LAB RECORD 2022-23.

D.BANUMAIAH’S POLYTECHNIC.

COMPUTER SCIENCE & ENGGNERING DEPT.

|  |  |
| --- | --- |
| ***Sl. No*** | ***Lab Manual Rubrics*** |
| **01.** | Create and manage product backlog using Jira. |
| **02.** | Create sprint1 with required user stories. |
| **03.** | Create a wireframe for user stories. |
| **04.** | Write Test case for user stories. |
| **05.** | Create a repository in GitHub and cloning the repository using VS code. |
| **06.** | Perform the push and commit operations of project in GitHub using VS code. |
| **07.** | Launch a ubuntu server using AWS & install Jenkins software. |
| **08.** | Create the cloud account in amazon Aws. |
| **09.** | Create a student registration form using HTML & CSS. |
| **10.** | Create build pipeline of the new project using Jenkins. |
| **11.** | Demonstrate installation of Jenkins & build a hello world program using AWS. |
| **12.** | Create a login page with Java script & its function. |
| **13.** | Demonstrate installation of Apache2 in ubuntu server & perform copying of file from local machine to Server. |
| **14.** | Create a simple hello world application using typescript. |
| **15.** | Create a new React app & perform import & export operations. |
| **16.** | Demonstrate the following operations in docker concepts   1. Creating a docker file 2. Build a docker image with docker file. |

**Signature of Co-Ordinator**

**1. Create and manage product backlog using Jira.**

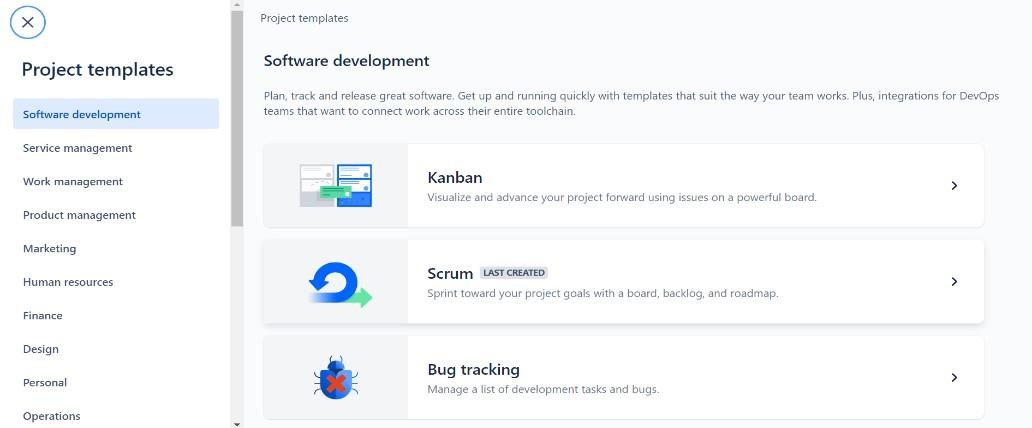
# 1.1 Steps to create project in Jira

Step 1 : Login into Atlassian Jira account.

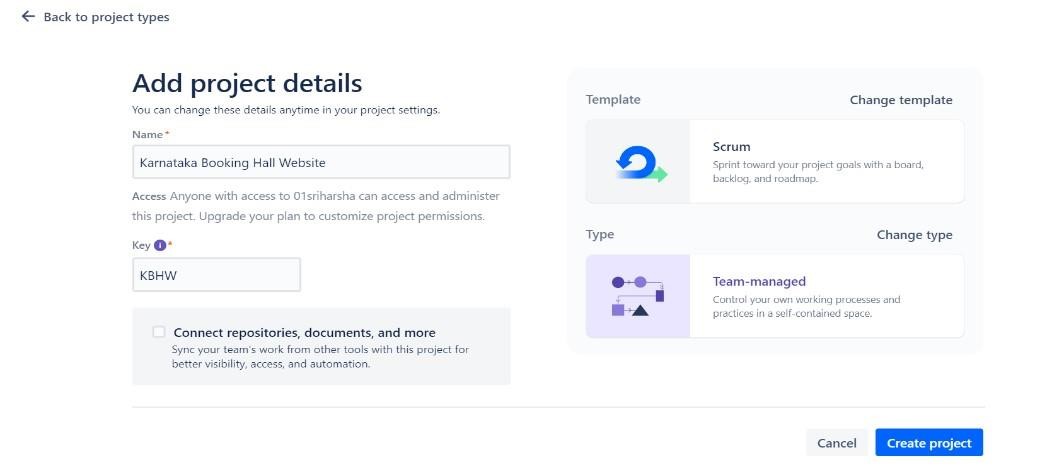
Step 2 : On the Jira software dashboard , Click on create project.



Step 3 : On next page , Select scrum project and then followed by team managed project.



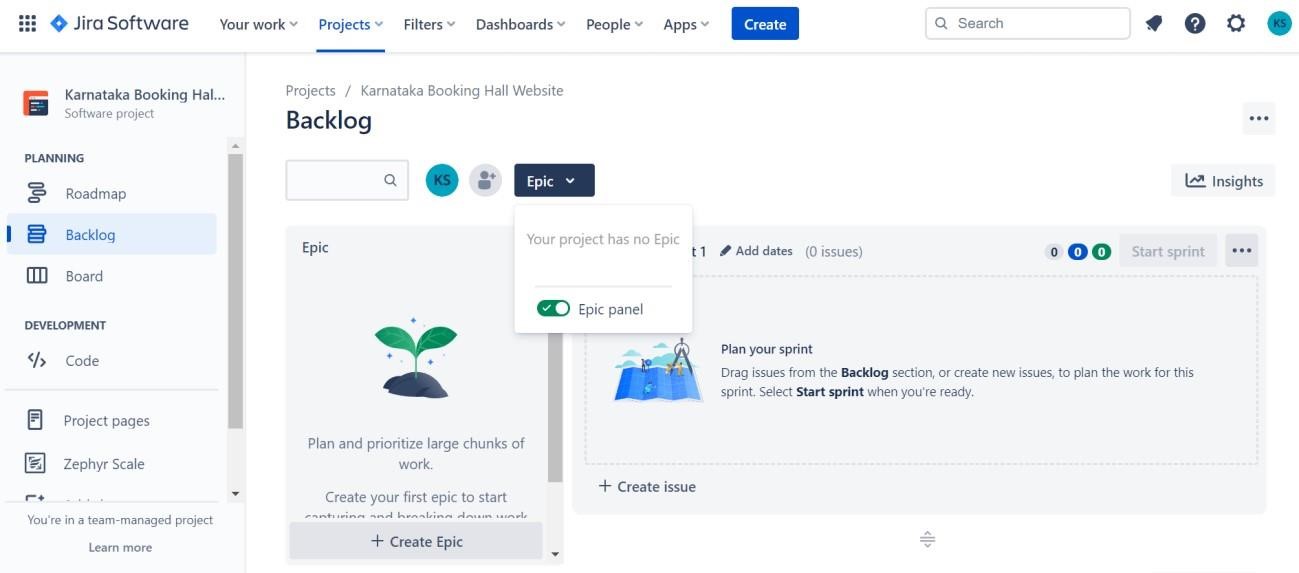
Step 4 : Enter project name and click on create project.



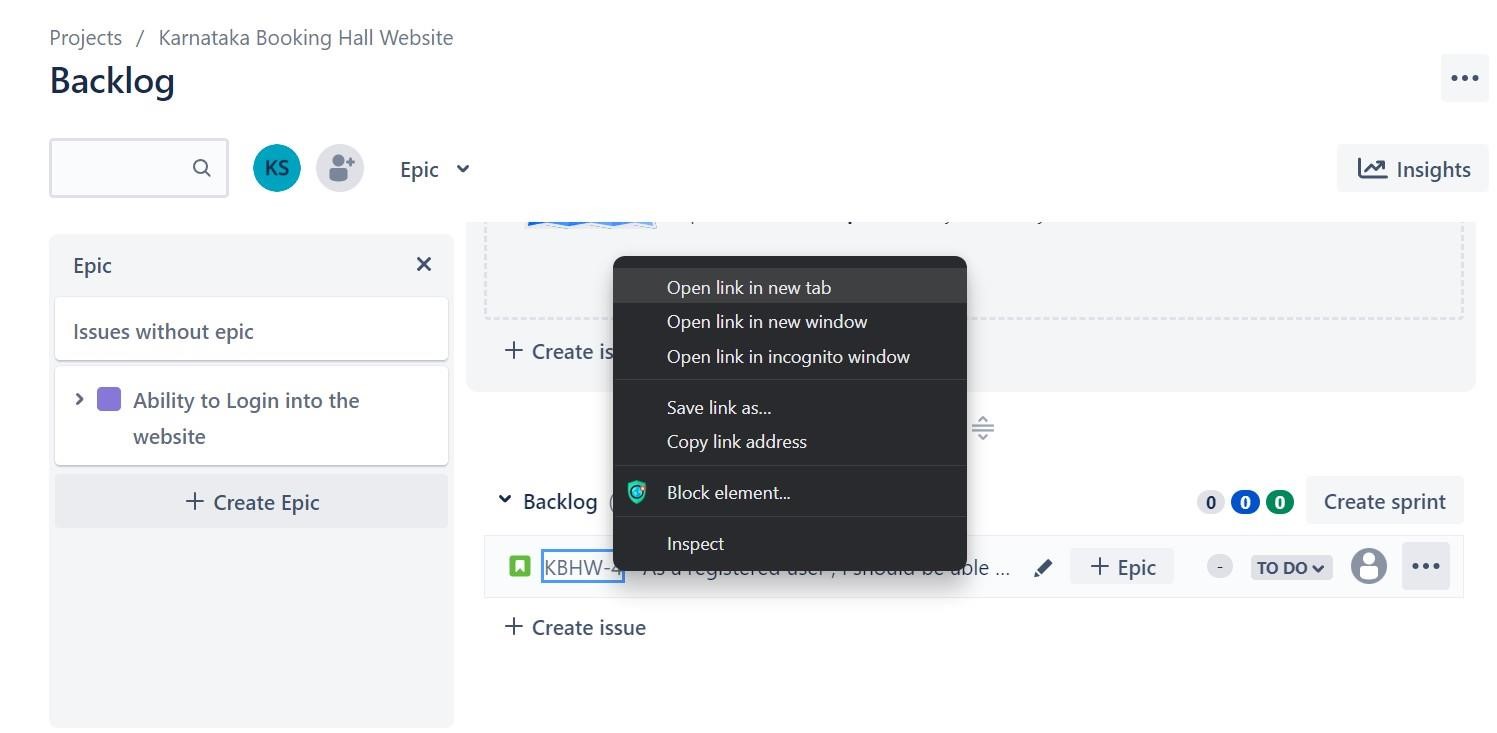
# 1.3 Steps to Manage product backlog using Jira

Step 1 : On the dashboard , Select backlog tab.

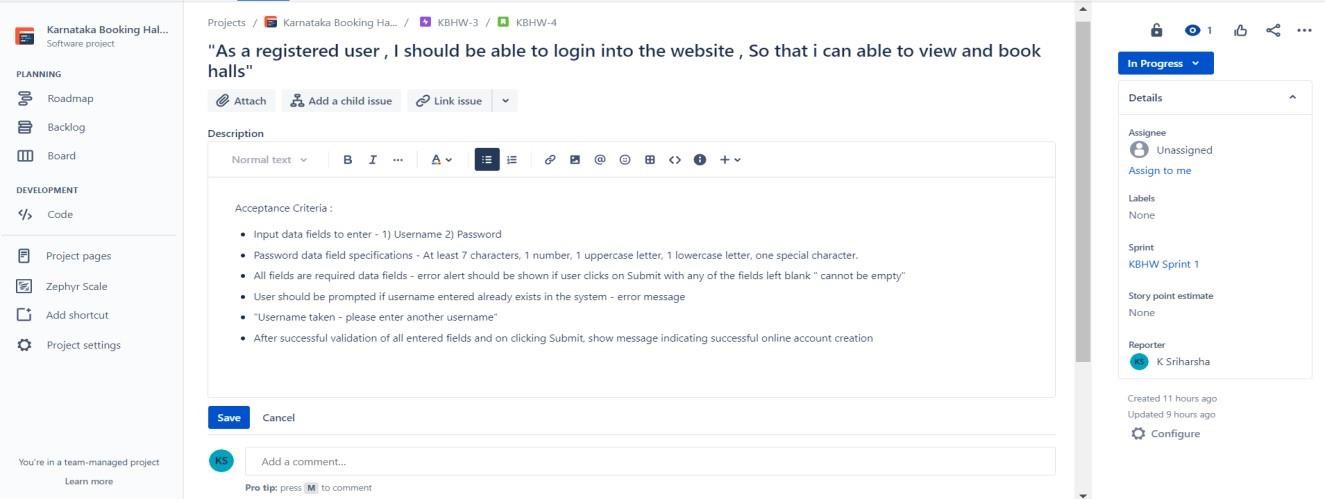
Step 2 : Select Epic option and toggle the epic switch to create a new Epic.



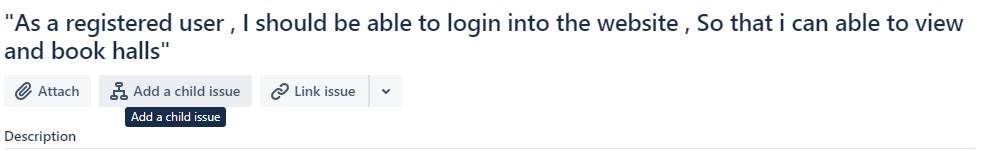
Step 3 : Add a new epic and then followed by user story by clicking on Create Issue option under backlog. Step 4 : After adding new issue right click on the issue id to open it in new tab.



Step 5 : Under the description tab , add acceptance criteria for the specific issue.



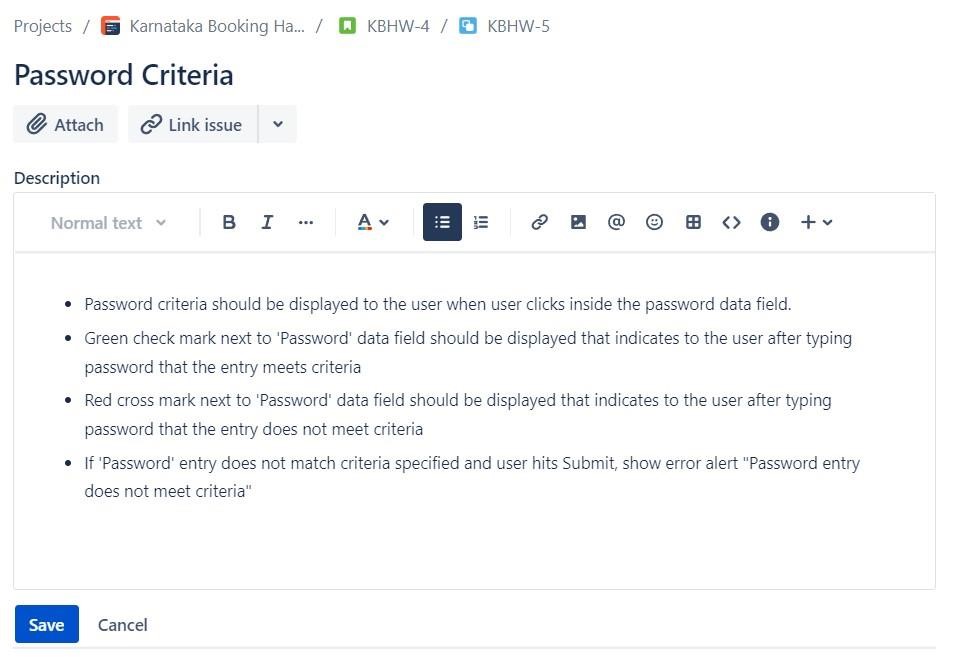
Step 6 : After adding acceptance criteria , click on Add child issue



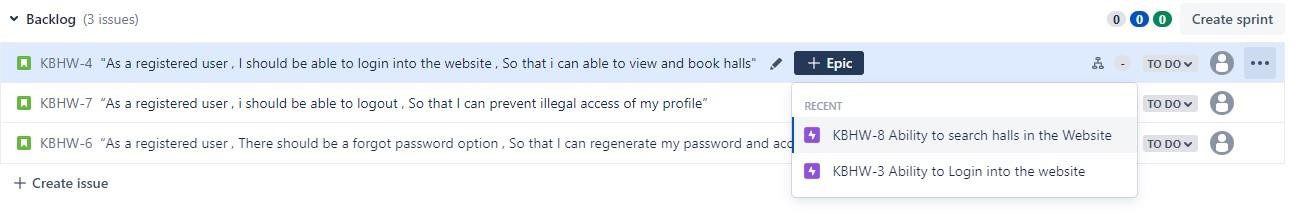
Step 7 : Enter the name for the child issue and Click on it.



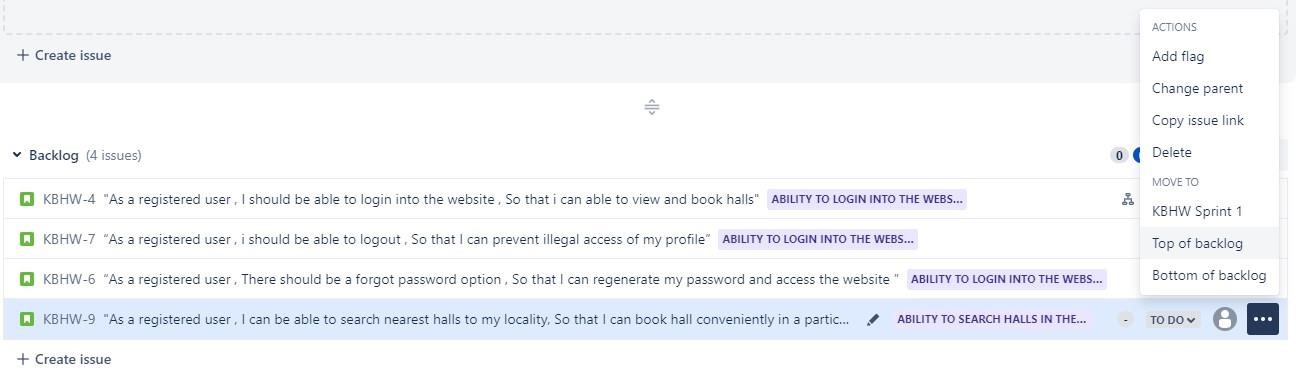
Step 8 : Enter the details to be done inside the description tab and click on save.



Step 9 : After creating the issue , head back to backlog dashboard and add Epic for the particular issue.



Step 10 : Now prioritize the issue according to the requirements either by dragging it to top to bottom or by selecting move option.



**2. Create Sprint 1 with required user stories in Jira.**

Step 1 : Login into Atlassian Jira account.

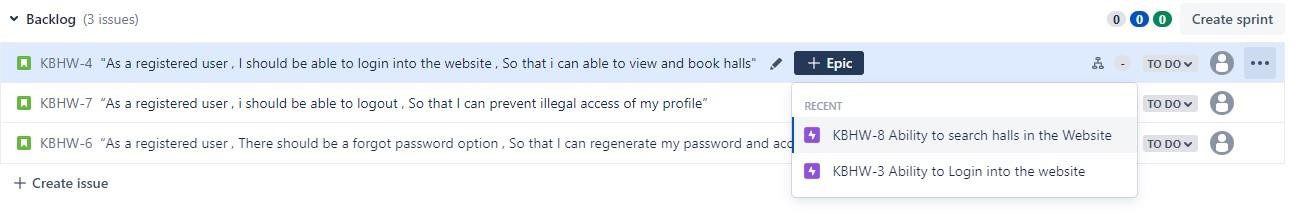
Step 2 : On the Jira software dashboard , Click on create project and create a new scrum project.

Step 3 : On the dashboard , Select backlog tab.

Step 4 : Select Epic option and toggle the epic switch to create a new Epic.

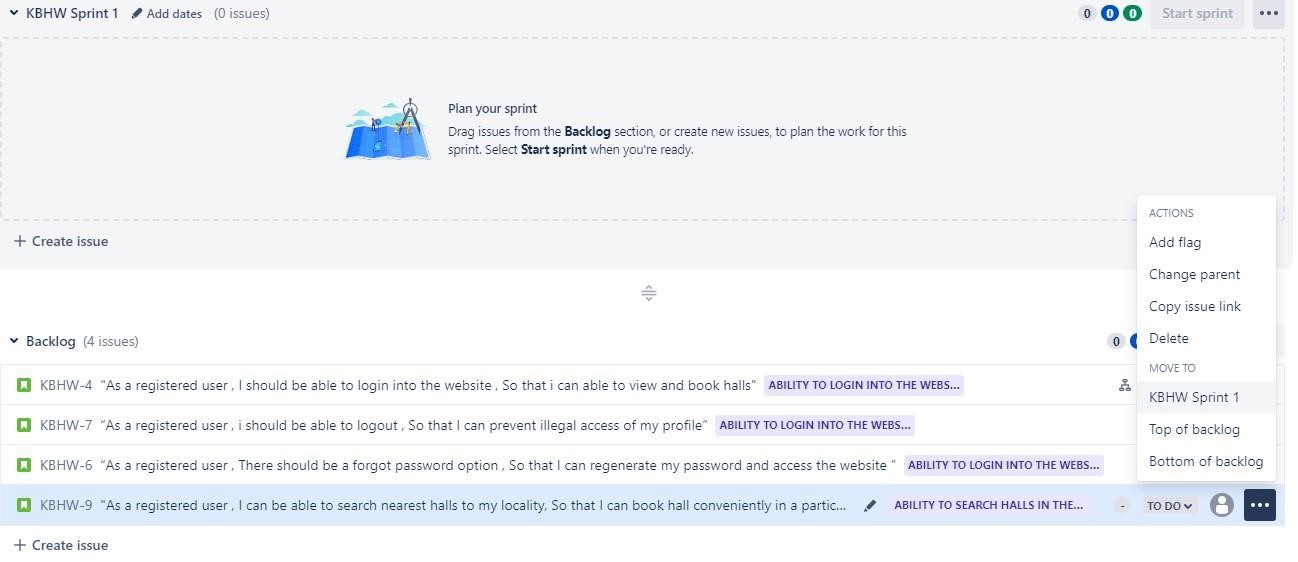
Step 5 : Add a new epic and then followed by user story and child issue by clicking on Create Issue option under backlog.

Step 6 : After creating the issue , add Epic for the particular issue.

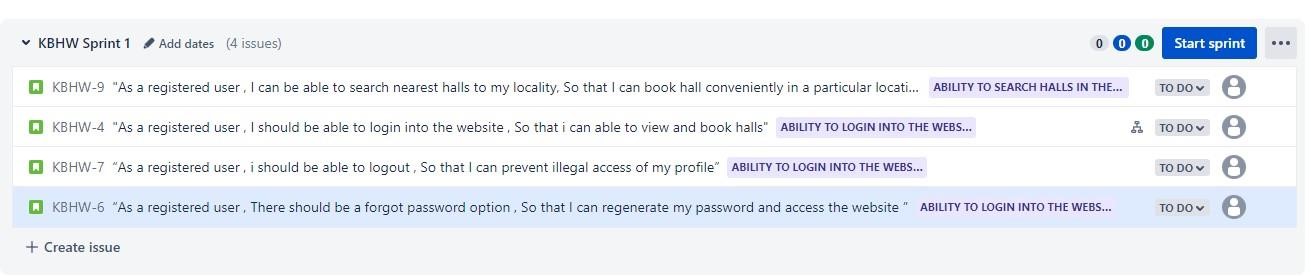


Step 7 : Now drag and drop the issues from backlog to Sprint tab , which is above the backlog tab.

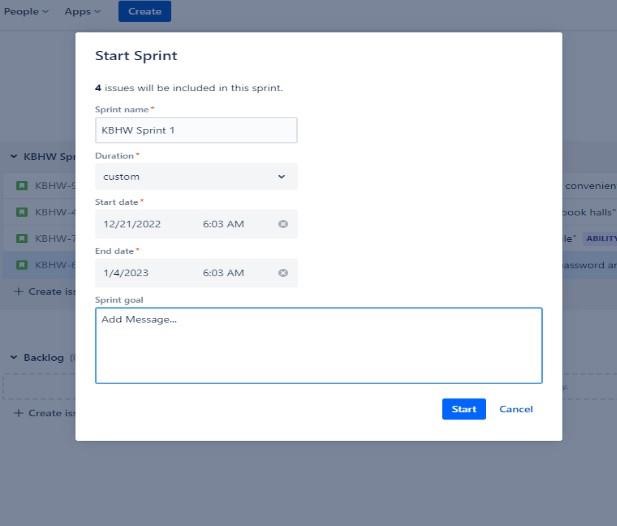
Step 8 : Or select an issue click on three dot menu and select Move to Sprint 1.



Step 9 : After moving the issues from backlog to sprint , Click on start sprint button.

