//Feature1

Feature: Checking the functionality of viewing a product

Scenario: While on homepage navigate to fish oils page and browse the first product

Given I am on the homepage

When I click on supplement

And I click on Digestive Probiotic

And I click on fish oil

And I click on the first product

Then Verify the label  
  
//Feature2

Feature: check register the pharmacy Feature

Scenario: navigate to the register the pharmacy page from homepage and check the functionality

Given I am on the homepage

When I click on Blogs

And I scroll down to the footer

And I click on register the pharmacy under the useful links section

And Enter your name

And Enter the phone number "91 9876543211"

Then click on submit button  
  
  
//PageActions.java

package pages;

import java.time.Duration;

import org.openqa.selenium.JavascriptExecutor;

import org.openqa.selenium.WebDriver;

import com.aventstack.extentreports.ExtentTest;

import com.aventstack.extentreports.Status;

import stepdefinition.Hooks;

import uistore.PageLocators;

import utils.ExcelFileReader;

import utils.LoggerHandler;

import utils.Reporter;

import utils.Screenshot;

import utils.WebDriverHelper;

public class PageActions {

    WebDriver driver;

    WebDriverHelper helper;

    PageLocators locators = new PageLocators();

    Screenshot screenshot;

    ExtentTest test1 = Hooks.reports.createTest("Testcase 1");

    ExtentTest test2 = Hooks.reports.createTest("Testcase 2");

    public PageActions(WebDriver driver){

        try{

            this.driver = driver;

            if(driver != null){

                helper = new WebDriverHelper(driver);

                screenshot = new Screenshot(driver);

            }

        }

        catch(Exception e){

            e.printStackTrace();

        }

    }

    public void clickOnSupplements(){

        driver.manage().timeouts().pageLoadTimeout(Duration.ofSeconds(30));

        helper.clickOn(locators.supplementBtn);

        test1.log(Status.PASS, "click on Supplement");

    }

    public void clickOnDigestive(){

        driver.manage().timeouts().pageLoadTimeout(Duration.ofSeconds(30));

        helper.clickOn(locators.digestiveBtn);

    }

    public void clickOnFishOil(){

        driver.manage().timeouts().pageLoadTimeout(Duration.ofSeconds(30));

        helper.clickOn(locators.fishOil);

    }

    public void clickOnFirstProduct(){

        driver.manage().timeouts().pageLoadTimeout(Duration.ofSeconds(30));

        helper.clickOn(locators.firstProduct);

    }

    public void clickBlogs(){

        helper.clickOn(locators.navBlogs);

        Screenshot.captureScreenshotWithTimeStamp("screenshot\_01");

        Screenshot.takeScreenshot("screenshot\_01");

    }

    public void scrollAndClickOnRegisterButton(){

        try{

        driver.manage().timeouts().pageLoadTimeout(Duration.ofSeconds(30));

        helper.elementWait(locators.registerPharmacy);

        // Thread.sleep(1000);

        JavascriptExecutor js = (JavascriptExecutor) driver;

        js.executeScript("window.scrollTo(0, document.body.scrollHeight)");

        // js.executeScript("window.scrollBy(0, -200)", “”);

        // js.executeScript("window.scrollTo(0, document.body.scrollHeight)");

        helper.hoverOverElement(locators.registerPharmacy);

        }

        catch(Exception e){

            e.printStackTrace();

        }

        helper.clickOn(locators.registerPharmacy);

    }

    public void enterFullName(){

        driver.manage().timeouts().pageLoadTimeout(Duration.ofSeconds(30));

        helper.sendText(locators.fullName,  ExcelFileReader.readData(0, 1, 2));

    }

    public void enterPhoneNumber(){

        helper.sendText(locators.phoneNumber,  ExcelFileReader.readData(0, 1, 3));

    }

    public void clickSubmitButton(){

        try{

            test2 = Hooks.reports.createTest("Testcase 2");

            Thread.sleep(3000);

            helper.clickOn(locators.submitBtn);

            LoggerHandler.info("click on Submit");

            Screenshot.captureScreenshotWithTimeStamp("arogga\_screenshot");

            Screenshot.takeScreenshot("arogga\_screenshot");

            // Reporter.attachScreenshotToReport(test2,"arogga\_screenshot" ,"arogga\_screenshot");

            Reporter.attachScreenshotToReport("arogga\_screenshot", test1, "arogga\_screenshot");

        }

        catch(Exception e){

            e.printStackTrace();

        }

    }

    public void verifyOTP(){

       String text = driver.findElement(locators.verifyOtp).getText();

       assert(text.contains("OTP!"));

