PROJECT - DRIVE YOUR WAY

This Application Consist of:
• Home Page
• Login Page
• Register Page
• Subscription Plans and Pricing Page
• Car Categories Page
• Shortlisted Products Page
• Admin Page
• Adding filters in the search option

Core concepts used in project
• Selenium Library
• Eclipse IDE
• TestNG Library
• Maven

Project is completed using 4 sprint

Sprint 1 :
Backend Part :
I. Create a spring boot starter project in eclipse
II. Write all java code for Drive Your Way project . Create bean class, controller class, repository class and service class and then edit application.java class.
III. Start the project on port 9090

Sprint 2 :
Frontend Part :

I. Create an angular project using VSCode.
II. Create add-product, admin-product-retrieve, admindashboard, login, signup, userdashboard component using ng g c command.
III. Write code in components.
IV. Create login.ts and product.ts file.
V. Update app module.ts file.
VI. Create one karma.conf.js file.
VII. Run the project using ng serve command.

Sprint 3 :
Testing Part :

I. Convert the spring boot starter project into testng that will create a testng.xml file.
II. Create one testng class to run all the test cases.
III. Test all the scenario.
IV. Check the result status in testng result part.
V. Create testng report file.

Sprint 4 :

DevOps
I. Connect to ec2 instance.
II. Check java and git install or not.
III. Create separate repository and upload the spring boot project.
IV. Open jenkins and create a new job using freestyle project.
V. Update source code management and build environment and build the project
VI. And finally push the code in github repository.

SOURCE CODE FOR DRIVE YOUR WAY

BACK-END APP:

```
package com;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;
import org.springframework.boot.autoconfigure.domain.EntityScan;
import
org.springframework.data.jpa.repository.config.EnableJpaRepositories;

@SpringBootApplication(scanBasePackages = "com")
@EntityScan(basePackages = "com.onlineshop.bean")
@EnableJpaRepositories(basePackages = "com.onlineshop.repository")
public class MyAppApplication {

public static void main(String[] args) {
SpringApplication.run(MyAppApplication.class, args);
System.out.println("Server running on port number 9090");
}

}
```

LOGIN.JAVA

```
package com.onlineshop.bean;

import javax.persistence.Column;
import javax.persistence.Entity;
import javax.persistence.Id;

@Entity
public class Login {
@Id
private String emailid;
private String password;
@Column(name = "typeofuser")
private String typeOfUser;
public String getEmailid() {
return emailid;
}
public void setEmailid(String emailid) {
this.emailid = emailid;
}
public String getPassword() {
return password;
}
public void setPassword(String password) {
this.password = password;
}
public String getTypeOfUser() {
return typeOfUser;
}
public void setTypeOfUser(String typeOfUser) {
this.typeOfUser = typeOfUser;
}
@Override
```

```
public String toString() {
return "Login [emailid=" + emailid + ", password=" + password + ",
typeOfUser=" + typeOfUser + "]";
}

}
```

PRODUCT.JAVA

```
package com.onlineshop.bean;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;

@Entity
public class Product {
@Id
@GeneratedValue(strategy = GenerationType.IDENTITY)            // auto_generate
private int pid;
private String pname;
private float price;
private String url;
public int getPid() {
        return pid;
}
public void setPid(int pid) {
        this.pid = pid;
}
public String getPname() {
        return pname;
}
public void setPname(String pname) {
        this.pname = pname;
}
public float getPrice() {
        return price;
}
public void setPrice(float price) {
        this.price = price;
}
public String getUrl() {
        return url;
}
public void setUrl(String url) {
        this.url = url;
}
@Override
public String toString() {
```

```
	return "Product [pid=" + pid + ", pname=" + pname + ", price=" + price + ",
url=" + url + "]";
}

}
```

LOGINCONTROLLER.JAVA

```
package com.onlineshop.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.MediaType;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.onlineshop.bean.Login;
import com.onlineshop.service.LoginService;

@RestController
@RequestMapping("login")
@CrossOrigin
public class LoginController {


	@Autowired
	LoginService loginService;


	@PostMapping(value = "signIn",consumes =
MediaType.APPLICATION_JSON_VALUE)
	public String signIn(@RequestBody Login login) {
		System.out.println("I cam here");
		return loginService.signIn(login);

	}

	@PostMapping(value = "signUp",consumes =
MediaType.APPLICATION_JSON_VALUE)
	public String signUp(@RequestBody Login login) {
		System.out.println(login);
		return loginService.signUp(login);
	}
}
```

PRODUCTCONTROLLER.JAVA

```
package com.onlineshop.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.MediaType;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PatchMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.onlineshop.bean.Product;
import com.onlineshop.service.ProductService;

@RestController
@RequestMapping("product")
@CrossOrigin
public class ProductController {

@Autowired
ProductService productService;
@PostMapping(value = "storeProduct",consumes =
MediaType.APPLICATION_JSON_VALUE)
public String storeProduct(@RequestBody Product product) {
return productService.storeProduct(product);
}
@PatchMapping(value = "updateProduct",consumes =
MediaType.APPLICATION_JSON_VALUE)
public String updateProduct(@RequestBody Product product) {
return productService.updateProduct(product);
}
@GetMapping(value="findAllProduct",produces =
MediaType.APPLICATION_JSON_VALUE)
public List<Product> getAllProduct() {
return productService.getAllProducts();
}
@GetMapping(value="findProductByPrice/{price}",produces =
MediaType.APPLICATION_JSON_VALUE)
public List<Product> findProductByPrice(@PathVariable("price") float
price) {
return productService.findProductByPrice(price);
}
@GetMapping(value="findAllProduct/{pid}")
public String findProductById(@PathVariable("pid") int pid) {
return productService.findProductById(pid);
}
@DeleteMapping(value="deleteProduct/{pid}")
public String deleteProductUsingId(@PathVariable("pid") int pid) {
return productService.deleteProduct(pid);
}
}
```

LOGINREPOSITORY.JAVA

```
package com.onlineshop.repository;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;

import com.onlineshop.bean.Login;

@Repository
public interface LoginRepository extends JpaRepository<Login,
String>{

}
```

PRODUCTREPOSITORY.JAVA

```
package com.onlineshop.repository;

import java.util.List;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.data.jpa.repository.Query;
import org.springframework.data.repository.query.Param;
import org.springframework.stereotype.Repository;

import com.onlineshop.bean.Product;

@Repository
public interface ProductRepository extends JpaRepository<Product,
Integer>{

//JPQL
@Query("select p from Product p where p.price > :price")
public List<Product> findProductByPrice(@Param("price") float price);
}
```

LOGINSERVICE.JAVA

```
package com.onlineshop.service;

import java.util.Optional;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.onlineshop.bean.Login;
import com.onlineshop.repository.LoginRepository;

@Service
public class LoginService {
@Autowired
LoginRepository loginRepository;
public String signIn(Login login) {
Optional<Login> result = loginRepository.findById(login.getEmailid());
```

```
if(result.isPresent()) {
Login ll = result.get();
if(ll.getPassword().equals(login.getPassword())) {
if(login.getTypeOfUser().equals(ll.getTypeOfUser()) &&
login.getTypeOfUser().equals("admin")) {
return "Admin sucessfully login";
}else if(login.getTypeOfUser().equals(ll.getTypeOfUser()) &&
login.getTypeOfUser().equals("user")){
return "User successfully login";
}else {
return "Invalid details";
}
}else {
return "InValid password";
}
}else {
return "InValid emailId";
}
}
public String signUp(Login login) {
Optional<Login> result = loginRepository.findById(login.getEmailid());
if(result.isPresent()) {
return "Email Id alreay exists";
}else {
if(login.getTypeOfUser().equals("admin")) {
return "You can't create admin account";
}else {
loginRepository.save(login);
return "Account created successfully";
}
}
}
}
```

PRODUCTSERVICE.JAVA

```
package com.onlineshop.service;

import java.util.List;
import java.util.Optional;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.onlineshop.bean.Product;
import com.onlineshop.repository.ProductRepository;

@Service
public class ProductService {

@Autowired
ProductRepository productRepository;
public String storeProduct(Product product) {
productRepository.save(product);
return "Product details stored";
}
public List<Product> getAllProducts() {
return productRepository.findAll();
}
```

```
public String findProductById(int pid) {
Optional<Product> result = productRepository.findById(pid);
if(result.isPresent()) {
Product p = result.get();
return p.toString();
}else {
return "Product not present";
}
}
public List<Product> findProductByPrice(float price){
return productRepository.findProductByPrice(price);
}
public String deleteProduct(int pid) {
Optional<Product> result = productRepository.findById(pid);
if(result.isPresent()) {
Product p = result.get();
productRepository.delete(p);
return "Product deleted successfully";
}else {
return "Product not present";
}
}
public String updateProduct(Product product) {
Optional<Product> result =
productRepository.findById(product.getPid());
if(result.isPresent()) {
Product p = result.get();
p.setPrice(product.getPrice());
p.setUrl(product.getUrl());
productRepository.saveAndFlush(p);
return "Product updated successfully";
}else {
return "Product not present";
}
}
}
```

FRONTEND-APP:

APP.COMPONENET.HTML

```
<!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>
</head>
<body>
<div align="center" class="a">
<h1>Drive Your way</h1>
<!-- <h2> Admin Login</h2> -->
<!-- <hr/> -->
<router-outlet></router-outlet>
</div>
</body>
</html>
```

APP.COMPONENET.TS

```
import { Component } from '@angular/core';
@Component({
selector: 'app-root',
templateUrl: './app.component.html',
styleUrls: ['./app.component.css']
})
export class AppComponent {
title = 'frontend-app';
}
```

APP.COMPONENET.CSS

```
body{
    background: rgb(238,174,202);
    background: radial-gradient(circle, rgba(238,174,202,1) 0%, rgba(148,187,233,1) 100%);
    height: 100vh;
    padding: 0px;
    margin: 0px;
    }
    .a{
    padding-top: 10px;
    color: crimson;
}
```

LOGIN.SERVICE.TS

```
import { HttpClient } from '@angular/common/http';
import { Injectable } from '@angular/core';
import { Observable } from 'rxjs';
@Injectable({
providedIn: 'root'
})
export class LoginService {
baseURL:string ="http://localhost:9090/login";
constructor(public http:HttpClient) { }
signIn(login:any):Observable<string> {
return this.http.post(this.baseURL+"/signIn",login,{responseType:"text"});
}
signUp(login:any):Observable<string> {
    return this.http.post(this.baseURL+"/signUp",login,{responseType:"text"});
    }
    }
```

LOGIN.TS

```
// map to entity class or json data.
export class Login {
constructor(public emailid:string,
public password:string,
public typeOfUser:string){}
}
```

PRODUCT.SERVICE.TS

```
import { HttpClient } from '@angular/common/http';
import { Injectable } from '@angular/core';
import { Observable } from 'rxjs';
import { Product } from './product';
@Injectable({
providedIn: 'root'
})
export class ProductService {
baseUrl:string ="http://localhost:9090/product"
constructor(public http:HttpClient) { }
storeProduct(product:any):Observable<string> {
return this.http.post(this.baseUrl+"/storeProduct",product,{responseType:"text"});
```

```
}
updateProduct(product:any):Observable<string> {
    return this.http.patch(this.baseUrl+"/updateProduct",product,{responseType:"text"});
    }
    findAllProduct():Observable<Product[]> {
    return this.http.get<Product[]>(this.baseUrl+"/findAllProduct");
    }
    findAllProductByPrice(price:number):Observable<Product[]> {
    return this.http.get<Product[]>(this.baseUrl+"/findProductByPrice/"+price);
    }
    findAllProductById(pid:number):Observable<string> {
    return this.http.get(this.baseUrl+"/findAllProduct/"+pid,{responseType:"text"});
    }
    deleteProductById(pid:number):Observable<string> {
    return this.http.delete(this.baseUrl+"/deleteProduct/"+pid,{responseType:"text"});
    }
    }
```

LOGIN.COMPONENET.HTML

```
<!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>
</head>
<body>
<div class="admin">
<div class="signIn">
<h2>Login Page</h2>
<form [formGroup]="loginRef" (ngSubmit)="signIn()">
<div class="form-container-t3">
<label>EmailId</label>
<input type="email" formControlName="emailid" placeholder="enter email"/><br/>
<label>Password</label>
<input type="password" formControlName="password" placeholder="enter password"/><br/>
<label>TypeOfUser</label>
<input type="radio" id="admin1" name="typeOfUser" value="admin"
formControlName="typeOfUser"/>admin
<input type="radio" id="user1" name="typeOfUser" value="user"
formControlName="typeOfUser"/>user<br/>
<input type="submit" id="signIn12" class="btn" value="signIn"/>
<input type="reset" class="btn" value="reset"/>
</div>
</form>
<br/>
<span style="color:red">{{msg}}</span><br/>
<a routerLink="/signUp">SignUp</a>
</div>
</div>
</body>
</html>
```

