LoginPage:

```java
package PageObjects;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class LoginPage {

    public static WebDriver driver;

    public LoginPage(WebDriver driver)
    {
        this.driver=driver;
        PageFactory.initElements(driver,this);
    }

    @FindBy(xpath="//a[text()=\"Login\"]")
    public WebElement loginLink;

    @FindBy(xpath="//input[@id=\"username\"]")
    public WebElement username;

    @FindBy(xpath="//input[@id=\"password\"]")
    public WebElement password;

    @FindBy(xpath="//input[@value=\"Log In\"]")
    public WebElement loginBtn;

    @FindBy(xpath="//div[text()=\"Username and Password are required!!\"]")
    public WebElement errorMessage;

    @FindBy(xpath="//div[text()=\"Logout\"]")
    public WebElement logoutBtn;

}
```

ElectronicsProductDetailPage:

```java
package PageObjects;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class ElectronicsProductDetailPage {
    public static WebDriver driver;

    public ElectronicsProductDetailPage(WebDriver driver)
    {
        this.driver=driver;
        PageFactory.initElements(driver,this);
    }

    @FindBy(xpath="//div[text()=\"s21\"]")
    public WebElement s21name;
    @FindBy(xpath="//div[text()=\"65000\"]")
    public WebElement s21Price;

    @FindBy(xpath="//div[text()=\"Add To Cart\"]")
    public WebElement addToCartBtn;

    @FindBy(xpath="//a[text()=\"Go To Cart\"]")
    public WebElement gotoCartBtn;

}
```

ElectronicsCartPage:

```java
package PageObjects;

import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.support.FindBy;
import org.openqa.selenium.support.PageFactory;

public class ElectronicsCartPage {
    public static WebDriver driver;

    public ElectronicsCartPage(WebDriver driver)
    {
        this.driver=driver;
        PageFactory.initElements(driver,this);
    }

    @FindBy(xpath="//div[text()=\"s21\"]")
    public WebElement s21name;

    @FindBy(xpath="//div[text()=\"65000.00\"]")
    public WebElement s21price;

    @FindBy(xpath="//div[text()=\"Remove\"]")
    public WebElement removeBtn;

    @FindBy(xpath="//a[text()=\"Checkout\"]")
    public WebElement checkoutBtn;

    @FindBy(xpath="//a[text()=\"Continue Shopping\"]")
    public WebElement continueShopping;
}
```

HerokuappTestCases :

```java
 package AutomationScripts ;
import Context.TestContext;
import ObjectManager.DriverManager;
import PageObjects.*;
import com.microsoft.schemas.vml.STStrokeArrowType;
import dataProvider.ConfigFileReader;
import dataProvider.ReadWriteExcel;
import extentReport.ExtentReport;
import org.apache.log4j.PropertyConfigurator;
import org.apache.poi.xssf.usermodel.XSSFSheet;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriver;
import org.testng.ITestResult;
import org.testng.annotations.*;
import org.testng.asserts.SoftAssert;
import utils.Listener;
import utils.Logging;
import utils.Utility;

import java.io.IOException;
import java.time.Duration;

 @Listeners(Listener.class)
public class HerokuappTestCases {

    public static WebDriver driver;

    public static ExtentReport extentReport;

    TestContext testContext;
    ReadWriteExcel readWriteExcel;
    SoftAssert softAssert;
    public static LoginPage loginPage;
    public static EcommerceMenuPage ecommerceMenuPage;
    public static ElectronicsPage electronicsPage;
    public static KitchenItemsPage kitchenItemsPage;
    public static SportsPage sportsPage;

    public static ElectronicsProductDetailPage electronicsProductDetailPage;

    public static ElectronicsCartPage electronicsCartPage;

    @BeforeSuite
    public void setupSuite() throws IOException {
        driver= DriverManager.getDriver();
        driver.get(ConfigFileReader.getUrl());
        testContext=new TestContext();
        extentReport = new ExtentReport();
        readWriteExcel= new ReadWriteExcel();
        softAssert=new SoftAssert();
        loginPage= new LoginPage(driver);
        ecommerceMenuPage = new EcommerceMenuPage(driver);
        electronicsPage=new ElectronicsPage(driver);
        kitchenItemsPage= new KitchenItemsPage(driver);
        sportsPage=new SportsPage(driver);
        electronicsProductDetailPage=new ElectronicsProductDetailPage(driver);
        electronicsCartPage= new ElectronicsCartPage(driver);
        PropertyConfigurator.configure("src/main/resources/log4j.properties");
    }

    @AfterSuite
    public void afterSuite()
    {
        softAssert.assertAll();
        extentReport.flush();
    }

    @BeforeMethod()
    public void startTest()
    {
        Logging.info("starting the execution of test cases");
    }

    @AfterMethod()
    public void CloseTest(ITestResult result) throws IOException {
        Logging.info("Ending the test case Execution");
        if(ITestResult.FAILURE==result.getStatus())
        {
           Logging.info("test case is failed");
           extentReport.addScreenshot(driver);
        }


        if(driver.findElements(By.xpath("//div[text()=\"Logout\"]")).size()>0)
        {
            loginPage.logoutBtn.click();
            Logging.info("clicked on logout button");
        }
        else
        {
            Logging.info("Logout button is not displayed");
        }
    }

    @Test(description = "TC-01:verifies the validation message when user enters 
blank username and password")