       Screenshot.takeScreenshot("arogga\_screenshot");

       Screenshot.captureScreenshotWithTimeStamp("arogga\_screenshot");

    //    Reporter.attachScreenshotToReport(test, "arogga\_screenshot", "screenshot attached");

    }

}

//TestRunner.java

package runner;

import org.junit.runner.RunWith;

import io.cucumber.junit.Cucumber;

import io.cucumber.junit.CucumberOptions;

@RunWith(Cucumber.class)

@CucumberOptions(

    features = "features",

    glue = "stepdefinition"

)

public class TestRunner {

}

//Hooks

package stepdefinition;

import com.aventstack.extentreports.ExtentReports;

import io.cucumber.java.After;

import io.cucumber.java.Before;

import utils.Base;

import utils.LoggerHandler;

import utils.Reporter;

public class Hooks extends Base{

    public static ExtentReports reports;

    @Before

    public void setUp(){

        openBrowser();

        reports = Reporter.generateExtentReport();

        for(int i = 0; i<45; i++){

            LoggerHandler.info("click on Submit");

            LoggerHandler.warn("Logging warning message");

        }

    }

    @After

    public void tearDown(){

        if(driver!= null){

            driver.quit();

        }

        reports.flush();

    }

}

//StepDef

package stepdefinition;

import io.cucumber.java.en.Given;

import io.cucumber.java.en.Then;

import io.cucumber.java.en.When;

import pages.PageActions;

import utils.Base;

public class StepDefinition {

    public PageActions arogga = new PageActions(Base.driver);

    @Given("I am on the homepage")

    public void i\_am\_on\_the\_homepage() {

    }

    @When("I click on supplement")

    public void i\_click\_on\_supplement() {

        arogga.clickOnSupplements();

    }

    @When("I click on Digestive Probiotic")

    public void i\_click\_on\_digestive\_probiotic() {

        arogga.clickOnDigestive();

    }

    @When("I click on fish oil")

    public void i\_click\_on\_fish\_oil() {

        arogga.clickOnFishOil();

    }

    @When("I click on the first product")

    public void i\_click\_on\_the\_first\_product() {

        arogga.clickOnFirstProduct();

    }

    @Then("Verify the label")

    public void verify\_the\_label() {

    }

    @When("I click on Blogs")

    public void i\_click\_on\_blogs() {

        arogga.clickBlogs();

    }

    @When("I scroll down to the footer")

    public void i\_scroll\_down\_to\_the\_footer() {

        arogga.scrollAndClickOnRegisterButton();

    }

    @When("I click on register the pharmacy under the useful links section")

    public void i\_click\_on\_register\_the\_pharmacy\_under\_the\_useful\_links\_section() {

    }

    @When("Enter your name")

    public void enter\_your\_name() {

        arogga.enterFullName();

    }

    @When("Enter the phone number {string}")

    public void enter\_the\_phone\_number(String string) {

        arogga.enterPhoneNumber();

    }

    @Then("click on submit button")

    public void click\_on\_submit\_button() {

        arogga.clickSubmitButton();

    }

}

//PageLocators

package uistore;

import org.openqa.selenium.By;

public class PageLocators {

    public By supplementBtn = By.xpath("//a[text() = 'Supplement']");

    public By digestiveBtn = By.xpath("(//a[text() = 'Digestive Probiotic '])[2]//parent::div");

    public By fishOil = By.xpath("//a[text() = 'Fish Oil']//parent::div");

    public By firstProduct = By.xpath("//img[@alt = 'Himega  ']//parent::div");

    public By navBlogs = By.xpath("//p[text() = 'Blogs']//parent::div//parent::a");

    public By registerPharmacy = By.xpath("//div[text() = 'Register the Pharmacy']");

    public By fullName = By.xpath("//input[contains(@placeholder, 'name')]");

    public By phoneNumber = By.xpath("//input[contains(@placeholder, 'number')]");

    public By submitBtn = By.xpath("(//span[contains(text(), 'Submit')]//parent::div//parent::div)[1]");

    public By verifyOtp = By.xpath("//div[text() = 'Please input your OTP!']");

}

//Base

package utils;

import java.net.URL;

import java.time.Duration;

import java.util.Properties;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.chrome.ChromeOptions;

import org.openqa.selenium.remote.RemoteWebDriver;

import org.openqa.selenium.support.events.EventFiringDecorator;

import org.openqa.selenium.support.events.WebDriverListener;

public class Base {

    public static Properties prop;

    public static WebDriver driver;

    public static void openBrowser(){

        try{

            prop = new Properties();

