Jenkinsfile, which is created in the root directory of the project.

Optional: GitHub Hook Trigger for the End-User Project

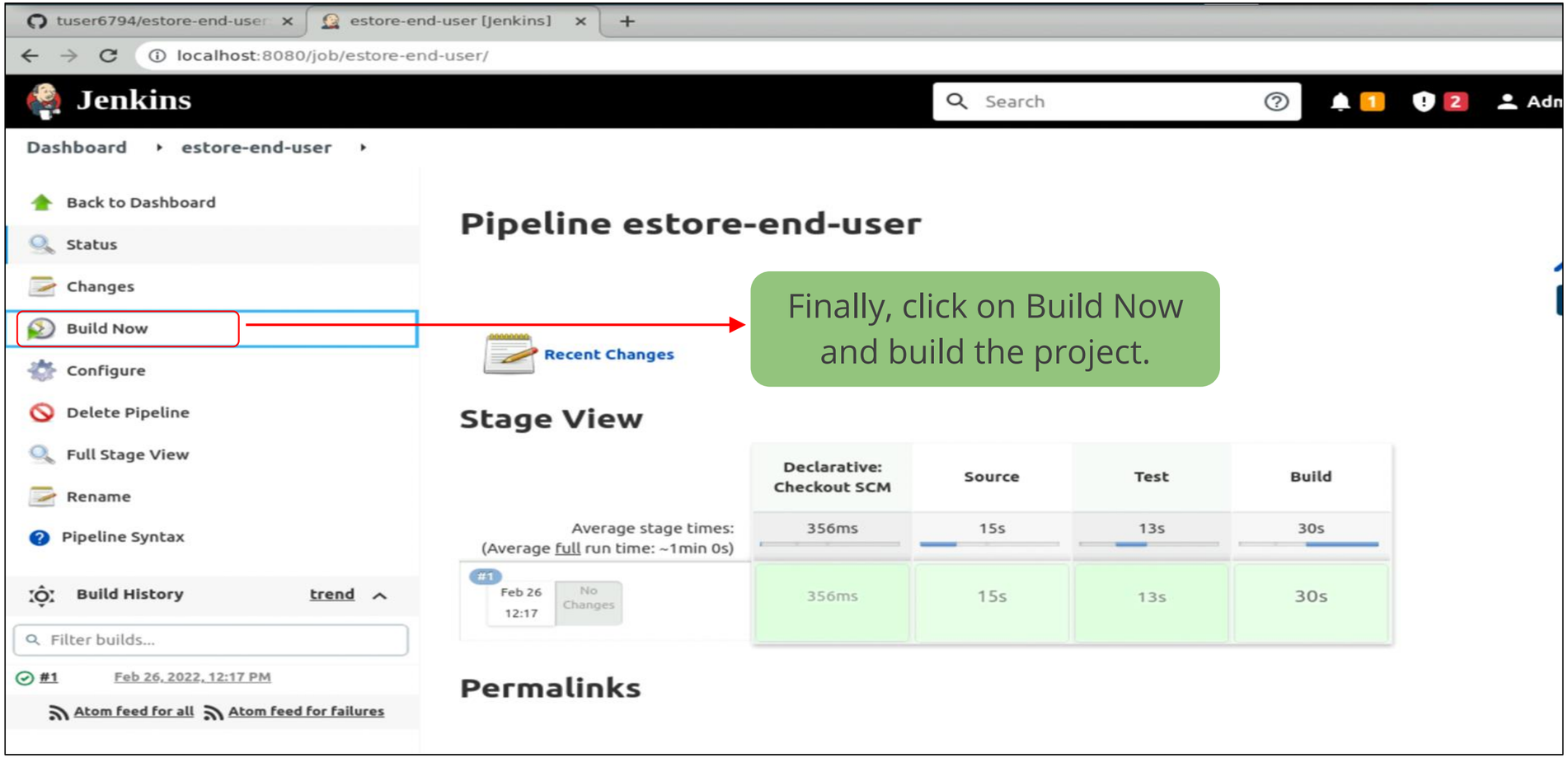
Configure GitHub trigger for GITScm Polling. This option will work when Jenkins is running with a proper URL instead of the localhost.



Configure the same using ngrok.

Run the Jenkins Pipeline Project for the End-User Project

Notice the stages appearing as mentioned in the Jenkinsfile.
Source > Test > Build



Finally, click on Build Now and build the project.

