1. Why did you select Selenium WebDriver for front end automation of your application? ****OR****What factors you will take into consideration before deciding a tool to automate front end of your application?

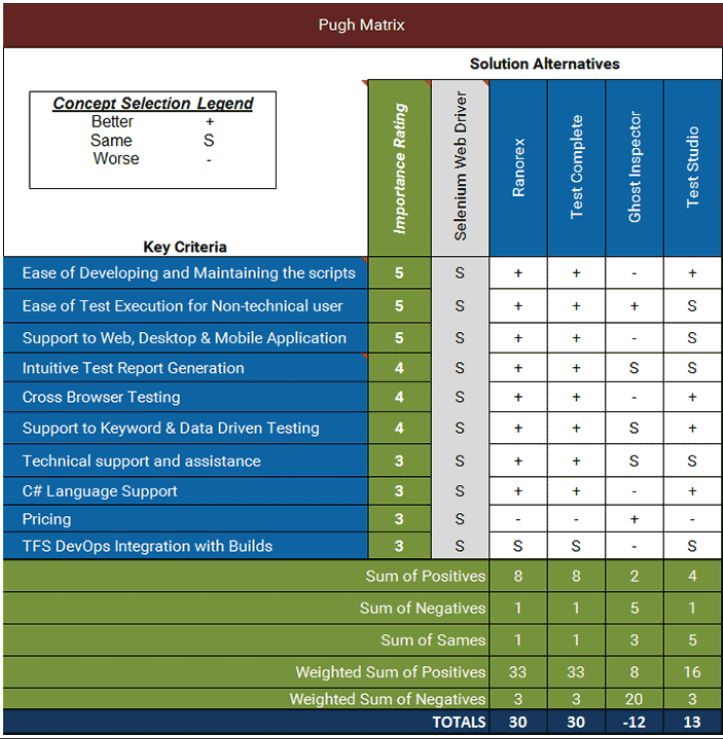
Ans: <https://www.saviantconsulting.com/blog/4-steps-select-test-automation-tool.aspx>

****Step 1: Understand your project requirements thoroughly****

****Step 2: Consider your existing test automation tool as a benchmark****

****Step 3: Identify the key criteria suitable for a project****

****Step 4: Leverage Pugh Matrix Technique for Analysis****



1. How Selenium WebDriver works? ****OR****What is architecture of Selenium WebDriver? OR Why do we require browser drivers?

Ans:

Every browser may have different internal logic of performing actions like loading a webpage ( URL) , closing the browser, getting the title, clicking on an element etc.  Selenium WebDriver plays a role of mediator so a programming language and a browse can communicate easily. Programming statements send commands to browsers through Selenium WebDriver APIs and vice versa. This is the reason we have different language binding of Selenium WebDriver.

Selenium WebDriver APIs can not directly communicate with browsers as well. They also need some mediator. Selenium WebDriver requires exclusive browser executable files ( browser specific server i.e. chromedriver.exe for chrome -Windows ). Selenium WebDriver launches browser specific server first then send instructions provided by programming statements to launched server such as load a URL. To be more technical, WebDriver API that communicate with the browser use a common wire protocol. This wire protocol defines a RESTful web service using JSON over HTTP. Response from browser after execution of command is also sent back to Selenium WebDriver API through the same server.

1. What are the limitations of Selenium WebDriver?

Ans:

* Selenium does not support automation testing for desktop applications.
* Selenium requires high skill sets in order to automate tests more effectively.
* Selenium doesn’t automate barcode,captcha,otp
* Since Selenium is open source software, you have to rely on community forums to get your technical issues resolved.
* We can?t perform automation tests on web services like SOAP or REST using Selenium.
* We should know at least one of the supported programming languages to create tests scripts in Selenium WebDriver.
* It does not have built-in Object Repository like UTF/QTP to maintain objects/elements in centralized location. However, we can overcome this limitation using Page Object Model.
* Selenium does not have any inbuilt reportingcapability; you have to rely on plug-ins like **JUnit** and **TestNG** for test reports.
* It is not possible to perform testing on images. We need to integrate Selenium with **Sikuli** for image based testing.
* Creating test environment in Selenium takes more time as compared to vendor tools like UFT, RFT, Silk test, etc.
* No one is responsible for new features usage; they may or may not work properly.
* Selenium does not provide any test tool integration for Test Management.

