# **CA650 Software Process Quality**

|  |  |
| --- | --- |
| Name | Aditya Gupta |
| Student no. | 19210195 |
| Programme | MSc. Cloud Computing |
| Module Code | CA650 |
| Assignment title | Test Tool Evaluation Project |
| Submission Date | 27th March, 2020 |
| Module Coordinator | Renaat Verbruggen |

I declare that this material, which I now submit for assessment, is entirely my own work and has not been taken from the work of others, save and to the extent that such work has been cited and acknowledged within the text of my work. I understand that plagiarism, collusion, and copying are grave and serious offences in the university and accept the penalties that would be imposed should I engage in plagiarism, collusion or copying. I have read and understood the Assignment Regulations set out in the module documentation. I have identified and included the source of all facts, ideas, opinions, and viewpoints of others in the assignment references. Direct quotations from books, journal articles, internet sources, module text, or any other source whatsoever are acknowledged, and the source cited are identified in the assignment references. This assignment, or any part of it, has not been previously submitted by me or any other person for assessment on this or any other course of study.

I have read and understood the referencing guidelines found recommended in the assignment guidelines.

Name: Aditya Gupta Date: 02 April 2020

**TESTING OF WEB APPLICATIONS WITH SELENIUM**

**Introduction:**

Testing is a repetitive and iterative process. So, what does Selenium do? It automates routine tests and making work less boring. Selenium is restricted only to web application testing and no mobile applications testing. It works in different browsers such as Google Chrome, Firefox, Opera, Safari and among others can be accommodated in different operating systems such as iOS, Windows, Linux and it can also be integrated with different programming languages such as Perl, c-sharp, PHP and among others including java.

**Inadequacies in Selenium?**

Selenium does not have an inbuilt object repository and that’s an inadequacy in Selenium. Objects which I am referring here is nothing but a centralised repository where objects information can be stored. Now this object repository acts as an interface between the test script and the application to identify the objects during the execution of test cases, so the test script is written in a programming language and here I used java so java has certain objects in it and the web application will be tested upon using this test script So, this in build object repository helps in identifying the objects in the test script when the web application is being tested. If suppose an in-built object repository is not there so in a scenario like that, we will have to access multiple file and that will make it tedious, now this feature that is the feature of inbuilt object repository can be included in selenium by integrating it with Git.

**What does web application testing involve?**

So, web application testing involves testing the functionality, usability and consistency. So, functionality testing involves image testing as well applications have images embedded in them right so we need to check if the image are performing as required or not selenium cannot perform this on its own and hence it needs to be integrated with a tool known as ‘Sikuli’. Selenium is an open source software as it is built and developed by different people and hence it has no official support, but this is compensated by community support. Now coming to test reports selenium cannot generate test reports so in these test reports they record the data which has been obtained from an evaluation experimental and in an organized manner so it shows the comparison of test results with the test objectives, in simple words what it means is we tested web application and we get certain outcomes so these outcomes are recorded in one particular column and how these web application actually performing are recorded in another column. So, as we now know that these test reports are quite essential so selenium cannot generate these reports, but it can generate in one case if it is integrated with ‘testNG’ tool.

**Why do we need a programming language and What’s the benefit of using a programming language?**

We use programming language so that we can keep track of data flow and it makes our work much easier. The first thing is DDT i.e. data driven testing it is a test automation framework which stores test data in a table so it enables a single test script to execute tests for all the test data suppose if we have different values that we want to provide or we have different datasets so we can store these datasets or whatever data we have in an excel file say in a table in an excel file and we write a test script in java and now both of these can be integrated so instead of giving different data values every single time so all we have to do is to provide the data set on a table and impose that data set into the test script so this is known as DDT, now secondly we need to run test cases now in those test cases we provide hard-coded input values so when we have these hard coded input values for different data sets every single time now so to reduce the iterative nature of work we include parameters now this brought about by parameterization so what happens in parameterization? In this we can provide different input values through the external parameters so now we have only parameters in our test script and don’t have any input values so, these parameters accept different input values so now what is the utility of this? It enables us to reuse a test commands because there are no hard-coded values out there, we can share the test data and create more flexible test process. Now DDT and parameterization both are dependent on programming and hence we need an IDE. Selenium has an IDE of its own, but it is meant only for prototyping and hence we needed an external IDE, so this is where programming comes into picture.