Containerize the Angular Web-Admin Project

Configure Docker Using Dockerfile for the Web-Admin Project

The Dockerfile is a multi-stage docker build with two stages:

Dockerfile - estore-admin-dashboard - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER

ESTORE-ADMIN-DASHBOARD

- .angular
- dist
- node_modules
- src
- .browserslistrc
- .editorconfig
- .gitignore
- angular.json
- Dockerfile U
- Jenkinsfile
- karma.conf.js
- package-lock.json
- package.json
- README.md
- tsconfig.app.json
- tsconfig.json
- tsconfig.spec.json

Dockerfile U

Dockerfile > ...

```
# Use official node image as the base image
FROM node:latest as build

# Set the working directory as app
WORKDIR /usr/local/app

# Add the source code to app
COPY ./ /usr/local/app/

# Install the dependencies
RUN npm install

# Generate the build of the application
RUN npm run build

# Use official nginx image as the base image
FROM nginx:latest

# Copy the build output for nginx contents
COPY --from=build /usr/local/app/dist/admin-dashboard /usr/share/nginx/html

# Expose port 80
EXPOSE 80
```

Building the angular project into dist directory for production.

Serving the application using NGINX web server on port 80.

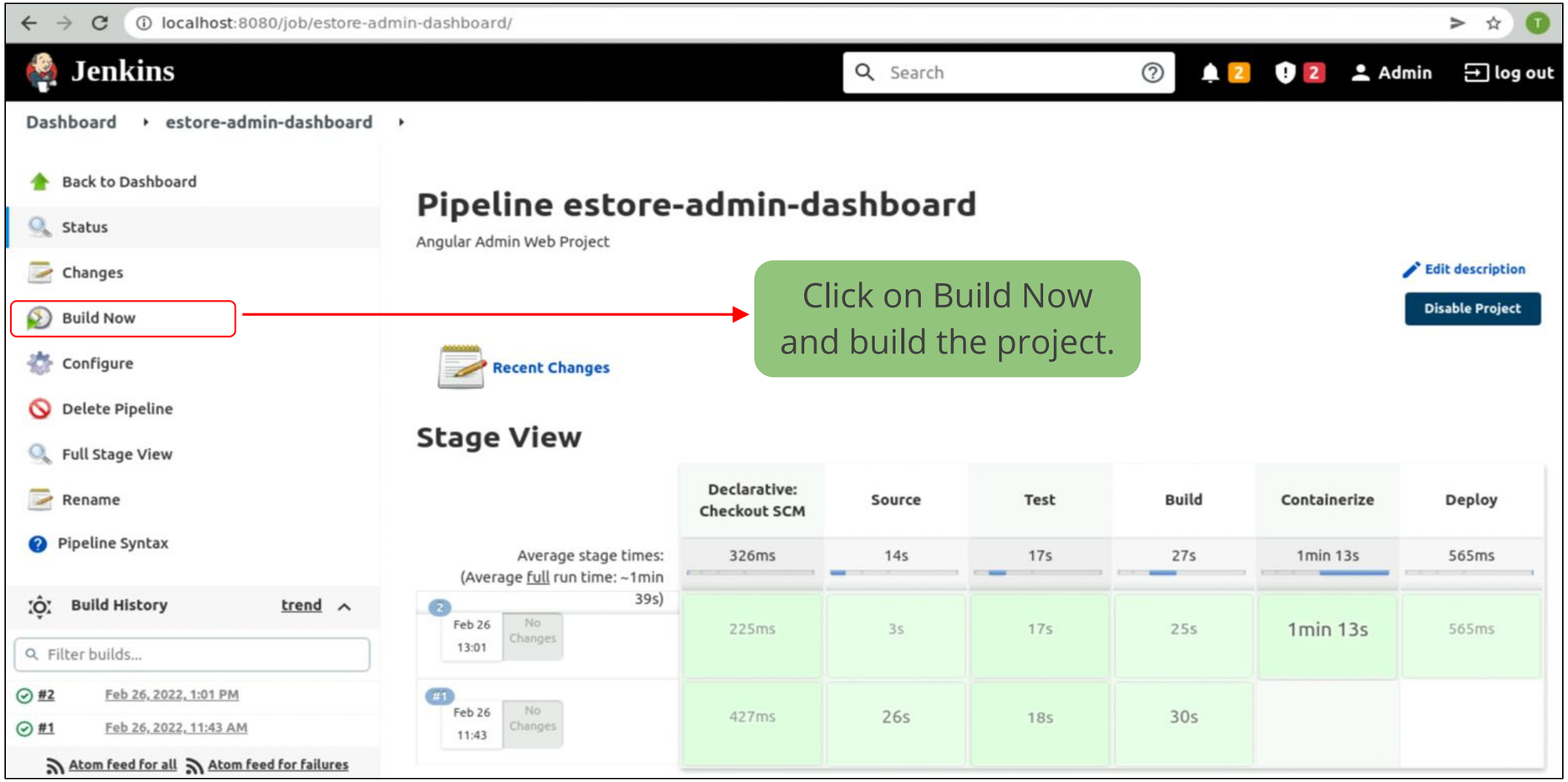
Configure Jenkins Pipeline Stages for Docker

