AUTOMATE A WEB APPLICATION

SOURCE CODE

# LoginFormTestCases.java

package com.simplilearn.webapp.login;

import static org.junit.jupiter.api.Assertions.assertEquals; import static org.junit.jupiter.api.Assertions.assertTrue;

import java.io.File;

import java.io.IOException;

import org.junit.jupiter.api.AfterEach; import org.junit.jupiter.api.BeforeEach; import org.junit.jupiter.api.DisplayName; import org.junit.jupiter.api.Test;

import org.openqa.selenium.By;

import org.openqa.selenium.Dimension; import org.openqa.selenium.OutputType; import org.openqa.selenium.TakesScreenshot; import org.openqa.selenium.WebDriver; import org.openqa.selenium.WebElement; import org.openqa.selenium.edge.EdgeDriver; import org.openqa.selenium.io.FileHandler;

public class LogInFormTestcases {

final String siteURL = "file:///C:/Users/nayak/Desktop/simplilearn/Phase\_5\_FSD/FSD- 5%20WorkSpace-Practice- Projects/AutomateWebApp/static/loginForm.html";

final String driverPath ="driver\\msedgedriver.exe"; WebDriver driver;

@BeforeEach

void setUp() throws Exception {

//Set Up

System.setProperty("webdriver.edge.driver",driverPath); driver = new EdgeDriver(); driver.get(siteURL);

//Add maximize size for snapshot driver.manage().window().maximize();

}

@AfterEach

void tearDown() throws Exception { driver.close();

}

@Test

@DisplayName("Test :: Verify User Input")

void testUserInput() throws InterruptedException, IOException {

//Find Input Field

WebElement input = driver.findElement(By.cssSelector("body > form > fieldset > div:nth-child(1) > input[type=text]:nth-child(2)"));

assertTrue(input.isDisplayed()); assertTrue(input.isEnabled());

//Enter yourEmail input.sendKeys("nayaksofia"); takeScreenShot("UserInput.png"); Thread.sleep(2000);

}

private void takeScreenShot(String filename)

throws IOException { cast

driver;

//Instantiate Driver Object and Type- TakesScreenshot tsc = (TakesScreenshot)

//Generate Screenshot as file File rsc =

tsc.getScreenshotAs(OutputType.FILE);

FileHandler.copy(rsc, new File("C:\\Users\\nayak\\Desktop\\simplilearn\\Phase\_5\_FSD\\F SD-5 WorkSpace-Practice- Projects\\AutomateWebApp\\screenshot" + filename));

}

@Test

@DisplayName("Test :: Verify User Password")

void testUserPassword() throws InterruptedException, IOException {

//Find Input Field

WebElement input = driver.findElement(By.cssSelector("body > form > fieldset > div:nth-child(1) > input[type=password]:nth-child(6)"));

assertTrue(input.isDisplayed()); assertTrue(input.isEnabled());

//Enter yourEmail input.sendKeys("1234qwe"); Thread.sleep(2000);

Dimension d = new Dimension(480, 620); driver.manage().window().setSize(d);

//Step-1 :: type cast driver object into takeScreenshot instance

TakesScreenshot tsc = (TakesScreenshot)

driver;

//Step-2 :: Generate a screenshot as file File rsc =

tsc.getScreenshotAs(OutputType.FILE);

FileHandler.copy(rsc,new File("C:\\Users\\nayak\\Desktop\\simplilearn\\Phase\_5\_FSD\\F SD-5 WorkSpace-Practice- Projects\\AutomateWebApp\\screenshot\\pasword.png"));

}

@Test

@DisplayName("Test :: Verify Login Button")

void testLogInButton() throws InterruptedException, IOException {

//Find Input Field

WebElement button = driver.findElement(By.cssSelector("body > form > fieldset > div:nth-child(1) > button"));

assertTrue(button.isDisplayed()); assertTrue(button.isEnabled());

Dimension d = new Dimension(480, 620);

driver.manage().window().setSize(d); button.submit();