**Now why Java is preferred?** **What is the speciality of Java?**

When java came into existence, when it was created one of the major reasons came was to have a programming language that was architecture neutral which was platform independent i.e. not dependent on any platform.

Why java is preferred specifically for selenium? There are 3 primary reasons the first one being IDE. Generally, IDE’s provide a code editor, compiler or interpreter and a debugger rather developer access is through a unified GUI but java IDE ‘s also includes language specific elements such as ‘ant’ and ’maven’ built tools as well as ‘Junit’ and ‘testNG’ testing. Secondly, java programs are now even more performant thanks to JIT compiler, and their improved JVMs. Also the performance of Java programs can be optimized in real time to help run it faster now Java is a preferred language for developing web applications that involve serious business processes and database access on the server so the application that are widely tested are generally written in Java and also the Java’s promise of write once and run anywhere is quite powerful. So, these are the 3 primary reasons why java is preferred.

**About Selenium web driver:**

So, this is where Selenium web driver comes into picture, so product is developed for multiple platforms here by product I mean the different browsers, so the choice of platform is endless and hence cross-platform testing is performed to determine the behaviour of the applications in different environments. Say user interface of a web application appears correctly in the chrome browser but the layout might be disturbed in the IE browser this is not preferred by the developer and hence cross-platform testing becomes essential. Initially we had something known as selenium RC, meaning remote control now selenium RC acted as a proxy server but now this has been merged with selenium web driver and hence the web driver directly communicates with the browser.

**Integration of Java program with Selenium:**

**A picture containing food, drawing

Description automatically generated A picture containing clock

Description automatically generated**

**Invokes**

Figure 1: Java invokes the Selenium web-driver

Selenium web driver acts as a programming interface to create and execute the test cases so the

test script which is out here which is written in Java invokes the selenium web driver which runs

the test cases on different browsers so selenium web driver per se is written in Java and hence an understanding of the java language is quite beneficial.

**Expected output of test cases:**

In this project, when we run the test script then the Google chrome browser will be launched and then web application will be opened which is Amazon.co.uk now the window will be maximized after this in the search text box I’m going to type in ‘Stephen King’ so I can send text also into web applications and this whole process/entire thing is automated and then the search results will come and then I’m going to go ahead and choose ‘the stand’ book of Stephen King and then I’m going to click on it so once I click on it then will click the ‘Hardcover’ which is the format of the book and then I’m going to add this book to the cart and then I’m just going to scroll down the webpage just to show that scroll down can also be performed next I’m going to open a new demo application to test high iframes and from there I’m going to shift the control to the inner iframe and inside the text box I’m going to type in “HEY” and then close the browser so this is all going to happen in this project.

**Implementation of Java Selenium**

So, first is the integration of java with selenium and we have three agenda here firstly the installation of software of Java Eclipse and selenium. Secondly, we would run the test case and finally we will see each test case that has been executed.

**Downloaded and Installed:**

Now firstly I downloaded java from Oracle java website and clicked on development kit 8. As version 8 is mostly compatible with selenium and with all the bugs free. Now after clicking on version 8 as we know that java is available for different operating systems Linux Mac Windows as well, so I have the 64-bit version and downloaded accepting the license agreement. Then after downloading I changed the path and environment variable. Make sure to have both folders jdk and jre as they both are essential.

After downloading java, we need to download Selenium for this we have official website of this and from here I downloaded 2 files one is selenium server standalone jar file and other is selenium zip folder version 3.9 as this version is best compatible with java version 8. In zip folder we have 2 jar files and a library folder and inside this library folder we’ve got multiple jar files. Now when we integrate selenium with java and we start coding in java to test web applications these jar files are essentials and hence we need to download all of these we may not be using everything but we will be using most of them so all these jar files needs to be integrated with java.

