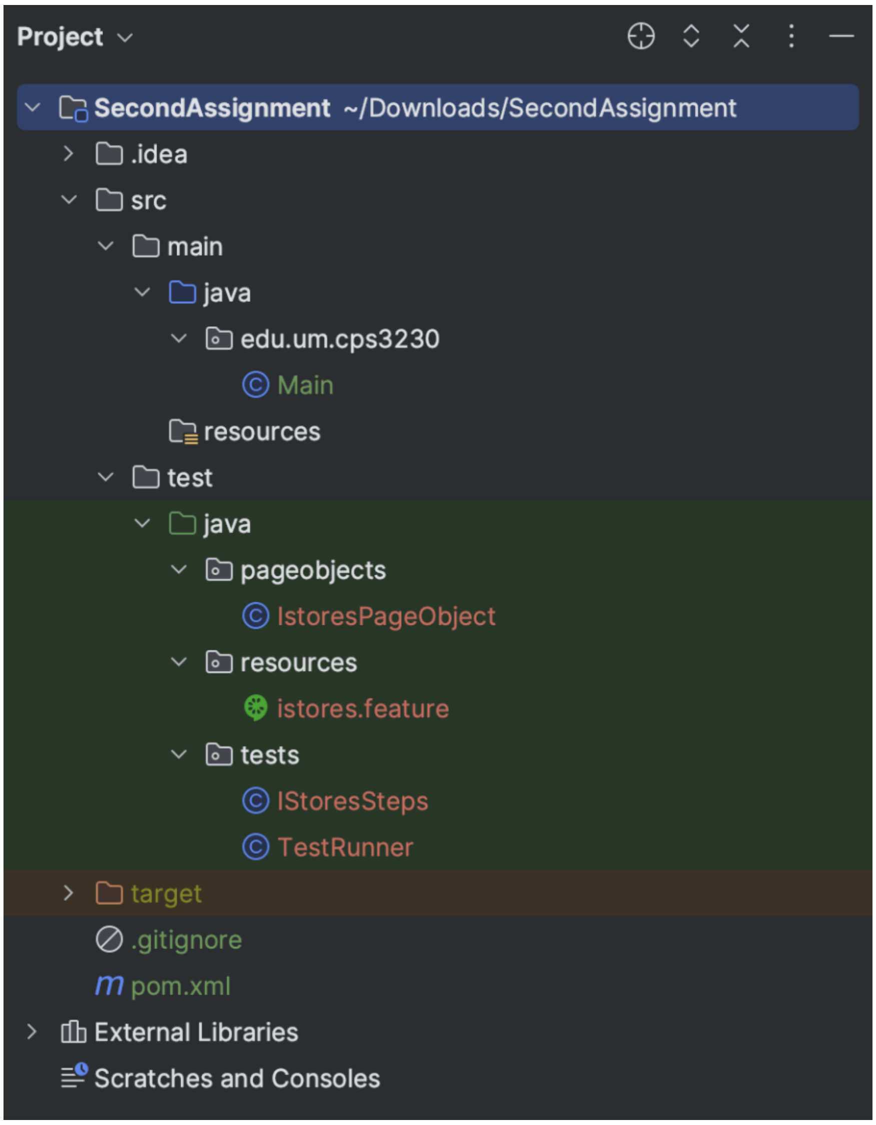
Assignment Part 2 (of 3) - Unit Testing

First Part

My project structure



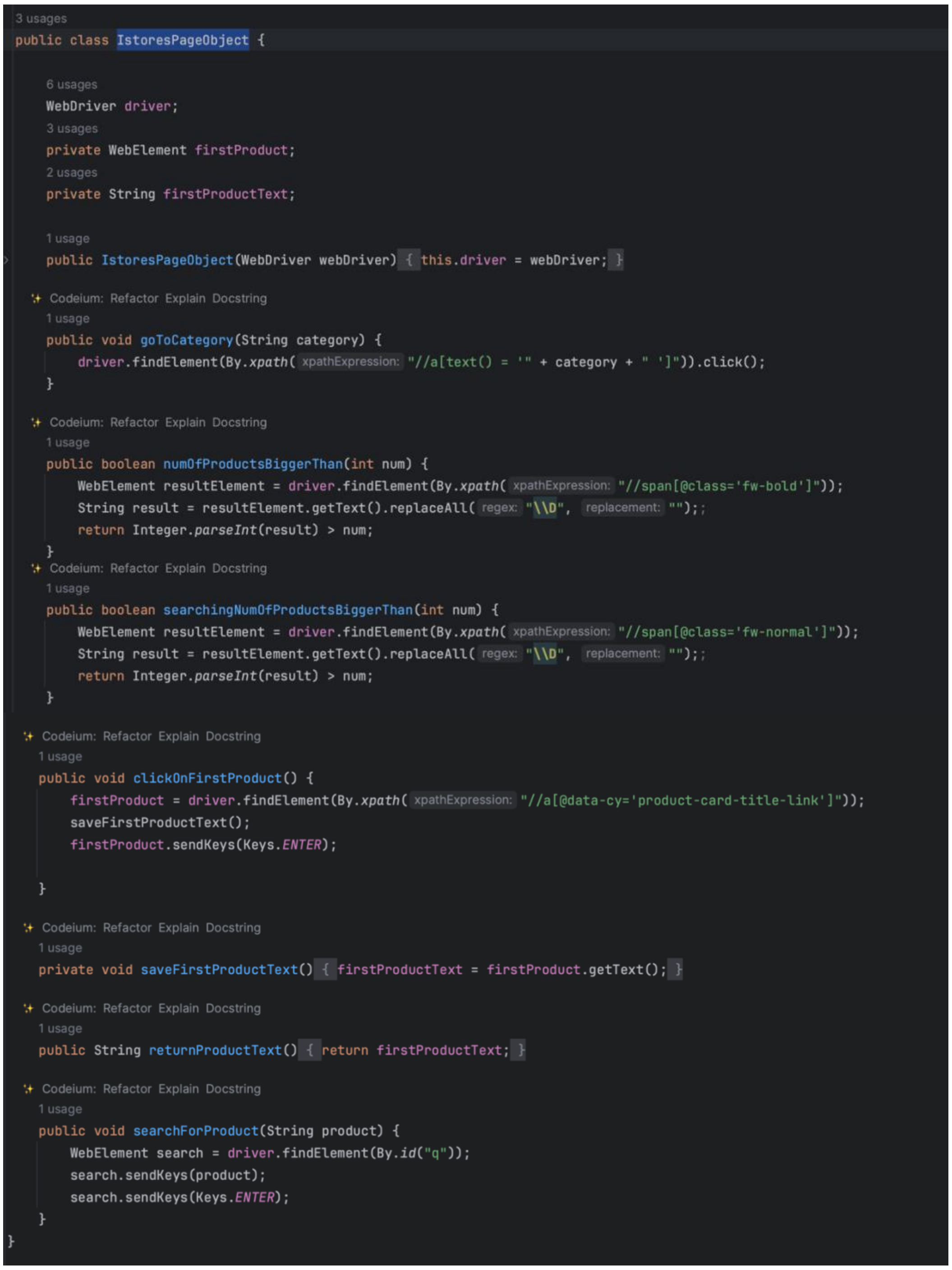
```
Project
SecondAssignment ~/Downloads/SecondAssignment
  .idea
  src
    main
      java
        edu.um.cps3230
          Main
      resources
    test
      java
        pageobjects
          IstoresPageObject
        resources
          istores.feature
        tests
          IStoresSteps
          TestRunner
  target
  .gitignore
  pom.xml
External Libraries
Scratches and Consoles
```

