**Total days 9**

**PG FSD: Integration and Deployment Aug cohort**

Devops tools

Docker

Docker compose

CI and CD tool using Jenkin jobs and Pipe line jobs

Overview AWS : S3, RDS and EC2 instance

9 – Mar – 2024 Day 1

Multi OS machine.

VWare software. : it is tool which help to run more than one OS. We can run multi os like window xp, linux, mac etc.

Oracle Virtual Box

If we want to run Guest OS with help of VMWare software we need to share resource for VM

Base machine 16 GM RAM 1TB hard disk

Guest OS 4GB 100 Gib Hard disk

If we want to run 10 OS 1GB

VM is use to create abstraction version of Physical machine or OS.

Docker is an advanced OS virtualization an open source software platform which help create application, build application with required dependencies, test the application and deploy the application very easily.

Docker is use to create Containerization application.

Virtualization Vs Containerization

Virtualization is use to create the abstract version of an operation system.

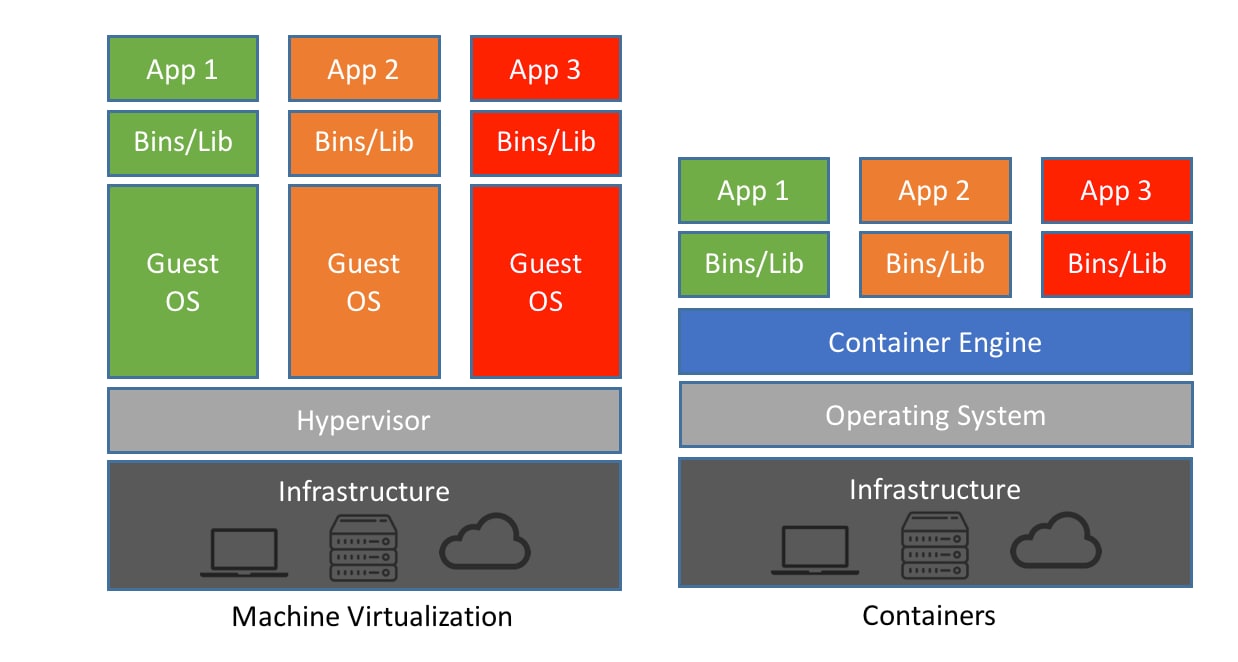
Containerization is use to create abstract version of an application. To run the application software develop in any language we need system software. This system software provide by docker engine.

Container : run the environment.

JRE

Web container : responsible to run servlet and jsp

Spring container



If we develop any application like using java, python, angular, database mysql or oracle or mongo db.

To run that application in your machine you need to install all required software as well as their dependencies.

Docker platform which help to pack all application with their dependencies in form of image.

Docker image is a read only template created using Docker file. Which contains application details as well as required dependencies which help to run the application.

Docker-file : set of instruction or set of rules which help to create the image.

Once we run the image actual application become up with the help of container

Docker container java Demo

Docker Image (read only template format) -🡪 source code

Docker file --🡪 Demo.java

Docker hub : Docker hub an open source public repository (by default one private). Which help to publish or push as well as pull the Docker image. It is like a git hub. In git hub we can push as well as pull any type of data.

In Window 10 we can install Docker. Docker internally use kernel.

1. If you want to install docker in local machine.
2. Please use Virtual Lab.

Docker software

1. Docker Desktop is GUI base docker software which we can do all operation using command prompt as well gui base.
2. Please use Virtual machine

Open the terminal and command prompt

docker version this command is use to find the version of docker

or

sudo docker version

docker info this command is use to find the information about docker.

docker images this command is use to find the images.

docker pull imageName this command is use to pull the image

hello-world

docker pull hello-world

docker images

docker run imageName/imageId

hello-world image responsible to run the C program.

debian or busybox os images.

docker pull busybox

docker run -it busybox it means iterative mode

docker run -it debian pull and run the image

docker run -it ubuntu bash

if we want to create custom image we need take the help of pre defined OS images like busybox, ubuntu etc. create the docker file and inside this file we need to write the instruction to run the application with required dependencies.

If we want to create the image we need to create Dockerfile without extension.

Dockerfile (default name Dockerfile consider)

FROM busybox

CMD ["echo","Welcome Simple Docker example created by Akash"]

In command prompt

docker build -t imageName . -f Dockerfile

Example

docker build -t my-busybox . -f Dockerfile

docker build -t my-busybox . file name must be Dockerfile

docker images

docker run my-busybox

we will another image to run java program

create another separate folder java\_image

Docker provided openjdk images which internally use OS image and this image provide java run time environment.

**Demo.java**

import java.util.Scanner;

public class Demo {

    public static void main(String[] args) {

        System.out.println("Simple Basic Operation");

        Scanner sc = new Scanner(System.in);

        System.out.println("Enter the vaue of a");

        int a = sc.nextInt();

        System.out.println("Enter the vaue of b");

        int b = sc.nextInt();

        int sum  = a+b;

        System.out.println("Sum of two number is "+sum);

        sc.close();

    }

}

Dockerfile

FROM openjdk:11

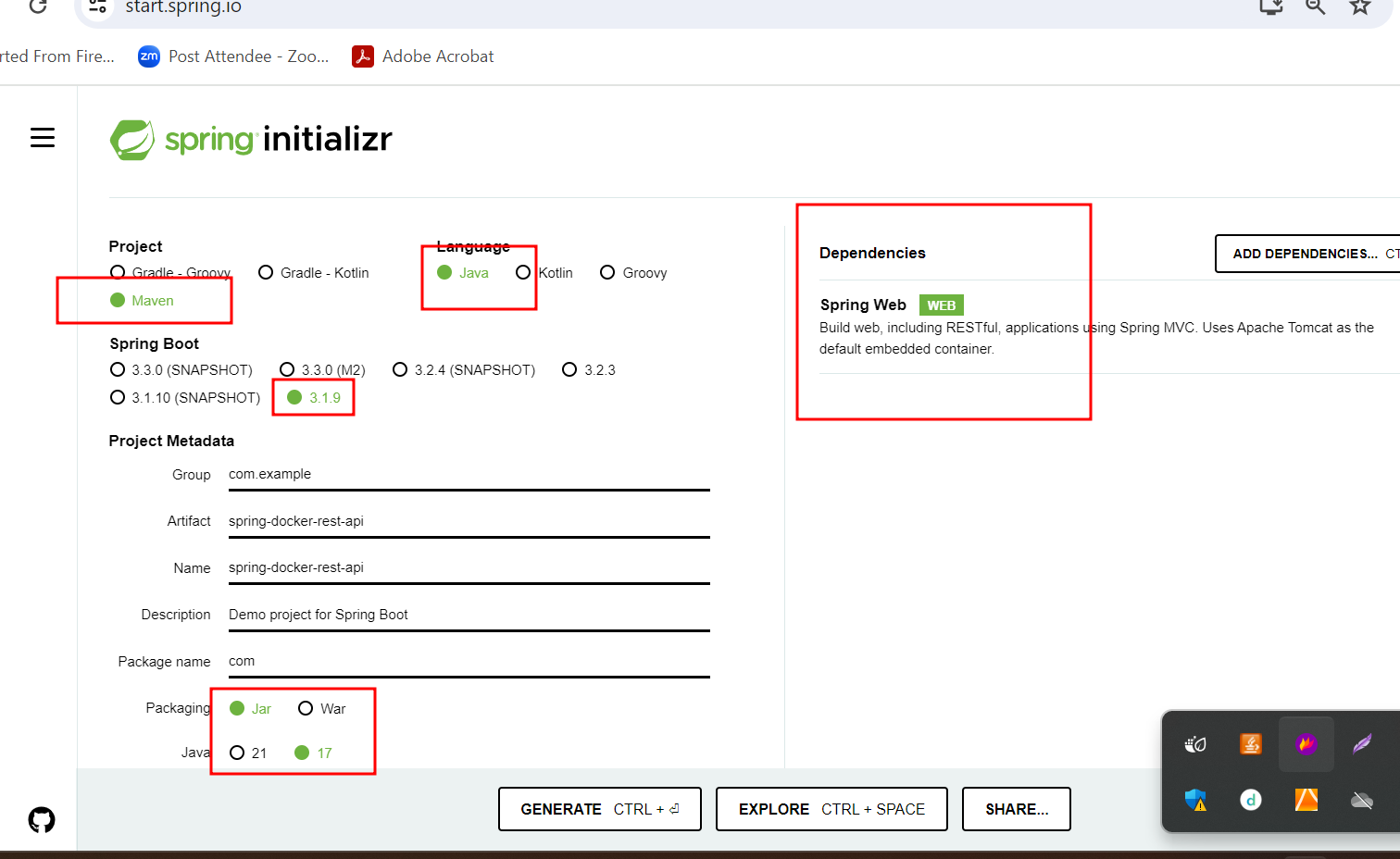
COPY Demo.java .

RUN javac Demo.java

CMD ["java","Demo"]

docker build -t my-java . -f Dockerfile

creating image to run the spring boot application



Create simple rest api and using eclipse IDE. After create application create the jar file using Eclipse IDE with help of run with install or build command.

It will create jar file inside target folder.

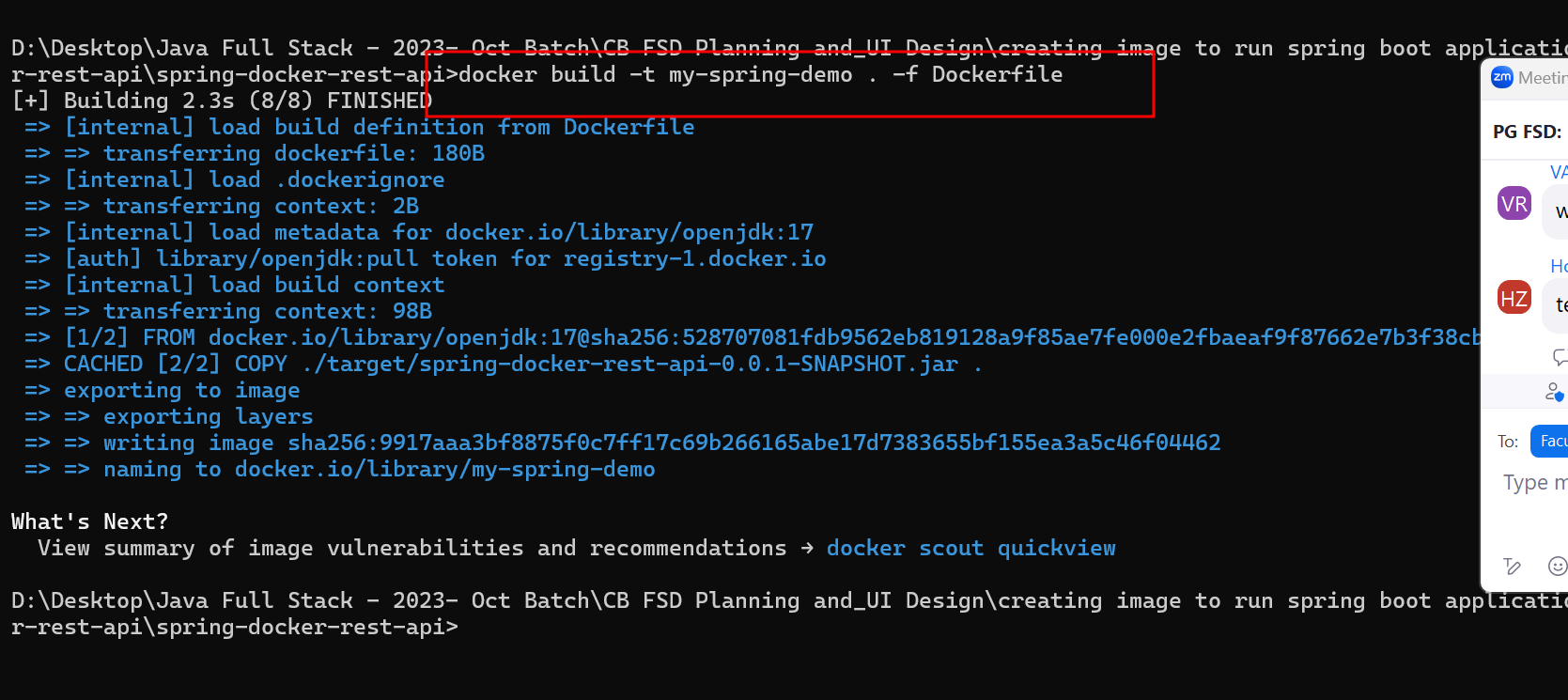
Dockerfile

FROM openjdk:17

COPY ./target/spring-docker-rest-api-0.0.1-SNAPSHOT.jar .

CMD ["java","-jar","spring-docker-rest-api-0.0.1-SNAPSHOT.jar"]

docker build -t my-spring-boot . -f Dockerfile



10 – Mar – 2024 Day 2

If image is responsible to run web application we need to run using below command.

docker run -d -p 8080:8080 imageName/Imageid

docker run -d -p 8081:8080 imageName/Imageid

docker run -d -p 8082:8080 imageName/Imageid

-d detached mode or background

-p public port number

Red colour port number 8080 actual application port number.

Blue colour port number 8080 publish port number. It can be same or different. But make sure in your machine that port number must be free.

After run web application run successfully you can see one container get started.

If we wan to see running container then we can run the command as

docker ps

or

docker container ls



then open the browser

<http://localhost:9090>

creating image to run the angular application

Develop simple angular project

Create angular project using ng command

ng new angular-docker-app

routing 🡪 no

styling -🡪 css

please open app.component.html page write simple static message.

<div>

<p>Welcome to angular with docker project created by akash</p>

</div>

ng serve -o to run the program in development mode

we need to build the project

ng build build the project

after build successfully it will create dist folder inside a project.

Inside dist folder it contains project folder name and that folder contains build files.

After build the project these build file we can give to backend developer like spring boot developer and we can deploy independently.

We need server

Like tomcat, web logic, IIS server.

NgInx

Engine -x : open source web server which help to deploy the application.

Docker hub provide us nginx server or tomcat etc.

Dockerfile

FROM nginx

COPY ./dist/angular-docker-app/ /usr/share/nginx/html

Then create the image

docker build -t my-angular-test-app . -f Dockerfile

Angular by default provide small web server and that server we can use in development mode which use internally 4200 port number.

ng serve -o (we run angular application using inbuild web server with port number 4200)

we created image of angular with build file with help of nginx server.

nginx server internally use 80 port number.

After created angular image we need to run this application using below command

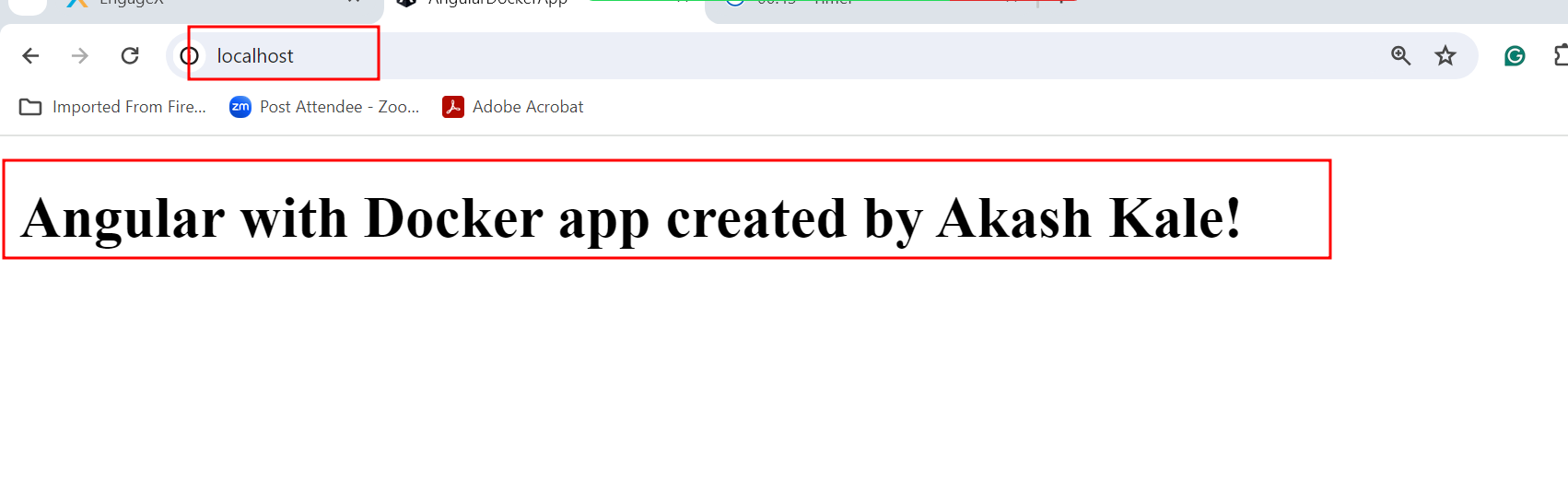
docker run -d -p 80:80 my-angular-test-app

(because nginx server default port number is 80)

docker ps check running container.

<http://localhost:80>

80 is default port number of http protocol.



If want to pause the container

docker stop containerId/containerName

docker start containerId/containerName : to start the container

**docker rm containerId/containerName**  : to remove container

if container running state we can’t remove we need to stop and then remove or we can use -f option

**docker rm containerId/containerName -f remove forcefully**

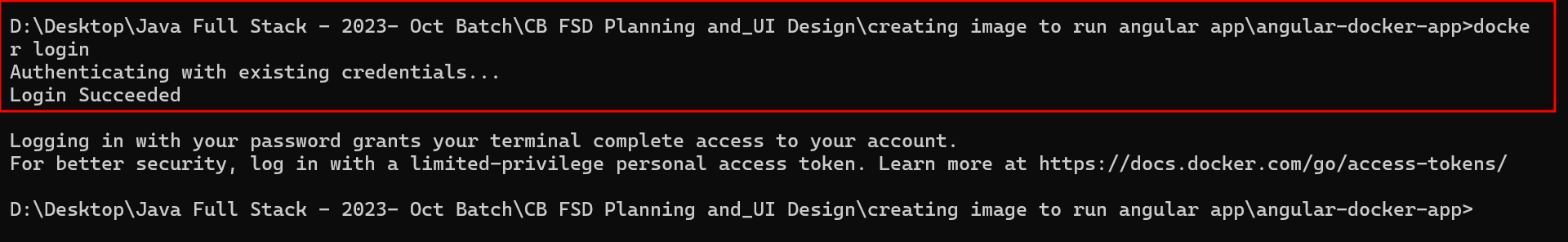
**docker rmi imageName/imageId**

**if image link with any container we can’t remove then we need to use option as -f**

**docker rmi imageName/imageId -f**

**Publish the docker images in Docker hub account**

**docker login**



**If ask emailid and password please provide docker hub account details.**

**Before push the image or publish the image we need to create the tag for that image. Tag is like a identity or version which help to keep the track between multiple update in our application**

**docker tag imagename dockerhubaccountid/imagename:version**

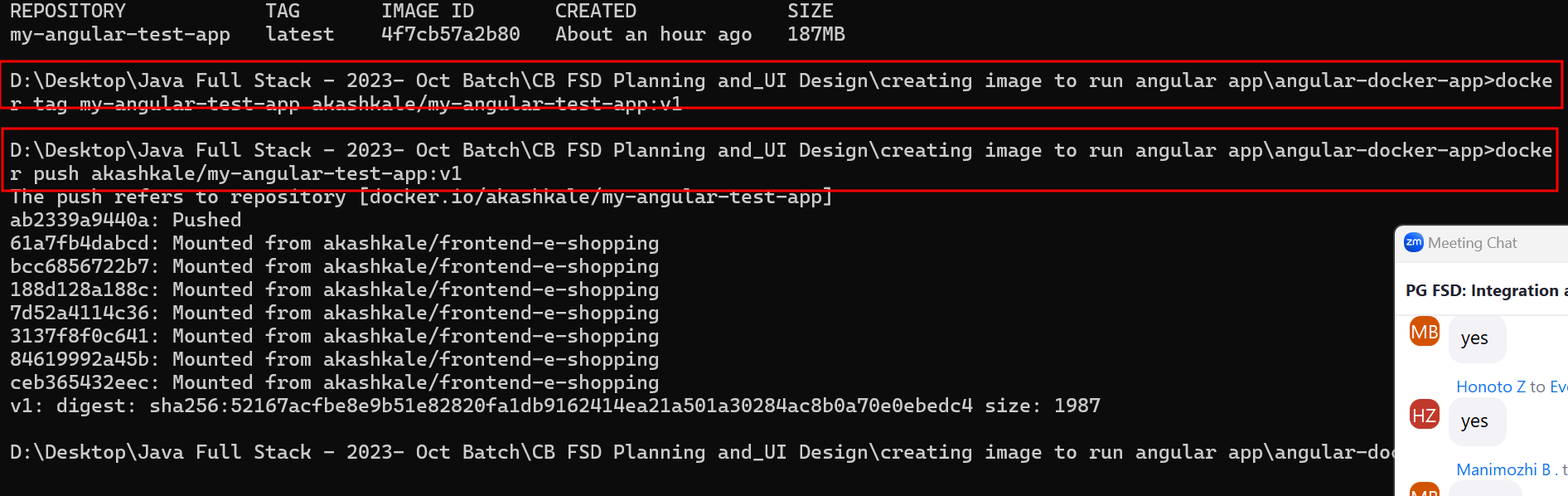
**docker tag my-angular-test-app akashkale/my-angular-test-app:v1**

**version can be number or alphabets etc.**

**after tag create now you can push or publish your image in docker hub account**

**docker push dockerhubaccountid/imagename:version**

**docker push akashkale/my-angular-test-app:v1**



**docker pull akashkale/my-angular-test-app:v1**

**docker run -d -p 91:80 akashkale/my-angular-test-app:v1**

we created image to run core java, spring boot, angular application and every image is responsible to run the application using container. All these images internally use OS image to run the application.

Day 3 : 16 Mar 2024

If we use data base connection in spring boot application the data running on local machine. Spring boot container doesn’t connect local machine database.

Mysql database image.

Docker hub provide mysql database.

docker run --name some-mysql -e MYSQL\_ROOT\_PASSWORD=my-secret-pw -d mysql:tag

below command is use to pull the image and run the image with custom name for container.

docker run -d -p 3307:3306 --name=my-sql-container -e MYSQL\_ROOT\_PASSWORD=root mysql:latest

-d detached mode

-p publish port number

-e environment

Once you run mysql image. Using container id or container name open the container OS.

**docker exec -it my-sql-container bash**

or

**docker exec -it containerId bash**

once you connect then please write below command to connect mysql. In my SQL container OS.

mysql -u root -p

password : root

after connected databases you can do database work

create database

use database

create table

insert records

We develop Spring boot application with MySQL database. Run this application using docker concept.

We can run spring boot image.

Spring boot image internally use OS to run the container

We need to create the jar, using docker file with help of jar file we can create the image. Using image we can run the container.

OS Os : spring boot application running

We need to create the network to communicate both OS. To share the data

We can pull and run mysql container. My SQL Container internally use OS to run the container.

OS : mysql container running

Docker provide . Docker compose : Docker-compose is a toolkit which is responsible to run more than one container. Those container can run independently or they can communicate with each other to share the data. All container as well image instruction we write in docker-compose.yml or yaml (Yet Another Markup of language).

We need to create docker-compose.yml file. Inside this yml file we provide all user -defined image or pre -defined image details with their configuration and with help of few command we can start all container, stop all container pause container, remove , publish etc.

1. Using docker-compose we run more than one angular application.

docker-compose --version

docker-compose build : this command is use to build the image using Dockerfile.

Create one folder as Docker-compose files

Another folder with name as docker compose to run angular projecter

Open the folder in VS code

Then create docker-compose.yml

version: "3.8"

services:

  akash-service:

   image: akashkale/my-angular-test-app:v1

   ports:

     - 81:80

  manimozhi-service:

   image: manimozhi11/myangularapp:v1

   ports:

     - 82:80

  varsh-service:

    image: varshur/angulardocker:v1

    ports:

      - 83:80

  swathi-service:

    image: swathitummuri/my-angular-test-app:v1

    ports:

      - 84:80

Then open the terminal (make sure terminal open in that location file must be present)

docker-compose up -d

then docker images

docker ps

then check all application with port number as 81

like http://localhost:81 , <http://localhost:82>, <http://localhost:83>, <http://localhost:84>

if you want to stop

docker-compose down

create docker compose file to run spring boot with mysql container.