After downloading selenium, we need to download the web driver it’s in the same official website of selenium and its available for different browsers like Mozilla, Google Chrome or Opera this is required to test the web application in a browser. I downloaded Google chrome’s latest version web driver and remembered the path of this file as this we need later when we need to integrate selenium with java. And now we are done with the driver installation.

After downloading all this we finally need Eclipse IDE for java enterprise. So, from its official website downloaded Eclipse IDE.

**Implementation:**

I started with integrating the different software’s so the first and foremost thing here is we need to create a workspace in a local system where our files and all the progress will be stored of Eclipse IDE. Next open Eclipse IDE choose and browse the location of workspace then it will load the workbench. After all this we create a new Java project and name it as ‘CA650Assignment’ and ensure that jre environment is of java. And now we have jar files of selenium which we have to import it to the project so for this we go to ‘configure build path’ and in libraries we select ‘add external jars’ so in this we upload selenium server standalone and the other one which we have to upload is zip folder which has the library folder. So after adding external jar files we notice that a new sub-folder named ‘referenced libraries’ has been added in our ‘CA650Assignment’ and in this referenced libraries we have selenium jar file and zip file so in this we can see that all jar files the library folder has been imported. Next I need the chrome driver as well, so I have created a sub folder in ‘CA650Assignment’ as ‘drivers’ and this folder is for chrome driver next I just simply copy and paste the chrome driver here in this folder. So, till here we’ve integrated selenium and chrome drive driver with a java project.

Now I created a package and type in a path as ‘com.assignment’ so this is where it will all be stored and the next, I created a class named ‘launch.java’. After doing the whole setup we then write the code.

**The Code:**

In code the first part is web driver class and I am creating an object out of it because the driver object is very essential and so I made it global. Now coming to the main function you’ll see ‘System.setProperty’ we know that there are different jar files so to access those jar files we need to access its memory location so ‘System.setProperty’ accesses a memory location of it then we got web driver chrome drive this is one of the argument and the other one is location of the chrome driver the location where it is stored in your local system. As we know first, we must launch the Chrome browser, so this driver chrome driver launches our browser using ‘System.setProperty’. The next part is ‘manage()’, ‘timeouts()’ and ‘implicitlywait()’ so in selenium we have 3 types of ways implicit wait, explicit wait and fluent wait. In this project I used implicit wait because it stays in place for the entire duration for which the browser is open so implicit wait ensures that entire test script waits for a certain duration that is a maximum duration is going to wait for 15 seconds till it identifies the web element so just in case if the web element loads within a period of 3 or 4 seconds it’s not going to wait for 15 seconds it's going to proceed with that but if the web element takes longer than 15 seconds to load then it throws up an error saying ‘No such element found ’. So, with this we launched our browser and will wait for certain period and the entire script is going to wait for a certain amount of time for different web elements at different points of time. Next I must open the web application so for this I have used ‘driver.navigate().to()’

Amazon.co.uk.

So, this opens a web application of our choice and navigates it to next once application is opened. And next I have maximized the window all those methods are inbuilt and accessing those methods. Now once the application is opened I have to get the title of web application using ‘getTtitle()’ method so I know the name of the application but this is only for validation that I’ve done so ‘getTitle()’ method access the title of the web application and it is stored in variable title so here I’ve got two strings one is title and the other one is Amazon.co.uk so using the if else statement I have compared the above two strings and once it compares it if both strings are equal, if both of them are same then the output will be ‘Title Matches’ in case if it has a variation in the strings between Amazon.co.uk and the title it will print out the exact title and in our output when we run we get ‘Amazon.co.uk: Low Prices in Electronics, Books, Sports Equipment & more’. I just typed Amazon.co.uk but the actual title is a bigger string and that’s the reason it has printed the string instead of ‘Title Matches’ this is only for validation that we’ve done so this thing is about web application and getTitle()’ method as well as launching a browser.