IstoresPageObject - A helper class that interacts directly with the web page.
Istores.feature – Cucumber file.
IStoresSteps - Class that defines the steps used in the cucumber file.
TestRunner – class for running the test.
Main - Pre-created, empty

GitHub link:
https://github.com/JanSkacel01/WebTestAutomationandBDD

Video link:
https://drive.google.com/drive/folders/1JSjtkU9v-SgxDei2irDalLOMjMAFEQZI?usp=share_link

IstoresPageObject



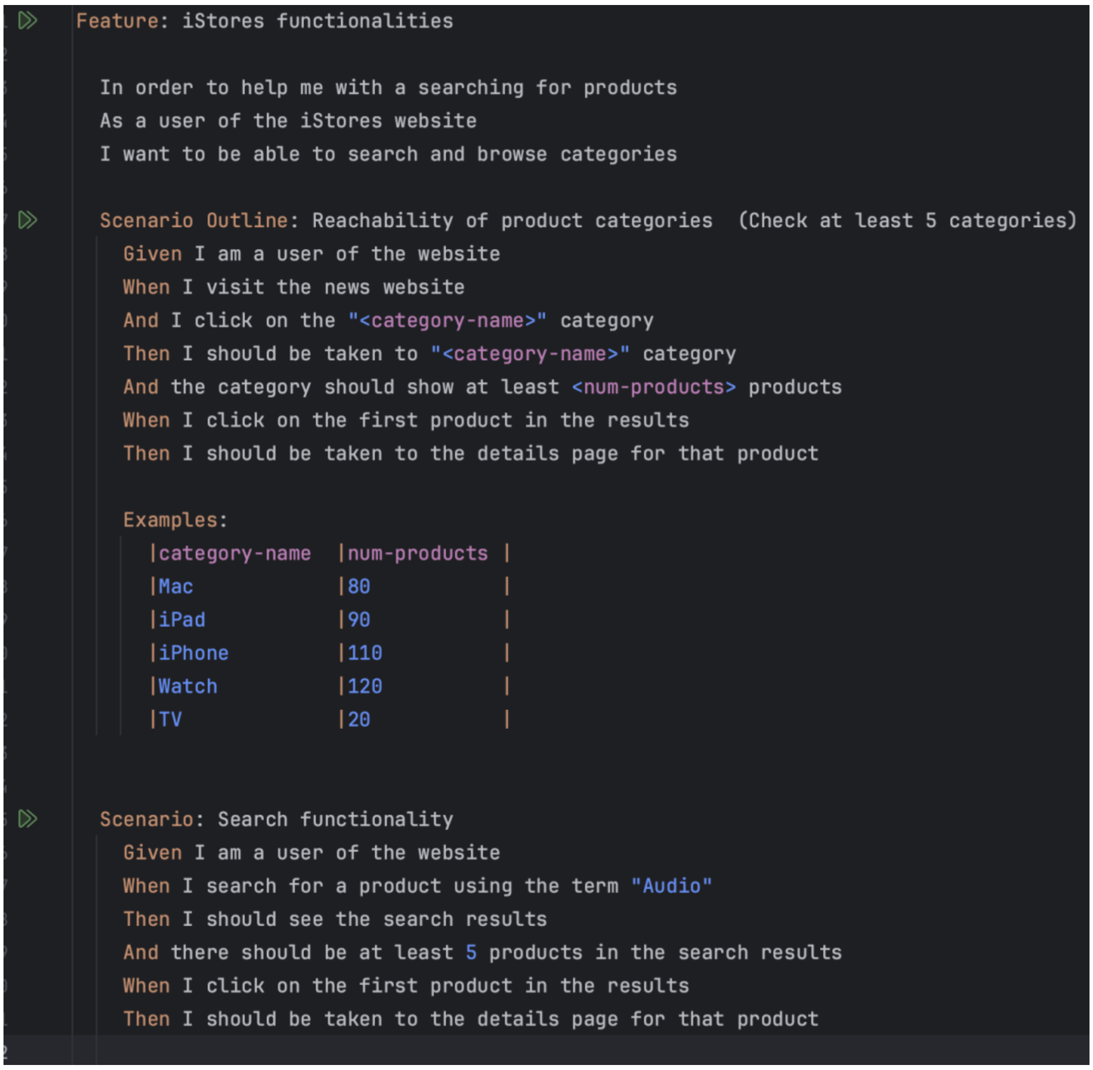
```
3 usages
public class IstoresPageObject {

    6 usages
    WebDriver driver;
    3 usages
    private WebElement firstProduct;
    2 usages
    private String firstProductText;

    1 usage
    public IstoresPageObject(WebDriver webDriver) { this.driver = webDriver; }

  Codeium: Refactor Explain Docstring
    1 usage
    public void goToCategory(String category) {
        driver.findElement(By.xpath( xpathExpression: "//a[text() = '" + category + " ']")).click();
    }

  Codeium: Refactor Explain Docstring
    1 usage
    public boolean numOfProductsBiggerThan(int num) {
        WebElement resultElement = driver.findElement(By.xpath( xpathExpression: "//span[@class='fw-bold']"));
        String result = resultElement.getText().replaceAll( regex: "\\D",  replacement: "");;
        return Integer.parseInt(result) > num;
    }
  Codeium: Refactor Explain Docstring
    1 usage
    public boolean searchingNumOfProductsBiggerThan(int num) {
        WebElement resultElement = driver.findElement(By.xpath( xpathExpression: "//span[@class='fw-normal']"));
        String result = resultElement.getText().replaceAll( regex: "\\D",  replacement: "");;
        return Integer.parseInt(result) > num;
    }

  Codeium: Refactor Explain Docstring
    1 usage
    public void clickOnFirstProduct() {
        firstProduct = driver.findElement(By.xpath( xpathExpression: "//a[@data-cy='product-card-title-link']"));
        saveFirstProductText();
        firstProduct.sendKeys(Keys.ENTER);

    }

  Codeium: Refactor Explain Docstring
    1 usage
    private void saveFirstProductText() { firstProductText = firstProduct.getText(); }

  Codeium: Refactor Explain Docstring
    1 usage
    public String returnProductText() { return firstProductText; }

  Codeium: Refactor Explain Docstring
    1 usage
    public void searchForProduct(String product) {
        WebElement search = driver.findElement(By.id("q"));
        search.sendKeys(product);
        search.sendKeys(Keys.ENTER);
    }
}
```

In this class I have created methods that directly communicate with the web page using the Webdriver embedded in the constructor. These methods are created to shorten the code, improve readability and create more abstraction.

It's worth mentioning that for most elements, I don't use the .click methods but the .sendKeys method. This is for a simple reason and that is that the elements were not clickable.