Thread.sleep(2000);

//Step-1 :: type cast driver object into takeScreenshot instance

TakesScreenshot tsc = (TakesScreenshot)

driver;

//Step-2 :: Generate a screenshot as file File rsc =

tsc.getScreenshotAs(OutputType.FILE);

FileHandler.copy(rsc,new File("C:\\Users\\nayak\\Desktop\\simplilearn\\Phase\_5\_FSD\\F SD-5 WorkSpace-Practice- Projects\\AutomateWebApp\\screenshot\\pasword-repet.png"));

}

@Test

@DisplayName("Test :: Verify and de-select Check Box") void testCheckBox() throws InterruptedException,

IOException {

//Find Input Field

WebElement checkbox = driver.findElement(By.cssSelector("body > form > fieldset > div:nth-child(1) > label:nth-child(11) > input[type=checkbox]"));

assertEquals(true, checkbox.isDisplayed()); assertEquals(true, checkbox.isEnabled());

Dimension d = new Dimension(480, 620); driver.manage().window().setSize(d); checkbox.click();

assertEquals(false, checkbox.isSelected()); Thread.sleep(2000);

//Step-1 :: type cast driver object into takeScreenshot instance

TakesScreenshot tsc = (TakesScreenshot)

driver;

//Step-2 :: Generate a screenshot as file

File rsc = tsc.getScreenshotAs(OutputType.FILE);

FileHandler.copy(rsc,new File("C:\\Users\\nayak\\Desktop\\simplilearn\\Phase\_5\_FSD\\F SD-5 WorkSpace-Practice- Projects\\AutomateWebApp\\screenshot\\checkBox.png"));

}

@Test

@DisplayName("Test :: Verify Cancel Button")

void testCancelButton() throws InterruptedException, IOException {

//Find Input Field

WebElement cnc = driver.findElement(By.className("cancelbtn"));

assertTrue(cnc.isDisplayed()); assertTrue(cnc.isEnabled());

Dimension d = new Dimension(480, 620); driver.manage().window().setSize(d);

cnc.submit(); Thread.sleep(2000);

//Step-1 :: type cast driver object into takeScreenshot instance

TakesScreenshot tsc = (TakesScreenshot)

driver;

//Step-2 :: Generate a screenshot as file File rsc =

tsc.getScreenshotAs(OutputType.FILE);

FileHandler.copy(rsc,new File("C:\\Users\\nayak\\Desktop\\simplilearn\\Phase\_5\_FSD\\F SD-5 WorkSpace-Practice- Projects\\AutomateWebApp\\screenshot\\cancel.png"));

}

@Test

@DisplayName("Test :: Verify Link")

void testLink() throws InterruptedException, IOException

{

//Find Input Field

WebElement link =

driver.findElement(By.cssSelector("body > form > fieldset > div:nth-child(3) > span > a"));

assertTrue(link.isDisplayed()); assertTrue(link.isEnabled());

Dimension d = new Dimension(480, 620); driver.manage().window().setSize(d); link.click();

Thread.sleep(2000);

//Step-1 :: type cast driver object into takeScreenshot instance

TakesScreenshot tsc = (TakesScreenshot)

driver;

//Step-2 :: Generate a screenshot as file File rsc

tsc.getScreenshotAs(OutputType.FILE);

FileHandler.copy(rsc,new File("C:\\Users\\nayak\\Desktop\\simplilearn\\Phase\_5\_FSD\\F SD-5 WorkSpace-Practice- Projects\\AutomateWebApp\\screenshot\\link- verification.png"));