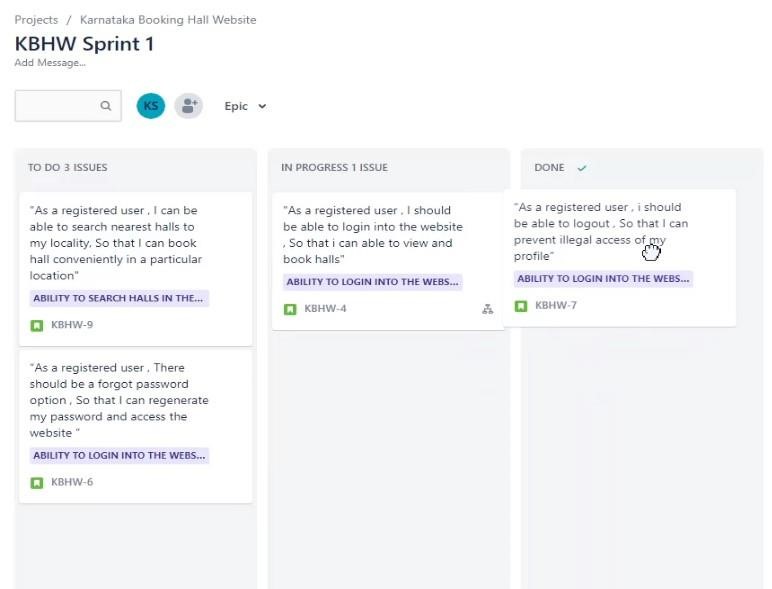


Step 10 : A dialogue will open asking for the sprint duration , set the duration accordingly and click on start.



Step 11 : Under the Board tab , we can see the issues that are in the To Do menu.

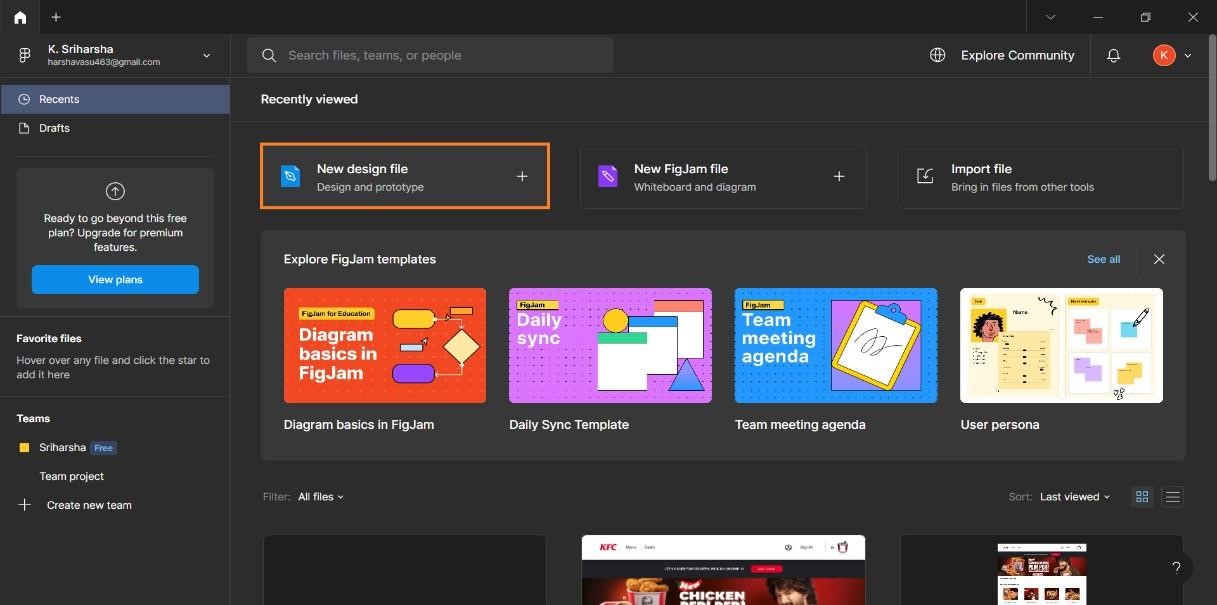
Step 12 : According to the status of each issue , drag and drop the issue from To Do menu to In Progress or Done menu.



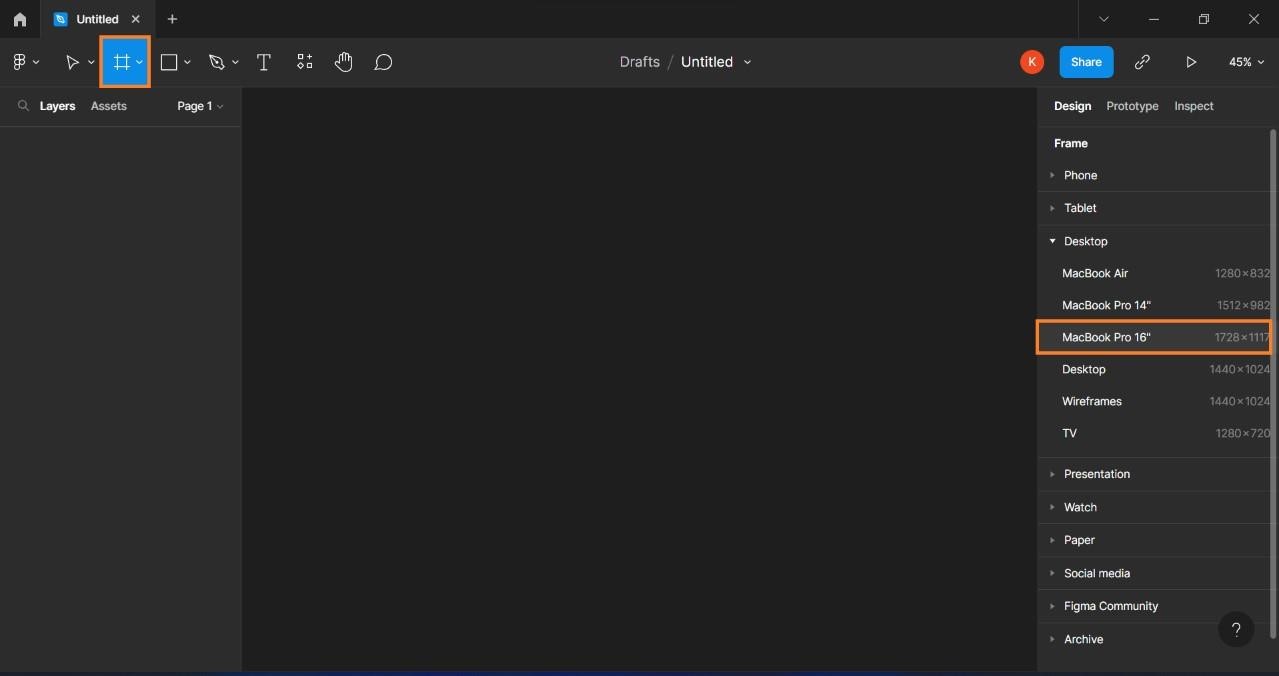
**3. Create a wireframe for user stories.**

Step 1 : Login into Figma website and download the figma desktop app.

Step 2 : In the figma dashboard , Select New Design file.

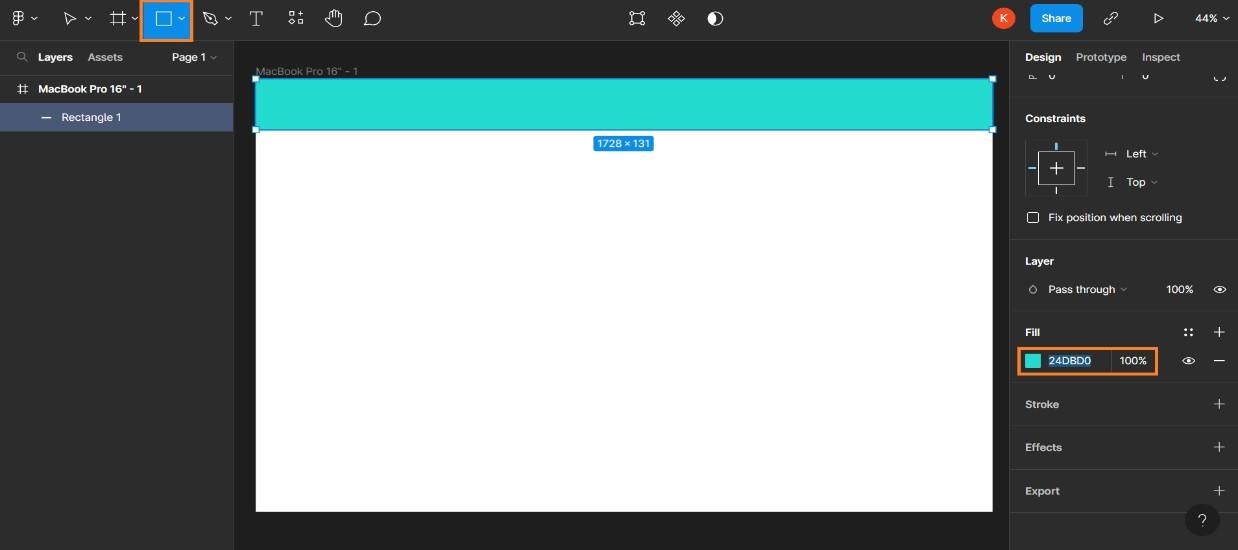


Step 3 : A blank workspace will open , Select the Frame and Reference device.

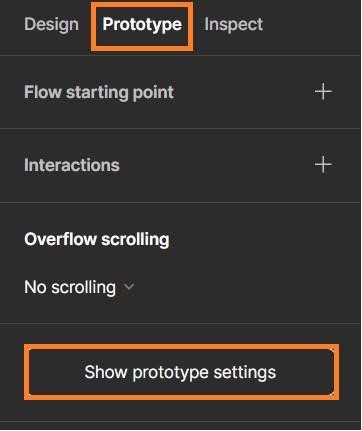


Step 4 : On the toolbar at the top , Select rectangle and start to design the UI of the web page.

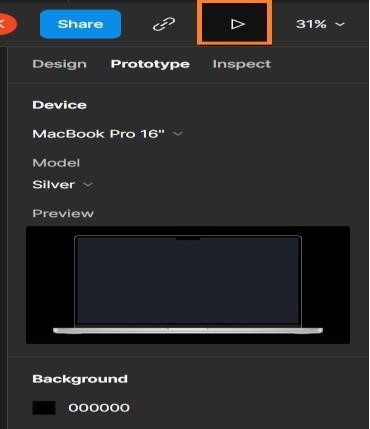
Step 5 : Use the necessary tools like line , text-box , hand tool and colour properties from the right side Design Tab.



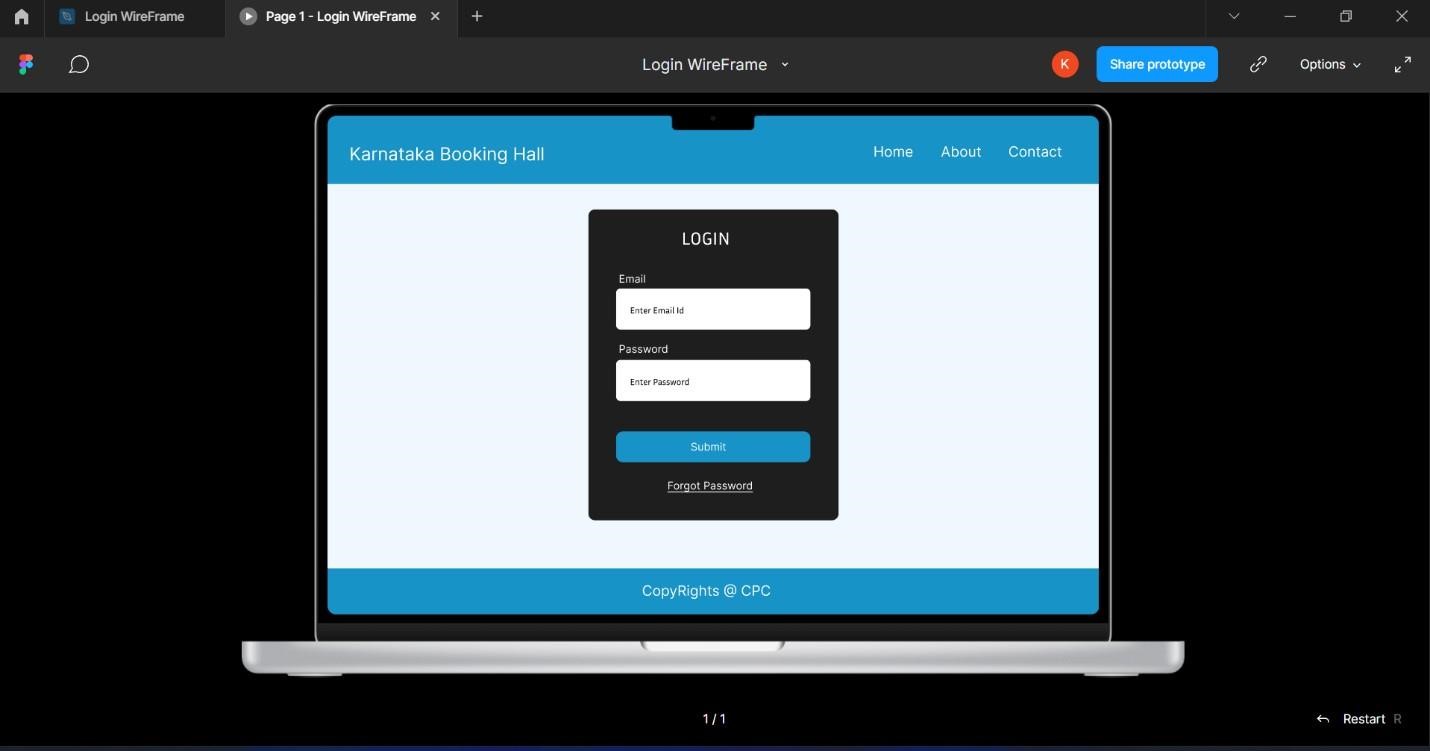
Step 6 : After building the Design , on the right menu Select Prototype menu and Click on show prototype settings.



Step 7 : Select the device to play, the model of the device and the background the prototype environment then click on play button on the top.



Step 8 : Now we can see how our designed UI will look in actual device.



**4. Write the Test case for the user stories. {Only writing}**

 **Test Case for Online convention booking hall application**

# 4.1 Test case for Login Page

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| sl no | test steps | test data | expected result | actual result | status |
| 1 | navigate to login page |  | user should able to login | User is navigated to home dashboard with successful login | pass |
| 2 | provide valid username | User =  example@gmail.com | pass |
| 3 | provide valid password | Password - 1234 | fail |
| 4 | click on login button |  | pass |

# 4.2 Test case for search bar

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| sl no | test steps | test data | expected result | actual result | status |
| 1 | navigate to search textbox | any place in karnataka | user should able to search halls | Users can find the halls in a specific locality | pass |
| 2 | provide valid locality name | any place in karnataka | pass |
| 3 | click on search button |  | fail |

1. **Create a repository in GitHub and cloning the repository using VS code.**

**5.1 Creating an empty repository in Github through VS Code.**

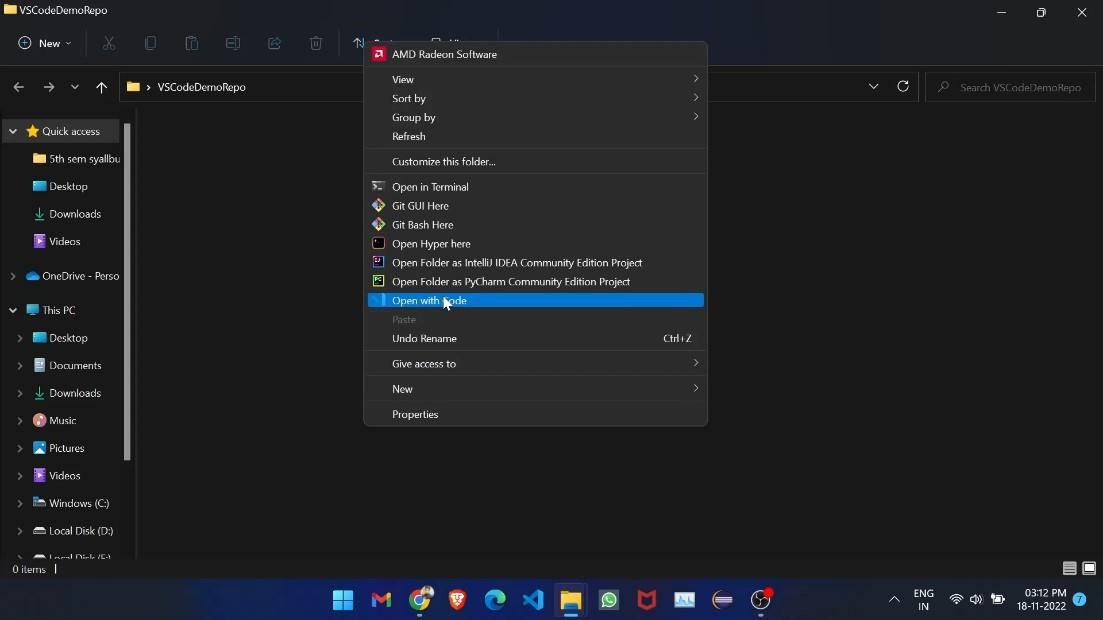
Step 1 : Login into github with the credentials.

Step 2 : Install VS Code editor to your desktop.

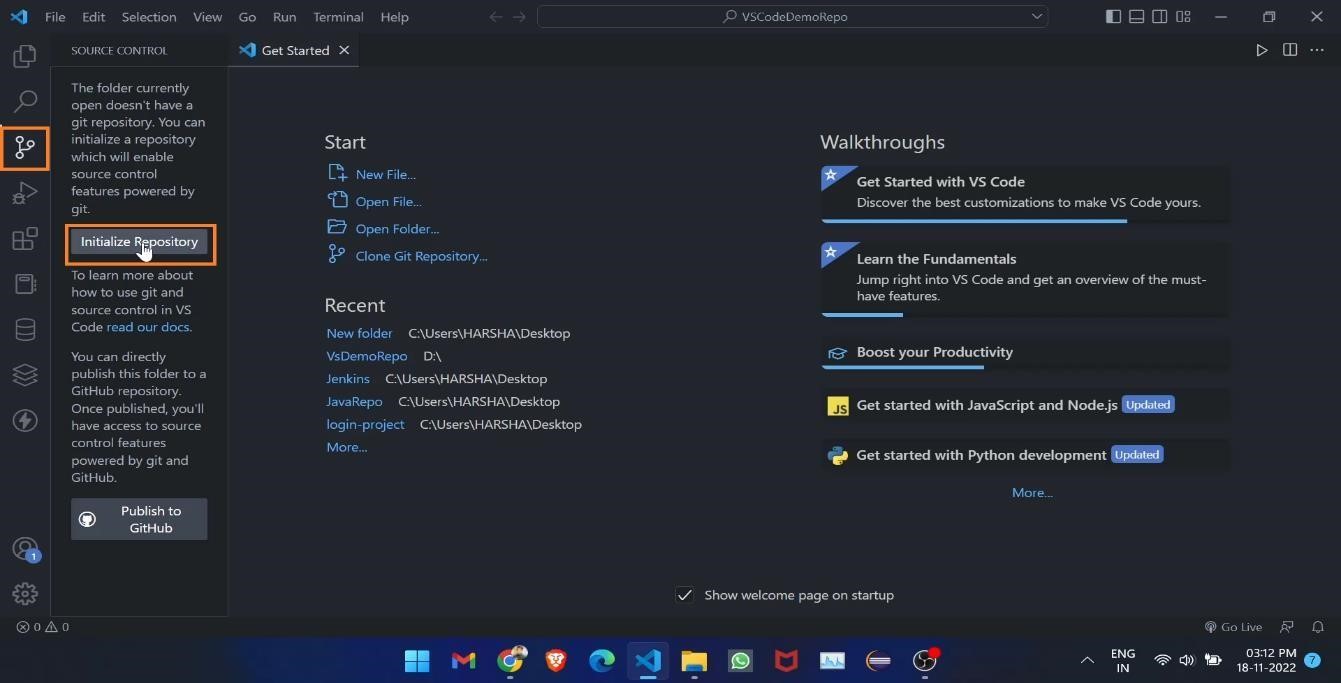
Step 3 : In the desktop , create a new empty folder by right click  new  folder.



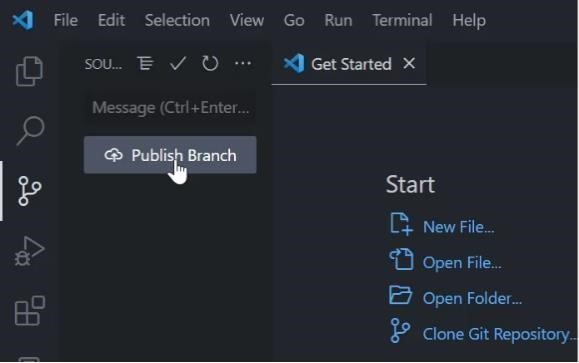
Step 4 : Open the folder and right click  open with code.



Step 5 : VS Code will be opened with the selected folder . On the left menu bar , Select Git icon and then select initialize repository option.



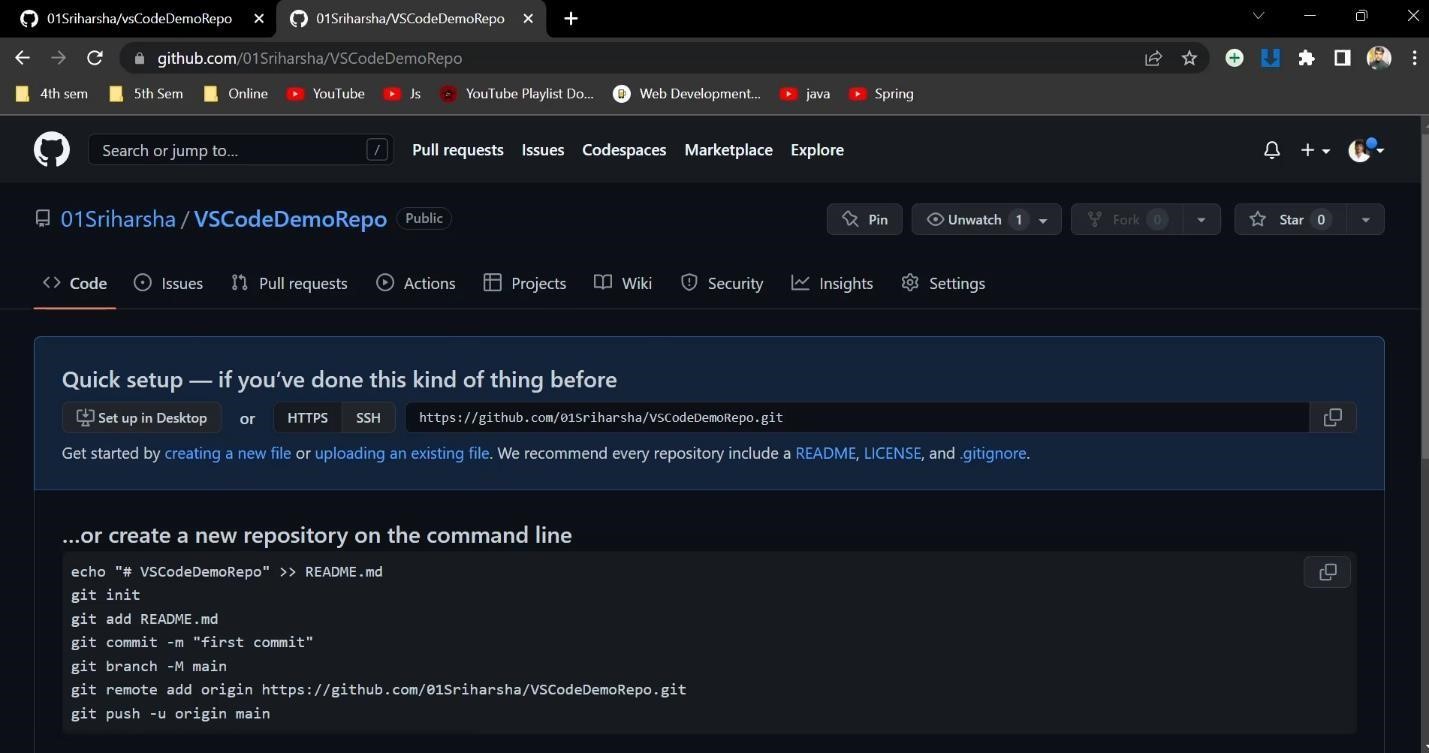
Step 6 : On next page , Click on Publish Branch . This will create a default Main branch.



Step 7 : A pop up window will open , Select Publish to public repository.



Step 8 : Now head back to github, A new empty repository will be created with the folder name that was given at the beginning.

