**Application#:**

**Flight Ticket Booking**

1. **Introduction**
2. **Accomplished Items**
3. **Assumptions**
4. **Future scope for enhancements**
5. **Simple Design**
6. **Validations**
7. **Database Tables Scripts**
8. **Solution Submission**
9. **Introduction**

The requirement is to develop an application for “Flight Tickets Booking”, using

* Angular for designing the front end User Interface
* ASP.Net Web API to provide services to the Angular application
* Data will be stored in SQL server database

The application design should be based on user roles, where a user will be authenticated and authorized to access certain functionalities of the application

* User
* Admin
* Agent

**User or Agent should be able to do the following**

* Flights List:
* Search for Bookings:
* Book a Ticket:
* Check Availability:

**Admin should be able to do the following**

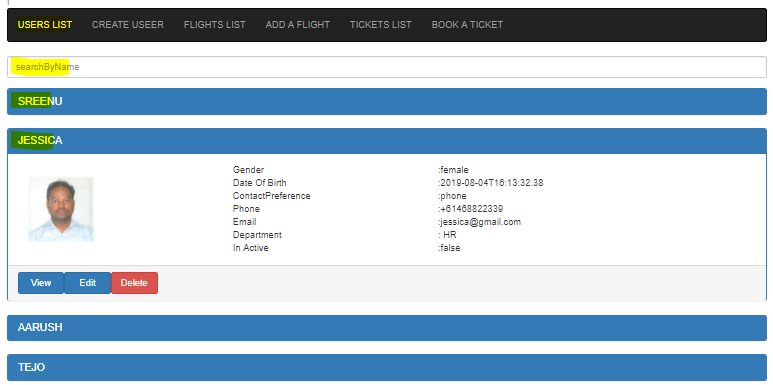
* Add/Delete/Update Flights
* Set the maximum booking numbers (let us say 10)
* Should be able to Make the flight full but still shown to user this flight is full but if they want they can apply for waitlist
* When user apply for waitlist admin will get notification

1. **Accomplished Items**

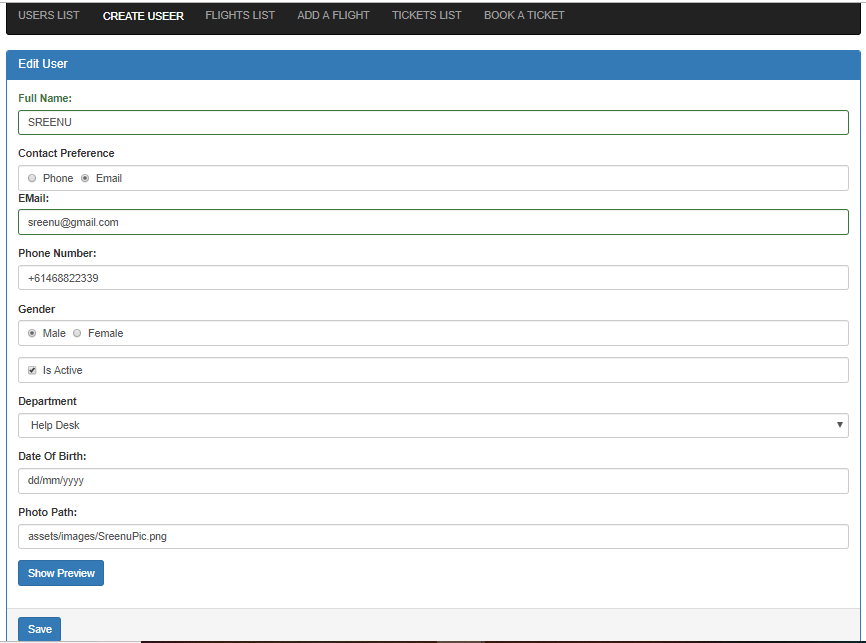
* A working code for Creating Users, Viewing the list, Apply operations like Edit, Delete or Viewing end to end from Angular User Interface through .Net
* Developed a basic User Interface in Angular with Menu Items to view and select the
  + For users to view see the
  + Users List
  + Create User
  + Flights List
  + Add A Flight
  + Tickets List
  + Book a Ticket



* The plan is to show a similar listing mechanism for viewing flights, ticket bookings & users. Currently the listing mechanism for users is working. Similar structure will be applied for flights and ticket booking.
* Clicking any item in the list displays the details of that User or Flight or Booking details respectively. On the item selected you can apply the operations for viewing, editing or deleting.