1. How you will setup a Selenium WebDriver project from scratch?
2. Which type of Selenium WebDriver framework you used for your application and how did you choose that?

Ans:

Create a Maven project.Add dependencies for Selenium,Test NG,Apache POI etc.,

**POM (Page Object Model pattern) with Data Driven Framework: (Page Chaining Model)**

* **PageLayer - Layer 1**

This contains different java files for different pages like **loginpage.java, homepage.java, registrationpage.java** etc.,These pages contain webobjects/web elements specific to those pages and also functions that are specific those page like signup(),logochk,linkschk() etc.,

* **TestLayer - Layer 2**

This contains different java files for different tests like loginpagetest.java,homepagetest.java etc.,

* **TestBase.java (Concept of Inheritance) - Layer 3**

1. Both PageLayer and TestLayer extends the TestBase.java class which means TestBase will be the **Parentclass** and the PageLayer and TestLayer will be the **child classes** extending the parent class
2. TestBase.java will have the webdriver initialization,Properties File initialization ,Maximizing the window,deleting cookies,PageLoad timeout,implicit wait,get(url)

* **Config.Properties file - Layer 4**

1. This is also referred as environment variables .This contains info regarding browser to use,uname,pwd ,url etc.,

* **TestData.xlsx - Layer 5**

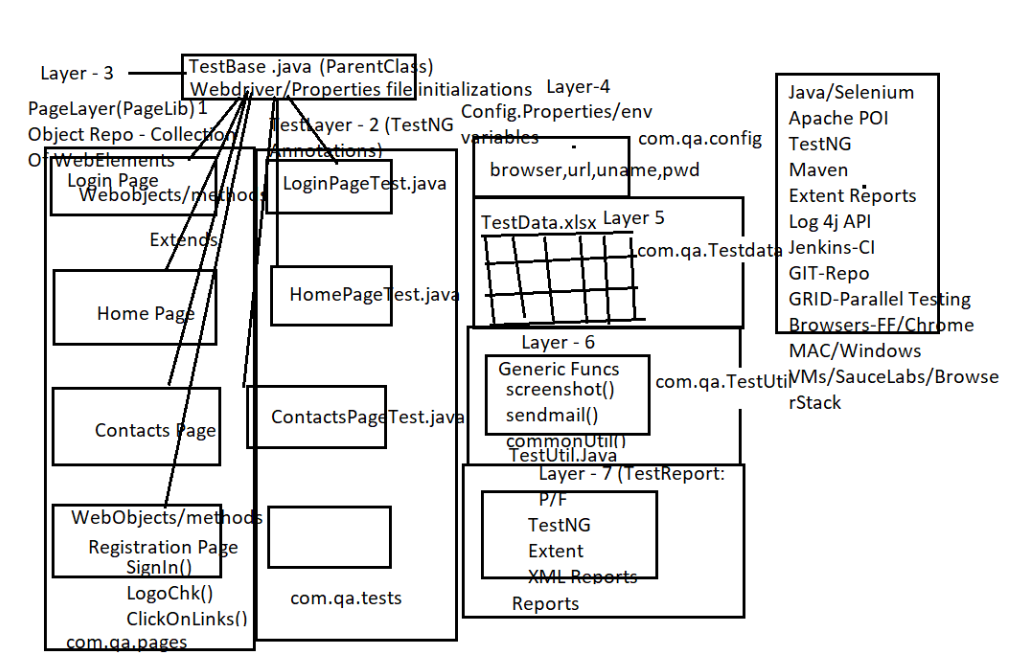
1. This uses Apache POI API to read data from excel sheet for implementing data driven approach.We can have different sheets for different modules