It is also worth mentioning the saveFirstProductText() and returnProductText() methods. These methods allow us to see if we were actually referred to the product we clicked on.

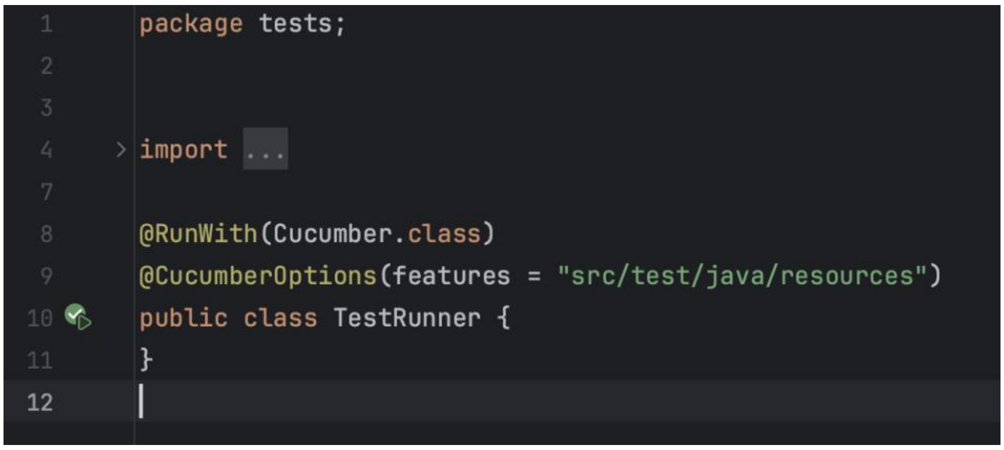
Istores.feature



```
Feature: iStores functionalities

  In order to help me with a searching for products
  As a user of the iStores website
  I want to be able to search and browse categories

  Scenario Outline: Reachability of product categories  (Check at least 5 categories)
    Given I am a user of the website
    When I visit the news website
    And I click on the "<category-name>" category
    Then I should be taken to "<category-name>" category
    And the category should show at least <num-products> products
    When I click on the first product in the results
    Then I should be taken to the details page for that product

    Examples:
      |category-name  |num-products |
      |Mac            |80           |
      |iPad           |90           |
      |iPhone         |110          |
      |Watch          |120          |
      |TV             |20           |


  Scenario: Search functionality
    Given I am a user of the website
    When I search for a product using the term "Audio"
    Then I should see the search results
    And there should be at least 5 products in the search results
    When I click on the first product in the results
    Then I should be taken to the details page for that product
```