}

}

# RegistrationFormTestCases.java

package com.simplilearn.webapp.registration;

import static org.junit.jupiter.api.Assertions.assertEquals; import static org.junit.jupiter.api.Assertions.assertTrue;

import java.io.File;

import java.io.IOException;

import org.junit.jupiter.api.AfterEach; import org.junit.jupiter.api.BeforeEach; import org.junit.jupiter.api.DisplayName; import org.junit.jupiter.api.Test; import org.openqa.selenium.By;

import org.openqa.selenium.Dimension; import org.openqa.selenium.OutputType; import org.openqa.selenium.TakesScreenshot; import org.openqa.selenium.WebDriver; import org.openqa.selenium.WebElement; import org.openqa.selenium.edge.EdgeDriver; import org.openqa.selenium.io.FileHandler;

public class RegistrationFormTestCases {

final String siteURL = "file:///C:/Users/nayak/Desktop/simplilearn/Phase\_5\_FSD/ FSD-5%20WorkSpace-Practice- Projects/AutomateWebApp/static/registrationForm.html";

final String driverPath ="driver\\msedgedriver.exe"; WebDriver driver;

@BeforeEach

void setUp() throws Exception {

//Set Up

System.setProperty("webdriver.edge.driver",driverPath); driver = new EdgeDriver(); driver.get(siteURL);

//Add maximize size for snapshot driver.manage().window().maximize();

}

@AfterEach

void tearDown() throws Exception { driver.close();

}

@Test

@DisplayName("Registration Form :: Email Verification")

void testEmail() throws InterruptedException {

//Find Input Field WebElement input =

driver.findElement(By.id("email")); assertTrue(input.isDisplayed()); assertTrue(input.isEnabled());

//Enter yourEmail input.sendKeys(["nay](mailto:nayaksofia@gmail.com)a[ksofia@gmail.com](mailto:nayaksofia@gmail.com)"); Thread.sleep(2000);

}

@Test

@DisplayName("Registration Form :: Password Verification")

void testPassword() throws InterruptedException, IOException {

//Find Input Field WebElement input =

driver.findElement(By.id("psw")); assertTrue(input.isDisplayed()); assertTrue(input.isEnabled());

//Enter yourEmail input.sendKeys("abcd1234"); Thread.sleep(2000);

Dimension d = new Dimension(480, 800); driver.manage().window().setSize(d);

//Step-1 :: type cast driver object into takeScreenshot instance

TakesScreenshot tsc = (TakesScreenshot) driver;

//Step-2 :: Generate a screenshot as file

File rsc = tsc.getScreenshotAs(OutputType.FILE); FileHandler.copy(rsc,new

File("C:\\Users\\nayak\\Desktop\\simplilearn\\Phase\_5\_FS D\\FSD-5 WorkSpace-Practice- Projects\\AutomateWebApp\\screenshot\\reg- pasword.png"));

}

@Test

@DisplayName("Registration Form :: Repeat Password Verification")

void testRepeatPassword() throws InterruptedException, IOException {

//Find Input Field WebElement input =

driver.findElement(By.id("psw-repeat")); assertTrue(input.isDisplayed()); assertTrue(input.isEnabled());

//Enter yourEmail input.sendKeys("abcd1234"); Thread.sleep(2000);

Dimension d = new Dimension(480, 800); driver.manage().window().setSize(d);

//Step-1 :: type cast driver object into takeScreenshot instance

TakesScreenshot tsc = (TakesScreenshot) driver;

//Step-2 :: Generate a screenshot as file

File rsc = tsc.getScreenshotAs(OutputType.FILE); FileHandler.copy(rsc,new

File("C:\\Users\\nayak\\Desktop\\simplilearn\\Phase\_5\_FS D\\FSD-5 WorkSpace-Practice-

Projects\\AutomateWebApp\\screenshot\\reg-repet- password.png"));

}

@Test

@DisplayName("RegistrationTest :: To verify radio button exist.")

void testToVerifyRadioButtonExist() throws InterruptedException, IOException {

WebElement radio1 = driver.findElement(By.id("admin"));

assertEquals(true, radio1.isDisplayed()); assertEquals(true, radio1.isEnabled()); assertEquals(false, radio1.isSelected());

WebElement radio2 = driver.findElement(By.id("user"));

assertEquals(true, radio2.isDisplayed()); assertEquals(true, radio2.isEnabled()); assertEquals(false, radio2.isSelected());

