

SHEREÉ GREEFF

31083625

19 NOVEMBER 2020

CMPG 323

PROJECT 2

Contents

[*Introduction & Overview* 3](#_Toc56680993)

[*ERD* 4](#_Toc56680994)

[*Use cases* 6](#_Toc56680995)

[*Data flow diagram* 8](#_Toc56680996)

[*How to guide* 10](#_Toc56680997)

[Register 10](#_Toc56680998)

[Login 11](#_Toc56680999)

[Create 12](#_Toc56681000)

[Edit 13](#_Toc56681001)

[Details 14](#_Toc56681002)

[Delete 15](#_Toc56681003)

**Table of Figures**

[*Figure 1: Register process 10*](#_Toc56602652)

[*Figure 2: Login process 11*](#_Toc56602653)

[*Figure 3: Home page 11*](#_Toc56602654)

[*Figure 4: Visual Studio Startup class 12*](#_Toc56602655)

[*Figure 5: Create process 12*](#_Toc56602656)

[*Figure 6: Create View 12*](#_Toc56602657)

[*Figure 7: Edit process 13*](#_Toc56602658)

[*Figure 8: Edit View 13*](#_Toc56602659)

[*Figure 9: View Details process 14*](#_Toc56602660)

[*Figure 10: Details View 14*](#_Toc56602661)

[*Figure 11: Delete process 15*](#_Toc56602662)

[*Figure 12: Delete View 15*](#_Toc56602663)

# *Introduction & Overview*

ASP.NET Core is a framework for building web applications and web APIs using the Model-View-Controller design pattern. User requests are routed to a Controller, which works with the Model to perform actions or retrieve queries. The controller chooses the View to display to the user and provides it with Model data. The Model represents the state of the application and operations that should be performed by it. Views present the content through the user interface. The Controller is responsible for handling user interaction, working with the model and selecting a view to render.

ASP.NET Core MVC Framework is a lightweight, open source, highly testable presentation framework optimized for the use of ASP.NET Core. It provides a pattern-based way to build dynamic websites that enables a clean separation of concerns. It gives you full control over mark-up, supports friendly development and uses the latest web standards.

ASP.NET Core Web API is designed for RESTful (Representational State Transfer) services, which means it provides interoperability between computer systems on the internet.

ASP.NET Core Web Application is designed to develop web applications that return views and data.

I chose the Web Application route because I am more familiar with working in Visual Studio, comfortable with the coding as well as the environment.

I made use of the following software:

* Visual Studio 2019 (ASP.NET MVC Core 3.0)
* Microsoft SQL Server Management Studio

I will be providing an ERD to show all the tables with attributes, primary keys and relationships between the tables. An authentication, login, CRUD and register uses cases will follow. Data flow diagrams will show how data flows through the system and between the MVC pattern that is used for my web application. Finally, I will be providing a How-to-Guide, to show the steps that must be followed for a registration process, login process and completing CRUD operations.

# *ERD*

Entity relationship diagrams I planned for my project:

1…\*

|  |
| --- |
| RATE |
| RATE\_ID |
| HOURLY\_RATE |
| MONTHLY\_RATE |
| OVERTIME |
| INCOME |

|  |
| --- |
| DEPARTMENT |
| DEPARTMENT\_ID(PK) |
| DEPART\_DESCRIPTION |

1 1

1 1

1 1

1 1

1

|  |
| --- |
| **EMPLOYEE** |
| EMP\_ID (PK) |
| NAME |
| AGE |
| IDENTIFICATION\_NR |
| ADDRESS |
| CONTACT\_NR |
| EMP\_NUMBER |
| RATE\_ID (FK) |
| HISTORY\_ID (FK) |
| EDUCATION\_ID (FK) |
| JOB\_ID(FK) |
| EMP\_DETAILS(FK) |

|  |
| --- |
| **EMP\_DETAIL** |
| GENDER\_ID(PK)(FK) |
| MARITAL\_STATUS\_ID(PK)(FK) |
| DEPARTMENT\_ID (PK)(FK) |

|  |
| --- |
| **GENDER** |
| GENDER\_ID(PK) |
| GENDER\_CODE |

1

1

|  |
| --- |
| **EMP\_HISTORY** |
| HISTORY\_ID(PK) |
| YEARS\_WORKED |
| YEARS\_CURRENT\_ROLE |
| LAST\_PROMOTION |
| YEAR\_LAST\_TRAINING |

