**Web Technologies PC Assessment (JavaScript, JQuery, Ajax, Java EE)**

**Date**:

**Time**: 11.15 – 13.30

**What to submit** : Your JavaScript file (main.js). Also include a word document with Figures similar to Fig. 5,8,10 and 11 depending on how much you completed.

Step 1. Find All Cars (test with browser) Completed

Step 2. Find a car by id (test with browser ) Completed

Step 3. Display list of all cars in browser (Figure 5) **[35 marks]**

Step 4 :Click on “Info” icon (Figure 8) **[30 marks]**

Step 5 Search criteria – (Figure 10, 11) **[35 marks]**

NB: You have been given a number of files. The only client side file you need to create/edit is main.js. You will also need to add some methods in your server side java code in Step 5. Do not edit any other files such as the index.html or the css file. Otherwise it may not be possible to correct your assessment.

In the JavaScript use jQuery methods for fetching the data and manipulating the DOM.

**NB Notes on marking scheme**: This exercise needs to be completed on a step by step basis. Marks cannot be allocated for code written but not working. Marks will not be allocated for code that partially implements a Step or code that does not work.

**Step 1:Find all cars and display in browser**

Already completed completed

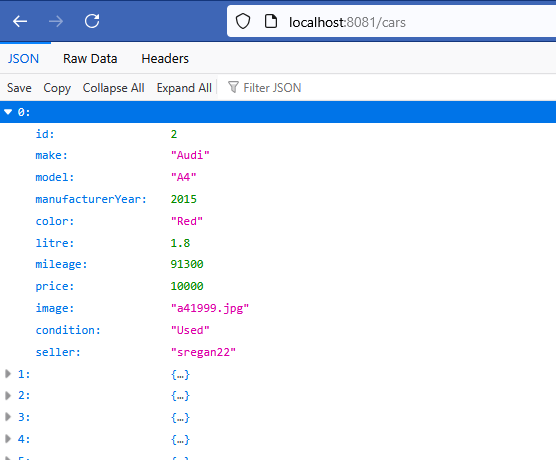


Figure 1: Browser – get all cars

**Step 2:Find one car by its id and display result in browser**

Already completed.

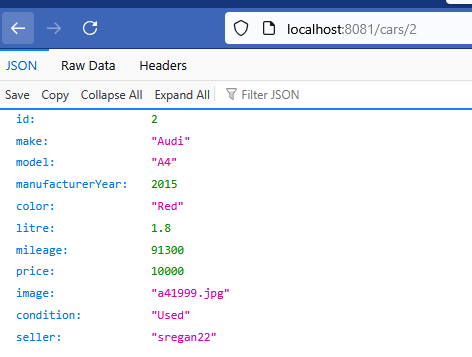


Figure 2: Output from browser – get car by id

**Step 3:Display a list of all cars in the browser**

You have been given a html file and a css file ( index.html and style.css). Do **not** make any updates to these files. You have also been given a folder of images and some library css, fonts and js.

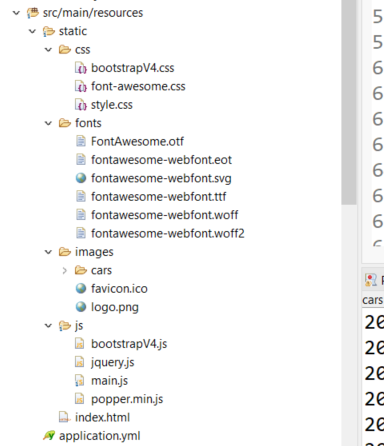


Figure 3: Files

Write the code needed in main.js to get the functionality described below. This is what the page looks like initially before you add any javascript code to main.js.