WebElement radio3 = driver.findElement(By.id("vendor"));

assertEquals(true, radio3.isDisplayed()); assertEquals(true, radio3.isEnabled()); assertEquals(false, radio3.isSelected());

Thread.sleep(2000);

Dimension d = new Dimension(480, 800); driver.manage().window().setSize(d);

//Step-1 :: type cast driver object into takeScreenshot instance

TakesScreenshot tsc = (TakesScreenshot) driver;

//Step-2 :: Generate a screenshot as file

File rsc = tsc.getScreenshotAs(OutputType.FILE); FileHandler.copy(rsc,new

File("C:\\Users\\nayak\\Desktop\\simplilearn\\Phase\_5\_FS D\\FSD-5 WorkSpace-Practice- Projects\\AutomateWebApp\\screenshot\\reg-radio- buttons.png"));

}

@Test

@DisplayName("Test : To verify radio button-1 select.")

void testToVerifyRadioButtonOneSelected() throws InterruptedException, IOException {

WebElement radio = driver.findElement(By.id("admin"));

assertEquals(true, radio.isDisplayed()); assertEquals(true, radio.isEnabled());

//select radio button radio.click(); Thread.sleep(2000);

assertEquals(true, radio.isSelected());

//Snapshot:: size

Dimension d = new Dimension(480, 800); driver.manage().window().setSize(d);

//Step-1 :: type cast driver object into takeScreenshot instance

TakesScreenshot tsc = (TakesScreenshot) driver;

//Step-2 :: Generate a screenshot as file

File rsc = tsc.getScreenshotAs(OutputType.FILE); FileHandler.copy(rsc,new

File("C:\\Users\\nayak\\Desktop\\simplilearn\\Phase\_5\_FS D\\FSD-5 WorkSpace-Practice- Projects\\AutomateWebApp\\screenshot\\radio-button- 1.png"));

}

@Test

@DisplayName("Test : To verify radio button-2 select.")

void testToVerifyRadioButtonTwoSelected() throws InterruptedException, IOException {

WebElement radio = driver.findElement(By.id("user"));

assertEquals(true, radio.isDisplayed()); assertEquals(true, radio.isEnabled());

//select radio button radio.click(); Thread.sleep(2000);

assertEquals(true, radio.isSelected());

//Snapshot:: size

Dimension d = new Dimension(480, 800); driver.manage().window().setSize(d);

//Step-1 :: type cast driver object into takeScreenshot instance

TakesScreenshot tsc = (TakesScreenshot) driver;

//Step-2 :: Generate a screenshot as file

File rsc = tsc.getScreenshotAs(OutputType.FILE); FileHandler.copy(rsc,new

File("C:\\Users\\nayak\\Desktop\\simplilearn\\Phase\_5\_FS D\\FSD-5 WorkSpace-Practice- Projects\\AutomateWebApp\\screenshot\\radio-button- 2.png"));

}

@Test

@DisplayName("Test : To verify radio button-3 select.")

void testToVerifyRadioButtonSelected() throws InterruptedException, IOException {

WebElement radio = driver.findElement(By.id("vendor"));

assertEquals(true, radio.isDisplayed()); assertEquals(true, radio.isEnabled());

//select radio button radio.click(); Thread.sleep(2000);

assertEquals(true, radio.isSelected());

//Snapshot:: size

Dimension d = new Dimension(480, 800); driver.manage().window().setSize(d);

//Step-1 :: type cast driver object into takeScreenshot instance

TakesScreenshot tsc = (TakesScreenshot)

driver;

//Step-2 :: Generate a screenshot as file File rsc =

tsc.getScreenshotAs(OutputType.FILE);

FileHandler.copy(rsc,new File("C:\\Users\\nayak\\Desktop\\simplilearn\\Phase\_5\_FS D\\FSD-5 WorkSpace-Practice- Projects\\AutomateWebApp\\screenshot\\radio-button- 3.png"));

}

@Test

@DisplayName("Test : To verify checkbox exist.") void testToVerifyCheckboxExist() throws IOException

