This assignment is to create a LoadRunner TruClient script with below provided steps and questions

**Step 1 – Visit the bol.com homepage**

Payal’s Steps:

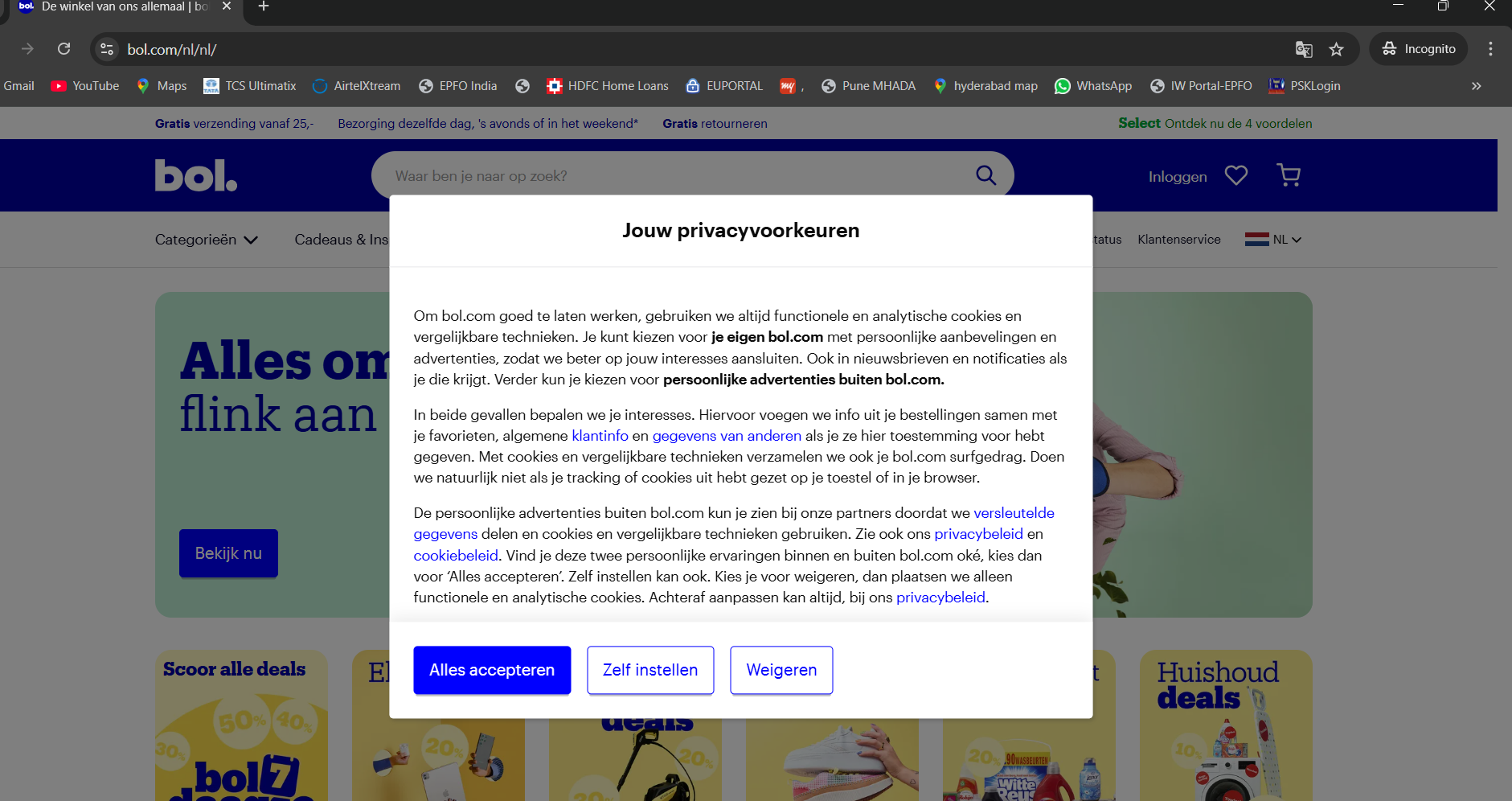
1.launch page with url “"https://www.bol.com/nl/nl/"”

2. Click “Alles accepteren**”**

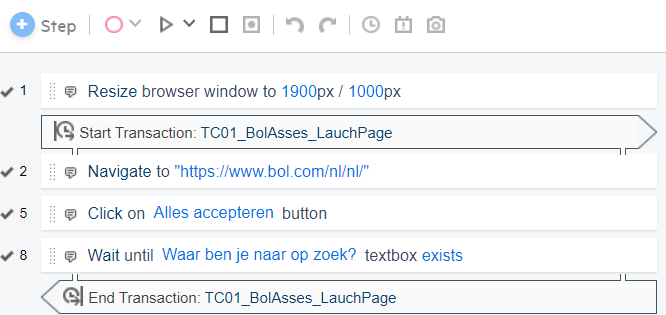
Above 2 steps together forms -🡪 (**TC01\_BolAsses\_LauchPage)**

**Question 1 – How do you determine when this page is done?**

Ans- Generally a page os considered fully loaded when all the elements,and dynamic content have finished loading. Hence in the script when navigating to bol.com homepage, I have considered document is loaded(DOM content loaded + external resources;images etc) as end event for this page.