* User can either Edit or “Add A Flight”/”Book A Ticket”/”Create A User” with below similar UI screen, which working for User. Similarly it will be done for Flights, Tickets booking etc.



* Required field validations are applied.
* Modified data in the screen is acknowledged either to save or discard
* One can View, Edit or Delete entities.
* A .Net Web API was provided for retrieving the data from the database.
* Search mechanism was implemented and working for users

1. **Yet to complete**

* Log on screen and applying role based on authentication and authorization.
* Listing mechanism has to be applied for Flights, Ticket Bookings, Check Availability etc which is similar to user listing mechanism which is working
* Searching mechanism has to be implemented for Flights & Ticket Bookings similar to search mechanism implemented for users.
* Role based authentication mechanism has to be applied.

1. **Assumptions**

* Assumed to implement a basic implementation for flight tickets booking.
* Hope the code shared for Service works at client’s location without any configuration issues. Otherwise we can use the below to make the available for the Angular application.working.

npm install -g json-server

json-server --watch db.json

* The Angular application to work correctly we need to provide the correct baseURL mentioned in the code.

1. **Future scope for enhancements**
   1. The plan is to implement a generic application design so that it should be applicable similarly to any transport ticket booking mechanism.
2. **Simple High Level Design**

Below is a simple design of how the communication happens between the User Interface JavaScript/HTML application and the Web Service

**Angular Application**

**.Net Web API**

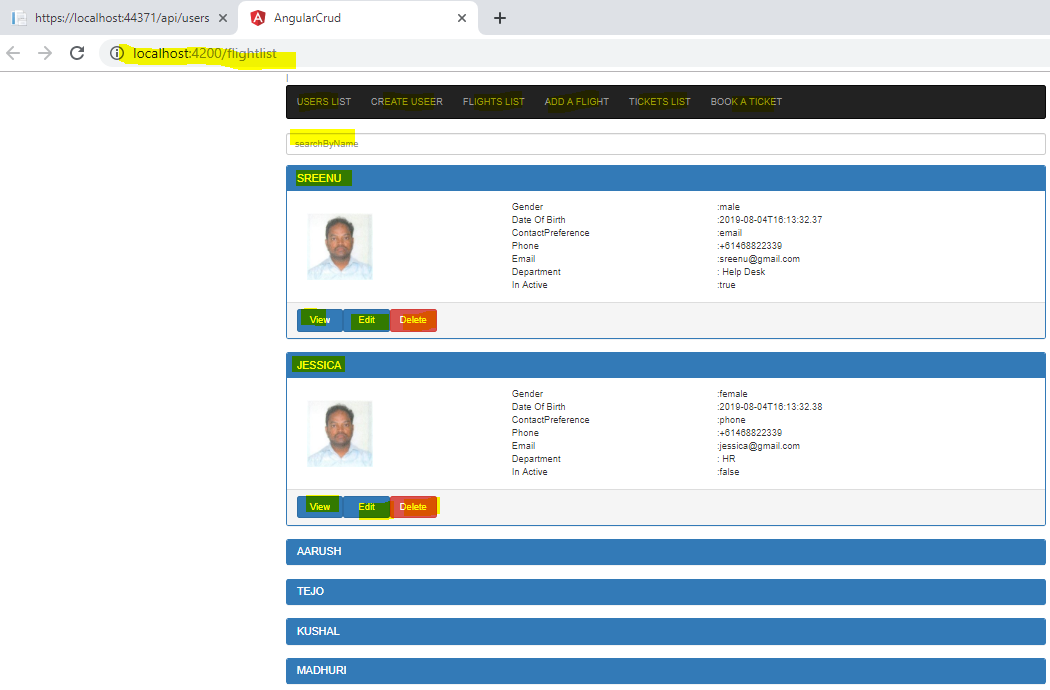
**SQL Server DB**

Entity Frame work

Http Request

Http Request

1. **Simple User Interface**



1. **Database Tables Scripts**

CREATE TABLE UserGroups( -- USER, ADMIN

UserGroupID INT NOT NULL,

UserGroupType VARCHAR (10) NOT NULL,

UserGroupDescription VARCHAR (25),

PRIMARY KEY (UserGroupID)