|  |
| --- |
| **MARITUAL\_STATUS** |
| MARITAL\_STATUS\_ID(PK) |
| MARITAL\_STATUS\_CODE |

1…\*

|  |
| --- |
| **EMP\_EDUCATION** |
| EDUCATION\_ID(PK) |
| FIELD |

1..\*

|  |
| --- |
| **JOB** |
| JOB\_ID(PK) |
| JOB\_ROLE |
| JOB\_LEVEL |
| JOB\_INVOLVEMENT |
| JOB\_SATISFACTION |

ER for authentication:

|  |
| --- |
| **ASP\_ROLE** |
|  |
|  |
|  |

|  |
| --- |
| **ASP\_ROLE\_CLAIM** |
|  |
|  |

|  |
| --- |
| **ASP\_USER** |
|  |
|  |

|  |
| --- |
| **ASP\_USER\_CLAIM** |
|  |
|  |

|  |
| --- |
| **EMPLOYEE** |
|  |
|  |

|  |
| --- |
| **ASP\_USER\_ROLE** |
|  |
|  |

|  |
| --- |
| **ASP\_USER\_TOKEN** |
|  |
|  |

|  |
| --- |
| **ASP\_USER\_LOGIN** |
|  |
|  |

# *Use cases*

Authentication use case:

User sends credentials to service’s login page

Authentication

Server

Returns access token to user

.NET Core

Console

Client

User browses to web application

WEB

Application

Web app takes user to authentication service

Web app returns response to user

User provides access token

Login use case:

User

CRUD use case:

Web Application - Admin

User - Administrator

Register Use Case:

Login/register

<<Includes>>

User

# *Data flow diagram*

MVC Data flow diagram:

Database

Repository services

Infrastructure

-Logging

-Security

Business rules

Controller

Model

View

System Data flow diagram:

Login Master

Yes

Yes

No

Verification Data

Data and Control flow diagram for request:

Controller

Set

Database

Model

Get

View

Data flow

Control flow

* Controller:
* Event Handling
* Update Application Data
* Definition of Control Flow
* Model:
* Definition of Application Data
* Connection to Business Functionality
* View:
* Visualization of Data

# *How to guide*

Guide that must be followed to do any of the following processes: login, register, create, update/edit, view or delete.

## Register

The register page contains validation checks. *Name* and *last name* have a text validation, both can have a maximum string length of 15 characters and a minimum of 3. *Age* needs to be an integer value, ranging between 18 and 55. *Number* are only allowed to have 10 digits, allowing values between 0 and 9. A valid *email address* needs to be entered or else an error message will be displayed. For a valid *password*, the user must choose at least 6 characters, and must contain one nonalphabetic value. *Confirm password* will confirm that the two password properties are identical. All properties are required and once entered, the register button can be clicked.