****

Before moving on to to the next step, I have added Wait for object(“Waar ben je naar op zoek?”) and it is also part of first transaction. The verification of this step will ensure as a web\_reg\_text check for the Launch Page and also my script will be ready for the next Step, i.e. Search

****

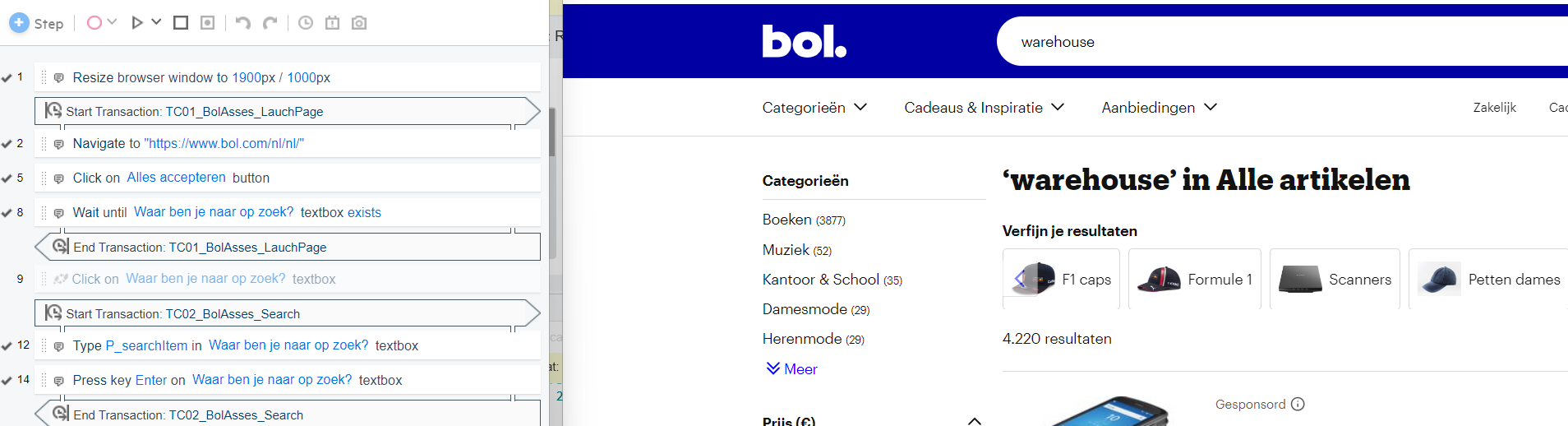
**Step 2 – Search for the following word in the search field: ‘warehouse’.**

Payal’s Steps:

1. First iteration script will search with “warehouse” and second iteration with a wrong input. I have parameterized the search word in this script.
2. After entering the search keyword, Press Enter