);

CREATE TABLE Users(

UserID INT NOT NULL,

UserName VARCHAR (20) NOT NULL,

Password VARCHAR (20) NOT NULL,

FirstName VARCHAR (20) NOT NULL,

MiddleName VARCHAR (20) ,

LastName VARCHAR (20) NOT NULL,

DateOfBirth DATETIME NOT NULL,

UserGroupType VARCHAR (10) NOT NULL,

Gender VARCHAR (10) NOT NULL,

EmailID VARCHAR (50) NOT NULL,

PRIMARY KEY (UserID)

);

CREATE TABLE Addresses (

AddressID INT NOT NULL,

UserID INT NOT NULL,

AddressEntityType VARCHAR (20) NOT NULL, -- USER Address or FLIGHT OFFICE Address

AddressType VARCHAR (20) NOT NULL, -- Home, Work/Business

HouseOrUnitNo VARCHAR (20) NOT NULL,

Street VARCHAR (20) NOT NULL,

Suburb VARCHAR (20) NOT NULL,

City VARCHAR (20) NOT NULL,

State VARCHAR (20) NOT NULL,

Country VARCHAR (20) NOT NULL,

Pin Code VARCHAR (20) NOT NULL,

PRIMARY KEY (AddressID)

}

CREATE TABLE Contacts(

ContactID INT NOT NULL,

UserID INT NOT NULL,

ContactType VARCHAR (20) NOT NULL, -- Home, Business, Mobile

ContactDetails VARCHAR (15) NOT NULL,

PRIMARY KEY (ContactID)

}

CREATE TABLE Passengers(

PassengerID INT NOT NULL,

FirstName VARCHAR (20) NOT NULL,

MiddleName VARCHAR (20) ,

LastName VARCHAR (20) NOT NULL,

DateOfBirth DATETIME NOT NULL,

Gender VARCHAR (10) NOT NULL,

PRIMARY KEY (PassengerID)

);

CREATE TABLE DBFieldCategoryTypes(

CategoryTypeID INT NOT NULL,

CategoryType VARCHAR (20) NOT NULL, -- "Address" --"Contact"/Phone

CategoryTypeDesc VARCHAR (20) NOT NULL, -- Home, Work/Business -- Home/Business/Mobile

PRIMARY KEY (CategoryTypeIDID)

);

CREATE TABLE AirLinesDetails(

AirLineID INT NOT NULL,

AirLineRouteNo INT NOT NULL,

AirLineName VARCHAR (20) NOT NULL,

PRIMARY KEY (AirLineID)

);

CREATE TABLE FlightsInformation(

FlightNo INT NOT NULL,

FlightRouteNo INT NOT NULL,

FlightSource VARCHAR (20) NOT NULL,

FlightDestination VARCHAR (20),

PRIMARY KEY (FlightNo)

);

CREATE TABLE FlightBookings(

FlightBookingTicketNo INT NOT NULL,

FlightNo INT NOT NULL,

FlightRouteNo INT NOT NULL,

FlightDepartureDate DATETIME NOT NULL,

FlightArrivalDate DATETIME NOT NULL,

UserID INT NOT NULL,

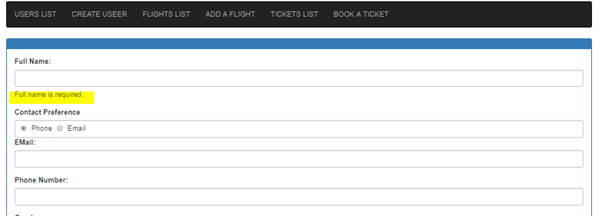
Price float,

PRIMARY KEY (PassengerID)

);

1. **Validations**

Required field validations are applied.



1. **Solution Submission**
   * The Angular code was developed using Visual Studio Code.
   * The .Net Web API was developed in Visual Studio 2019 free version
   * Code is shareD in GitHub.