            ChromeOptions opt = new ChromeOptions();

            opt.addArguments("--disable-notifications");

            opt.addArguments("--start-maximized");

            PropertiesReader.loadProps();

            driver = new RemoteWebDriver(new URL(PropertiesReader.remURL), opt);

            WebDriverListener listener = new EventHandler();

            driver = new EventFiringDecorator<>(listener).decorate(driver);

            driver.get(PropertiesReader.appURL);

            driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(30));

            driver.manage().timeouts().pageLoadTimeout(Duration.ofSeconds(60));

        }

        catch(Exception e){

            e.printStackTrace();

        }

    }

}

//EventHandler

package utils;

import org.openqa.selenium.support.events.WebDriverListener;

public class EventHandler implements WebDriverListener {

}

//ExcelFileReader

package utils;

import java.io.FileInputStream;

import org.apache.poi.xssf.usermodel.XSSFCell;

import org.apache.poi.xssf.usermodel.XSSFRow;

import org.apache.poi.xssf.usermodel.XSSFSheet;

import org.apache.poi.xssf.usermodel.XSSFWorkbook;

public class ExcelFileReader {

    public static String cellValue;

    public static String readData(int sheetNum, int rowNum, int cellNum){

        try{

            FileInputStream fis = new FileInputStream(System.getProperty("user.dir")+"/testdata/data.xlsx");

            XSSFWorkbook wb = new XSSFWorkbook(fis);

            XSSFSheet sheet = wb.getSheetAt(sheetNum);

            XSSFRow row = sheet.getRow(rowNum);

            XSSFCell cell = row.getCell(cellNum);

            cellValue = cell.getStringCellValue();

        }

        catch(Exception e){

            e.printStackTrace();

        }

        return cellValue;

    }

}

//LoggerHandler

package utils;

import java.text.SimpleDateFormat;

import java.util.Date;

import org.apache.log4j.FileAppender;

import org.apache.log4j.Logger;

import org.apache.log4j.PatternLayout;

public class LoggerHandler {

  private static final Logger logger = Logger.getLogger(LoggerHandler.class);

    static {

        try {

            // Set up a file appender with a timestamp in the filename

            String timestamp = new SimpleDateFormat("yyyy.MM.dd.HH.mm.ss").format(new Date());

            // String logFileName = "logs/logfile\_" + timestamp + ".log";

            String logFileName = "logs/logfile\_" + timestamp + ".log";

            FileAppender fileAppender = new FileAppender(new PatternLayout("%d{ISO8601} %-5p %c - %m%n"), logFileName, true);

            logger.addAppender(fileAppender);

        } catch (Exception e) {

            logger.error("Failed to initialize logger file appender", e);

        }

    }

    public static void trace(String message) {

        logger.trace(message);

    }

    public static void debug(String message) {

        logger.debug(message);

    }

    public static void info(String message) {

        logger.info(message);

    }

    public static void warn(String message) {

        logger.warn(message);

    }

    public static void error(String message) {

        logger.error(message);

    }

    public static void fatal(String message) {

        logger.fatal(message);

    }

}

//Properties

package utils;

import java.io.FileInputStream;

import java.util.Properties;

public class PropertiesReader {

    public static Properties prop = new Properties();

    public static String appURL;

    public static String remURL;

    public static void loadProps(){

        try{

            FileInputStream fis = new FileInputStream(System.getProperty("user.dir")+"/config/browser.properties");

            prop.load(fis);

            appURL = prop.getProperty("applicationURL");

            remURL = prop.getProperty("remoteURL");

        }

        catch(Exception e){

            e.printStackTrace();

        }

    }

}

//Reporter.java

package utils;

import java.io.ByteArrayOutputStream;

import java.io.File;

import java.io.FileInputStream;

import java.io.FileOutputStream;

import java.io.IOException;

import java.util.Base64;

import java.util.Date;

import java.text.SimpleDateFormat;

import java.util.Properties;

import java.util.TimeZone;

import org.openqa.selenium.OutputType;

import org.openqa.selenium.TakesScreenshot;

import org.openqa.selenium.WebDriver;

import com.aventstack.extentreports.ExtentReports;

import com.aventstack.extentreports.ExtentTest;

import com.aventstack.extentreports.MediaEntityBuilder;

import com.aventstack.extentreports.Status;

import com.aventstack.extentreports.reporter.ExtentSparkReporter;

import com.aventstack.extentreports.reporter.configuration.Theme;

import com.google.common.io.Files;

public class Reporter extends Base {

    public static TakesScreenshot ts;

    public static Properties prop;

    public static ExtentReports extentReport;

    public static ExtentTest test;

    public static ExtentReports generateExtentReport() {

        return generateExtentReport(null);