* **TestUtil - Layer 6**

1. Here all the common utilities are listed like Screenshot,extent reports ,Generic functions etc.,

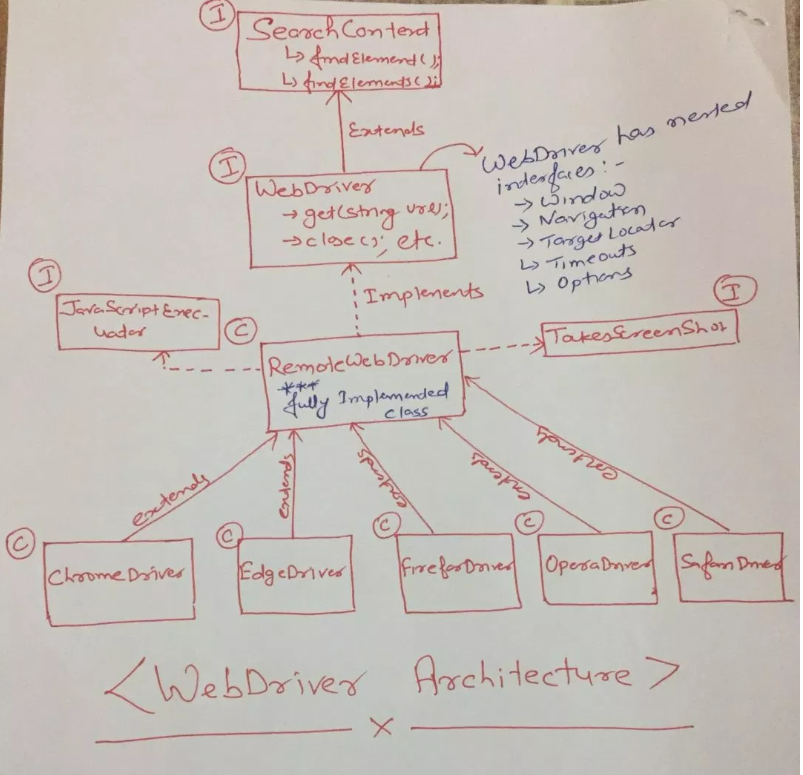
* **Reports Layer 7**

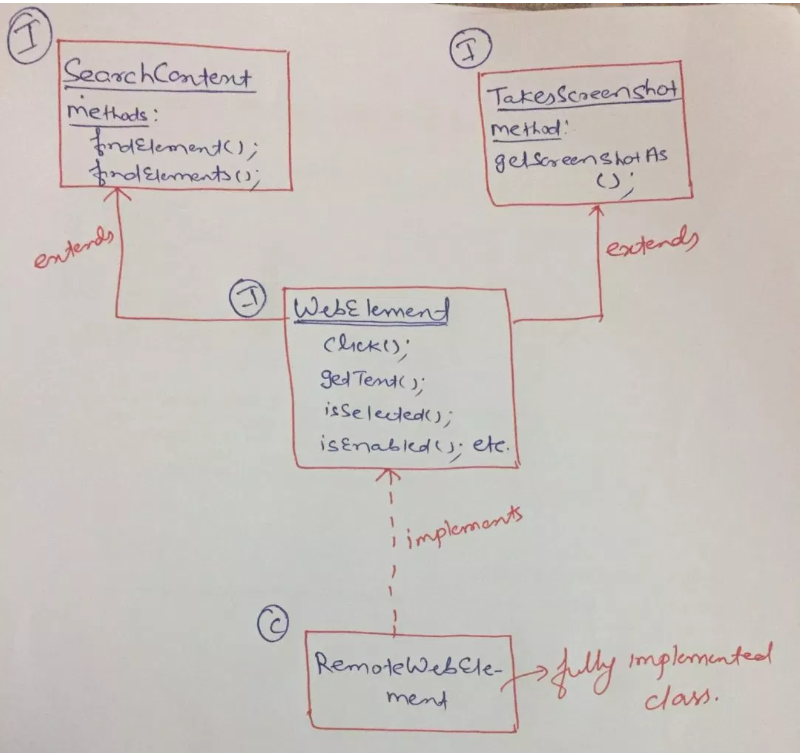
Output reports generated are listed (TestNG reports,xml reports,Extent Reports)



1. Explain hierarchy of Classes and interfaces in Selenium WebDriver.

<http://makeseleniumeasy.com/2017/04/02/hierarchy-of-selenium-classes-and-interfaces/>





1. What problem you faced during automation?

<https://www.cigniti.com/blog/challenges-solutions-in-managing-selenium-test-automation/>

1. ****Testing Flash apps:**** To automate flash apps with Selenium, one can use Flex Monkium. The application source code must be compiled with the swc files generated by Flex Monkium. Then the app and the Selenium IDE are connected, and the tests can be recorded with IDE.
2. ****Unexpected error launching Internet Explorer****. Browser zoom level should be set to 100% by default for the IE browser to overcome this error
3. ****Protected Mode**** must be set to the same valueerror occurs when trying to run Selenium WebDriver on a fresh Windows machine. This issue can be fixed by using capabilities as below when launching IE

The IEDriverServer exectuable must be downloaded and placed in your PATH.