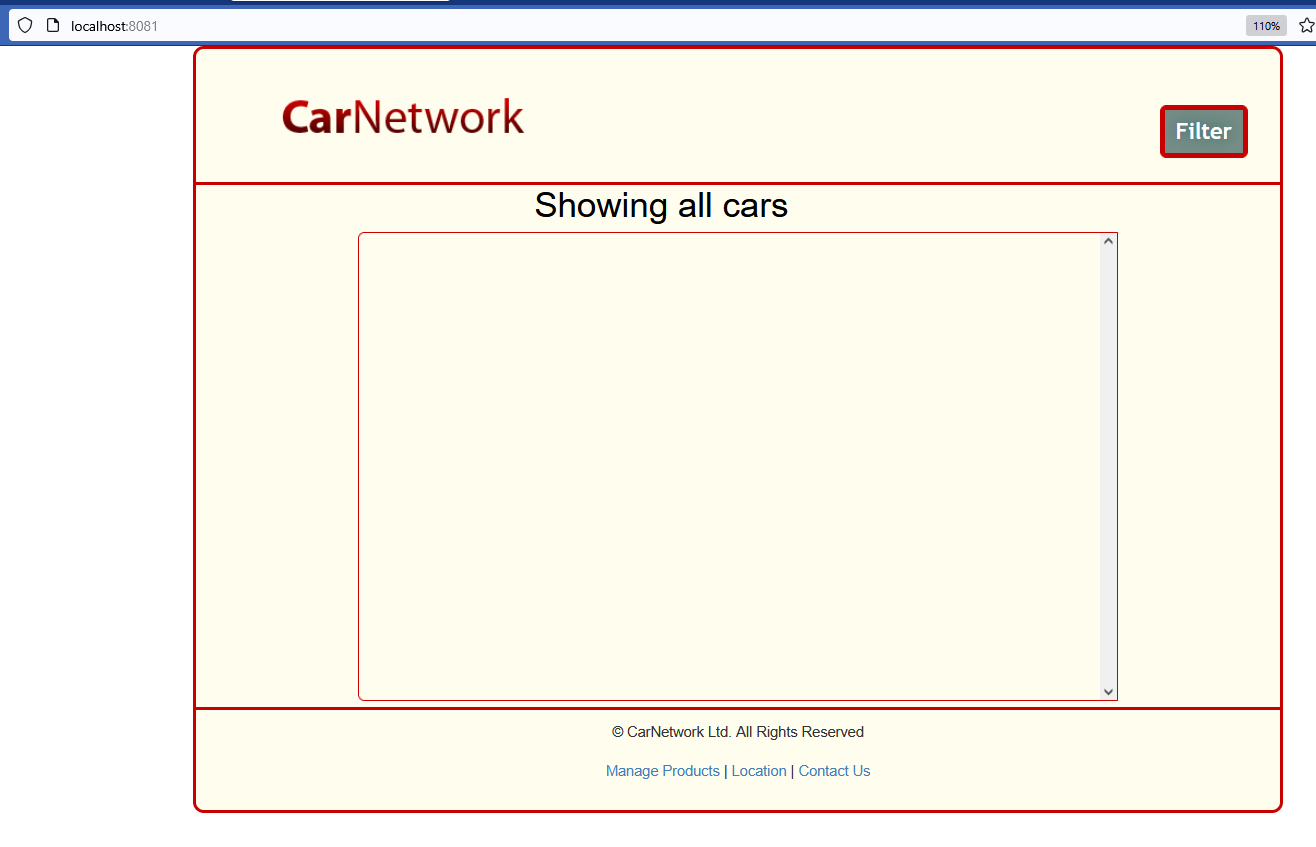