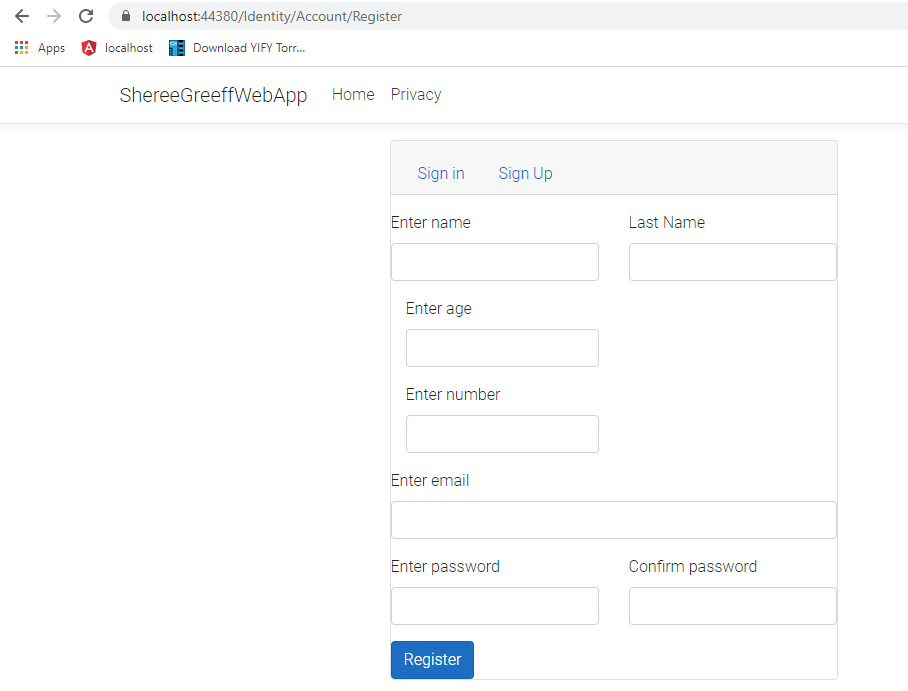


Figure : Register process

## Login

Once a user has registered, the user can login with the email and password that they registered with.

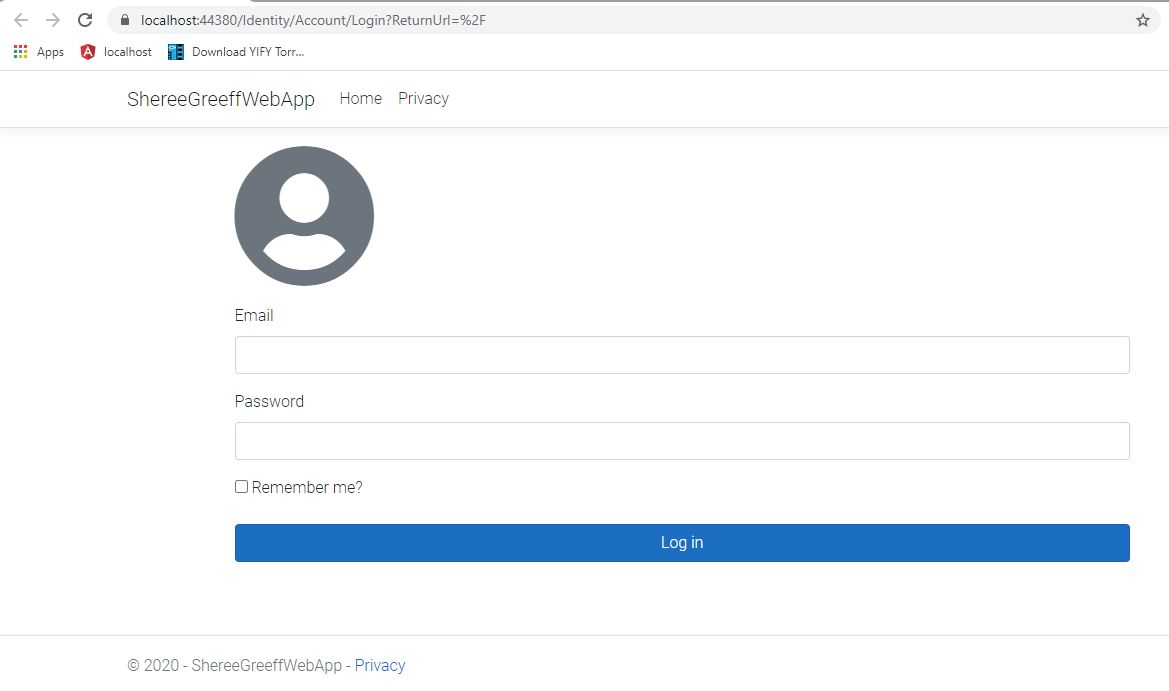


Figure : Login process

A successful login will redirect the user to the home page, and the user will be given the option to log out.