On IE 7 or higher on Windows Vista or Windows 7, you must set the Protected Mode settings for each zone to be the same value. The value can be on or off, as long as it is the same for every zone. To set the Protected Mode settings, choose "Internet Options..." from the Tools menu, and click on the Security tab. For each zone, there will be a check box at the bottom of the tab labeled "Enable Protected Mode".

other possible solutions from that thread include setting the following settings:

var options = new InternetExplorerOptions();

options.IntroduceInstabilityByIgnoringProtectedModeSettings = true;

options.IgnoreZoomLevel = true;

options.UnhandledPromptBehavior = UnhandledPromptBehavior.Accept;

options.EnablePersistentHover = true;

options.EnableNativeEvents = false;

options.EnsureCleanSession = true;

options.ElementScrollBehavior = InternetExplorerElementScrollBehavior.Top;

The [official selenium documentation](https://github.com/SeleniumHQ/selenium/wiki/InternetExplorerDriver/_compare/0cf46848abd86f06876ee6ad0a41ee098c7327a0...74561c47d0300289087a9ea154164ffeb0e9b778) suggests that you need to both:

****1.**** Have your internet explorer's zoom level set to 100% and

If you're running windows 10, you need to "Change the size of text, apps, and other items" in "Display Settings" to 100%. (Make sure that you're doing this action on the monitor that you're running tests on.)

1. How do you handle dynamic web elements in Selenium WebDriver?

Absolute Xpath: but its very fragile and prone to breakage

Xpath methods like :

* Starts-with : //button[starts-with(@id,’Submit-’)]
* Contains :

//button[contains(@id,’Submit-’)]

Xpath=//\*[contains(@type,'sub')]

Xpath=//\*[contains(@name,'btn')]

Xpath=//\*[contains(@id,'message')]

* Using OR & AND:

Xpath=//\*[@type='submit' or @name='btnReset']

Xpath=//input[@type='submit' and @name='btnLogin']

*****driver.findElement(“//i[@class=’ico-header-blankUserImage sprite-header’] | //span[@class=’userBlockLogin’]/i”);*****

*****@FindBy(xpath=”//i[@class=’ico-header-blankUserImage sprite-header’] | //span[@class=’userBlockLogin’]/i”)*****  
*****WebElement someElementName;*****

* Index : driver.findElements(By.Xpath(“//\*submit”).get(0).click();

Xpath Axes can be used to locate nearest stable element:

### **a) Following:**

Selects all elements in the document of the current node( ) [ UserID input box is the current node] as shown in the below screen.

Xpath=//\*[@type='text']//following::input

There are 3 "input" nodes matching by using "following" axis- password, login and reset button. If you want to focus on any particular element then you can use the below XPath method:

Xpath=//\*[@type='text']//following::input[1]

### **b) Ancestor:**

The ancestor axis selects all ancestors element (grandparent, parent, etc.) of the current node as shown in the below screen.

In the below expression, we are finding ancestors element of the current node("ENTERPRISE TESTING" node).

Xpath=//\*[text()='Enterprise Testing']//ancestor::div

There are 13 "div" nodes matching by using "ancestor" axis. If you want to focus on any particular element then you can use the below XPath, where you change the number 1, 2 as per your requirement:

Xpath=//\*[text()='Enterprise Testing']//ancestor::div[1]

### **c) Child:**Selects all children elements of the current node (Java) as shown in the below screen.

Xpath=//\*[@id='java\_technologies']/child::li

### **d) Preceding:**

Select all nodes that come before the current node as shown in the below screen.

In the below expression, it identifies all the input elements before "LOGIN" button that is ****Userid**** and ****password**** input element.

Xpath=//\*[@type='submit']//preceding::input

### **e) Following-sibling:**

Select the following siblings of the context node. Siblings are at the same level of the current node as shown in the below screen. It will find the element after the current node.

xpath=//\*[@type='submit']//following-sibling::input

### **f) Parent:**

Selects the parent of the current node as shown in the below screen.