    public void verifyErrorMessage() throws IOException {
        extentReport.createTest("TC-01:verifies the validation message when 
user enters blank username and password");
        loginPage.loginLink.click();
        Logging.info("user has clicked on login link ");
        extentReport.info("user has clicked on login link");
        loginPage.loginBtn.click();
        Logging.info("user has clicked on login button");
        extentReport.info("user has clicked on login button");

        if(loginPage.errorMessage.isDisplayed())
        {
            String actualErrormsg = loginPage.errorMessage.getText();
            String expectedErrormsg="Username and Password are required!!";
            softAssert.assertEquals(actualErrormsg,expectedErrormsg,"actual and 
expected error message is not same");
            Logging.info("username and password are required error message is 
displayed");
```

```
            extentReport.pass("username and password are required error message 
is displayed");
            extentReport.addScreenshot(driver);
            Logging.endTestCase();
        }

        else
        {
            Logging.info("username and password are required error message is 
not displayed");
            extentReport.fail("username and password are required error message 
is not displayed");
            extentReport.addScreenshot(driver);
            Logging.endTestCase();
        }
    }


    @Test(description = "TC-02:validate the login functionality with valid 
username and password")
    public void verifyLoginFunctionality() throws IOException {
        XSSFSheet sheet =  readWriteExcel.getSheet("Sheet2");
        for(int i=1;i<=sheet.getLastRowNum();i++)
        {
            extentReport.createTest("TC02-Validate the login functionality with 
valid username and password");
            loginPage.loginLink.click();
            Logging.info("user has clicked on login link");
            extentReport.info("user has clicked on login link");
            String username=sheet.getRow(i).getCell(0).getStringCellValue();
            String password = sheet.getRow(i).getCell(1).getStringCellValue();
            loginPage.username.sendKeys(username);
            loginPage.password.sendKeys(password);
            Logging.info("user has entered username and password");
            extentReport.info("user has entered username and password");
            loginPage.loginBtn.click();
            Logging.info("user has clicked on login button");
            extentReport.info("user has clicked on login button");
            if(driver.findElements(By.xpath("//
div[text()=\"Logout\"]")).size()>0)
            {
                Logging.info("Logout button is displayed");
                extentReport.info("Logout button is displayed");
                Logging.info("user logged in successfully");
                extentReport.pass("user logged in successfully");
                extentReport.addScreenshot(driver);
                loginPage.logoutBtn.click();
                Logging.endTestCase();
            }
            else
            {
                Logging.info("Logout button is not displayed");
                extentReport.info("Logout button is not displayed");
                Logging.info("user is not loggedin ");
                extentReport.fail("user is not loggedin");
                extentReport.addScreenshot(driver);
                Logging.endTestCase();
                throw new RuntimeException("Logout button is not displayed");
            }

        }
    }

 @Test(description = "TC -03:verify the search functionality in all the 
categories")
  public void searchFunctionality() throws IOException {
         XSSFSheet sheet = readWriteExcel.getSheet("Sheet1");
         for(int i=1;i<=sheet.getLastRowNum();i++)
         {
             extentReport.createTest("TC-03:verify the search functionality in 
all the categories ");
             String category = sheet.getRow(i).getCell(0).getStringCellValue();
             switch (category)
             {
                 case "Electronics":
                     ecommerceMenuPage.electronicLink.click();
                     Logging.info("User clicked on electronics category");
                     extentReport.info("user clicked on electronics category");

electronicsPage.searchBar.sendKeys(sheet.getRow(i).getCell(1).getStringCellValu
e());
                     Logging.info("user searched the electronics category");
                     extentReport.info("user searched the electronics 
category");
                     String actualproductname = 
electronicsPage.s21Nametext.getText();
                     String expectedProductname = 
sheet.getRow(i).getCell(1).getStringCellValue();
                     String actualproductPrice = 
electronicsPage.s21Price.getText();
                     double expectedproductprice = 
sheet.getRow(i).getCell(2).getNumericCellValue();

                     double actualproductPrice1;

actualproductPrice1=Double.parseDouble(actualproductPrice);
                     System.out.println("actual product name is :"
+actualproductname +"expected product name is:"+expectedProductname);
                     if(actualproductname.equals(expectedProductname.trim())) {
                         if (actualproductPrice1 == expectedproductprice) {