    }

    public static ExtentReports generateExtentReport(String reportName) {

        if (extentReport == null) {

            extentReport = createExtentReport(reportName);

        }

        return extentReport;

    }

    private static ExtentReports createExtentReport(String reportName) {

        ExtentReports extentReport = new ExtentReports();

        // Load properties from browser.properties file

        String filepath = System.getProperty("user.dir") + "/config/browser.properties";

        try {

            FileInputStream file = new FileInputStream(filepath);

            prop = new Properties();

            prop.load(file);

        } catch (IOException e) {

            System.out.println(e.getLocalizedMessage());

        }

        // Get the current timestamp for the report name

        SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy.MM.dd.HH.mm.ss");

        TimeZone istTimeZone = TimeZone.getTimeZone("Asia/Kolkata"); // IST timezone

        dateFormat.setTimeZone(istTimeZone);

        String timestamp = dateFormat.format(new Date());

        // Define the report file path with the timestamp and provided report name

        String reportFilePath = System.getProperty("user.dir") + "/reports/";

        if (reportName == null || reportName.isEmpty()) {

            reportName = "Test Report";

        }

        reportFilePath += reportName + "\_" + timestamp + ".html";

        File extentReportFile = new File(reportFilePath);

        ExtentSparkReporter sparkReporter = new ExtentSparkReporter(extentReportFile);

        sparkReporter.config().setTheme(Theme.DARK);

        sparkReporter.config().setReportName("test Report");

        sparkReporter.config().setDocumentTitle("test Automation Report");

        sparkReporter.config().setTimeStampFormat("yyyy.MM.dd.HH.mm.ss");

        extentReport.attachReporter(sparkReporter);

        extentReport.setSystemInfo("Application URL", prop.getProperty("url"));

        extentReport.setSystemInfo("Browser Name", prop.getProperty("browserName"));

        extentReport.setSystemInfo("Email", prop.getProperty("validEmail"));

        extentReport.setSystemInfo("Password", prop.getProperty("validPassword"));

        extentReport.setSystemInfo("Operating System", System.getProperty("os.name"));

        extentReport.setSystemInfo("Username", System.getProperty("user.name"));

        extentReport.setSystemInfo("Java Version", System.getProperty("java.version"));

        return extentReport;

    }

    public static String captureScreenshotAsBase64(WebDriver driver, String screenshotName) {

        SimpleDateFormat dateFormat = new SimpleDateFormat("yyyy.MM.dd.HH.mm.ss");

        TimeZone istTimeZone = TimeZone.getTimeZone("Asia/Kolkata"); // IST timezone

        dateFormat.setTimeZone(istTimeZone);

        String timestamp = dateFormat.format(new Date());

        TakesScreenshot screenshotDriver = (TakesScreenshot) driver;

        byte[] screenshotBytes = screenshotDriver.getScreenshotAs(OutputType.BYTES);

        String base64Screenshot = "";

        try {

            ByteArrayOutputStream baos = new ByteArrayOutputStream();

            baos.write(screenshotBytes);

            base64Screenshot = Base64.getEncoder().encodeToString(baos.toByteArray());

            // Save the screenshot to a file for reference

            saveScreenshotToFile(screenshotBytes, screenshotName + "\_" + timestamp + ".png");

        } catch (IOException e) {

            e.printStackTrace();

        }

        return base64Screenshot;

    }

    private static String saveScreenshotToFile(byte[] screenshotBytes, String fileName) {

        String screenshotsDirPath = System.getProperty("user.dir") + "/reports/errorScreenshots/";

        try {

            File screenshotsDir = new File(screenshotsDirPath);

            if (!screenshotsDir.exists())

             {

                screenshotsDir.mkdirs();

            }

            String destinationScreenshotPath = screenshotsDirPath + fileName;

            FileOutputStream outputStream = new FileOutputStream(destinationScreenshotPath);

            outputStream.write(screenshotBytes);

            outputStream.close();

        }

        catch (IOException e) {

            e.printStackTrace();

        }

        String destinationScreenshotPath = screenshotsDirPath + fileName;

        return destinationScreenshotPath;

    }

    public static String captureScreenShot(String filename) {

        String timestamp = new SimpleDateFormat("yyyy.MM.dd.HH.mm.ss").format(new Date());

        String name = filename + timestamp + ".png";

