

# Automation Capstone - Medicare

## Table of Contents

<b>Introduction.....</b>	<b>- 2 -</b>
<b>Problem Statement.....</b>	<b>- 2 -</b>
<b>Requirements Detail .....</b>	<b>- 2 -</b>
<b>Project Sprint.....</b>	<b>- 2 -</b>
<b>Testing Scenarios, Cases &amp; Flow.....</b>	<b>- 3 -</b>
1. <i>API Testing.....</i>	<i>- 3 -</i>
2. <i>User based validations &amp; Backend .....</i>	<i>- 3 -</i>
1.     Site Map Test Suite.....	- 3 -
2.     Shopping Test Suite .....	- 4 -
3.     Admin Action Test Suite .....	- 4 -
<b>Project Structures &amp; Snapshot .....</b>	<b>- 5 -</b>
<i>PostMan.....</i>	<i>- 5 -</i>
<i>Eclipse .....</i>	<i>- 6 -</i>
Project Structure & TestNg XML file .....	- 6 -
Page Objects & Test Suite Class Files .....	- 6 -
TestNg Execution Result.....	- 7 -
Result details in Console .....	- 7 -
Log File .....	- 7 -
MySQL Connection & Usage .....	- 8 -
Maven Test.....	- 8 -
<i>GitHub.....</i>	<i>- 9 -</i>
<i>Jenkins Job .....</i>	<i>- 9 -</i>
<b>Conclusion .....</b>	<b>- 9 -</b>

## Introduction

- This is the project report for the Project 2: **Testing a Healthcare Website** for “Automation Test Engineer Capstone” course of Simplilearn.
- Project submitted by : Krishnaveni Rajan
- Email Id: [krishnaveni07.rajan@gmail.com](mailto:krishnaveni07.rajan@gmail.com)
- Submission Date: 26-November-2021
- Language/Tools Used : Java, TestNg, Selenium Web driver, JDBC connectors, PostMan, Eclipse
- Automation Framework – Page Object Module
- CI/CD : GitHub, Jenkins, Maven

## Problem Statement

This project requires end-to-end development of a comprehensive QA and test environment for a healthcare website. This QA and test environment should be inclusive of the following testing layers:

1. Browser-based end user testing using Selenium WebDriver
2. Unit testing for back-end elements of the website using TestNG
3. API testing with Postman on AWS cloud
4. Automating the whole testing process by a Jenkins job

The end-deliverables will be executable scripts and modules, which can be run on demand to test the web app

## Requirements Detail

1. Write an automation script using page object design to store the web elements of the home page
2. Create Selenium scripts to test all the pages on the website
3. Perform unit testing for all back-end classes and methods using TestNG
4. Create Postman scripts to test the API endpoints mentioned
5. Create and build Jenkins job for all the automation testing phases performed in the previous steps

## Project Sprint

- Iterations – 5
  1. Postman for API testing
  2. Site Maps implementation using TestNg

- 3. End user shopping using TestNg
- 4. Admin actions using TestNg
- 5. Documentation & Final testing
- No. of days in Each iterations – Iterations 1-4 = 4 days & Iteration 5 – 2 days
- Sprint Team Size - 1

## Testing Scenarios, Cases & Flow

### 1. API Testing

- Environment variable set created for IP, Port in the name ‘Medicare’
- Access the APIs using Postman under collections
- Cases covered
  - Home page
  - Membership signup page
  - Login page
  - All Products list page
  - Product category – 1, 2 and 3
  - Contact page
  - Logout
- Exceptions handled using try, catch and throws
- Test conditions used
  - Response code expected vs actual received validation
  - Title of the Page that was part of Response vs expected title.