Now locating web elements so to do this we go to Amazon.co.uk where our applications generally have a number of elements and each web element can be uniquely identified for this we right click on the page and go to inspect element so its inspecting the web application page that we want to select. Now I have to type in ‘Stephen King’ in the text box so there are 3 things which are being done one I need to identify the text box second I have to type in Stephen King and in three I have to click on search button so for this 3 functions were made. So the first thing I have to identify the text box so I’m identifying the text box using ID’s for this I simply inspect the text box and found out the id so this is the part of text box ‘WebElement myElement = driver.findElement(By.id("twotabsearchtextbox"));’. Next using myElement object I typed in ‘Stephen King’ so anytime if I need to type in text into any text box I need to use ‘sendKeys’. So ‘sendKeys’ is what performs sending the text option, so Stephen King is the text which I am sending into the text box. Next I must click on search button for this we inspect search button on the webpage and check for the class so in this we got class as ‘nav-input’ so here I am identifying the search button using its class name. Now after identifying the search button we then must click it for this part we used ‘click()’. After this it will display everything related to Stephen King in amazon.co.uk.

After these the next 3 steps which I have performed was I randomly picked book name as ‘The Stand’ and by using ‘By.partialLinkText()’. Initially I used ‘linkText’ to search for the exact word but here I used ‘partialLinkText’ so just because if someone doesn’t know the exact word. I searched the book naming ‘Stand’ instead of ‘The Stand’ and then clicked it. Second step which I chose was to select ‘Hardcover’ and for this I simply used its ID while using inspect element and then finally clicked it. And then the final step was to ‘Add to Cart’ and I’ve done it using CSS selector.

Below I’ve attached 2 screenshots of the program that is used to test the web application with Selenium.

A screenshot of a computer

Description automatically generated

Figure 2: The Code

A screenshot of a computer

Description automatically generated

Figure 3: The Code Part 2

So at the end I’ve added just a scroll down function now in this I’ve only showed you how to scroll down towards the end of the page so JavaScript executor executes script and the remaining are the arguments ‘0’ and ‘scrollHeight’ and then it says ‘Thread.sleep()’ and again a pause for given seconds so I’ve identified the scroll and then I am scrolling it down towards the end of the page.

So, with this I’ve tested various features on the web application Amazon.co.uk.

After this I’ve tested something known as iframes now amazon.co.uk does not have something as iframes. Iframes is something when there’s more than one HTML document in the same web page so we need to shift the control from one HTML to another HTML i.e. not there in amazon.co.uk so I ran it on a demo testing web server. For this I identified the frame using XPath so once I identify the frame to which I shift control to using the XPath used ‘switchTo()’ to the tab switch and then I shifted the frame of my choice and then inside the frame to which have shifted the control to there a ‘text box’ and in that text box I’ve typed in “HEY” just only to show that control has been shifted from one frame to another frame then is going to identify the web element using XPath first to identify the frame shift the control then is to identify the text box which is there in the new frame and then ‘sendKey’ to send text into the web elements into a text box. Finally used ‘driver.close()’ and ‘driver.quit()’ so with this we closed the browser. Here you can see one thing that if the application has not been launched completely till then selenium will not start testing meaning it should load completely in order to selenium to start testing.

**Result:**

When we run the test script then the following will happen consecutively, and we achieve our desired output:

1. Google Chrome browser will be launched.
2. Opening the web application i.e. Amazon.co.uk.
3. Window will be maximized.
4. Automated type of ‘Stephen King’.
5. Click on text ‘search’ button.
6. Search for the book naming ‘Stand’.
7. Click the book naming ’The Stand’.
8. Click ‘Hardcover’ which is format of the book.
9. ‘Add to Cart’ button will be clicked.
10. Scroll down to last page.
11. Shifts control to the inner frame.
12. In text box of iframe typing “HEY”
13. Finally closes the browser after successful execution of all steps.