Cucumber file that was specified in the assignment

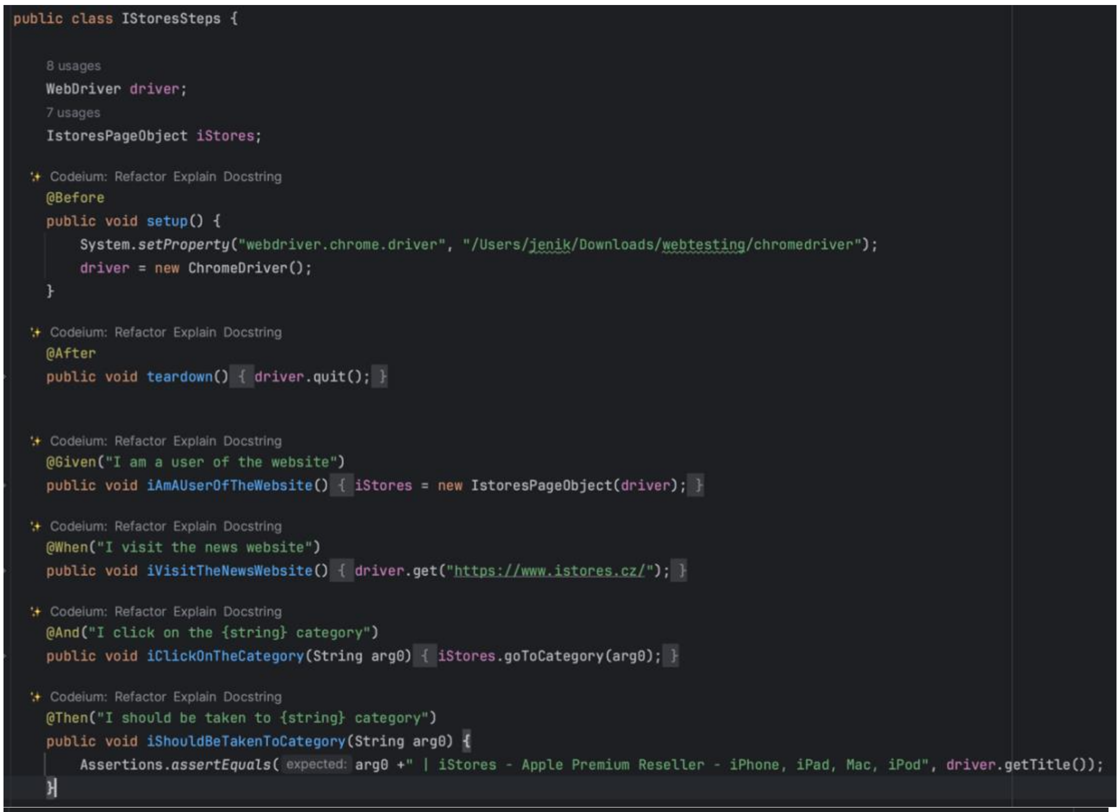
TestRunner



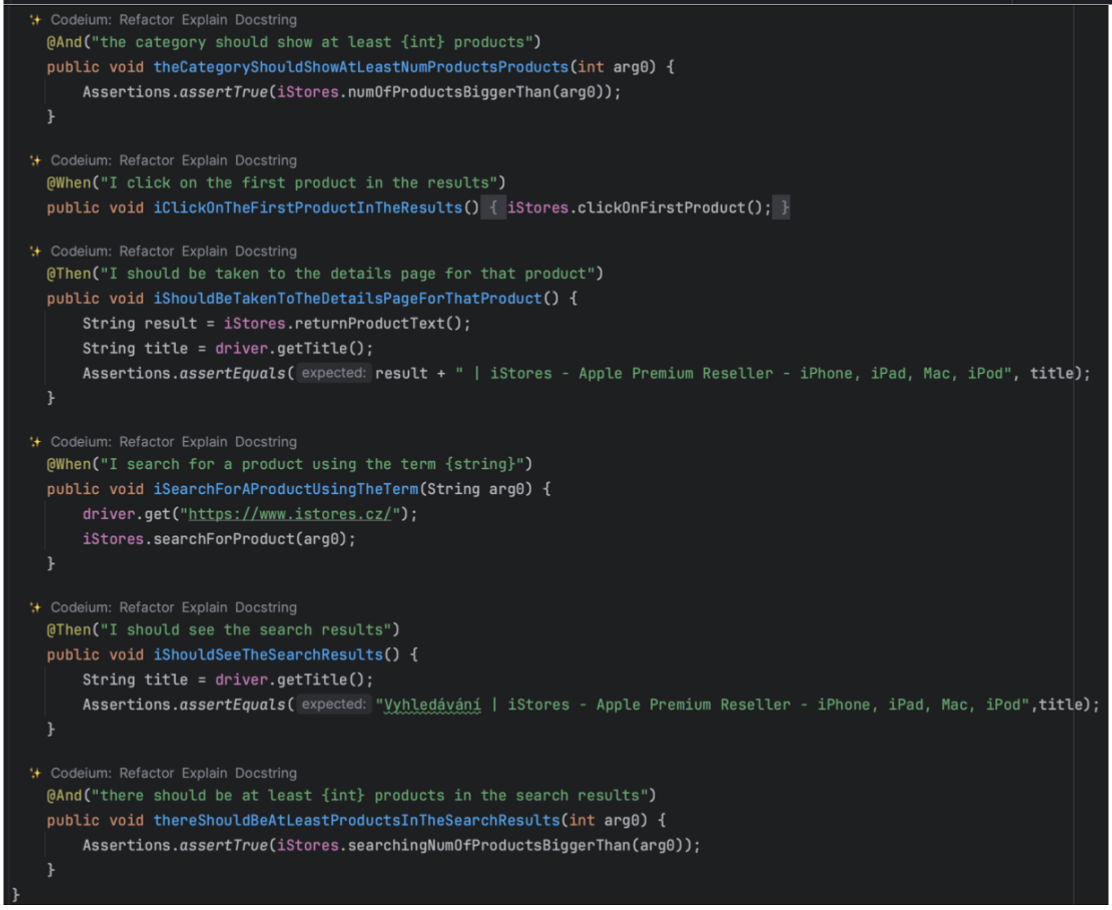
```
package tests;


import ...

@RunWith(Cucumber.class)
@CucumberOptions(features = "src/test/java/resources")
public class TestRunner {
}
```

Class for running the test.

IStoresSteps



```
public class IStoresSteps {

    8 usages
    WebDriver driver;
    7 usages
    IstoresPageObject iStores;

  Codeium: Refactor Explain Docstring
    @Before
    public void setup() {
        System.setProperty("webdriver.chrome.driver", "/Users/jenik/Downloads/webtesting/chromedriver");
        driver = new ChromeDriver();
    }

  Codeium: Refactor Explain Docstring
    @After
    public void teardown() { driver.quit(); }


  Codeium: Refactor Explain Docstring
    @Given("I am a user of the website")
    public void iAmAUserOfTheWebsite() { iStores = new IstoresPageObject(driver); }

  Codeium: Refactor Explain Docstring
    @When("I visit the news website")
    public void iVisitTheNewsWebsite() { driver.get("https://www.istores.cz/"); }

  Codeium: Refactor Explain Docstring
    @And("I click on the {string} category")
    public void iClickOnTheCategory(String arg0) { iStores.goToCategory(arg0); }

  Codeium: Refactor Explain Docstring
    @Then("I should be taken to {string} category")
    public void iShouldBeTakenToCategory(String arg0) {
        Assertions.assertEquals( expected: arg0 +" | iStores - Apple Premium Reseller - iPhone, iPad, Mac, iPod", driver.getTitle());
    }
```

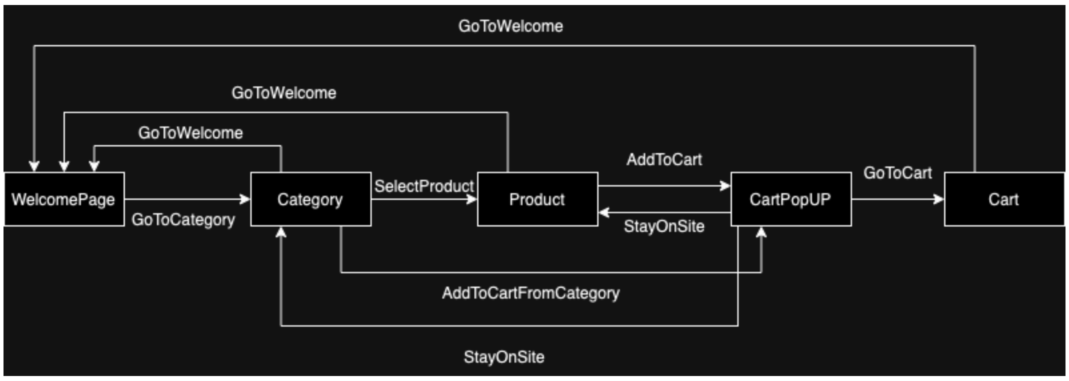


```
  Codeium: Refactor Explain Docstring
    @And("the category should show at least {int} products")
    public void theCategoryShouldShowAtLeastNumProductsProducts(int arg0) {
        Assertions.assertTrue(iStores.numOfProductsBiggerThan(arg0));
    }

  Codeium: Refactor Explain Docstring
    @When("I click on the first product in the results")
    public void iClickOnTheFirstProductInTheResults() { iStores.clickOnFirstProduct(); }

  Codeium: Refactor Explain Docstring
    @Then("I should be taken to the details page for that product")
    public void iShouldBeTakenToTheDetailsPageForThatProduct() {
        String result = iStores.returnProductText();
        String title = driver.getTitle();
        Assertions.assertEquals( expected: result + " | iStores - Apple Premium Reseller - iPhone, iPad, Mac, iPod", title);
    }

  Codeium: Refactor Explain Docstring
    @When("I search for a product using the term {string}")
    public void iSearchForAProductUsingTheTerm(String arg0) {
        driver.get("https://www.istores.cz/");
        iStores.searchForProduct(arg0);
    }

  Codeium: Refactor Explain Docstring
    @Then("I should see the search results")
    public void iShouldSeeTheSearchResults() {
        String title = driver.getTitle();
        Assertions.assertEquals( expected: "Vyhledávání | iStores - Apple Premium Reseller - iPhone, iPad, Mac, iPod",title);
    }

  Codeium: Refactor Explain Docstring
    @And("there should be at least {int} products in the search results")
    public void thereShouldBeAtLeastProductsInTheSearchResults(int arg0) {
        Assertions.assertTrue(iStores.searchingNumOfProductsBiggerThan(arg0));
    }
}
```