{

WebElement checkBox1 =

driver.findElement(By.id("skill1")); assertEquals(true,checkBox1.isDisplayed()); assertEquals(true,checkBox1.isEnabled()); assertEquals(false,checkBox1.isSelected());

WebElement checkBox2 = driver.findElement(By.id("skill2"));

assertEquals(true,checkBox2.isDisplayed()); assertEquals(true,checkBox2.isEnabled()); assertEquals(false,checkBox2.isSelected());

WebElement checkBox3 = driver.findElement(By.id("skill3"));

assertEquals(true,checkBox3.isDisplayed()); assertEquals(true,checkBox3.isEnabled()); assertEquals(false,checkBox3.isSelected());

//Snapshot:: size

Dimension d = new Dimension(480, 800); driver.manage().window().setSize(d);

//Step-1 :: type cast driver object into takeScreenshot instance

TakesScreenshot tsc = (TakesScreenshot)

driver;

//Step-2 :: Generate a screenshot as file File rsc =

tsc.getScreenshotAs(OutputType.FILE);

FileHandler.copy(rsc,new File("C:\\Users\\nayak\\Desktop\\simplilearn\\Phase\_5\_FS D\\FSD-5 WorkSpace-Practice- Projects\\AutomateWebApp\\screenshot\\reg-check- box.png"));

}

@Test

@DisplayName("Test :: To verify checkbox1 select") void testToVerifyCheckBoxOneSelected() throws

InterruptedException, IOException { WebElement checkBox1 =

driver.findElement(By.id("skill1")); assertEquals(true,checkBox1.isDisplayed());

assertEquals(true,checkBox1.isEnabled());

//Select Check Box One checkBox1.click(); Thread.sleep(2000);

assertEquals(true,checkBox1.isSelected());

//Snapshot:: size

Dimension d = new Dimension(480, 800); driver.manage().window().setSize(d);

//Step-1 :: type cast driver object into takeScreenshot instance

TakesScreenshot tsc = (TakesScreenshot)

driver;

//Step-2 :: Generate a screenshot as file File rsc =

tsc.getScreenshotAs(OutputType.FILE);

FileHandler.copy(rsc,new File("C:\\Users\\nayak\\Desktop\\simplilearn\\Phase\_5\_FS D\\FSD-5 WorkSpace-Practice- Projects\\AutomateWebApp\\screenshot\\reg-check-box- 1.png"));

}

@Test

@DisplayName("Test :: To verify checkbox2 select") void testToVerifyCheckBoxTwoSelected() throws

InterruptedException, IOException { WebElement checkBox2 =

driver.findElement(By.id("skill2")); assertEquals(true,checkBox2.isDisplayed()); assertEquals(true,checkBox2.isEnabled());

//Select Check Box One checkBox2.click(); Thread.sleep(2000);

assertEquals(true,checkBox2.isSelected());

//Snapshot:: size

Dimension d = new Dimension(480, 800); driver.manage().window().setSize(d);

//Step-1 :: type cast driver object into takeScreenshot instance

TakesScreenshot tsc = (TakesScreenshot)

driver;

//Step-2 :: Generate a screenshot as file File rsc =

tsc.getScreenshotAs(OutputType.FILE);

FileHandler.copy(rsc,new File("C:\\Users\\nayak\\Desktop\\simplilearn\\Phase\_5\_FS D\\FSD-5 WorkSpace-Practice-

Projects\\AutomateWebApp\\screenshot\\reg-check-box- 2.png"));

}

@Test

@DisplayName("Test :: To verify checkbox3 select") void testToVerifyCheckBoxThreeSelected() throws

InterruptedException, IOException { WebElement checkBox3 =

driver.findElement(By.id("skill3")); assertEquals(true,checkBox3.isDisplayed()); assertEquals(true,checkBox3.isEnabled());

//Select Check Box One checkBox3.click(); Thread.sleep(2000);

assertEquals(true,checkBox3.isSelected());

