Summary

You are required to put together an ASP.NET web application (ASP.NET MVC) utilising Visual Studio 2019 or below and Microsoft .NET Framework v4.6 or above (or .NET Core). The application can be based on Visual Studio 2019 or above version default ASP.NET Web Application template. You are also required to create a database with appropriate tables and fields.

Scoring

This test generally takes between 4 and 7 hours to complete. You are encouraged to cover every requirement as the more you cover the more you are likely to score. However, if you are not familiar with every requirement or you feel you may not be able to finish it all in time, rather than covering every requirement we would encourage you to focus on efficiency and best practices.

For high score it is critical that your application uses the following standards and best practices;

1. N-Tier Architecture or whatever prefer
2. Object Oriented Programming
3. Utilisation of latest ADO.NET technologies OR Entity Framework OR other.
4. ASP.NET Integrated Security
5. Coding Best Practices
6. User Input Validation and display meaningful message to end user
7. Exception Handling and display meaningful message to end user

Preferable Technologies and Platforms

In order to complete your test, it is recommended that you use following technologies, tools and platforms.

- Microsoft Visual Studio 2019
- .NET Core or ASP.NET MVC (do not use Windows Forms, ASP.NET Classic)
- .NET Framework v4.6 or above or .NET Core
- Visual C#
- REST API (Optional)
- Ajax
- JavaScript / jQuery/Angular/React
- LINQ to SQL / LINQ to Entity Framework
- Microsoft SQL Server 2014 or above
- Microsoft SQL Management Studio 2014 or above

High Level Scope

- Use Visual Studio 2019 and create an ASP.NET Web Application (.NET MVC) targeting .NET Framework 4.6 or above
- Create a Registration Page with the following fields
  - Full Name
  - Email
  - Password
  - Confirm Password
- Create a Login Page with the following fields
  - Email
  - Password
- This should be a login system.
- Create a SQL database with necessary tables, fields and stored procedures (if necessary). If you use database first, you must generate script of tables or stored procedure or function.

Detail Requirements

1. Database & Data Access Layer

Functional Requirements

- Use SQL Management Studio and connect to local database server
- Create a database with appropriate tables and fields
- Create a stored procedure to obtain user list (if not using code first model)
- Use ADO.NET/EF technology such as LINQ to SQL or LINQ to Entity Framework for database communication and CRUD operation. If you are not familiar with these technologies then use any other ADO.NET/EF technology.
- Store database connection string with Integrated Security as per standard recommendation.
- For the font-end application, can use MVC/Angular/React/JavaScript
- For the back-end, can use REST API to fetch data or any other which you prefer.

2. Registration Page

Data Elements

Full Name

- The full name may contain both first names and last name.

Email

- You need a minimum validation check to see if email address meets the appropriate Regex
- Must be unique in the database

Password

- Minimum 6 characters in length
- Use one way encryption using built-in .NET cryptography before storing in database

Confirm Password

- Must match with Password

Functional Requirements

- A save button must be placed at the bottom of the form to accept registration information and save in SQL database
- Appropriate validation message should be shown to end user
- Using Ajax technology to call WEB API, you may choose to validate the email address against the database to warn the user if the email address already exists
- On successful registration, show message "Your registration is successful"

3. Login Page

Data Element

Email
User email address

Password
Password of the user

Functional Requirements

- A Sign in button must be placed at the bottom of the form so that user can sign in to the application.
- Appropriate validation message should be shown to the end user.
- If email and password match then authenticate user redirect to Registered Users page.
- Capture user's login date time and save in database. You must keep a log of previous login date time.

4. Registered Users Page (List Page)

Data Elements
Use SQL Stored Procedure to produce a list of registered users.

Surname
Use stored procedure to split full name and make last word as surname

Given Names
Use remaining words as given names

Email
Email of the user

Last Login Date & Time
Last login date and time, show "NEVER" if doesn't exist

Functional Requirements

Refresh Last Login Date Time

- A refresh link or icon should be placed at the end of each data row

Surname	Given Names	Email	Last Login Date Time	
JOHN	Smith D	john@johns.com.au	25/12/2012 10:00	Refresh
SMITH	Tom	smith@smith.com.au	26/12/2012 16:00	Refresh
PERRY	Matt	perry@web.com.au	27/12/2012 9:00	Refresh
DICK	Tom	dick@tom.com.au	28/12/2012 11:00	Refresh

- When user clicks on refresh link, you can either use Ajax or Ajax Callback to obtain Login Date Time of the individual user.
- Use jQuery or JavaScript to replace appropriate cells of the data row with obtained Login Date Time.

Sign out

- A Sign out link must be provided at the top right corner of the page along with Full Name of the signed in user. For example: **John Smith** [Sign out]
- When user clicks on Sign out, use Forms Authentication to sign out the user and redirect to login page.