### 2. User based validations & Backend

- Project Creation in Eclipse with Java, Maven which includes the dependency of TestNg, Selenium Web driver, MySQL connector
- Page Object Modelling used to collect the Web elements and have methods to perform the actions and assert conditions.
- ‘pom.xml’ and ‘testng.xml’ used for Project building and testing

#### 1. Site Map Test Suite

- Test annotations used for validating the Pages of Medicare without user Login.
- Use case pages covered
  - Home page
  - Contact page
  - About page
  - View Product page

- View Antipyretics Product page
  - View Analgesics Product page
  - View Antibiotics Product page
  - Sign up – Personal page
  - Login page
  - Web elements collected using Xpath, Id and Link
2. Shopping Test Suite
- Test annotations used for validating the purchase of user defined medicines by logging in as End user
  - Use cases covered
    - Login with invalid username
    - Login with invalid password
    - Login with valid username and password
    - View Product page
    - Add 2 user defined medicine to cart from product list
    - Validate the medicines in the cart
    - Select default shipping address
    - Make Payment
    - Confirm Order was placed successfully
    - Log out
  - Web elements collected using Xpath, Id and Link with various Hard Asserts, Soft Assert, Iterators, Loops.
3. Admin Action Test Suite
- Test annotations used for validating the addition and edit of values defined by Admin user
  - Use cases covered
    - Login with valid username and password
    - View Manage Product page
    - Add medicine
    - Edit Medicine quantity
    - Confirm updated Medicine Quantity with query data pulled from MySQL database
    - Log out
  - Web elements collected using Xpath, Id and Link with various Hard Asserts, Soft Assert, Iterators, Loops.

# Project Structures & Snapshot

## PostMan

The screenshot shows the Postman interface with the 'AutomationCapstone' workspace selected. On the left, there are sections for 'Collections', 'APIs', and 'Environments'. The 'Medicare' collection is currently selected. The 'Globals' section contains variables: 'ip' (localhost), 'port' (8080), and 'xsrf-token' (undefined). A table at the bottom allows for adding new variables.

The screenshot shows the Postman Runner interface. The 'AutomationCapstone' workspace is selected. The 'RUN ORDER' section lists a sequence of API requests under the 'Common' collection. The requests include: GET Home Page, GET Membership, GET Login Page, GET All Product, GET Product Category 1, GET Product Category 2, GET Product Category 3, GET Show Cart, GET Cart 2, GET Cart Validation, GET Shipping, GET Contact, and GET Logout. To the right of the run order, there are settings for 'Iterations' (1), 'Delay' (0), and various checkboxes for response handling and cookie management. A 'Run Common' button is at the bottom right.

The screenshot shows a detailed view of a POST request for the 'Home Page'. The URL is set to `http://{{ip}}:{{port}}//medicare/`. The 'Tests' tab is active, displaying a JavaScript code block that uses cheerio to parse the response and check if the title equals 'Medicare - Home'. Other tabs visible include 'Params', 'Authorization', 'Headers (8)', 'Body', 'Pre-request Script', and 'Settings'.

```

1 const resp = cheerio.load(pm.response.text());
2
3 pm.test("Home Page", function () {
4     pm.response.to.have.status(200);
5     pm.expect(resp("title").text()).to.equal("Medicare - Home");
6     console.debug("Received Response Code" + responseCode.code);
7     console.debug("Received page title is " + resp("title").text());
8 });

```

Below the request, a 'Response' section is visible.

## Eclipse

### Project Structure & TestNg XML file

The screenshot shows the Eclipse IDE interface. The Project Explorer on the left displays the project structure for 'medicarecaps' with packages like medicare, medicarecaps, and src. The TestNG XML editor on the right shows the content of 'testng.xml'.