The output of this program ‘SPQ.mp4’ is attached in this zip folder a for the references the screenshot of 9th  and 12th output is attached below:

**A screenshot of a cell phone

Description automatically generated**

Figure 4: Output of 9th test case

**A screenshot of a computer

Description automatically generated**

Figure 5: Output of 12th test case

**About Test Coverage**

Test coverage is an essential part of software testing it's defined as a metric and software testing that measures the amount of testing performed by a set of tests, by the amount of testing we mean that what part of the application program are exercised when we run a group of tests. In simple terms test coverage measures the effectiveness of our testing so when we can count to bond some things in the application and also tell whether the test cases are covering those things of the application then we can say that how much our test cases have covered so for example if we have four features and we tested only three of them then our test coverage is 75% of the features or if we have 1,000 lines of code and our tests visited or exercised 600 lines of those lines then our test coverage is 60% so the effectiveness of testing is not measured by the number of test cases but rather by how much those test cases can cover.

So, we have **Syntax Based Test Coverage** in this the model is basically syntax of the software artefact. BNF grammars for programming languages, modelling languages for design models like UML diagrams and FSMs, XML for specification etc. Syntax can be used to generate artefacts that are valid and those are invalid. The structures/artefacts that we generate are sometimes test cases but most of the time they are not. They are generally used to generate test cases.

Syntax of many programming language is given in three levels:

Words: The lexical level, defines how characters from tokens generally specified using regular expressions.

Phrases: The grammar level, determines how tokens form phrases generally specified using(deterministic) context free grammars in Backus-Naur form.

Context: Deals with types of variable, what they refer to etc. Generally specified using context-sensitive grammars

Generally, syntax testing relies on description of software artefacts. So, tests are created with two general goals i.e. first is to cover the syntax in some way that generated artefacts are valid (correct syntax) and the other is to violate the syntax to generate artefacts that are invalid (incorrect syntax). In this program as we can see that I have followed the proper syntax of Java language.

//Typing the text

WebElement myElement = *driver*.findElement(By.*id*("twotabsearchtextbox"));

myElement.sendKeys("Stephen King");

*driver*.findElement(By.*className*("nav-input")).click();

Thread.*sleep*(2000);

For every test case these are clarified:

* The input of every component which we have used in the program which we have taken from inspecting the particular element which directs to another test case like to insert the text ‘Stephen king’ we used the id of the element as ‘twotabsearchtextbox’ so proper syntax should be used in the statement of the program in order to validate the testing.
* The syntax element to which the mutation is applied like we checked for the mutation testing which states that when we try to mutate certain statements in the source code and then check if the test cases can find the errors. In this program when we take the previous example of text box in which we wrote ‘Stephen King’ we used the id as taken from inspect element from the website ‘twotabsearchtextbox’ and when we try to mutate with different class or id then it will show error and will not proceed further.

Next we have is **Logic Based Test Coverage** is also known as predicate coverage for logic is same as edge coverage for graphs hence mutation subsumes predicate coverage. In logical coverage criteria the ROR, COR and LOR (Relational, Conditional and Logical Operator Replacement) mutation operators will together replace each clause in each predicate with both true and false.

* To kill the true mutant, a test case must cause the clause (and the predicate) to be true.
* To kill the false mutant, a test case must cause the clause (and the predicate) to be true.

And these above two are clause coverage too.

**if**(title.equalsIgnoreCase("Amazon.co.uk"))

System.***out***.println("Title matches");

**else**

System.***out***.println(title);

In this program, I have used different if else statements so basically I have compared the two strings and once it compares it if both strings are equal, if both of them are same then the output will be ‘Title Matches’ in case if it has a variation in the strings between Amazon.co.uk and the title it will print out the exact title and in our output when we run we get ‘Amazon.co.uk: Low Prices in Electronics, Books, Sports Equipment & more’. I just typed Amazon.co.uk but the actual title is a bigger string and that’s the reason it has printed the string instead of ‘Title Matches’ this is only for validation that we’ve done. Condition coverage also known as predicate coverage in which each one of the Boolean expressions have been evaluated to both True and False.