//Snapshot:: size

Dimension d = new Dimension(480, 800); driver.manage().window().setSize(d);

//Step-1 :: type cast driver object into takeScreenshot instance

TakesScreenshot tsc = (TakesScreenshot) driver;

//Step-2 :: Generate a screenshot as file

File rsc = tsc.getScreenshotAs(OutputType.FILE); FileHandler.copy(rsc,new

File("C:\\Users\\nayak\\Desktop\\simplilearn\\Phase\_5\_FS D\\FSD-5 WorkSpace-Practice- Projects\\AutomateWebApp\\screenshot\\reg-check-box- 3.png"));

}

@Test

@DisplayName("Test :: To Verify Registration Button")

void testToVerifyRegistrationButton() throws InterruptedException, IOException {

WebElement button = driver.findElement(By.className("registerbtn"));

assertEquals(true,button.isDisplayed()); assertEquals(true,button.isEnabled());

//Select Check Box One button.submit(); Thread.sleep(2000);

//Snapshot:: size

Dimension d = new Dimension(480, 800); driver.manage().window().setSize(d);

//Step-1 :: type cast driver object into takeScreenshot instance

TakesScreenshot tsc = (TakesScreenshot) driver;

//Step-2 :: Generate a screenshot as file

File rsc = tsc.getScreenshotAs(OutputType.FILE); FileHandler.copy(rsc,new

File("C:\\Users\\nayak\\Desktop\\simplilearn\\Phase\_5\_FS D\\FSD-5 WorkSpace-Practice- Projects\\AutomateWebApp\\screenshot\\reg-button.png"));

}

}

# Pom.xml

<?xml version="1.0" encoding="UTF-8"?>

<project xmlns=["http://mave](http://maven.apache.org/POM/4.0.0)n[.apache.org/POM/4.0.0](http://maven.apache.org/POM/4.0.0)" xmlns:xsi=["h](http://www.w3.org/2001/XMLSchema-instance)t[tp://www.w3.org/2001/XMLSchema-instance"](http://www.w3.org/2001/XMLSchema-instance)

xsi:schemaLocation=["http:](http://maven.apache.org/POM/4.0.0)/[/maven.apache.org/POM/4.0.0](http://maven.apache.org/POM/4.0.0) [http://maven.apache.org/xsd/maven-4.0.0.xsd"](http://maven.apache.org/xsd/maven-4.0.0.xsd)>

<modelVersion>4.0.0</modelVersion>

<groupId>com.simplilearn.webapp</groupId>

<artifactId>AutomateWebApp</artifactId>

<version>0.0.1-SNAPSHOT</version>

<name>AutomateWebApp</name>

<!-- FIXME change it to the project's website -->

<url>[http://www.example.com](http://www.example.com/)</url>

<properties>

<project.build.sourceEncoding>UTF- 8</project.build.sourceEncoding>

<maven.compiler.source>8</maven.compiler.source>

<maven.compiler.target>8</maven.compiler.target>

</properties>

<dependencies>

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-api</artifactId>

<version>5.4.2</version>

<scope>test</scope>

</dependency>

<!--junit-jupiter-engine-->

<dependency>

<groupId>org.junit.jupiter</groupId>

<artifactId>junit-jupiter-engine</artifactId>

<version>5.4.2</version>

<scope>test</scope>

</dependency>

<!--selenium-java-->

<dependency>

<groupId>org.seleniumhq.selenium</groupId>

<artifactId>selenium-java</artifactId>

<version>3.141.59 </version>

</dependency>

</dependencies>

<build>

<!-- Add Maven jUnit5 Surefire Plugin -->

<plugins>

<plugin>

<groupId>org.apache.maven.plugins</groupId>

<artifactId>maven-surefire- plugin</artifactId>

<version>3.0.0-M5</version>

</plugin>

</plugins>

</build>

</project>

# LoginForm.html

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>LogIn Form</title>

<style type="text/css">

/\* Bordered form \*/ form {

border: 3px solid #f1f1f1;