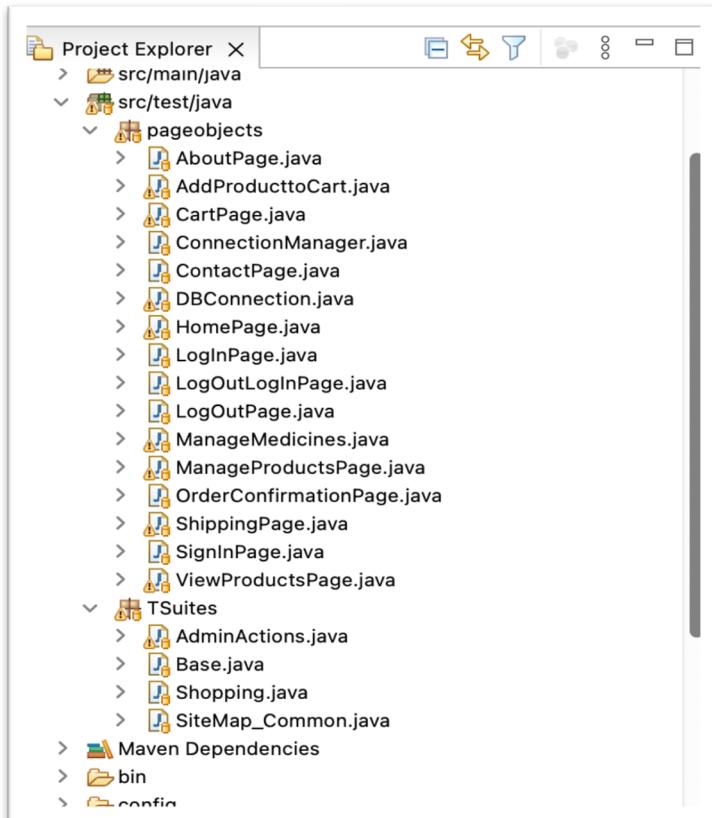
```

<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE suite SYSTEM "https://testng.org/testng-1.0.dtd">
<suite name="Medicare Capstone">
  <test name="TSuites">
    <classes>
      <class name="TSuites.SiteMap_Common" />
      <class name="TSuites.Shopping">
        <parameter name="username" value="tu@gmail.com"/>
        <parameter name="password" value="12345"/>
        <parameter name="ivusername" value="tu"/>
        <parameter name="epassword" value="" />
        <parameter name="med1" value="Combiflame"/>
        <parameter name="med2" value="Amoxicillin"/>
      </class>
      <class name="TSuites.AdminActions">
        <parameter name="ausername" value="vk@gmail.com"/>
        <parameter name="apassword" value="admin"/>
        <parameter name="name" value="Ibuprofen 400mg"/>
        <parameter name="brand" value="Reddys"/>
        <parameter name="unitPrice" value="10"/>
        <parameter name="category" value="1"/>
        <parameter name="medname" value="Paracetamol"/>
        <parameter name="qty" value="100"/>
      </class>
    </classes>
  </test>
</suite>

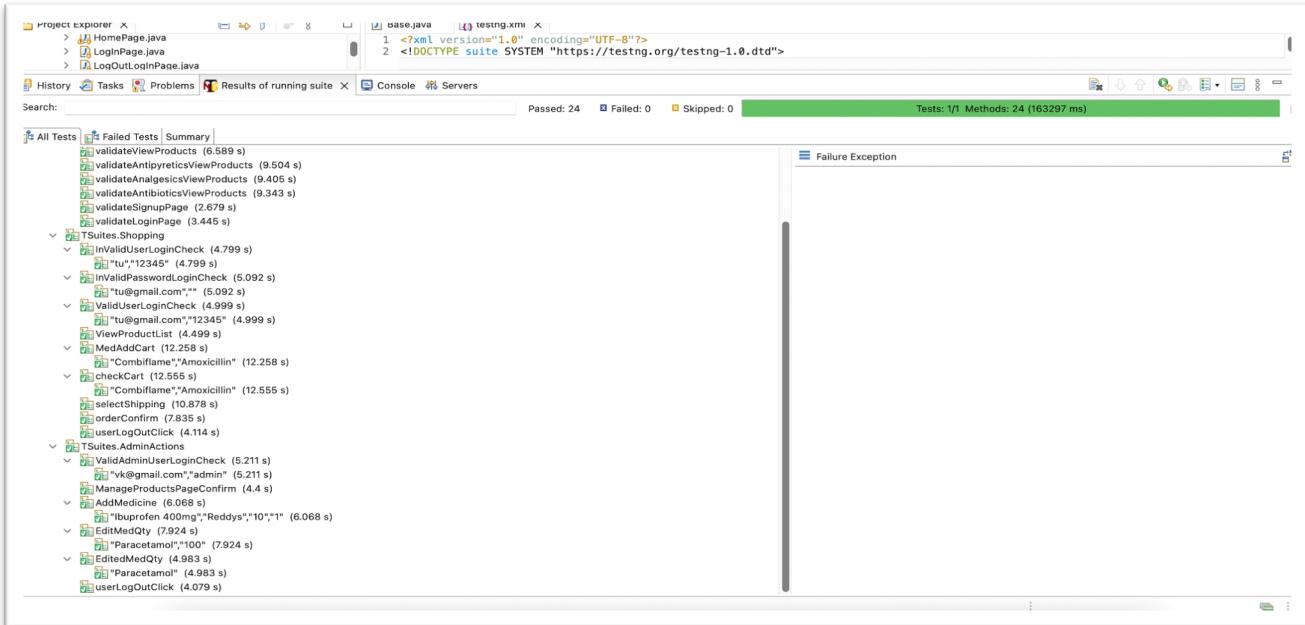
```