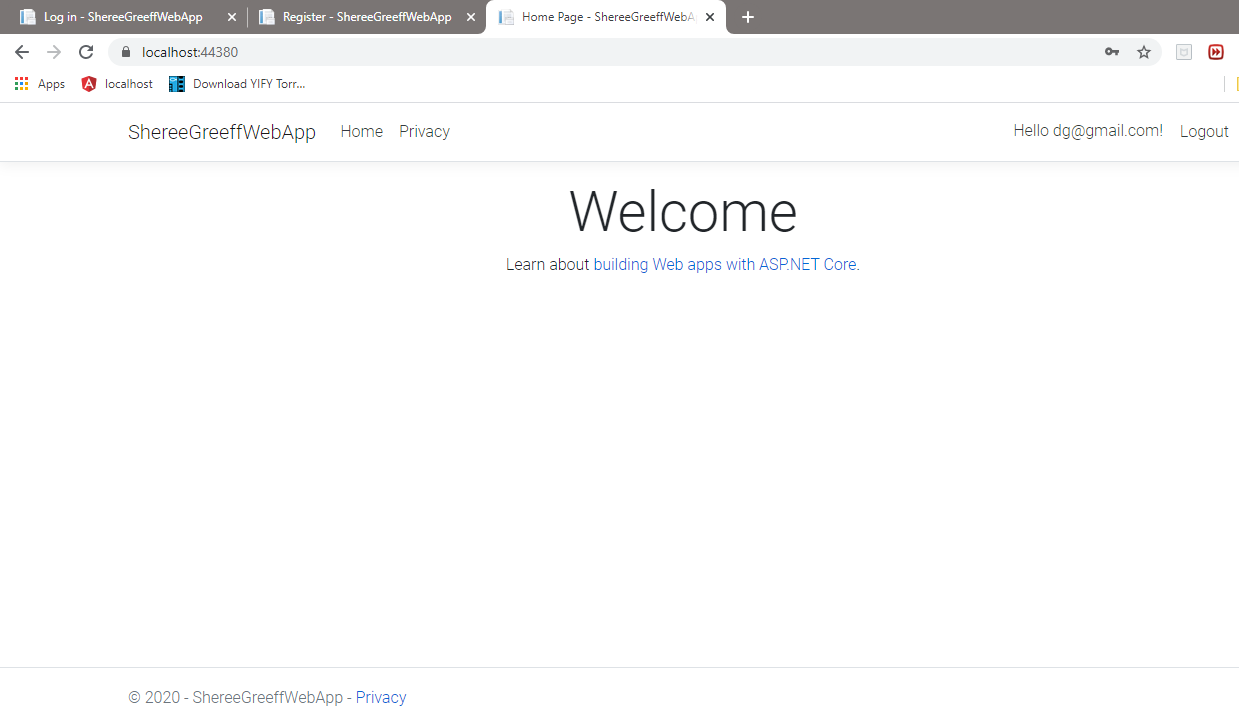


Figure : Home page

The following CRUD operations requires the controller to be changed to ‘Emp’

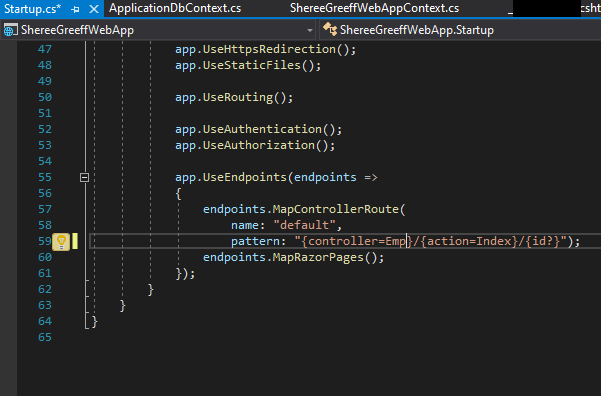
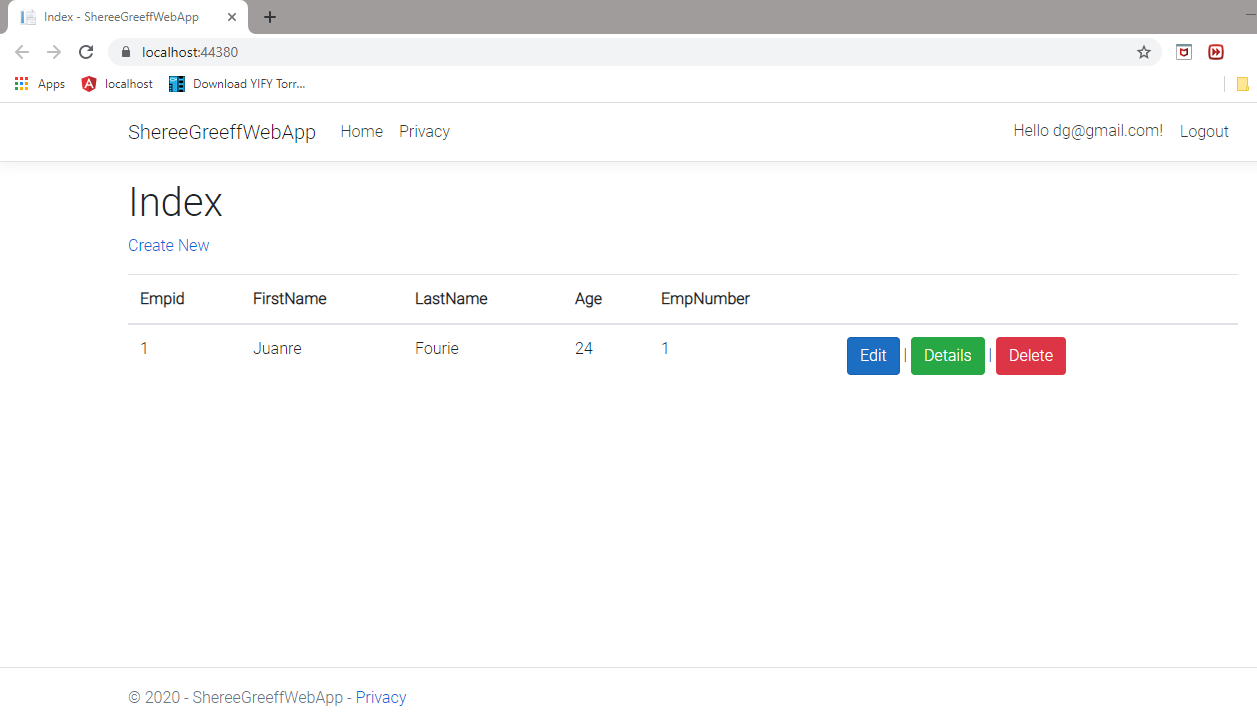