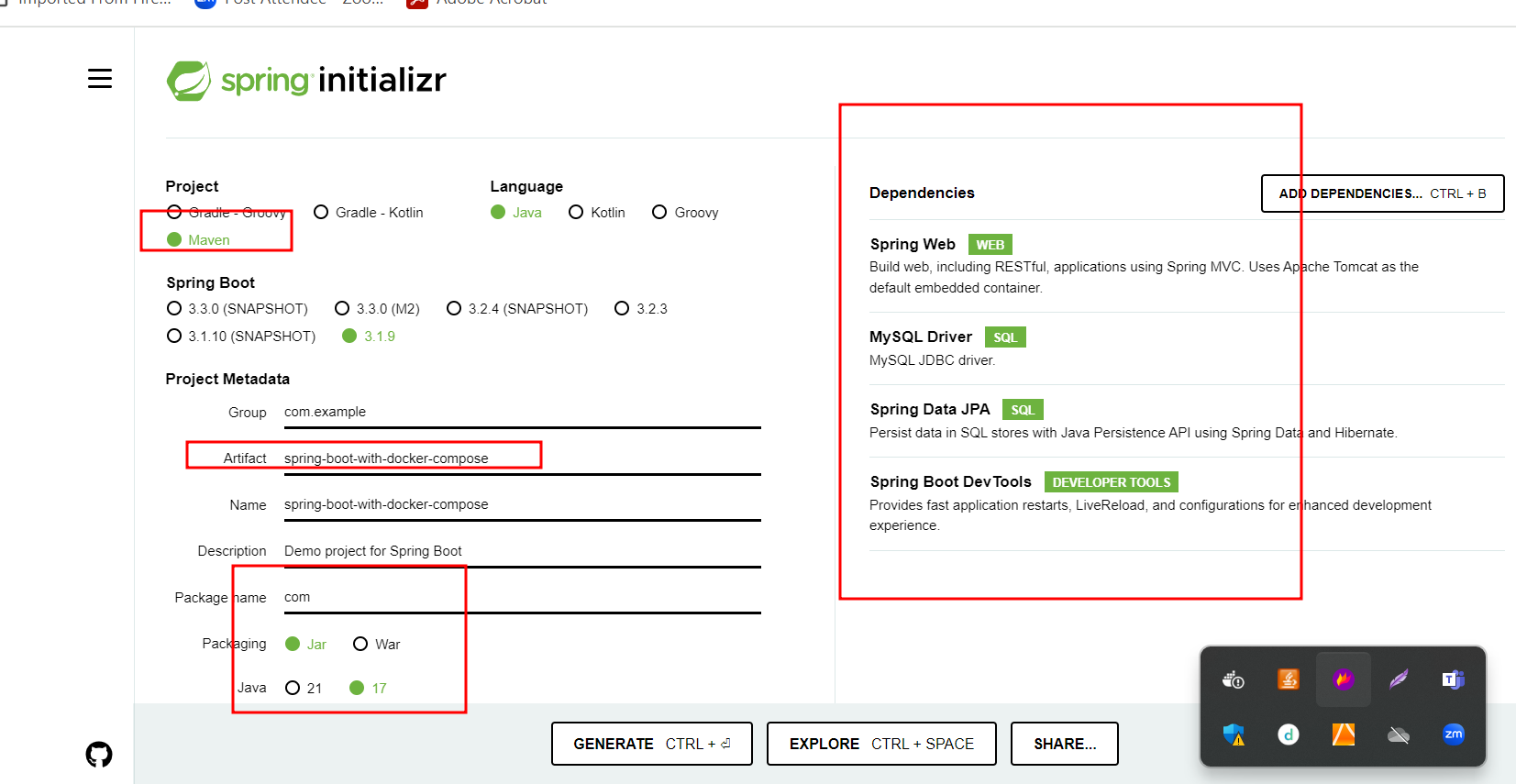
We create spring boot application using spring initializer with starter as

Web starter

Jpa starter

Mysql connector

Devtool : refresh the project whenever we do any changes



Please create more than one rest api which is responsible to interact with database using controller layer, service layer, entity layer and repository layer etc and test with local machine mysql database.

Now we need to create jar file using maven command or using eclipse run install option.

After create jar file we need to create the Docker file.

FROM openjdk:17

COPY ./target/spring-boot-with-docker-compose-0.0.1-SNAPSHOT.jar .

CMD ["java","-jar","spring-boot-with-docker-compose-0.0.1-SNAPSHOT.jar"]

Now we create docker-compose.yml file and we provide two container details ie

Spring boot container

Mysql-container

docker-compose.yml

version: '3.8'

services:

mysql-db:

image: mysql:8

environment:

MYSQL\_ROOT\_PASSWORD: root

MYSQL\_DATABASE: productdb

ports:

- 3307:3306

restart: always

springboot-container:

build: .

container\_name: springboot-container

depends\_on:

- mysql-db

ports:

- 8080:8080

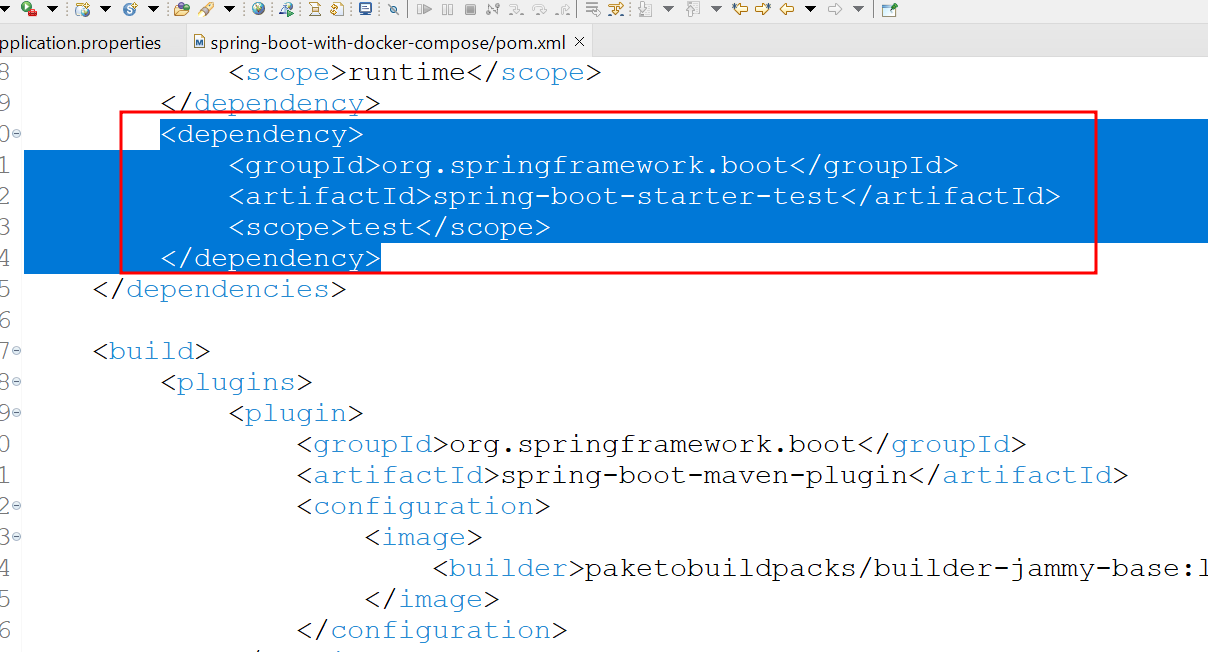
restart: always

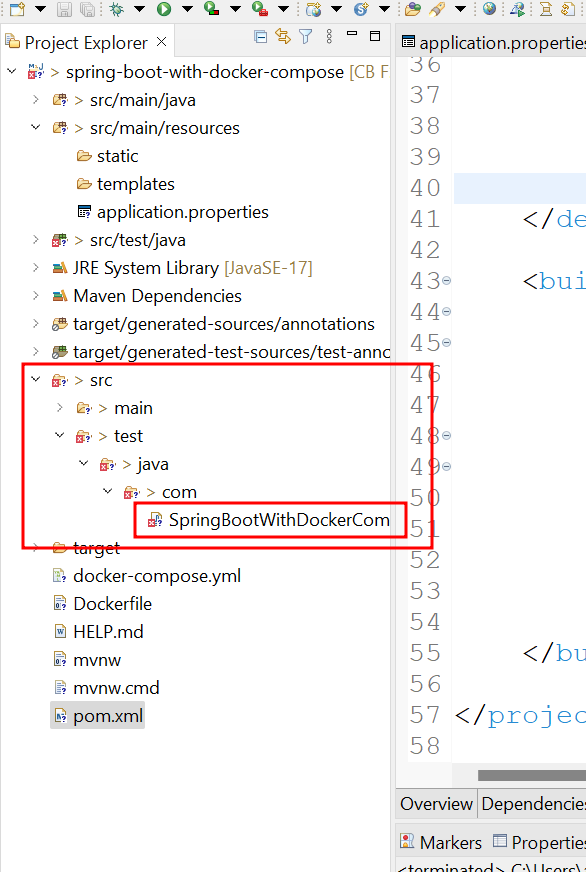
Then in application.properties file please provide your mysql container details as contaienrname, password and database name.

After changes done please re-build maven project. From pom.xml file please remove testing dependencies.

When we build the project (mvn package command execute). Before mvn package mvn test command execute that command check database connection.

From pom.xml file remove testing tags





After removed testing dependencies and testing file. Please re-build the project.

Please open terminal and using docker-compose command to create the image and run the image

docker-compose up --build -d

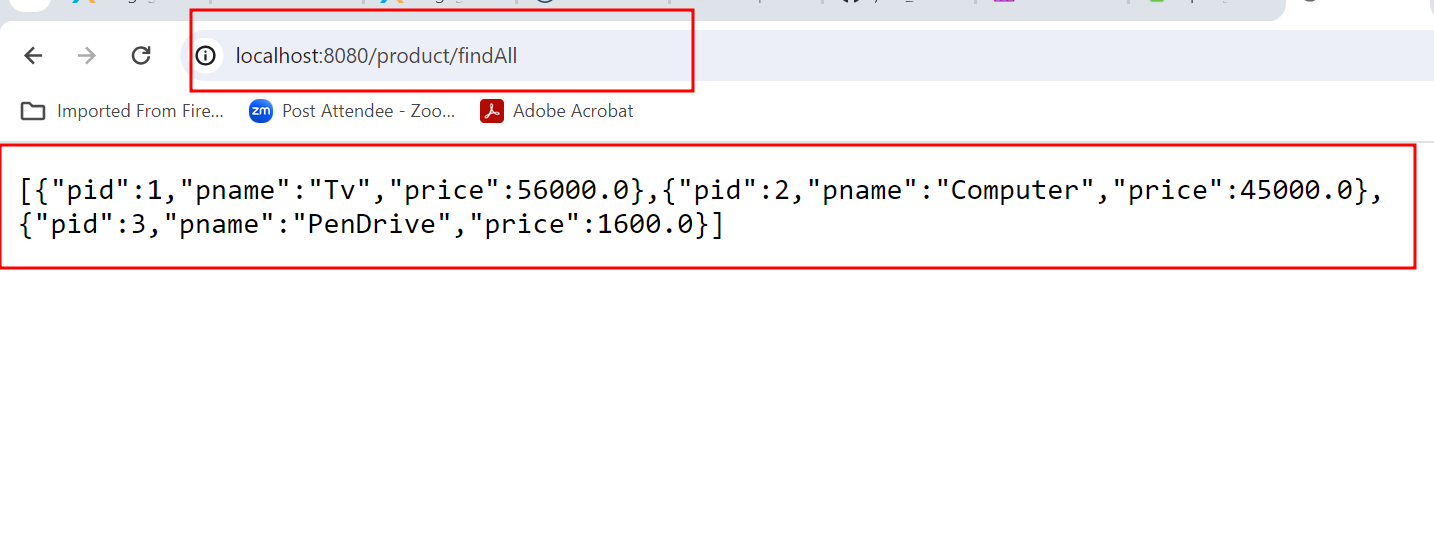
docker images

docker ps

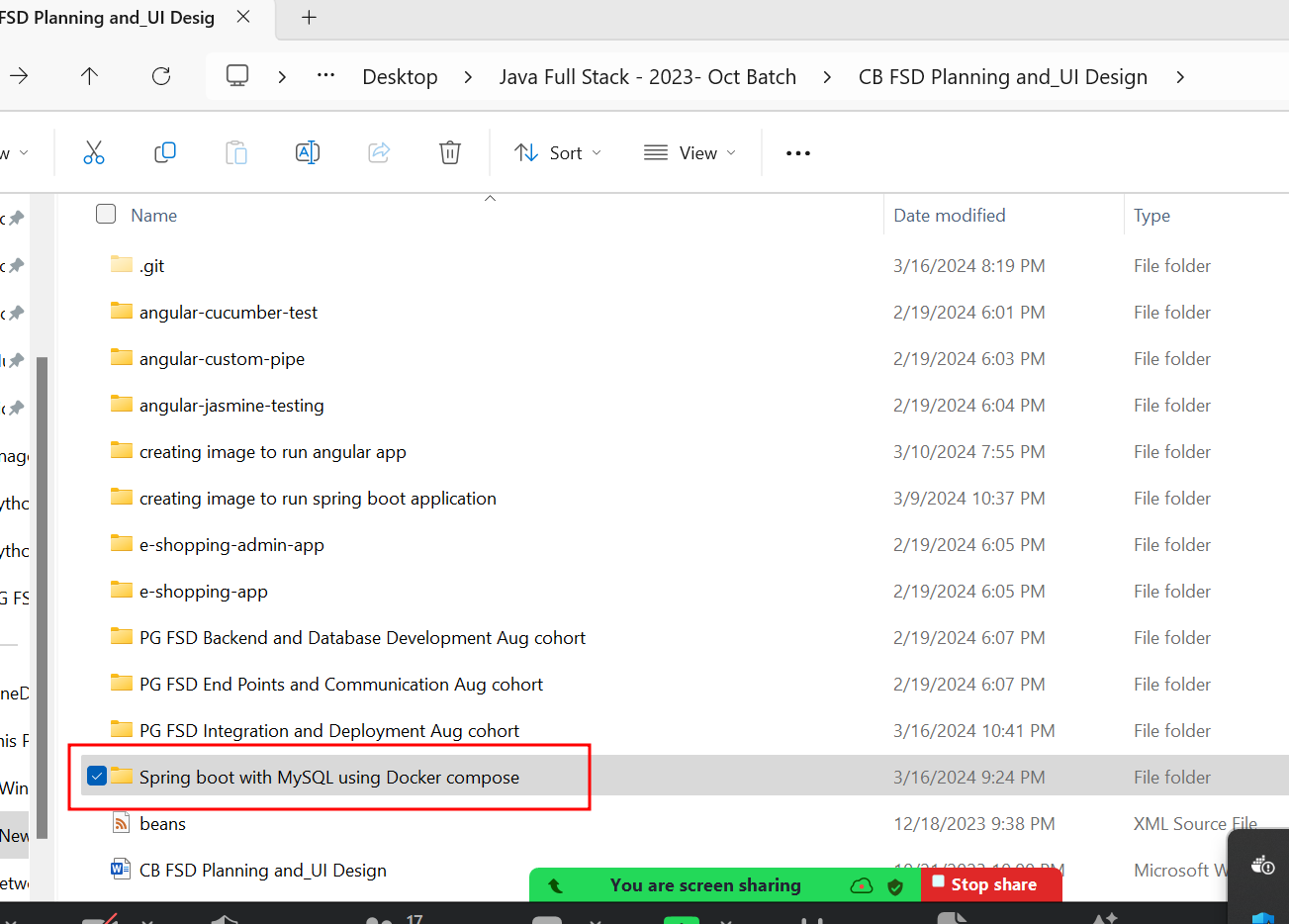
then store the record using post man client



Using browser to verify the data



Please refer below folder path for docker-compose project



Day 4 : 17 Mar 2024

CI and CD : Continuous Integration and Continuous Deployment or Delivery

Dev1

Login push

Dev2 push Shared Repository (Git Hub Account)

Application

Dev3 push

Feedback Create structure for project

Manager and push in remote repository

Manager need to merge the code and build the project which contains dev1 and dev2 code. It may build successfully or we get some error.

Continuous Integration means combining or merging more than one developer code.

If it build successfully we need deploy or delivery to other team for other purpose. If any error generate we need to inform to respective developer.

CI/CD

Dev1 java

Dev2 shared repository CI/CD tool

Machine (build more than

One developer merged code)

Dev3 python when ever we push the code in shared repository. CI/CD tool pull the code and build the project (compile program, run the program, test the program, creating jar or war file etc). if anything go wrong it inform to respective developer else it pass this code to other team or client or production environment.

CI and CD tools

Jenkin : Jenkin is a type of open source automation CI and CD tools. Jenkin internally base upon java technologies. Jenkin is plugin base ci and cd tools.

Jenkin GUI base Ci and cd tool

1. We can download the software respective OS and do the Jenkin task.
2. We can download the Jenkin.war file and with the help of external tomcat software we can run this war file.
3. We can run Jenkin using docker image

2 better options

Download the war file.

Open the command prompt

minimum java must be 11

java -jar jenkins.war by default port number 8080

<http://localhost:8080>

or

java -jar jenkins.war --httpPort=9090

<http://localhost:9090>

it will ask some password.

Password present in your terminal. Please copy and paste and click next.

It display two options installed plugin

Installed suggested plugin

1. First job to display simple Echo message.
2. Second job we use trigger option to execute set of command again and again.
3. Third job is going to pull the project from git remote repository and we run Java program ie javac and java

23 – Mar – 2024 Day 5

CI and CD tools

Build the project : compile the program, run the program, create the jar file or create war file, run the test etc.

java -jar jenkins.war

<http://localhost:8080>

or

java -jar jenkins.war --httpPort=9090

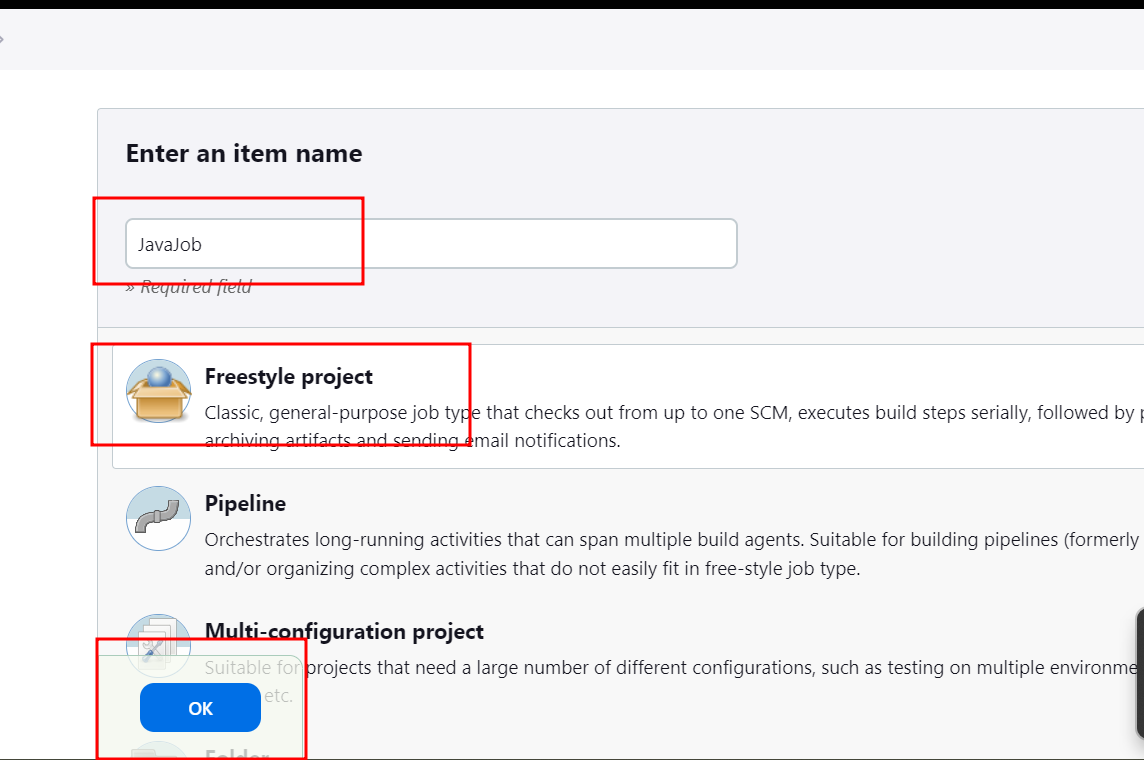
<http://localhost:9090>

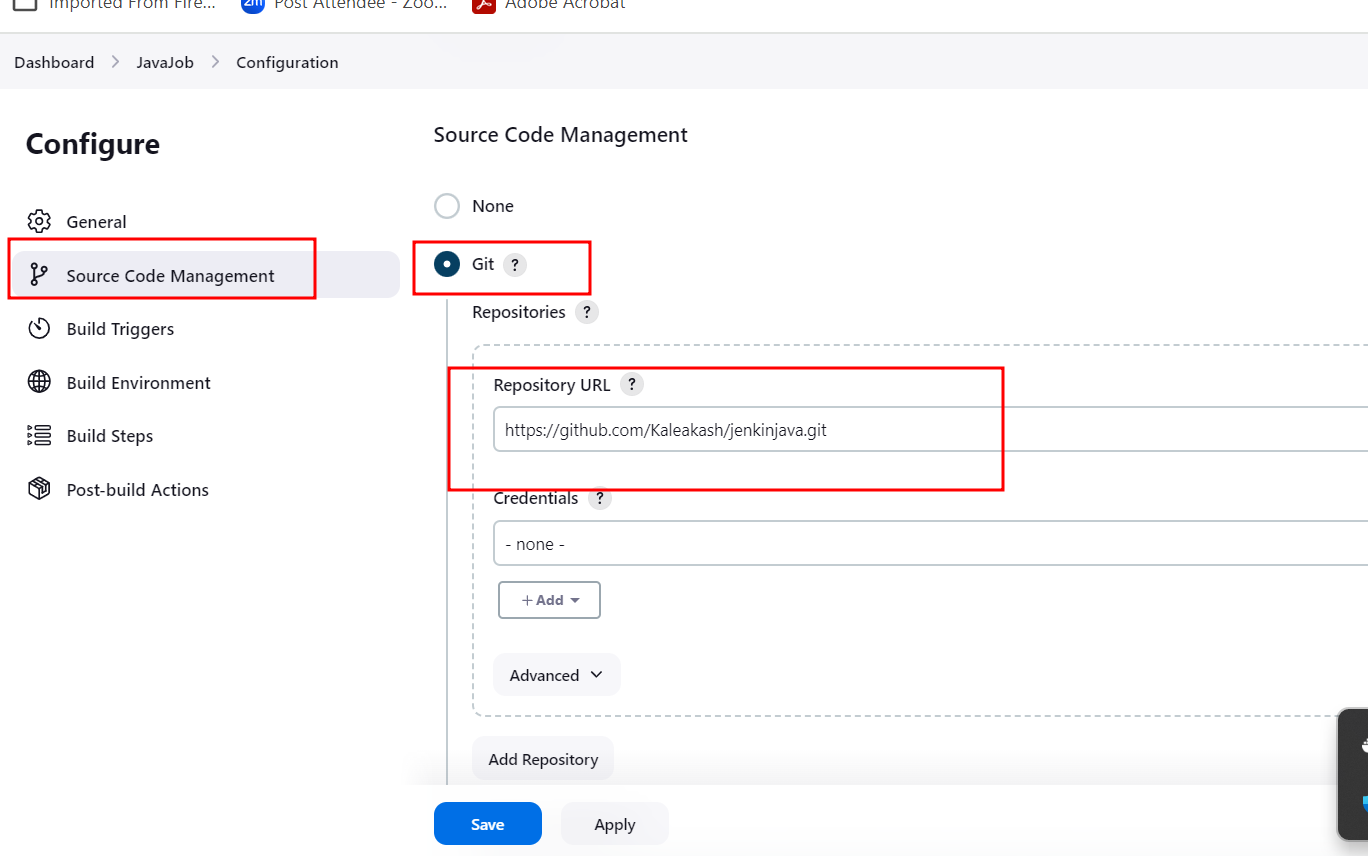
ls

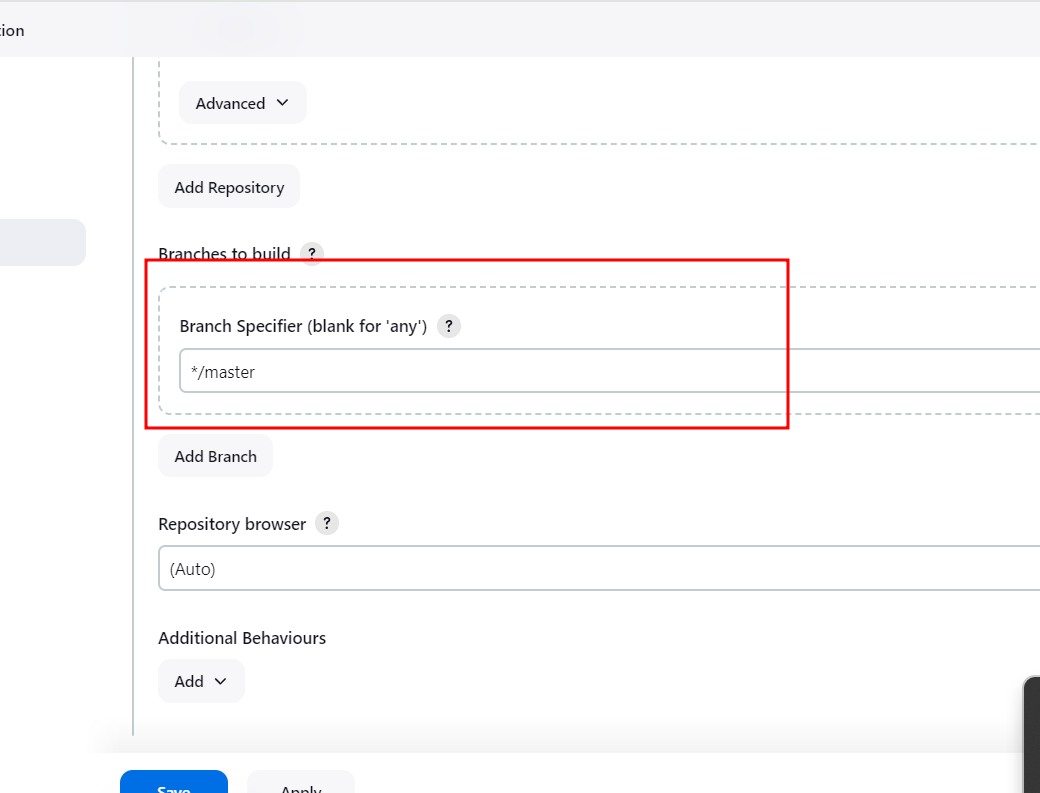
**Running Jenkin software using Docker**