LOGIN.COMPONENT.TS

```
import { Component, OnInit } from '@angular/core';
import {FormGroup,FormControl} from '@angular/forms';
import { Router } from '@angular/router';
import { LoginService } from '../login.service';
@Component({
selector: 'app-login',
templateUrl: './login.component.html',
styleUrls: ['./login.component.css']
})
export class LoginComponent implements OnInit {
loginRef = new FormGroup({
emailid:new FormControl(),
```

```
password:new FormControl(),
typeOfUser:new FormControl()
});
msg:string=""
constructor(public ls:LoginService,public router:Router) { }
ngOnInit(): void {
}
signIn(){
let login = this.loginRef.value;
console.log(login);
this.ls.signIn(login).subscribe({
next:(result:any)=>{
console.log(result);
if(result=="Admin sucessfully login"){
sessionStorage.setItem("userDetails",login.emailid);
this.router.navigate(["adminHome"])
}else if(result=="User successfully login"){
sessionStorage.setItem("userDetails",login.emailid);
this.router.navigate(["userHome"])
}else {
this.msg=result;
}
},
error:(error:any)=>console.log(error),
complete:()=>console.log("completed")
})
}
}
```

SIGNUP.COMPONENET.HTML

```
<div class="user">
    <div class="signUp">
    <h2>Account Create</h2>
    <form [formGroup]="loginRef" (ngSubmit)="signUp()">
    <label>EmailId</label>
    <input type="email" formControlName="emailid"/><br/>
<label>Password</label>
<input type="password" formControlName="password"/><br/>
<label>TypeOfUser</label>
<input type="radio" name="typeOfUser" value="admin" formControlName="typeOfUser"/>admin
<input type="radio" name="typeOfUser" value="user" formControlName="typeOfUser"/>user<br/>
<input type="submit" class="btn"value="signUp"/>
<input type="reset" class="btn" value="reset"/>
</form>
<br/>
<span style="color:red">{{msg}}</span><br/>
<a routerLink="/login">login</a>
</div>
</div>
```

SIGNUP.COMPONENT.TS

```
import { Component, OnInit } from '@angular/core';
import {FormGroup,FormControl} from '@angular/forms';
import { LoginService } from '../login.service';
@Component({
selector: 'app-signup',
templateUrl: './signup.component.html',
styleUrls: ['./signup.component.css']
})
export class SignupComponent implements OnInit {
loginRef = new FormGroup({
emailid:new FormControl(),
password:new FormControl(),
typeOfUser:new FormControl()
});
msg:string=""
constructor(public ls:LoginService) { }
```

```
ngOnInit(): void {
}
signUp() {
let login = this.loginRef.value;
this.ls.signUp(login).subscribe({
next:(result:any)=>this.msg=result,
error:(error:any)=>console.log(error),
complete:()=>console.log("completed")
})
}
}
```

ADMINDASHBOARD.COMPONENT.HTML

```
<div>
    <h2>Welcome to home page admin {{user}}</h2>
    <a id="add1" routerLink="addProduct">Add Product</a> |
    <a routerLink="findAllProduct">View Product</a>
    <br/>
    <hr/>
    <router-outlet></router-outlet>
    <hr/>
<br/>
<input type="button" value="logout" (click)="logout()"/>
</div>
```

ADMINDASHBOARD.COMPONENT.TS

```
import { Component, OnInit } from '@angular/core';
import { Router } from '@angular/router';
@Component({
selector: 'app-admindashboard',
templateUrl: './admindashboard.component.html',
styleUrls: ['./admindashboard.component.css']
})
export class AdmindashboardComponent implements OnInit {
user:string ="";
constructor(private router:Router) { }
ngOnInit(): void {
let obj = sessionStorage.getItem("userDetails");
if(obj!=null){
this.user=obj;
}
}
logout() {
    sessionStorage.removeItem("userDetails");
    this.router.navigate(["login"]);
    }
    }
```

ADD-PRODUCT.COMPONENET.HTML

```
<div>
    <h2>Add Product</h2>
    <form [formGroup]="productRef" (ngSubmit)="storeProduct()">
    <label>PName</label>
    <input id ="pname11" type="text" formControlName="pname"><br/>
    <label>Price</label>
    <input id="price11" type="number" formControlName="price"><br/>
    <label>URL</label>
    <input id="url11" type="url" formControlName="url"><br/>
    <input id="submit11" type="submit" value="store Product"/><br/>
    <input type="reset" value="reset"/><br/>
    </form><br/>
    <span style="color:red">{{storeMsg}}</span>
    </div>
```

ADD-PRODUCT.COMPONENT.TS

```
import { Component, OnInit } from '@angular/core';
import {FormGroup,FormControl} from '@angular/forms'
import { ProductService } from '../product.service';
@Component({
selector: 'app-add-product',
templateUrl: './add-product.component.html',
styleUrls: ['./add-product.component.css']
})
export class AddProductComponent implements OnInit {
productRef = new FormGroup({
pname:new FormControl(),
price:new FormControl(),
url:new FormControl()
})
storeMsg :string =""
constructor(public ps:ProductService) { }
ngOnInit(): void {
}

storeProduct() {
    let product = this.productRef.value;
    this.ps.storeProduct(product).subscribe({
    next:(result:any)=>this.storeMsg=result,
    error:(error:any)=>console.log(error),
    complete:()=>console.log("completed")
    })
    this.productRef.reset();
    }
    }
```

ADMIN-PRODUCT-RETRIEVE.COMPONENT.HTML

```
<div>
    <h2>Add Product Details</h2>
    <div *ngIf="flag">
    <h2>Update Product</h2>
    <form (ngSubmit)="updateDataFromDb()">
    <label>PId</label>
    <input type="number" name="pid" [(ngModel)]="pid" readonly/><br/>
    <label>Price</label>
    <input type="number" name="price" [(ngModel)]="price"/><br/>
    <label>URL</label>
    <input type="URL" name="url" [(ngModel)]="url"/><br/>
    <input type="submit" value="update data"/>
    <input type="reset" value="reset"/>
    </form>
    </div>
    <span *ngFor="let p of products">
    <img src={{p.url}} width="300px" height="300px"/>
    <span>{{p.pname}} {{p.price}}</span>
    <input type="button" value="delete" (click)="deleteProduct(p.pid)"/>
    <input type="button" value="update" (click)="updateProduct(p)"/>
    </span>
    </div>
```

ADMIN-PRODUCT-RETRIEVE.COMPONENT.TS

```
import { Component, OnInit } from '@angular/core';
import { Product } from '../product';
import { ProductService } from '../product.service';
@Component({
selector: 'app-admin-product-retrieve',
templateUrl: './admin-product-retrieve.component.html',
styleUrls: ['./admin-product-retrieve.component.css']
})
export class AdminProductRetrieveComponent implements OnInit {
```

```
products:Array<Product>=[];
constructor(public ps:ProductService) { }
ngOnInit(): void {
this.findAllProduct();
}
flag:boolean = false;
pid:number =0;
price:number =0;
url:string ="";
findAllProduct() {
this.ps.findAllProduct().subscribe({
next:(result:any)=>this.products=result,
error:(error:any)=>console.log(error),
complete:()=>console.log("completed")
})
}
deleteProduct(pid:number){
//console.log(pid)
this.ps.deleteProductById(pid).subscribe({
next:(result:any)=>console.log(result),
error:(error:any)=>console.log(error),
complete:()=>{
this.findAllProduct();
}
})
}
updateProduct(product:any){
this.flag= true;
this.pid=product.pid;
this.price=product.price;
this.url=product.url;
}
updateDataFromDb(){
let product = {pid:this.pid,price:this.price,url:this.url};
this.ps.updateProduct(product).subscribe({
next:(result:any)=>console.log(result),
error:(error:any)=>console.log(error),
complete:()=>{
this.findAllProduct();
}
})
this.flag=false;
}
}
```

SELENIUM WITH TESTNG

```
package com;

import org.testng.annotations.Test;
//      import org.testng.annotations.AfterClass;
//      import org.testng.annotations.Test;
//      import org.testng.AssertJUnit;
//      import org.testng.annotations.Test;
        import org.testng.asserts.SoftAssert;
//      import java.util.concurrent.TimeUnit;

        import org.openqa.selenium.By;
//import org.openqa.selenium.By.ById;
//import org.openqa.selenium.By.ByXPath;
////import org.openqa.selenium.JavascriptExecutor;
//      import org.openqa.selenium.NoSuchElementException;
        import org.openqa.selenium.WebDriver;
        import org.openqa.selenium.WebElement;
```

```
import org.openqa.selenium.chrome.ChromeDriver;
//	import org.openqa.selenium.support.ui.FluentWait;
//	import org.openqa.selenium.support.ui.Wait;
//	import org.testng.annotations.AfterClass;
//	import org.testng.annotations.AfterMethod;

	public class driver_your_way_test {

		// Step 1: Initialize the webdriver
		WebDriver driver = null;
		SoftAssert soft = new SoftAssert();

		@Test
		public void initialization_T0() {
			// Step 2: Declare a path and set property for google chrome
driver
			String path =
"C:\\Users\\KIIT\\Downloads\\chromedriver_win32\\chromedriver.exe";
			System.setProperty("webdriver.chrome.driver", path);
			driver = new ChromeDriver();

		}

		@Test(groups = "Chrome", dependsOnMethods =
{ "initialization_T0" })
		public void cross_T1() {
			System.out.println("Testcases Starting...");
			System.out.println();

			// starting chrome
			driver.get("http://localhost:4200/login");

			try {
				Thread.sleep(5000);

			} catch (InterruptedException e) {
				e.printStackTrace();
			}
			WebElement
email=driver.findElement(By.xpath("/html/body/app-root/html/body/div/app-
login/html/body/div/div/form/div/input[1]"));
			email.sendKeys("admin@gmail.com");
			WebElement
password=driver.findElement(By.xpath("/html/body/app-root/html/body/div/app-
login/html/body/div/div/form/div/input[2]"));
			password.sendKeys("admin@123");
			WebElement admin=driver.findElement(By.id("admin1"));
			admin.click();
			WebElement signIn=driver.findElement(By.id("signIn12"));
			signIn.submit();
```

```
		try {
			Thread.sleep(5000);

		} catch (InterruptedException e) {
			e.printStackTrace();
		}

	}


	@Test(groups = "Chrome", dependsOnMethods = {"cross_T1"})
	public void cross_T2() {
		try {
			Thread.sleep(5000);

		} catch (InterruptedException e) {
			e.printStackTrace();
		}
		// Clicking Search Button

		WebElement
addproduct11=driver.findElement(By.xpath("//*[@id=\"add1\"]"));
		addproduct11.click();
		WebElement pname=driver.findElement(By.id("pname11"));
		pname.sendKeys("I10");
		try {
			Thread.sleep(3000);

		} catch (InterruptedException e) {
			e.printStackTrace();
		}
		WebElement price=driver.findElement(By.id("price11"));
		price.sendKeys("500000");
		try {
			Thread.sleep(3000);

		} catch (InterruptedException e) {
			e.printStackTrace();
		}
		WebElement url=driver.findElement(By.id("url11"));

	url.sendKeys("https://upload.wikimedia.org/wikipedia/commons/thumb/4/44/
Hyundai_i10_1.0_Intro_%28III%29_%E2%80%93_f_03012021.jpg/640px-
Hyundai_i10_1.0_Intro_%28III%29_%E2%80%93_f_03012021.jpg");
		WebElement store=driver.findElement(By.id("submit11"));
		try {
			Thread.sleep(5000);

		} catch (InterruptedException e) {
			e.printStackTrace();
```