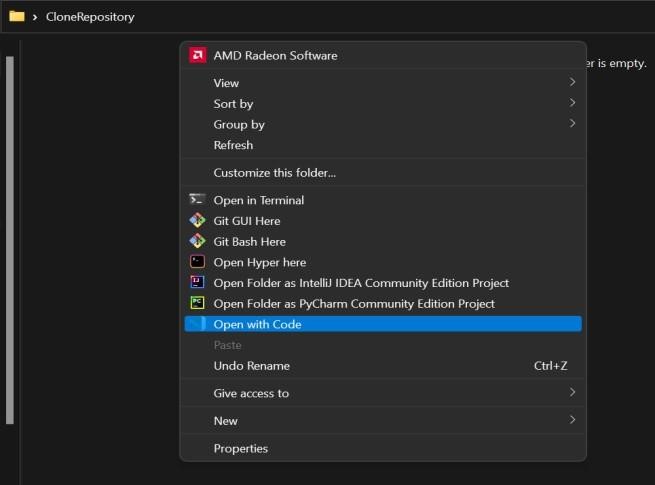


# 5.2 Steps to Clone a github repository in VS Code

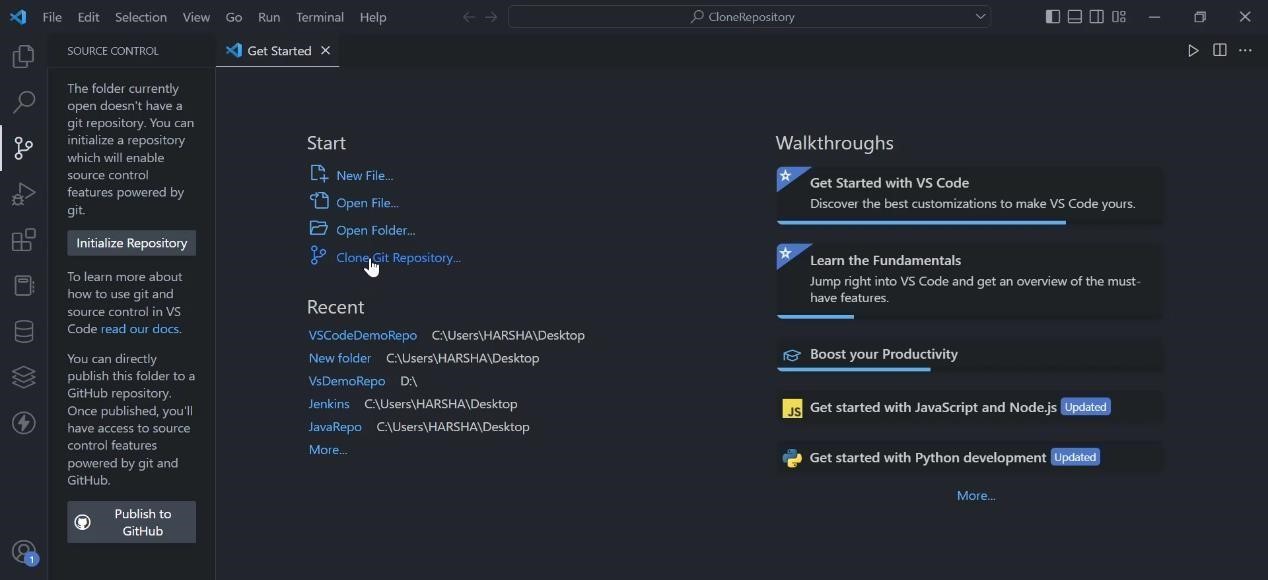
Step 1 : Step 1 : Login into github with the credentials.

Step 2 : Install VS Code editor to your desktop.

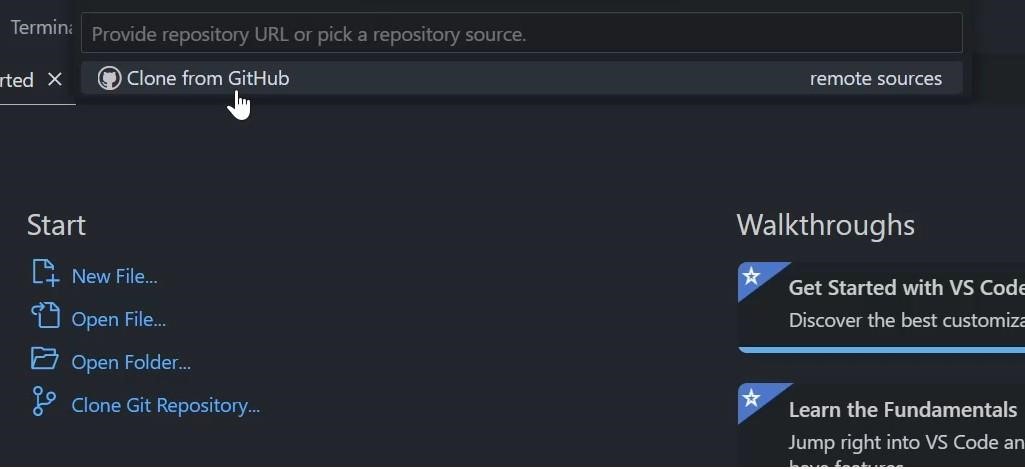
Step 3 : In the desktop , create a new empty folder by right click  new  folder. Step 4 : Open the folder and right click  open with code.



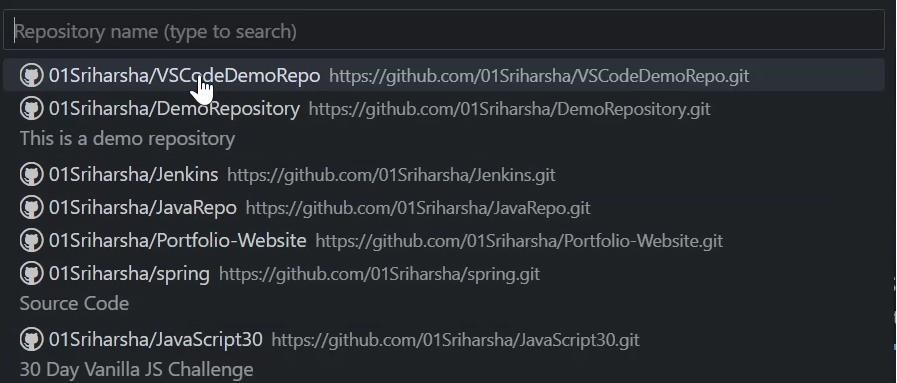
Step 5 : VS Code will be opened with the selected folder . Click on clone repository option.



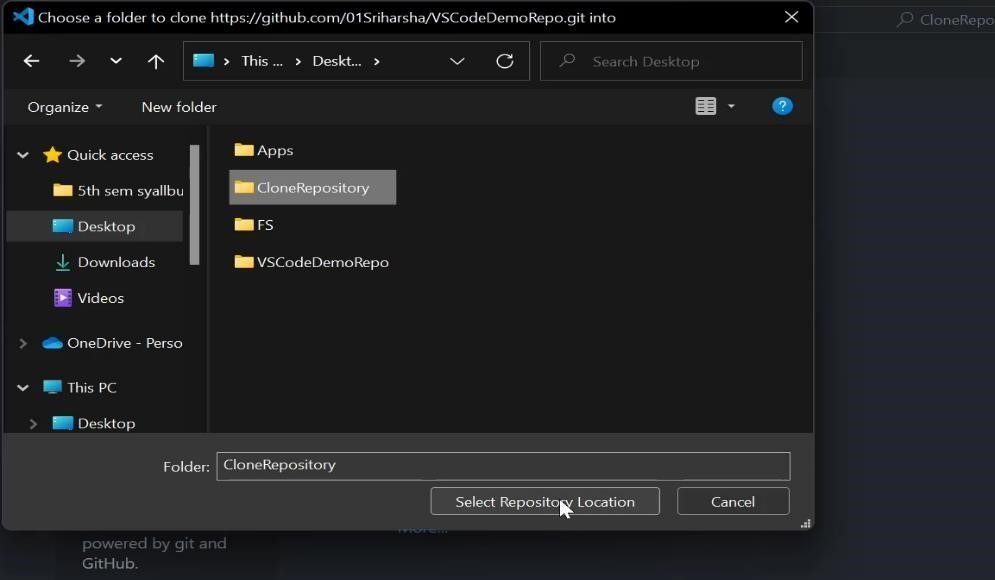
Step 6 : On the pop up window , Select clone from github . It will fetch all the repositories that are available globally and locally.



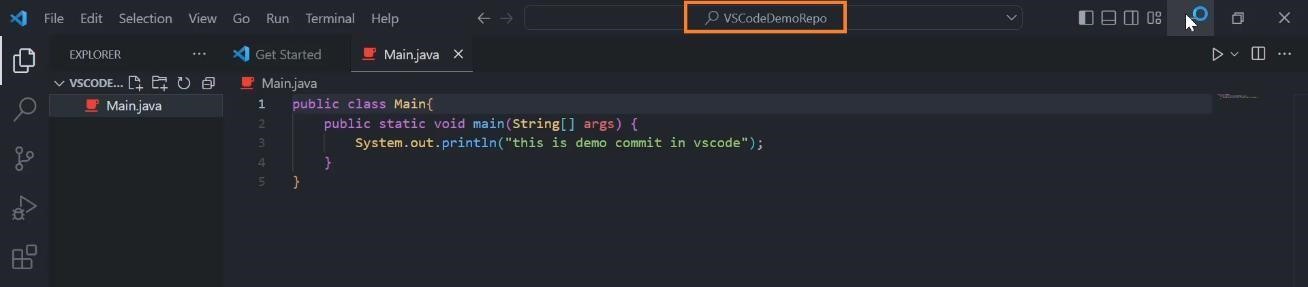
Step 7 : Search the repository you want to clone and click on it.



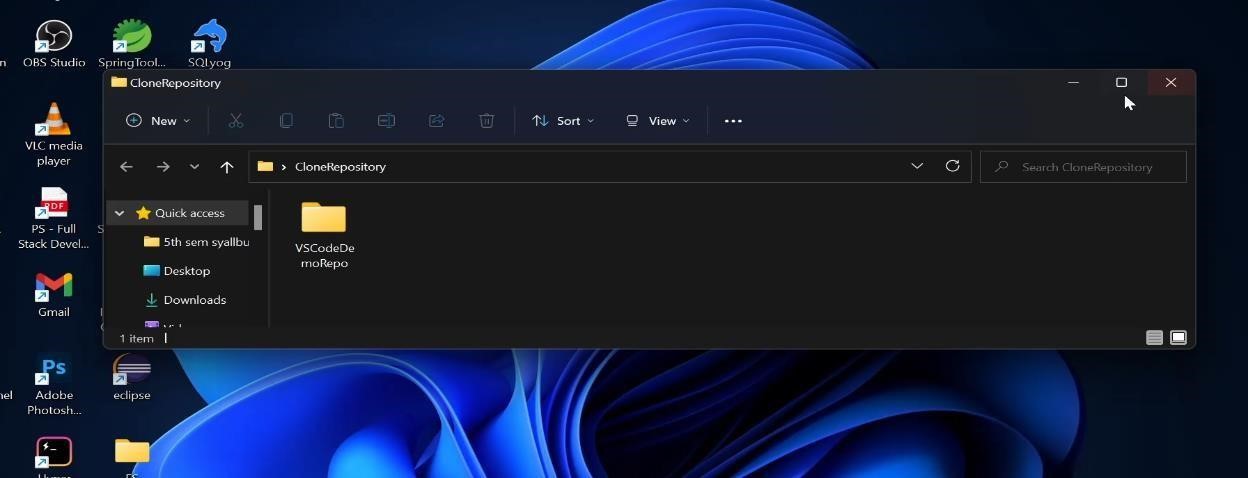
Step 8 : A dialogue will open , Select the folder where you want to clone the repository.



Step 9 : By clicking on select repository location , It will automatically open the cloned repository in VS Code .



Step 10 : The cloned repository will be stored in the folder that e created at the beginning.



**6. Perform the push and commit operation of project in Github through VS Code.**

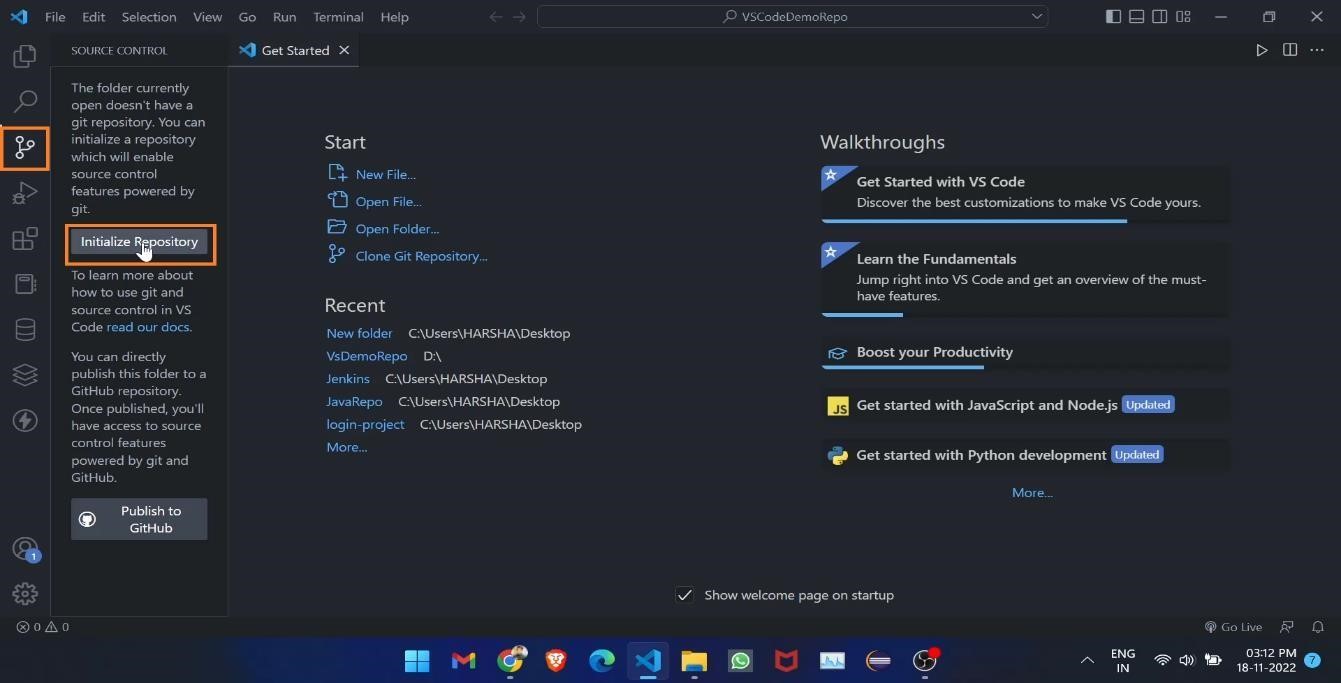
Step 1: Step 1 : Step 1 : Login into github with the credentials.

Step 2 : Install VS Code editor to your desktop.

Step 3 : In the desktop , create a new empty folder by right click  new  folder.

Step 4 : Open the folder and right click  open with code.

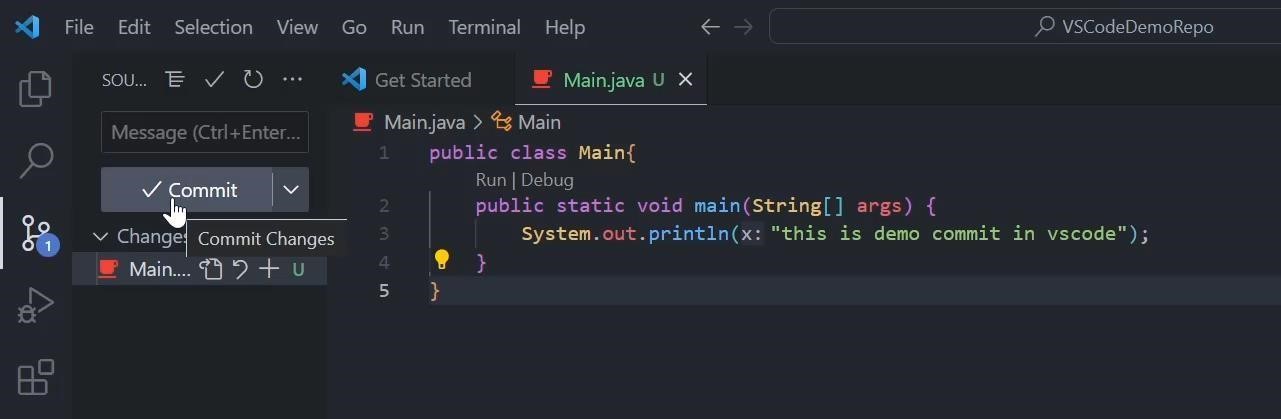
Step 5 : Step 5 : VS Code will be opened with the selected folder . On the left menu bar , Select Git icon and then select initialize repository option.



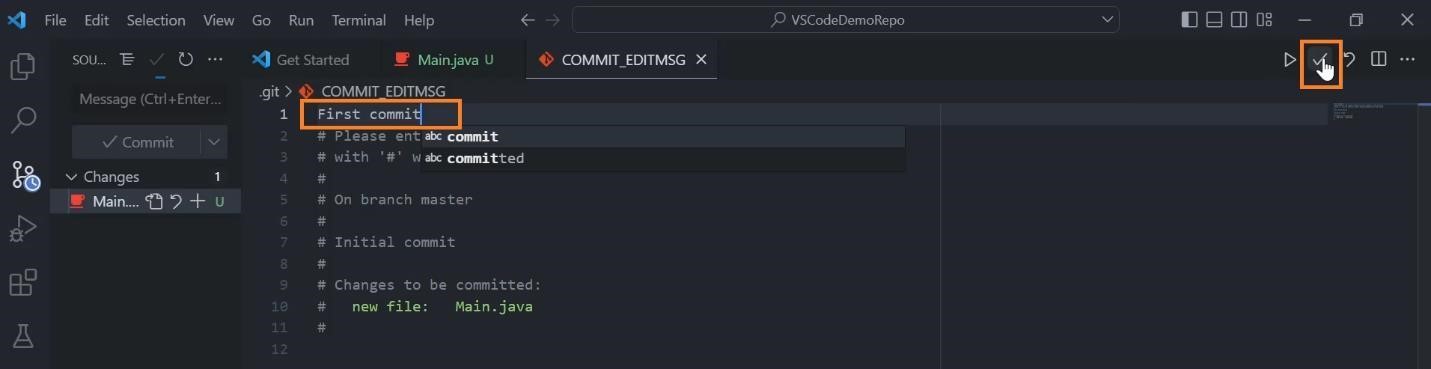
Step 6 : After initialization of the repository , Create a new file.



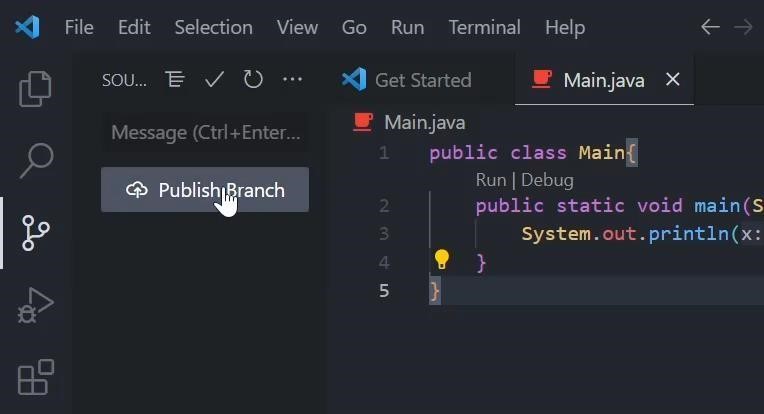
Step 7 : Select git menu on left , where we could see the modified or untracked file. Click on commit.



Step 8 : Now enter the commit message and click on check mark on the right.



Step 9 : Now click on publish branch (for new project) or Sync Changes (for existing project) to **push** the code into the github repository.



Step 10 : The push will be reflected on the github with the commit message.

**07. Launch a ubuntu server using AWS & install Jenkins software.**

→**Launching Ubuntu Server**

Step 1 - Firstly login into AWS

Step 2 – Click on services→Compute→EC2 server→ Launch instance.

Step 3 – Name server→ select any OS(‘Ubuntu’)→generate a “key-pair”→Launch instance.

Step 4 – go to Security → Security groups →Edit inbound rules → add rule.

Step 5

–

Custom type ‘Custom TCP’

→

Set port ‘8080’

→

sourc

e IPv4

→

Save Rules

Step

6

–

Go to Instance

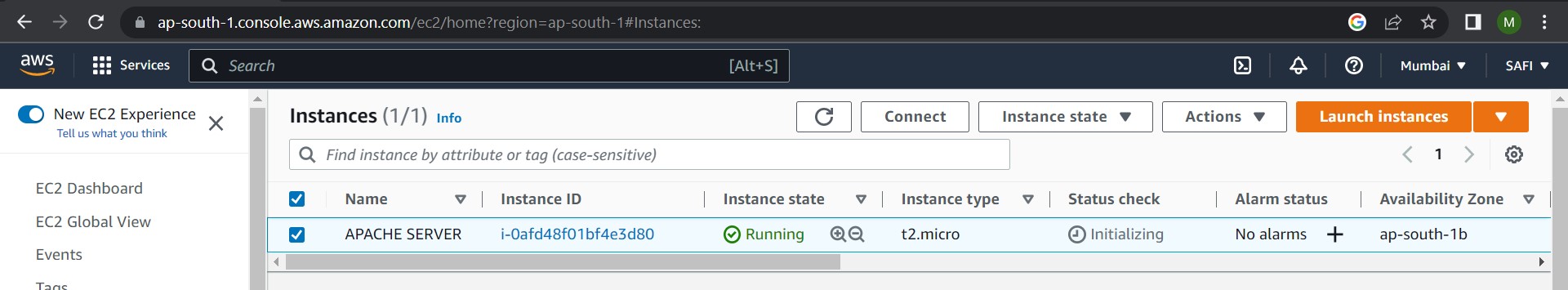
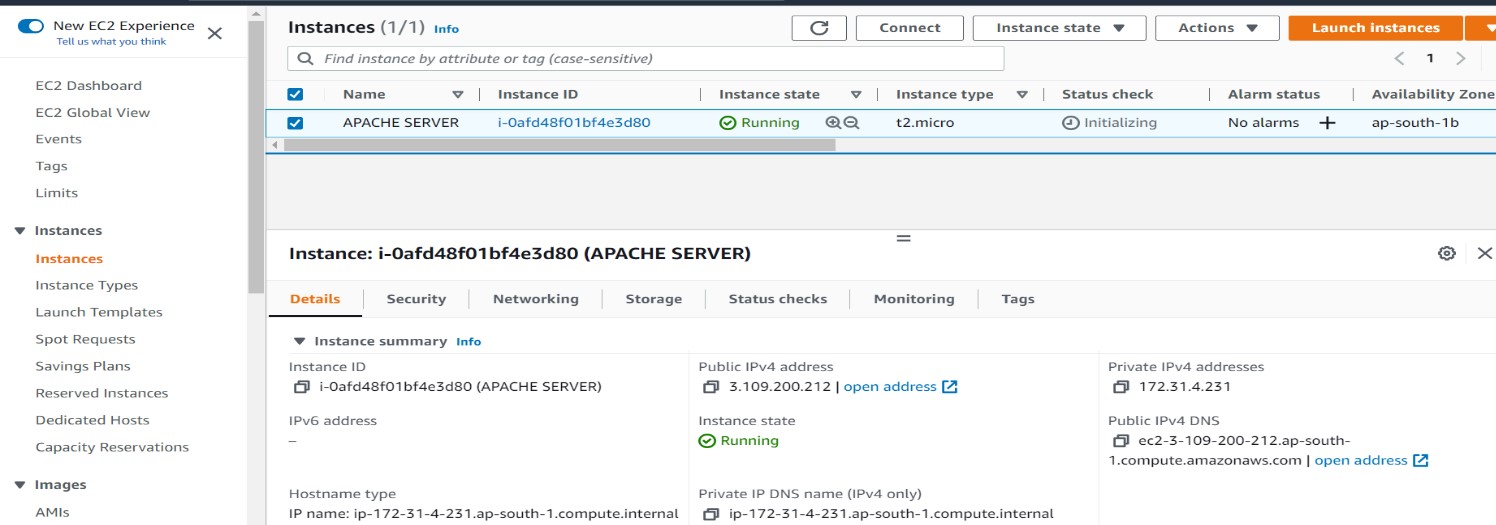
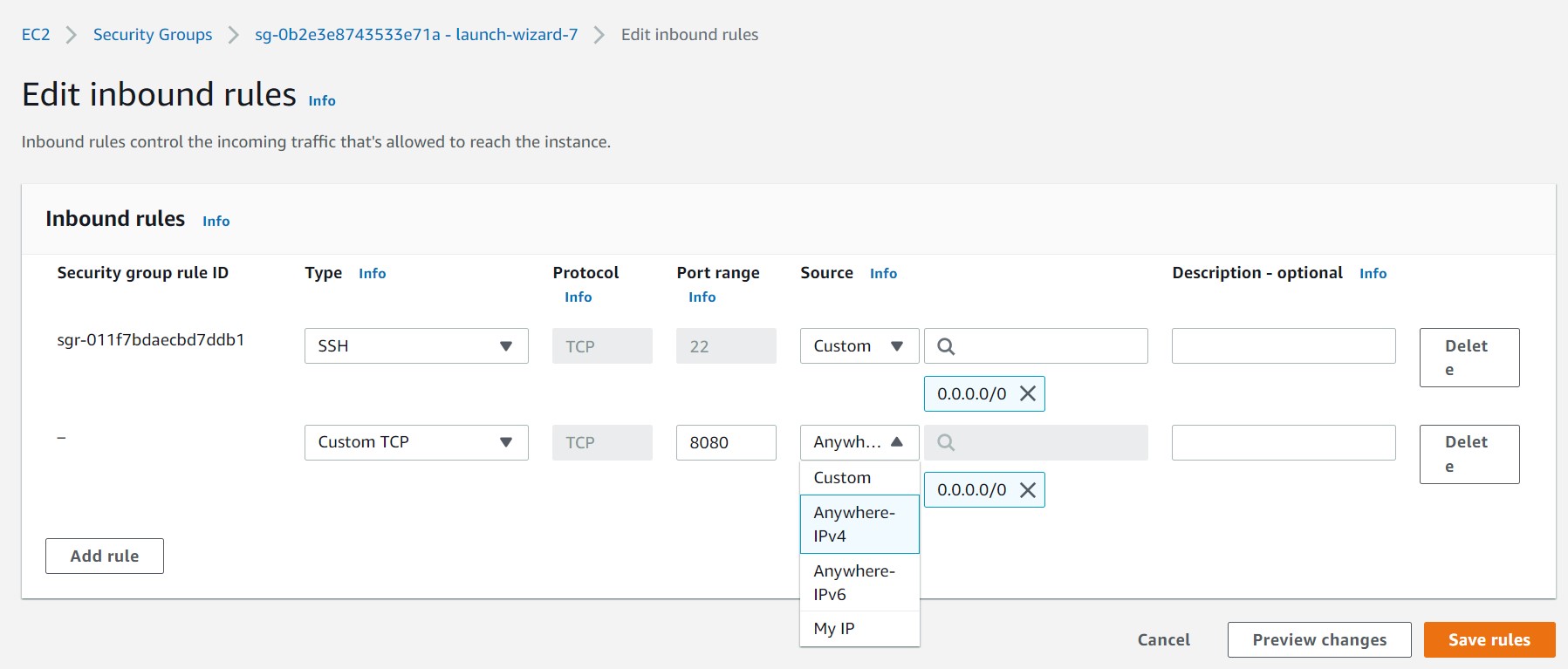
→