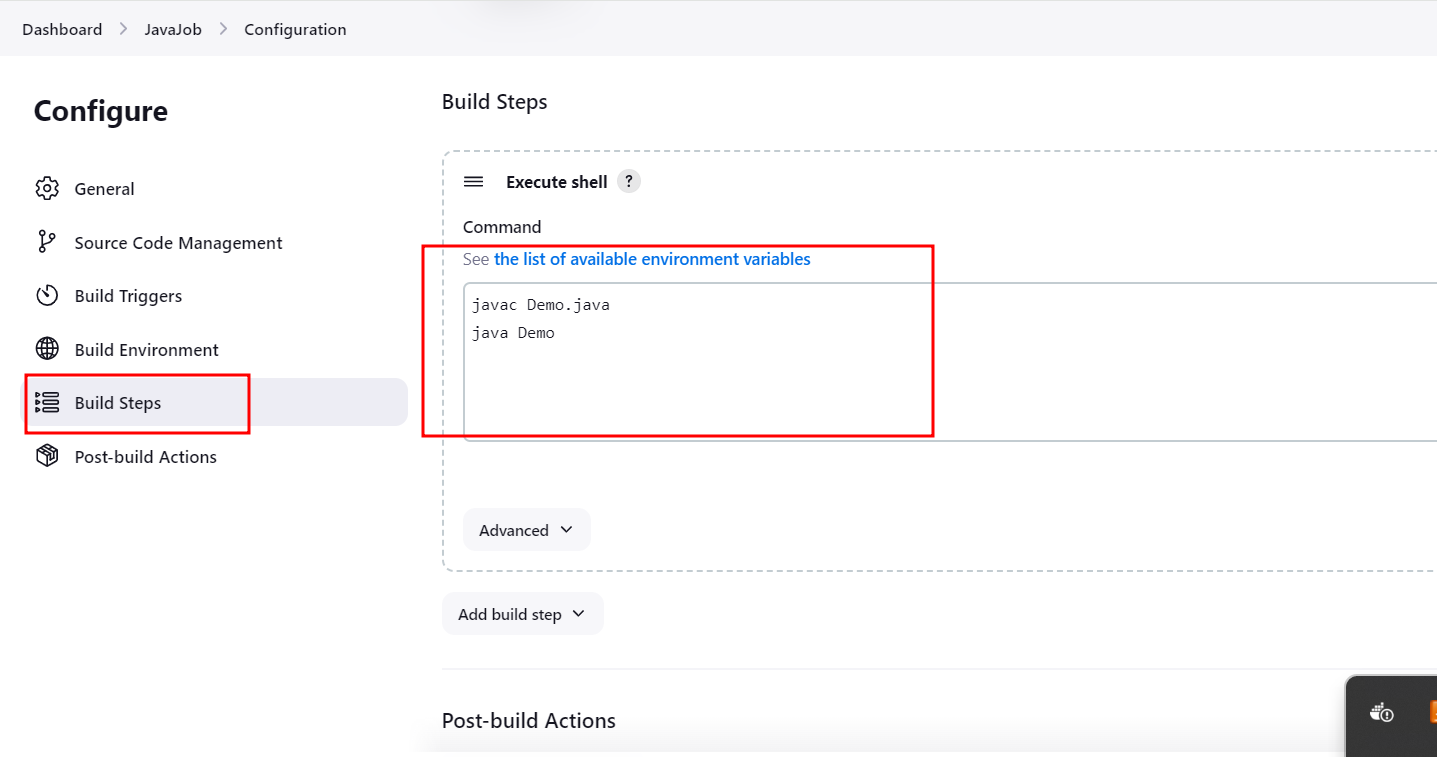
docker run -p 8181:8080 -p 50001:50000 --restart=on-failure jenkins/jenkins:lts-jdk17

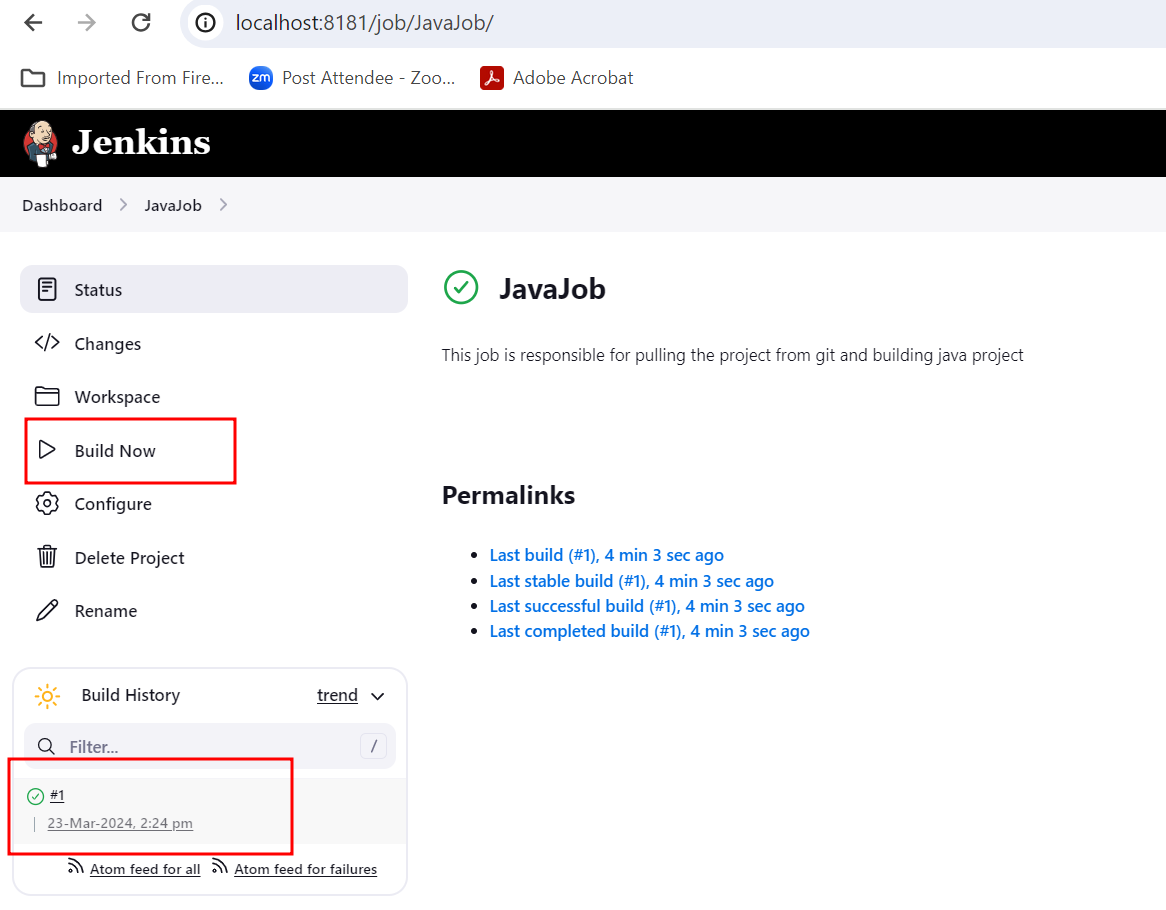
Creating Job to pull project from github and run java projects.

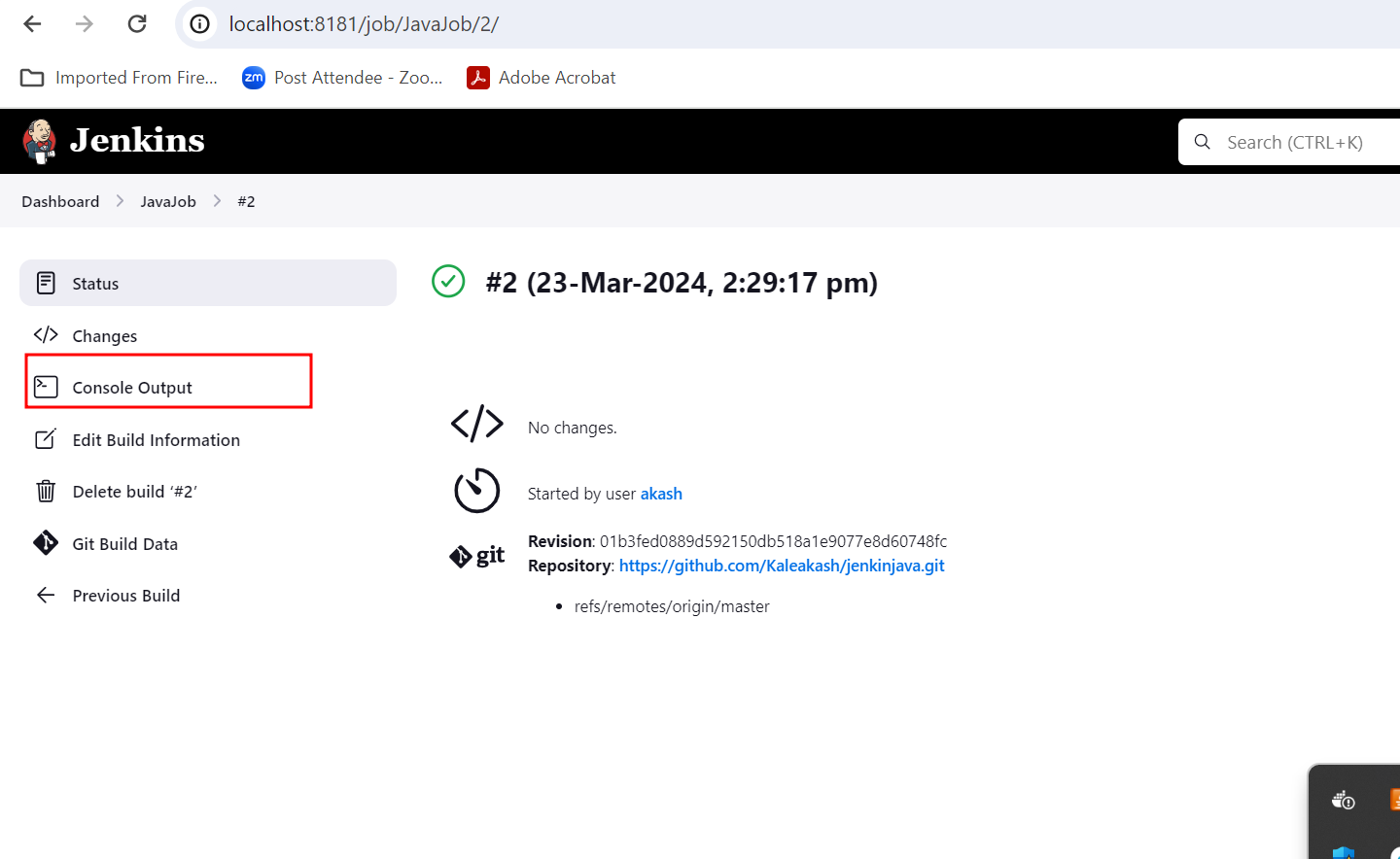


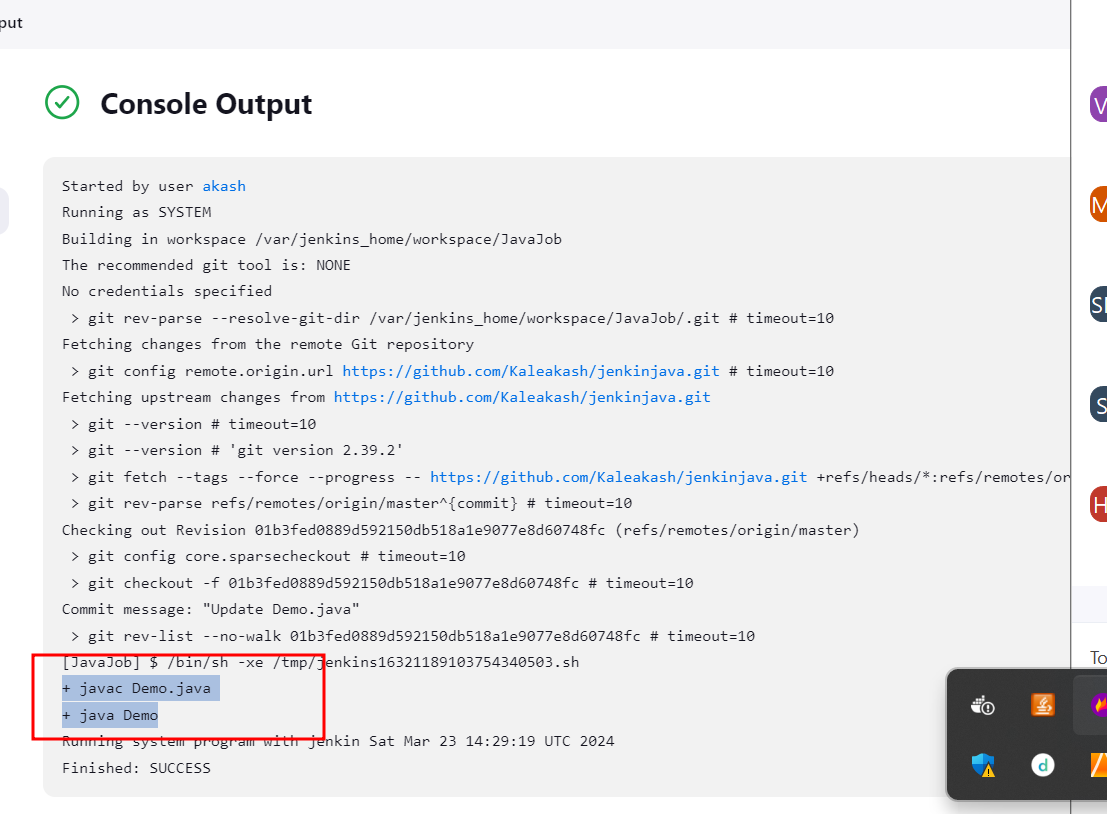




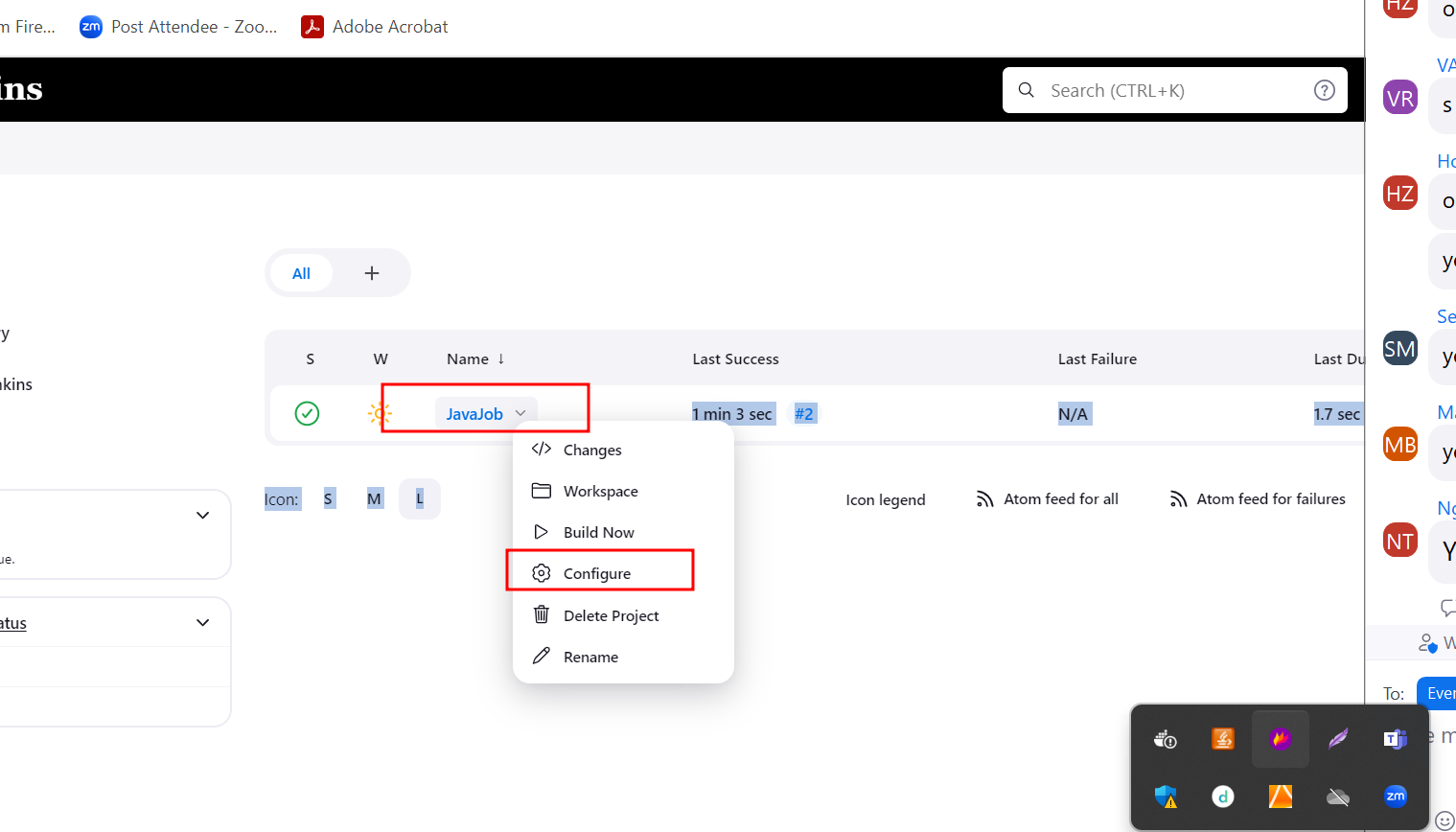


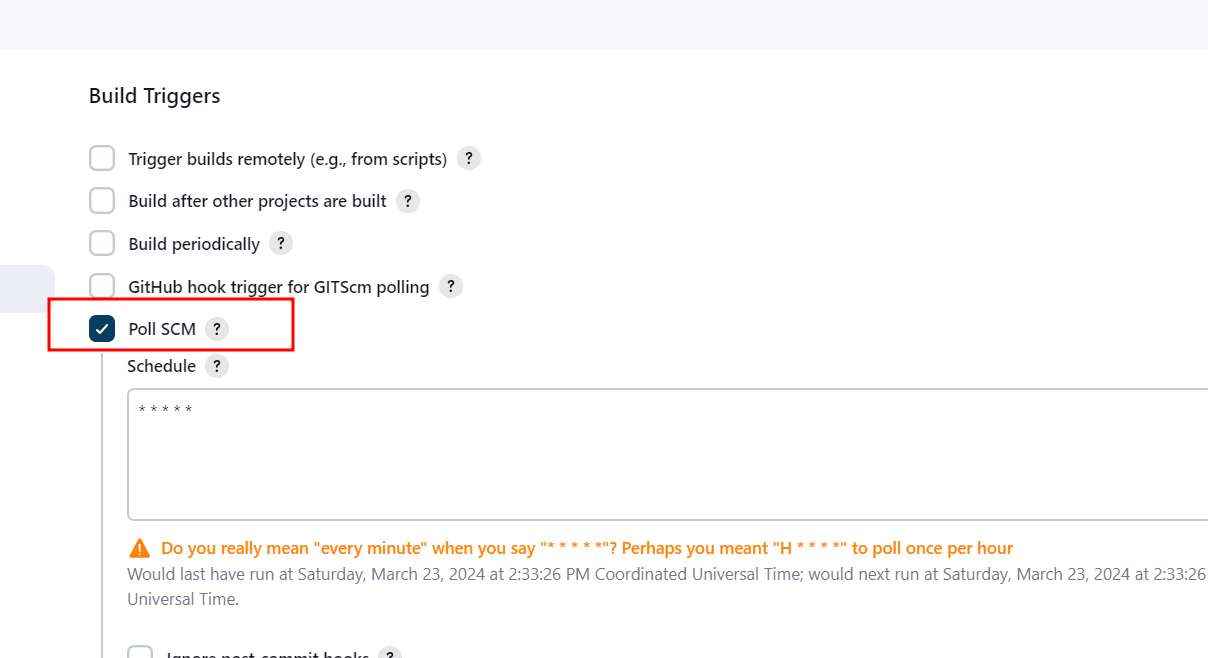






Now in same job we add trigger option





Poll SCM. Base upon time we provided it check again and again in git hub account. If changes happen in remote repository in specific branch then it build once again.

We create another Jenkin job to run spring boot projects

mvn clean : this command is use to clean previous build project

mvn compile this command is use to compile all java classes part of maven spring boot project.

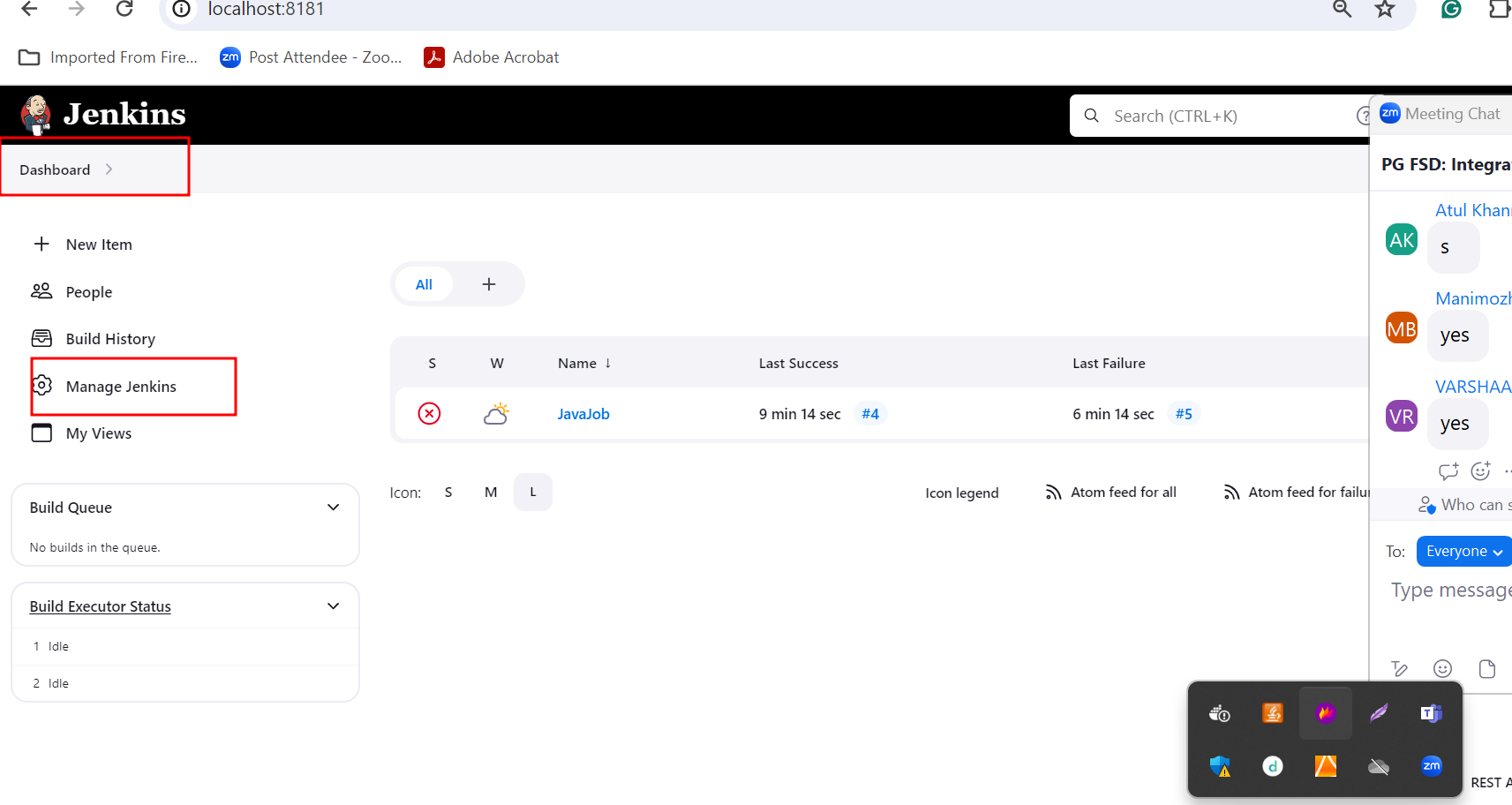
mvn test this command is use to run the test case

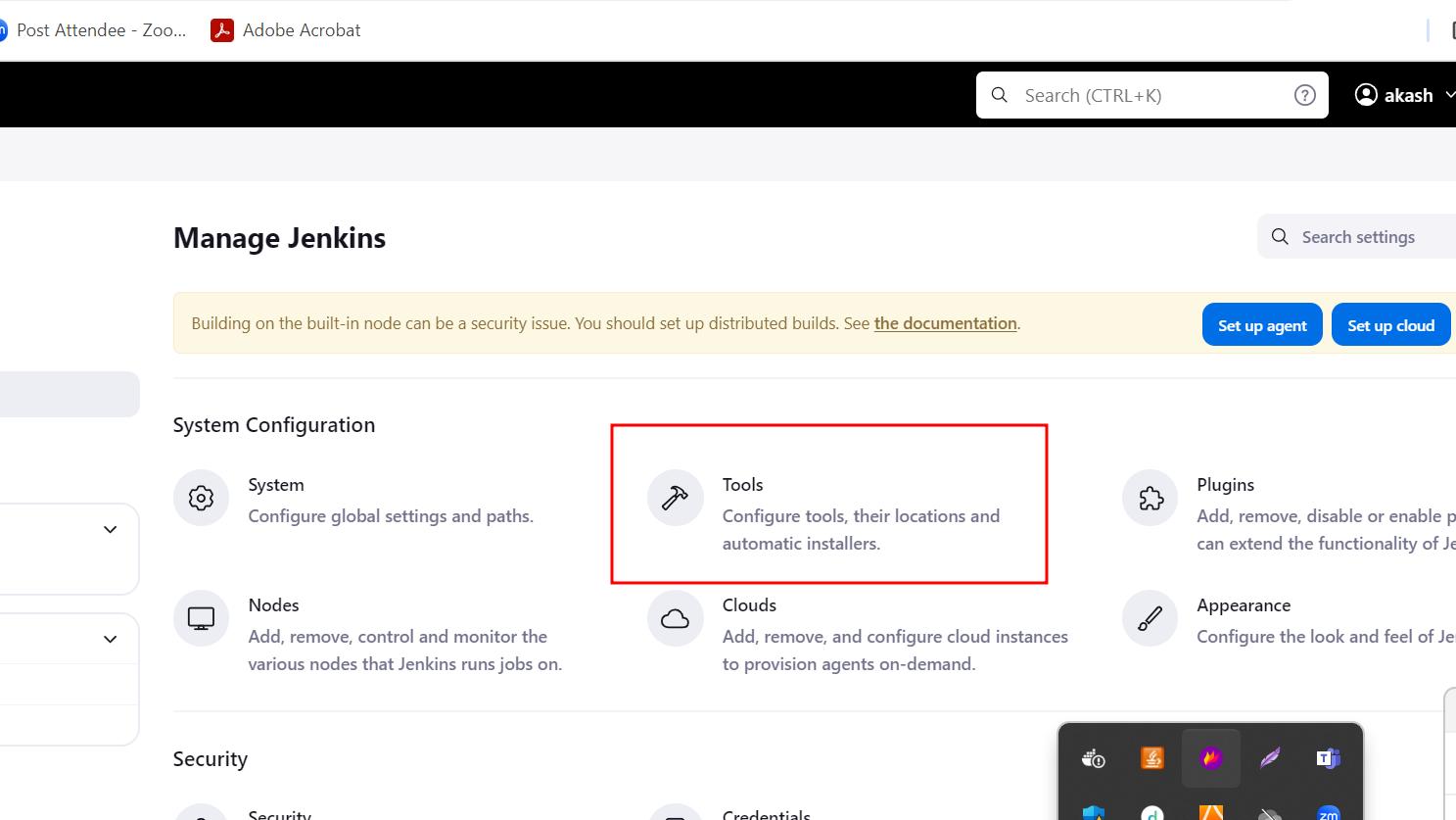
mvn package this command is use to create jar or war file for spring boot maven project

by default maven software doesn’t provide by Jenkin dashboard.