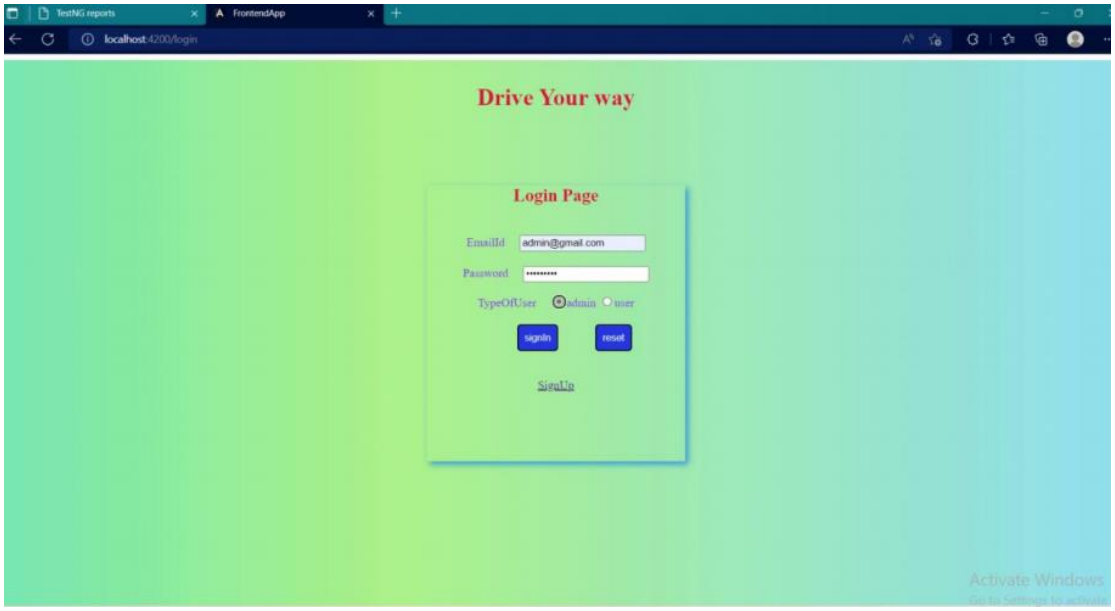
```
                }
                store.submit();

            }

}
```

APPLICATION.PROPERTIES

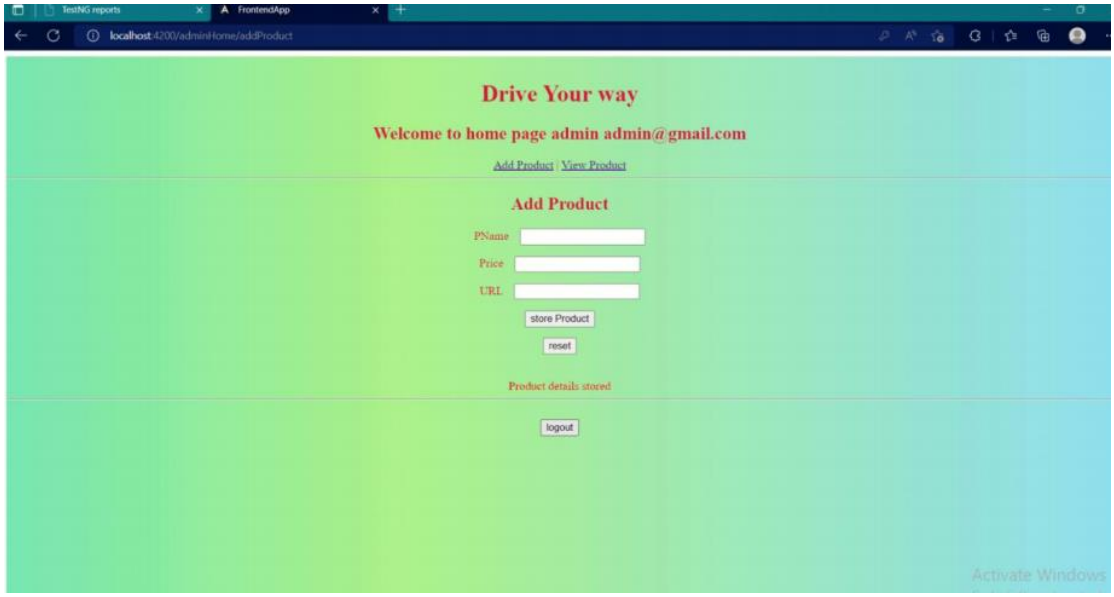
```
spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
spring.datasource.url=jdbc:mysql://localhost:3306/capstone
spring.datasource.username=root
spring.datasource.password=soumyaranjan@261412
spring.jpa.hibernate.ddl-auto=update
server.port=9090
```

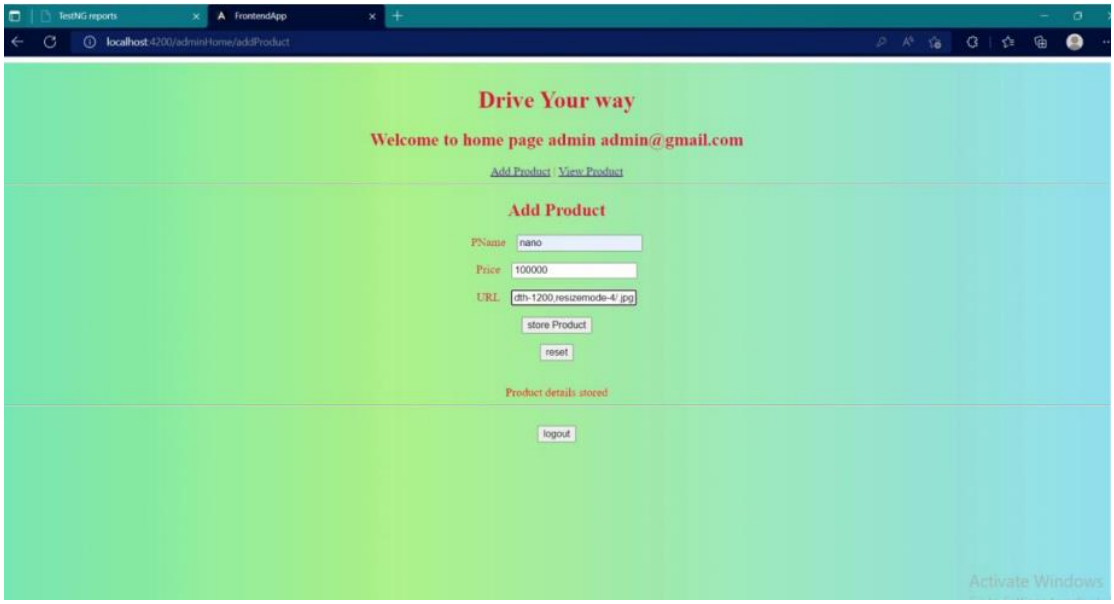
OUTPUT FOR DRIVE YOUR WAY

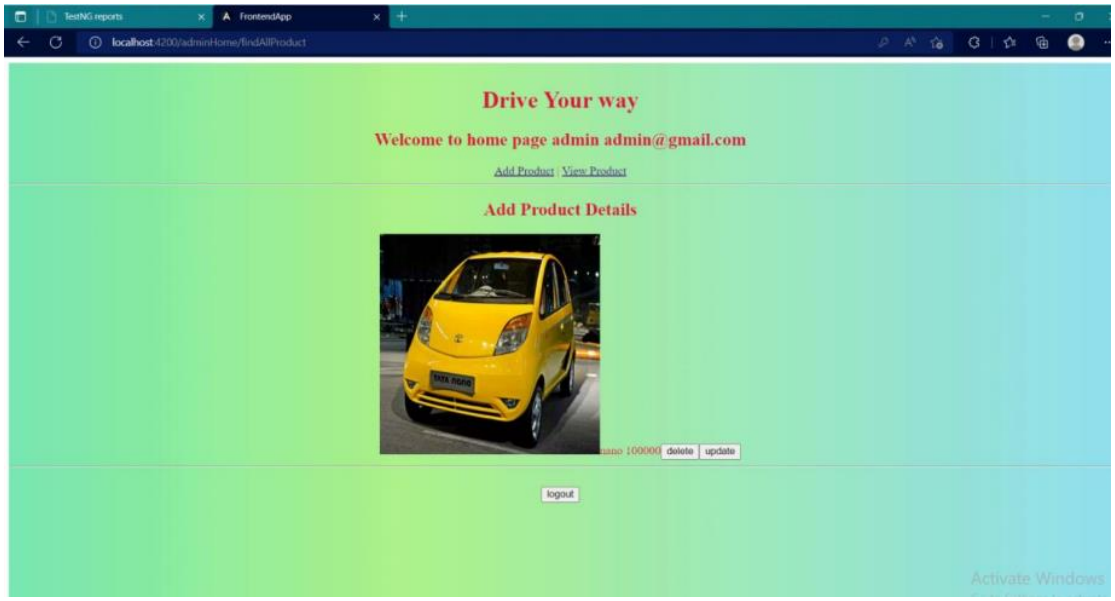
Creating an admin account



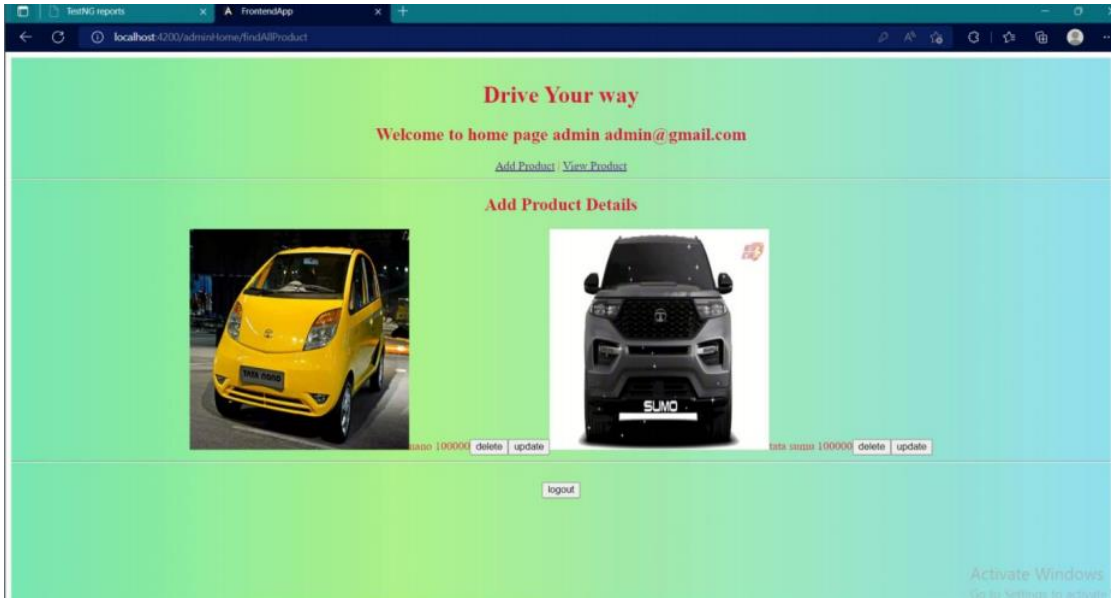
Add a Product


TestNG reports
FrontendApp
localhost:4200/adminHome/addProduct
Drive Your way
Welcome to home page admin admin@gmail.com
Add Product | View Product
Add Product
PName
Price
URL
store Product
reset
Product details stored
logout
Activate Windows

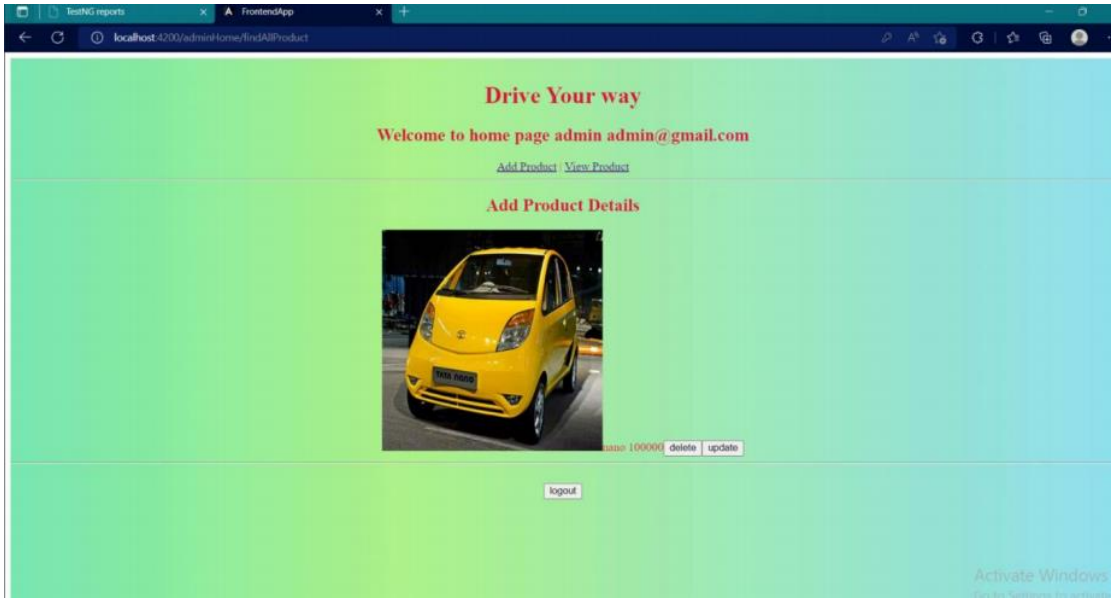

TestNG reports
FrontendApp
localhost:4200/adminHome/addProduct
Drive Your way
Welcome to home page admin admin@gmail.com
Add Product | View Product
Add Product
PName
nano
Price
100000
URL
dth-1200,resizemode-4/.jpg
store Product
reset
Product details stored
logout
Activate Windows