Combining above 2 steps are inside transaction-🡪 **(TC02\_BolAsses\_Search)**

**Search with “warehouse”**

****

**Wrong Search Word:**

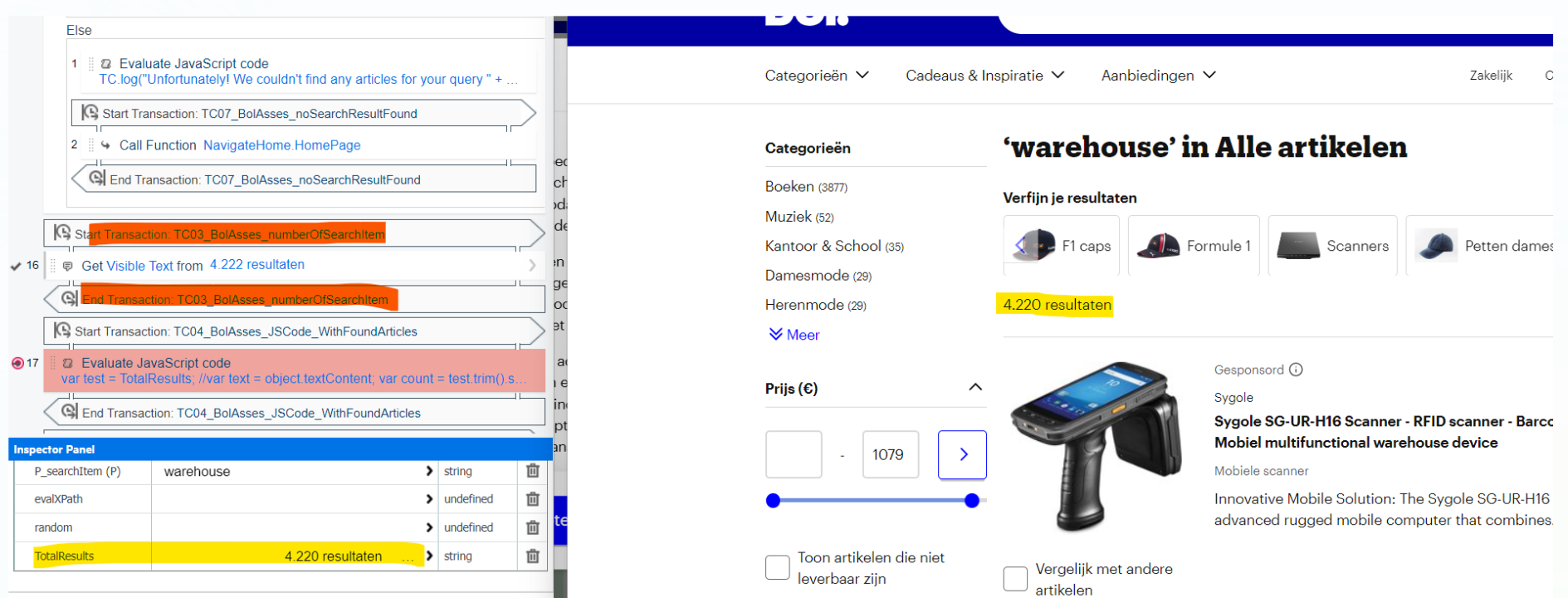
****

**Step 3 – Store the number of articles found.**

Payal’s Step:

Encapsulated this step inside (**TC03\_BolAsses\_Search**)

Identified the object using Javascript and captured the visible test in a JS variable “TotalResults” (this variable will be used in step 4)

****

**Step 4 – Turn the saved number of articles into a JavaScript variable (integer) that could be used**

**later in the script.**

Payal’s steps:

Encapsulated a Step with “Evaluate Javascript code ” inside transaction (**TC04\_BolAsses\_JSCode\_WithFoundArticles**)

Inside that added below code:

var test = TotalResults;

var count = test.trim().split(' ')[0];

var articlesCount = count.replace(".", "");

articlesCount = Number(articlesCount);

TC.log("Articles Present in page - " + articlesCount);

TC.setParam("ArticleCount",articlesCount.toString());

**Question 2 – How did you do this and why?**

Ans – Below are the steps performed to achieve this –

Stored value of article founds in JavaScript variable ‘test’.

Used JS trim function to split value of ‘test’ into array of ["4.222", " resultaten"] and selected the first value.

Remove ‘.’ and convert variable to integer.

*var test = TotalResults; // stored value to test variable from step 3*

*var count = test.trim().split(' ')[0]; //this function will search for space and split the values into different arrays and will select the first value. ([4.222] in this case)*

*var articlesCount = count.replace(".", ""); // this function will remove ‘.’ from the string.*

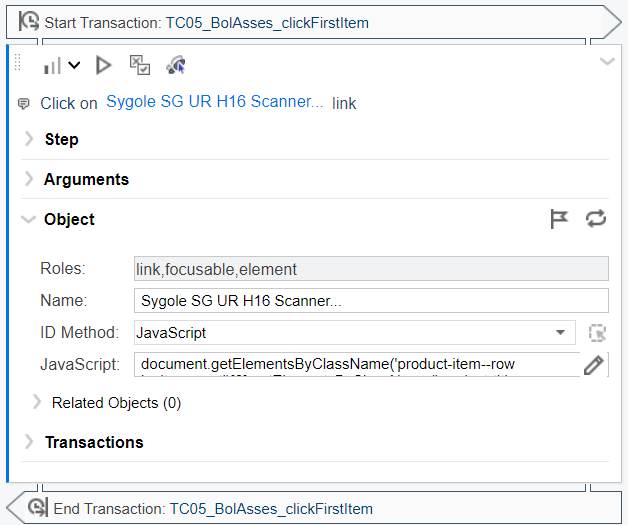
*articlesCount = Number(articlesCount); // this function will store value as an integer.*

As we have to use this number in step 8 , so it was necessary to store number without any separator. JavaScript manipulation was comparatively easy to store number into JS variable

**Step 4 – Click the top result.**

Payal’s Steps:

Added a Click Step on fist displayed item inside transaction(**TC05\_BolAsses\_clickFirstItem**)



Question 3 – What object recognition method did you use to click the object and what was your