Add the stages for containerizing the Angular web admin that is building and running container.

Create it in the root directory of project and push it to GitHub so that Jenkins Project can use Jenkinsfile and Dockerfile.

Run the Jenkins Pipeline Project for the Web-Admin Project

Notice stages appearing as mentioned in the Jenkinsfile.
Source > Test > Build > Containerize > Deploy



Click on Build Now and build the project.

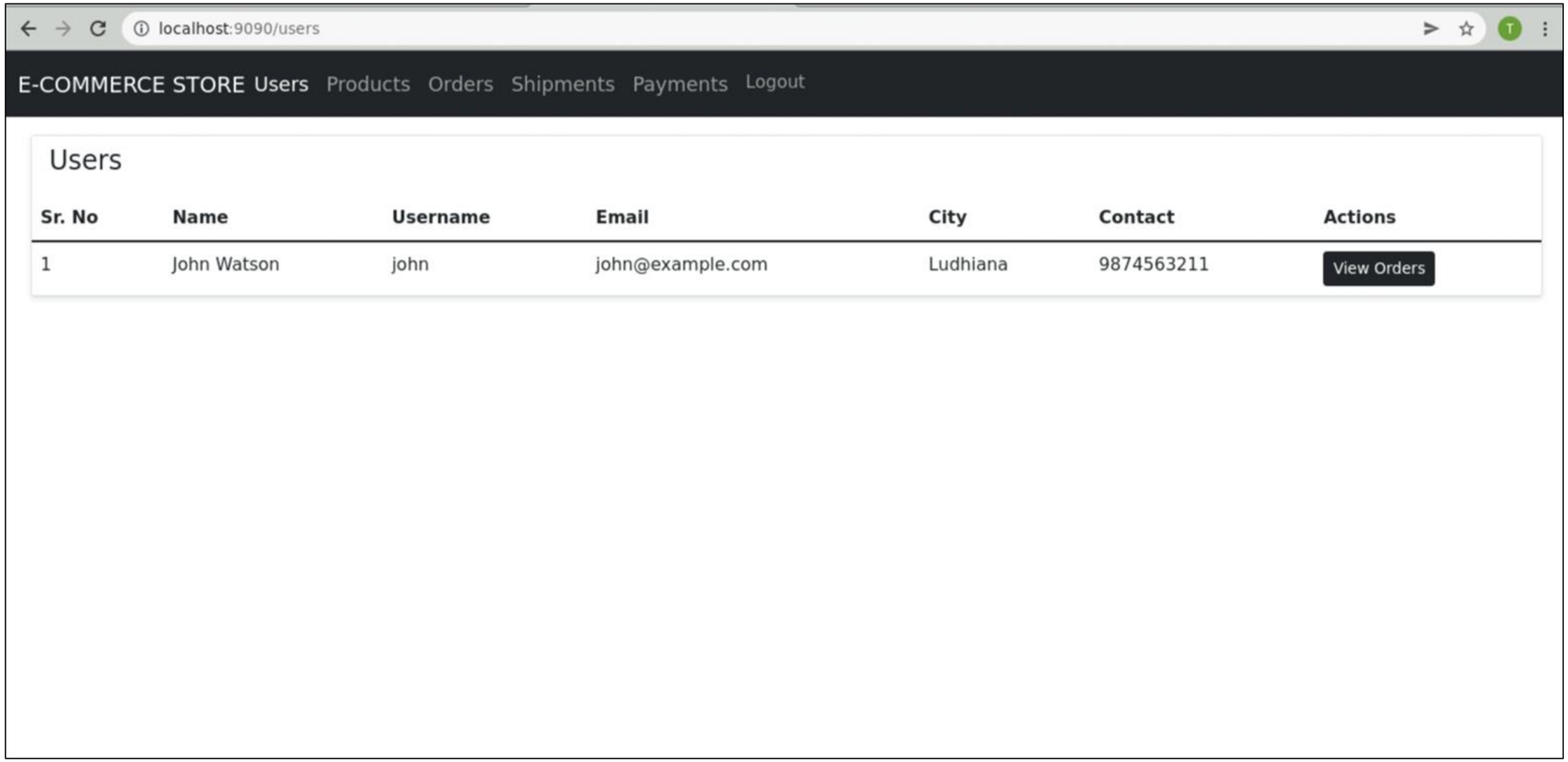
Check Docker ps -a

Check the status of running container by **docker ps –a**, which is a command to validate Jenkins build.

```
erishantgmail@ip-172-31-84-97: ~
File  Edit  View  Search  Terminal  Help
erishantgmail@ip-172-31-84-97:~$ docker ps -a
CONTAINER ID   IMAGE          COMMAND                  CREATED          STATUS          PO
RTS                                   NAMES
0834bf25aea2   estore-admin   "/docker-entrypoint.…"   10 minutes ago   Up 10 minutes   0.
0.0.0:9090->80/tcp, :::9090->80/tcp   jovial_heisenberg
erishantgmail@ip-172-31-84-97:~$
```

Check the Web-Admin Project on Localhost

The web-admin project is up and running in a docker container.



TECHNOLOGY

Computing

Caltech | Center for Technology & Management Education

Dockerise the Pipeline

Powerd by simplilearn

Containerize Angular End-User Project

Configure Docker Using Dockerfile for the End-User Project

The Dockerfile is a multi-stage docker build with two stages:

Dockerfile - estore-end-user - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER