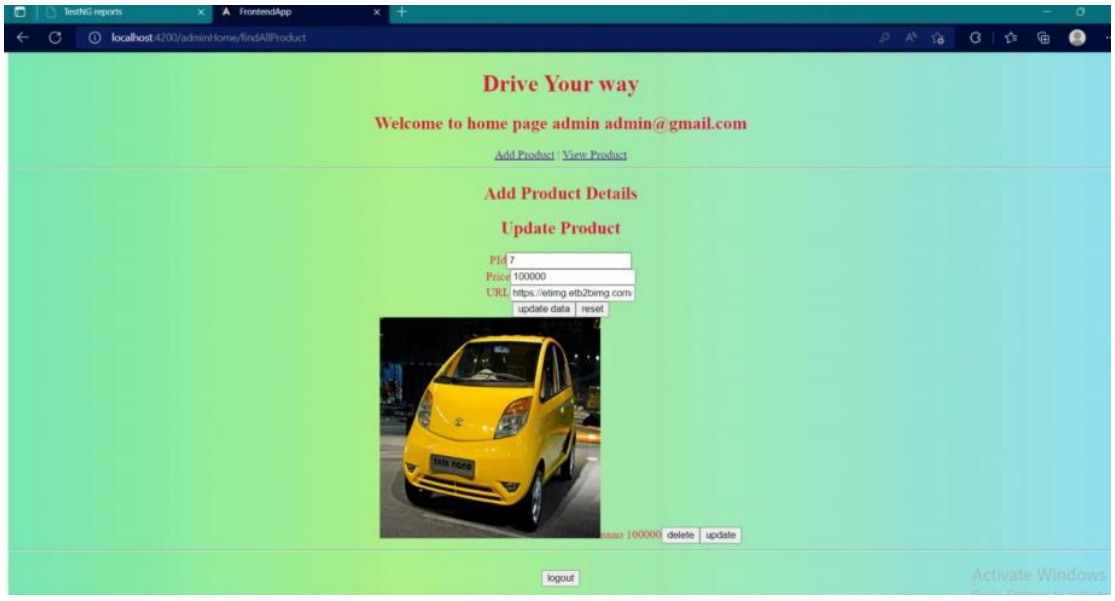

TestNG reports
FrontendApp
localhost:4200/adminHome/findAllProduct
Drive Your way
Welcome to home page admin admin@gmail.com
Add Product | View Product
Add Product Details
nano 100000
delete
update
logout
Activate Windows


localhost:4200/adminHome/addProduct
Drive Your way
Welcome to home page admin admin@gmail.com
Add Product | View Product
Add Product
PName
tata sumu
Price
100000
URL
s/2020/04/Tata-Sumo-2.jpg
store Product
reset
logout
Activate Windows

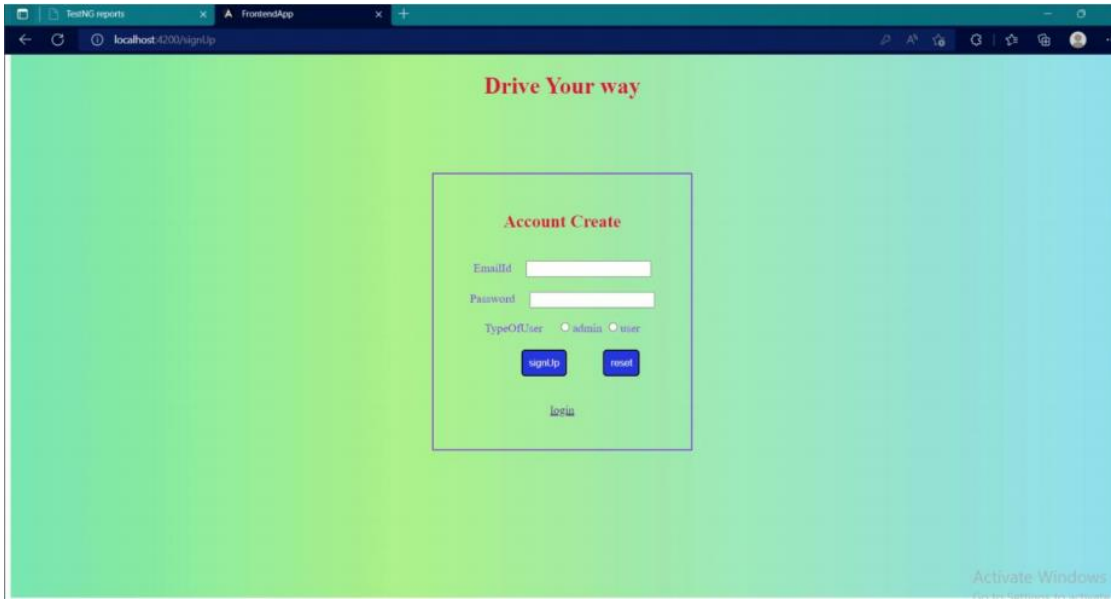

TestNG reports
FrontendApp
localhost:4200/adminHome/findAllProduct
Drive Your way
Welcome to home page admin admin@gmail.com
Add Product | View Product
Add Product Details
SUMO
nano 100000
delete
update
tata sumu 100000
delete
update
logout
Activate Windows