Figure : Visual Studio Startup class

## Create

The user will be directed to the Index page, where a ‘Create New’ option is displayed to create/ add a new employee.



Click here

Figure : Create process

The create view will display, where input is required. Click the ‘create’ button to add the new employee or the ‘back to list’ option if user wishes to go back to the index view.

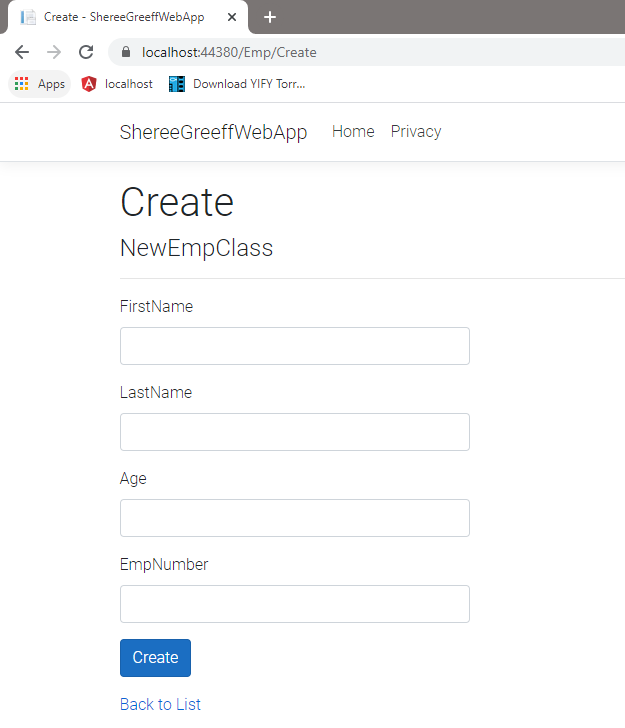
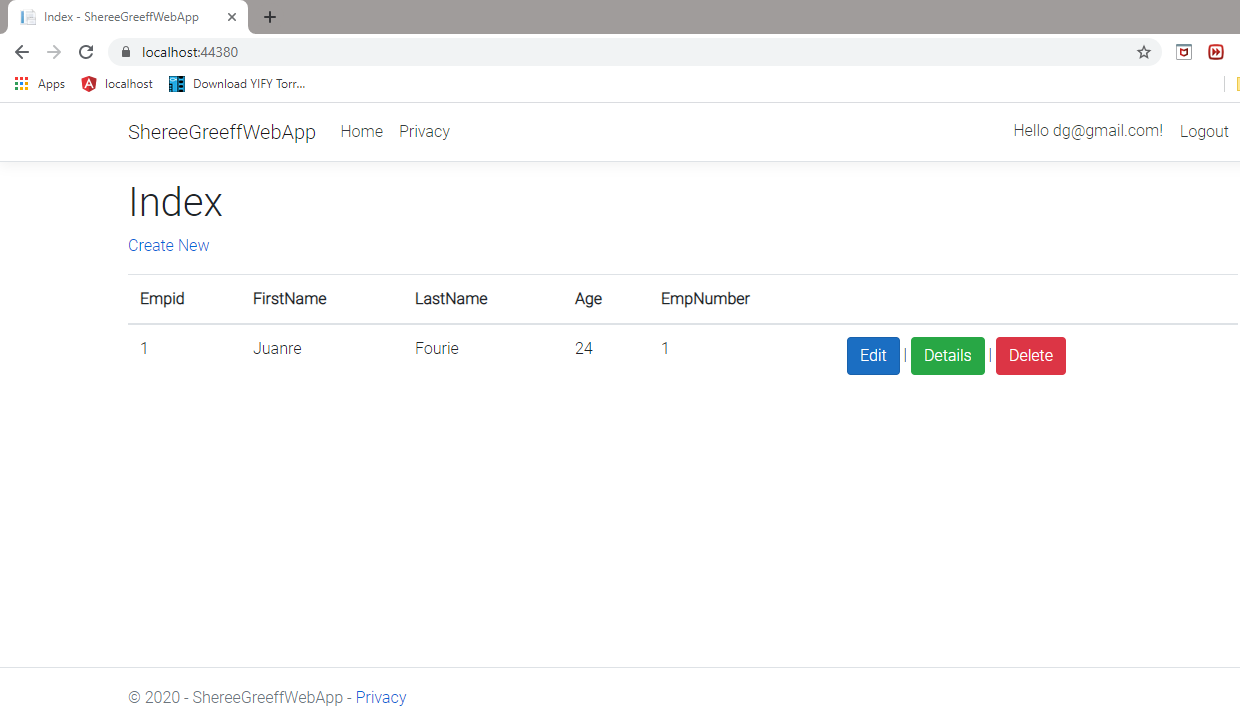