We need to configure maven software in Jenkin dashboard

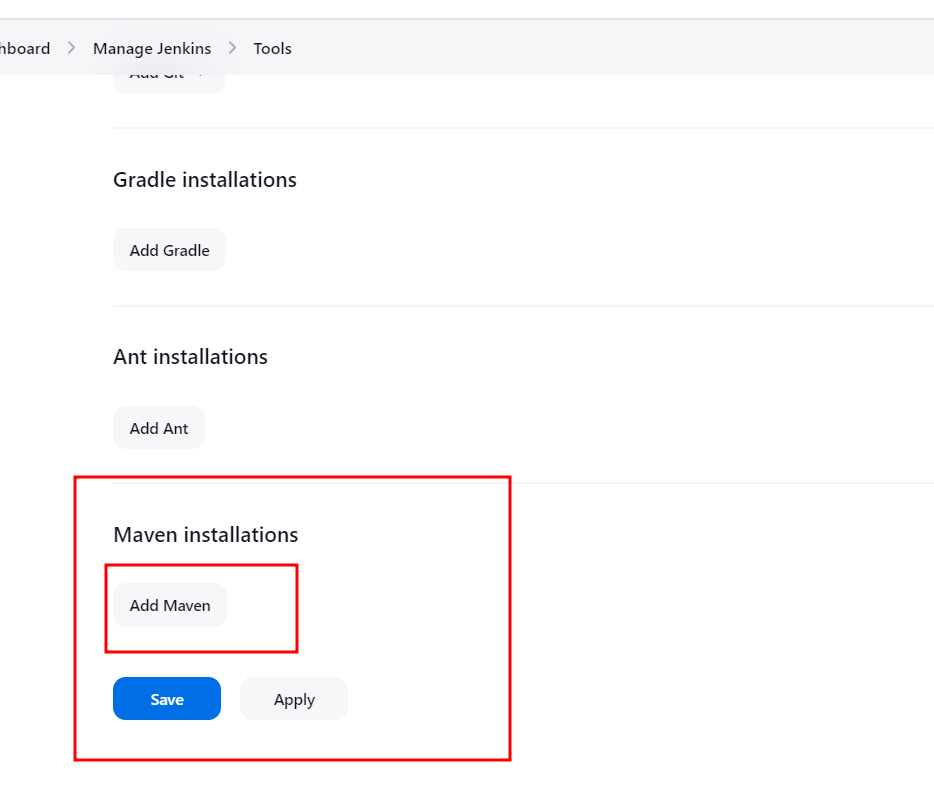
Steps to configure maven in Jenkin dashboard

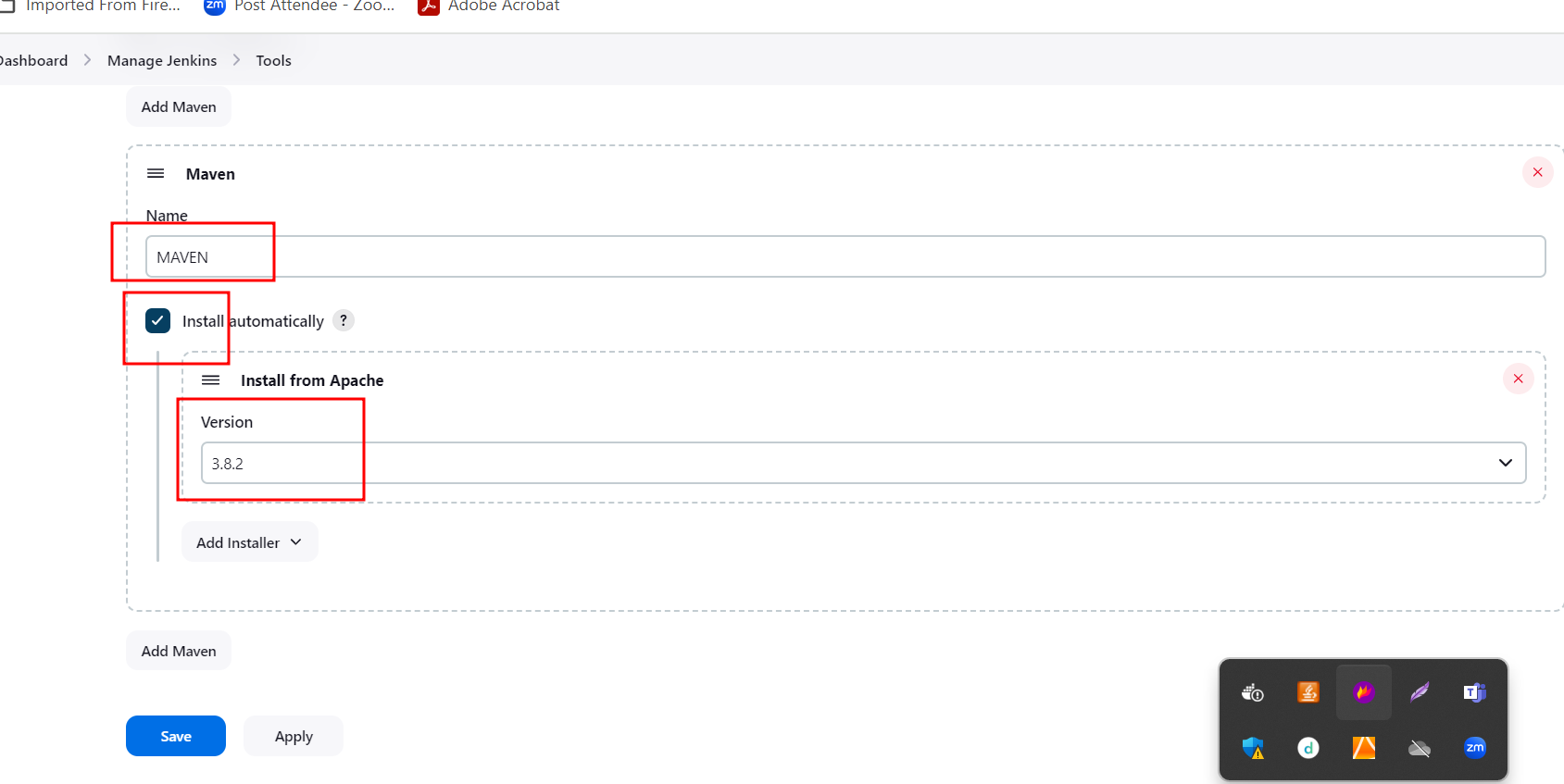




By default Jenkin provide

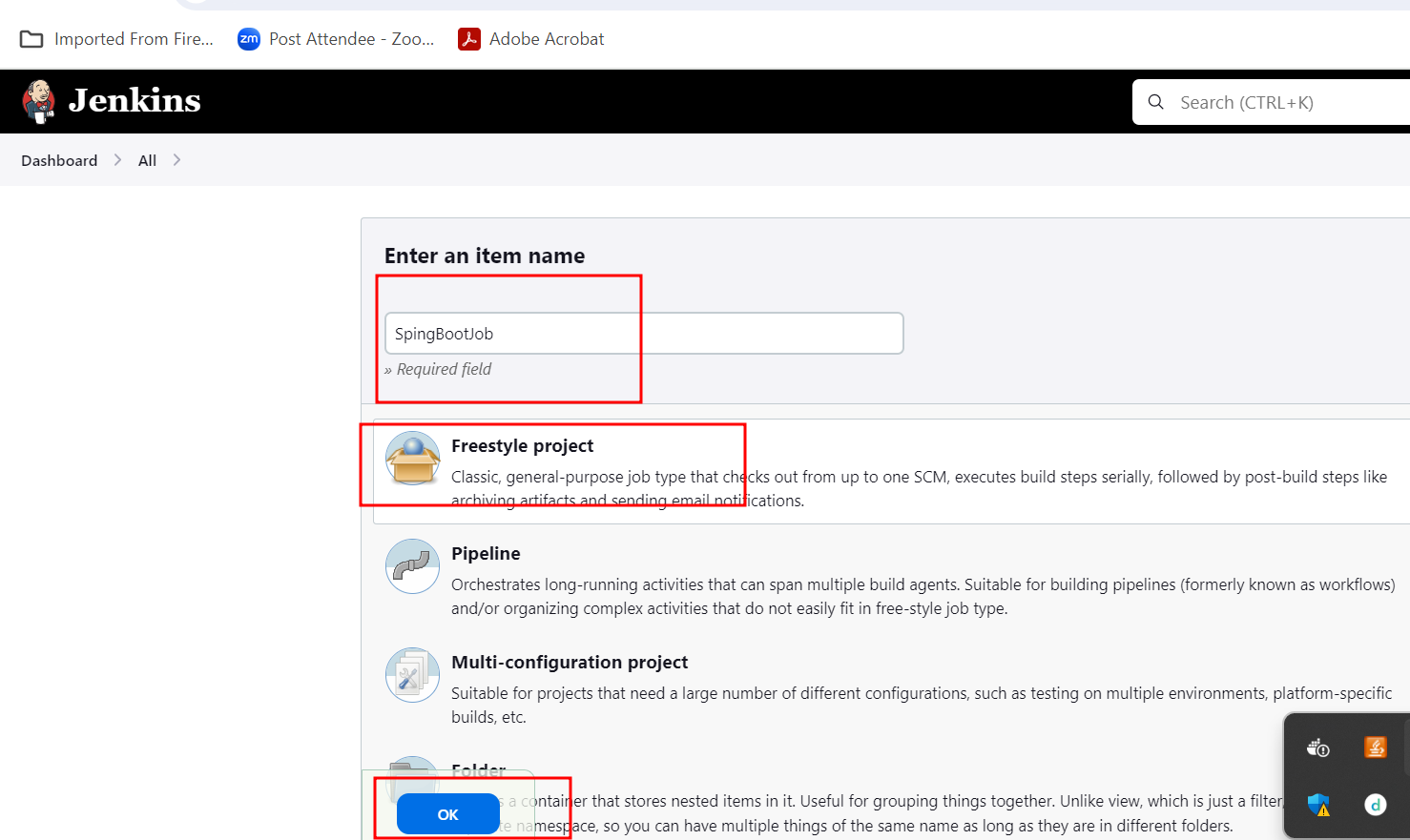
Maven, ant, Gradle, java, and git tools.

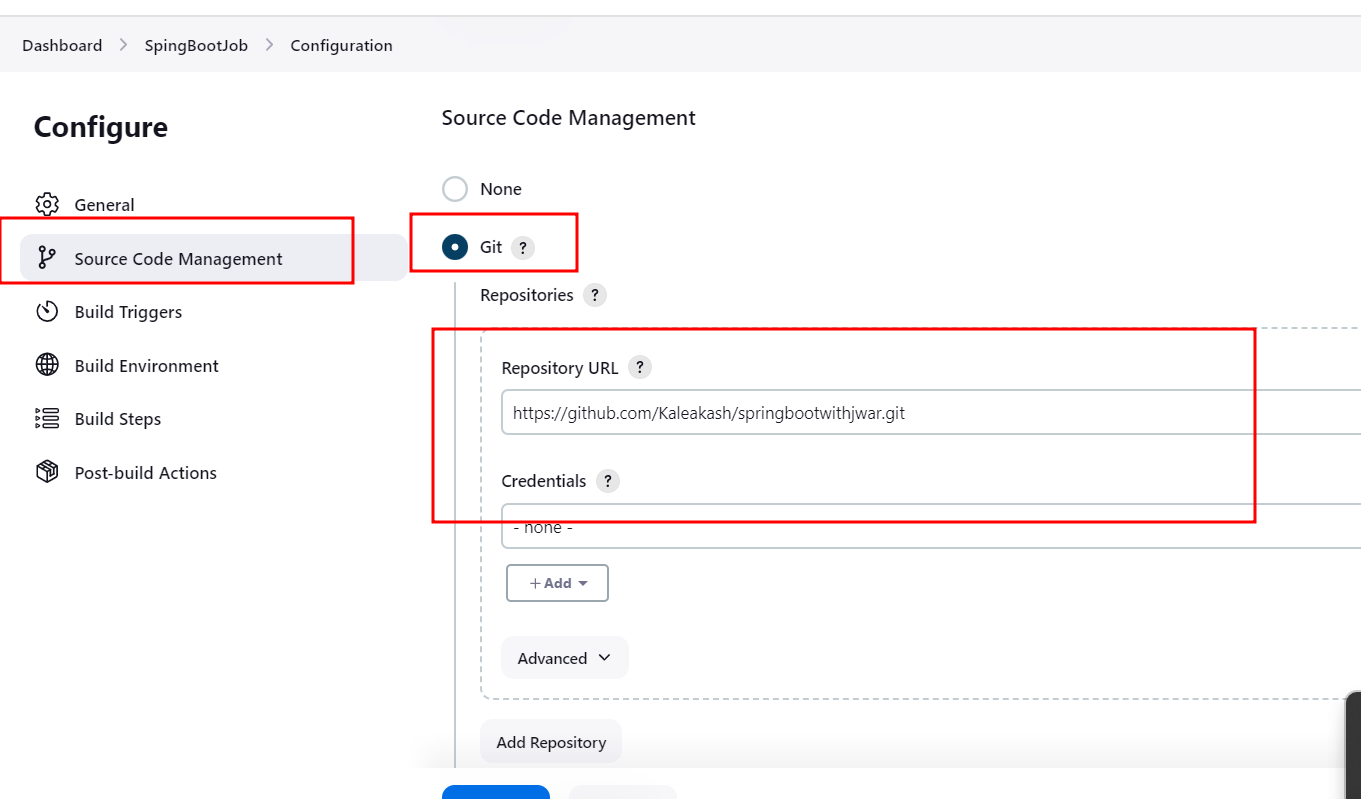


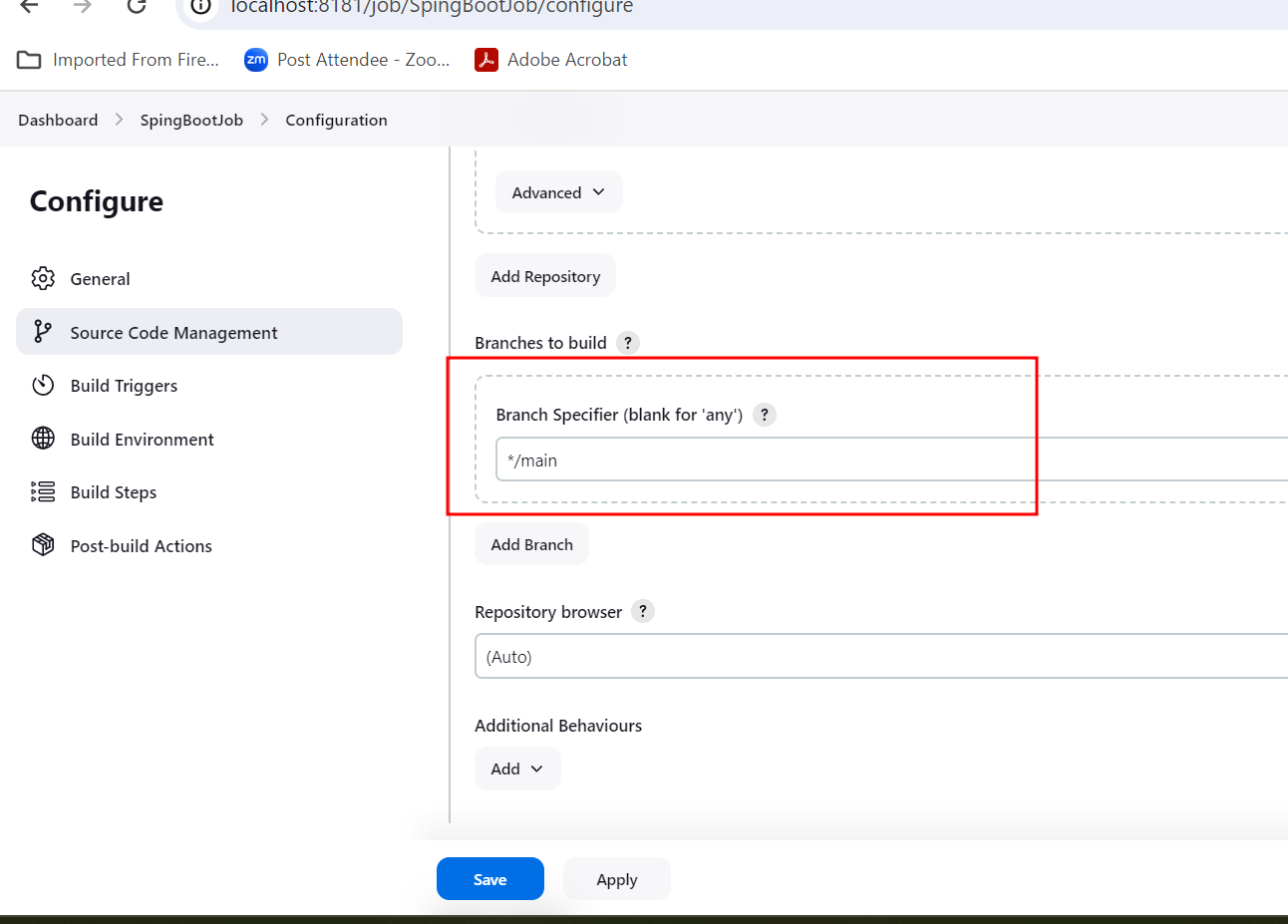


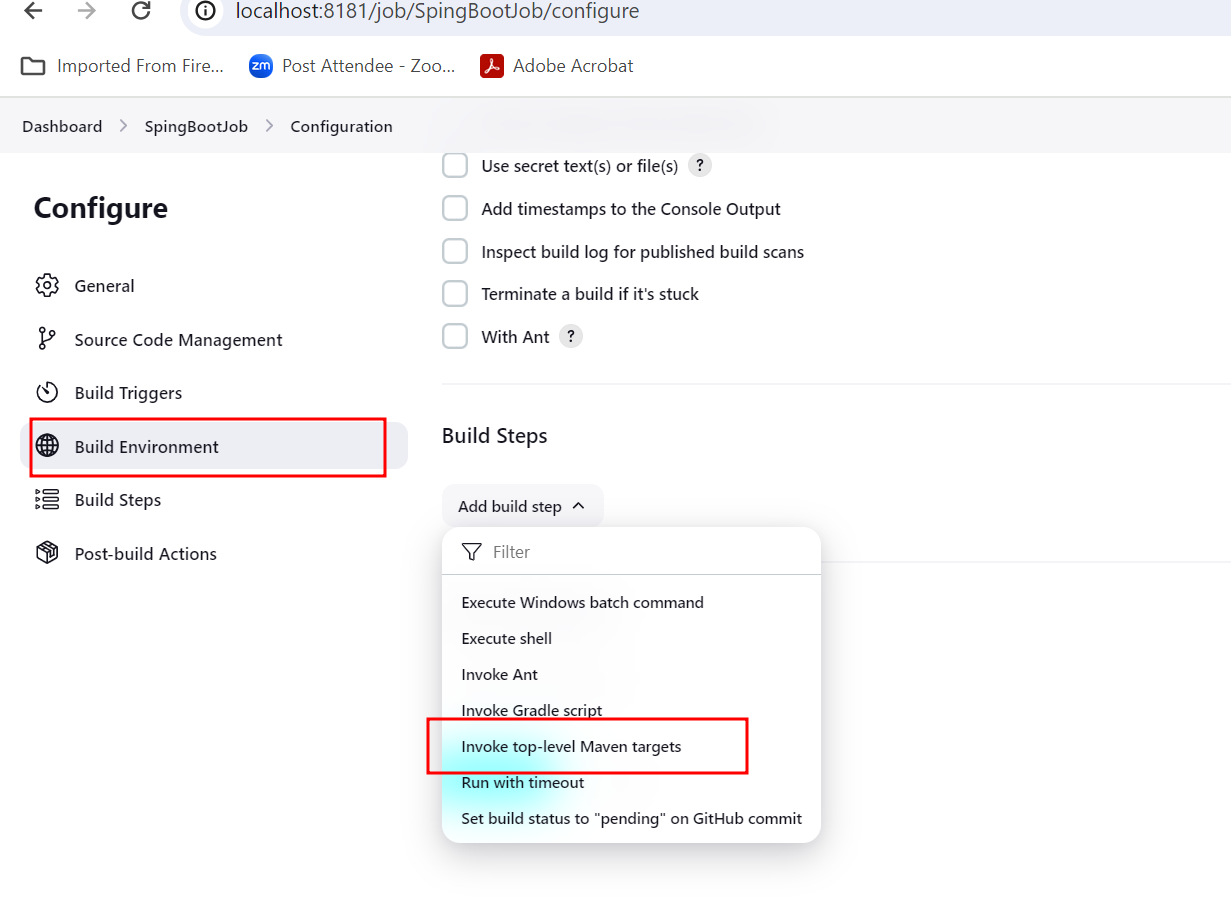
Now we are creating Jenkin job to build spring boot maven project this project responsible to create war file.

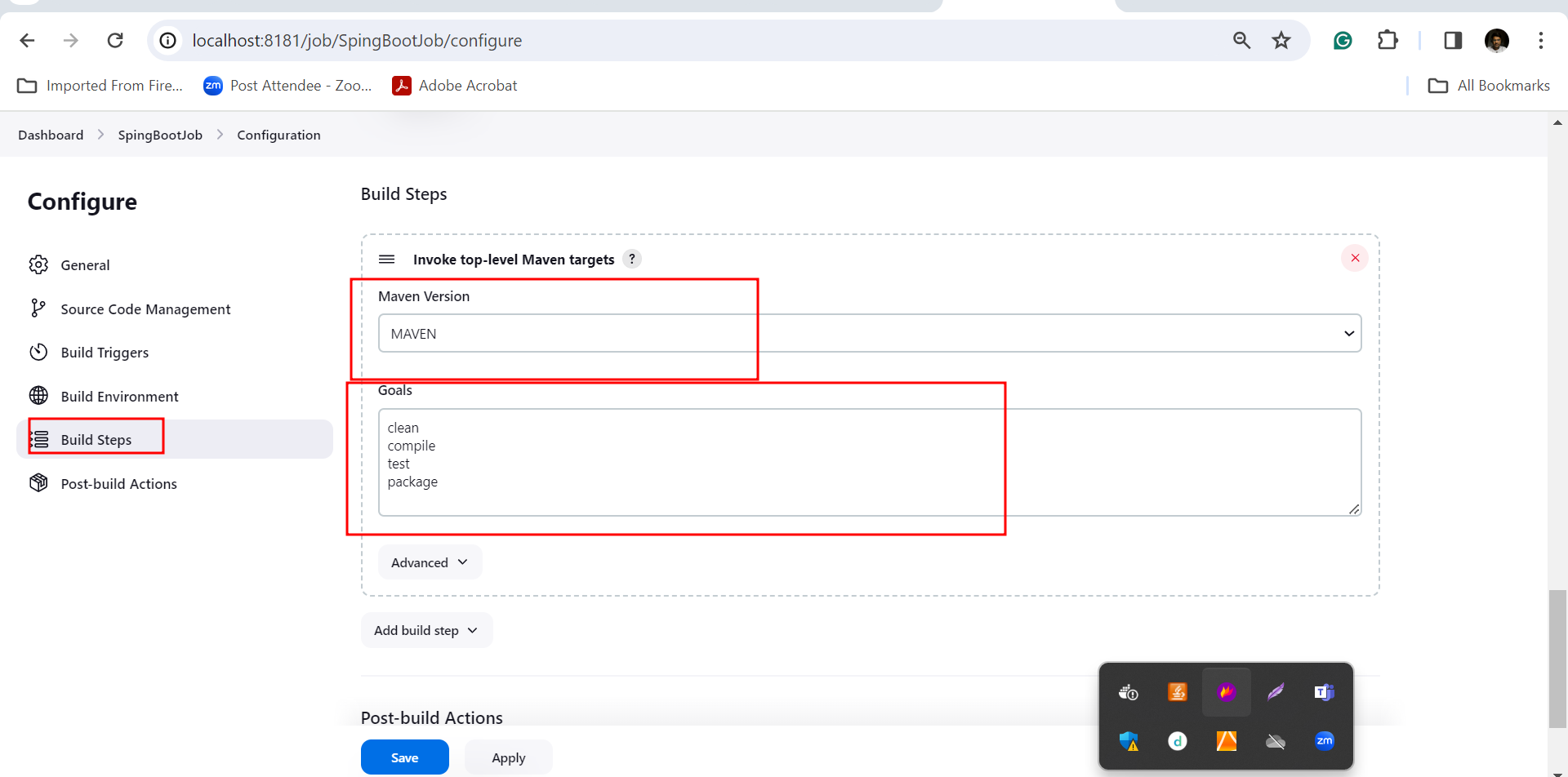
<https://github.com/Kaleakash/springbootwithjwar.git> with branch name as main

