Interview Practical Assessment

Introduction

This is a practical assessment of your ability to code C#, T-SQL, CSS and ASP.NET MVC. You have one hour to complete as much of the assessment as you can. If you are struggling on a particular section, try to move on to the next one.

The following items are on the computer provided

1. Visual Studio with the ASP.NET MVC Interview Test solution open
2. SQL Server with an InterviewTest database
3. SQL Server Management Studio for database development
4. Design Files & Assets for the finished product

Complete as much of the assessment as you can in the time allowed. At the end of the test the code will be reviewed with a senior developer, giving you the opportunity to explain any decisions made or how you might choose a different approach if this were a larger public facing system.

Each section can be done independently, however you may wish to use the output of the previous step in subsequent tasks.

Database

Locate the database in SQL Server Management Studio. In it are 2 tables based on car manufacturers & ranges. These tables already have their primary keys created for you and contain some sample data.

- Create a suitable relationship that links manufacturers with ranges.
- Create a view that displays the following data for each range
  - ManufacturerId
  - ManufacturerName
  - RangeId
  - RangeName
- Create a stored procedure that will return data about all ranges for a given manufacturer id.
- Given the following query, add an appropriate index to the manufacturer table

```
SELECT *
FROM Manufacturer
WHERE ManufacturerName = 'Ford'
```

C# ASP.NET MVC

The provided Visual Studio solution contains an ASP.NET MVC project.

- The List action of the RangeController should return a view for each Manufacturer that can present the following data
    - The name of the manufacturer
    - A list of all the ranges produced by that manufacturer
- The data should come from the SQL Server database provided
- The connection string for the database is set in the Web.Config
- You will need to produce code to fetch the appropriate data from the database and return it to the view.
- Unit tests should be written as appropriate
- Requests in the format "/[manufacturer]" should execute the List action of the RangeController. E.g. /ford will display a page about the manufacturer Ford.

Front End

- Implement the mobile and desktop versions of the page to match the provided designs
- Images for each vehicle are provided in the Content/Images/Vehicles folder of the solution. The file name of each image is stored in the Range table in the database
- Manufacturer logos are provided in the Content/Images/Manufacturers folder of the solution
- A sprite of the manufacturer logos is provided in the Content/Images/Sprites folder
- On mobile (devices less than 480px)
    - The manufacturer logo should not be displayed
    - Each vehicle image should be collapsed by default. When the user taps a range name the image should then get displayed (as illustrated by the BMW 2 Series in the design). When they tap the range name again, the image should be hidden