Figure : Create View

## Edit

Choose the edit option.



Click here

Figure : Edit process

The details of the employee will be displayed and the user can choose which information needs to be edited. Click ‘save’ to update the information.

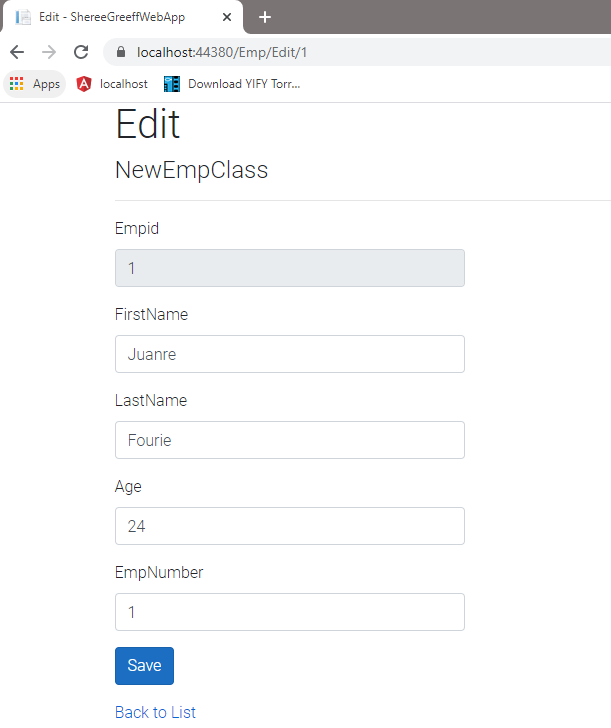
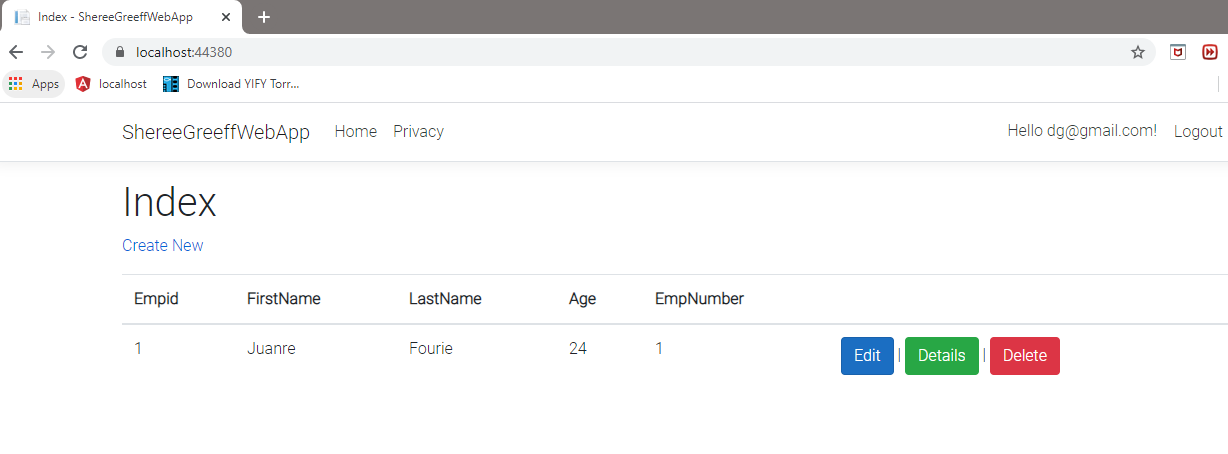