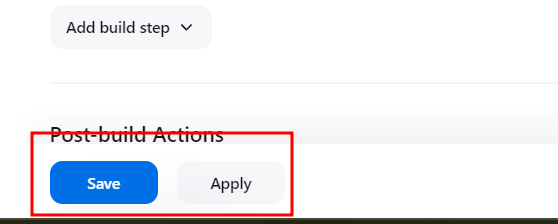


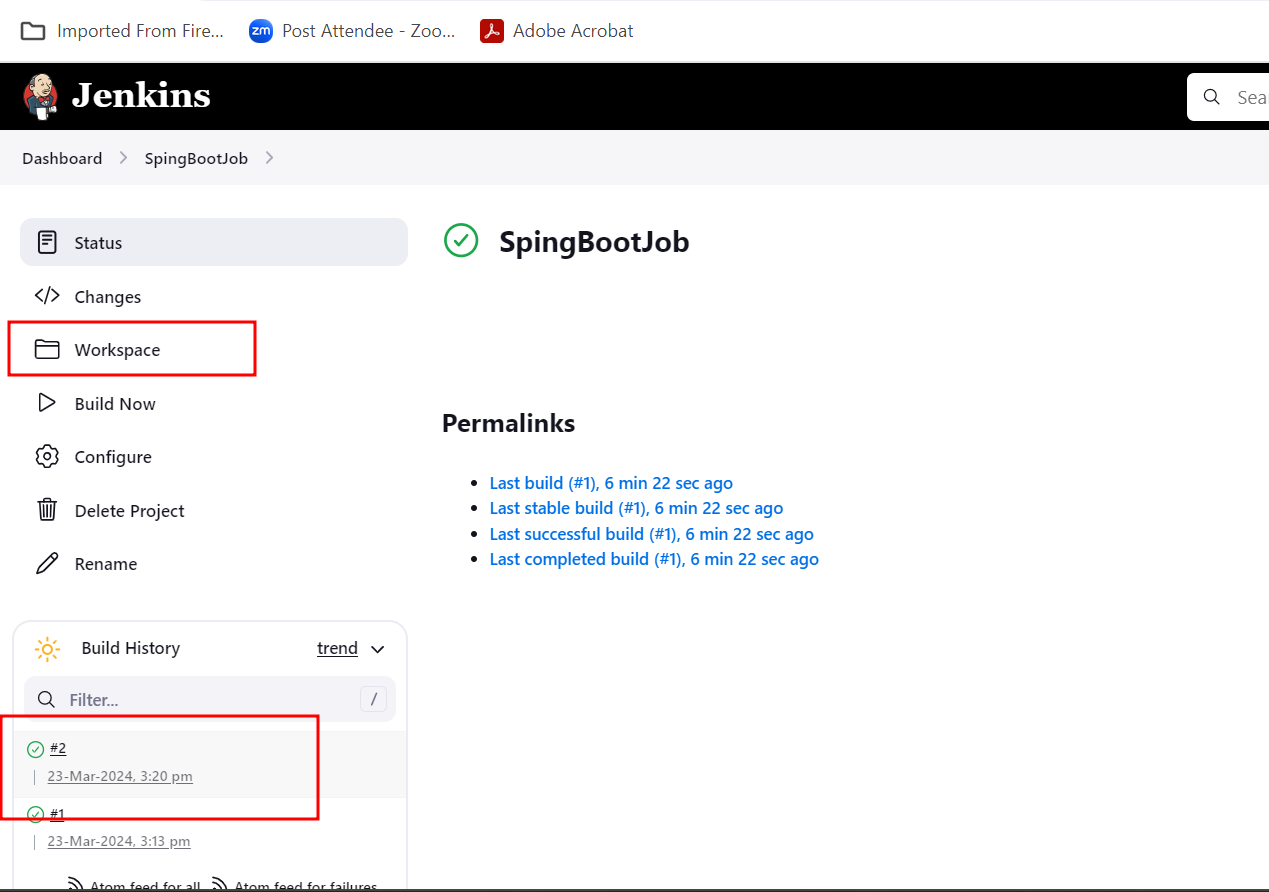


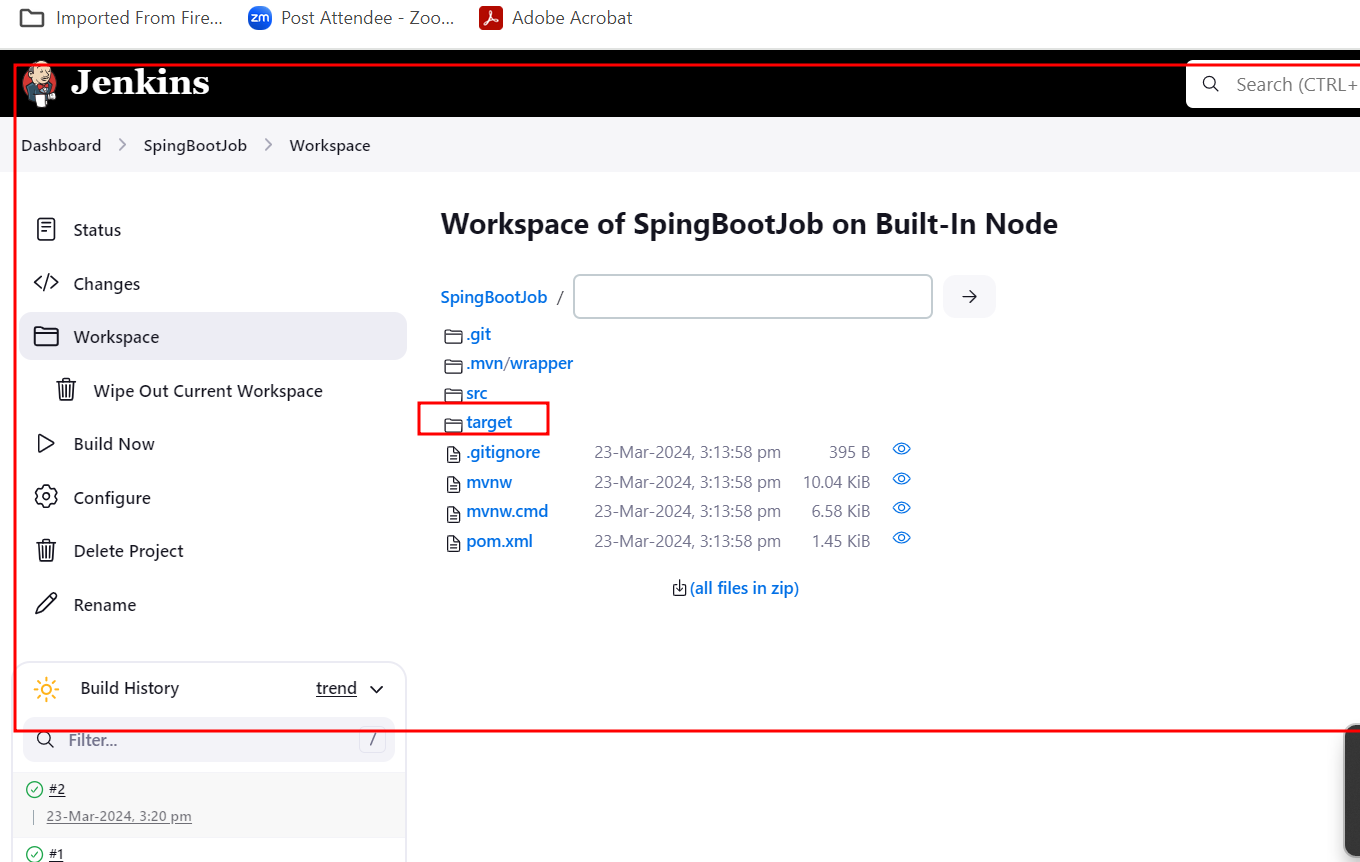


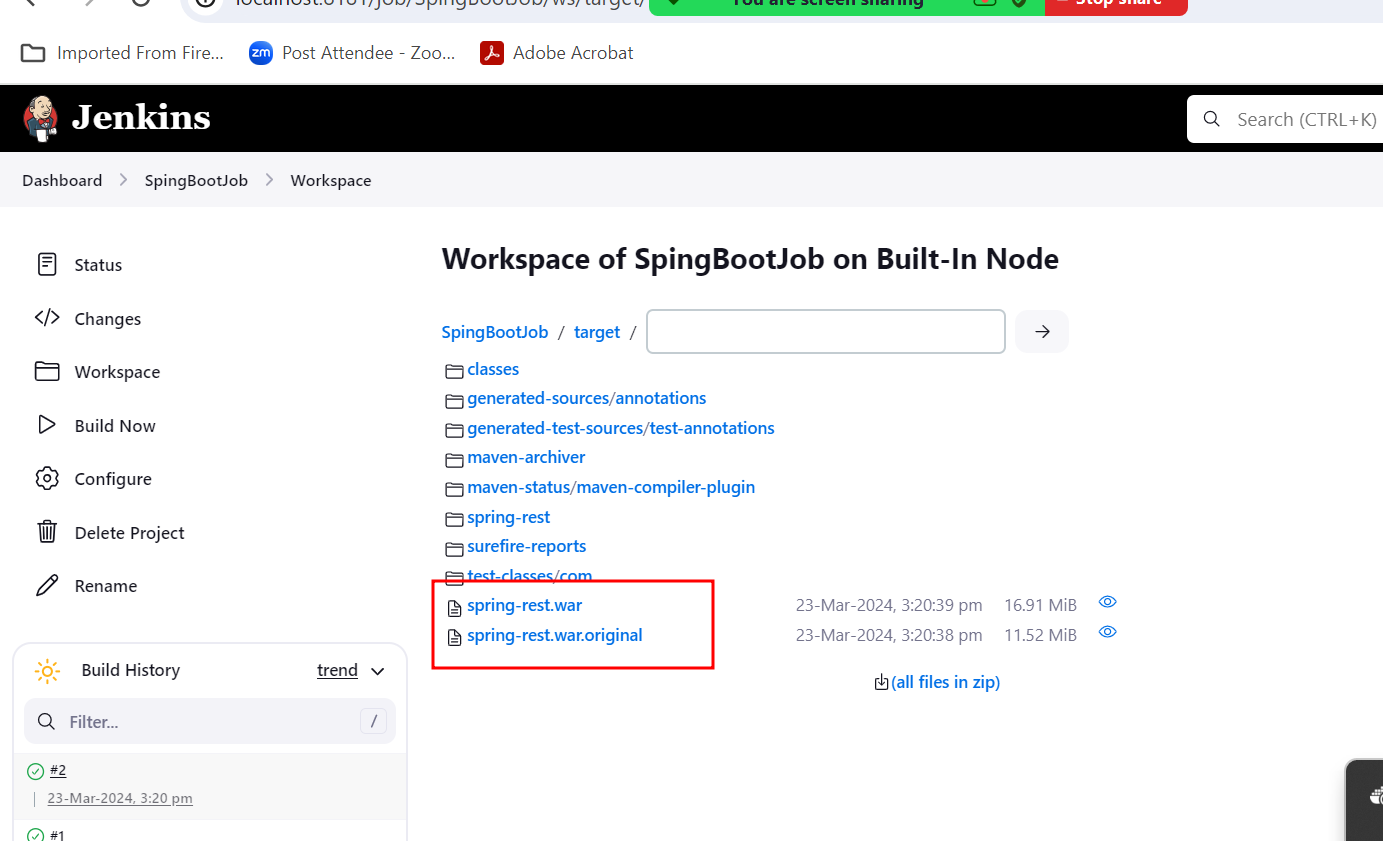












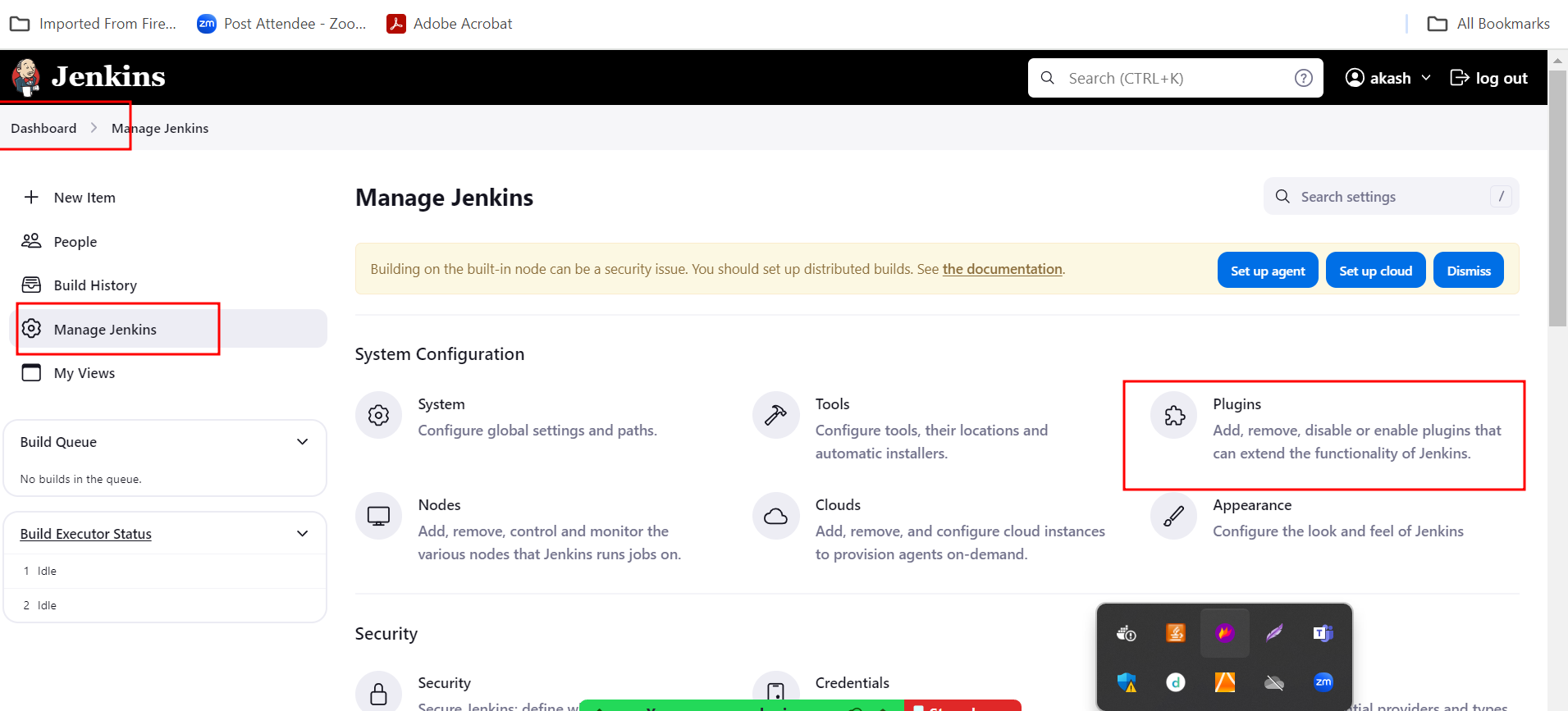
Creating job to build angular projects

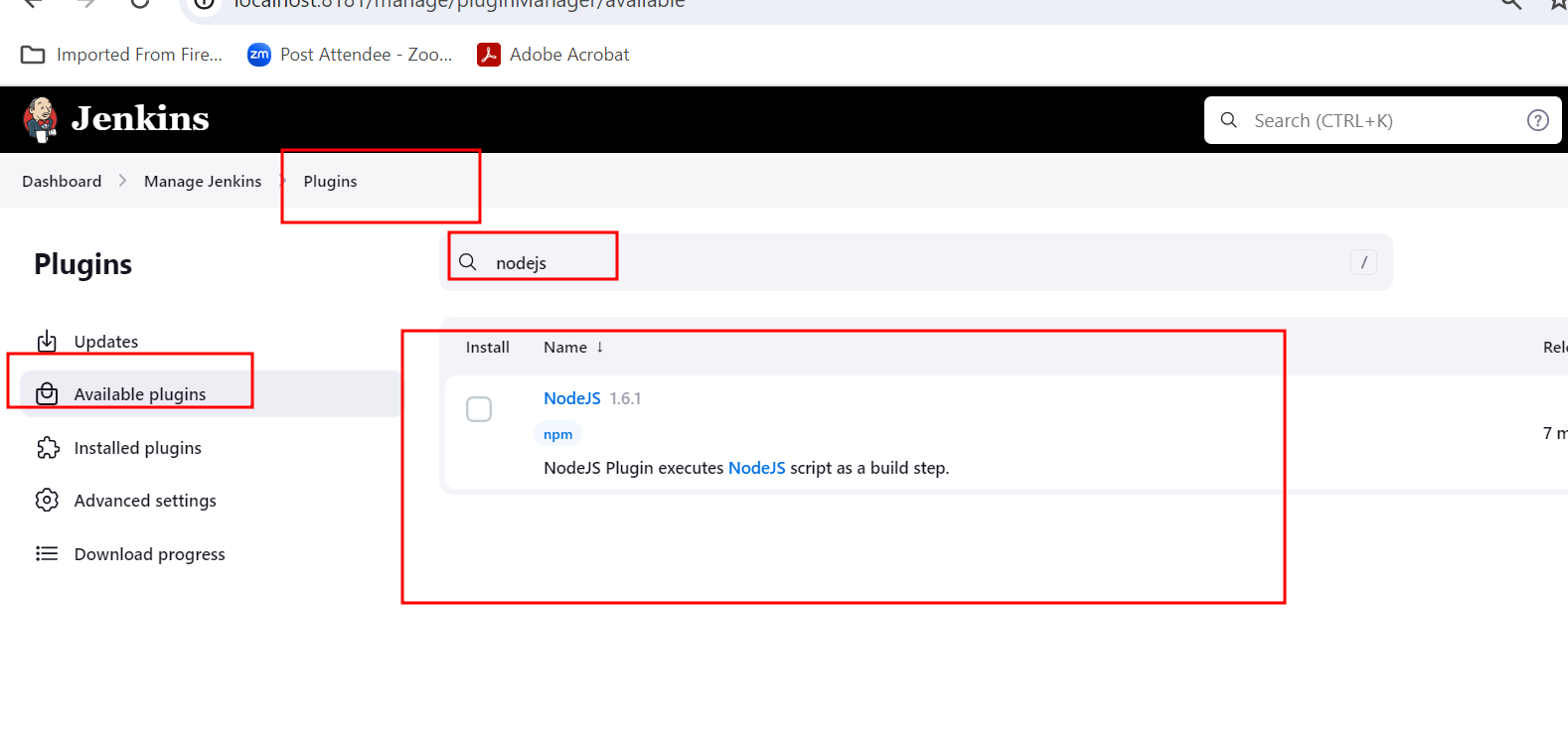
To build the angular project we need to node js software.

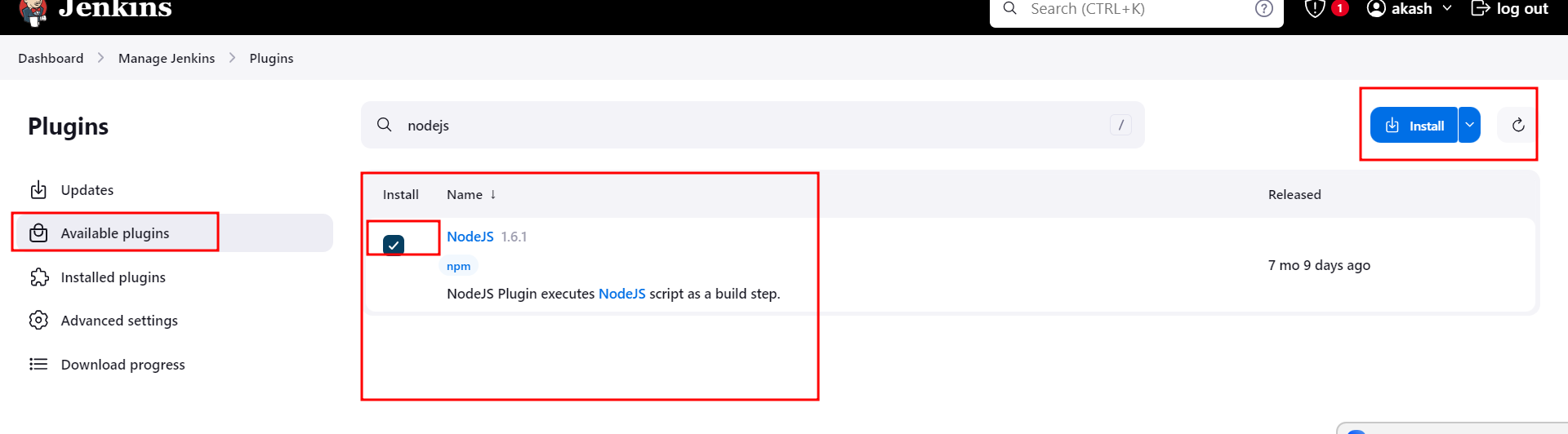
By default node js software not present in tool options

Using plugin we need download node js plugin software.

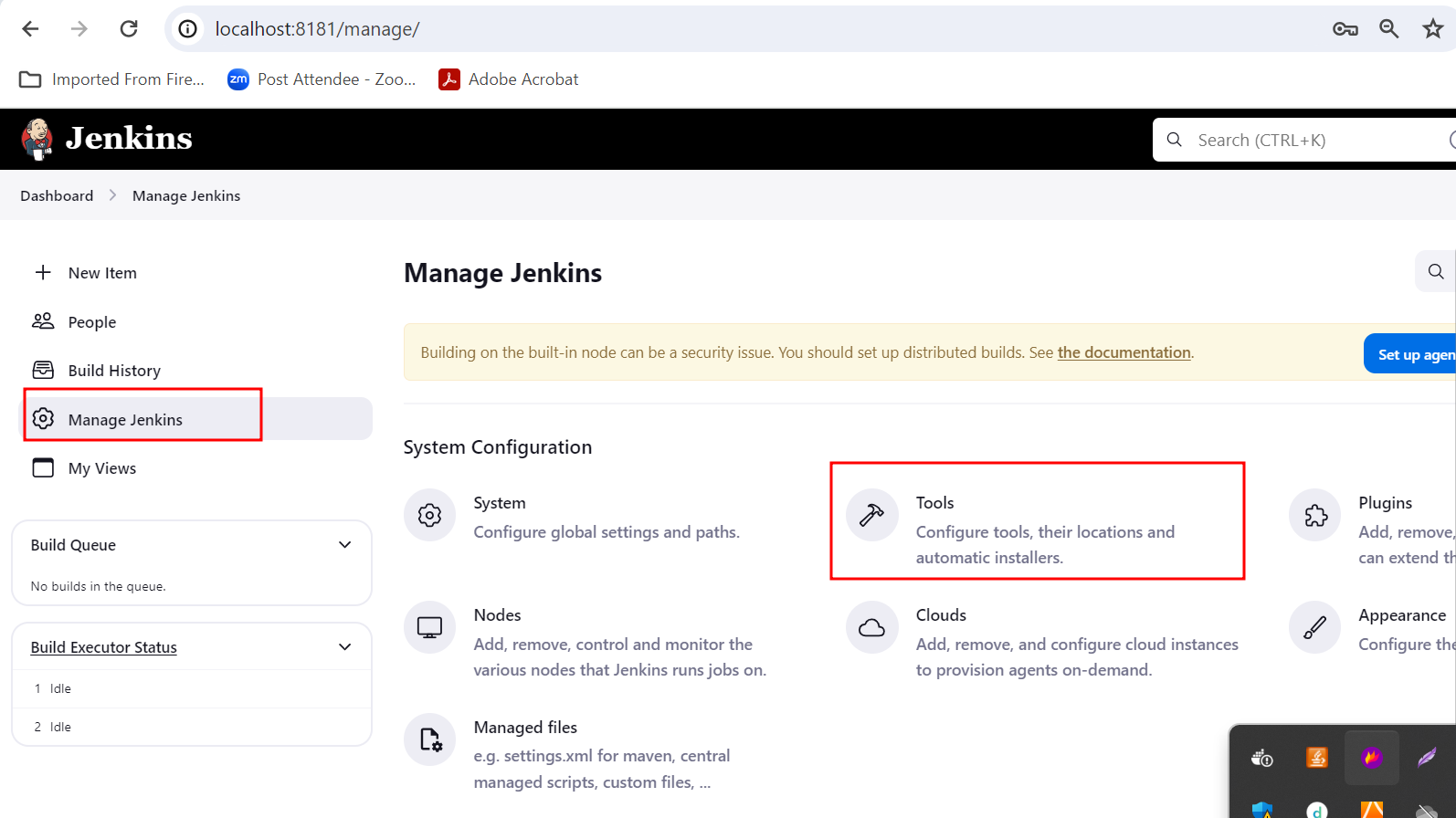
Adding new plugin depending upon the requirement ie node js plugin

