package com.example;

import io.github.bonigarcia.wdm.WebDriverManager;

import org.openqa.selenium.\*;

import org.openqa.selenium.chrome.ChromeDriver;

import org.openqa.selenium.chrome.ChromeOptions;

import org.openqa.selenium.support.ui.\*;

import org.testng.Assert;

import org.testng.annotations.\*;

import java.time.Duration;

import java.util.HashMap;

import java.util.Map;

public class AppTest {

    WebDriver driver;

    WebDriverWait wait;

    By userId = By.id("formEmail");

    By password = By.id("formPassword");

    By loginBtn = By.className("login-button");

    By eyeIcon = By.className("passowrd-visible");

    By errorMsg = By.xpath("//\*[contains(text(),'Invalid') or contains(text(),'invalid') or contains(text(),'incorrect')]");

    @BeforeMethod

        public void setUp() {

        Map<String, Object> prefs = new HashMap<>();

        prefs.put("profile.default\_content\_setting\_values.notifications", 1);

        ChromeOptions options = new ChromeOptions();

        options.setExperimentalOption("prefs", prefs);

        options.addArguments("--disable-popup-blocking");

        options.addArguments("start-maximized");

        driver = new ChromeDriver(options);

        wait = new WebDriverWait(driver, Duration.ofSeconds(10));

        driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(5));

        driver.get("[https://dev-dash.janitri.in](https://dev-dash.janitri.in/)");

        try {

            WebElement reloadButton = wait.until(ExpectedConditions.visibilityOfElementLocated(

                By.xpath("//button[contains(text(),'Reload')]")));

            reloadButton.click();

        } catch (TimeoutException ignored) {

        }

    }

    @Test

    public void testLoginWithBlankFieldsShowsError() {

        driver.findElement(loginBtn).click();

        WebElement error = wait.until(ExpectedConditions.visibilityOfElementLocated(errorMsg));

        Assert.assertTrue(error.isDisplayed(), "Error message should appear for blank login.");

    }

    @Test

    public void testPasswordMaskToggle() {

        WebElement pwdField = driver.findElement(password);

        pwdField.sendKeys("test123");

        Assert.assertEquals(pwdField.getAttribute("type"), "password", "Password should be masked initially.");

        driver.findElement(eyeIcon).click();

        Assert.assertEquals(pwdField.getAttribute("type"), "text", "Password should be visible after toggle.");

    }

    @Test

    public void testInvalidLoginShowsErrorMessage() {

        driver.findElement(userId).sendKeys("[wrong@janitri.in](mailto:wrong@janitri.in)");

        driver.findElement(password).sendKeys("wrongpass");

        driver.findElement(loginBtn).click();

        WebElement error = wait.until(ExpectedConditions.visibilityOfElementLocated(errorMsg));

        Assert.assertTrue(error.isDisplayed(), "Error message should be displayed for invalid login.");

    }

    @AfterMethod

    public void tearDown() {

        if (driver != null) driver.quit();

    }

}