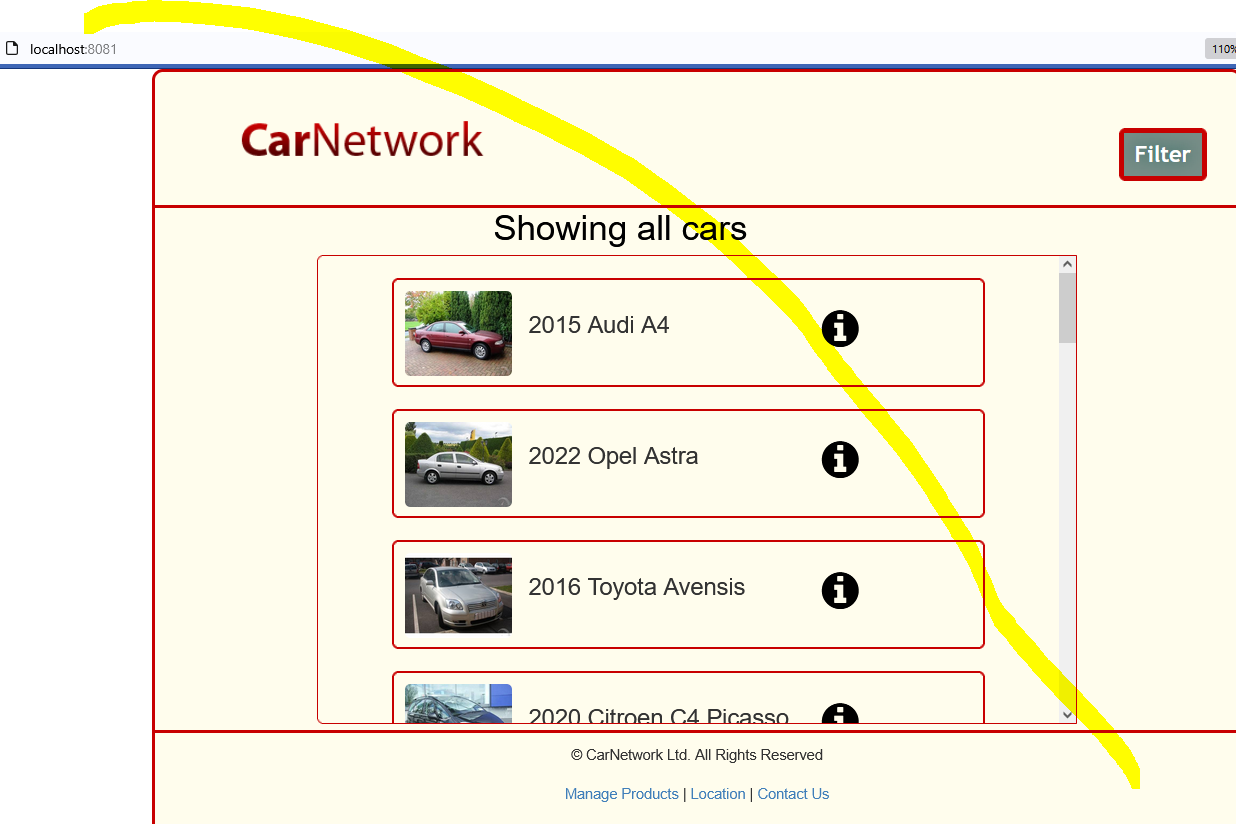