/\*Make the form in center place, using max-width:500px and margin:auto; \*/

max-width: 600px; margin: auto;

}

/\* Full-width inputs \*/ input[type=text], input[type=password] {

width: 100%; padding: 12px 20px; margin: 8px 0;

display: inline-block; border: 1px solid #ccc; box-sizing: border-box;

}

/\* Set a style for all buttons \*/ button {

background-color: #04AA6D; color: white;

padding: 14px 20px; margin: 8px 0; border: none; cursor: pointer; width: 100%;

}

/\* Add a hover effect for buttons \*/ button:hover {

opacity: 0.8;

}

/\* Extra style for the cancel button (red) \*/

.cancelbtn { width: auto;

padding: 10px 18px; background-color: #f44336;

}

/\* Add padding to containers \*/

.container { padding: 16px;

}

/\* The "Forgot password" text \*/ span.psw {

float: right; padding-top: 16px;

}

/\* Change styles for span and cancel button on extra small screens \*/

@media screen and (max-width: 300px) { span.psw {

display: block; float: none;

}

.cancelbtn { width: 100%;

}

}

</style>

</head>

<body>

<form action="/login" method="post">

<fieldset>

<div class="container">

<label for="uname"><b>Username</b></label>

<input type="text" placeholder="Enter Username" name="uname" required> <br><br>

<label for="psw"><b>Password</b></label>

<input type="password" placeholder="Enter Password" name="psw" required><br><br>

<button type="submit">Login</button> <br>

<label>

<input type="checkbox" checked="checked" name="remember"> Remember me

</label>

</div>

<hr>

<div class="container" style="background- color:#f1f1f1">

<button type="button" class="cancelbtn">Cancel</button>

<span class="psw">Forgot <a href="#">password?</a></span>

</div>

</fieldset>

</form>

</body>

</html>

# RegistrationForm.html

<!DOCTYPE html>

<html>

<head>

<meta charset="ISO-8859-1">

<title>Register here</title>

<style type="text/css">

form {

border: 3px solid #f1f1f1;

/\*Make the form in center place, using max-width:500px and margin:auto; \*/

max-width: 700px; margin: auto;

}

.container {

padding: 1px;

background-color: #f1f1f1;

}

/\* Full-width input fields \*/ input[type=text], input[type=password] {

width: 95%; padding: 15px;

margin: 5px 0 22px 0; display: inline-block; border: none; background: light;

}

input[type=text]:focus, input[type=password]:focus { background-color: #ddd;

outline: none;

}

.registerbtn{ padding: 16px 20px; margin: 8px 0; border: none; cursor: pointer; width: 100%;

opacity: 0.9;

background-color: #04AA6D; color: white;

}

</style>

</head>

<body>

<form action = "/register">

<div class = "container">

<fieldset>

<h1>Registration Form </h1>

<h3> Please fill in this form to create an account:

</h3><br>

<label for ="email"><b>Email:</b></label>

<input type ="text" placeholder = "Enter Email" name="email" id="email" required><br><br>

<label for ="psw"><b>Password:</b></label>

<input type ="password" placeholder = "Enter Password" name="psw" id="psw" required><br><br>

<label for ="psw-repeat"><b>Repeat Password:</b></label>

<input type ="password" placeholder = "Repeat Password" name="psw-repeat" id="psw-repeat" required><br><br>

<label for="role"><b>Select Role</b></label>

<input type="radio" name="radio" id="admin" required> Admin

<input type="radio" name="radio" id="user" required> User

<input type="radio" name="radio" id="vendor" required> Vendor <br> <br>

<label for="skill"><b>Select Skill:</b></label>

<input type="checkbox" name ="skill" id="skill1" required> Java

<input type="checkbox" name ="skill" id="skill2" required> MySql

<input type="checkbox" name ="skill" id="skill3" required> Selenium Webdriver<br><br>

<button type="submit" class="registerbtn">Register</button>

</fieldset>

</div>

</form>

</body>

</html>