The bottom status bar shows 'Tomcat v8.5 Server at localhost [Stopped, Synchronized]' and 'medicare [Synchronized]'. The menu bar includes History, Tasks, Problems, TestNG, Console, and Servers.

### Page Objects & Test Suite Class Files



## TestNG Execution Result



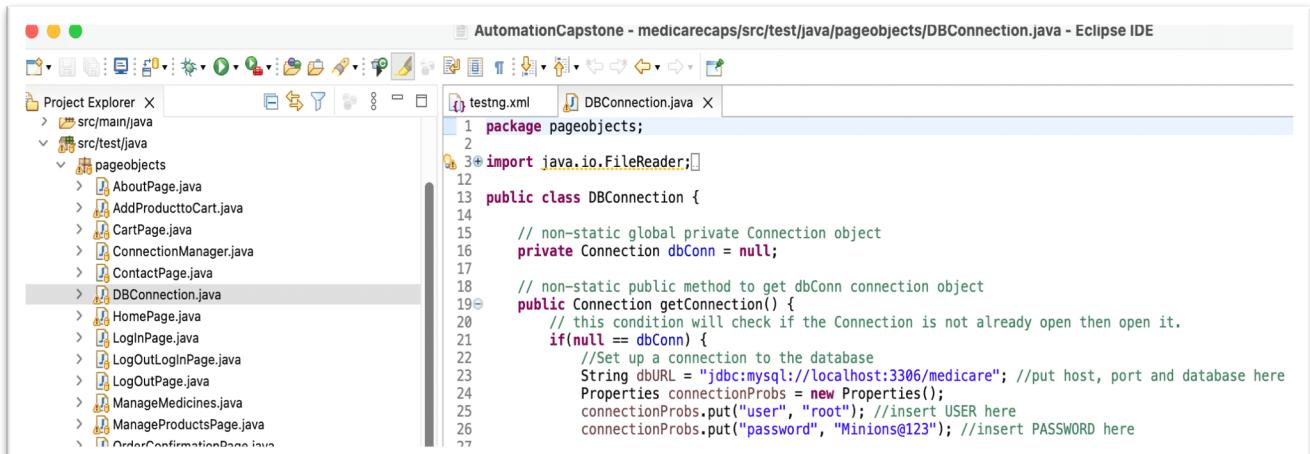
## Result details in Console



## Log File

```
~/CodeBase/AutomationCapstone/medicarecaps/logs -- view testrunlog.txt
[25-Nov-2021 09:33:45] :: [INFO] :: Setup for testing initiated.
[25-Nov-2021 09:33:56] :: [INFO] :: Medicare Website Launched
[25-Nov-2021 09:33:58] :: [INFO] :: About page validation passed
[25-Nov-2021 09:34:01] :: [INFO] :: Home Page Passed
[25-Nov-2021 09:34:05] :: [INFO] :: Contact Us page test passed
[25-Nov-2021 09:34:10] :: [INFO] :: View Product Page validation Passed
[25-Nov-2021 09:34:19] :: [INFO] :: Antipyretics validation Passed
[25-Nov-2021 09:34:29] :: [INFO] :: Analgesics validation Passed
[25-Nov-2021 09:34:38] :: [INFO] :: Antibiotics Page validation Passed
[25-Nov-2021 09:34:43] :: [INFO] :: Sign Up - Personal page validation passed
[25-Nov-2021 09:34:46] :: [INFO] :: Login page validation passed
[25-Nov-2021 09:34:52] :: [WARN] :: Error Message on the Login Page ---> Please enter a valid email address!
[25-Nov-2021 09:34:52] :: [INFO] :: Invalid user message validation passed
[25-Nov-2021 09:34:57] :: [WARN] :: Error Message on the Login Page ---> Please enter your password!
[25-Nov-2021 09:34:57] :: [INFO] :: Login with proper username and password validation passed
[25-Nov-2021 09:35:02] :: [INFO] :: Login Successful for tu@gmail.com
[25-Nov-2021 09:35:06] :: [INFO] :: Total number of products in the page = 5
[25-Nov-2021 09:35:06] :: [INFO] :: Products listed in the current page are :
[25-Nov-2021 09:35:06] :: [INFO] :: Paracetamol
[25-Nov-2021 09:35:07] :: [INFO] :: Combiflare
[25-Nov-2021 09:35:07] :: [INFO] :: Diclofenac
[25-Nov-2021 09:35:07] :: [INFO] :: Aceclofenac
[25-Nov-2021 09:35:07] :: [INFO] :: Amoxicillin
[25-Nov-2021 09:35:07] :: [INFO] :: All Products list page validation Passed
[25-Nov-2021 09:35:11] :: [INFO] :: Product not matched
[25-Nov-2021 09:35:11] :: [INFO] :: Product Added : Combiflare
[25-Nov-2021 09:35:15] :: [INFO] :: Product not matched
[25-Nov-2021 09:35:15] :: [INFO] :: Product not matched
[25-Nov-2021 09:35:15] :: [INFO] :: Product Added : Amoxicillin
[25-Nov-2021 09:35:19] :: [INFO] :: Current URLhttp://localhost:8990/medicare/cart/show?result=added
[25-Nov-2021 09:35:19] :: [INFO] :: Add product to cart validation passed
[25-Nov-2021 09:35:24] :: [INFO] :: 2
[25-Nov-2021 09:35:26] :: [INFO] :: Medicines Combiflare and Amoxicillin successfully added in the cart
[25-Nov-2021 09:35:26] :: [INFO] :: Products in the cart validation passed.
[25-Nov-2021 09:35:28] :: [INFO] :: Checkout is enabled to proceed.
[25-Nov-2021 09:35:35] :: [INFO] :: [[ChromeDriver: chrome on MAC (44921d62e876c38904e98a9fe1aff005)] -> link text: Select]
[25-Nov-2021 09:35:35] :: [INFO] :: Shipping address selection validation passed
[25-Nov-2021 09:35:37] :: [INFO] :: Payment validation passed
[25-Nov-2021 09:35:46] :: [INFO] :: Your Order is Confirmed!!
Expected: Your Order is Confirmed!!
[25-Nov-2021 09:35:46] :: [INFO] :: Order Confirmation Page validation passed
[25-Nov-2021 09:35:49] :: [INFO] :: Continue Shopping Button validation passed
[25-Nov-2021 09:35:54] :: [INFO] :: Logout button validation passed
[25-Nov-2021 09:35:54] :: [ERROR] :: org.openqa.selenium.ElementNotInteractableException: element not interactable
```