Figure 4: Browser output with no javascript in main.js

Add the javascript so that the cars are added to the <div class=”list\_scroll\_cars”> that is contained in index.html. This is what the page should look like when the list of cars is displayed.

A screenshot of the generated markup is shown on the next page. This is the markup that you need to generate using JavaScript. NB Generate this markup up with JavaScript – do not write it in HTML.



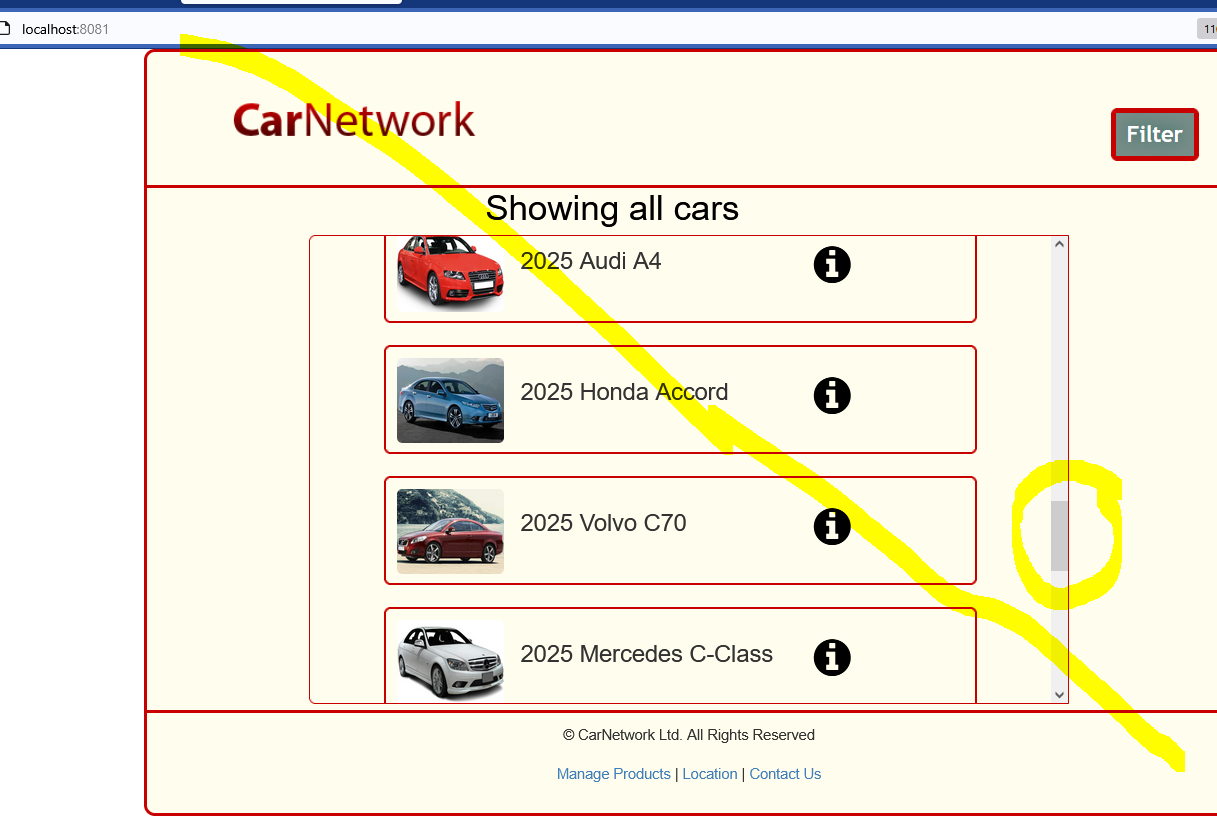


Figure 5: Browser output showing list of cars



Figure 6: Example of generated markup – from inspect element

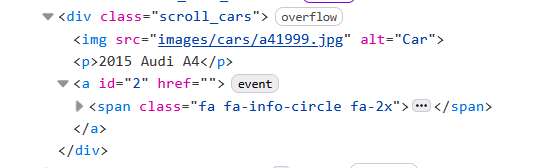


Figure 7: Example of generated markup for each div – from inspect element

Hint: If you don’t use the <div class=”scroll\_cars”> as a wrapper styling will not be applied correctly. All Cars should be displayed.

Note: There is no content in the <span> element.

**Step 4:Click on the “Info” icon**

When the “Info” icon  is clicked for a particular car, more details for that car are displayed in a model. Pay attention to the modal title and the Engine and Mileage fields. Use the modal with the id “detailsModal” that is provided in the html.

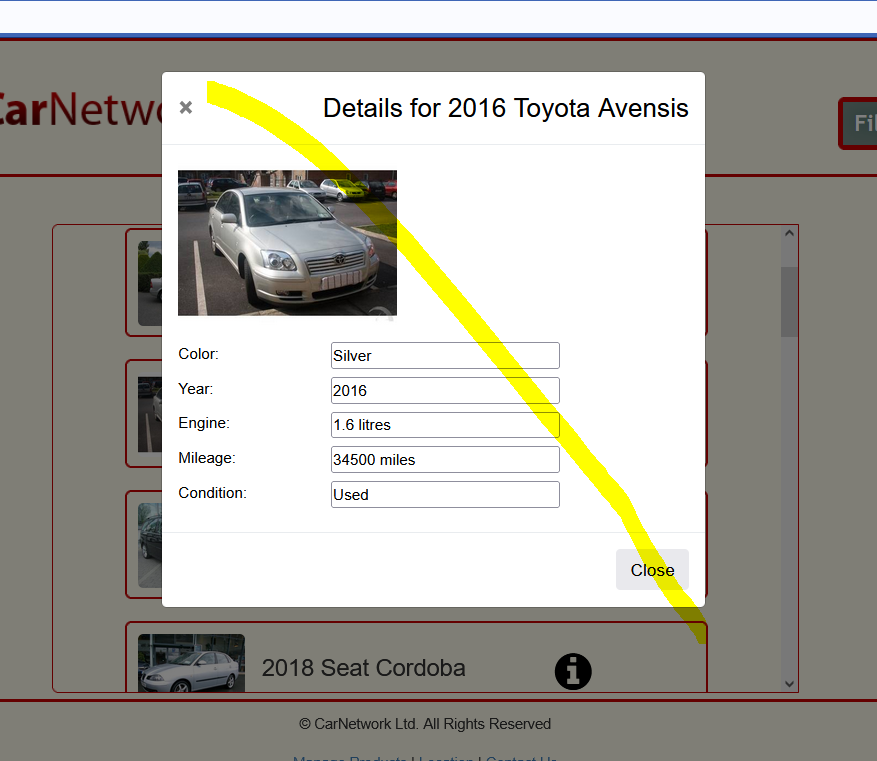


Figure 8: After clicking on “Info” icon

**Step 5: Adding search functionality on the server**

Add a filter option based on minYear and maxYear on the server. You need to update CarController and CarRepository