        String destPath =  "./"+name;

        ts = (TakesScreenshot) driver;

        File file = ts.getScreenshotAs(OutputType.FILE);

        // Create the screenshots directory if it doesn't exist

        File screenshotsDir = new File(System.getProperty("user.dir") + "/reports");

        if (!screenshotsDir.exists()) {

            screenshotsDir.mkdirs();

        }

        File target = new File(screenshotsDir, name);

        try {

            Files.copy(file, target);

        } catch (IOException e) {

            e.printStackTrace();

        }

        return destPath;

    }

    public static void attachScreenshotToReport(String filename, ExtentTest test, String description) {

        try {

            test.log(Status.INFO, description, MediaEntityBuilder.createScreenCaptureFromPath(captureScreenShot(filename)).build());

        } catch (Exception e) {

            e.printStackTrace();

        }

    }

    public static void flushReport(){

        try {

            extentReport.flush();

        } catch (Exception e) {

            // TODO: handle exception

            e.printStackTrace();

        }

    }

}

//Screenshot

package utils;

import java.io.File;

import java.io.IOException;

import java.text.DateFormat;

import java.text.SimpleDateFormat;

import java.time.Duration;

import org.apache.commons.io.FileUtils;

import org.apache.poi.hpsf.Date;

import org.openqa.selenium.OutputType;

import org.openqa.selenium.TakesScreenshot;

import org.openqa.selenium.WebDriver;

public class Screenshot {

    public static WebDriver driver;

    public Screenshot(WebDriver driver){

        Screenshot.driver = driver;

    }

     public static void takeScreenshot(String fileName){

        try{

            File screenshot = ((TakesScreenshot) driver).getScreenshotAs(OutputType.FILE);

            File path = new File(System.getProperty("user.dir")+"/screenshots/"+fileName+".png");

            File parent = new File(path.getParent());

            if(!parent.exists()){

                parent.mkdirs();

            }

            FileUtils.copyFile(screenshot, path);

        }

        catch(Exception e){

            e.printStackTrace();

        }

    }

    public static void captureScreenshotWithTimeStamp(String fileName){

        try{

            String date = new SimpleDateFormat("yyyy.MM.dd.HH.mm.ss").format(new Date());

            File screenshot = ((TakesScreenshot) driver).getScreenshotAs(OutputType.FILE);

            File path = new File(System.getProperty("user.dir")+"/screenshots/"+fileName+"\_"+date+".png");

            File parent = new File(path.getParent());

            if(!parent.exists()){

                parent.mkdirs();

            }

            FileUtils.copyFile(screenshot, path);

        }

        catch(Exception e){

            e.printStackTrace();

        }

    }

    private DateFormat SimpleDateFormat(String string) {

        // TODO Auto-generated method stub

        throw new UnsupportedOperationException("Unimplemented method 'SimpleDateFormat'");

    }

}

//WebDriverHelper

package utils;

import java.io.File;

import java.io.IOException;

import java.time.Duration;

import java.util.Set;

import org.apache.commons.io.FileUtils;

import org.openqa.selenium.By;

import org.openqa.selenium.Keys;

import org.openqa.selenium.OutputType;

import org.openqa.selenium.TakesScreenshot;

import org.openqa.selenium.WebDriver;

import org.openqa.selenium.WebElement;

import org.openqa.selenium.interactions.Actions;

import org.openqa.selenium.support.ui.ExpectedConditions;

import org.openqa.selenium.support.ui.WebDriverWait;

public class WebDriverHelper {

    WebDriver driver;

    Actions action;

    public WebDriverHelper(WebDriver driver){

        this.driver = driver;

        action = new Actions(driver);

    }

    public void elementWait(By locBy){

        WebDriverWait wait = new WebDriverWait(driver, Duration.ofSeconds(30));

        wait.until(ExpectedConditions.visibilityOfElementLocated(locBy));

    }

    public void clickOn(By loc){

        WebElement element = driver.findElement(loc);

        element.click();

    }

    public void sendText(By loc, String text){

        driver.findElement(loc).sendKeys(text);

    }

    public void switchWindow(){

        String parent = driver.getWindowHandle();

        Set<String> allWindowSet = driver.getWindowHandles();

        for(String child: allWindowSet){

            if(!parent.equalsIgnoreCase(child)){

                driver.switchTo().window(child);

                break;

            }

        }

    }

    public void hoverOverElement(By loc){

        WebElement element = driver.findElement(loc);

        action.moveToElement(element).perform();

    }

    public void enterKey(By loc){

        driver.findElement(loc).sendKeys(Keys.ENTER);

    }

}