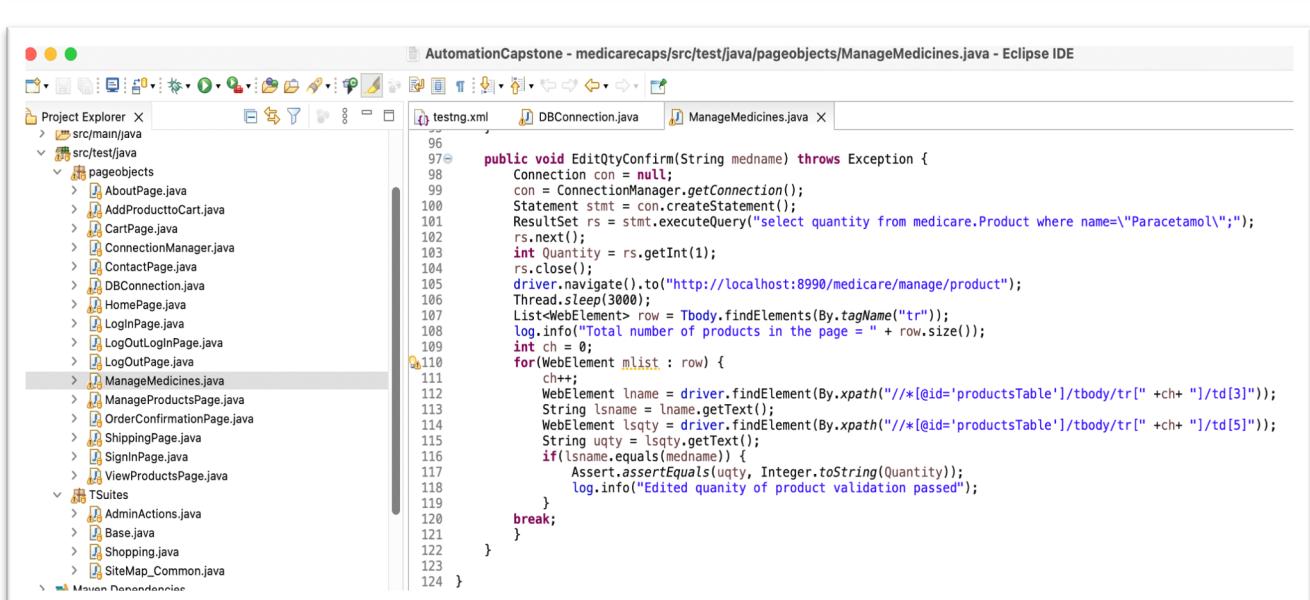
## MySQL Connection & Usage



```

package pageobjects;
import java.io.FileReader;
public class DBConnection {
    // non-static global private Connection object
    private Connection dbConn = null;
    // non-static public method to get dbConn connection object
    public Connection getConnection() {
        // this condition will check if the Connection is not already open then open it.
        if(null == dbConn) {
            //Set up a connection to the database
            String dbURL = "jdbc:mysql://localhost:3306/medicare"; //put host, port and database here
            Properties connectionProps = new Properties();
            connectionProps.put("user", "root"); //insert USER here
            connectionProps.put("password", "Minions@123"); //insert PASSWORD here
        }
    }
}

```

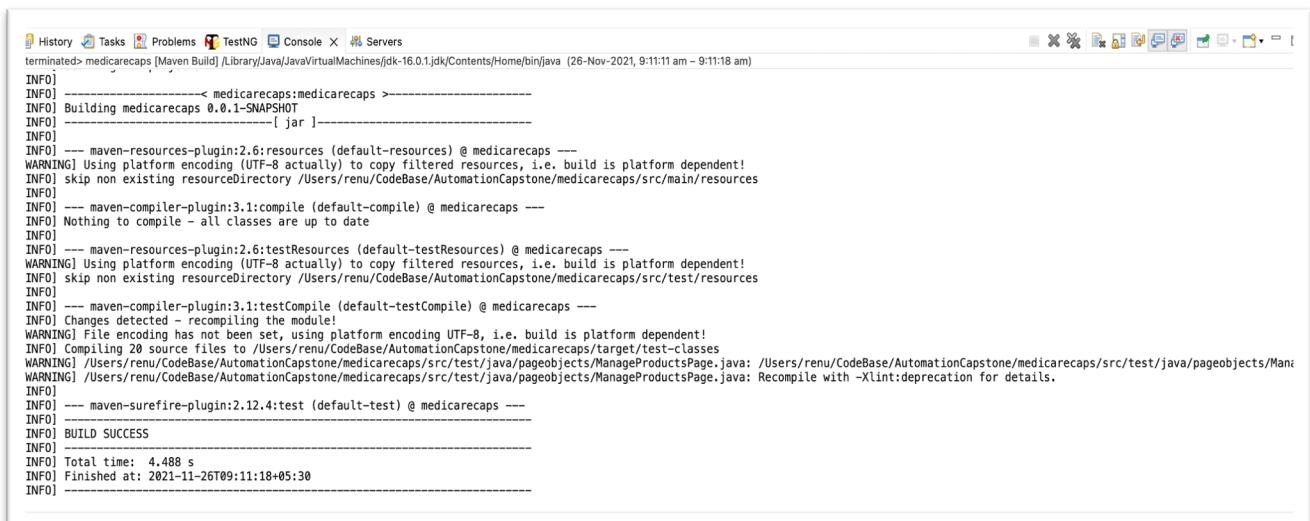
  


```

public void EditQtyConfirm(String medname) throws Exception {
    Connection con = null;
    con = ConnectionManager.getConnection();
    Statement stmt = con.createStatement();
    ResultSet rs = stmt.executeQuery("select quantity from medicare.Product where name='Paracetamol'");
    rs.next();
    int Quantity = rs.getInt(1);
    rs.close();
    driver.navigate().to("http://localhost:8990/medicare/manage/product");
    Thread.sleep(3000);
    List<WebElement> row = tbody.findElements(By.tagName("tr"));
    log.info("Total number of products in the page = " + row.size());
    int ch = 0;
    for(WebElement mlist : row) {
        ch++;
        WebElement lname = driver.findElement(By.xpath("//*[@id='productsTable']/tbody/tr[" +ch+"]/td[3]"));
        String lname = lname.getText();
        WebElement lsqty = driver.findElement(By.xpath("//*[@id='productsTable']/tbody/tr[" +ch+"]/td[5]"));
        String uqty = lsqty.getText();
        if(!lname.equals(medname)) {
            Assert.assertEquals(uqty, Integer.toString(Quantity));
            log.info("Edited quantity of product validation passed");
        }
        break;
    }
}