ESTORE-END-USER

- .angular
- .vscode
- dist/estore-user-app
- node_modules
- src
- .browserslistrc
- .editorconfig
- .gitignore
- angular.json
- Dockerfile U
- Jenkinsfile M
- karma.conf.js
- package-lock.json
- package.json
- README.md
- tsconfig.app.json
- tsconfig.json
- tsconfig.spec.json

Dockerfile U

Dockerfile > ...

```
# Use official node image as the base image
FROM node:latest as build

# Set the working directory as app
WORKDIR /usr/local/app

# Add the source code to app
COPY ./ /usr/local/app/

# Install the dependencies
RUN npm install

# Generate the build of the application
RUN npm run build

# Use official nginx image as the base image
FROM nginx:latest

# Copy the build output for nginx contents
COPY --from=build /usr/local/app/dist/estore-user-app /usr/share/nginx/html

# Expose port 80
EXPOSE 80
```

Building the angular project into dist directory for production.

Serving the application using NGINX web server on port 80.

Configure Jenkins Pipeline Stages for Docker

Add the stages for containerizing the Angular web admin that isnbuilding and running container.

Jenkinsfile - estore-end-user - Visual Studio Code

File Edit Selection View Go Run Terminal Help

EXPLORER

ESTORE-END-USER

- .angular
- .vscode
- dist/estore-user-app
- node_modules
- src
- .browserslistrc
- .editorconfig
- .gitignore
- angular.json
- Dockerfile U
- Jenkinsfile M
- karma.conf.js
- package-lock.json
- package.json
- README.md
- tsconfig.app.json
- tsconfig.json
- tsconfig.spec.json