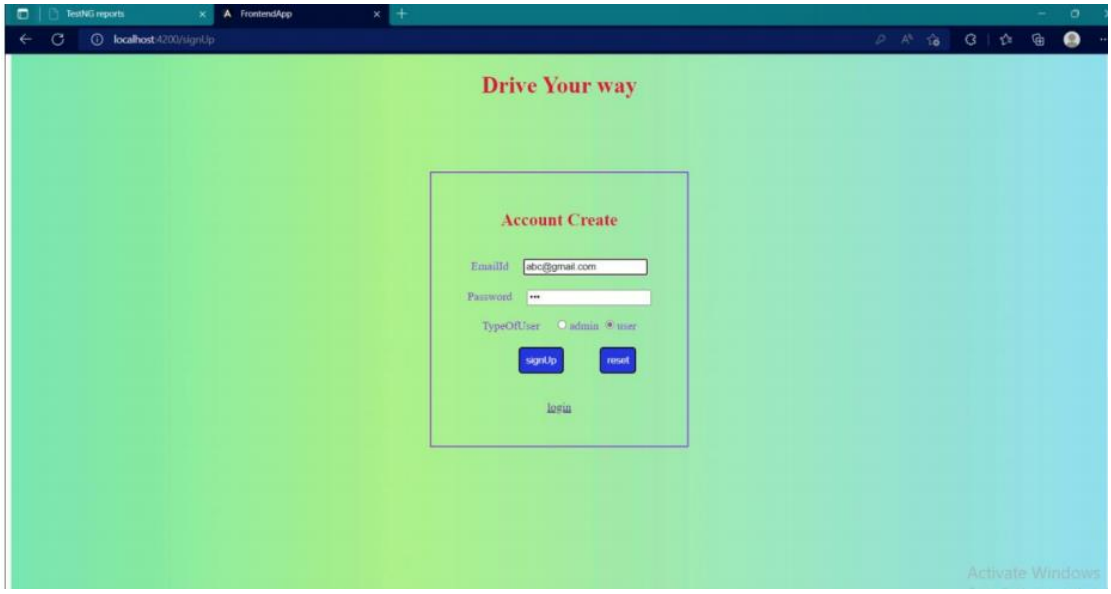
Delete/Update a Product

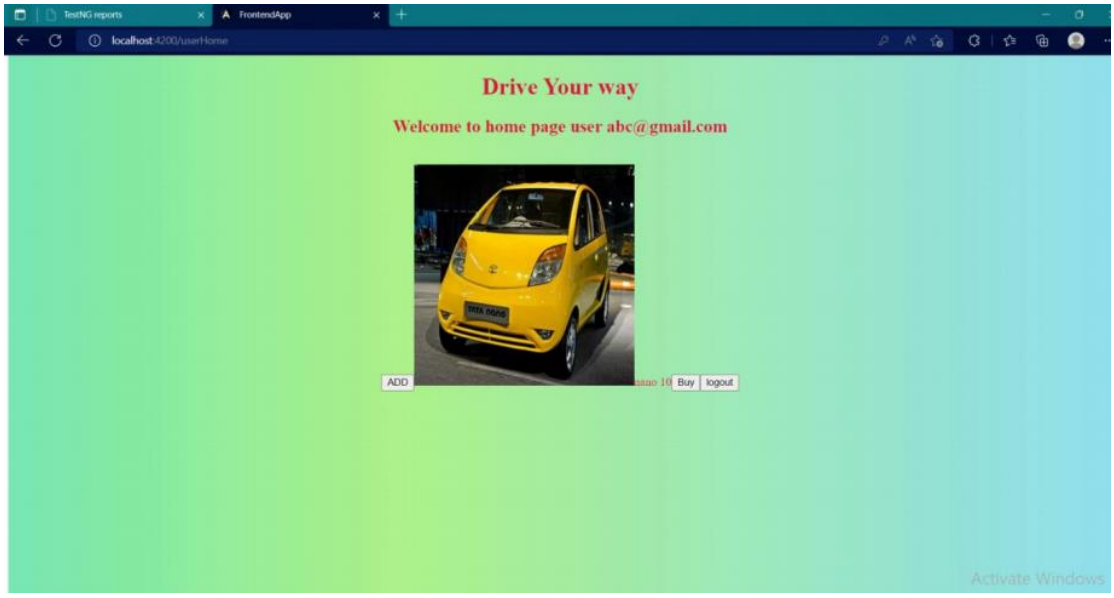




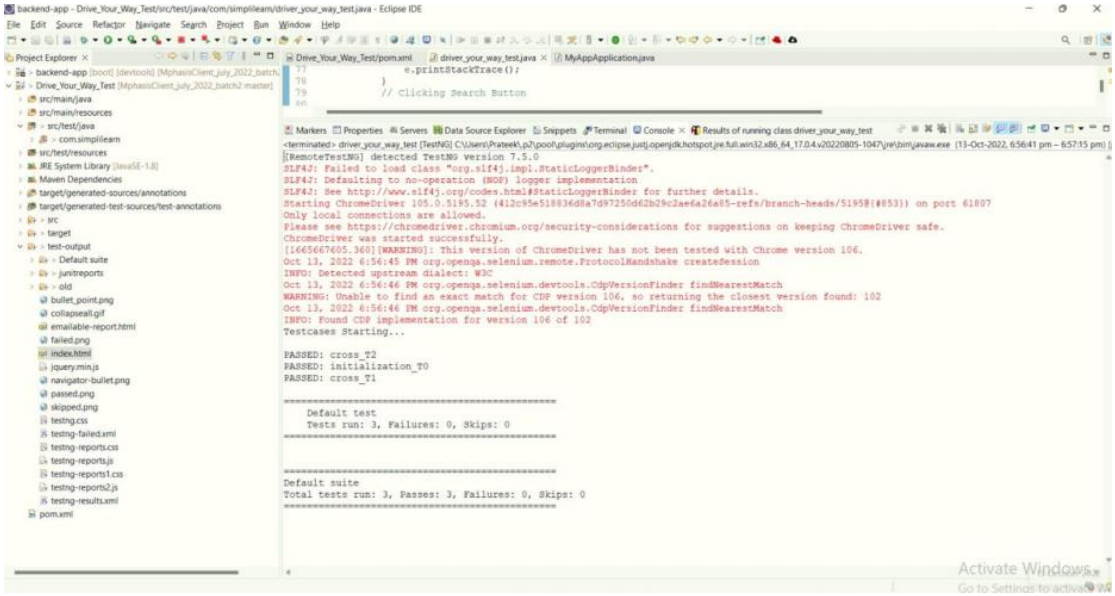
Creating a user account

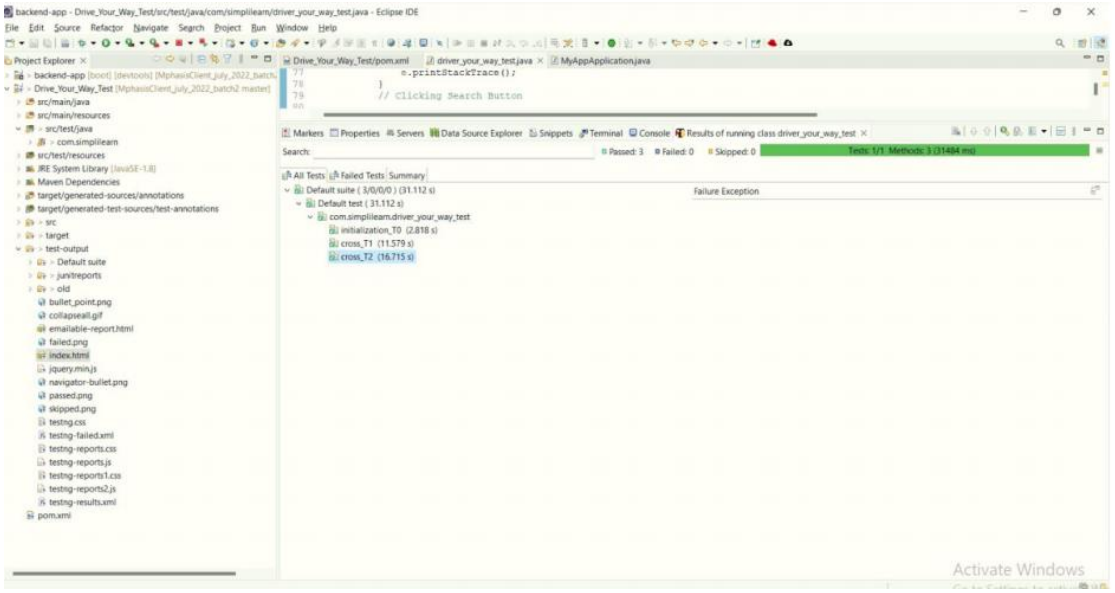


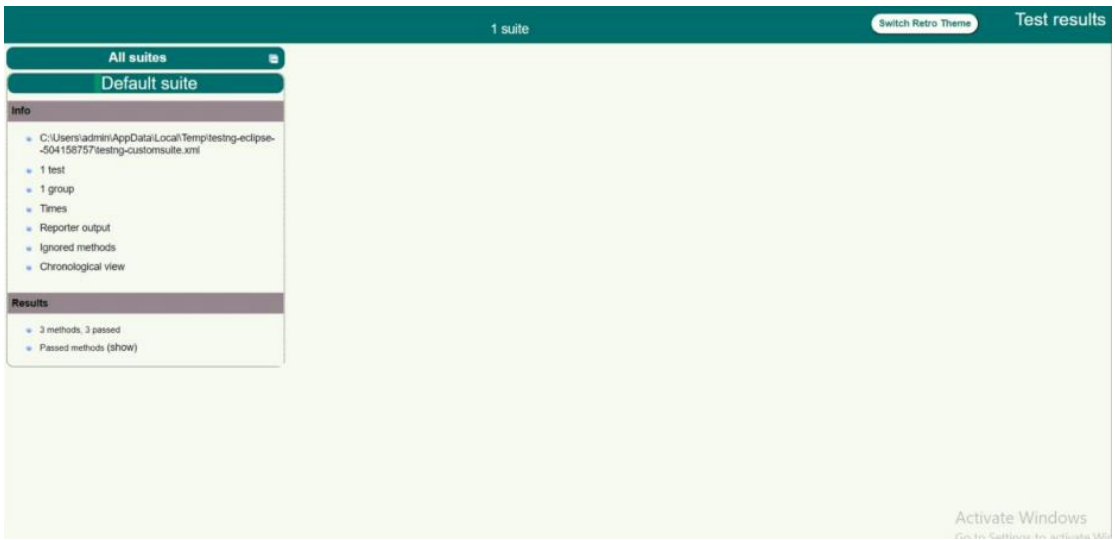

TestNG reports
FrontendApp
localhost:4200/signUp
Drive Your way
Account Create
EmailId
abc@gmail.com
Password
TypeOfUser
admin
user
signUp
reset
login
Activate Windows

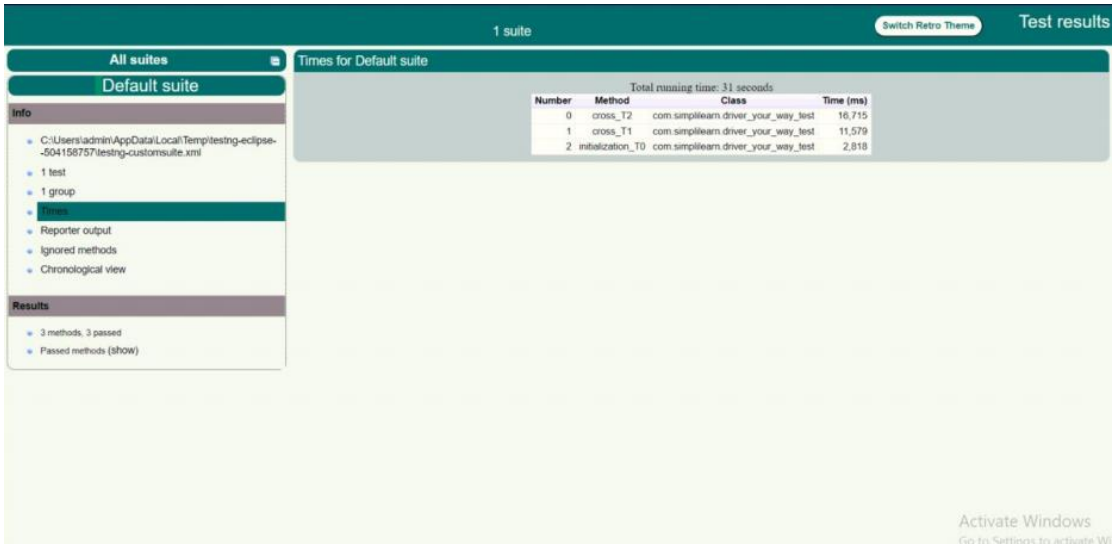

TestNG reports
FrontendApp
localhost:4200/userHome
Drive Your way
Welcome to home page user abc@gmail.com
ADD
Buy
logout
Activate Windows

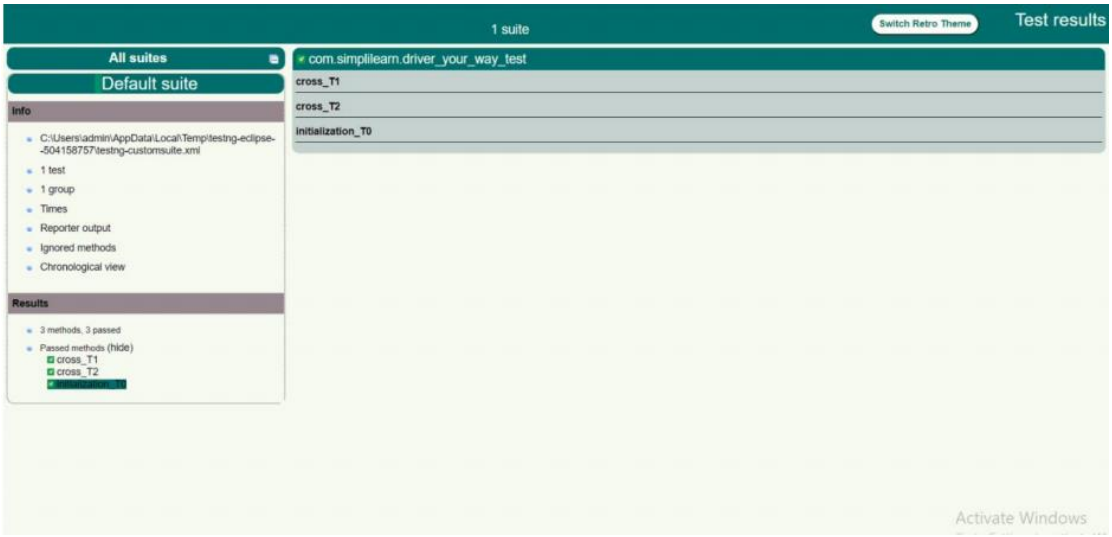
Testing


backend-app - Drive_Your_Way_Test/src/test/java/com/simplilearn/driver_your_way_test.java - Eclipse IDE
File Edit Source Refactor Navigate Search Project Run Window Help
Project Explorer
Drive_Your_Way_Test/pom.xml
driver_your_way_test.java
MyAppApplication.java
e.printStackTrace();
}
// Clicking Search Button
Markers Properties Servers Data Source Explorer Snippets Terminal Console Results of running class driver_your_way_test
[RemoteTestNG] detected TestNG version 7.5.0
SLF4J: Failed to load class "org.slf4j.impl.StaticLoggerBinder".
SLF4J: Defaulting to no-operation (NOP) logger implementation
SLF4J: See http://www.slf4j.org/codes.html#StaticLoggerBinder for further details.
Starting ChromeDriver 105.0.5195.52 (412c95e518836d8a7d97250d62b29c2ae6a26a65-refs/branch-heads/5195@{#853}) on port 61807
Only local connections are allowed.
Please see https://chromedriver.chromium.org/security-considerations for suggestions on keeping ChromeDriver safe.
ChromeDriver was started successfully.
[1665667605.360][WARNING]: This version of ChromeDriver has not been tested with Chrome version 106.
Oct 13, 2022 6:56:45 PM org.openqa.selenium.remote.ProtocolHandshake createSession
INFO: Detected upstream dialect: W3C
Oct 13, 2022 6:56:46 PM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
WARNING: Unable to find an exact match for CDP version 106, so returning the closest version found: 102
Oct 13, 2022 6:56:46 PM org.openqa.selenium.devtools.CdpVersionFinder findNearestMatch
INFO: Found CDP implementation for version 106 of 102
Testcases Starting...
PASSED: cross_T2
PASSED: initialization_T0
PASSED: cross_T1
Default test
Tests run: 3, Failures: 0, Skips: 0
Default suite
Total tests run: 3, Passes: 3, Failures: 0, Skips: 0
Activate Windows

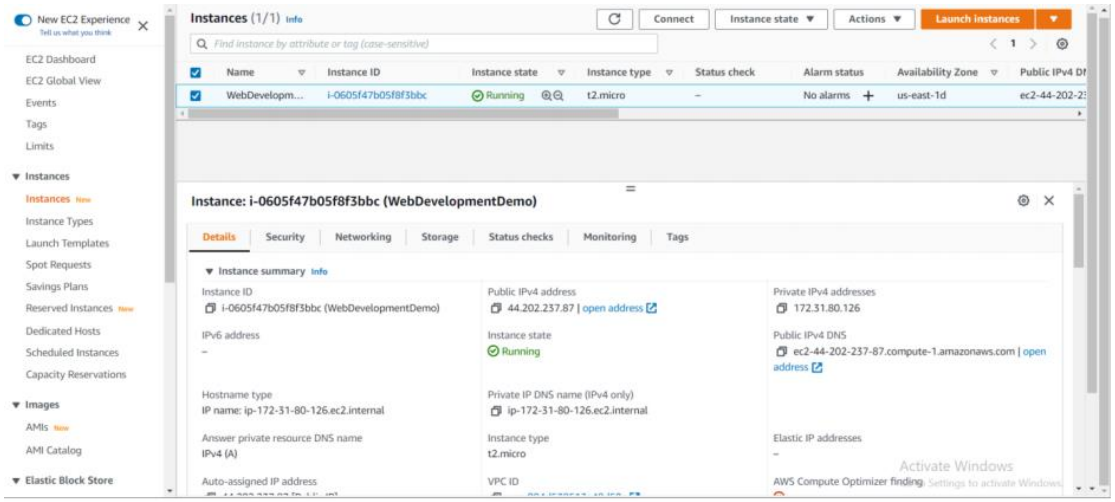

backend-app - Drive_Your_Way_Test/src/test/java/com/simplilearn/driver_your_way_test.java - Eclipse IDE
Markers Properties Servers Data Source Explorer Snippets Terminal Console Results of running class driver_your_way_test
Search:
Passed: 3 Failed: 0 Skipped: 0
Tests: 1/1 Methods: 3 (31484 ms)
All Tests Failed Tests Summary
Default suite (3/0/0/0) (31.112 s)
Default test (31.112 s)
com.simplilearn.driver_your_way_test
initialization_T0 (2.818 s)
cross_T1 (11.579 s)
cross_T2 (16.715 s)
Failure Exception
Activate Windows


1 suite
Switch Retro Theme
Test results
All suites
Default suite
Info
C:\Users\admin\AppData\Local\Temp\testng-eclipse--504158757\testng-customsuite.xml
1 test
1 group
Times
Reporter output
Ignored methods
Chronological view
Results
3 methods, 3 passed
Passed methods (show)
Activate Windows