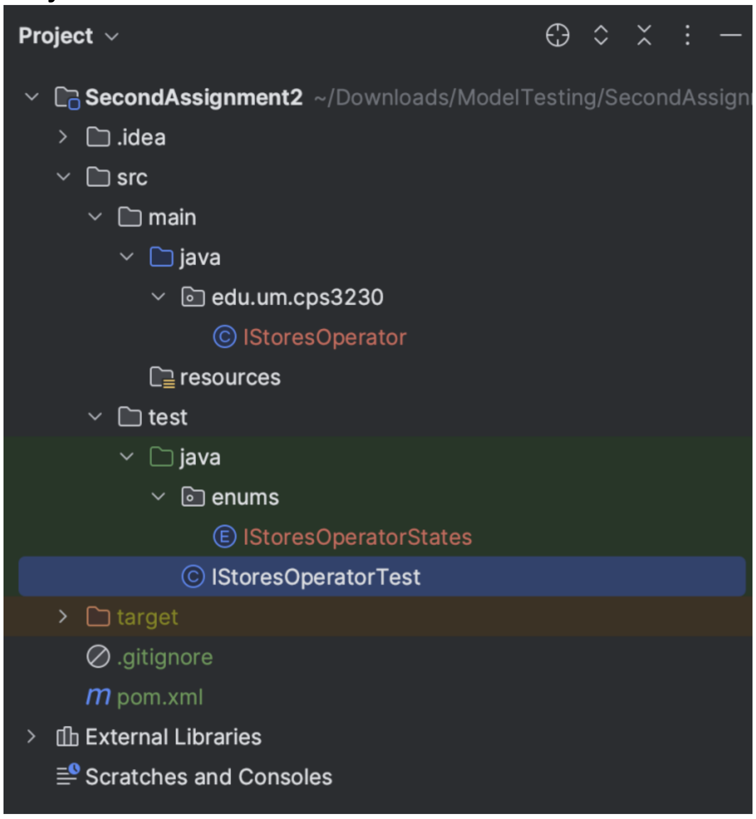
- Class that defines steps for cucumber file.

Second part

Model



Project structure



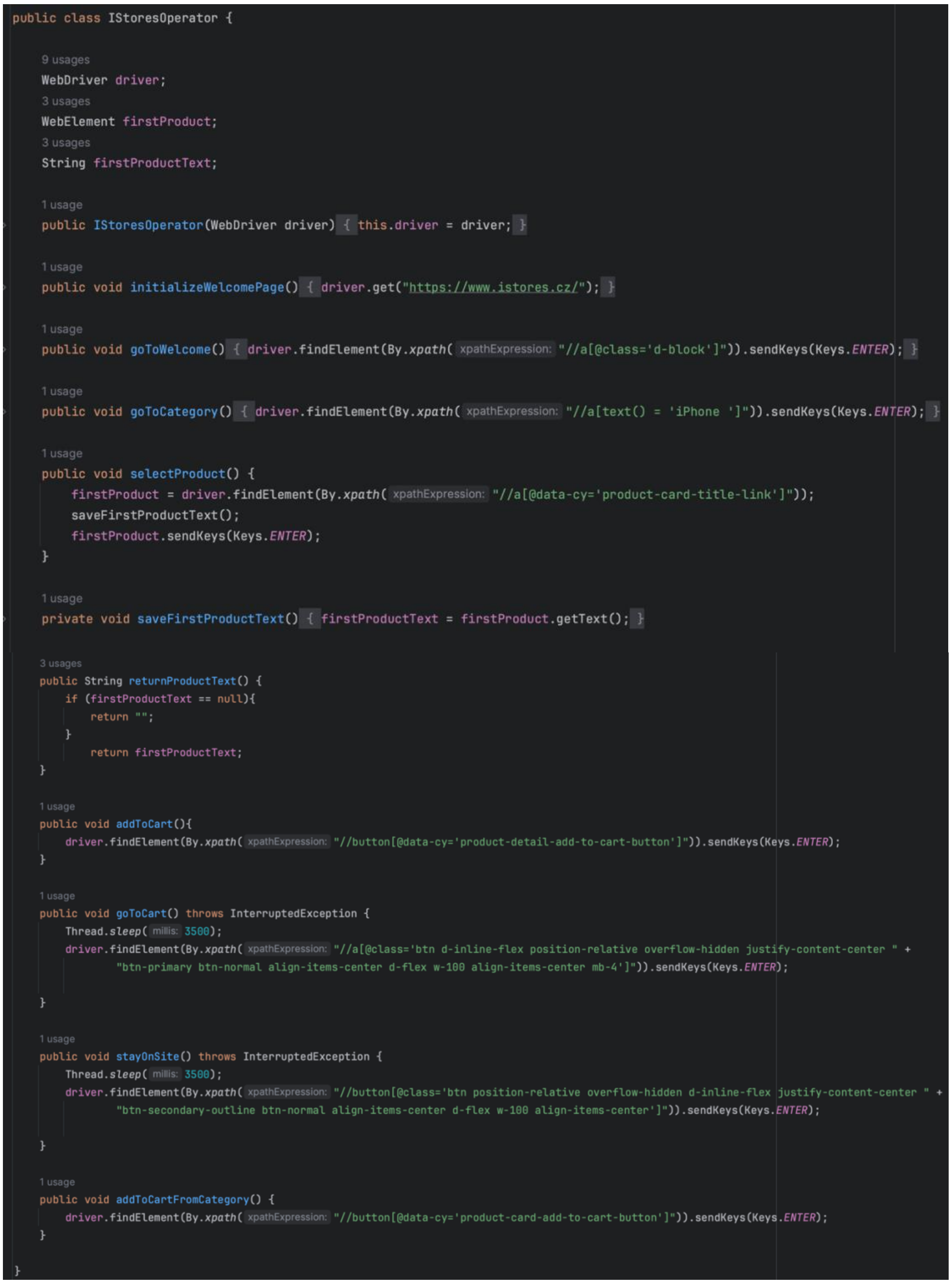
```
Project
SecondAssignment2 ~/Downloads/ModelTesting/SecondAssign
  .idea
  src
    main
      java
        edu.um.cps3230
          IStoresOperator
      resources
    test
      java
        enums
          IStoresOperatorStates
        IStoresOperatorTest
  target
  .gitignore
  pom.xml
External Libraries
Scratches and Consoles
```

IStoresOperator - Class that interacts directly with the web page.
IStoresOperatorStates – Enum for indicating website, that we are currently on.
IStoresOperatorTest – Test class

GitHub link:
https://github.com/JanSkacel01/ModelBasedTesting

Video link:
https://drive.google.com/drive/folders/1JSjtkU9v-SgxDei2irDalLOMjMAFEQZI?usp=share_link