Connect

→

SSH Client

→



Step 7

-

Copy Example link and paste it in gti bash

Step 8

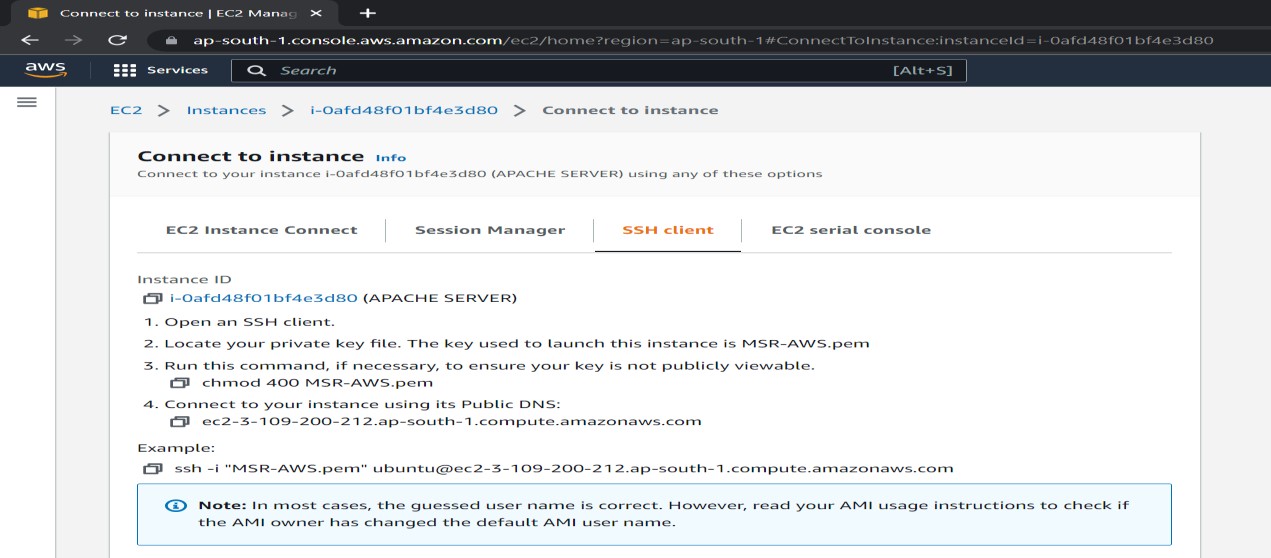
-

Follow this code in Git bash

**$sudo apt**

**-**

**get update.**



**$java --version.** → **sudo apt install openjdk-11-jre-headless {**copy & paste this}.

**$java –version. {**11.0.17**}**

**$sudo apt-get install Jenkins.** {Package 'jenkins' is not available}

Step 9 – Go to chrome→Install Jenkins ubuntu 22.04 →click Digital Ocean website & copy it

Step 10 – go to Git bash & paste the code copied

**$wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key |sudo gpg --dearmor -o /usr/share/keyrings/jenkins.gpg**

**$ sudo sh -c 'echo deb [signed-by=/usr/share/keyrings/jenkins.gpg] http://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list' $sudo apt-get update.**

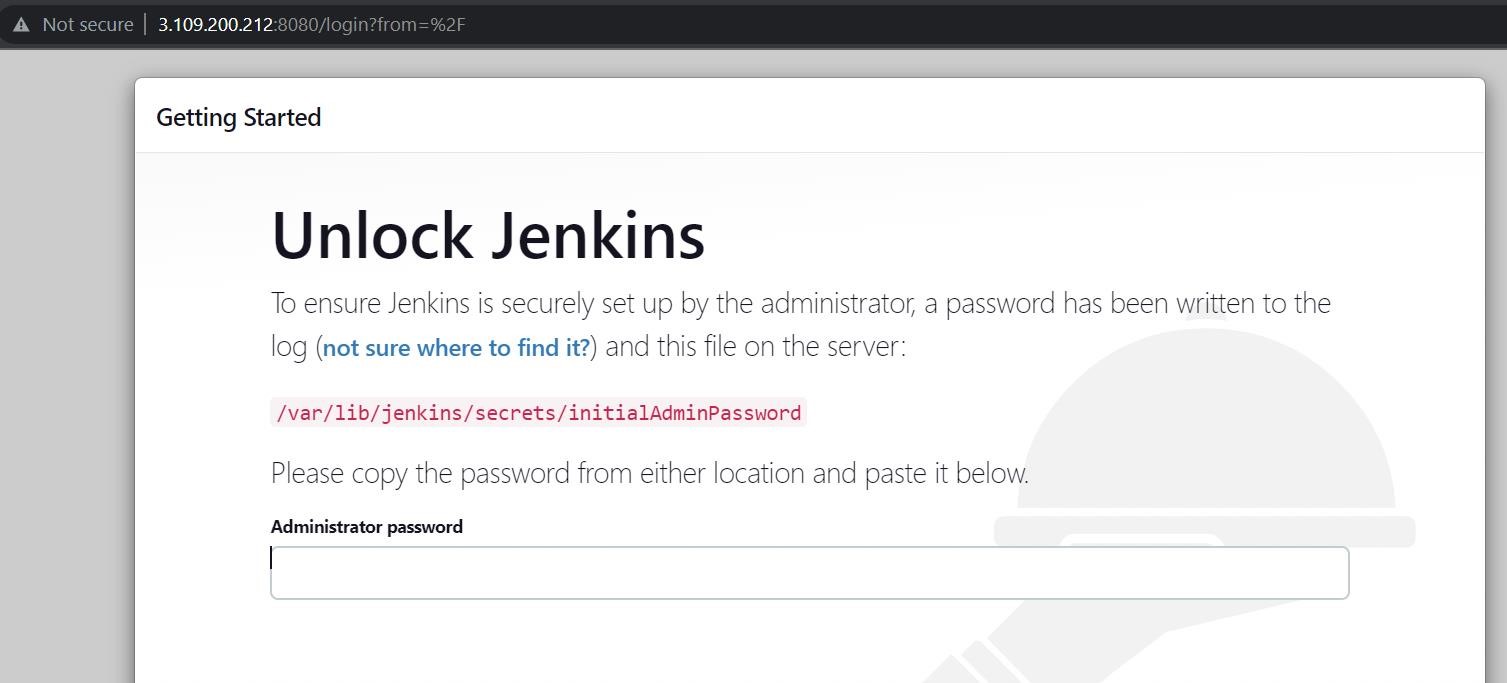
**$sudo apt-get install Jenkins. $sudo apt-get update.**

**$sudo service Jenkins status.** {active (running)} →”**:wq**”<to quit/exit>.

Step 11 – Go to chrome copy public IP from AWS & paste the public IP in new tab followed by port number{8080},

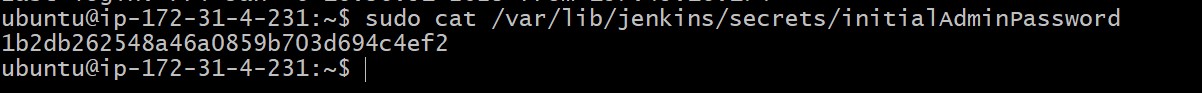
Ex: “3.109.200.212:8080”

Step 12 – Copy the password & paste it in Git bash :



Step 13 – Again, open git bash & paste above password link into git bash as command follows:- **$sudo cat /var/lib/jenkins/secrets/initialAdminPassword**

Step 14 – Copy the password displayed in git bash into Jenkins login page password box.



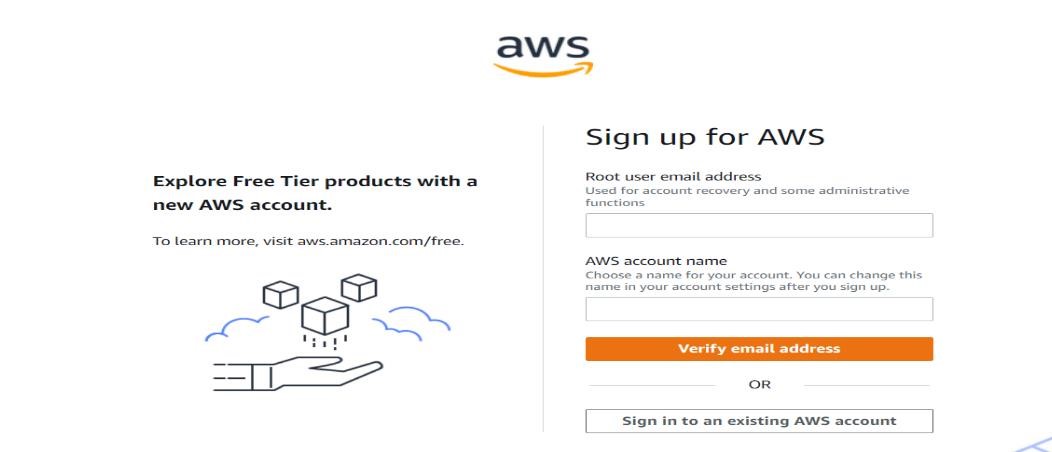
Step 15 – Install Suggested Plugins → Set User Id → User name → Password → E-mail.

Step 16 – At last Terminate the current running server.

**8. Create the cloud account in amazon Aws.**

Step1- Open the Amazon Web Services (AWS) home page.

Step-2 Choose Create an AWS Account, and then choose Create a new AWS account.



Step3 - Root user email address, enter your email address, & Verify email address.

Step4 - Create your password, enter your Root user password and Confirm root user password.

Step5 - Add your contact information → Select Personal or Business.

Step6 **-** Add a payment method.

Step7 - Verify your phone number. → Confirm your identity page, select a contact method to receive a verification code.

Step8 - Choose an AWS Support plan → Choose Complete sign up.

Step9 - Wait for account activation → Complete Sign Up.

**09.Create a student registration form using HTML & CSS.**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<title>Document</title>**

**<style> h2{color:brown;text-align: center;} div{margin: 10px;} label{color:green;}**

**input[type=text],input[type=password],input[type=number],select{width: 90%;padding: 10px;color: blue; font-size: 18px; background-color: antiquewhite;border: 1px solid green;} input:focus{outline: none; background-color: azure;}**

**.cssclr{color: red; }**

**button{border: 1px solid green;color: green;padding: 5px;}**

**.cssright{text-align: right;}**

**button:hover{background-color: green;color: white;border: none;padding: 5px;} </style>**

**</head>**

**<body>**

**<h2>Registration Form</h2>**

**<form>**

**<div>**

**<label>Enter Name:</label>**

**<input type="text" id="txtName" />**

**</div>**

**<div>**

**<label>Enter regno:</label>**

**<input type="text" id="txtpswd" />**

**</div>**

**<div>**

**<label>Enter Mobile no:</label>**

**<input type="number" id="txtName" />**

**</div>**

**<div>**

**<label>Select City:</label>**

**<select>**

**<option>--Select--</option> <option>Mysuru</option>**

**<option>Bangalore</option**

**<option>Hassan</option>**

**<option>Mandya</option>**

**</select>**

**</div>**

**<div class="cssclr">**

**<label >Select Course</label>**

**<input type="radio" name="rbtn" />Computer Science**

**<input type="radio" name="rbtn" />Mechanical**

**<input type="radio" name="rbtn" />Civil**

**<input type="radio" name="rbtn" />Electronics**

**</div>**

**<div class="cssright">**

**<button>Register</button>**

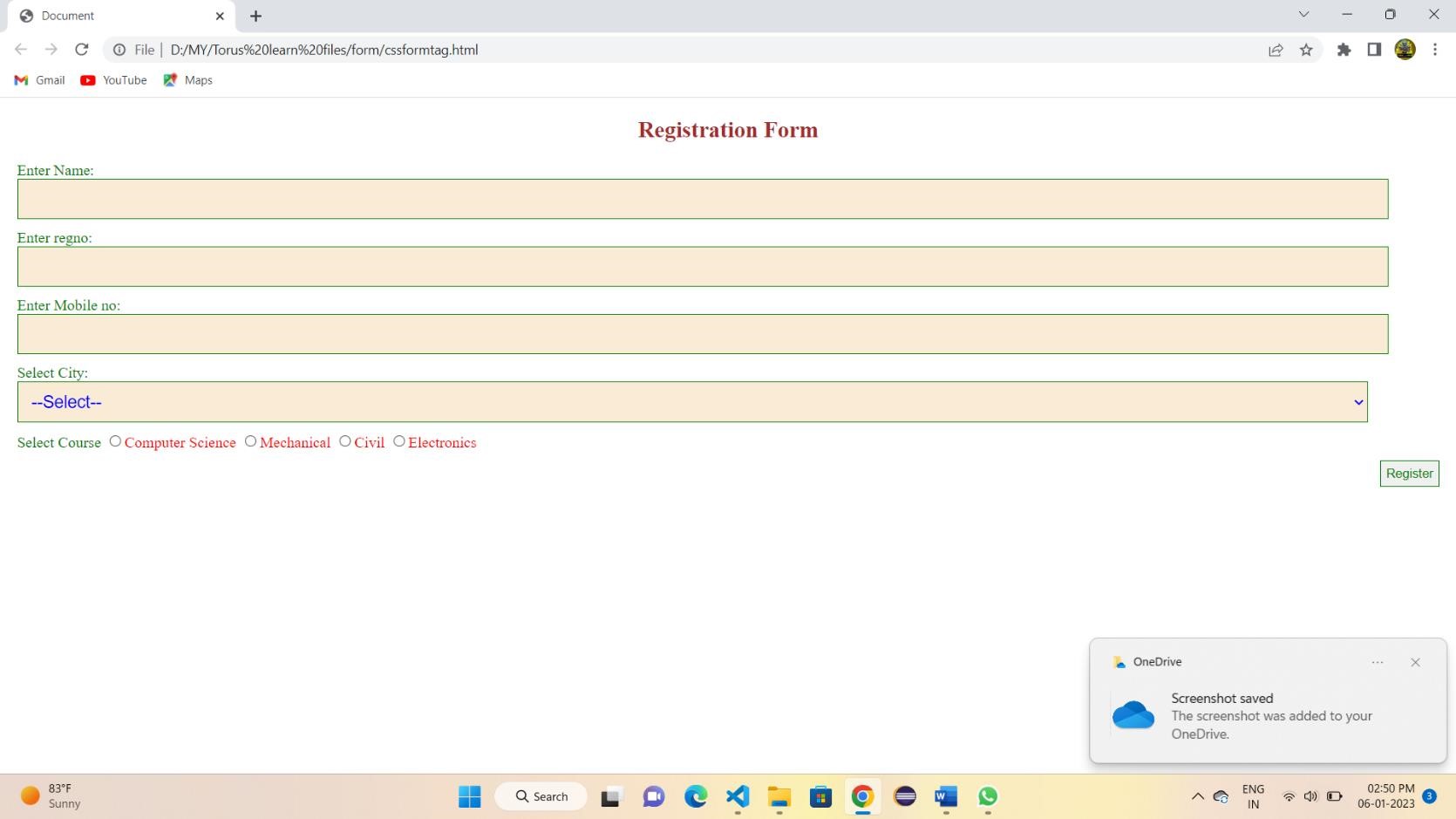
**</div>**

**</form>**

**</body>**

**</html>**

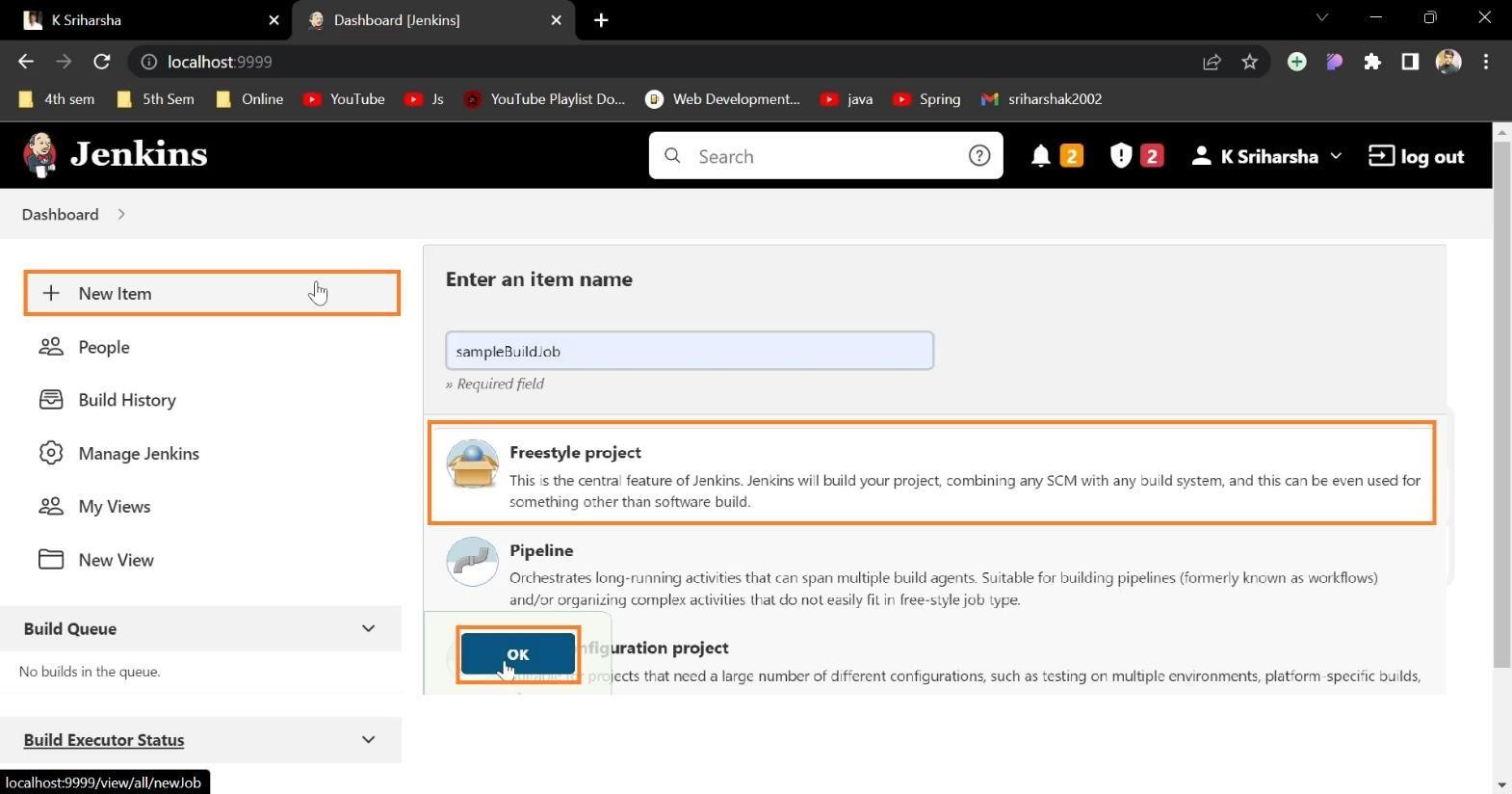
**OUTPUT:**



**10. Create build pipeline of the new project using Jenkins.**

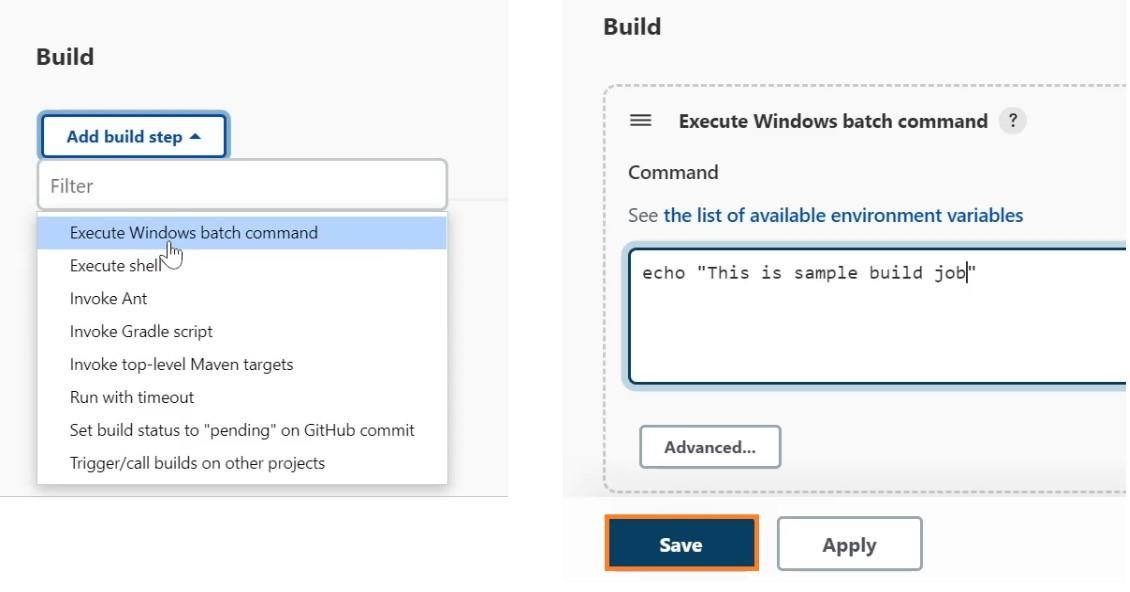
Step 1 : Go to the browser and enter the localhost followed by your jenkins running port number and hit enter.

Step 2 : On the Jenkins dashboard , Click on new item and create a free style project named **SampleBuildJob**.



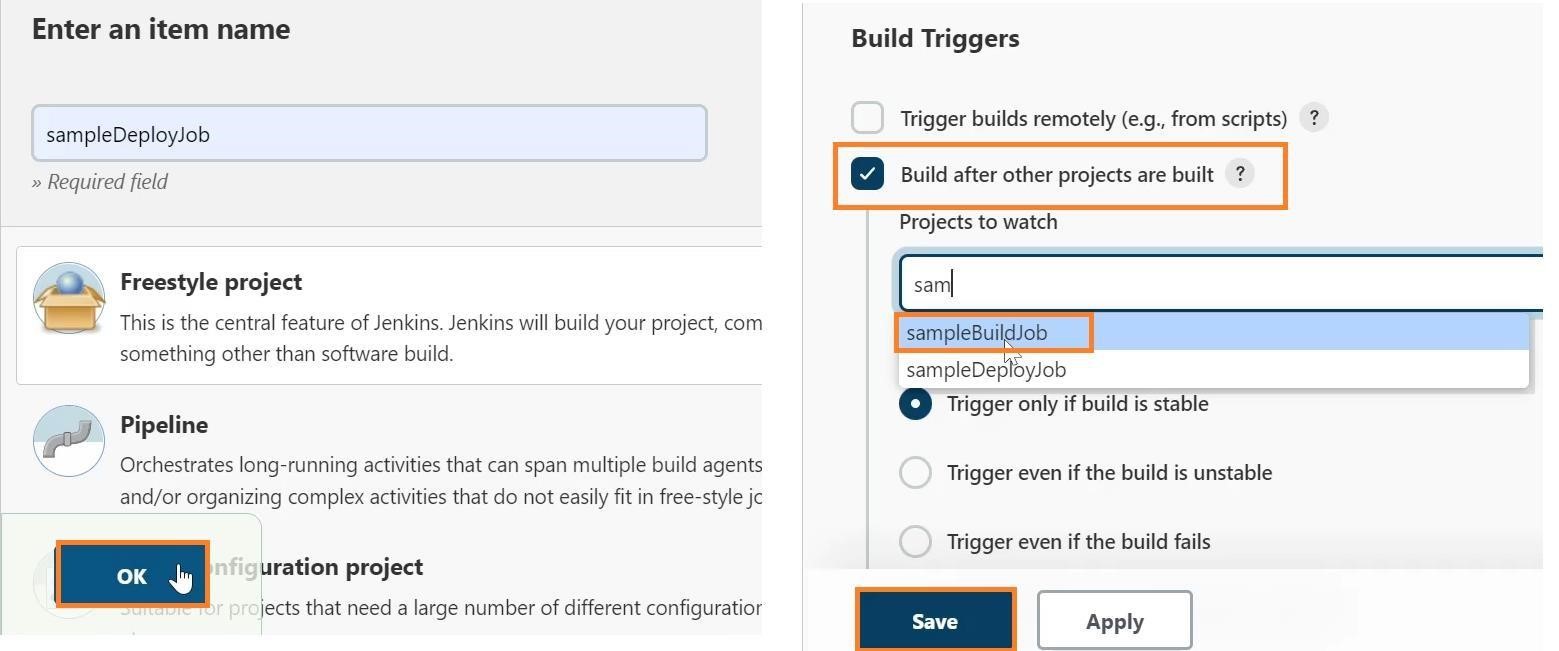
Step 3 : On the next page , add the project description and scroll down to the build tab.