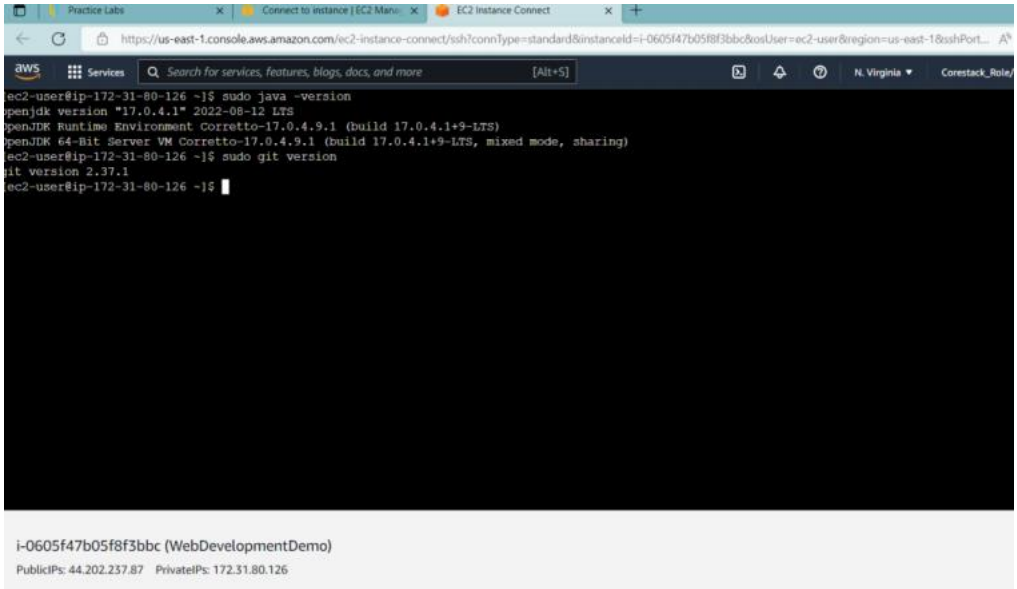




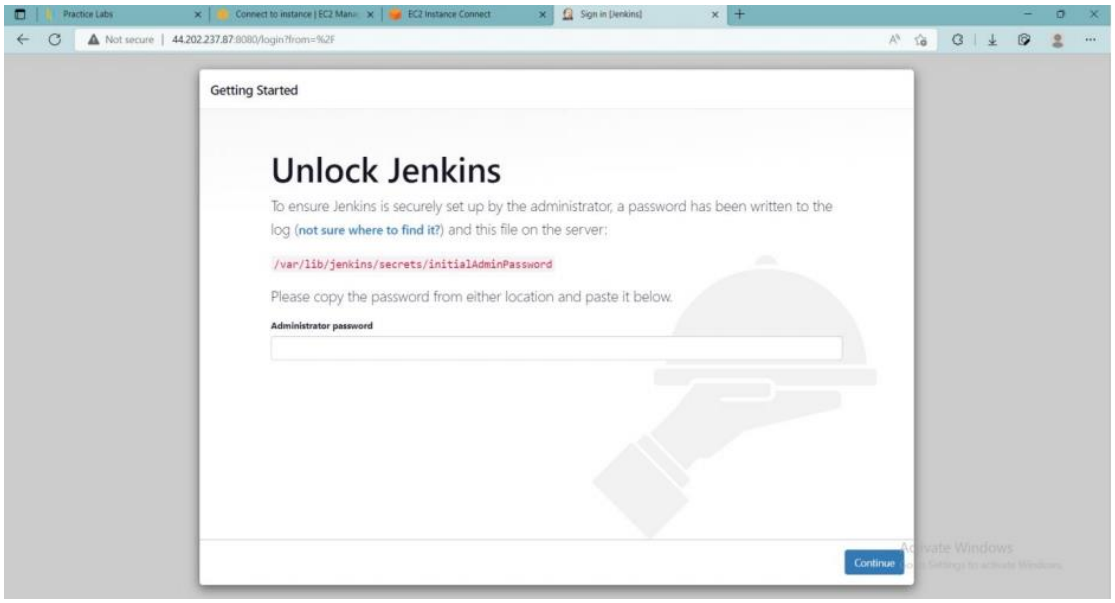
connect to ec2 instance

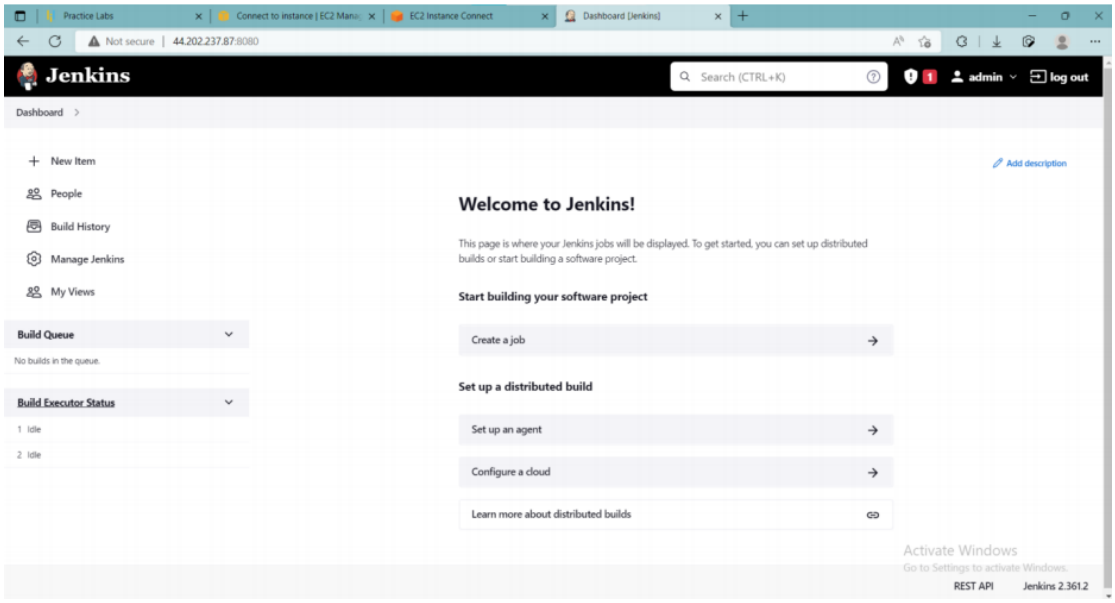


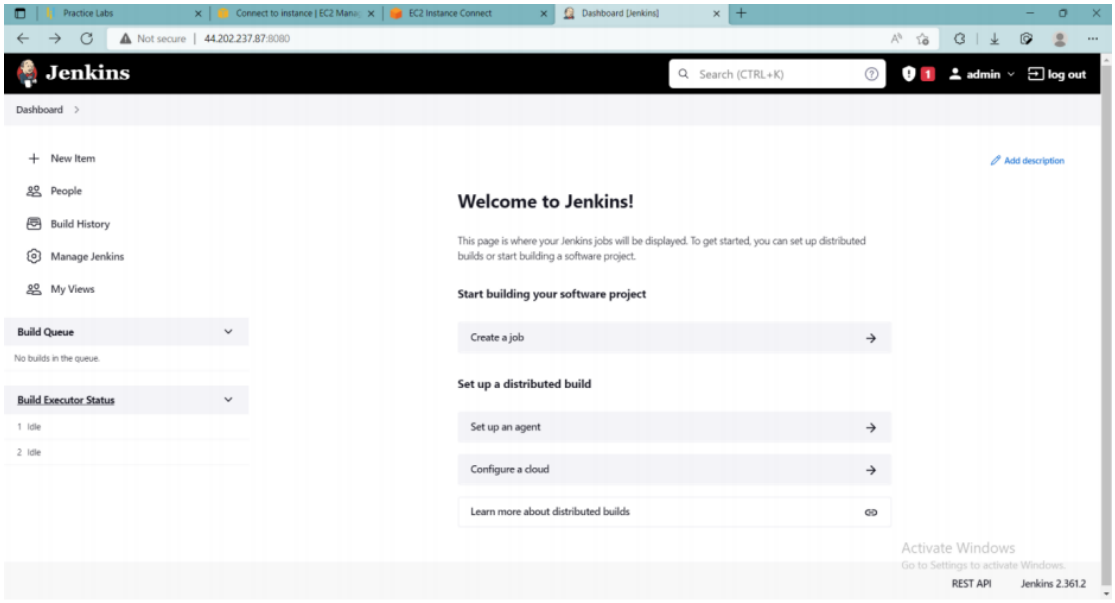
check java and git are installed or not.



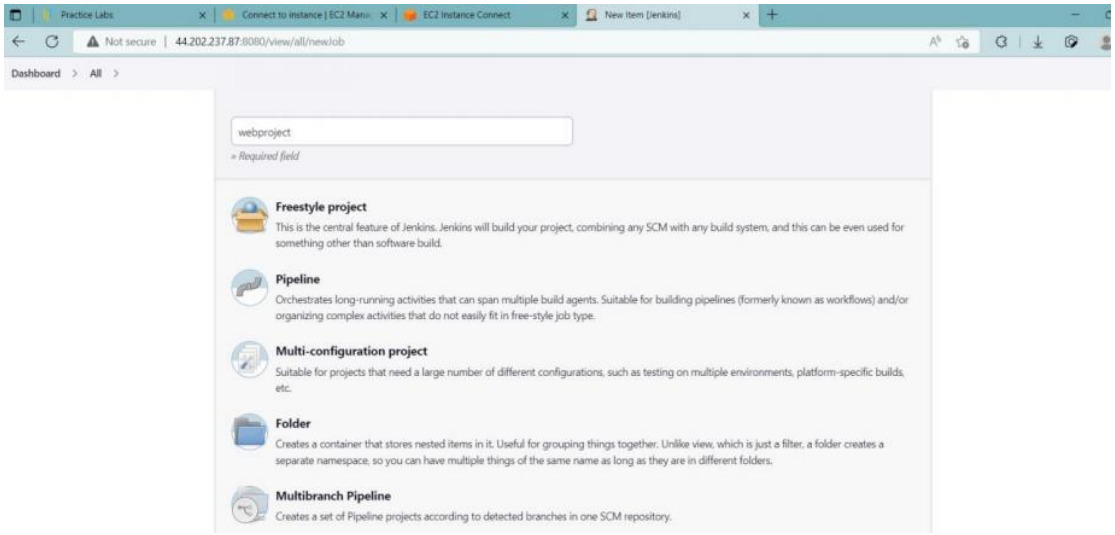
use Jenkkins
Use public IP :8080 as shown for jenkkins
Following page will open