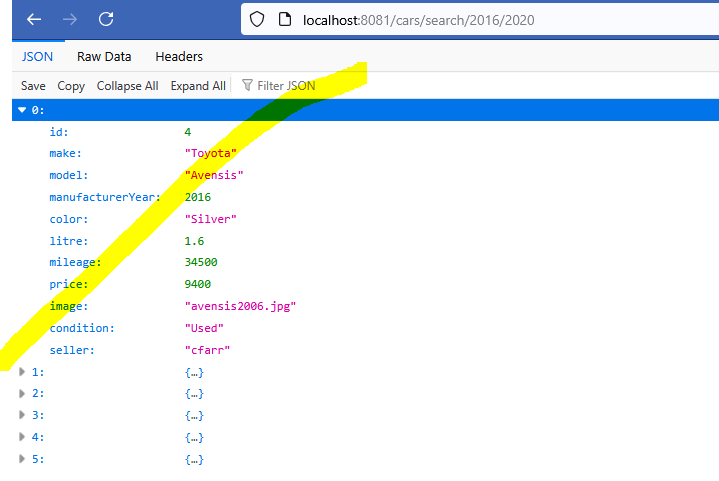
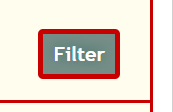


Figure 9: Search criteria – shows from 2016 to 2020

**Step 6: Filtering based on Search Criteria**

When the “Filter” button  is clicked, the modal with the id “filterModal” is displayed.

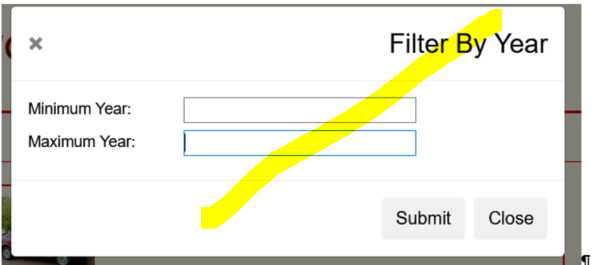


Figure 10: Filter Modal displayed

When the data is returned it is displayed as shown in Figure 12. Note: the heading, h2 with id” whichCars” in the html.

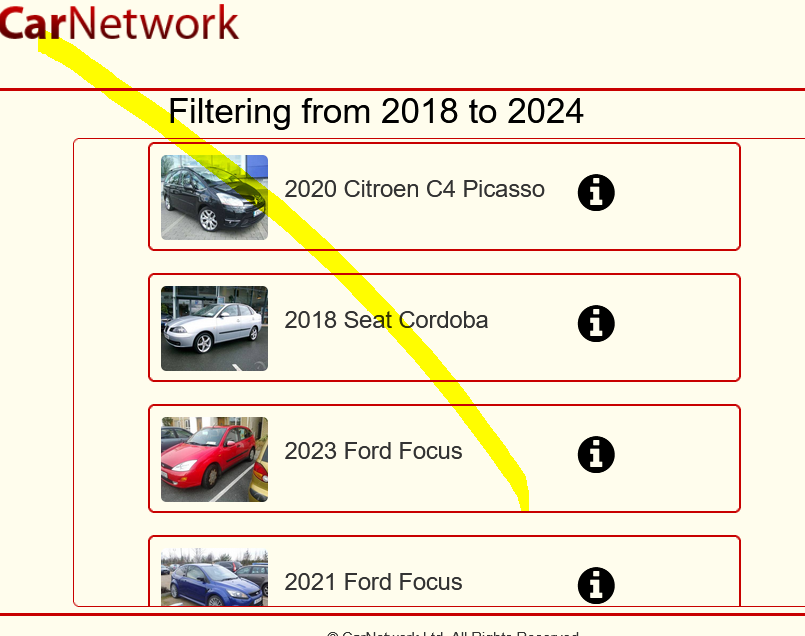


Figure 11 Search criteria