Step 4 : Select Execute windows batch command and enter a command and hit save.



Step 5 : Now go back to dashboard and create another freestyle project named **SampleDeployJob**.

Step 6 : Under the **Build Triggers** tab , **select Build after other projects are built** and select the first job we created i.e , **SampleBuildJob**



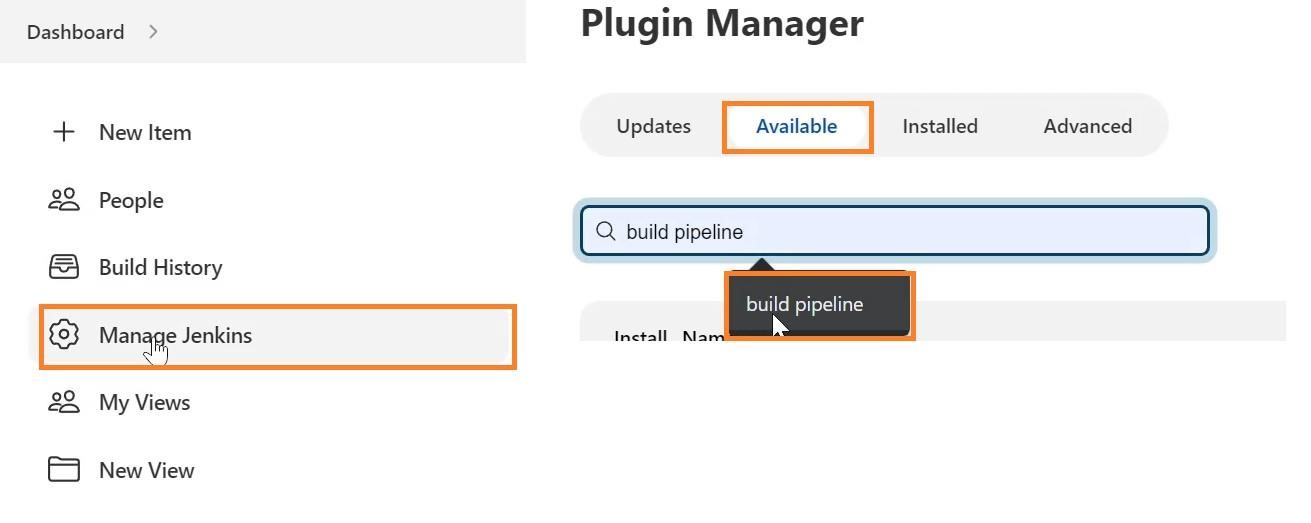
Step 7 : Follow the same procedures and create two more projects named **SampleTestJob** and **SampleReleaseJob** respectively.

Step 8 : : Under the Build Triggers tab , select Build after other projects are built and add **SampleDeployJob** for **SampleTestJob** project.

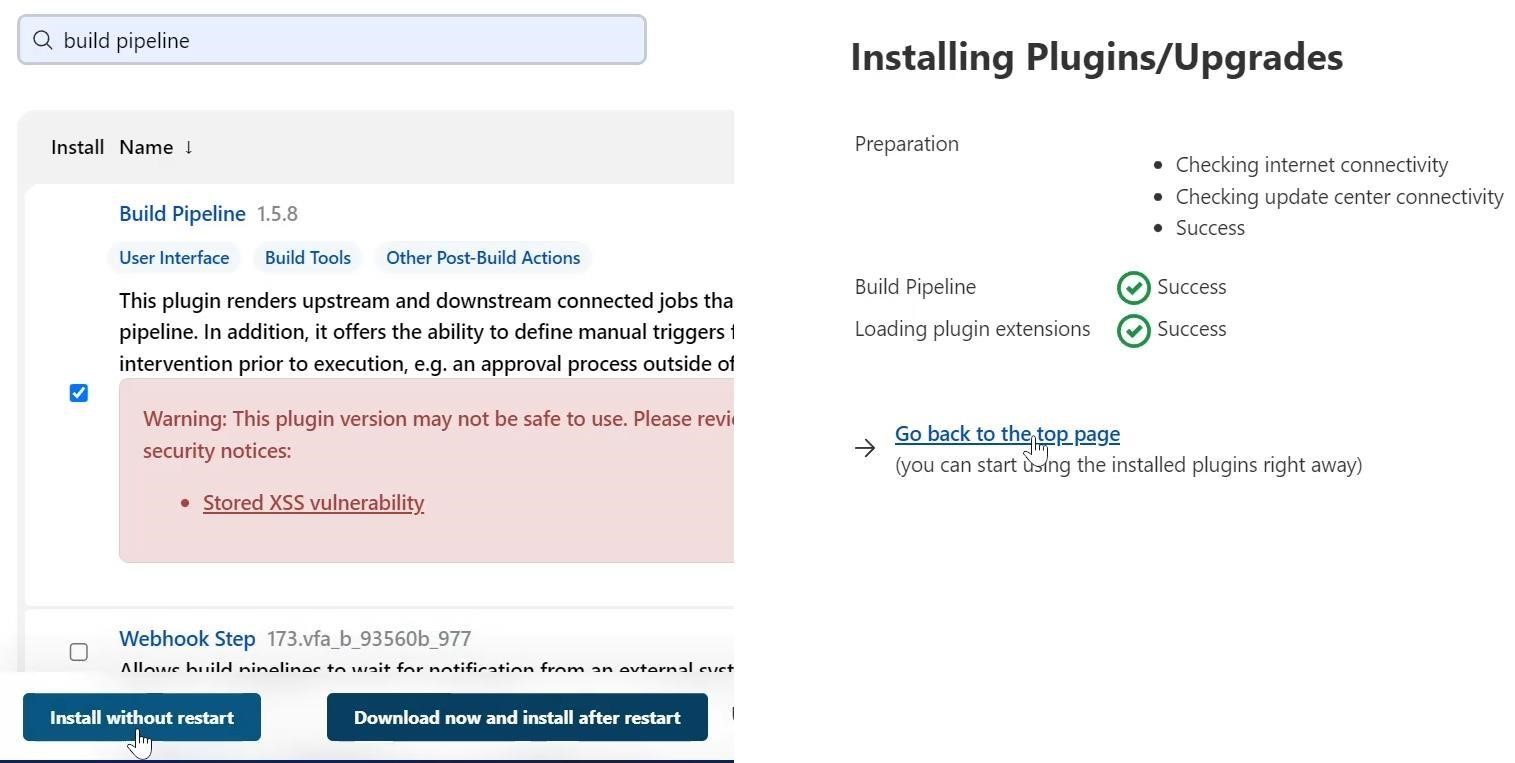
Step 9 : Likewise add **SampleTestJob** for **SampleReleaseJob** project .

Step 10 : After creation of all four projects , head back to dashboard and select **Manage Jenkins.**

Step 11 : In the **Plugin manager** section , select **Available** and search for **Build Pipeline** plugin.

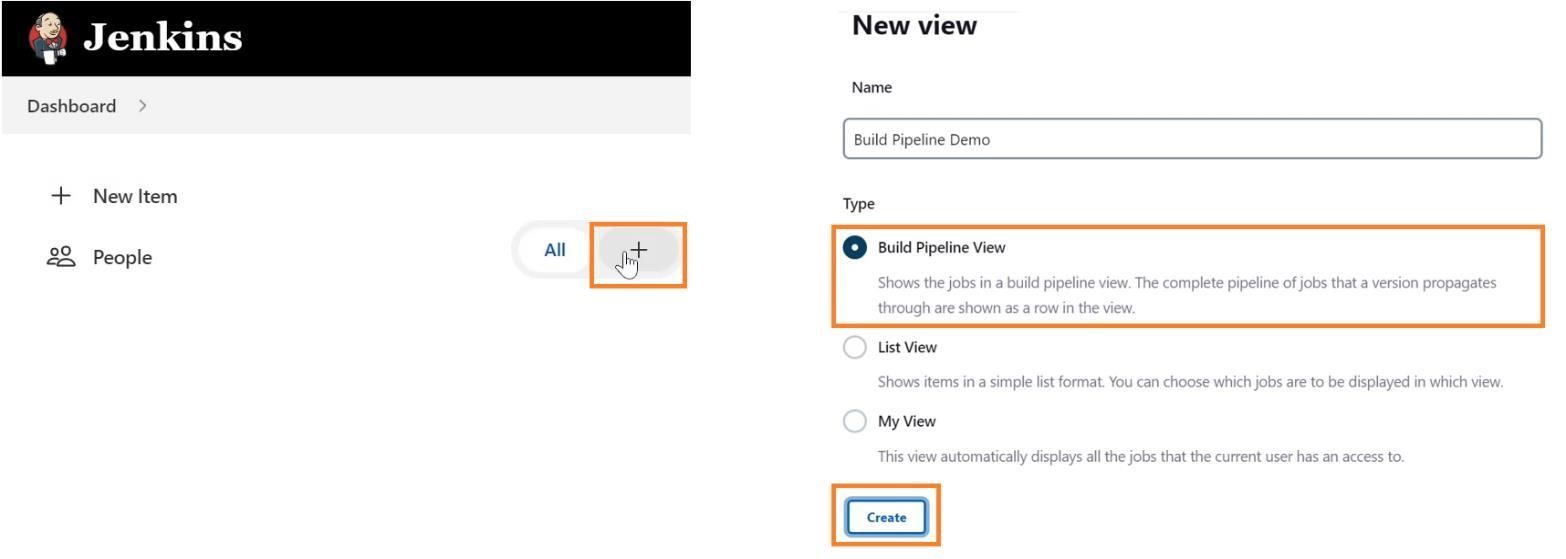


Step 12 : Once found , Select the checkbox and Click on Install without restart.



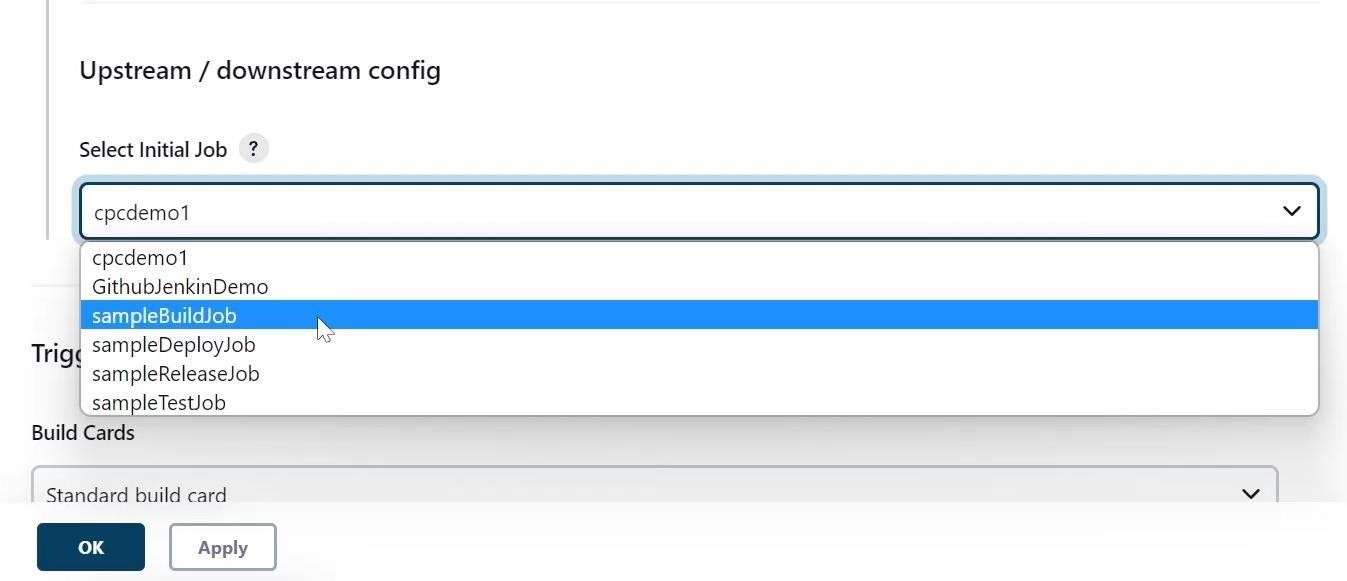
Step 13 : Now head back to dashboard , and click on + icon.

Step 14 : On the next page , Give a name to the pipeline and Select the view type to **Build Pipeline View** and click on **Create.**



Step 15 : On the next page , add pipeline description and scroll down to **Select Initial Job** section.

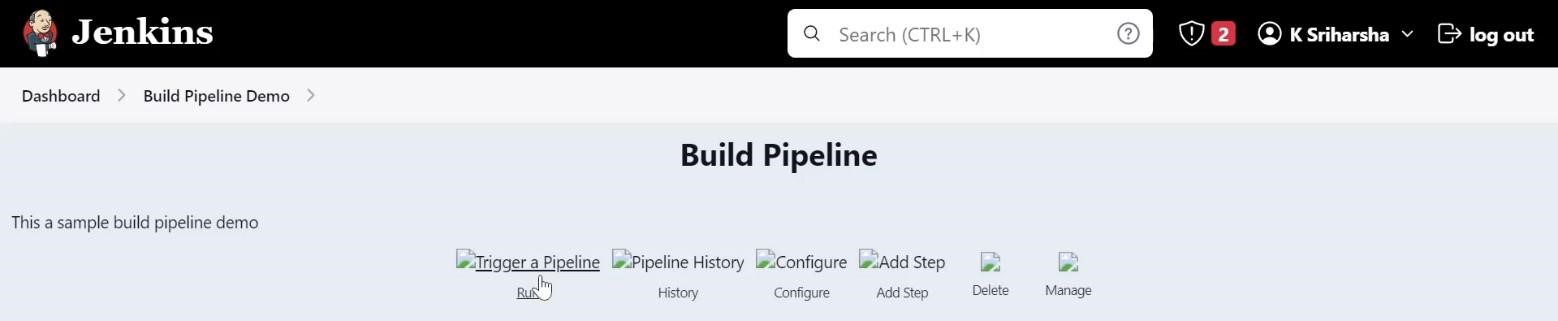
Step 16 : Select the initial job as SamplebuildJob.



Step 17 : Scroll down to the **Display Options** tab and select the number of displayed builds to be 5 , click apply and ok.



Step 18 : On the next page , Click on **Trigger a Pipeline** to view the pipeline.



Step 19 : Now you can see the pipeline build flow of all the four jobs.

Step 20 : All the successful built jobs are shown in Green , pending in blue and failed in red.



**11. Demonstrate installation of Jenkins & build a hello world program using ubuntu server.**

→**Launching Ubuntu Server**

Step 1 - Firstly login into AWS

Step 2 – Click on services→Compute→EC2 server→ Launch instance.

Step 3 – Name server→ select any OS(‘Ubuntu’)→generate a “key-pair”→Launch instance.

Step 4 – go to Security → Security groups →Edit inbound rules → add rule.

Step

5

–

Custom type ‘Custom TCP’

→

Set port ‘8080’

→

source IPv4

→

Save Rules

Step

6

–

Go to Instance

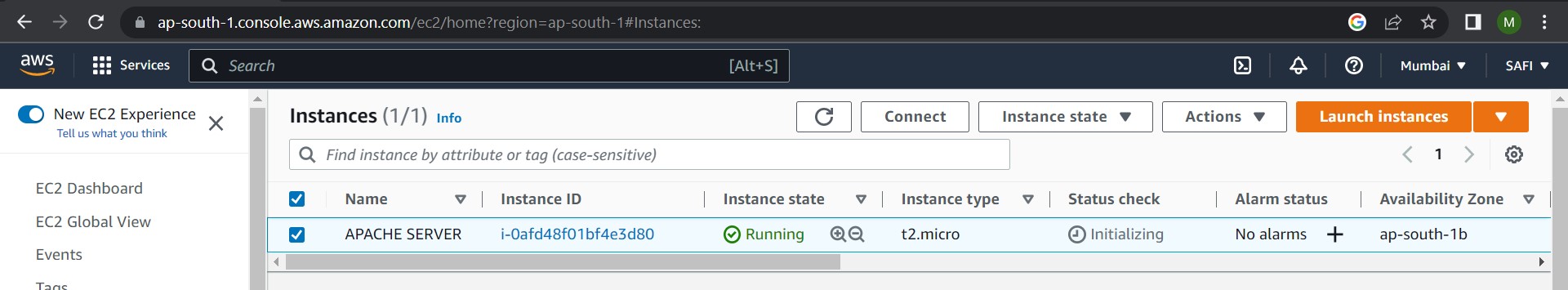
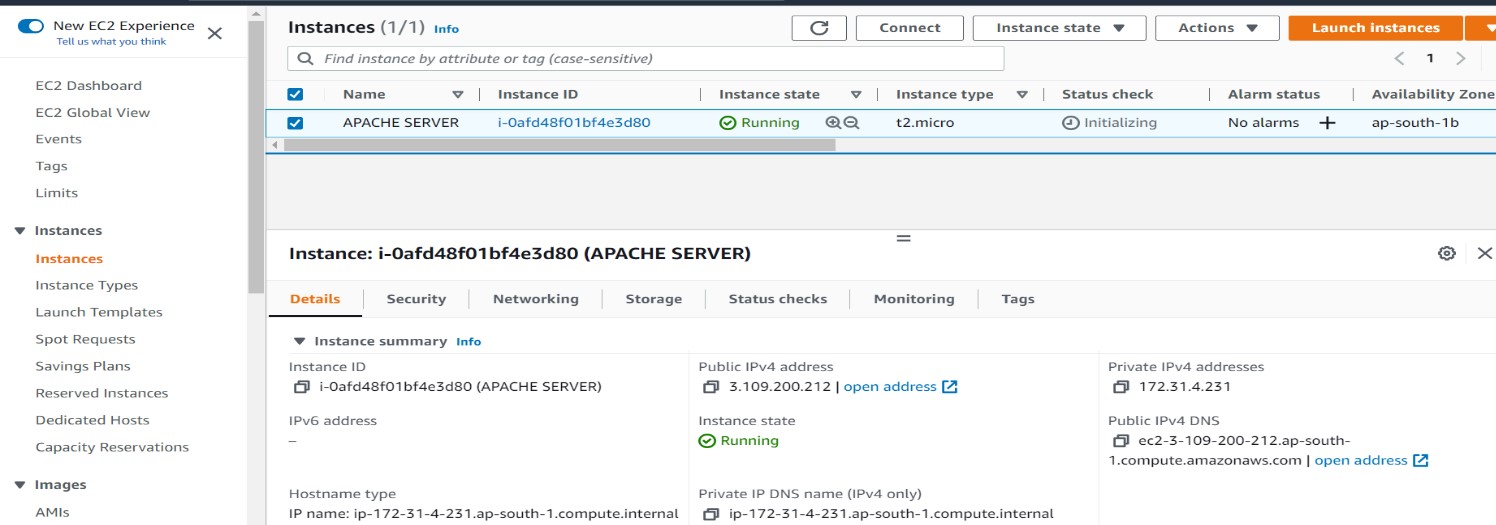
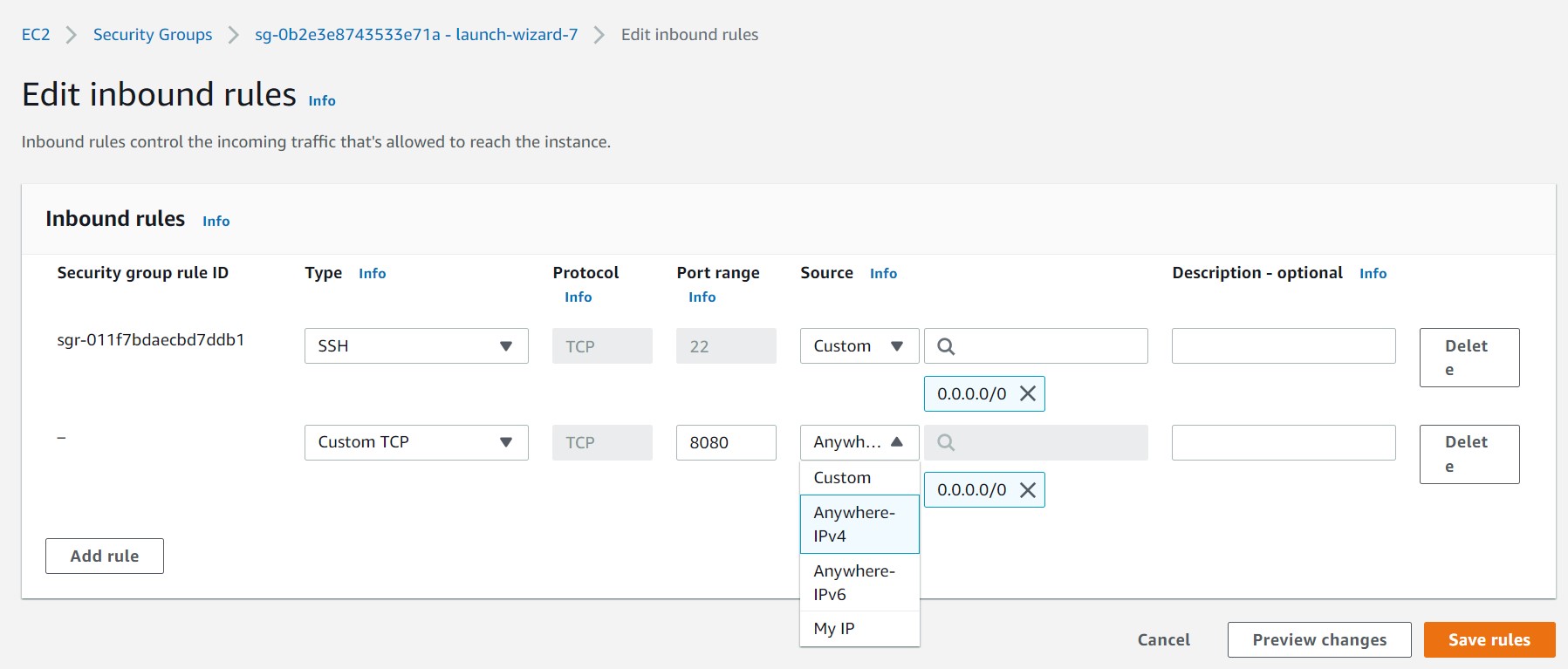
→

Connect

→

SSH Client

→



Step 7

-

Copy Example link and paste it in gti bash

Step

8

-

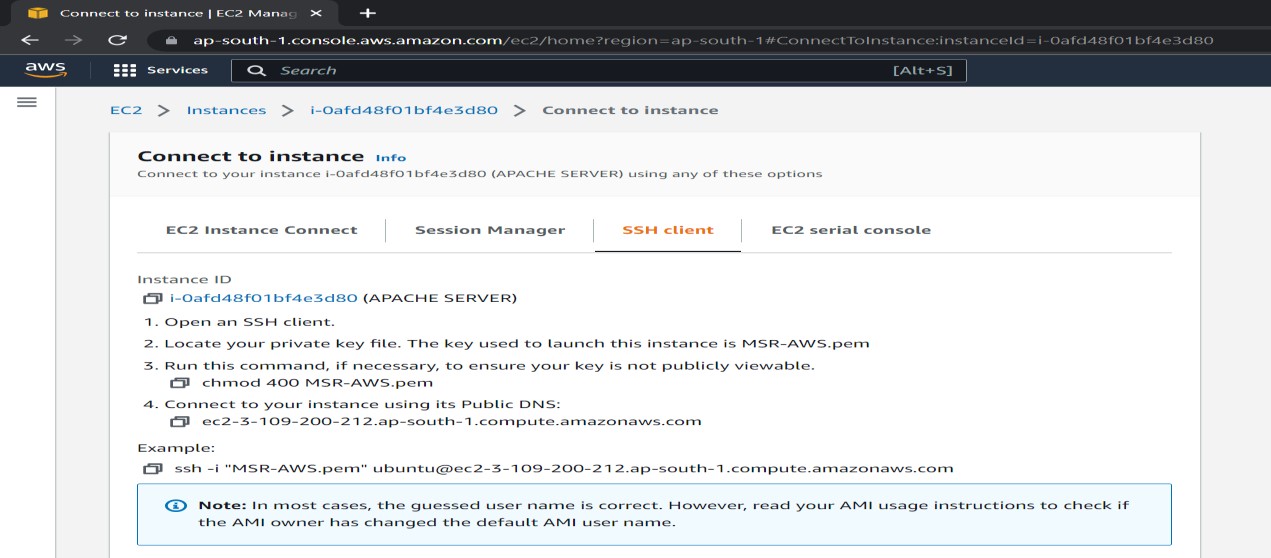
Follow

this code in Git bash

**$sudo apt**

**-**

**get update.**



**$java --version.** → **sudo apt install openjdk-11-jre-headless {**copy & paste this}.

**$java –version. {**11.0.17**}**

**$sudo apt-get install Jenkins** {Package 'jenkins' is not available}

Step 9 – Go to chrome→Install Jenkins ubuntu 22.04 →click Digital Ocean website & copy it