```

## Maven Test



```

terminated> medicarecaps [Maven Build] /Library/Java/JavaVirtualMachines/jdk-16.0.1.jdk/Contents/Home/bin/java (26-Nov-2021, 9:11:11 am - 9:11:18 am)
INFO] -----> medicarecaps:medicarecaps <-----
INFO] Building medicarecaps 0.0.1-SNAPSHOT
INFO] [jar]
INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ medicarecaps ---
WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
INFO] skip non existing resourceDirectory /Users/renu/CodeBase/AutomationCapstone/medicarecaps/src/main/resources
INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ medicarecaps ---
INFO] Nothing to compile - all classes are up to date
INFO] 
INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ medicarecaps ---
WARNING] Using platform encoding (UTF-8 actually) to copy filtered resources, i.e. build is platform dependent!
INFO] skip non existing resourceDirectory /Users/renu/CodeBase/AutomationCapstone/medicarecaps/src/test/resources
INFO] 
INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ medicarecaps ---
INFO] Changes detected - recompiling the module!
WARNING] File encoding has not been set, using platform encoding UTF-8, i.e. build is platform dependent!
INFO] Compiling 20 source files to /Users/renu/CodeBase/AutomationCapstone/medicarecaps/target/test-classes
WARNING] /Users/renu/CodeBase/AutomationCapstone/medicarecaps/src/test/java/pageobjects/ManageProductsPage.java: /Users/renu/CodeBase/AutomationCapstone/medicarecaps/src/test/java/pageobjects/ManageProductsPage.java: Recompile with -Xlint:deprecation for details.
INFO] 
INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ medicarecaps ---
INFO] 
INFO] BUILD SUCCESS
INFO] 
INFO] Total time: 4.488 s
INFO] Finished at: 2021-11-26T09:11:18+05:30
INFO] 

```

## GitHub

## Jenkins Job

## Conclusion

Automating testing for any product that needs to be tested repeatedly is very vital. Usage of Java language with Selenium web driver for testing the Web based product is good with Page object model. It is easily maintainable and reusable. Making use of TestNG enables creating suites as per required Test Plan using different options file Enabled, Groups. Integration of the code with GitHub enables having the base code available anywhere and maintain different branches. Medicare application automation has enabled me to implement different learning I had achieved in different phases of this Automation Test Engineer Master course. This also have helped me to debug, search for solutions, fix the issues and lot more that will be required for an ideal real time product automation.