Xpath=//\*[@id='rt-feature']//parent::div

### **g) Self:**

Selects the current node or 'self' means it indicates the node itself as shown in the below screen.

Xpath =//\*[@type='password']//self::input

### **h) Descendant:**

Xpath=//\*[@id='rt-feature']//descendant::a

1. How do you handle a Web table in selenium webdriver?

public class NofRowsColmns {

public static void main(String[] args) throws ParseException {

WebDriver wd;

System.setProperty("webdriver.chrome.driver","G://chromedriver.exe");

wd = new ChromeDriver();

wd.manage().timeouts().implicitlyWait(5, TimeUnit.SECONDS);

wd.get("http://demo.guru99.com/test/table.html");

//To locate table.

WebElement mytable = wd.findElement(By.xpath("/html/body/table/tbody"));

//To locate rows of table.

List < WebElement > rows\_table = mytable.findElements(By.tagName("tr"));

//To calculate no of rows In table.

int rows\_count = rows\_table.size();

//Loop will execute till the last row of table.

for (int row = 0; row < rows\_count; row++) {

//To locate columns(cells) of that specific row.

List < WebElement > Columns\_row = rows\_table.get(row).findElements(By.tagName("td"));

//To calculate no of columns (cells). In that specific row.

int columns\_count = Columns\_row.size();

System.out.println("Number of cells In Row " + row + " are " + columns\_count);

//Loop will execute till the last cell of that specific row.

for (int column = 0; column < columns\_count; column++) {

// To retrieve text from that specific cell.

String celtext = Columns\_row.get(column).getText();

System.out.println("Cell Value of row number " + row + " and column number " + column + " Is " + celtext);

}

System.out.println("-------------------------------------------------- ");

}

}

}

public class HandlingWebTable {

public static void main(String[] args) {

System.out.println("Execution Starts");

// Setting chrome driver property and opening chrome browser

System.setProperty("webdriver.chrome.driver", "./exefiles/chromedriver.exe");

WebDriver driver= new ChromeDriver();

System.out.println("Browser opened.");

// loading URL

driver.get("C:/Users/Amod Mahajan/Desktop/HTMLTable.html");

// Printing table header of a web table assuming first row as header

System.out.println("Printing all header of table assuming first row as header: ");

List allHeadersOfTable= driver.findElements(By.xpath("//table[@name='BookTable']/tbody/tr[1]/th"));

System.out.println("Headers in table are below:");

System.out.println("Total headers found: "+allHeadersOfTable.size());

**for**(WebElement header:allHeadersOfTable)

{

System.out.println(header.getText());

}

System.out.println("=====================================================================");

// Printing table header of a web table assuming no information about header row

System.out.println("Printing all header of table without information of row header ");

List allHeadersOfTable1= driver.findElements(By.xpath("//table[@name='BookTable']/tbody/tr/th"));

System.out.println("Headers in table are below:");

System.out.println("Total headers found: "+allHeadersOfTable1.size());

**for**(WebElement header:allHeadersOfTable1)

{

System.out.println(header.getText());

}

System.out.println("=====================================================================");

// Finding number of rows in a web table. We need to exclude header to get actual number of data rows

System.out.println("Retrieving total number of data rows:");

List allRows= driver.findElements(By.xpath("//table[@name='BookTable']/tbody/tr"));

System.out.println("Total data rows found in table:"+ (allRows.size()-1));

System.out.println("=====================================================================");

// Find number of columns in each row

System.out.println("Retrieving total number of columns for each row:");

**for**(int i=2;i&lt;=allRows.size();i++)

{

List allColumnsInRow=driver.findElements(By.xpath("//table[@name='BookTable']/tbody/tr["+i+"]/td"));

System.out.println("Number of columns in "+(i-1)+" data row is:"+allColumnsInRow.size());

}

System.out.println("=====================================================================");

//Print each rows and columns from web table

System.out.println("Printing all column value: ");

**for**(int i=2;i&lt;=allRows.size();i++)

{

List allColumnsInRow=driver.findElements(By.xpath("//table[@name='BookTable']/tbody/tr["+i+"]/td"));