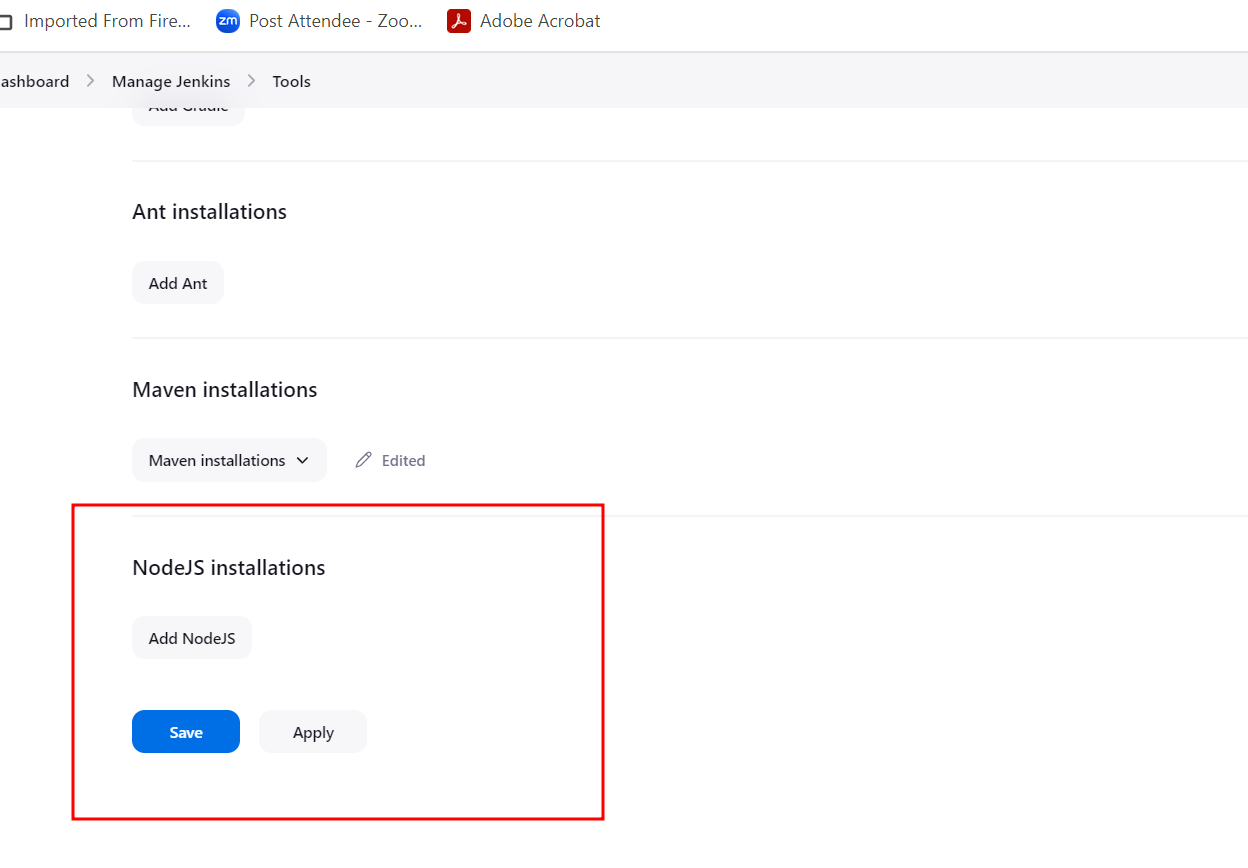


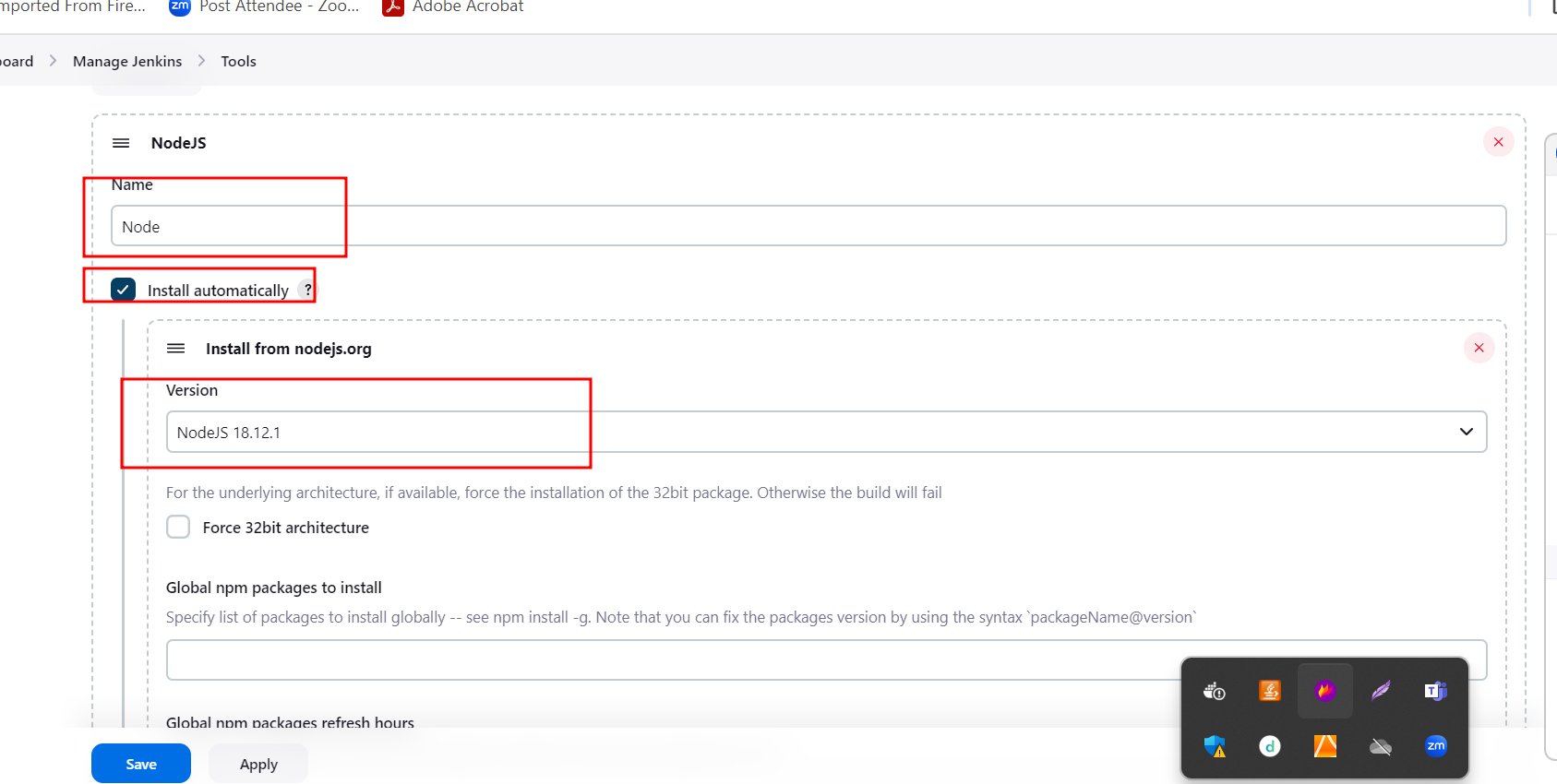




To verify node js plugin added or not we need to go to tools option.



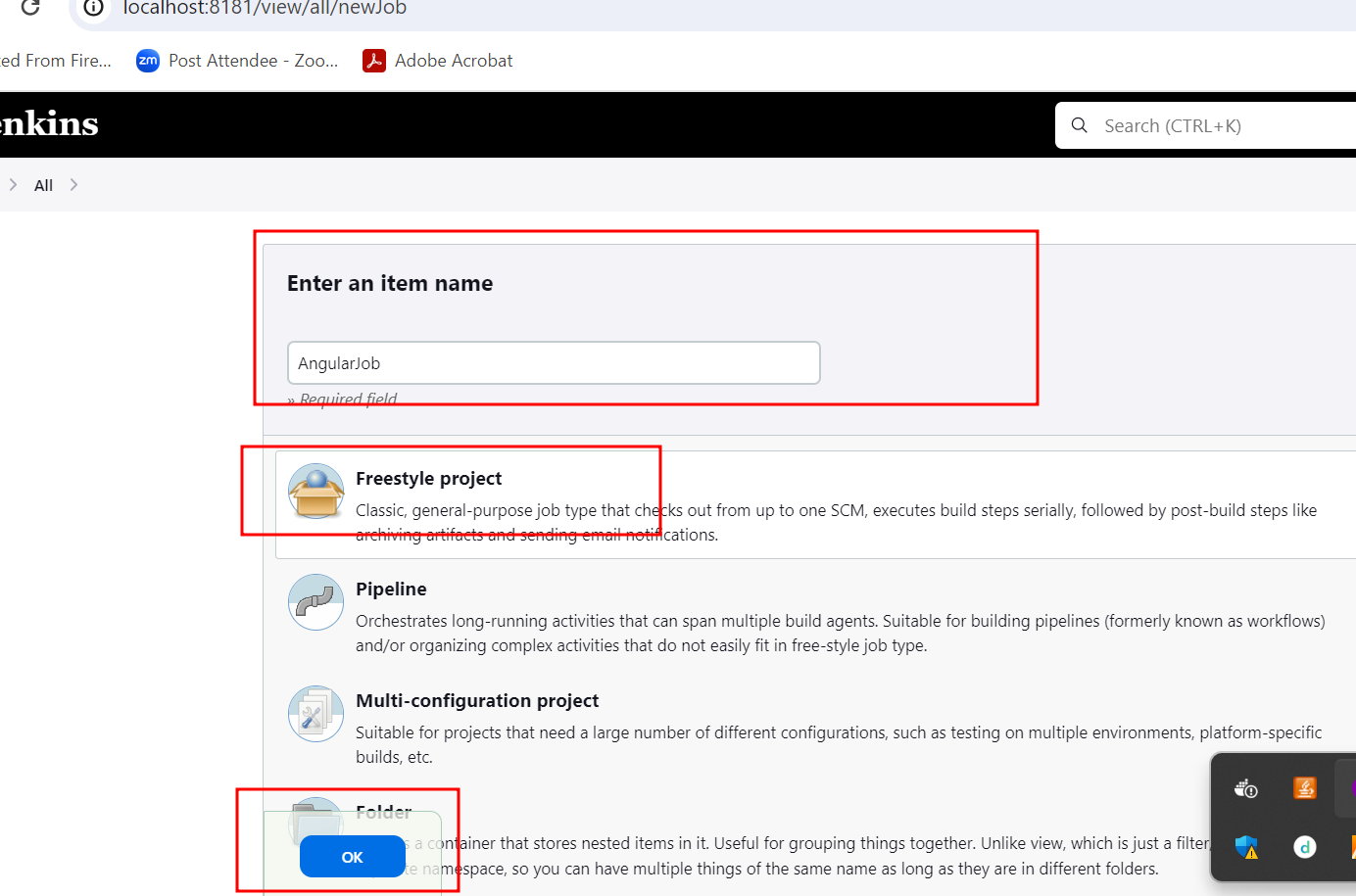


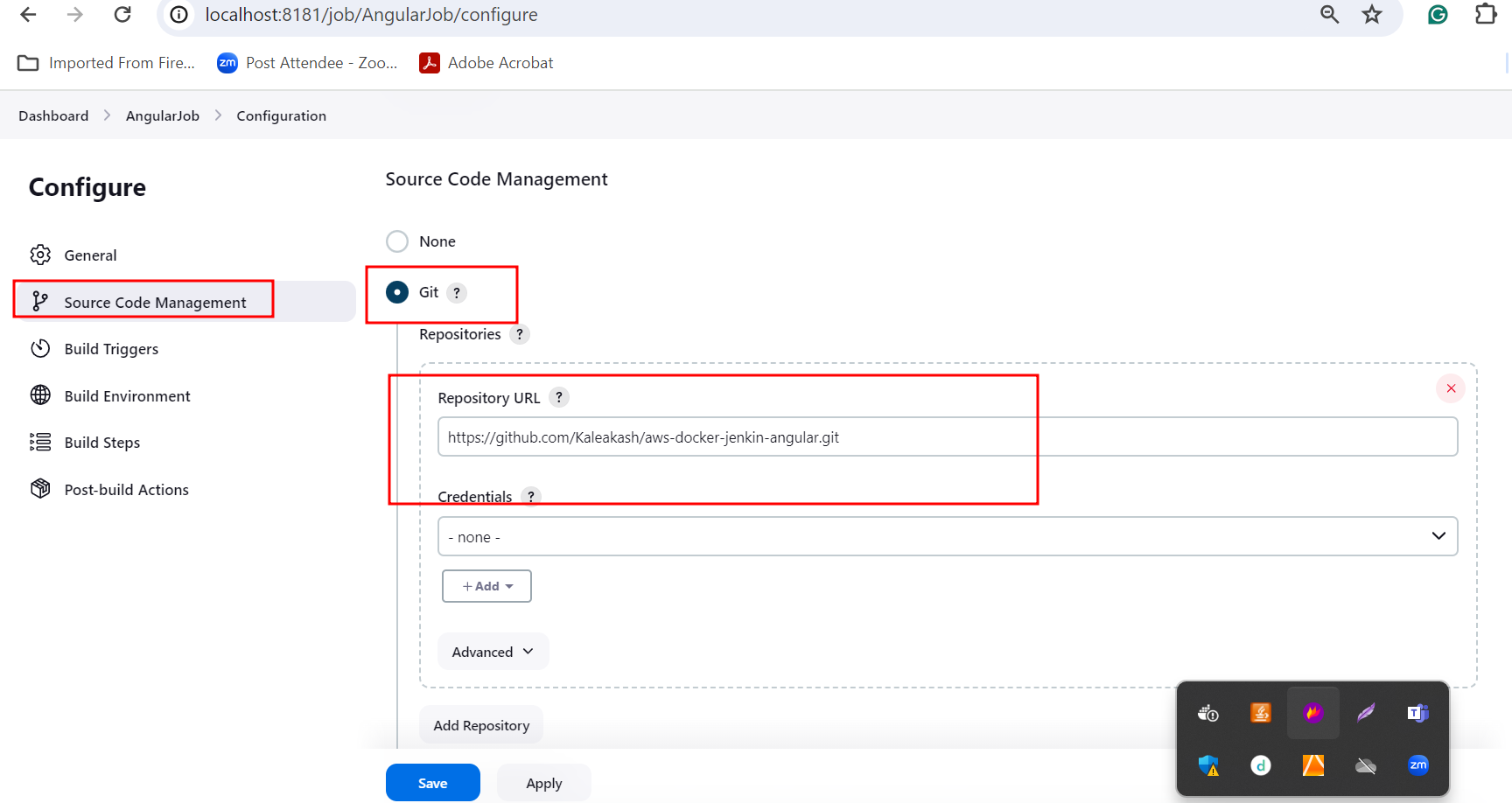


<https://github.com/Kaleakash/aws-docker-jenkin-angular.git>

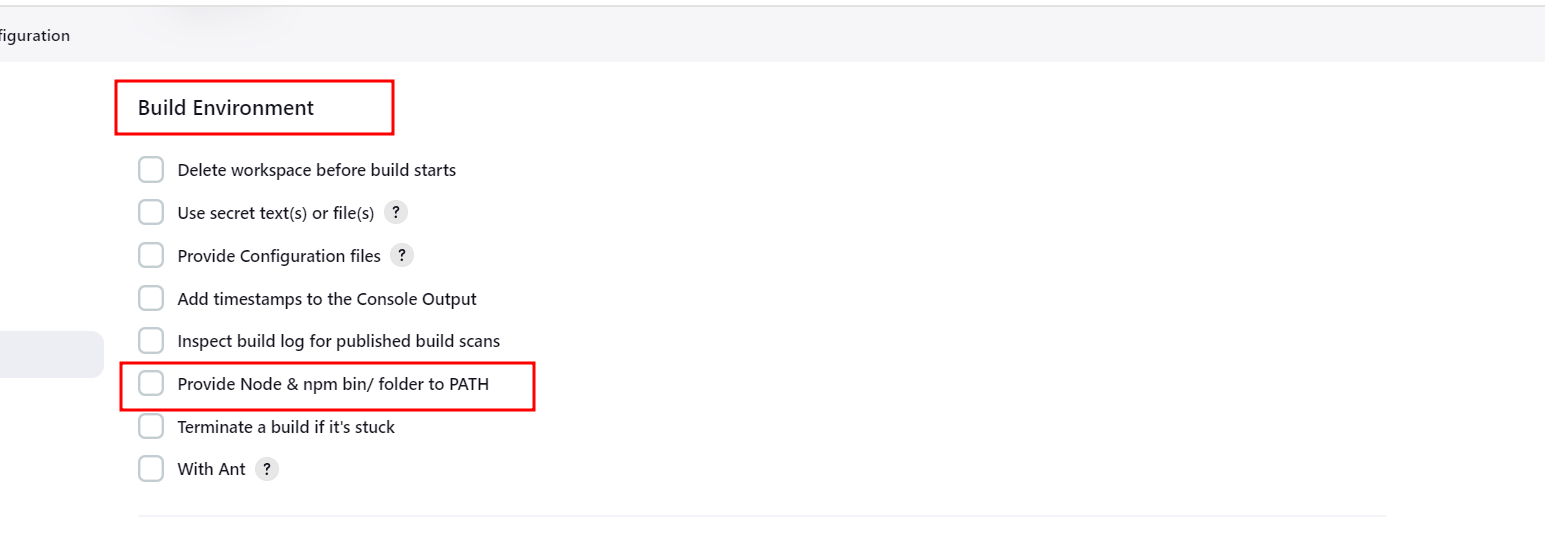
this github account contains angular project with branch as master

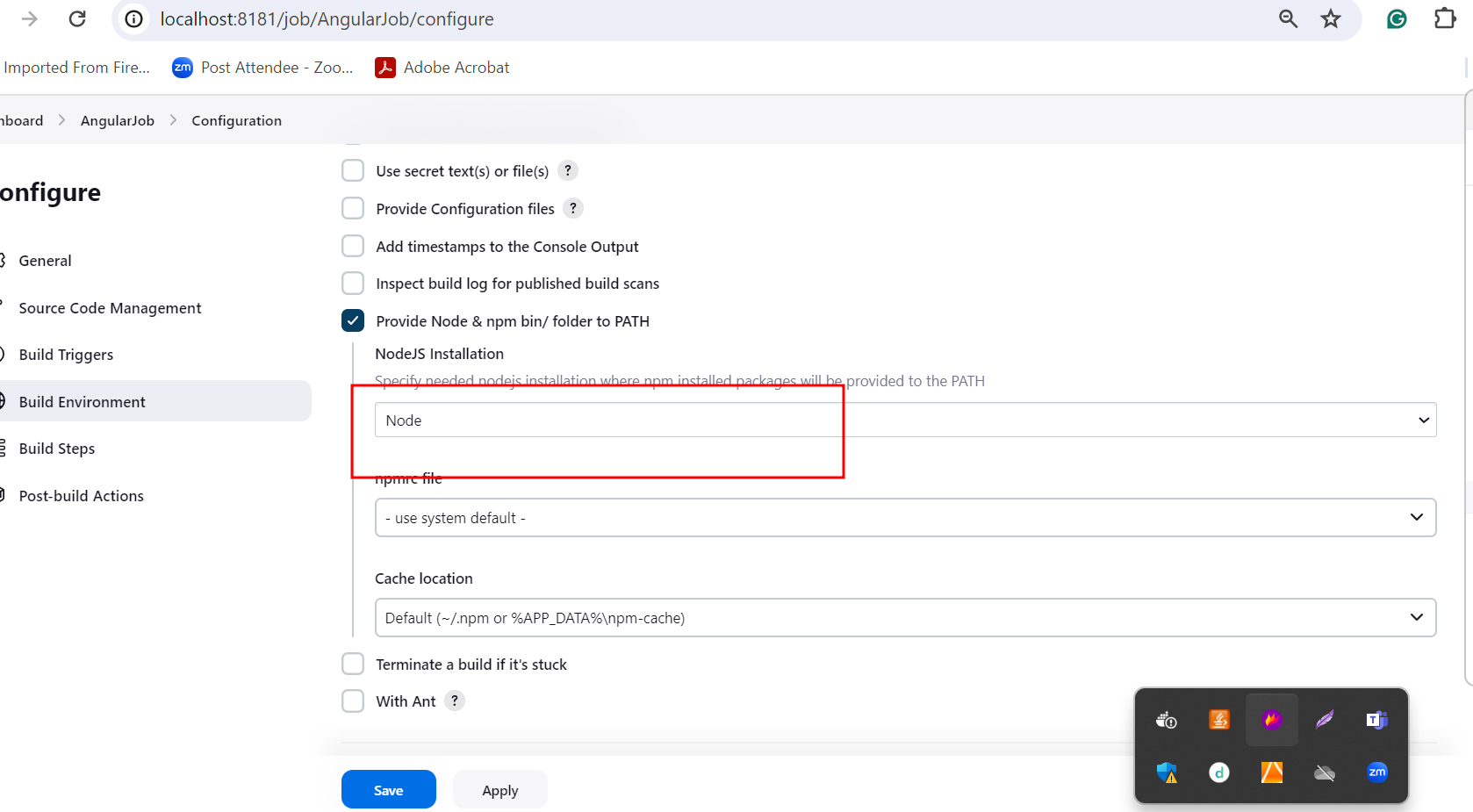


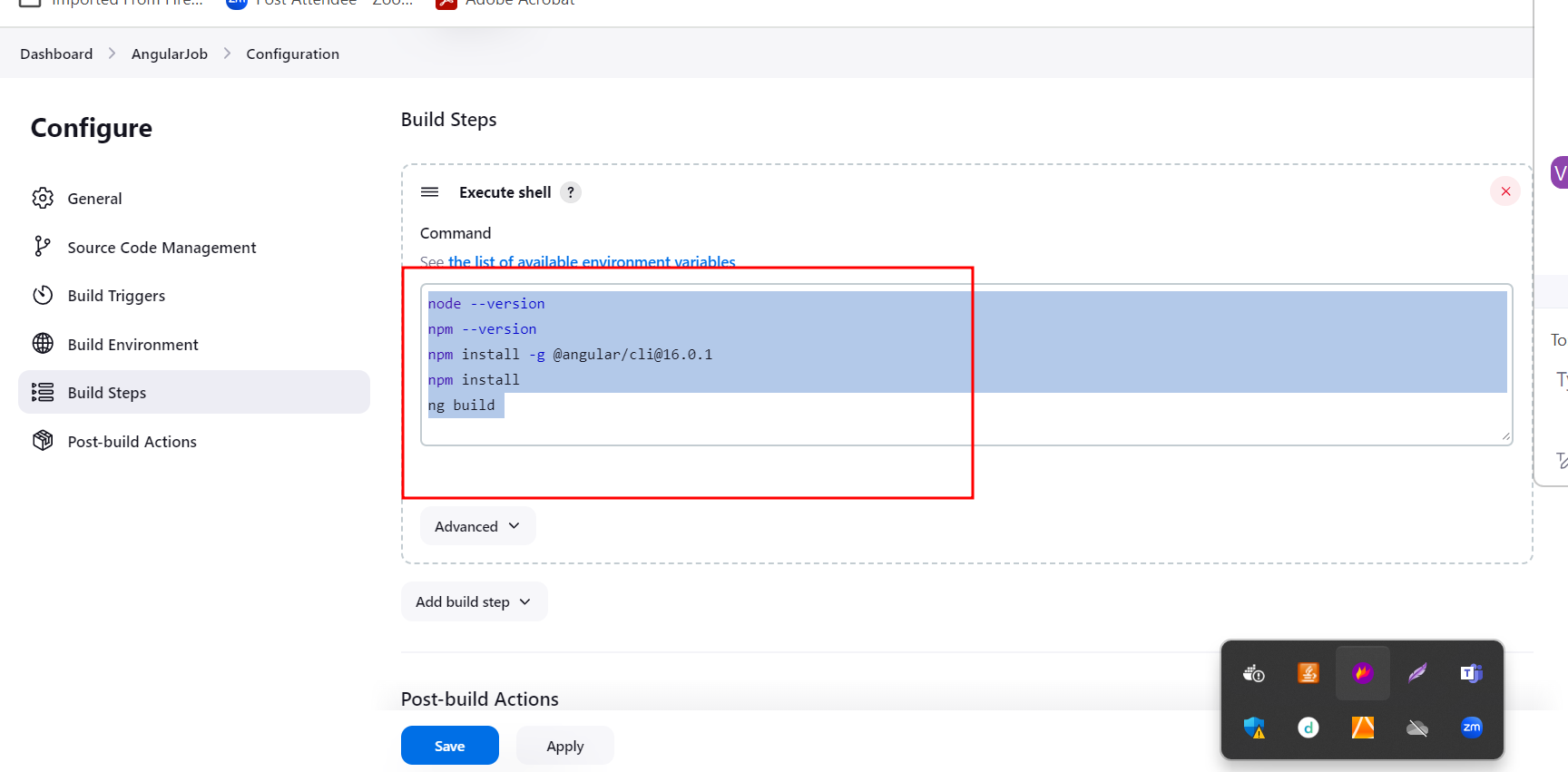


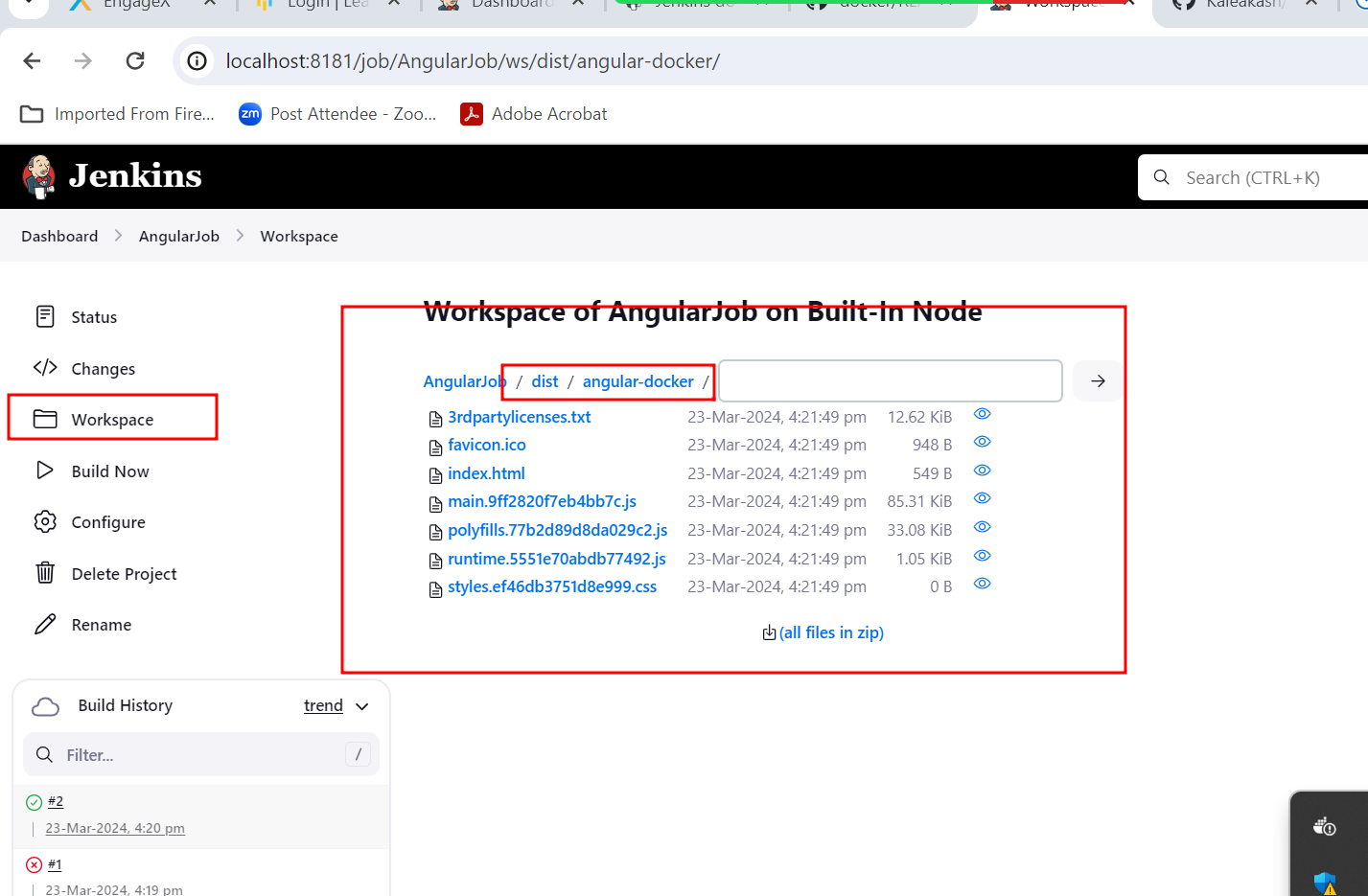


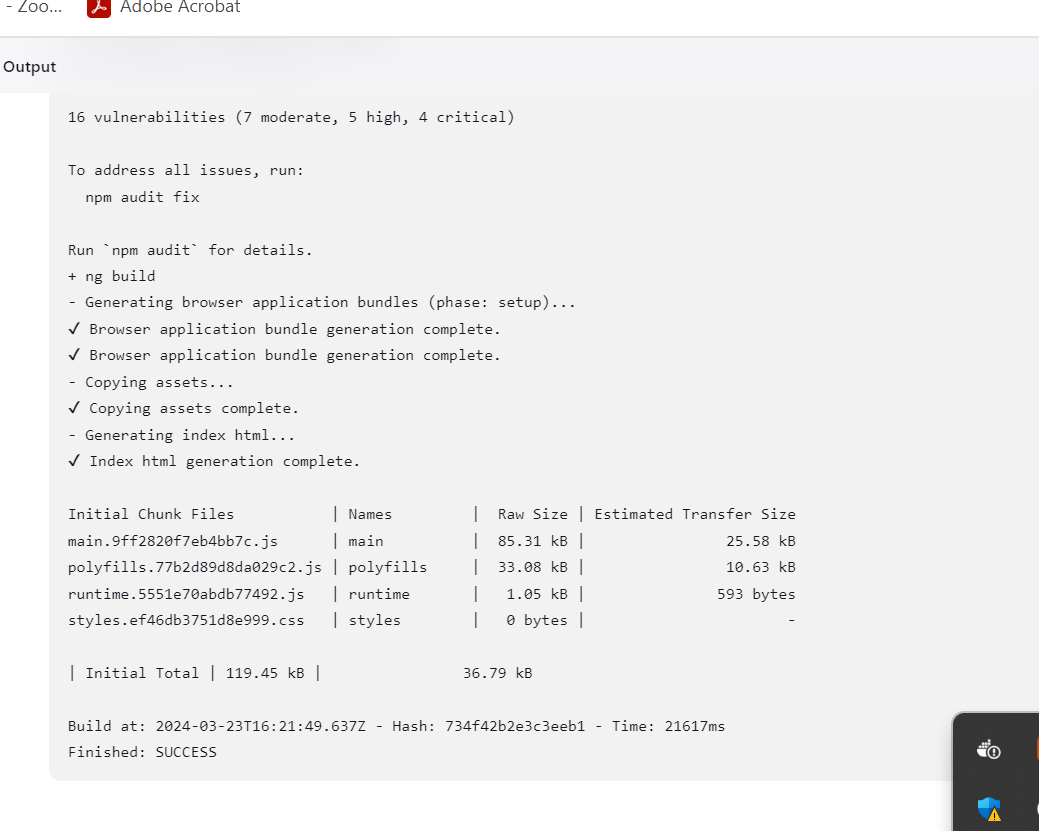












Create folder inside this folder create spring boot project or angular project.