Step 10 – go to Git bash & paste the code copied

**$wget -q -O - https://pkg.jenkins.io/debian-stable/jenkins.io.key |sudo gpg --dearmor -o /usr/share/keyrings/jenkins.gpg**

**$ sudo sh -c 'echo deb [signed-by=/usr/share/keyrings/jenkins.gpg] http://pkg.jenkins.io/debian-stable binary/ > /etc/apt/sources.list.d/jenkins.list' $sudo apt-get update.**

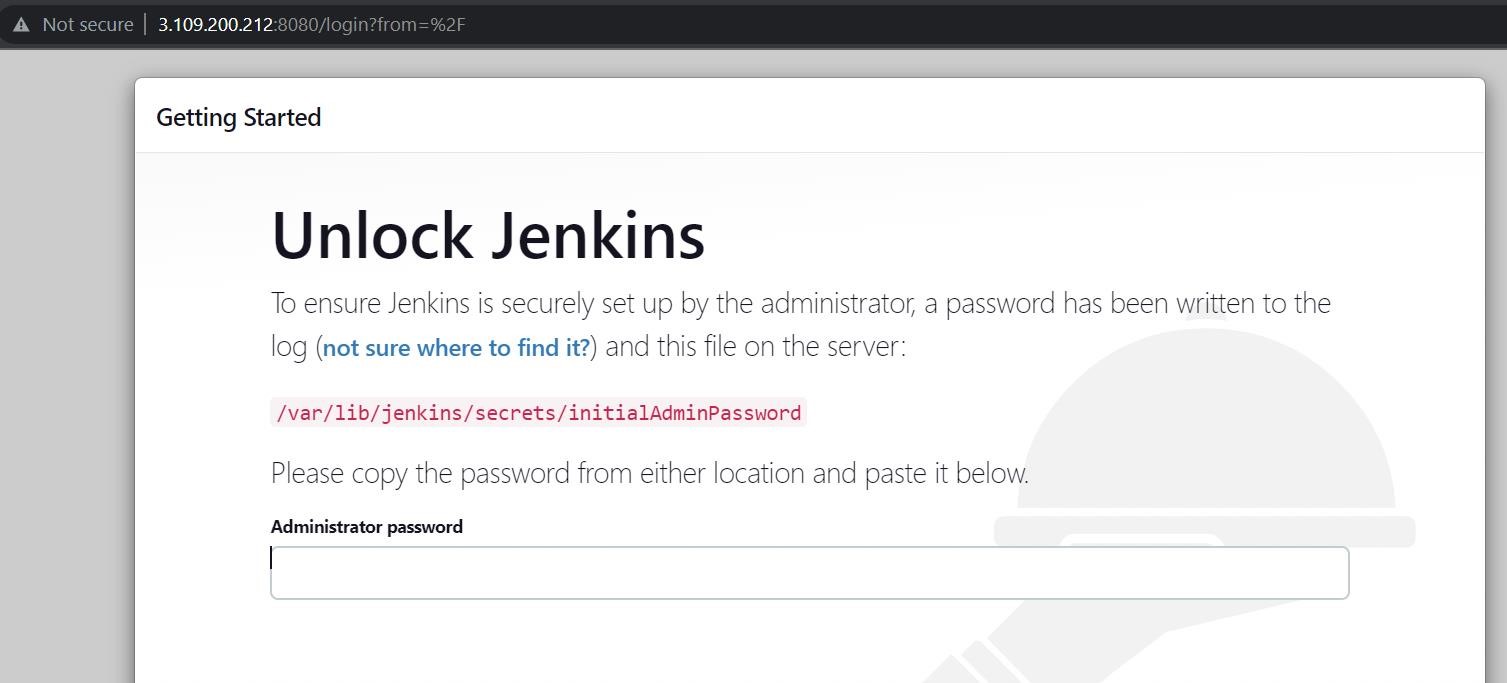
**$sudo apt-get install Jenkins. $sudo apt-get update.**

**$sudo service Jenkins status.** {active (running)} →”**:wq**”<to quit/exit>.

Step 11 – Go to chrome copy public IP from AWS & paste the public IP in new tab followed by port number{8080},

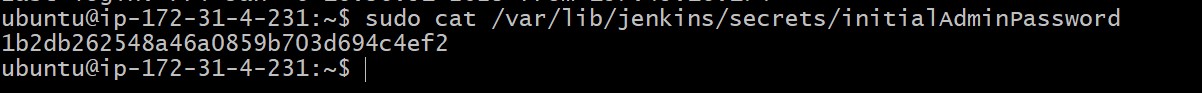
Ex: “3.109.200.212:8080”

Step 12 Copy the password & paste it in Git bash :



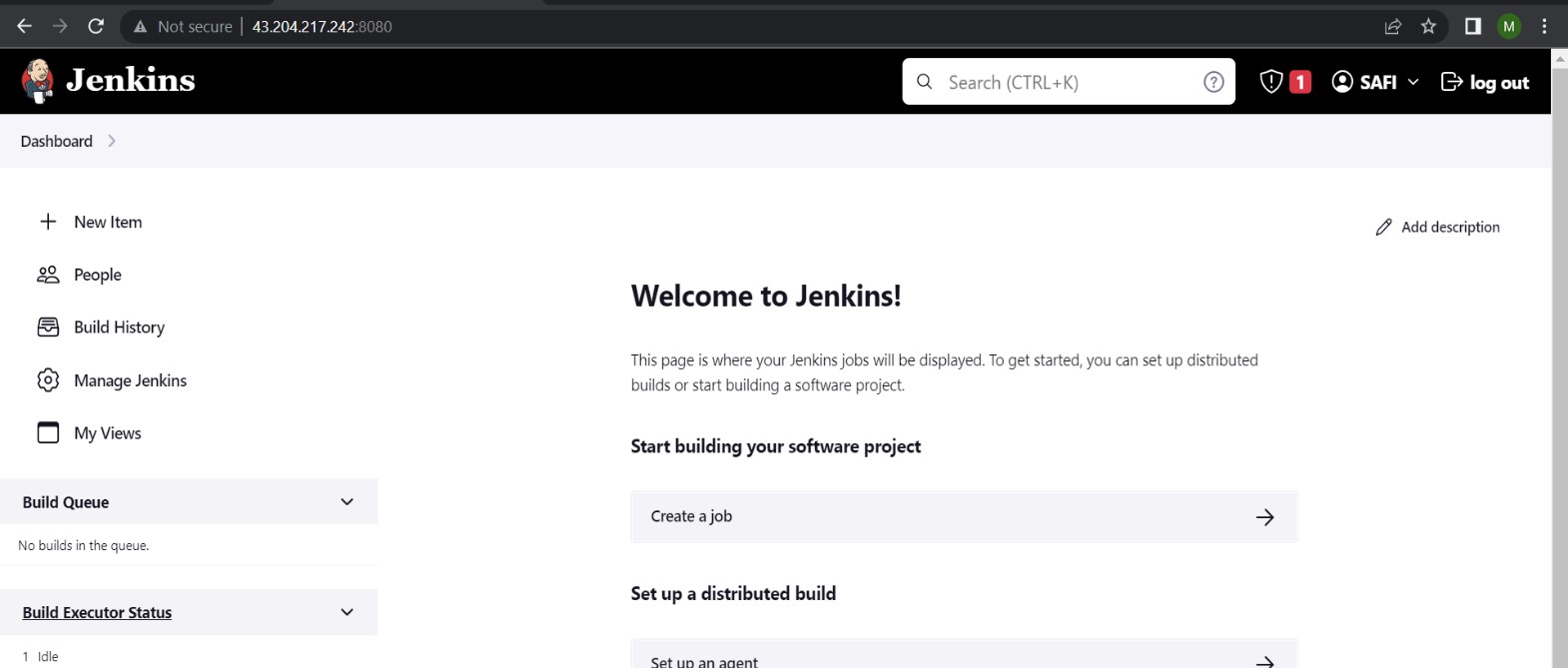
Step 13 – Again, open git bash & paste above password link into git bash as command follows:- **$sudo cat /var/lib/jenkins/secrets/initialAdminPassword**

Step 14 – Copy the password displayed in git bash into Jenkins login page password box.

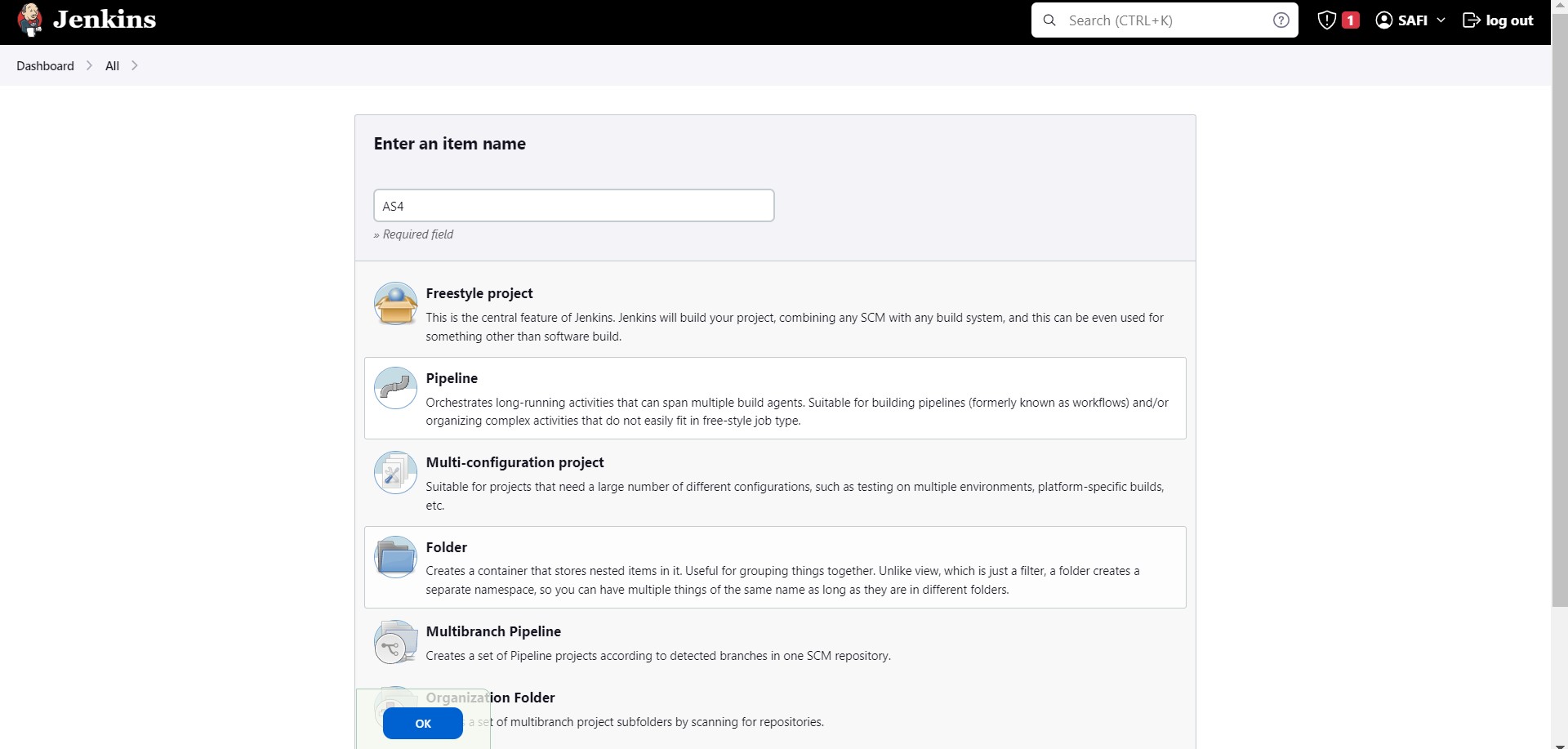


Step 15 – Install Suggested Plugins → Set User Id → User name → Password → E-mail.

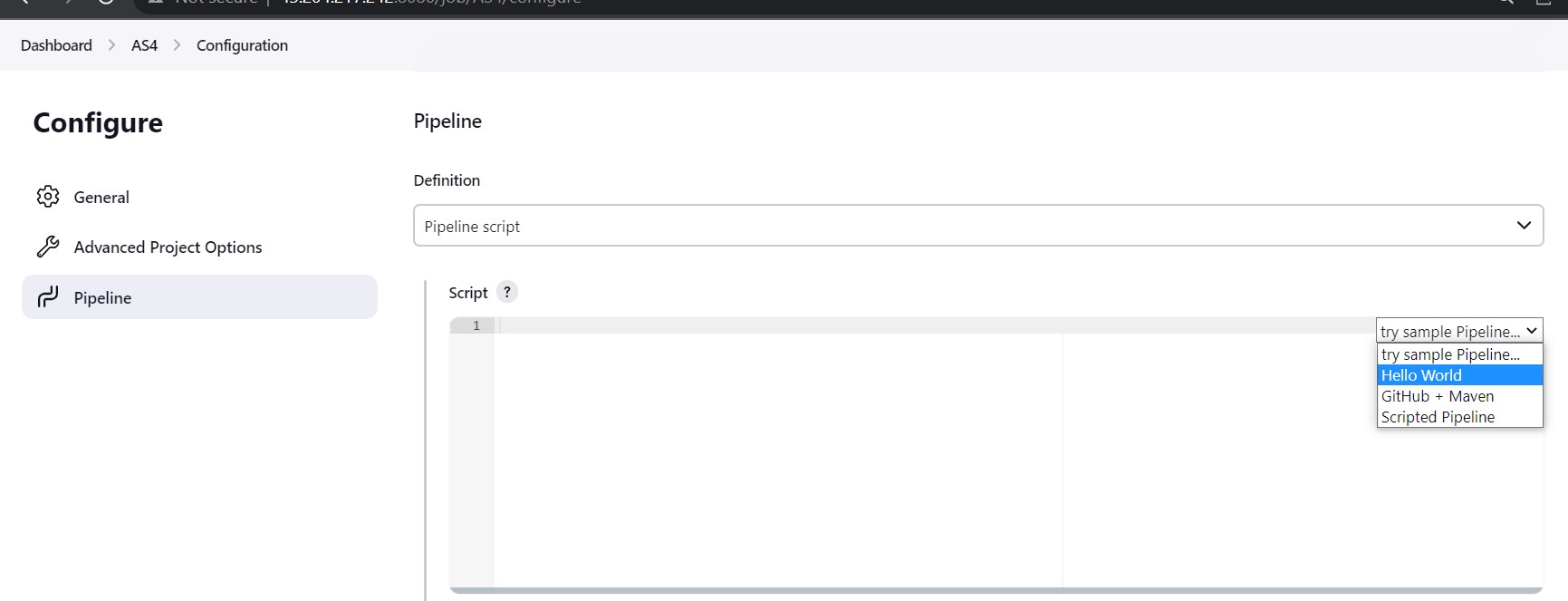
Step 16 – At Jenkins dashboard page → “+New Item” →



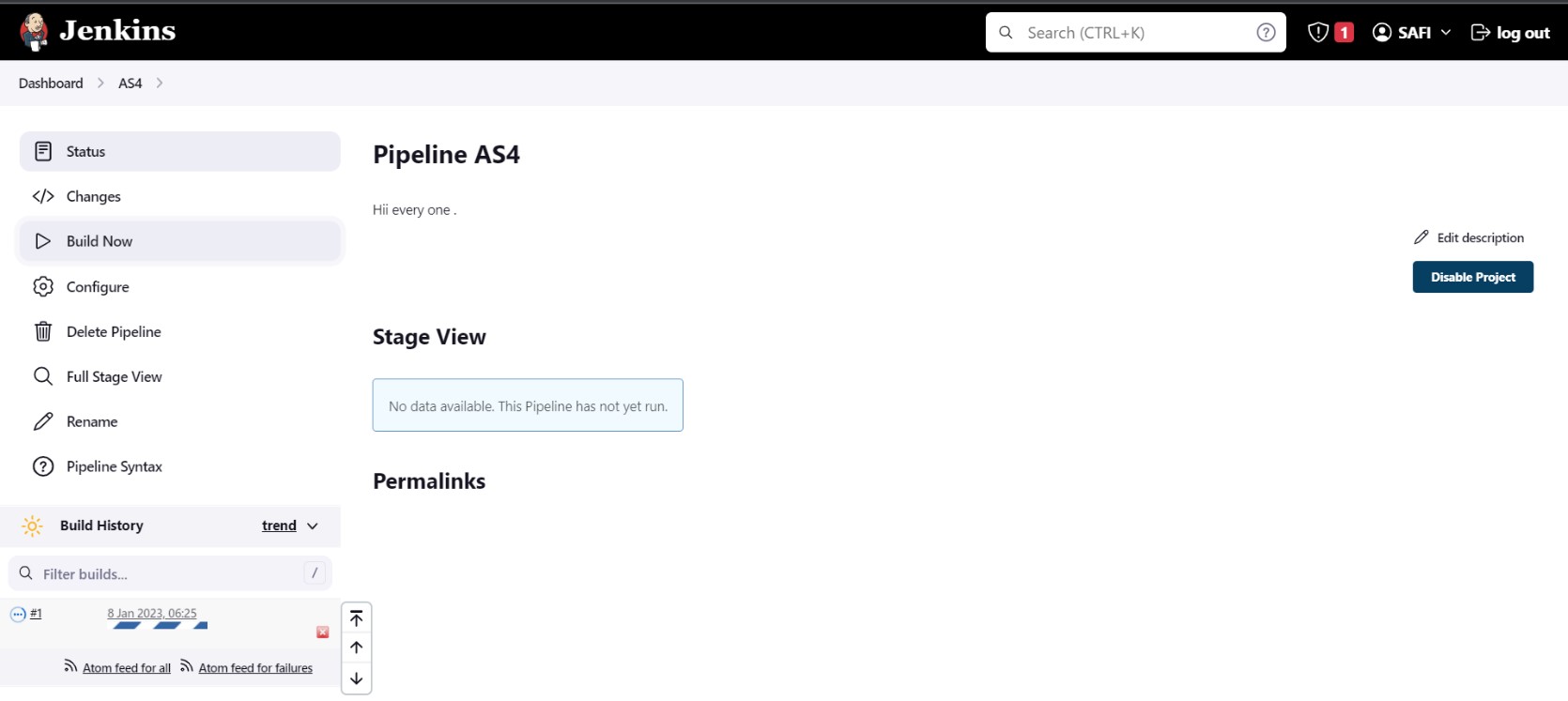
Step 17 Enter item name → select “Pipeline” below → Ok.



Step 18 – Go for Pipeline → select Hello World option from drop down list. → Apply → Save.



Step 19 Now re-direct for dashboard page → select “Build-now” option. → building process is on Step 20 – Right click on building option → Click console output.

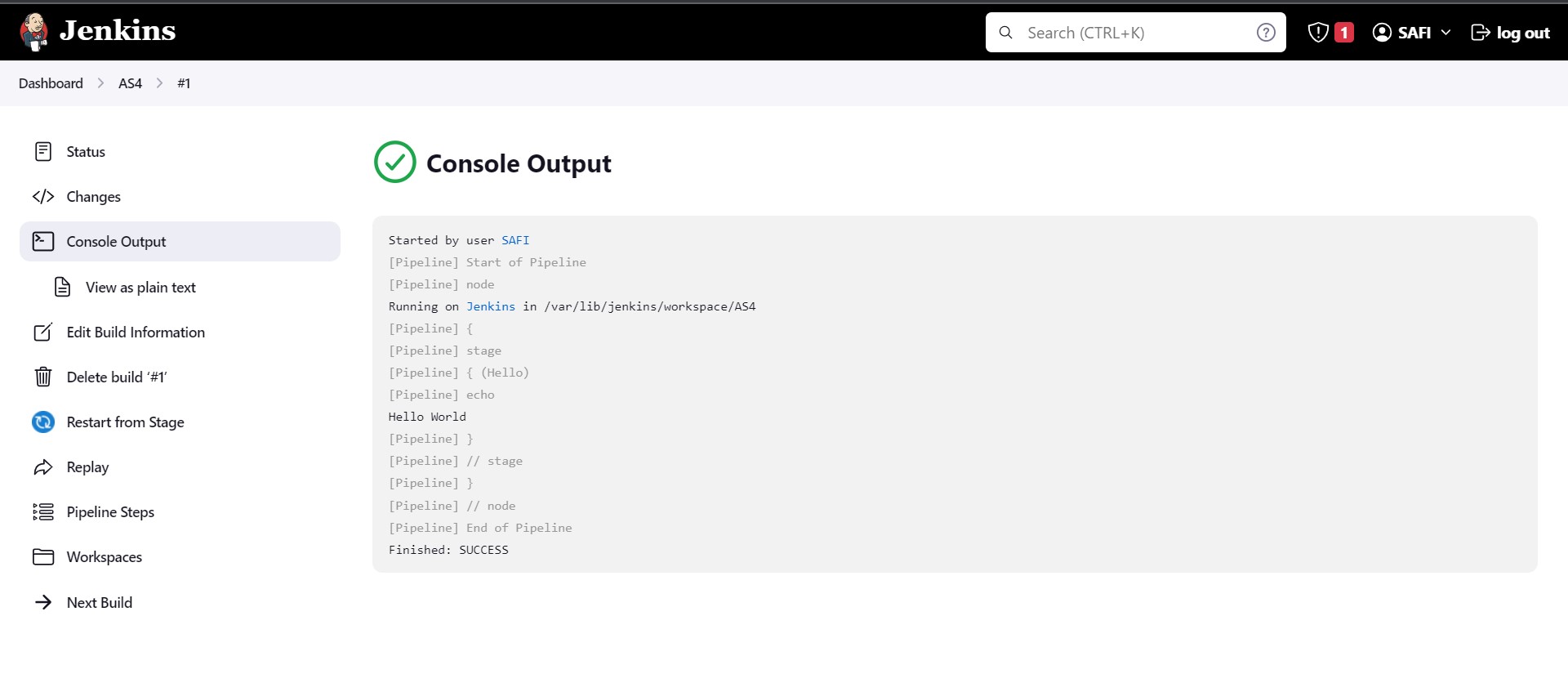


**OUTPUT:**

Step 21

–

Terminate the server.