Figure : Edit View

## Details

To view the details of an employee the ‘details’ button can be clicked.



Click here

Figure : View Details process

The user is provided with the details view, and given the option to edit these details or go back to the index view.

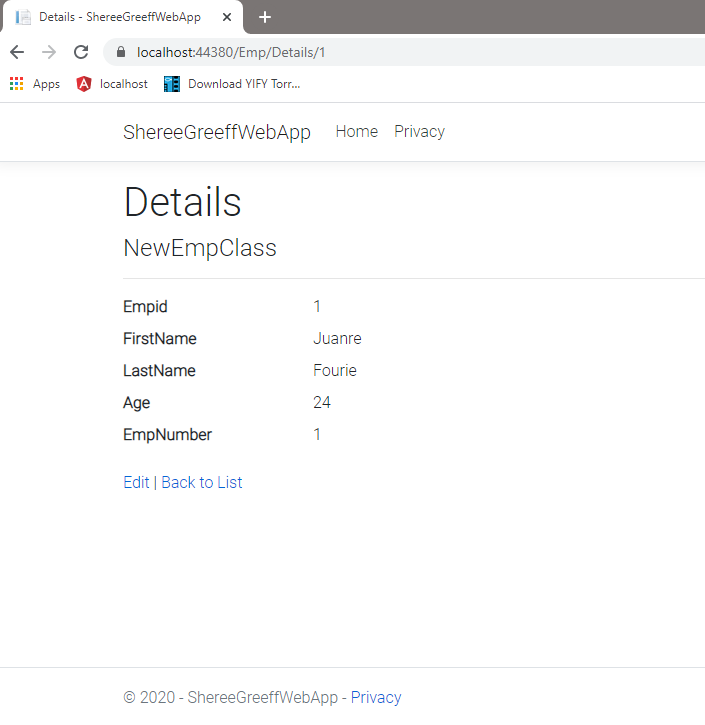


Figure : Details View

## Delete

To remove an employee, the ‘delete’ button will complete this action.