                             Logging.info("actual and expected prices and 
product names are equal");
                             extentReport.info("actual and expected prices and 
product names are equal");
                             Logging.info("search is working for electronics 
category");
                             extentReport.pass("search is working for 
electronics category");
                             extentReport.addScreenshot(driver);

                             Logging.endTestCase();
                         } else {
                             Logging.error("actual and expected product prices 
are not equal");
                             extentReport.info("actual and expected product 
prices are not equal");
                             extentReport.fail("actual and expected product 
prices are not equal");
                             extentReport.addScreenshot(driver);
                             Logging.endTestCase();
                             throw new RuntimeException();
                         }
                     }
                     else {
                         Logging.error("actual and expected product name are 
not equal");
                         extentReport.info("actual and expected product name 
are not equal");
                         Logging.error("search is not working for electronics 
category");
```

```
                        extentReport.fail("search is not working for 
electronics category");
                        extentReport.addScreenshot(driver);
                        Logging.endTestCase();
                        throw new RuntimeException();
                    }

                    driver.get(ConfigFileReader.getUrl());
                   break;

                case "Kitchen Items":
                ecommerceMenuPage.kitchenLink.click();
                Logging.info("user has clicked on kitchen item category");
                extentReport.info("user has clicked on kitchen item 
category");

kitchenItemsPage.searchBar.sendKeys(sheet.getRow(i).getCell(1).getStringCellVal
ue());
                Logging.info("user entered the item name in to kitchen item 
category search bar");
                extentReport.info("user entered the item name in to kitchen 
item category search bar");
                String actualproductName= 
kitchenItemsPage.prestigeStoveName.getText();
                String expectedproductName= 
sheet.getRow(i).getCell(1).getStringCellValue();
                String actualProductPrice = 
kitchenItemsPage.prestigeStovePrice.getText();
                double expectedProductPrice = 
sheet.getRow(i).getCell(2).getNumericCellValue();
                double actualProductPrice1 = 
Double.parseDouble(actualProductPrice);

                if(actualproductName.equals(expectedproductName.trim()))
                {
                    if(actualProductPrice1==expectedProductPrice)
                    {
                        Logging.info("actual and expected product name and 
prices are equal");
                        extentReport.info("actual and expected product name 
and prices are equal");
                        extentReport.pass("user is able to search the kitchen 
items category");
                        Logging.info("user is able to search the kitchen items 
category");
                        extentReport.addScreenshot(driver);
                        Logging.endTestCase();
                    }
                    else
                    {
                        Logging.error("actual and expected product prices are 
not equal");
                        extentReport.fail("actual and expected product prices 
are not equal");
                        extentReport.addScreenshot(driver);
                        Logging.endTestCase();
                        throw new RuntimeException();
                    }

                }

                else
                {
                    Logging.info("actual and expected product name are not 
same");
```

```
                        extentReport.info("actual and expected product name are 
not same");
                        extentReport.fail("user is not able to search the kitchen 
items category");
                        Logging.error("user is not able to search the kitchen 
items category");
                        extentReport.addScreenshot(driver);
                        Logging.endTestCase();
                        throw new RuntimeException();
                    }
                    driver.get(ConfigFileReader.getUrl());
                    break;

                    case "Sports":
                       ecommerceMenuPage.sportsLink.click();
                       Logging.info("user clicked on sports category ");
                       extentReport.info("user clicked on sports category ");

sportsPage.searchBar.sendKeys(sheet.getRow(i).getCell(1).getStringCellValue());
                       Logging.info("User has entered the sport item in to search 
bar");
                       extentReport.info("User has entered the sport item in to 
search bar");
                       String actualProductName=sportsPage.sgBatName.getText();
                       String expectedProductName = 
sheet.getRow(i).getCell(1).getStringCellValue();
                       String actualspProductPrice 
=sportsPage.sgBatPrice.getText();