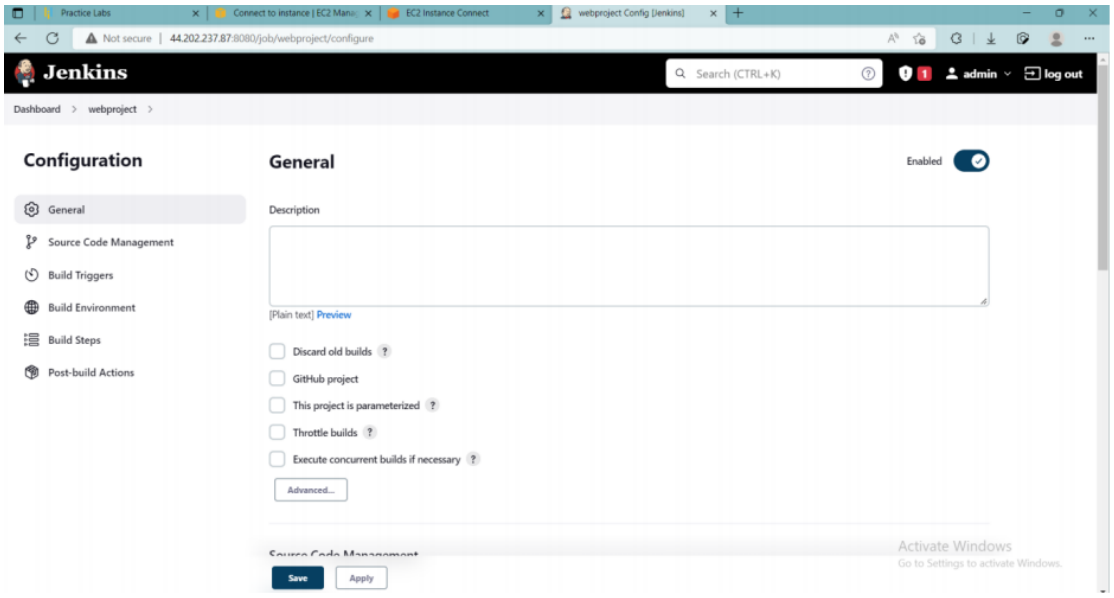


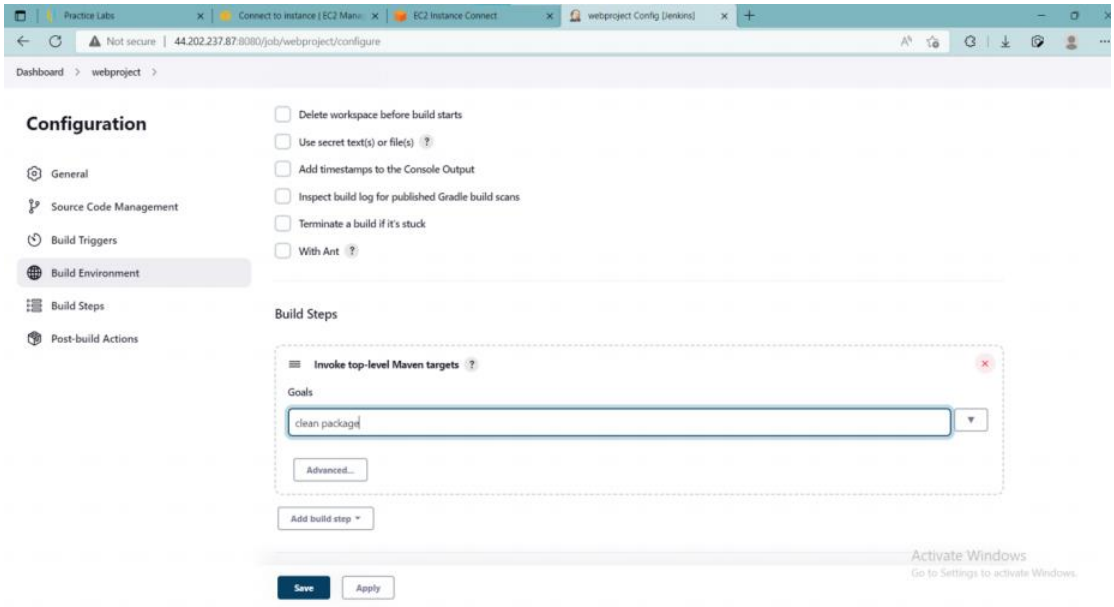


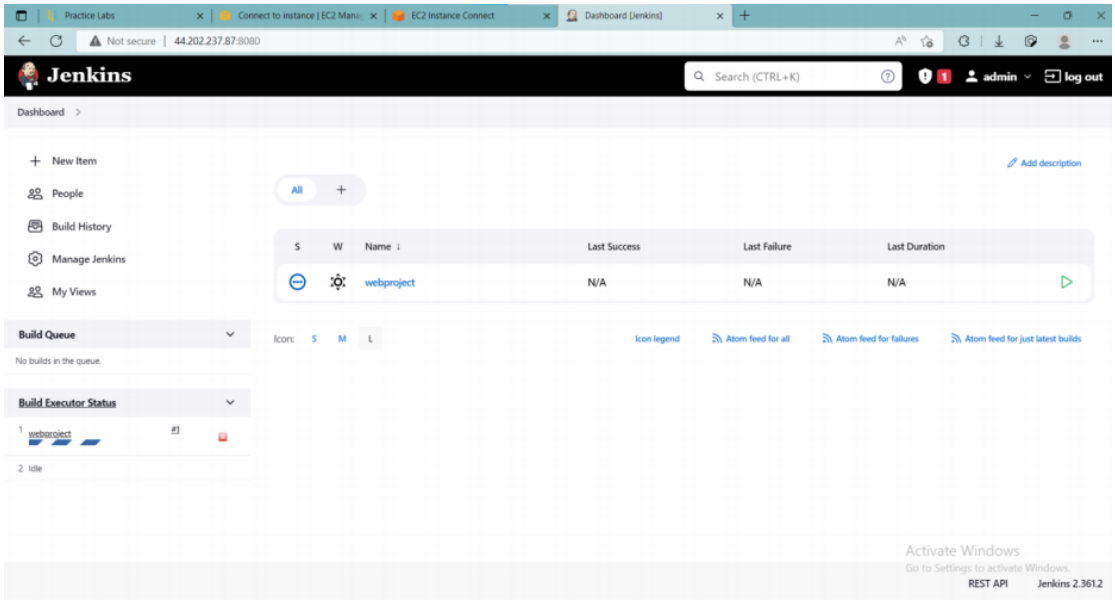


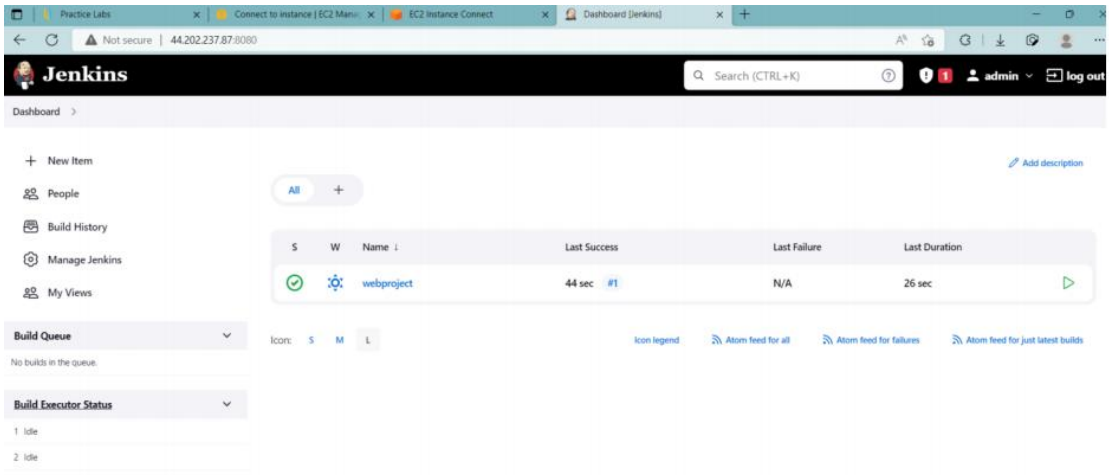
Build the project using Jenkins


Practice Labs
Connect to instance | EC2 Manag
EC2 Instance Connect
New Item [Jenkins]
Not secure | 44.202.237.87:8080/view/all/newJob
Dashboard > All >
webproject
» Required field
Freestyle project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.
Pipeline
Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
Multi-configuration project
Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
Folder
Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder creates a separate namespace, so you can have multiple things of the same name as long as they are in different folders.
Multibranch Pipeline
Creates a set of Pipeline projects according to detected branches in one SCM repository.

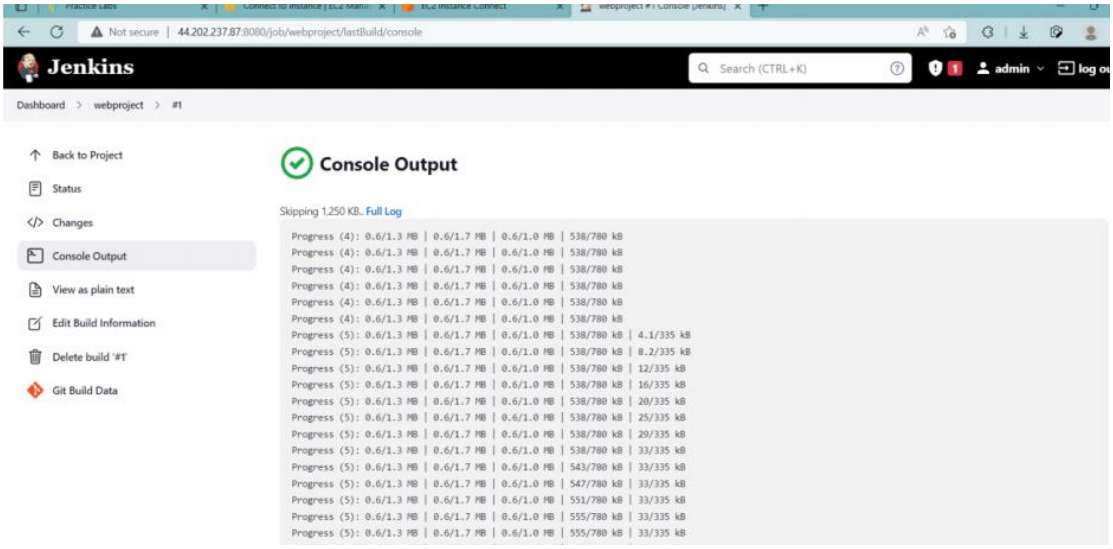

Practice Labs
Connect to instance | EC2 Manag
EC2 Instance Connect
webproject Config [Jenkins]
Not secure | 44.202.237.87:8080/job/webproject/configure
Jenkins
Search (CTRL+K)
admin
log out
Dashboard > webproject >
Configuration
General
Source Code Management
Build Triggers
Build Environment
Build Steps
Post-build Actions
General
Enabled
Description
[Plain text] Preview
Discard old builds
GitHub project
This project is parameterized
Throttle builds
Execute concurrent builds if necessary
Advanced...
Save
Apply
Activate Windows
Go to Settings to activate Windows.