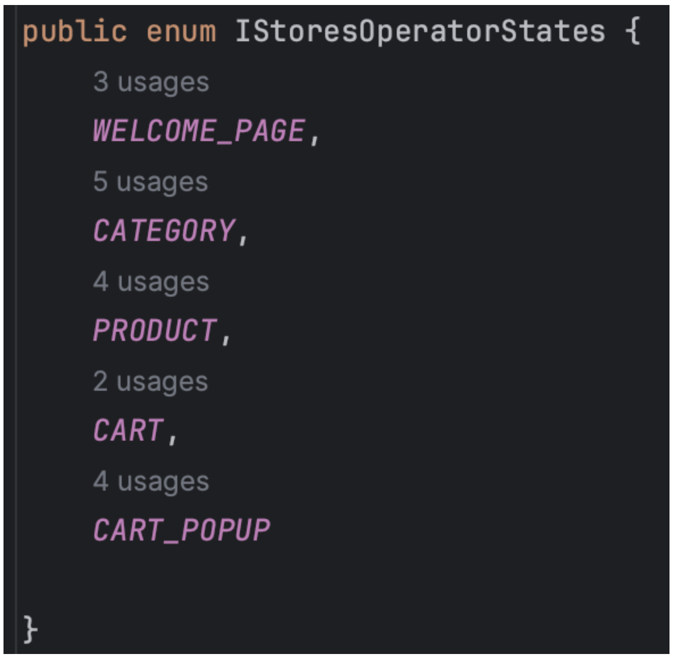
IStoresOperator



```java
public class IStoresOperator {

    9 usages
    WebDriver driver;
    3 usages
    WebElement firstProduct;
    3 usages
    String firstProductText;

    1 usage
    public IStoresOperator(WebDriver driver) { this.driver = driver; }

    1 usage
    public void initializeWelcomePage() { driver.get("https://www.istores.cz/"); }

    1 usage
    public void goToWelcome() { driver.findElement(By.xpath( xpathExpression: "//a[@class='d-block']")).sendKeys(Keys.ENTER); }

    1 usage
    public void goToCategory() { driver.findElement(By.xpath( xpathExpression: "//a[text() = 'iPhone ']")).sendKeys(Keys.ENTER); }

    1 usage
    public void selectProduct() {
        firstProduct = driver.findElement(By.xpath( xpathExpression: "//a[@data-cy='product-card-title-link']"));
        saveFirstProductText();
        firstProduct.sendKeys(Keys.ENTER);
    }

    1 usage
    private void saveFirstProductText() { firstProductText = firstProduct.getText(); }

    3 usages
    public String returnProductText() {
        if (firstProductText == null){
            return "";
        }
            return firstProductText;
    }

    1 usage
    public void addToCart(){
        driver.findElement(By.xpath( xpathExpression: "//button[@data-cy='product-detail-add-to-cart-button']")).sendKeys(Keys.ENTER);
    }

    1 usage
    public void goToCart() throws InterruptedException {
        Thread.sleep( millis: 3500);
        driver.findElement(By.xpath( xpathExpression: "//a[@class='btn d-inline-flex position-relative overflow-hidden justify-content-center " +
                "btn-primary btn-normal align-items-center d-flex w-100 align-items-center mb-4']")).sendKeys(Keys.ENTER);

    }

    1 usage
    public void stayOnSite() throws InterruptedException {
        Thread.sleep( millis: 3500);
        driver.findElement(By.xpath( xpathExpression: "//button[@class='btn position-relative overflow-hidden d-inline-flex justify-content-center " +
                "btn-secondary-outline btn-normal align-items-center d-flex w-100 align-items-center']")).sendKeys(Keys.ENTER);

    }

    1 usage
    public void addToCartFromCategory() {
        driver.findElement(By.xpath( xpathExpression: "//button[@data-cy='product-card-add-to-cart-button']")).sendKeys(Keys.ENTER);
    }

}
```

- Again we don’t use .click methods but the .sendKeys methods. For same reason.
- For the goToCart() and stayOnSite() methods we use Thread.sleep(3500). This is because when a product is added to the cart, the popup takes approximately 2 seconds to pop up.
- saveFirstProductText() and returnProductText() will help us find out if are on the PRODUCT page.

