EcommerceMenuPage:

```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 EcommerceMenuPage {

    public static WebDriver driver;

    public EcommerceMenuPage(WebDriver driver)
    {
        this.driver=driver;

                    PageFactory.initElements(driver,this);
    }

    @FindBy(xpath="//a[text()=\"All Products\"]")
    public WebElement allProductsLink;

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

    @FindBy(xpath="//a[text()=\"Kitchen Items\"]")
    public WebElement kitchenLink;



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

}
```

ElectronicsPage:

```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 ElectronicsPage {

    public static WebDriver driver;

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

    @FindBy(xpath="//input[@placeholder=\"Search by name...\"]")
    public WebElement searchBar;

    @FindBy(xpath="//div[text()=\"s21\"]")
    public WebElement s21Nametext;

    @FindBy(xpath="//div[text()=\"65000.00\"]")
    public WebElement s21Price;
}
```

KitchenItemsPage:

```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 KitchenItemsPage {

    public static WebDriver driver;

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

    @FindBy(xpath="//input[@placeholder=\"Search by name...\"]")
    public WebElement searchBar;

    @FindBy(xpath="//div[text()=\"Prestige Stove\"]")
    public WebElement prestigeStoveName;

    @FindBy(xpath="//div[text()=\"14500.00\"]")
    public WebElement prestigeStovePrice;
}
```

SportsPage:

```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 SportsPage {
    public static WebDriver driver;

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

    @FindBy(xpath="//input[@placeholder=\"Search by name...\"]")
    public WebElement searchBar;

    @FindBy(xpath="//div[text()=\"SG Bat\"]")
    public WebElement sgBatName;

    @FindBy(xpath="//div[text()=\"25500.00\"]")
    public WebElement sgBatPrice;
}
```

HerokuappTestCases:

```
 package AutomationScripts ;
import Context.TestContext;
import ObjectManager.DriverManager;
import PageObjects.*;
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 java.io.IOException;

@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;

    @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);
        PropertyConfigurator.configure("src/main/resources/log4j.properties");
    }

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

```java
    }

    @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 = "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 = "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);
                    //
expectedproductprice1=Double.parseDouble(expectedproductprice);
                    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();
                    }
                   break;
             }
         }
 }
}
```