Click here

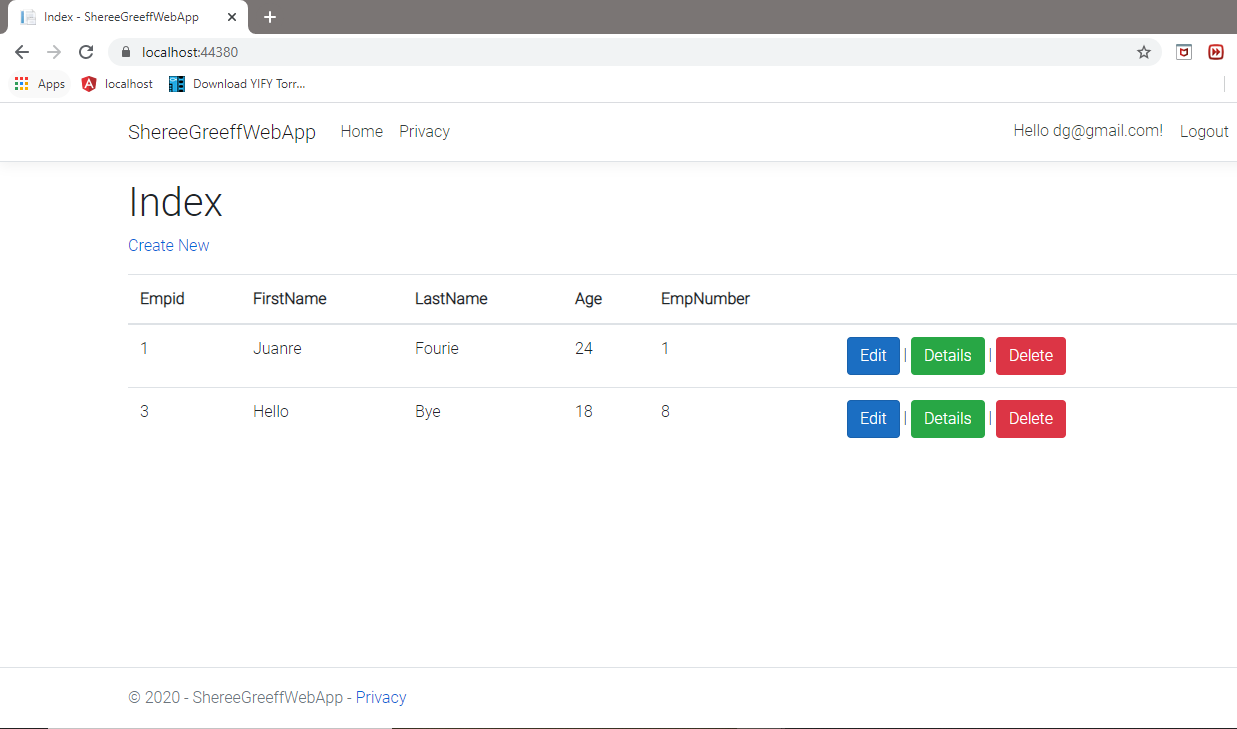


Figure : Delete process

The user will be directed to the delete view, and given the option to reconsider before completing the delete action. The user can click the delete button if they wish to proceed and the index page will show that the employee has successfully been removed.

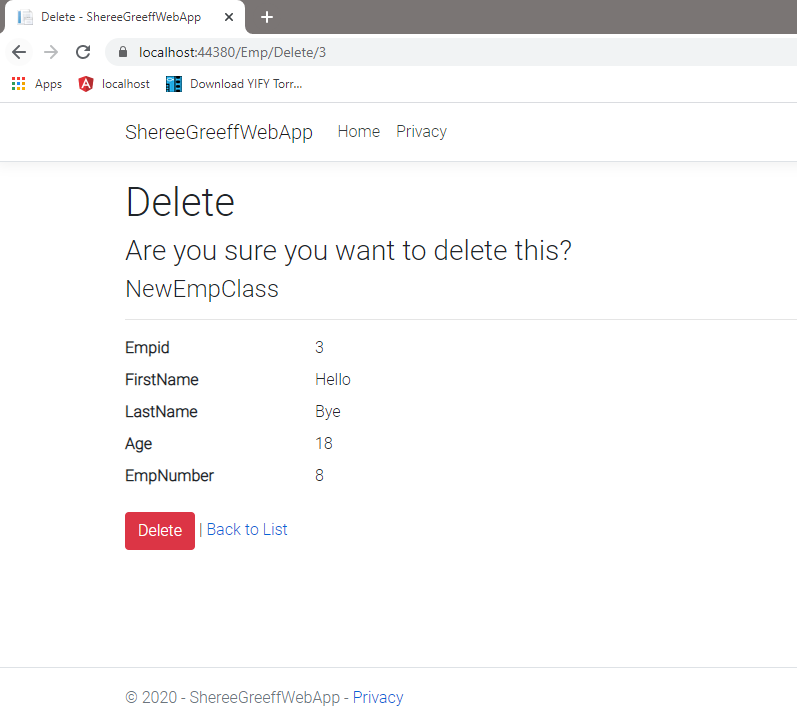


Figure : Delete View

|End of report|