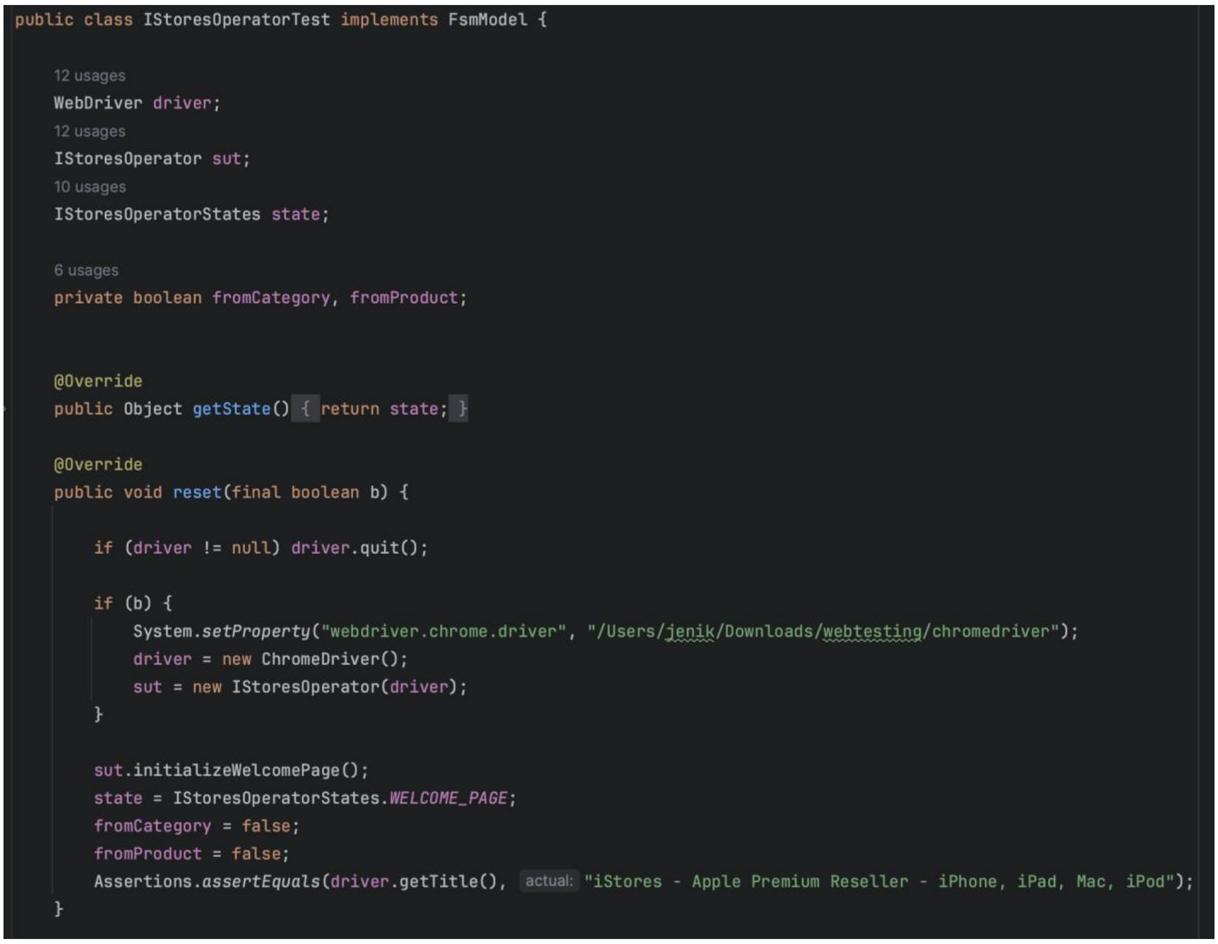
IStoresOperatorStates



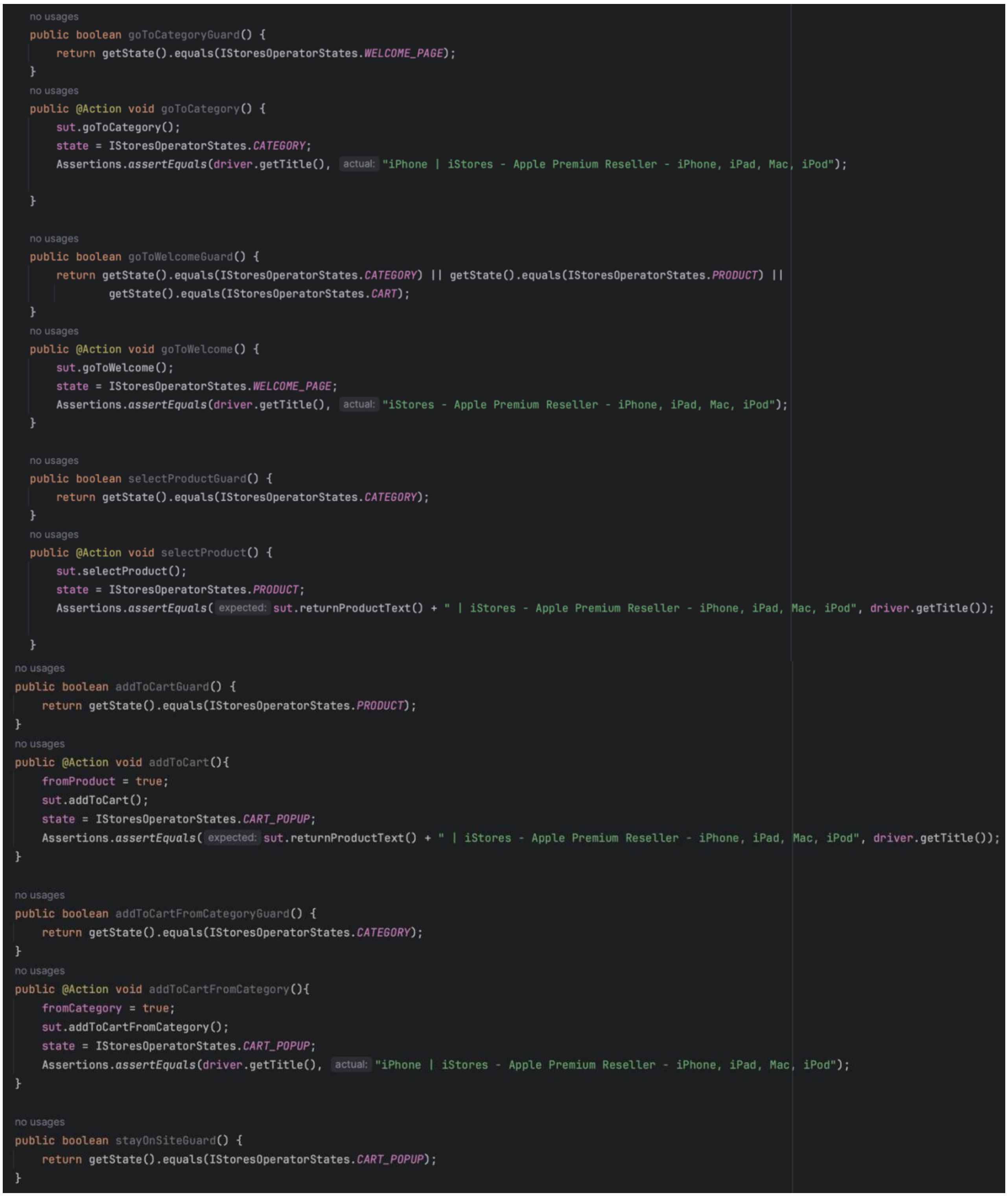
```
public enum IStoresOperatorStates {
    3 usages
    WELCOME_PAGE,
    5 usages
    CATEGORY,
    4 usages
    PRODUCT,
    2 usages
    CART,
    4 usages
    CART_POPUP

}
```

- Enum for indicating website, that we are currently on.