**12. Create a login page with Java script & its function.**

**<!DOCTYPE html>**

**<html lang="en">**

**<head>**

**<meta charset="UTF-8">**

**<meta http-equiv="X-UA-Compatible" content="IE=edge">**

**<meta name="viewport" content="width=device-width, initial-scale=1.0">**

**<title>Document</title>**

**<style> div{margin: 10px;padding: 5px;}**

**</style>**

**<script> function getamt()**

**{**

**debugger; var s=document.getElementById("source"); var sr=s.options[s.selectedIndex].text; var d=document.getElementById("des"); var ds=d.options[d.selectedIndex].text; if(sr==ds)**

**{**

**alert("Source City & Desination City Same!!!");**

**} else**

**{**

**var dsv=d.options[d.selectedIndex].value;**

**var ns=document.getElementById("txtSeat").value; var tm=parseInt(ns)\*parseInt(dsv); document.getElementById("txtAmt").value=tm;**

**}**

**}**

**</script>**

**</head>**

**<body>**

**<form>**

**<div>**

**<label>Select Source City</label>**

**<select id="source">**

**<option>--Select--</option>**

**<option>Mysuru</option>**

**<option>Bangalore</option>**

**<option>Kerala</option>**

**</select>**

**</div>**

**<div>**

**<label>Select Desination City</label>**

**<select id="des">**

**<option>--Select--</option>**

**<option value="150">Mysuru</option>**

**<option value="150">Bangalore</option>**

**<option value="250">Kerala</option>**

**</select>**

**</div>**

**<div>**

**<label>Enter No of Seats</label>**

**<input type="number" id="txtSeat" />**

**</div>**

**<div>**

**<label>Total Amount</label>**

**<input type="text" id="txtAmt" readonly />**

**</div>**

**<div>**

**<input type="button" value="Get Amount" onclick="getamt()"/>**

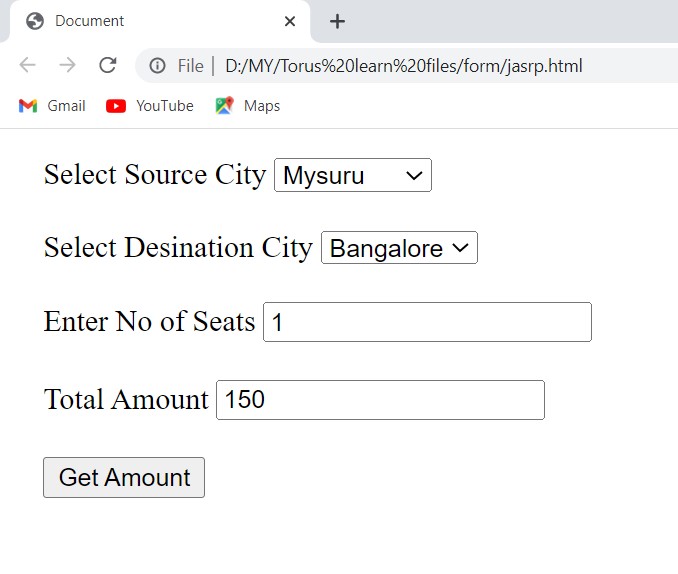
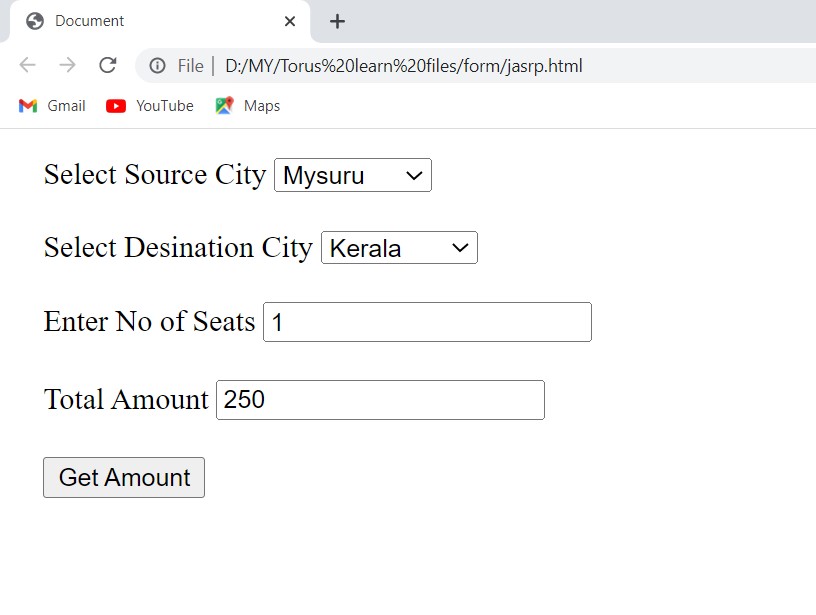
**</div>**

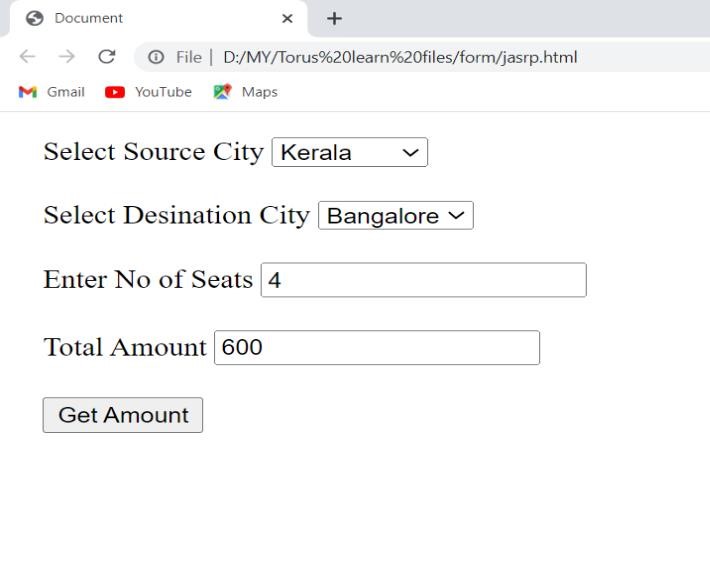
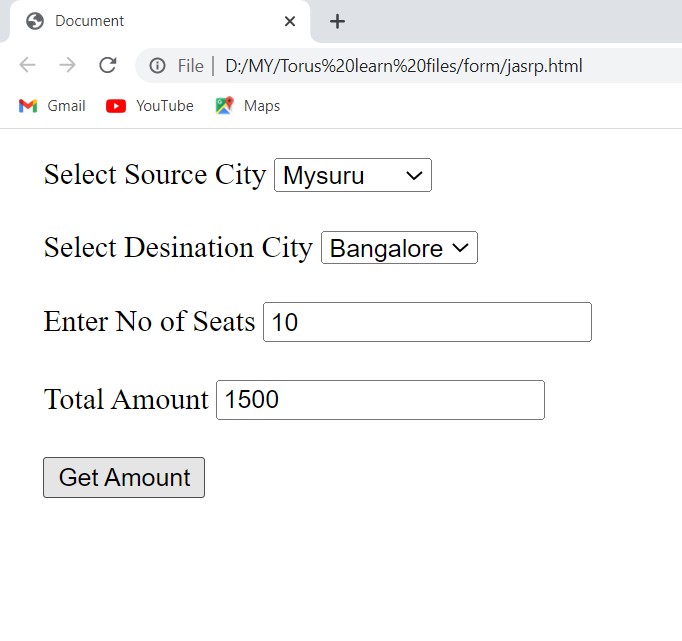
**</form>**

**</body>**

**</html>**

**OUTPUT:**



**13. Demonstrate installation of Apache2 in ubuntu server & perform copying of file from local machine to Server.**

→**Launching Ubuntu Server**

Step 1 - Firstly login into AWS

Step 2 – Click on services→Compute→EC2 server→ Launch instance.

Step 3 – Name server→ select any OS(‘Ubuntu’)→generate a “key-pair”→Launch instance.

Step 4 – go to Security → Security groups →Edit inbound rules → add rule.

Step 5

–

Custom type ‘HTTP’

→

source IPv4

→

Save Rules

Step

6

–

Go to Instance

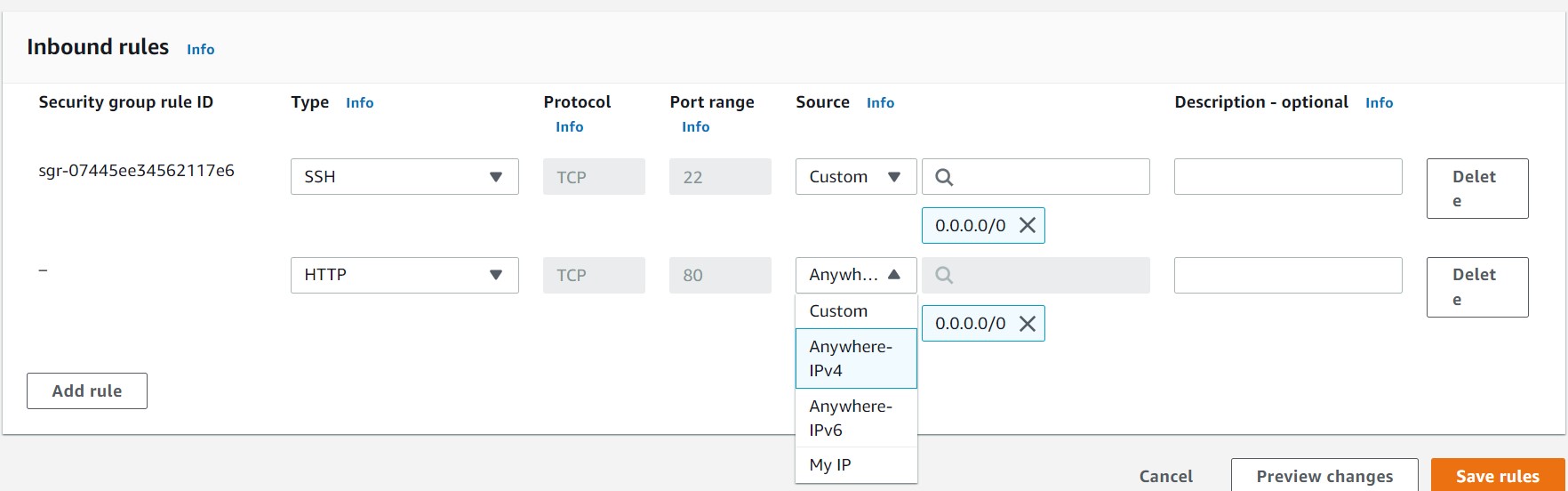
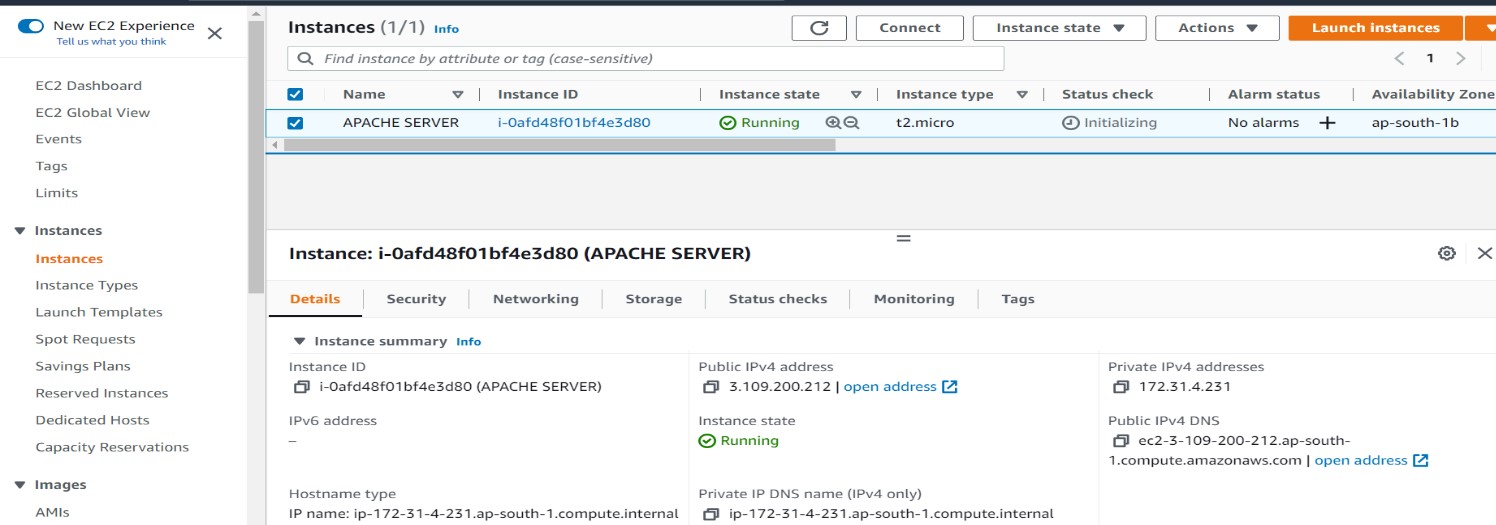
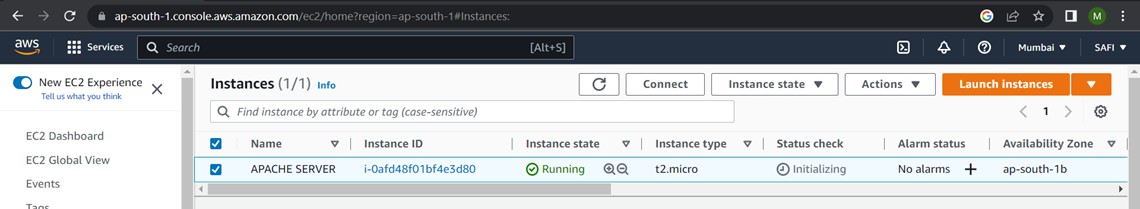
→

Connect

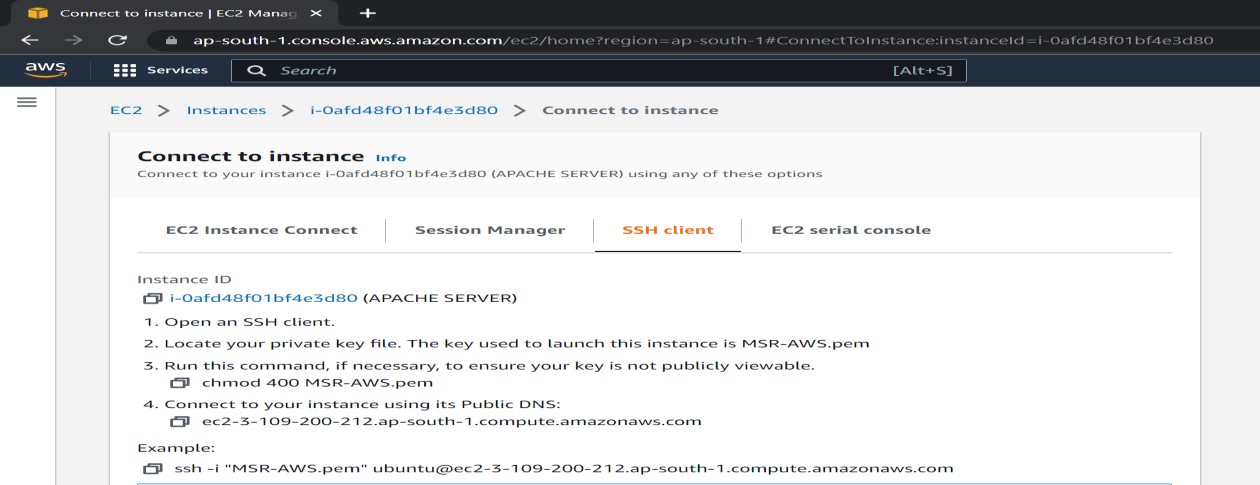
→

SSH Client

→



Step 7 - Copy Example link and paste it in gti bash



Step 8 – Open git bash → paste the above copied link to login into server.

**$sudo apt-get update**

**$sudo apt-get apache2**

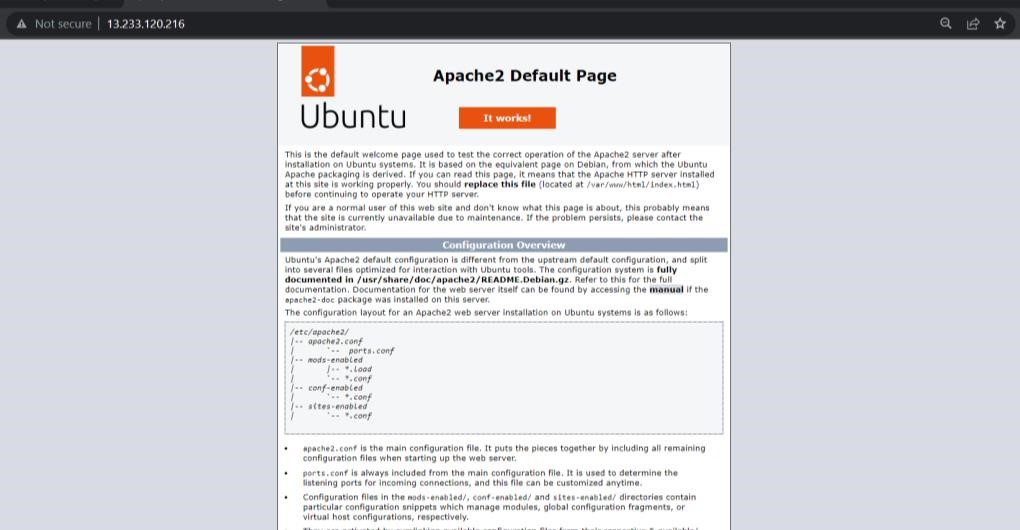
**$sudo apt-get update**

**$sudo service apache2 status** {active (running)}.

**$ cd /var/www/html/** {Default pathway}

**$ls {**index.html <inbuild file>}

Step 9 – Go to AWS → copy Public IP → paste it in new tab.



**Note**:<This is the Apache 2 default page which would be displayed & its made up of index.html file>

Step 10 – Now delete the index.html file’s containt program

Step 11- Go to git bash

**$sudo rm -f index.html**

**$ls** {No file would be displayed because we have deleted it above.}

**$cd .. {**to change directory**}**

**$exit** {exit from the login server}

Step 12 – Now copy file pathway of any html file in your local system → Open git bash.

**$ scp -i "MSR-AWS.pem" "C:\Users\Lenovo\OneDrive\Desktop\HTML FILES\prg2.html" ubuntu@ec213-233-120-216.ap-south-1.compute.amazonaws.com:/home/ubuntu/** <The copied file name would be displayed below>.

Step 13 – Now again login into ubuntu server & follow the code  **$ls**

**$sudo vi prg2.html** {to view the html code of file},

**$ mv prg2.html index.**html {to rename/move the current file},

**$ sudo cp index.html /var/www/html/** {to copy “index.html” file into “/var/www/html/” pathway} **$ls** {“index.html” would be shown}

Step 14 – Now copy public IP → paste it in new Tab {The index.html file would be displayed}

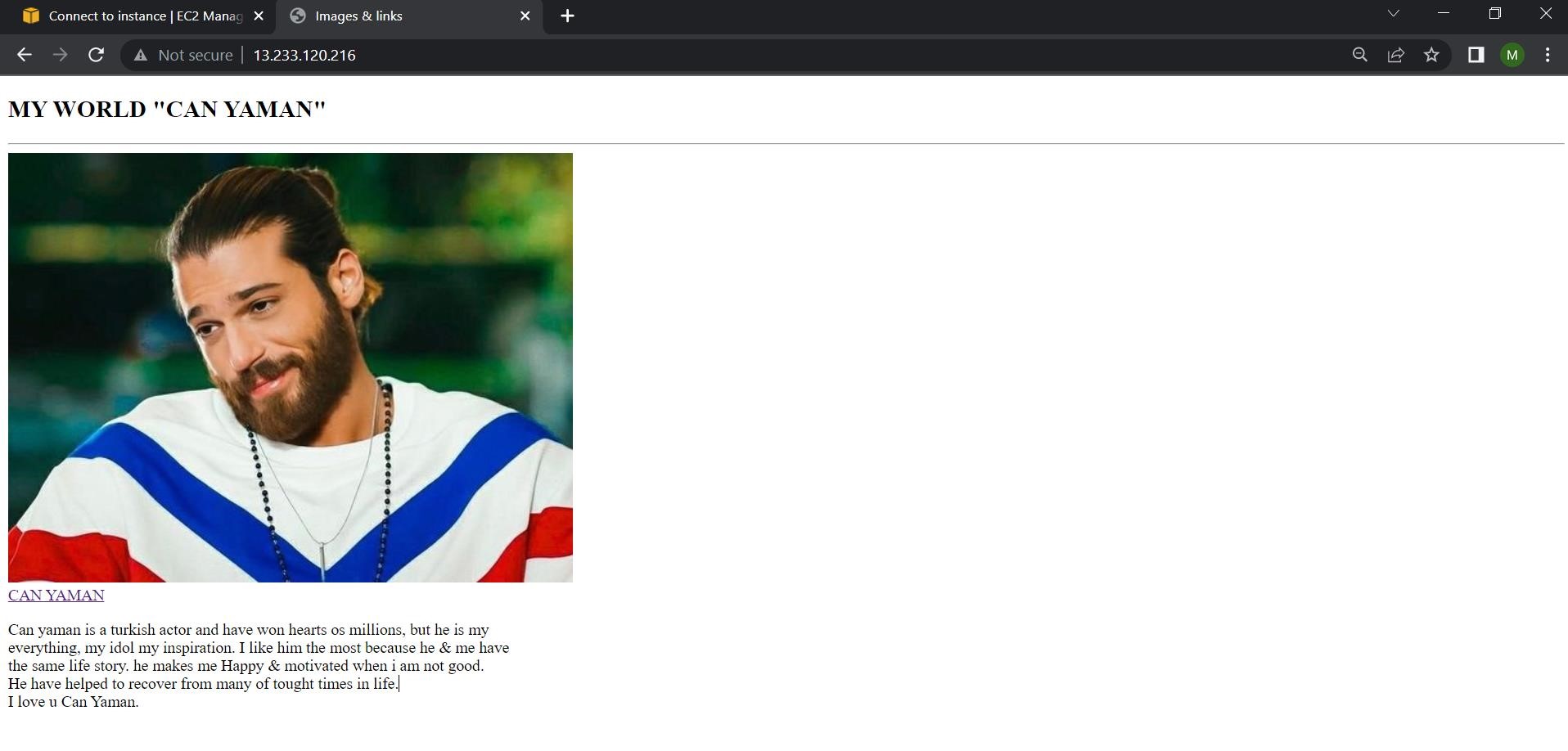
**OUTPUT:**

Step 15

–

Terminate the server

.



**14. Create simple hello world application using type script.**

Step 1 : Install TypeScript into the system

**npm install typescript --save-dev**

Step 2 : Check the TypeScript compiler version i.e **tsc --version**

Step 3 : Create a Html file , Index.ts file and link the Index.js file using script tag assuming it is already existed.

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <head>  <meta charset="UTF-8">  <meta http-equiv="X-UA-Compatible" content="IE=edge">  <meta name="viewport" content="width=device-width, initial-scale=1.0">  <title>Hello World</title>  </head>  <body>    <button onclick="handleClick()">Click here</button>  <script src="index.js"></script>  </body>  </html> |

Step 4 : write a function that prints hello world in **Index.ts** file.

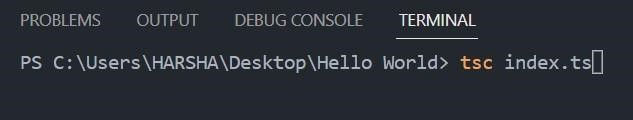
function handleClick(){

let message:string = "Hello World";

let root = document.createElement('h1'); root.textContent = message; document.body.appendChild(root); }

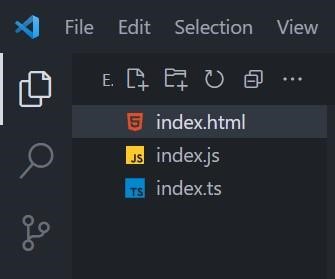
Step 5 : Compile the Index.ts file by opening a new terminal.

Step 6 : Type **tsc index.ts** and hit enter.



Step 7: **”ExecutionPolicy -ExecutionPolicy RemoteSigned”** enter this command in windows power shell/cmd.

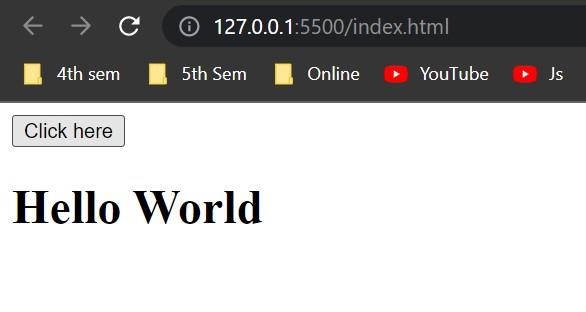
Step 8 : A new Javascript file will be created as Index.js.



Step 9 : Now run the Html file in the browser.

Step 10 : Click the button to display the hello world message.