                       double expectedspProductPrice = 
sheet.getRow(i).getCell(2).getNumericCellValue();
                       double actualspProductPrice1 = 
Double.parseDouble(actualspProductPrice);
                       if(actualProductName.equals(expectedProductName.trim()))
                       {
                           if(actualspProductPrice1==expectedspProductPrice)
                           {
                               Logging.info("actual and expected product price and 
names are equal");
                               extentReport.info("actual and expected product 
price and names are equal");
                               extentReport.pass("user is able to search sports 
category");
                               Logging.info("user is able to search sports 
category");
                               extentReport.addScreenshot(driver);
                               Logging.endTestCase();
                           }
                           else
                           {
                               Logging.error("actual and expected product prices 
are not equal");
                               extentReport.fail("actual and expected product 
prices are not equal");
                               extentReport.addScreenshot(driver);
                               Logging.endTestCase();
                               throw new RuntimeException();
                           }
                       }

                       else
                       {
                           Logging.error("actual and expected product names are 
not equal");
                           extentReport.info("actual and expected product names 
are not equal");
```

```
                            extentReport.fail("search is not working for sports 
category");
                            extentReport.addScreenshot(driver);
                            Logging.endTestCase();
                            throw new RuntimeException();
                        }

                        break;

                    default:
                        Logging.info("no matching category is found");
                        extentReport.fail("no matching category is found");
                        extentReport.addScreenshot(driver);
                        Logging.endTestCase();

                }


            }
 }

 @Test(description = "TC-04:add item into cart and verify the details on cart 
page")
    public void addItemToCart() throws IOException {
     extentReport.createTest("TC-04:add item into cart and verify the details 
on cart page");
     Utility.login(driver);
     Logging.info("user is loggedin successfully");
     extentReport.info("user is loggedin successfully");
     ecommerceMenuPage.electronicLink.click();
     Logging.info("user has clicked on electronics link");
     extentReport.info("user has clicked on electronics link");
     electronicsPage.s21Nametext.click();
     Logging.info("user has clicked on the product");
     extentReport.info("user has clicked on the product");
     String productnameonDetailPage= 
electronicsProductDetailPage.s21name.getText();
     String 
productpriceonDetailPage=electronicsProductDetailPage.s21Price.getText();
     double 
productpriceonDetailPage1=Double.parseDouble(productpriceonDetailPage);
     electronicsProductDetailPage.addToCartBtn.click();
     extentReport.info("user clicked on add to cart button");
     Logging.info("user clicked on add to cart button");

driver.manage().timeouts().implicitlyWait(Duration.ofSeconds(Long.parseLong(Con
figFileReader.getImplicitWait())));

driver.manage().timeouts().pageLoadTimeout(Duration.ofSeconds(Long.parseLong(Co
nfigFileReader.getPageLoadTimeOut())));
     electronicsProductDetailPage.gotoCartBtn.click();
     Logging.info("user clicked on gotocart button");
     extentReport.info("user clicked on gotocart button");
     String productnameoncartpage = electronicsCartPage.s21name.getText();
     String productpriceoncartpage=electronicsCartPage.s21price.getText();
     double productpriceoncartpage1= 
Double.parseDouble(productpriceoncartpage);
     if(productnameonDetailPage.equals(productnameoncartpage))
     {
         if(productpriceoncartpage1==productpriceonDetailPage1)
         {
             Logging.info("product name and price is same on product detail 
page and product cart page");
             extentReport.info("product name and price is same on product 
detail page and product cart page");
```

```
                extentReport.pass("add to cart is working and details are correct 
on cart page");
                extentReport.addScreenshot(driver);
                Logging.endTestCase();
           }

           else
           {
                Logging.error("product price is not correct on cart  page");
                extentReport.fail("product price is not correct on cart  page");
                extentReport.addScreenshot(driver);
                Logging.endTestCase();
                throw new RuntimeException();

           }
      }

      else
      {
           Logging.error("product name is not correct on cart page");
           extentReport.fail("product name is not correct on cart page");
           extentReport.addScreenshot(driver);
           Logging.endTestCase();
           throw new RuntimeException();
      }

 }
}
```