**for**(int j=0;j&lt;allColumnsInRow.size();j++)

{

System.out.print(allColumnsInRow.get(j).getText()+" ");

}

System.out.println();

}

System.out.println("=====================================================================");

// List books name and price whose author is mukesh

System.out.println("Way 1: Books written by Mukesh are below:");

**for**(int i=2;i&lt;=allRows.size();i++)

{

WebElement authorColumn=driver.findElement(By.xpath("//table[@name='BookTable']/tbody/tr["+i+"]/td[2]"));

**if**(authorColumn.getText().toLowerCase().equalsIgnoreCase("Mukesh"))

{

WebElement bookNameColumns=driver.findElement(By.xpath("//table[@name='BookTable']/tbody/tr["+i+"]/td[1]"));

System.out.println(bookNameColumns.getText());

}

}

System.out.println("=====================================================================");

// Another shortcut way

System.out.println("Way 2: Books written by Mukesh are below:");

List allColumnsInRow=driver.findElements(By.xpath("//table[@name='BookTable']/tbody/tr/td[text()='Mukesh']/../td[1]"));

**for**(WebElement e: allColumnsInRow)

{

System.out.println(e.getText());

}

// Print book name whose price is greater than and equal to 1000

System.out.println("========================================================================");

System.out.println("Books with price greater than and equal to 1000 are below:");

**for**(int i=2;i&lt;=allRows.size();i++) { WebElement priceColumn=driver.findElement(By.xpath("//table[@name='BookTable']/tbody/tr["+i+"]/td[4]")); **if**(Integer.parseInt(priceColumn.getText())&gt;=1000)

{

WebElement bookName=driver.findElement(By.xpath("//table[@name='BookTable']/tbody/tr["+i+"]/td[1]"));

System.out.println(bookName.getText());

}

}

System.out.println("========================================================================");

// How to print data from last row

System.out.println("Directly printing column values of last row of table: ");

List columnOfLastRow= driver.findElements(By.xpath("//table[@name='BookTable']/tbody/tr[last()]/td"));

**for**(WebElement e:columnOfLastRow)

{

System.out.println(e.getText());

}

System.out.println("========================================================================");

// find sum of cost of all books listed

List costColumns= driver.findElements(By.xpath("//table[@name='BookTable']/tbody/tr/td[4]"));

int sum\_price=0;

**for**(WebElement e:costColumns)

{

sum\_price= sum\_price+Integer.parseInt(e.getText());

}

System.out.println("total price: "+sum\_price);

System.out.println("========================================================================");

// Retrive cell value by providing row and column number

WebElement colValue= driver.findElement(By.xpath("//table[@name='BookTable']/tbody/tr[2]/td[3]"));

System.out.println("Cell Value : "+colValue.getText());

System.out.println("========================================================================");

System.out.println("Cell value using custom method: "+HandlingWebTable.getColValue(2, 3, driver));

// Printing column index based on column name

List allHeadersOfTable2= driver.findElements(By.xpath("//table[@name='BookTable']/tbody/tr[1]/th"));

**for**(int k=0;k&lt;allHeadersOfTable2.size();k++)

{

**if**(allHeadersOfTable2.get(k).getText().equalsIgnoreCase("price"))

{

System.out.println("Column index of Price column is: "+(k+1));

}

}

driver.quit();

}

public static String getColValue(int row, int col, WebDriver driver)

{

WebElement colValue= driver.findElement(By.xpath("//table[@name='BookTable']/tbody/tr["+row+"]/td["+col+"]"));

**return** colValue.getText();

}

}

<http://makeseleniumeasy.com/2017/07/14/how-to-handle-a-web-table-in-selenium-webdriver/>

1. What are locators available in selenium webdriver and which is best to use?

Locators available in selenium are :

ID,Name,CSS Selector,Classname,Tagname,Linktext,PartialLinkText,Xpath

 If we use slow locators, it will increase test execution time as well. So, we must use correct and faster locators.

ID is best to use as its unique and fast execution : ID locator is faster because at its roots, it calls “document.getElementById()”(A javaScript syntax) which is very much optimized by many browsers.

Locators like ClassName, TagName, LinkText and PartialLinkText are mostly used based on situation. ClassName and TagName can be used to locate a list of web elements while LinkText and PartialLinkText can be used for anchor tag web elements.