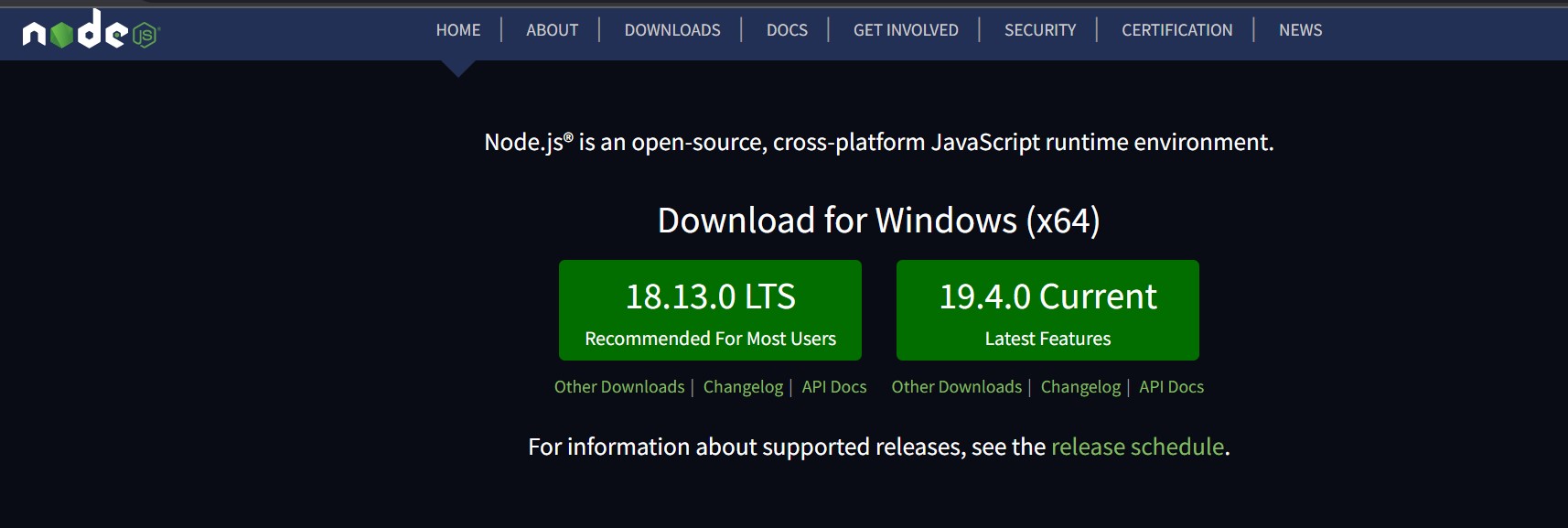
**OUTPUT:**



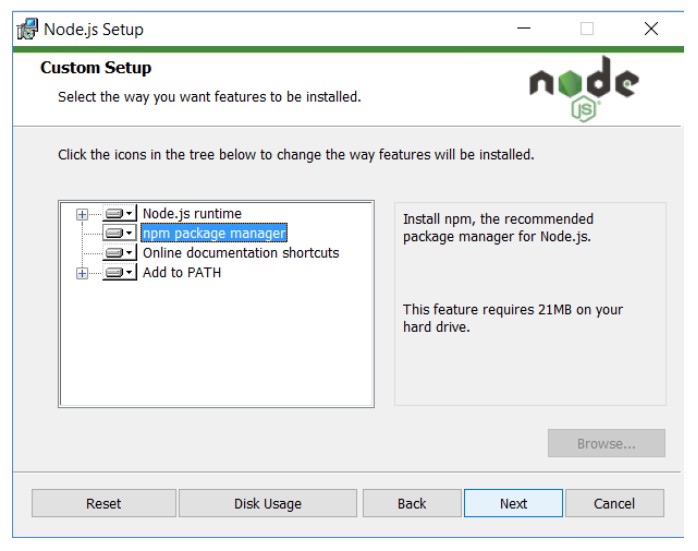
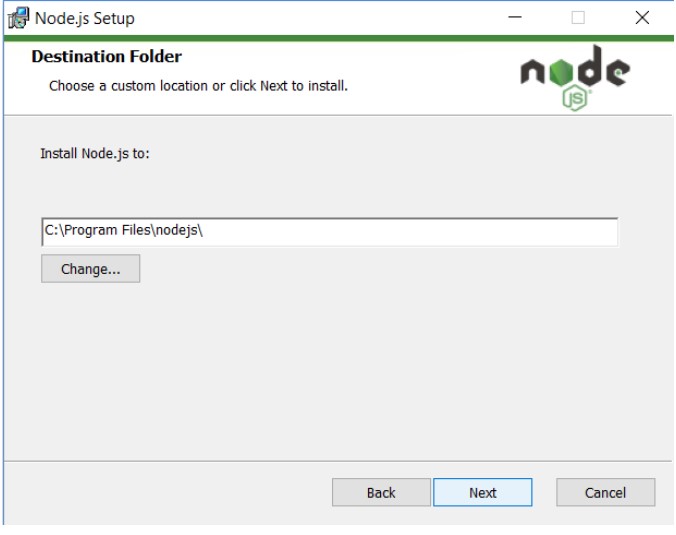
**15. Create a new React app & perform import & export operations**.

**NOTE: - <** You need to download “node js” to run the react component.>

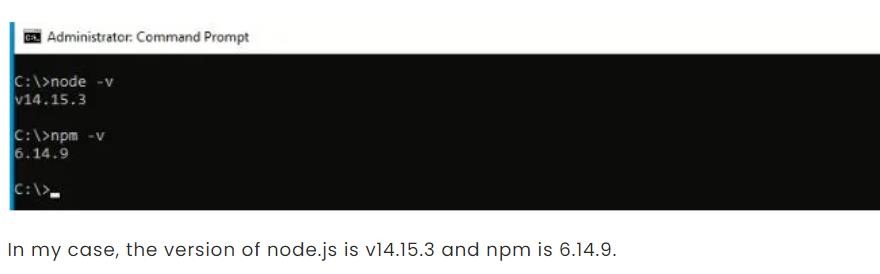
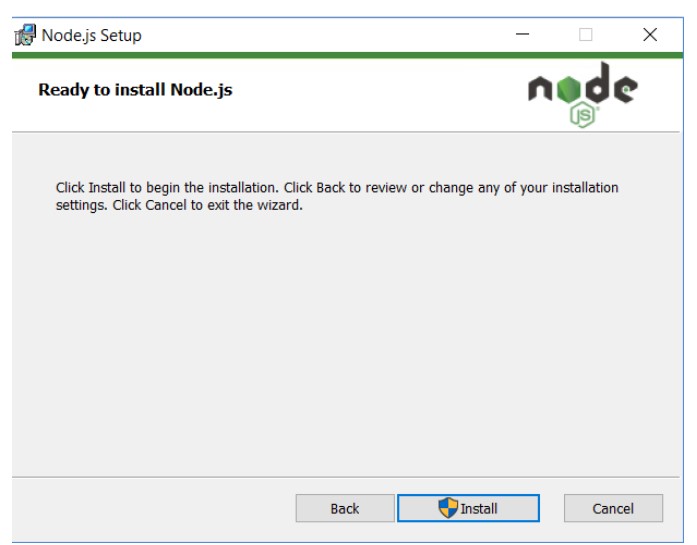
Step1- Go to browser → node js installer → select 18.13.0 LTS version → starts downloading file.



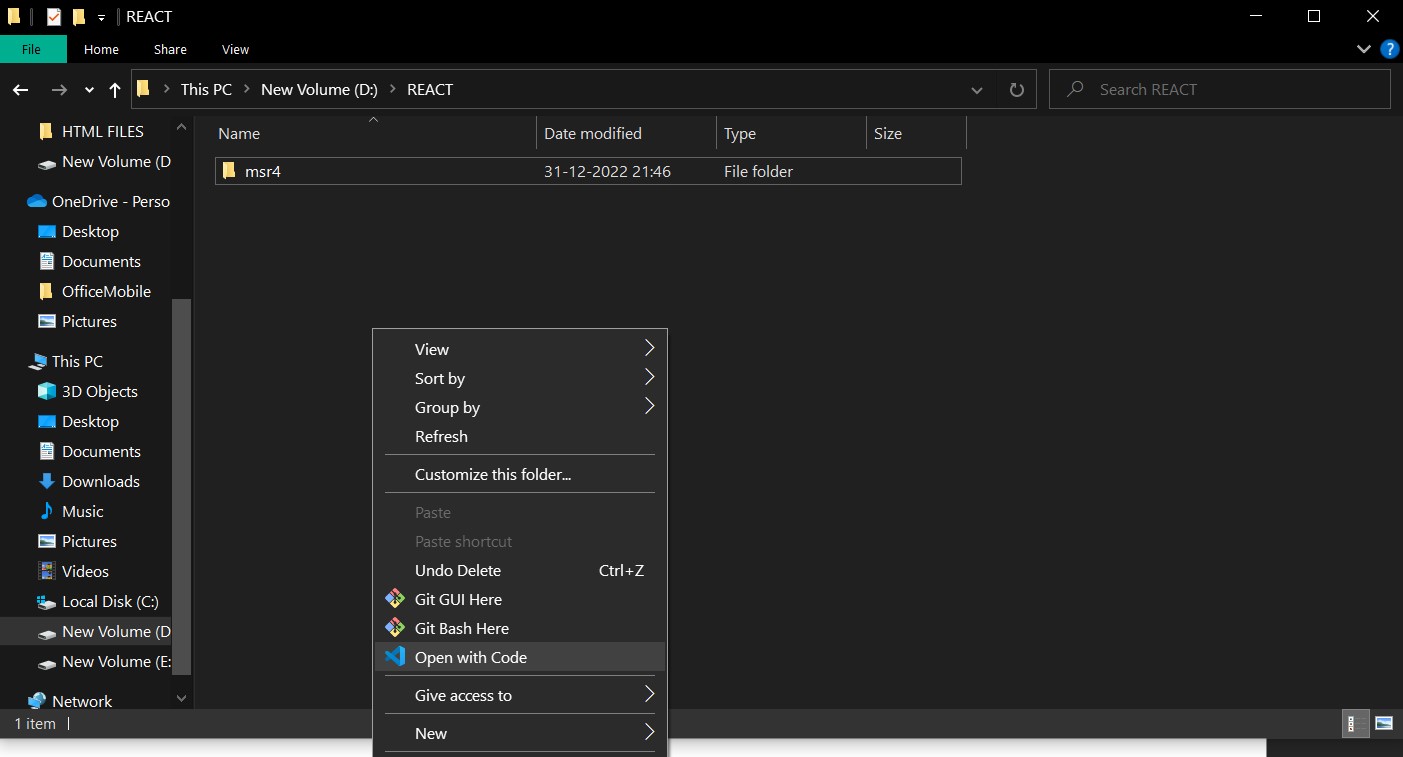
Step 2 – Set up the detail → Next → Next→ Next→ Add to Path →,



Step 3 - click on the Install button so hard! Step 4 - Check Node.js and NPM Version in CMD

;

Step 5 – Create a new Empty folder → open folder & right click → click on open with code



Step 6

–

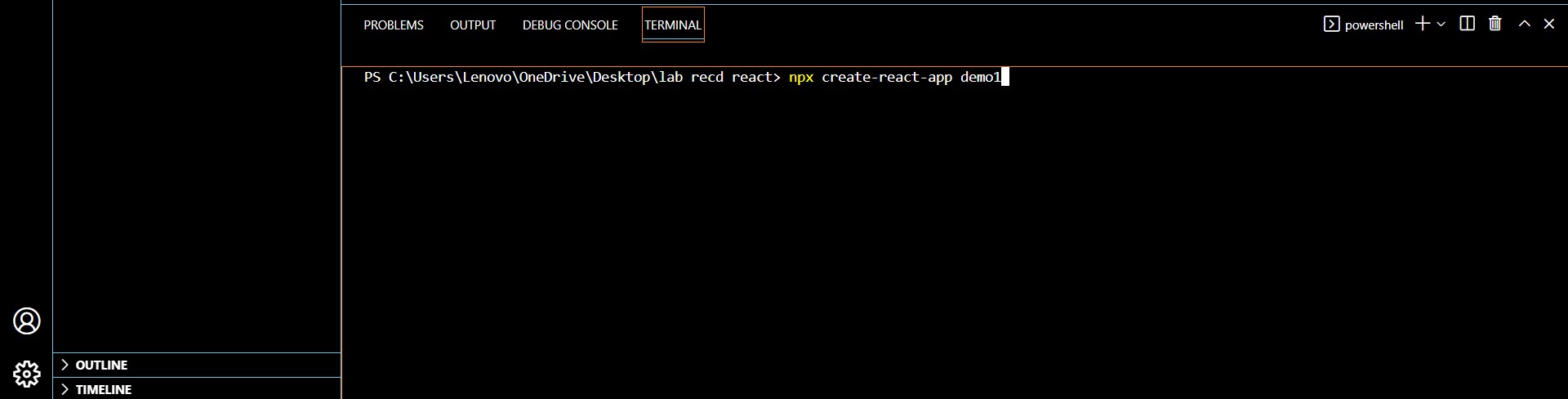
click on terminal

→

open new terminal



Step 7 – Now type **npx create-react-app demo1{**file name**}**



Step 8

–

now type

**cd demo1{**

file name

**}**

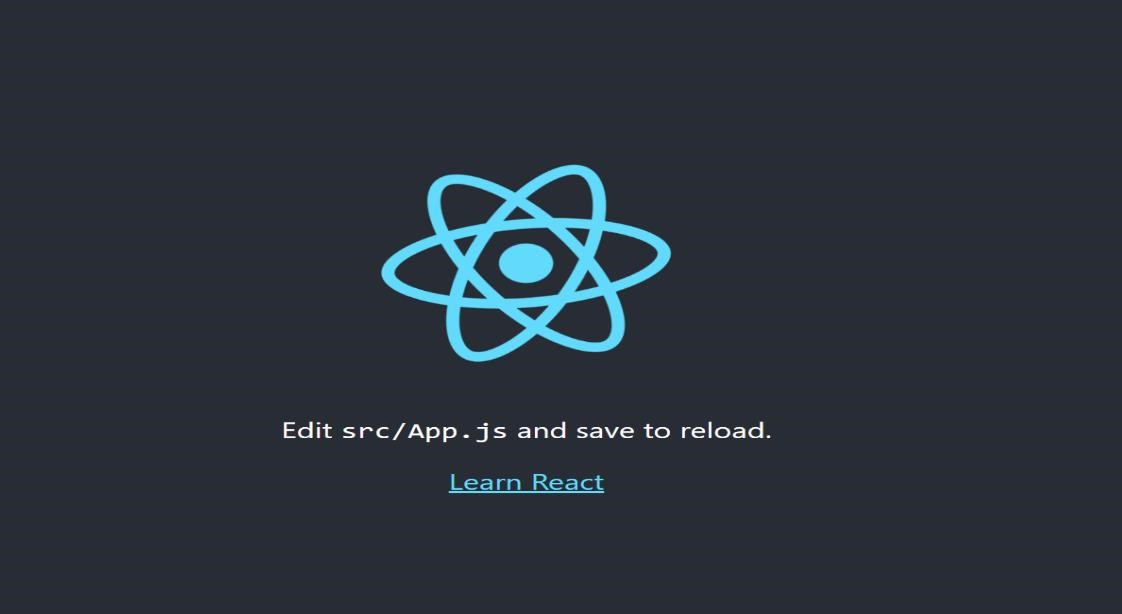
→

**npm start.**

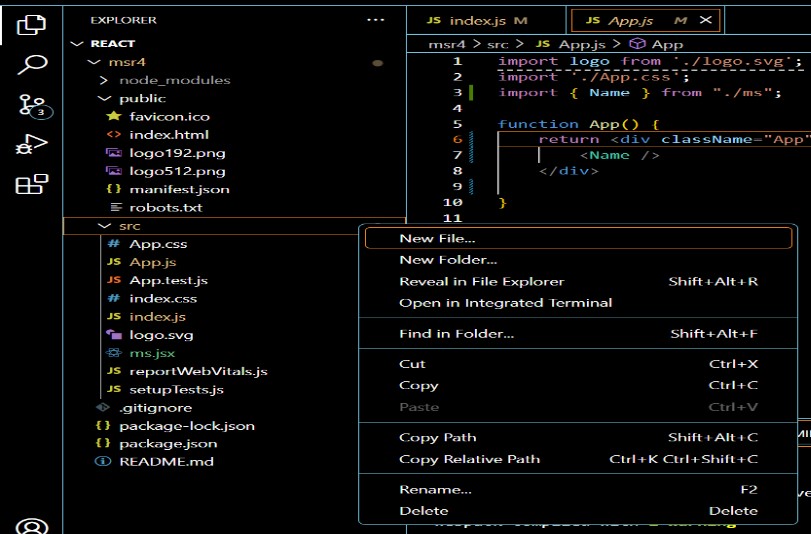
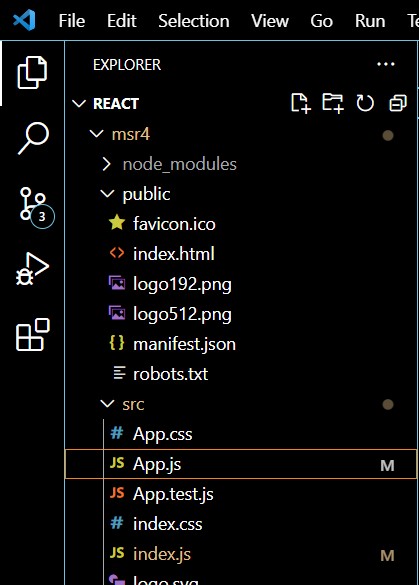
{

React app page gets arrive

}



Step 9 – open new file inside “src” folder → right click → New file→ create a new file **“form.jsx”**



Step 10 – Write code in in “**form.jsx”** →include import & export operation.

i**mport React from "react"; import ReactDOM from "react";**

**function Form(){**

**const [count,setCount]=React.useState(0); function increment(){ setCount(count + 1);**

**}**

**function decrement(){**

**setCount(count - 1);**

**} function reset(){ setCount(0);**

**}**

**return(**

**<div>**

**<button onClick={increment}> + </button>**

**<h1>{count}</h1>**

**<button onClick={decrement}> - </button>**

**<button onClick={reset}> Reset </button>**

**</div>**

**);**

**}**

**export{Form};**

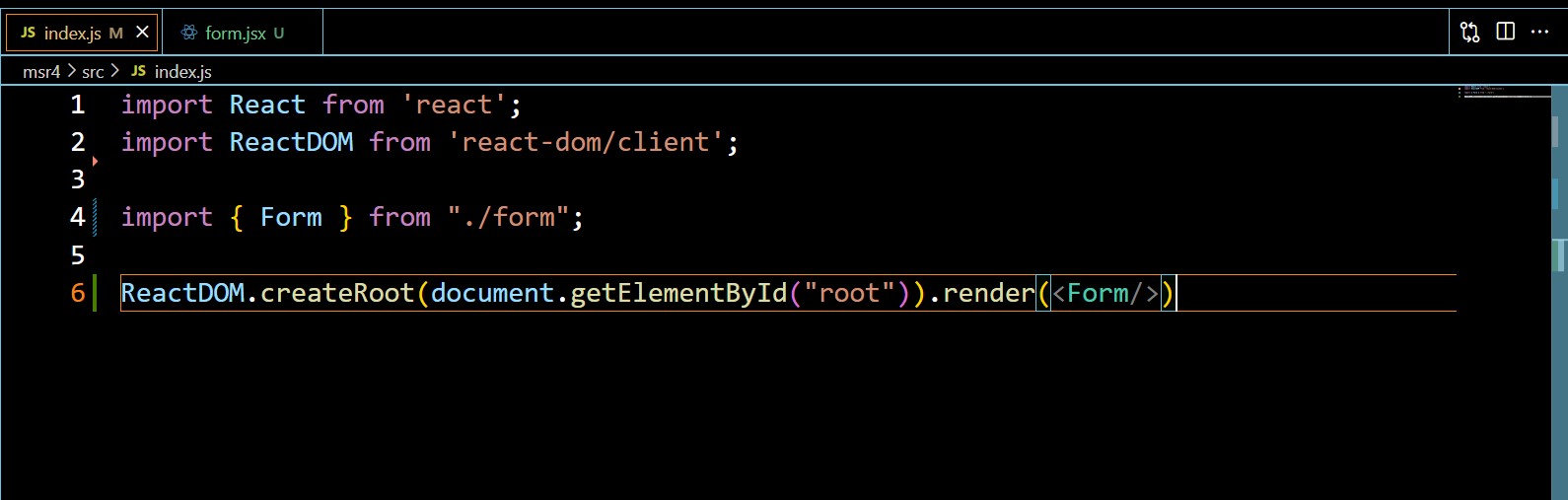


Step 11 – Go to index.js file → type following code there: {include import & export operation}. **import React from 'react';**

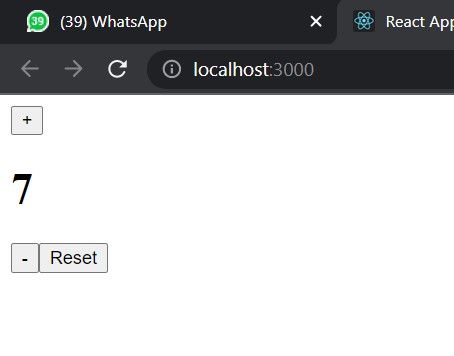
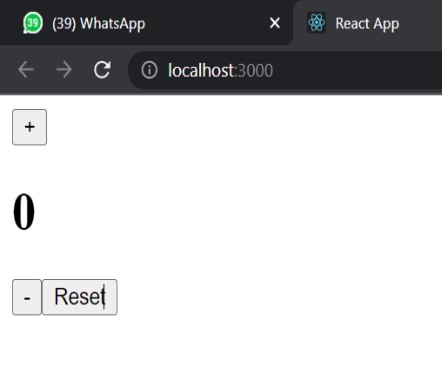
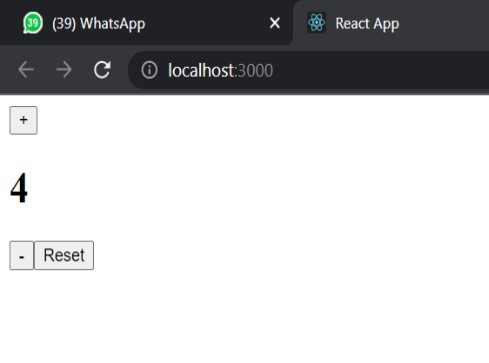
**import ReactDOM from 'react-dom/client';**

**import { Form } from "./form";**

**ReactDOM.createRoot(document.getElementById("root")).render(<Form/>)** Step 12 – Open new terminal → **cd msr4**{file name} → **npm start**



**OUTPUT:**



**16. Demonstrate the following operations in docker concepts:**

**a. Creating a docker file:**

Step 1 – Firstly, launch a ubuntu server using AWS, in git.

Step 2 – Follow the code given below

**$sudo apt-get update**

**$sudo apt-get install docker.io**

**$sudo service docker status**

**$sudo su {**typing **su** switches to the root user**}**

**# mkdir demo1 {**file name**}**

**# cd demo1 {**file name**}**

**# touch Dockerfile {**Create a touch file & Dockerfile -- D should be capital**} # vim Dockerfile.**

Step – 3 Now write this command inside Dockerfile**.**

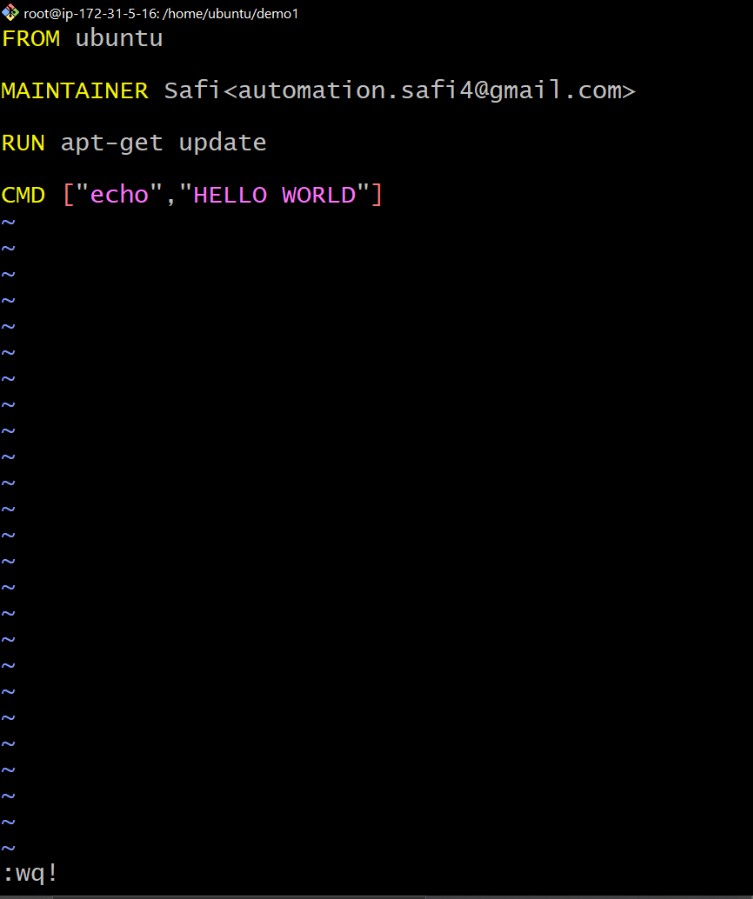
**FROM ubuntu**

**MAINTAINER safi<automation.safi4@gamil.com>**

**RUN apt-get update**

**CMD [“echo”, “HELLO WORLD”]**

**:wq! {**to exit out Dockerfile**}**



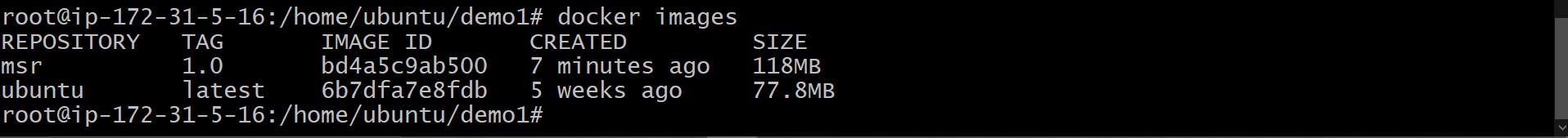
**16 b. Build a docker image with docker file.**

Step 1 – Firstly create a Dockerfile as per above concerned procedure.

Step 2 – follow this code in git

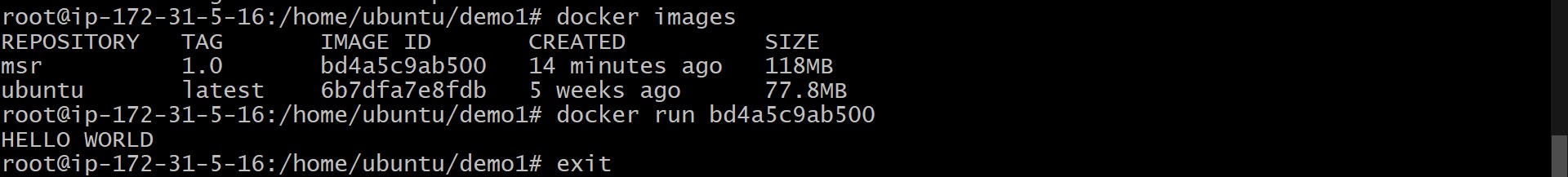
# # docker build -t msr:1.0 . {“msr” image name}

Step 4 – All the program written inside Dockerfile would get updated & run **# docker images**



Step 5 - Now at last run the docker image.

# # docker run bd4a5c9ab500{image id}



**OUTPUT:**