5. Call Centre Record Page

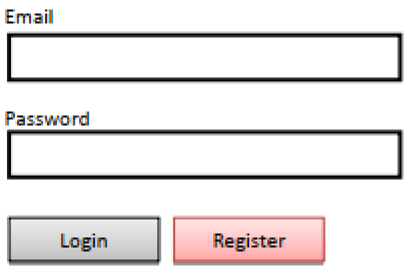
After authenticating the application should show menu 'Call Centre Record'.

Description of the page are below:

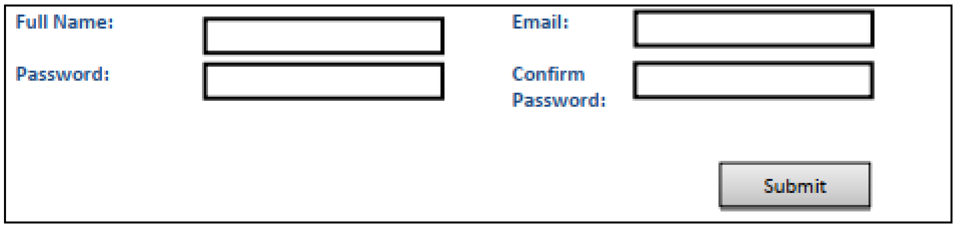
The XYZ company, they want to develop simple 'Call Centre' based application. The people will call to know about something. By the application, the system will keep record of caller information like first name, last name, phone etc. The system also needs to keep call duration of each callee how long he/she talked and based on the call duration, the system will automatically calculate charge of this call. The charge will be different based on time slots 06:01 – 12:00, charge will apply 5.00TK/Minutes; 12:01 – 18:00 charge will apply 5.50TK/Minutes; 18:01 – 24:00 charge will apply 6.00TK/Minutes; 12:01 – 06:00 charge will apply 7:00TK/Minutes. Eventually higher authority of XYZ company want to see various types of report like Monthly report, Weekly Report etc. Each report will contain meaningful information which will help to take managerial decision of XYZ management.

User Interface Mock-up

1. Login Page



2. Registration Page



3. User List Page

John Smith [Logout]

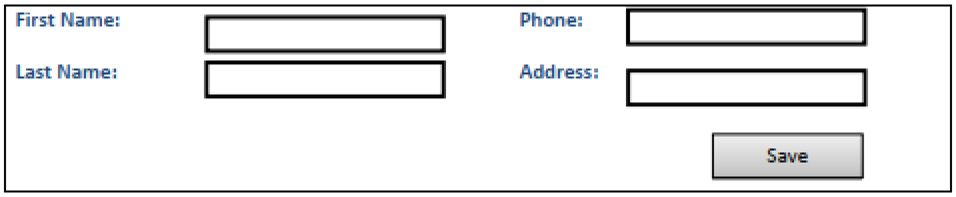
Surname	Given Names	Email	Last Login Date Time	
JOHN	Smith D	john@johns.com.au	25/12/2012 10:00	Refresh
SMITH	Tom	smith@smith.com.au	26/12/2012 16:00	Refresh
PERRY	Matt	perry@web.com.au	27/12/2012 9:00	Refresh
DICK	Tom	dick@tom.com.au	28/12/2012 11:00	Refresh

4. Callee List Page

Add New

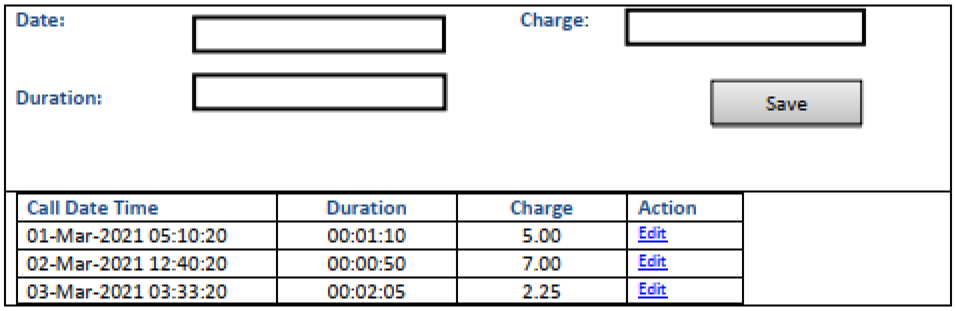
First Name	Last Name	Phone	Last Call Date Time	Total Duration	Total Charge	Action
JOHN	SMITH	01711010101	01-Mar-2021 05:10:20	00:01:10	5.00	Edit | Delete | Duration
PERRY	MATT	01911415454	02-Mar-2021 12:40:20	00:00:50	7.00	Edit | Delete | Duration
DICK	TOM	17112525252	03-Mar-2021 03:33:20	00:02:05	2.25	Edit | Delete | Duration

5. Callee Details Page



6. Duration Page (Callee)

Add New



Call Date Time	Duration	Charge	Action
01-Mar-2021 05:10:20	00:01:10	5.00	Edit
02-Mar-2021 12:40:20	00:00:50	7.00	Edit
03-Mar-2021 03:33:20	00:02:05	2.25	Edit

Deployment

- Use Visual Studio's built-in feature to publish your application in local IIS Server