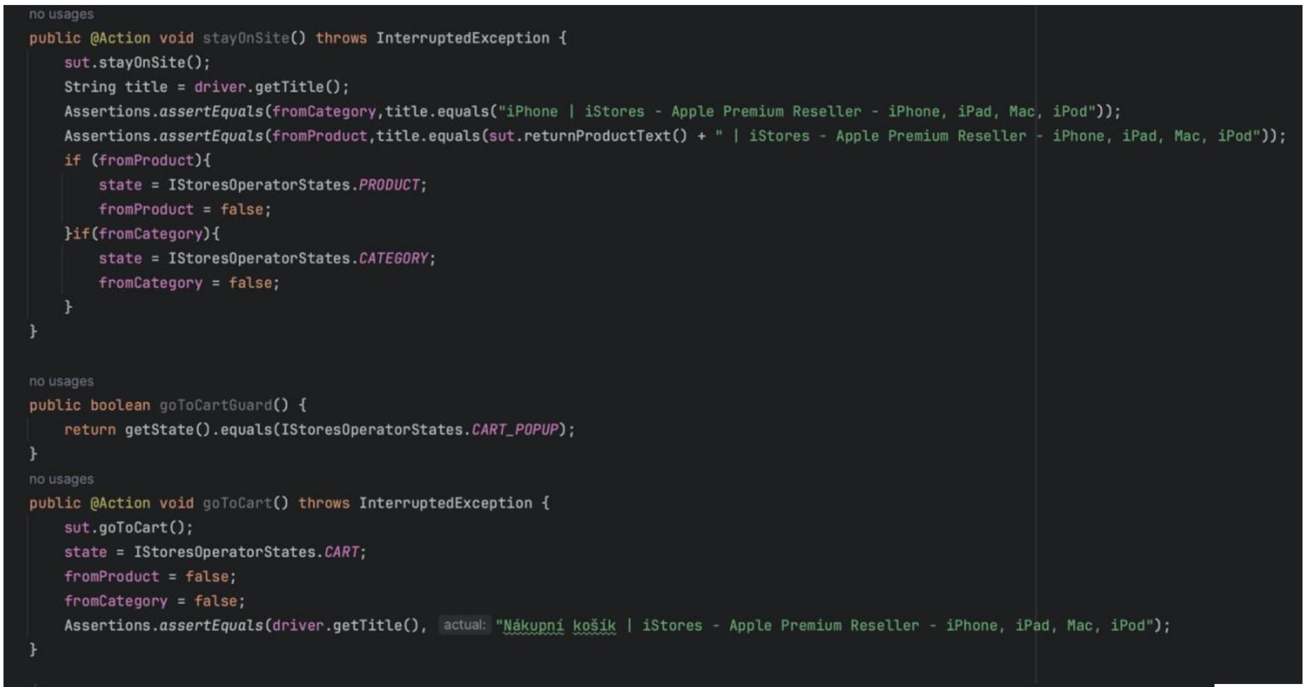
IStoresOperatorTest



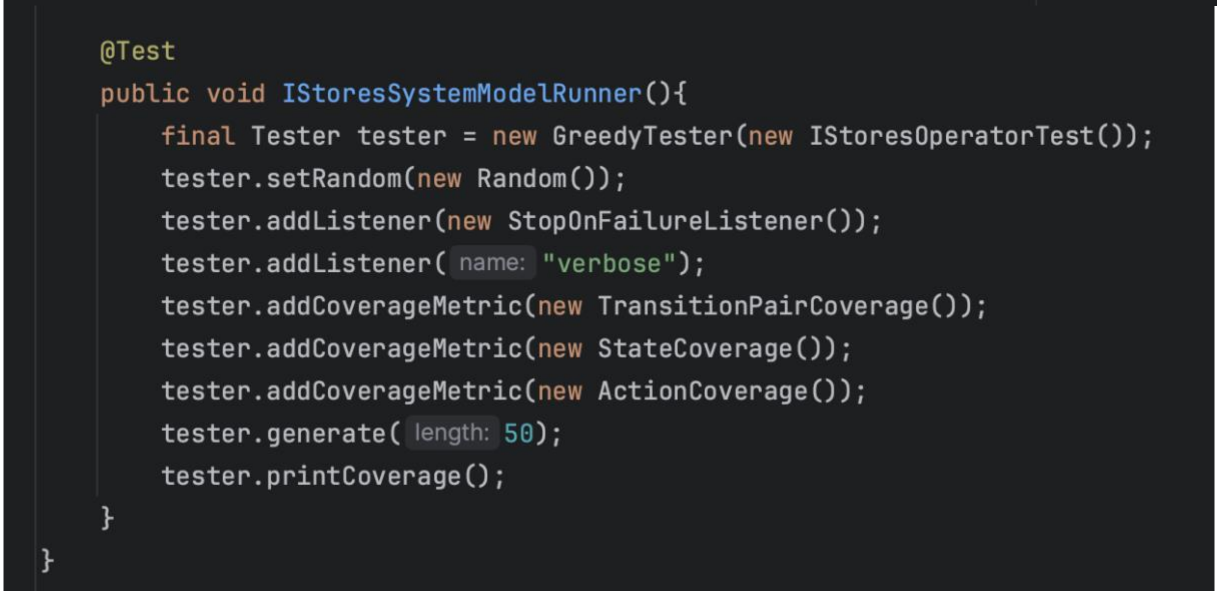
```
public class IStoresOperatorTest implements FsmModel {

    12 usages
    WebDriver driver;
    12 usages
    IStoresOperator sut;
    10 usages
    IStoresOperatorStates state;

    6 usages
    private boolean fromCategory, fromProduct;


    @Override
    public Object getState() { return state; }

    @Override
    public void reset(final boolean b) {

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

        if (b) {
            System.setProperty("webdriver.chrome.driver", "/Users/jenik/Downloads/webtesting/chromedriver");
            driver = new ChromeDriver();
            sut = new IStoresOperator(driver);
        }

        sut.initializeWelcomePage();
        state = IStoresOperatorStates.WELCOME_PAGE;
        fromCategory = false;
        fromProduct = false;
        Assertions.assertEquals(driver.getTitle(), actual: "iStores - Apple Premium Reseller - iPhone, iPad, Mac, iPod");
    }
```



```
    no usages
    public boolean goToCategoryGuard() {
        return getState().equals(IStoresOperatorStates.WELCOME_PAGE);
    }
    no usages
    public @Action void goToCategory() {
        sut.goToCategory();
        state = IStoresOperatorStates.CATEGORY;
        Assertions.assertEquals(driver.getTitle(), actual: "iPhone | iStores - Apple Premium Reseller - iPhone, iPad, Mac, iPod");

    }

    no usages
    public boolean goToWelcomeGuard() {
        return getState().equals(IStoresOperatorStates.CATEGORY) || getState().equals(IStoresOperatorStates.PRODUCT) ||
                getState().equals(IStoresOperatorStates.CART);
    }
    no usages
    public @Action void goToWelcome() {
        sut.goToWelcome();
        state = IStoresOperatorStates.WELCOME_PAGE;
        Assertions.assertEquals(driver.getTitle(), actual: "iStores - Apple Premium Reseller - iPhone, iPad, Mac, iPod");
    }

    no usages
    public boolean selectProductGuard() {
        return getState().equals(IStoresOperatorStates.CATEGORY);
    }
    no usages
    public @Action void selectProduct() {
        sut.selectProduct();
        state = IStoresOperatorStates.PRODUCT;
        Assertions.assertEquals( expected: sut.returnProductText() + " | iStores - Apple Premium Reseller - iPhone, iPad, Mac, iPod", driver.getTitle());

    }
no usages
public boolean addToCartGuard() {
    return getState().equals(IStoresOperatorStates.PRODUCT);
}
no usages
public @Action void addToCart(){
    fromProduct = true;
    sut.addToCart();
    state = IStoresOperatorStates.CART_POPUP;
    Assertions.assertEquals( expected: sut.returnProductText() + " | iStores - Apple Premium Reseller - iPhone, iPad, Mac, iPod", driver.getTitle());
}

no usages
public boolean addToCartFromCategoryGuard() {
    return getState().equals(IStoresOperatorStates.CATEGORY);
}
no usages
public @Action void addToCartFromCategory(){
    fromCategory = true;
    sut.addToCartFromCategory();
    state = IStoresOperatorStates.CART_POPUP;
    Assertions.assertEquals(driver.getTitle(), actual: "iPhone | iStores - Apple Premium Reseller - iPhone, iPad, Mac, iPod");
}

no usages
public boolean stayOnSiteGuard() {
    return getState().equals(IStoresOperatorStates.CART_POPUP);
}
```



```java
no usages
public @Action void stayOnSite() throws InterruptedException {
    sut.stayOnSite();
    String title = driver.getTitle();
    Assertions.assertEquals(fromCategory,title.equals("iPhone | iStores - Apple Premium Reseller - iPhone, iPad, Mac, iPod"));
    Assertions.assertEquals(fromProduct,title.equals(sut.returnProductText() + " | iStores - Apple Premium Reseller - iPhone, iPad, Mac, iPod"));
    if (fromProduct){
        state = IStoresOperatorStates.PRODUCT;
        fromProduct = false;
    }if(fromCategory){
        state = IStoresOperatorStates.CATEGORY;
        fromCategory = false;
    }
}

no usages
public boolean goToCartGuard() {
    return getState().equals(IStoresOperatorStates.CART_POPUP);
}
no usages
public @Action void goToCart() throws InterruptedException {
    sut.goToCart();
    state = IStoresOperatorStates.CART;
    fromProduct = false;
    fromCategory = false;
    Assertions.assertEquals(driver.getTitle(),  actual: "Nákupní košík | iStores - Apple Premium Reseller - iPhone, iPad, Mac, iPod");
}
```



```java
    @Test
    public void IStoresSystemModelRunner(){
        final Tester tester = new GreedyTester(new IStoresOperatorTest());
        tester.setRandom(new Random());
        tester.addListener(new StopOnFailureListener());
        tester.addListener( name: "verbose");
        tester.addCoverageMetric(new TransitionPairCoverage());
        tester.addCoverageMetric(new StateCoverage());
        tester.addCoverageMetric(new ActionCoverage());
        tester.generate( length: 50);
        tester.printCoverage();
    }
}
```

As can be seen, the initial state is WELCOME_PAGE, where I am on the home page. After that, the algorithm tests each state as it sees fit.

In this class, I use 2 boolean variables that indicate whether the CART_POPUP popup has popped up on the CATEGORY or PRODUCT page. In the stayOnSite() method, I then use these variables to determine if I have returned to the CATEGORY or PRODUCT page.