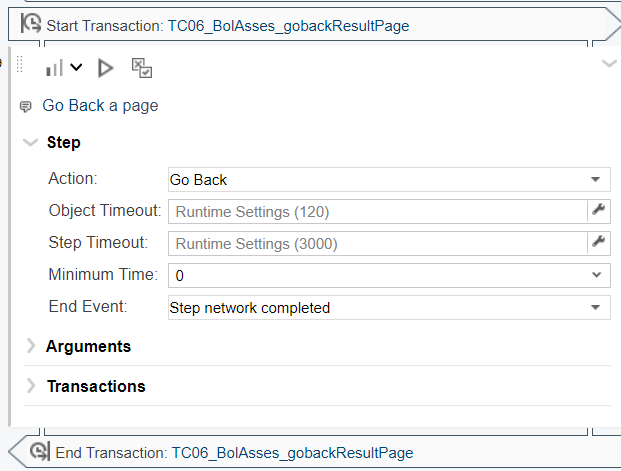
reasoning behind this?

Ans- I used JavaScript ID method to click the object. JavaScript uses DOM (getElementsbyClassName , etc..) to identify object. Reason to choose JavaScript was that, it is always easy to play around with it in browser console. Xpath object recognition has some risk as the path value may change if some object is added or deleted.

**Step 5 – Go back to the results page.**

Payal’s Steps:

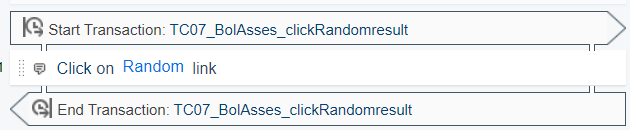
Used a “Generic Browser Action” from tools and modified it as Go Back and inserted in transaction (**TC06\_BolAsses\_gobackResultPage**)



**Step 6 – Click a random result from the first page.**

Payal’s Steps:

Used a Click Step inside transaction (**TC07\_BolAsses\_clickRandomresult**). In Object wrote a piece of javascript code for picking a random value



Question 4 – How did you decide to solve this?

Ans – I decided to use JavaScript for this. Two things came to my mind when I was solving this. 1st - how many articles are present in the page and 2nd – number of articles vary if I search something else (e.g. mobiles). And answer was YES , if I search different products , number of articles in the page will vary.

To solve this , this was my approach –

*var length = document.getElementsByClassName('product-item--row js\_item\_root').length;* //get the number of child in main class (same as number of articles in page) using JS length function.

*var index = Math.floor((Math.random() \* length ) +1)* // store a random number in index variable between 1 and total article on the page

*document.getElementsByClassName('product-item--row js\_item\_root')[index].getElementsByClassName('hit-area\_\_link medium--is-hidden')[0];* // replace index in array of main class to click and random link on the page.

**Advanced**

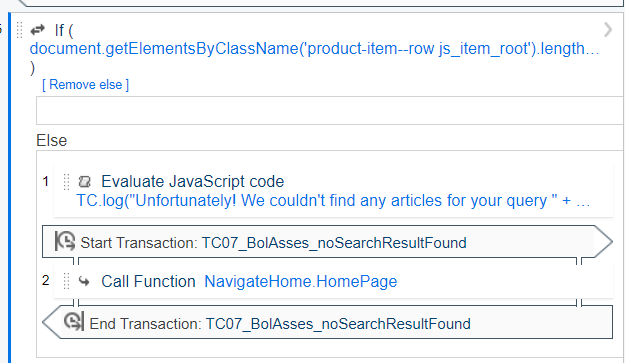
**Step 7 – Add a way to handle a situation when no article is found (you can use a weird search term**

**to test this).**

Question 5 – Why did you set it up like this?

Ans– If no article found , then go back to home page , log an error message and exit the iteration and continue other iterations if any.

I used **If Block step** to check if class ‘product-item--row js\_item\_roo’ contains anything. This class will exist in the DOM only if we are searching for valid articles. Hence **if length of this class is zero , it means we landed to ‘No result found ’ page**. I used condition if **document.getElementsByClassName('product-item--row js\_item\_root').length > 0**;, then continue the flow else use the handler and exit current iteration.

****

Reason for using event handler , was to make script more modular. Event handler is generic and can be used with other steps in the script.

Step 8 – Create a C function that uses the number of articles found (step 4) to output a random

article number. Note that normally this would make more sense to do in JavaScript, but this is just to

see how you handle the interaction between TruClient and C.

Payal’s Steps:

Thought process was to create a random number by using c function , which has range between 1 and Number of articles found for search result. To achieve this my thought process and steps were -

I wrote a normal c code to generate a random number between 1 to integer articles in Loadrunner.

After writing c function in C-Function.c , I used ‘Evaluate JS’ step to evaluate my c function.

Thought was to store value of JS variable (created in step 4 – articlesCount ) into articles.

I converted JavaScript variable into loadrunner variable using lr.setParam() function. Once number was converted to loadrunner variable , it was easy to use this in c function.