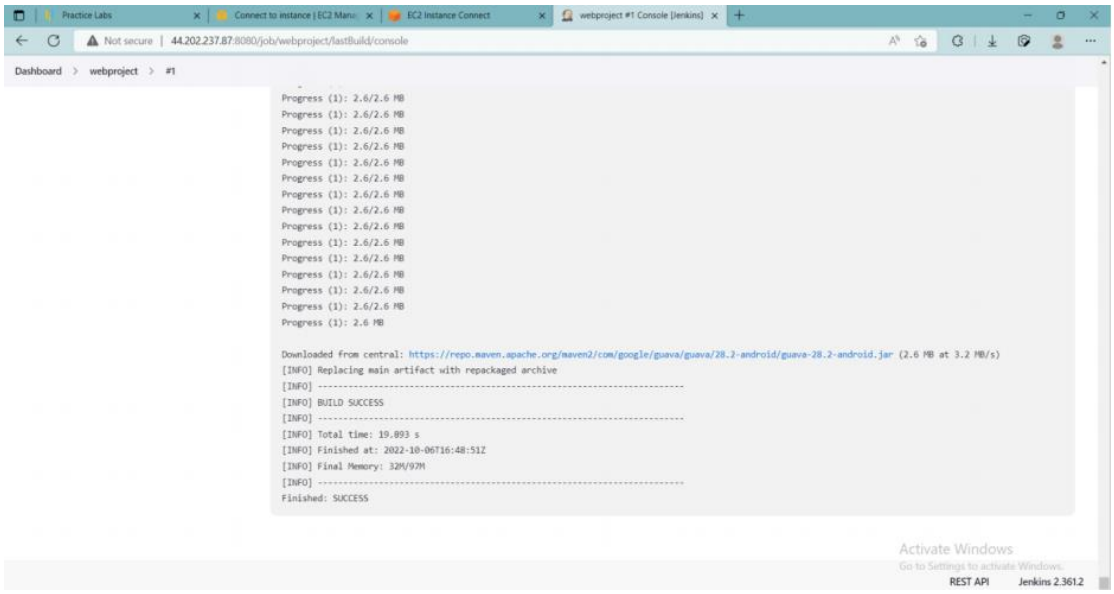




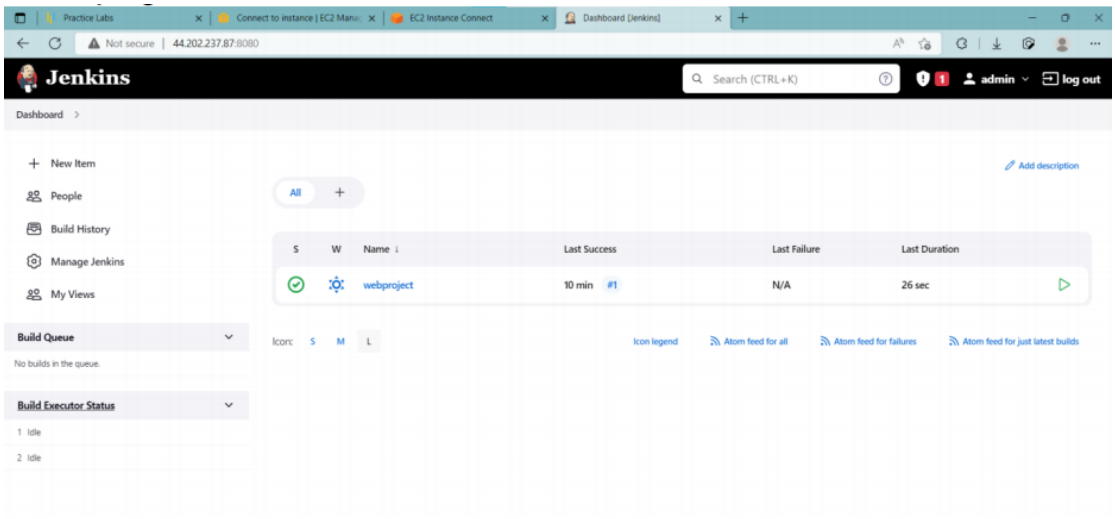


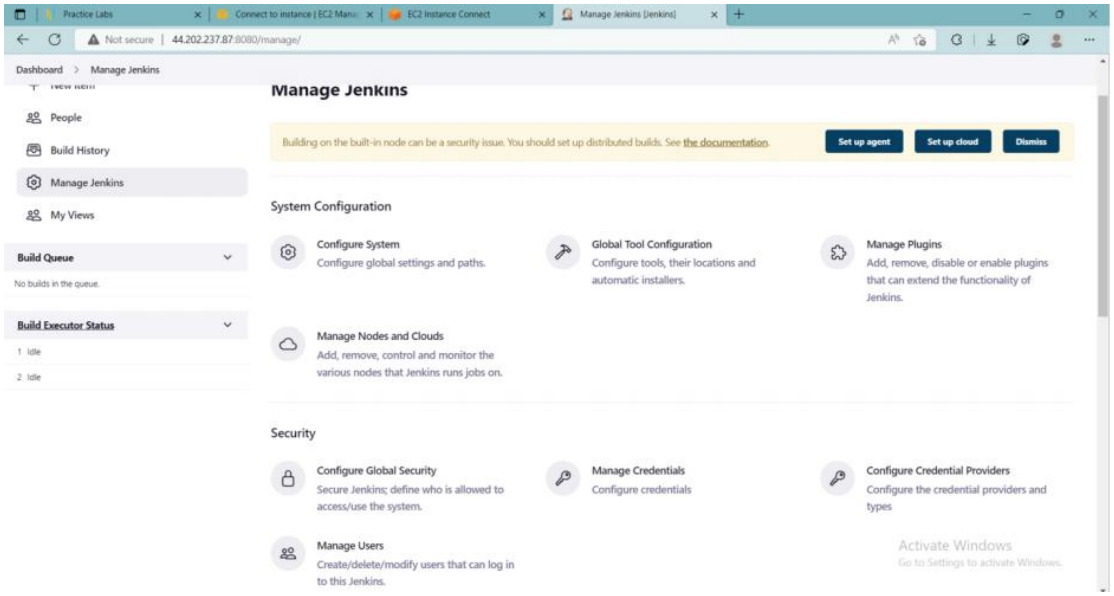
Console output

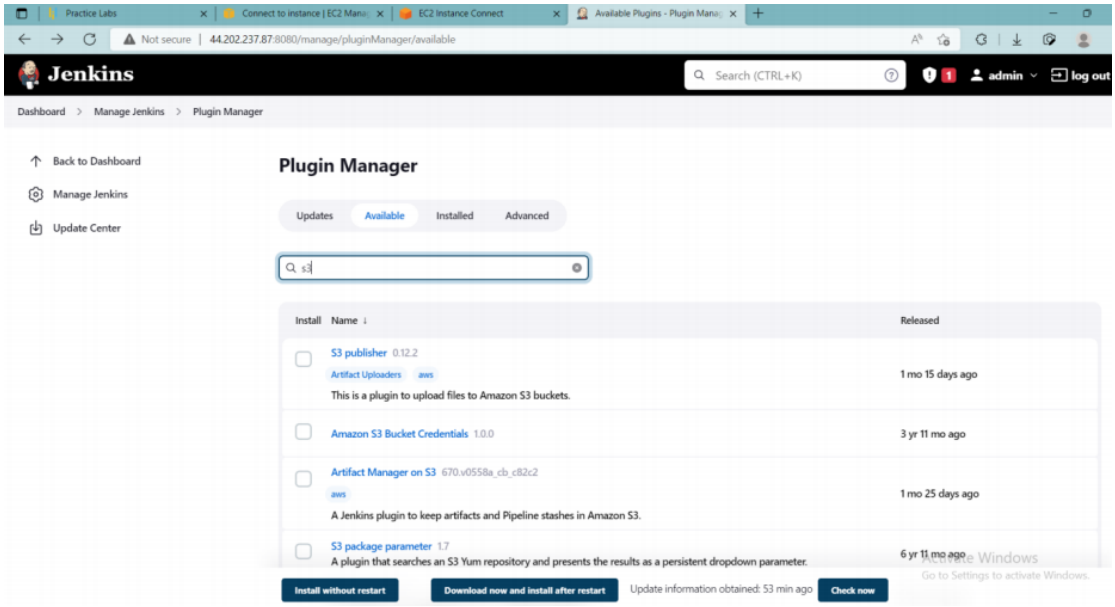


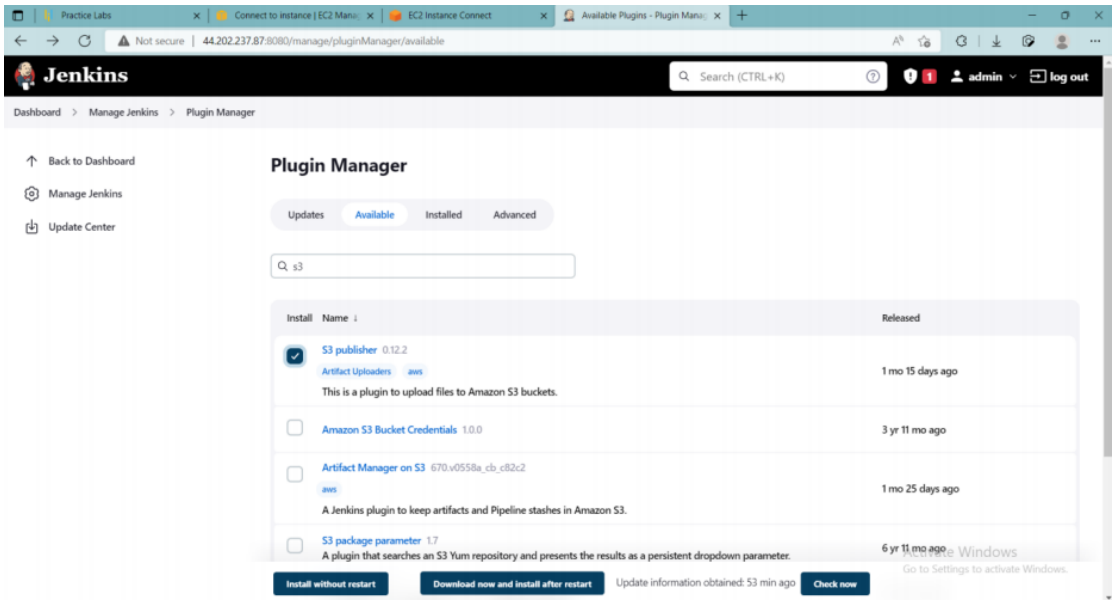


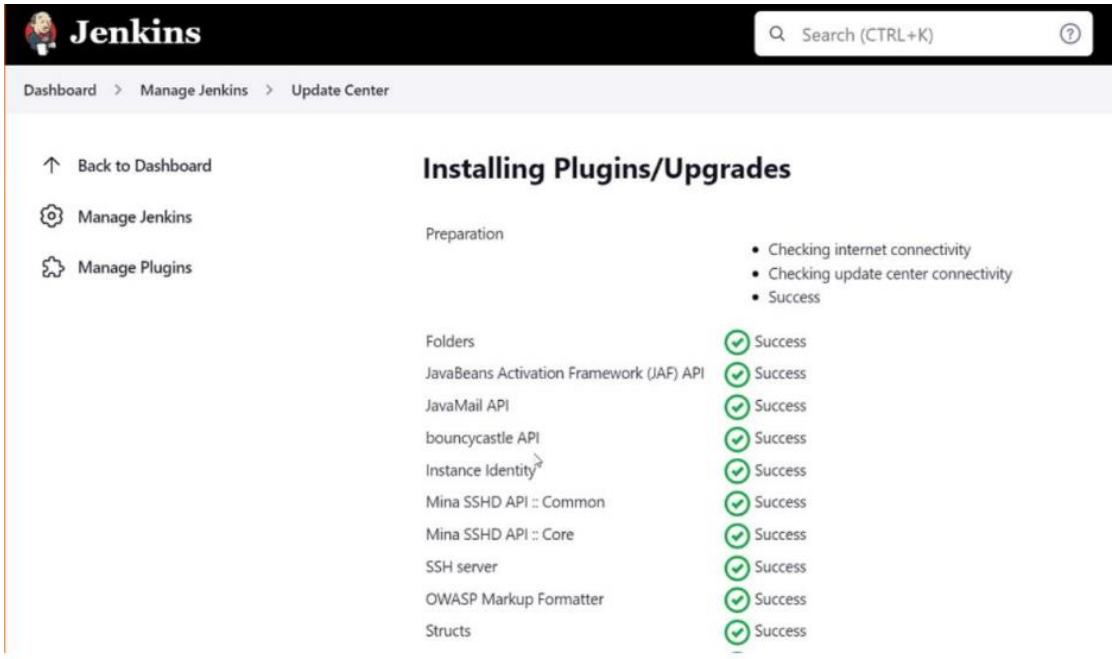
Now for s3 we have to configure it.
Install plugins


Practice Labs
Connect to instance | EC2 Mana
EC2 Instance Connect
Dashboard [Jenkins]
Not secure | 44.202.237.87:8080
Jenkins
Search (CTRL+K)
admin
log out
Dashboard
New Item
People
Build History
Manage Jenkins
My Views
Build Queue
No builds in the queue.
Build Executor Status
1 Idle
2 Idle
Add description
All
S
W
Name
Last Success
Last Failure
Last Duration
webproject
10 min #1
N/A
26 sec
Icon: S M L
Icon legend
Atom feed for all
Atom feed for failures
Atom feed for just latest builds


Practice Labs
Connect to instance | EC2 Mana
EC2 Instance Connect
Manage Jenkins [Jenkins]
Not secure | 44.202.237.87:8080/manage/
Dashboard > Manage Jenkins
People
Build History
Manage Jenkins
My Views
Build Queue
No builds in the queue.
Build Executor Status
1 Idle
2 Idle
Manage Jenkins
Building on the built-in node can be a security issue. You should set up distributed builds. See the documentation.
Set up agent
Set up cloud
Dismiss
System Configuration
Configure System
Configure global settings and paths.
Global Tool Configuration
Configure tools, their locations and automatic installers.
Manage Plugins
Add, remove, disable or enable plugins that can extend the functionality of Jenkins.
Manage Nodes and Clouds
Add, remove, control and monitor the various nodes that Jenkins runs jobs on.
Security
Configure Global Security
Secure Jenkins; define who is allowed to access/use the system.
Manage Credentials
Configure credentials
Configure Credential Providers
Configure the credential providers and types
Manage Users
Create/delete/modify users that can log in to this Jenkins.
Activate Windows


Practice Labs
Connect to instance | EC2 Mana
EC2 Instance Connect
Available Plugins - Plugin Mana
Not secure | 44.202.237.87:8080/manage/pluginManager/available
Jenkins
Search (CTRL+K)
admin
log out
Dashboard > Manage Jenkins > Plugin Manager
Back to Dashboard
Manage Jenkins
Update Center
Plugin Manager
Updates
Available
Installed
Advanced
s3
Install
Name
Released
S3 publisher 0.12.2
Artifact Uploaders
aws
This is a plugin to upload files to Amazon S3 buckets.
1 mo 15 days ago
Amazon S3 Bucket Credentials 1.0.0
3 yr 11 mo ago
Artifact Manager on S3 670.v0558a_cb_c82c2
aws
A Jenkins plugin to keep artifacts and Pipeline stashes in Amazon S3.
1 mo 25 days ago
S3 package parameter 1.7
A plugin that searches an S3 Yum repository and presents the results as a persistent dropdown parameter.
6 yr 11 mo ago
Install without restart
Download now and install after restart
Update information obtained: 53 min ago
Check now


Practice Labs
Connect to instance | EC2 Mana
EC2 Instance Connect
Available Plugins - Plugin Mana
Not secure | 44.202.237.87:8080/manage/pluginManager/available
Jenkins
Search (CTRL+K)
admin
log out
Dashboard > Manage Jenkins > Plugin Manager
Back to Dashboard
Manage Jenkins
Update Center
Plugin Manager
Updates
Available
Installed
Advanced
s3
Install
Name
Released
S3 publisher 0.12.2
Artifact Uploaders
aws
This is a plugin to upload files to Amazon S3 buckets.
1 mo 15 days ago
Amazon S3 Bucket Credentials 1.0.0
3 yr 11 mo ago
Artifact Manager on S3 670.v0558a_cb_c82c2
aws
A Jenkins plugin to keep artifacts and Pipeline stashes in Amazon S3.
1 mo 25 days ago
S3 package parameter 1.7
A plugin that searches an S3 Yum repository and presents the results as a persistent dropdown parameter.
6 yr 11 mo ago
Install without restart
Download now and install after restart
Update information obtained: 53 min ago
Check now



Finally → Docker Container
Docker file in angular