Jenkinsfile M

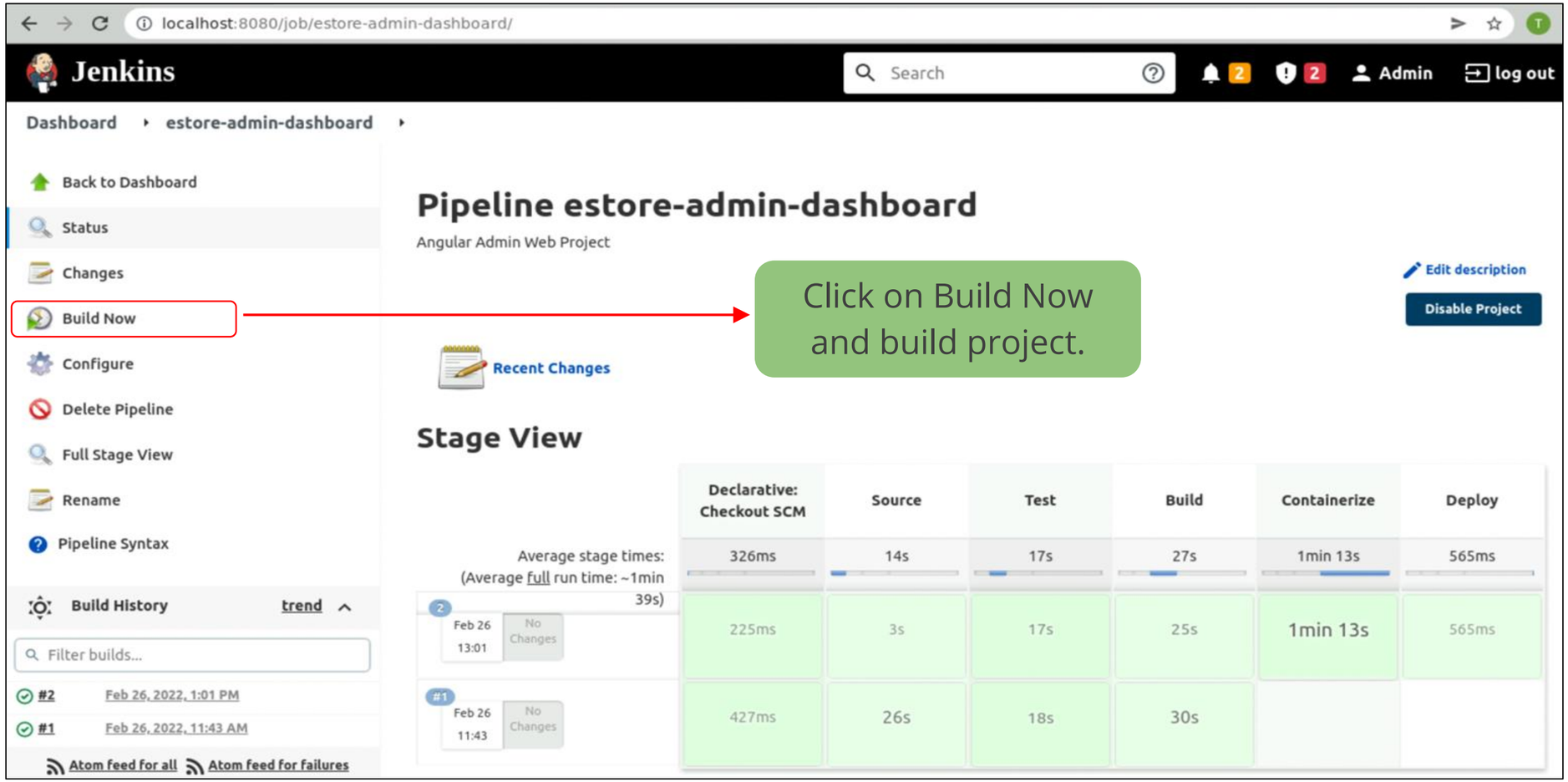
Jenkinsfile

```
                echo 'Test Stage Finished'
            }
        }

        stage('Build') {
            steps {
                // Run ng build command
                sh "ng build"
                echo 'Test Stage Finished'
            }
        }

        stage('Containerize') {
            steps {

                // Run docker command to build a container
                sh "docker build -t estore-end-user ."

                echo 'Containerizing the App with Docker'
            }
        }

        stage('Deploy') {
            steps {

                // Run the image in port 9090 and local port 8080 is blocked
                sh "docker run -d -p 9090:80 estore-end-user"

                echo 'Deploy the App with Docker'
            }
        }
    }
}
```

Create it in root directory of the project and push it to GitHub so that Jenkins project can use Jenkinsfile and Dockerfile.

Run the Jenkins Pipeline Project for the End-User Project

Notice stages appearing as mentioned in the Jenkinsfile.
Source > Test > Build > Containerize > Deploy



Click on Build Now and build project.