Spring boot project : create simple rest api

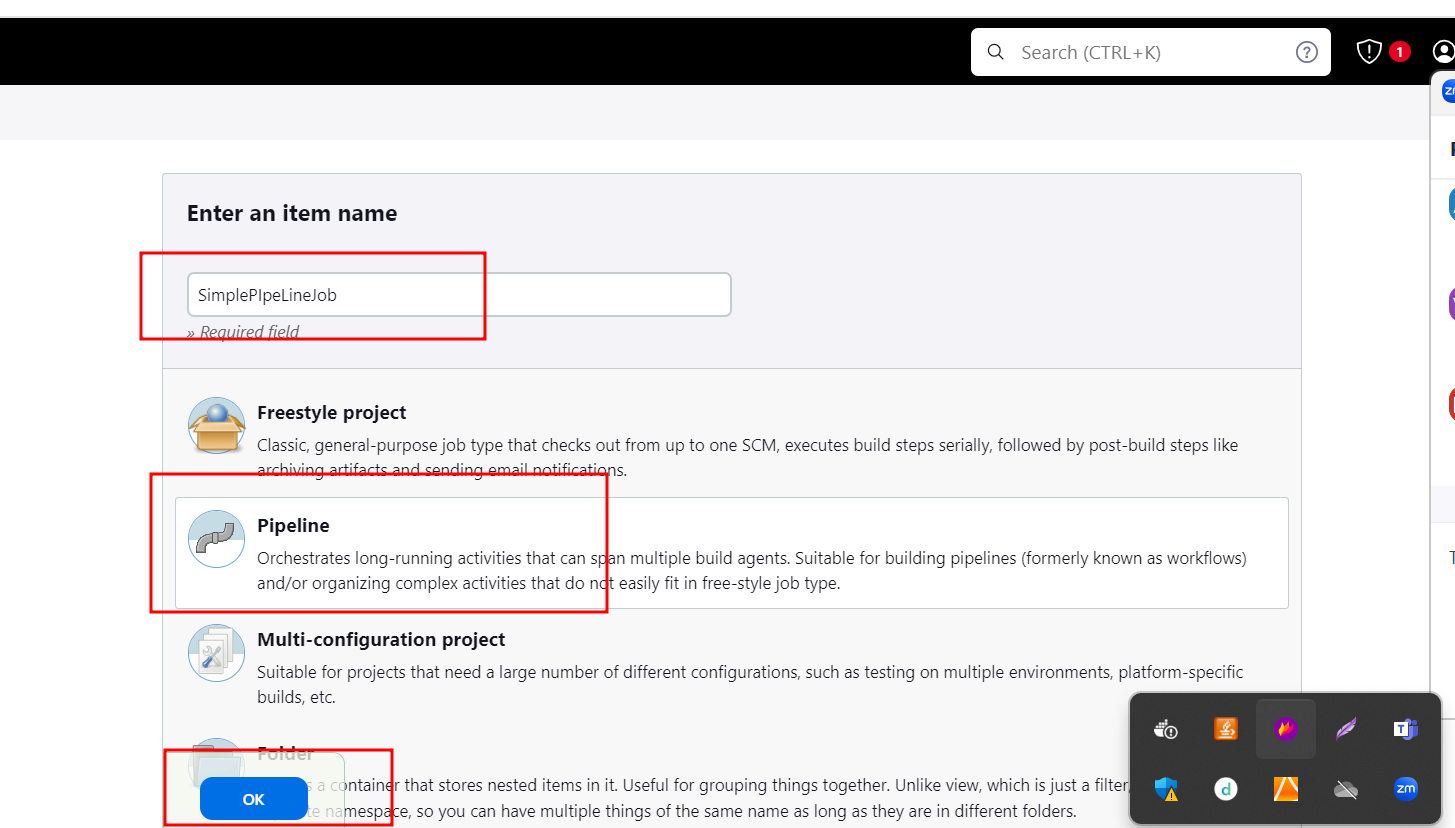
Angular project 🡪 simple message

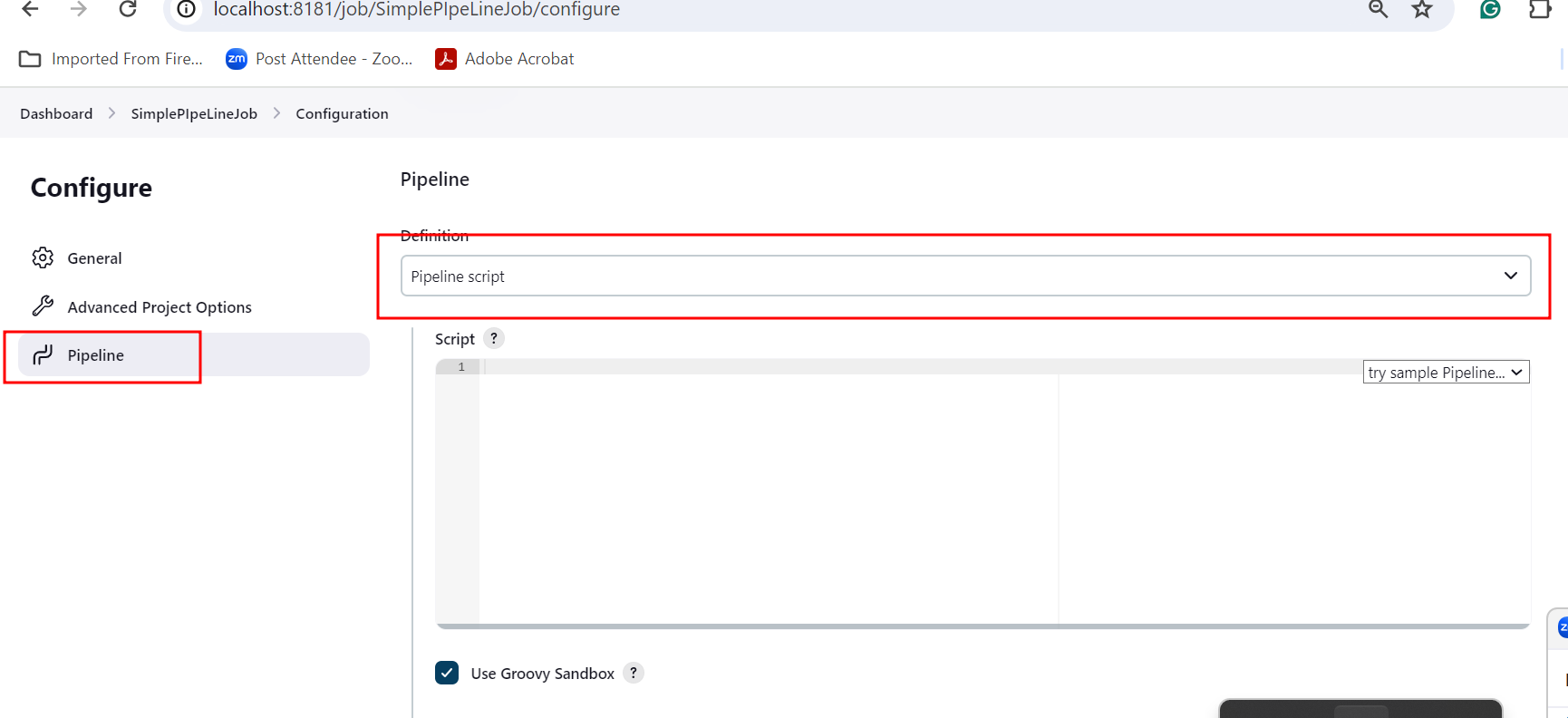
Create in your remote repository and push your local spring boot and angular project.

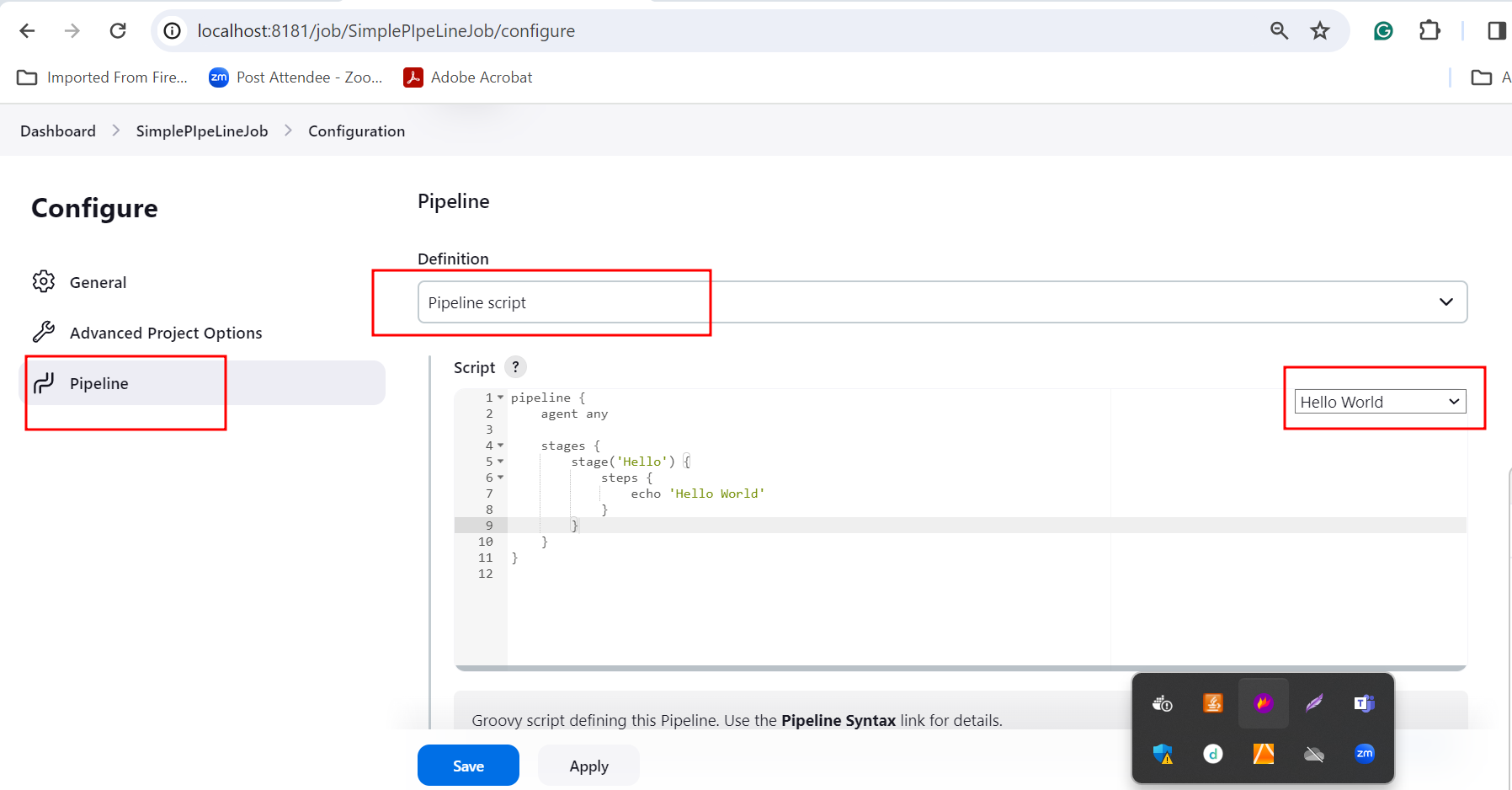
And push this project in new remote repository and create the Jenkin job which responsible to pull the project form git hub account use trigger option and build project.

24 – Mar – 2024 Day 6

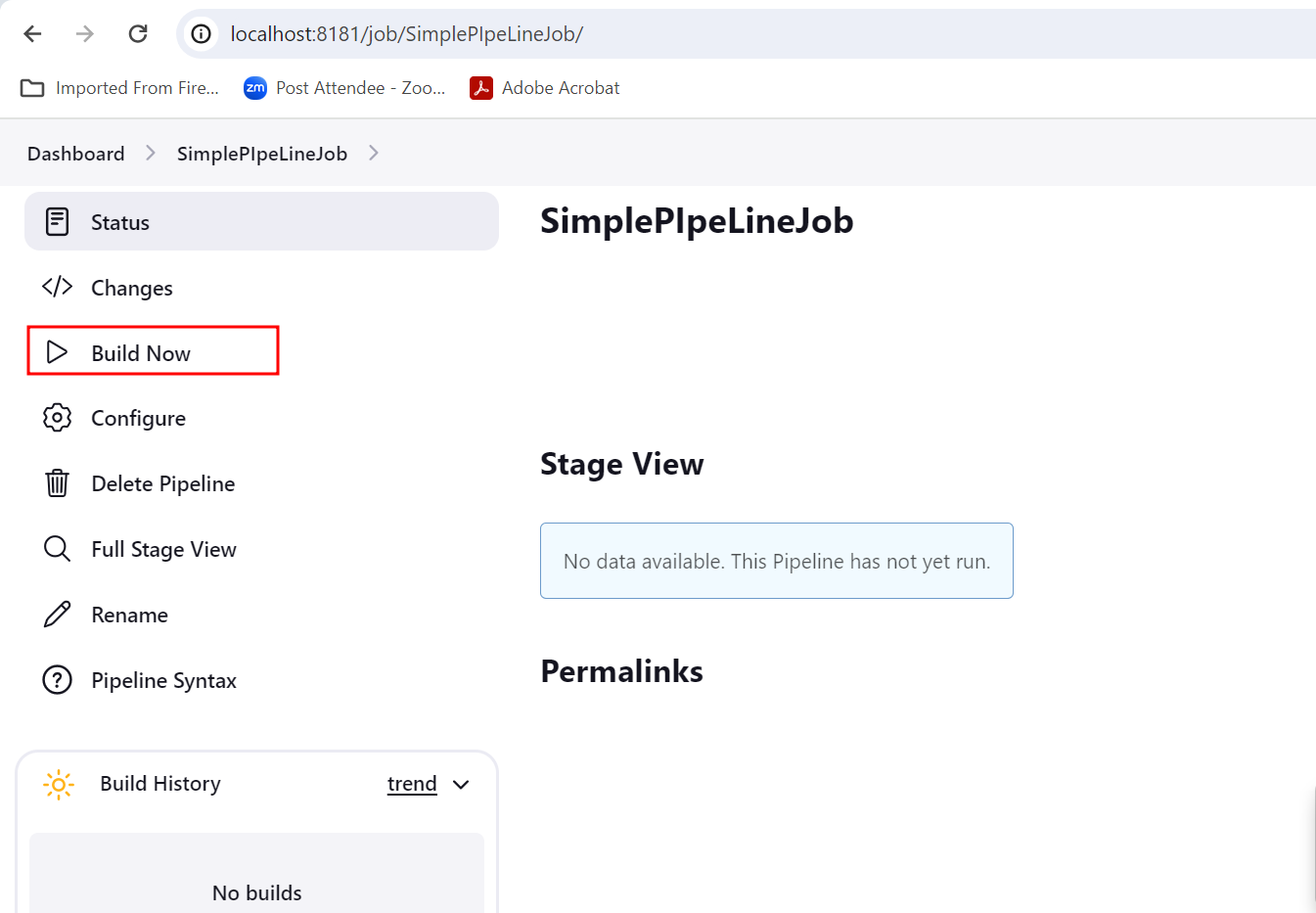
Jenkin Pipe line : It a collection of events or jobs which are interacted to each other to perform specific task.

