**Javascript**

The Fibonacci sequence is a series of numbers ,where a number is the addition of the last two numbers, starting with 0, and 1. e.g. **0,1,1,2,5,8,13,21,34,55**

In the “index.html” file provided in the javascript folder there is a input number field and a run button. Write a script which allows the user to enter the position of the Fibonacci sequence and return the value in the sequence at that position. i.e. if a user enters “**4**” the output should be “**2**” if a user enters “**11**” the value should be “**89**”. This should be dynamic.

**SQL:** (use **sql-script** file to create tables and entries)

Write queries to display the following fields.

1. firstName, lastName and departmentName of all employees. (department name column must be displayed as “departmentName”
2. all departments name,number of employees in each department, average salary in that department and order by average salary
3. firstName, lastName, departmentName, salary, salary + 10% increase order by salary increase.
4. departmentName, highest salary in that department
5. A millennial is anyone born between 1981 – 1996. Considering this display, the following firstName, lastName, isMillennial (yes or no)

**ASP MVC C#**

Complete the following in a Asp.net Mvc project.

1. Create a screen where each of the queries in the SQL section can be called and display them in an html table
2. Add a page which allows the user to enter a new employee with all fields including department, this should be inserted into the employee’s table. Please note that all fields are mandatory, add validations as necessary.
3. Create an export for each of the reports which are displayed in step 1 of this section. Export to .csv.
4. Create an upload function to upload employees into the system. Upload from a .csv file  
   This file should cater for first name, last name, date of birth, salary and department fields. The upload should also provide the user with feedback on whether the upload was successful or not as well as provide error messages if any.

Please create relevant models where necessary. Any logic dealing with the database should not be handled within any controllers but rather within a service/manager class.