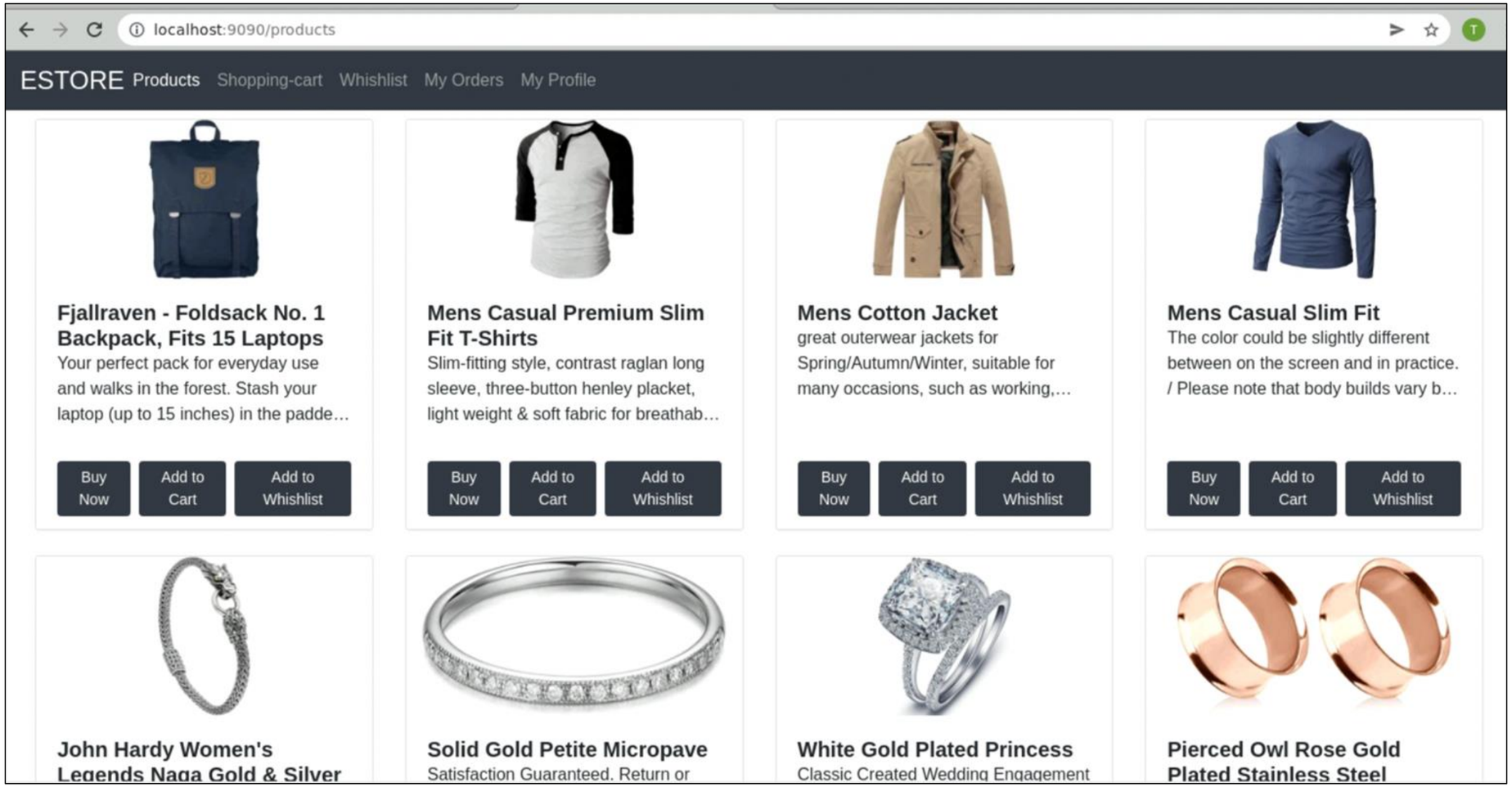
Check Docker ps -a

Check status of running container by **docker ps –a**, which is a command to validate Jenkins build.

```
erishantgmail@ip-172-31-84-97: ~
File  Edit  View  Search  Terminal  Help
erishantgmail@ip-172-31-84-97:~$ docker ps -a
CONTAINER ID   IMAGE             COMMAND                  CREATED         STATUS         P
ORTS                                   NAMES
d8f75bc5c8cd   estore-end-user   "/docker-entrypoint.…"   2 minutes ago   Up 2 minutes   0
.0.0.0:9090->80/tcp, :::9090->80/tcp   wonderful_colden
erishantgmail@ip-172-31-84-97:~$
```

Check the End-User Web Project on Localhost

The End-user web project is up and running in a docker container.



Key Takeaways

- Jenkins is an automation build server.
- Jenkins Pipeline for Angular apps is set using Jenkinsfile.
- Dockerfile is used to dockerize the Angular apps.
- Jenkins Pipeline is built to dockerize the Angular apps.

Before the Next Class

You have successfully completed this session.

Before the next discussion, you should go through:

- AWS

What’s Next?

Now we have finished our classes and design pattern for the back-end project with respect to end-user module. In our next live session, we will:

- See how to create Jenkins Pipeline for Java back end
- See how to work with Jenkinsfile for the Maven project
- Learn various commands in Dockerfile for the Java back-end project
- Explore CI/CD Pipeline with Jenkins