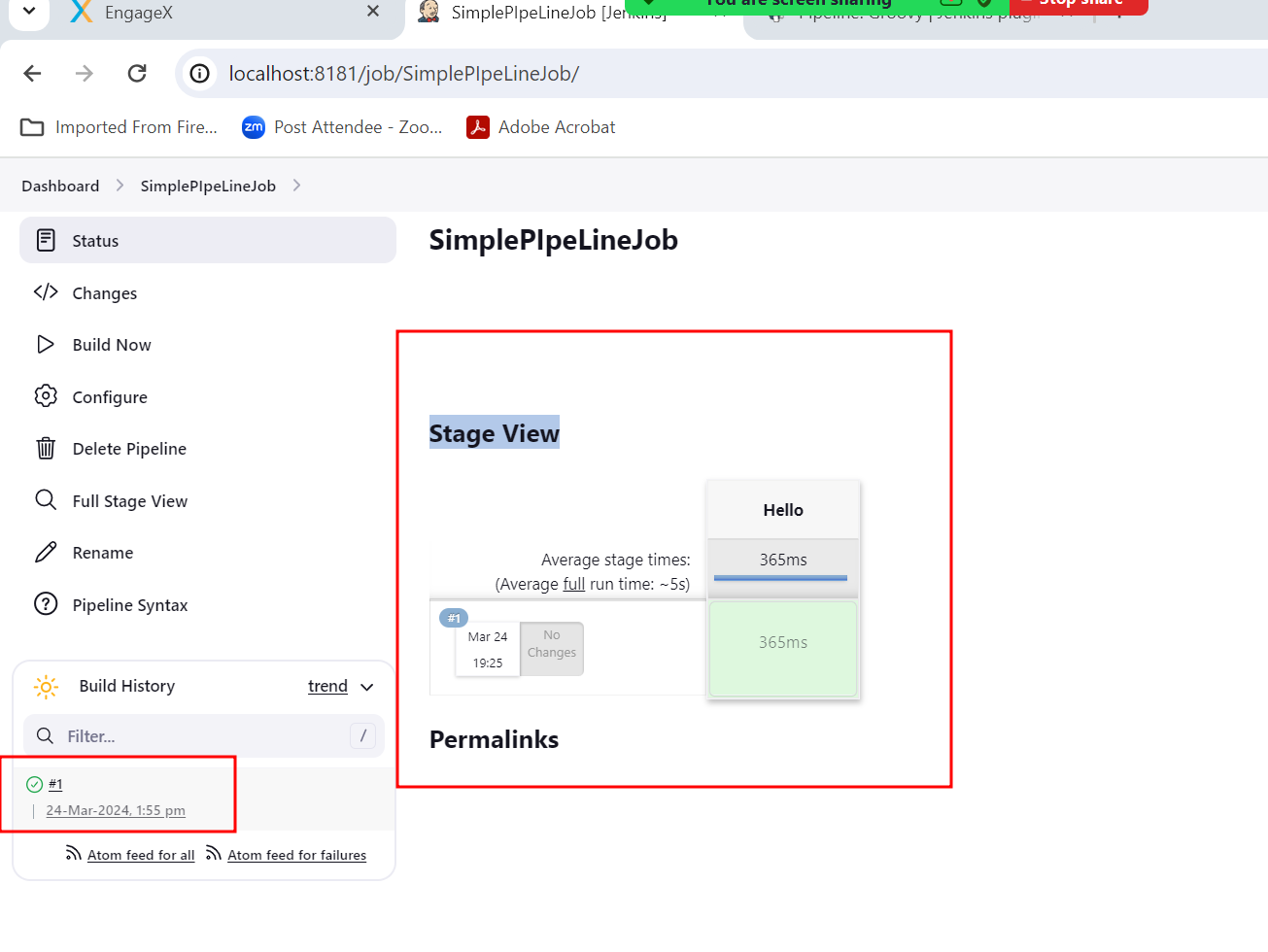






Apply and save

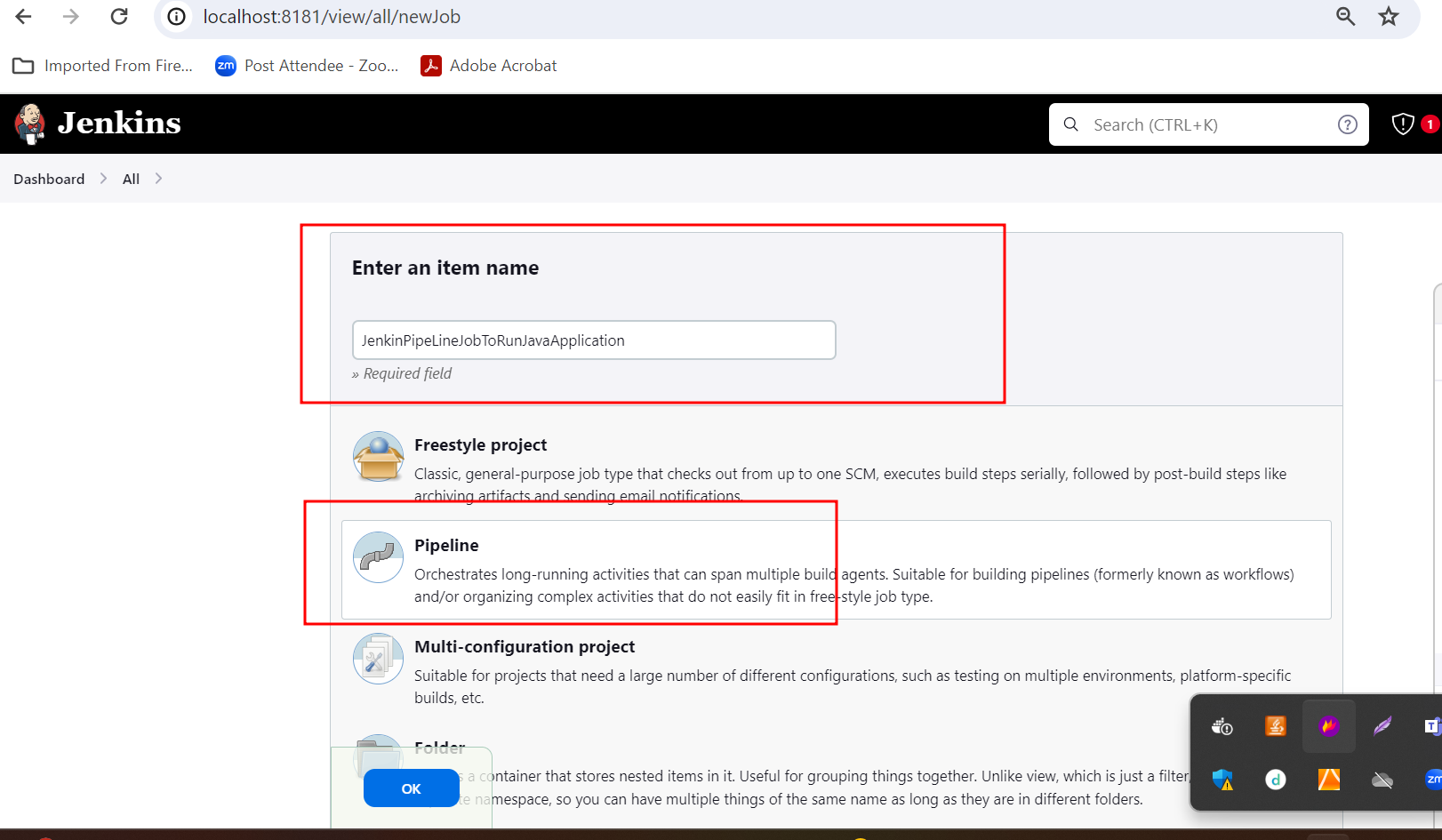


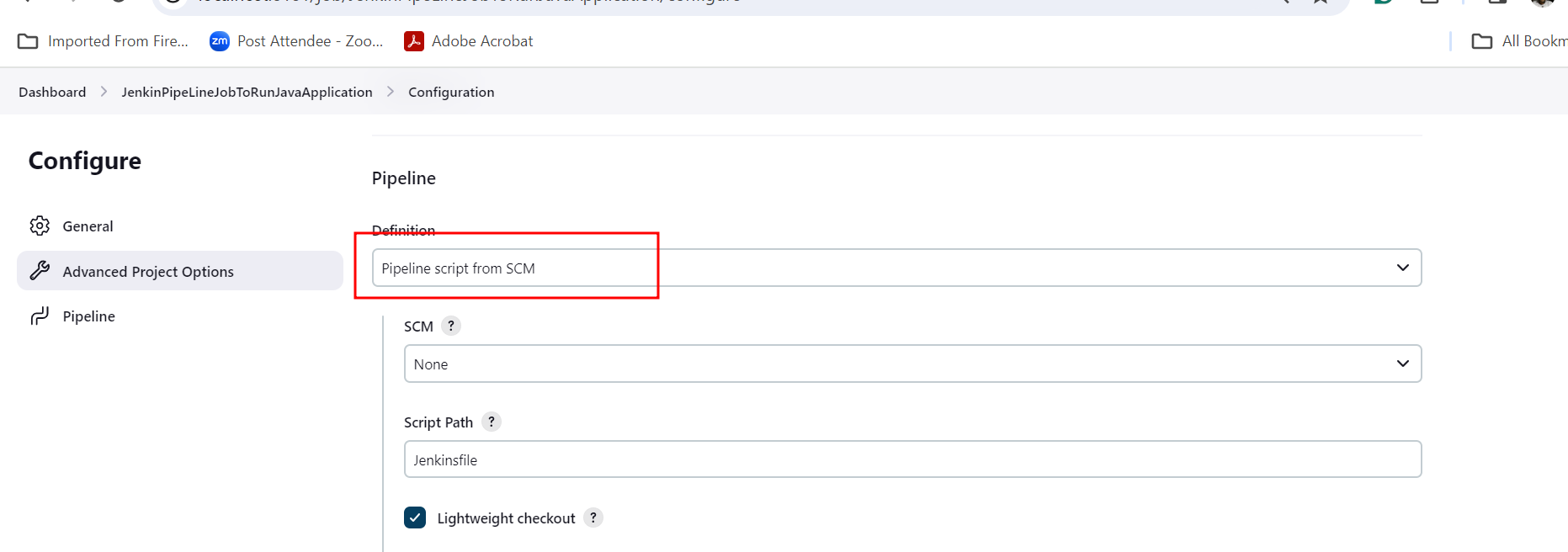


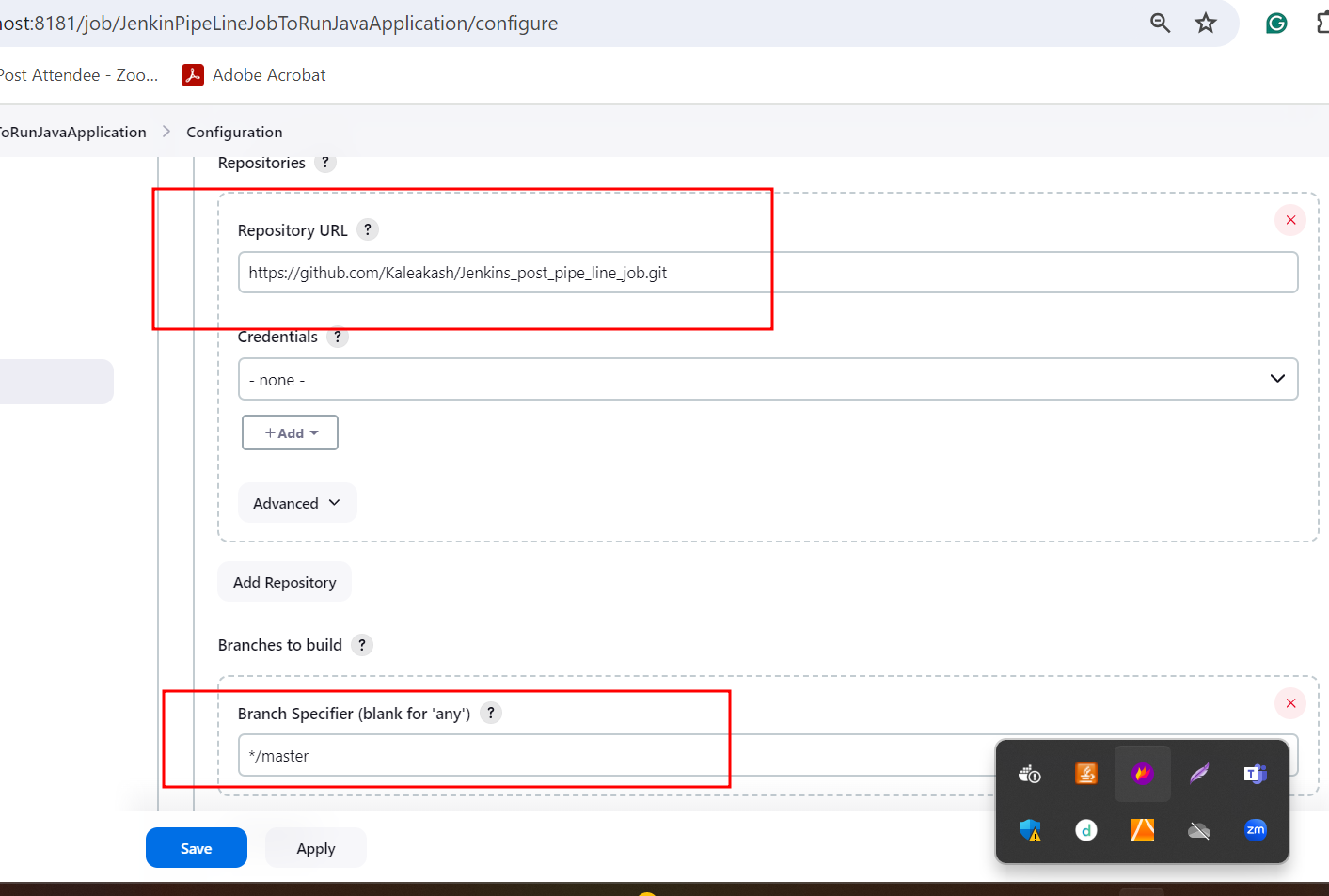
Creating pipeline job to build the maven project

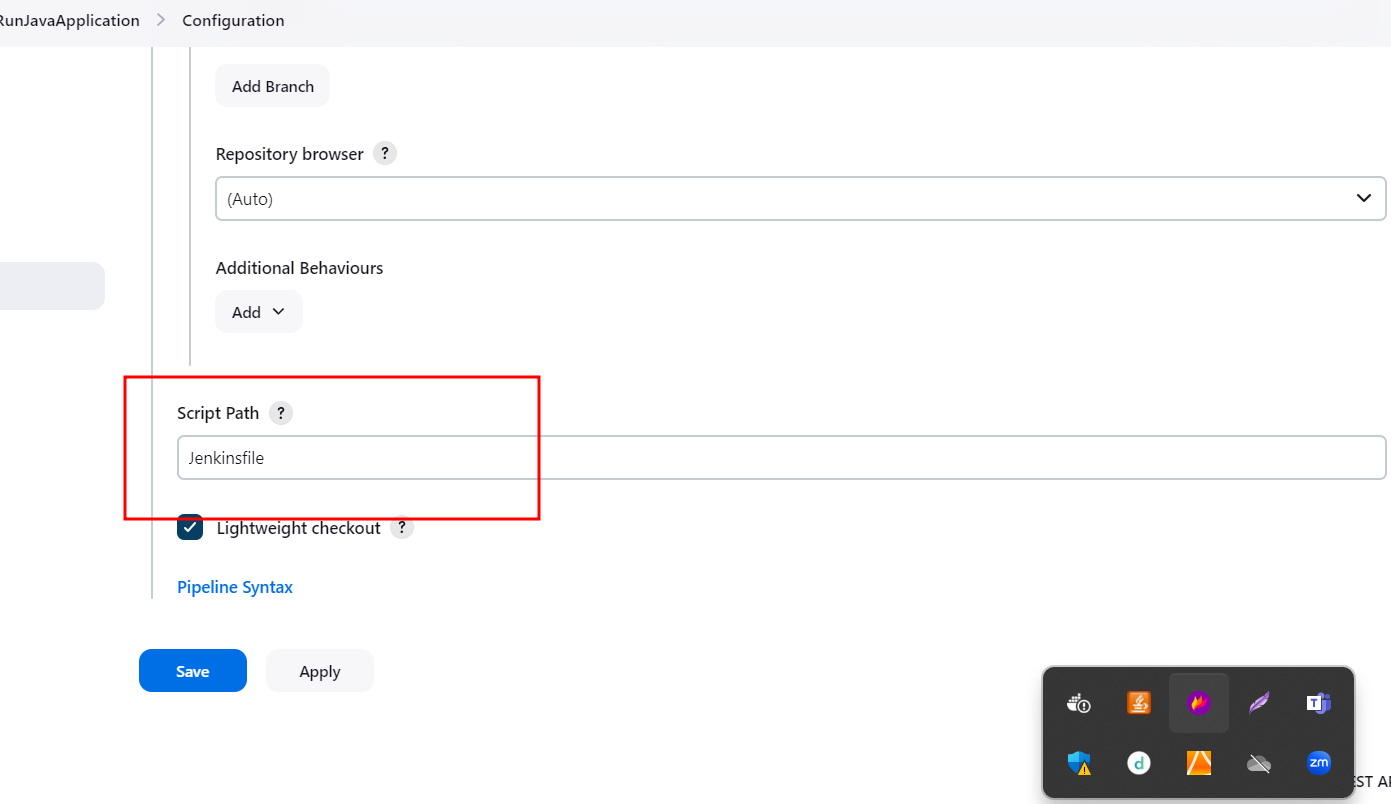
Creating Jenkin pipe line job and pull the project from our own git account with script file part of SCM(remote repository).

<https://github.com/Kaleakash/Jenkins_post_pipe_line_job>



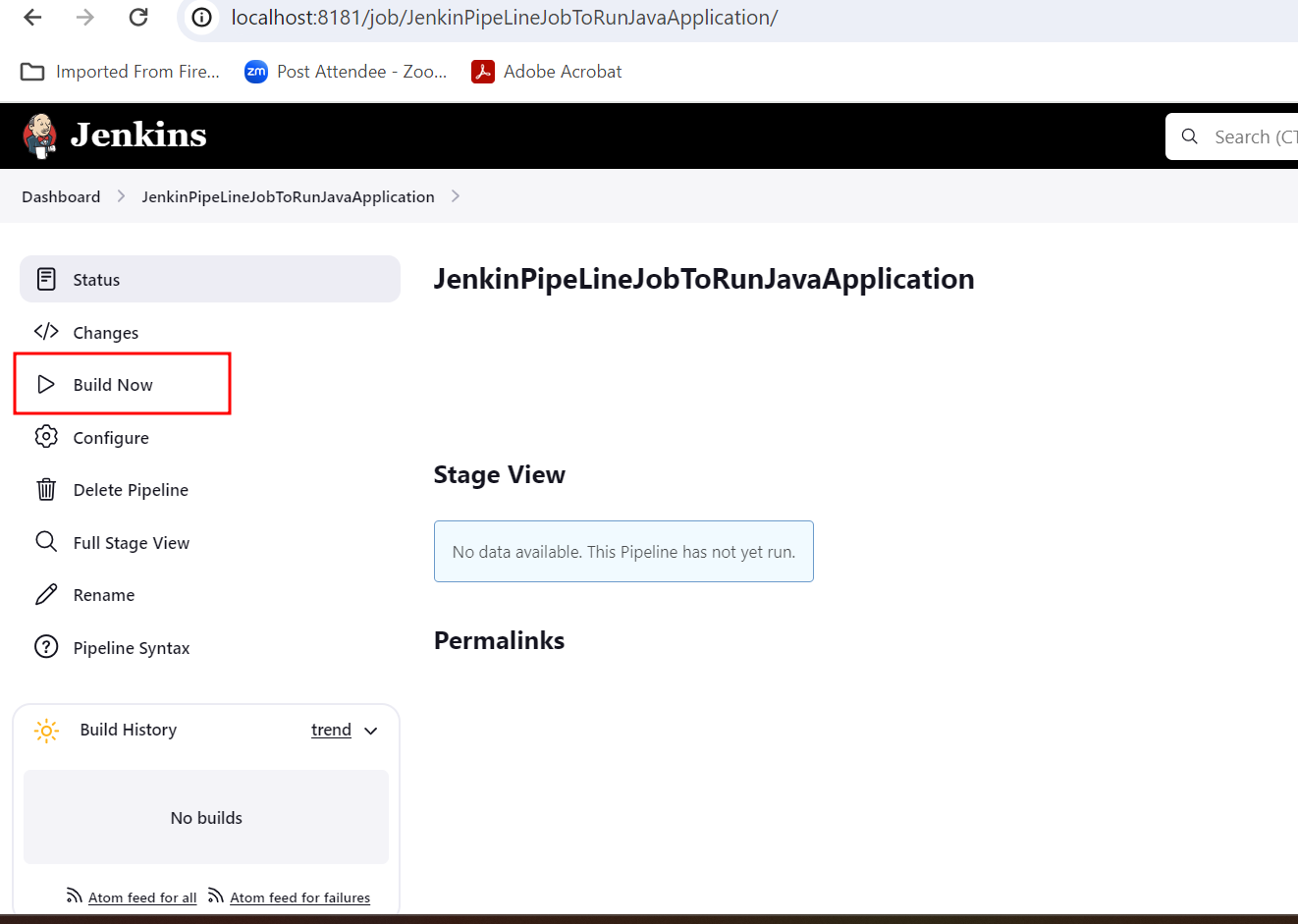




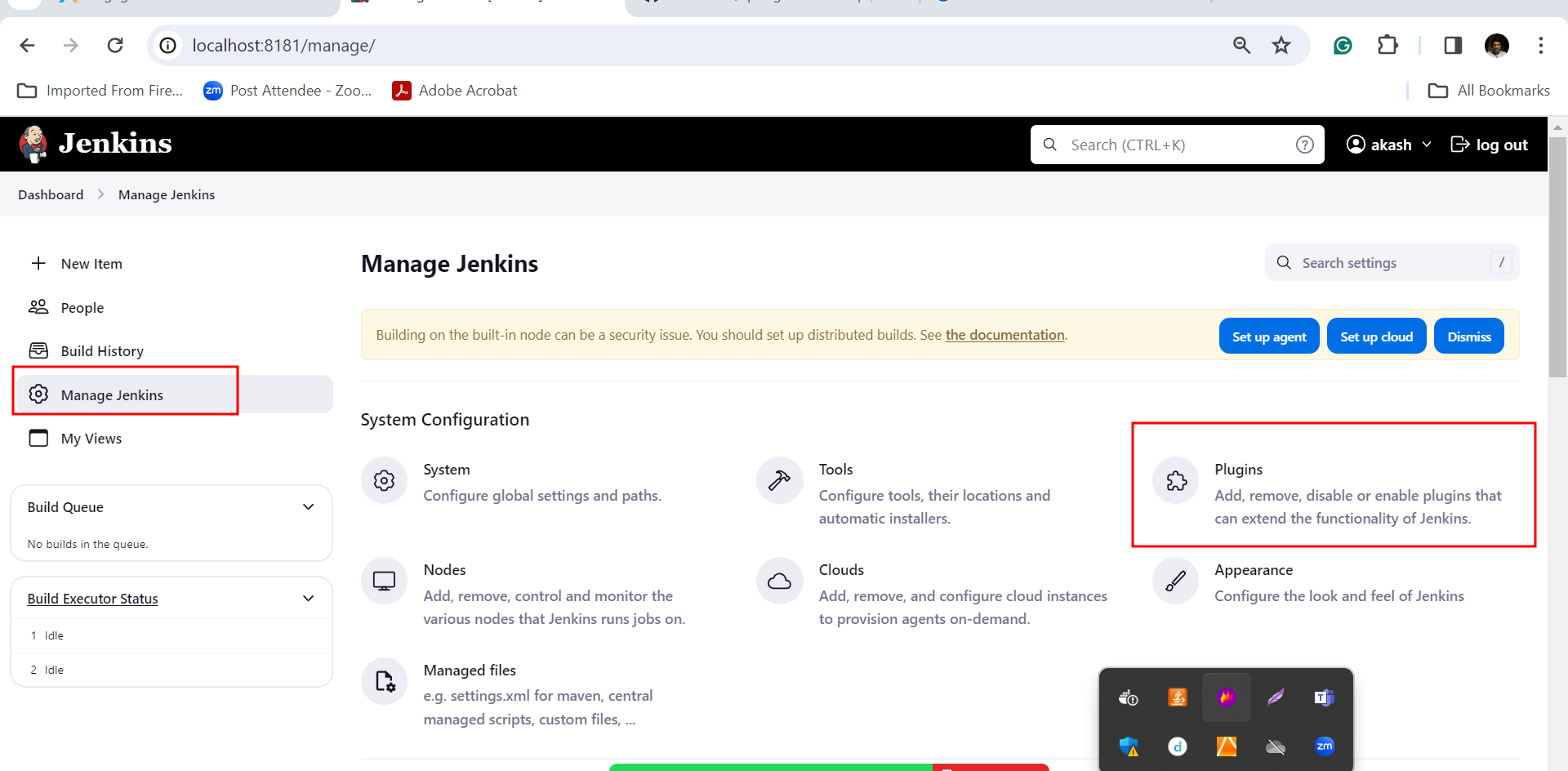


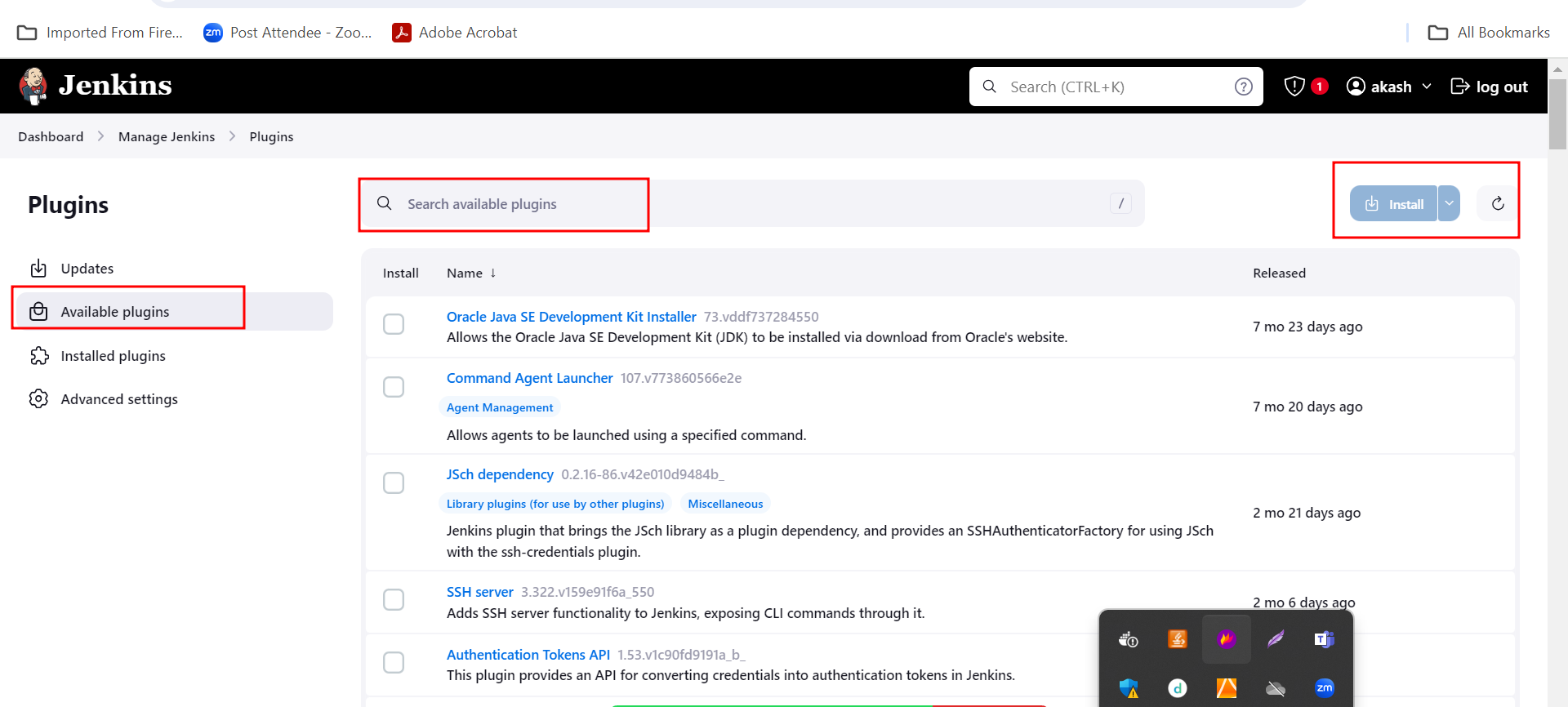
Apply and Save

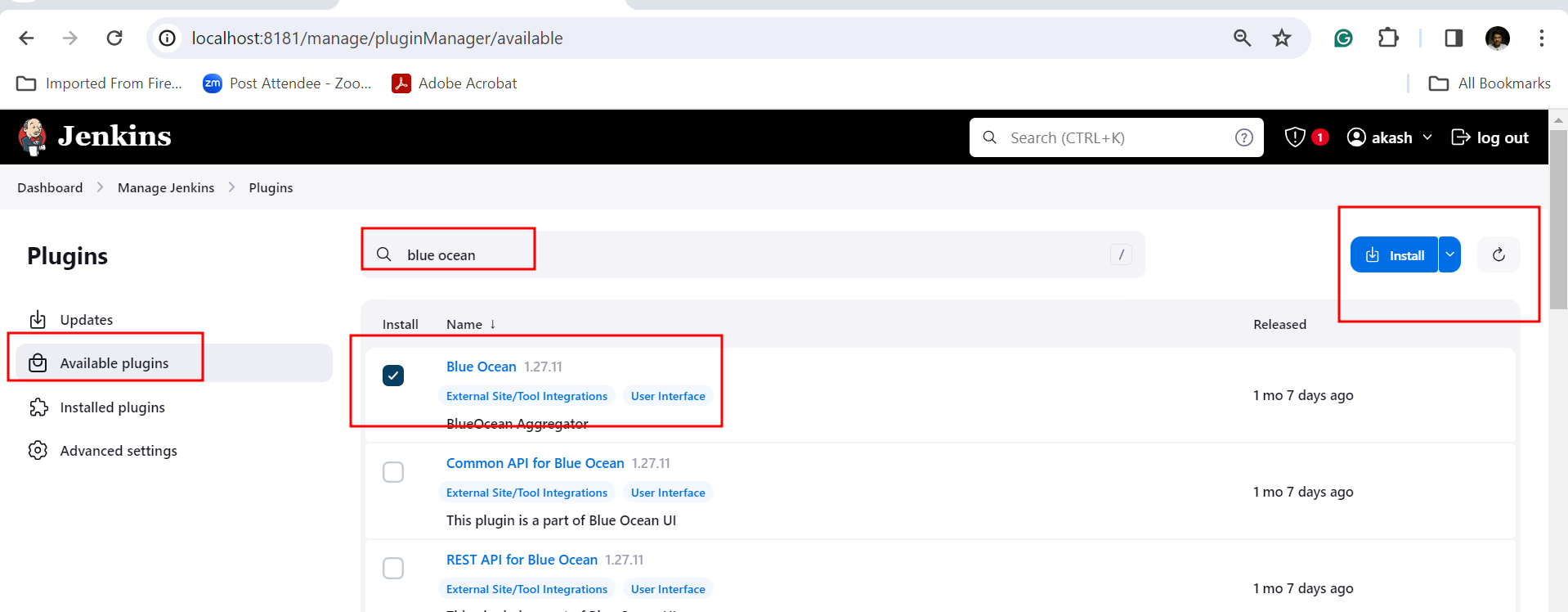
Then build the projects

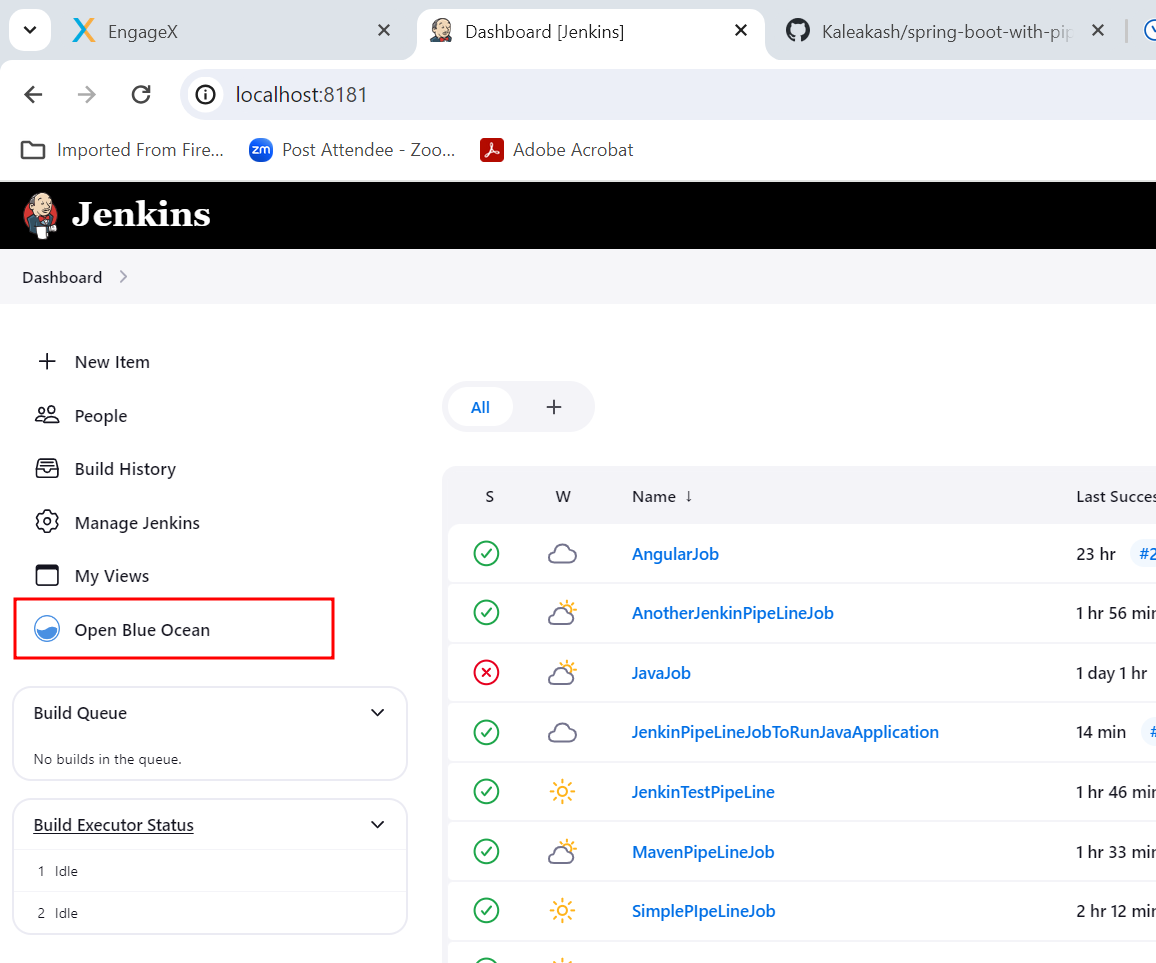


Blue Ocean plugin









1. Crete spring boot project with web starter with simple rest api running on port number 9090
2. Test this application in local machine with url as

<http://localhost:9090/>

Welcome to Dr.Shawn runs a pet clinic. He needs to record the visits and other details

1. Inside this create Dockerfile. This file responsible to create the image.
2. Jenkinsfile. This file is responsible to create the pipeline and build the project.
3. This project you need to push in your Git hub account.

After AWS

EC2 instance : which help to create virtual server machine.

If we run docker as well as Jenkin in AWS EC2 instance that instance provide us public as well as private IP address. So if run our spring boot application or angular application or jenkin job or jenking pipe with docker we can access that application using IPAddress ie.