Another coverage method is **Statement Coverage** this criterion is going to be characterized by two aspects the first one is which are the test requirement for the criteria and the second one is how we measure coverage for that criteria. In the case of statement coverage, these test requirements are all the statements in the program. So, this is the basic, the first the simplest coverage criteria in the white box arena. White box testing is based on the assumption that if there isn't a faulty element in the code, we need to exercise it. we need to execute it, in order to find the fault and that's exactly what statement coverage does. If there is a statement that is faulty in the code, we need to exercise it in order to find the fault. And therefore, a good measure of how well we exercise the code, is the ratio of number of executed statements to total number of statements in the program. The higher this number, the better I can exercise the code. So, statement coverage is satisfied when all the statements in the program have been executed and we can satisfy to different degrees and the degrees to which it's satisfied is measured by this value.

Coverage Measures = Number of executed statements / Total number of statements

If we look at this previous code that we used in the program, we can easily find the coverage measures by taking the ratio of the number of executed statements that are present in the program by the total number of statements in the program and it will come out to be 90%

**if**(title.equalsIgnoreCase("Amazon.co.uk"))

System.***out***.println("Title matches");

**else**

System.***out***.println(title);

And about **Performance Testing** if we see in the program it can be inferred that total time of execution of all tests is working as we asked for by using “Thread.sleep();” so when it comes to performance testing its working completely fine. The web browser loads its full website within 5 seconds and if after 15 seconds it does not load the website it will give us the error. We found out performance by determining the speed, responsiveness and stability of a website, software program under workload .

Now, as you can see the coverage percentage the program of web testing with selenium comes out to be 94.6%. By using Eclipse, we can simply run the code as coverage and then it will output the coverage percentage and covered, mixed and total instructions. It’s very simple and intuitive to find the coverage percentage. In the screenshot below, the colours are self-explanatory. The green coloured is all perfectly executed and whereas the yellow coloured means that it ran into a conditional if else statement and this also infers that it only executed one branch.

A screenshot of a computer

Description automatically generated

**Usability of the tool:**

Selenium automation tool is very easy to use, and it is also very compatible with end users. Selenium is a free and open source tool, although it is community based. One can run selenium tests under multiple browser by just adding the required browser driver. Selenium can be integrated with multiple programming languages including C#, java, python, php and others. Selenium tests can also be created manually with the help of different software’s.

**Why selenium is the best choice:**

Being open source it is available for free it does not really matter what operating system we are using also selenium will let you perform tests on almost all web browsers and let you pick which ever programming language we need in order to be comfortable in writing the test cases

Selenium is an automated testing tool I'm fundamentally referring to the software tools that can enable automation or automate the process of testing an application unlike the manual testing tools where every possibility of input output and interaction with the application must be recorded reported and compared manually.

**Accuracy**:

* Validation and verification of software product after development process.
* There should not be any outdated content in the software.
* There should be no broken links or missing links in the software.
* No site errors or poor design layout should be there.

**Advantages**:

* Supports very less CPU and RAM consumption.
* It makes the framework exceptionally compelling and proficient.
* It reveals usability issues before the software is launched.
* Saves time for both company and end users.
* Better user experience.

**Robustness of Selenium tool**

There were many challenges which were faced by the web services such as lack of assurance of trustworthiness, Unavailability of source code hinders white box testing, improper provision of clients input parameters. Now with the selenium automation you can do various things by scaling and running tests on multiple environments simultaneously and by creating bug reproductive scripts. We can also create browser-based automation tests and suites. Web services empower two machines or applications to speak with one another and trade data. The usefulness, convenience, execution, and other non-utilitarian parts of an application rely upon the adequacy of this correspondence. This is the thing that web services testing determines. With regards to automated website testing, one parameter that is critical is the insight accumulated from the yield of your automated test execution for example number of bugs reported. The drawback of less frequent testing is that the bugs may be uncovered during the later phases of advancement which can delay the item discharge. Sooner or later in time, the test group will coordinate Automation testing as a major aspect of the testing stage so throughput can be improved, and manual resources can be utilized for some different undertakings.

**References:**

<https://blog.qatestlab.com/2011/09/16/black-box-test-techniques-syntax-testing/>

<http://www.mcr.org.in/sureshmudunuri/stm/unit6.php>

<https://www.cigniti.com/blog/web-services-test-automation-framework-challenges-and-benefits/>

<https://www.edureka.co/blog/selenium-tutorial>