Transport Management System

A company wants to automate the vehicle allocation system. Every employee can be allocated to a vehicle that has seats open.

Users in this document refer to the transport Management staff. The system provided the below capabilities,

* Add Employee information
* show the routes and stops
* Show Vehicle information
* Fix a vehicle for employee based on its routs and stop .
* Fix another vehicle if vehicle one has been cancelled.
* Cancel the ride if the employee wants to cancel and redirect to Login Page.

The system will be secured that only logged in users can perform any of the above functions.

Detailed Requirements

User Setup

Each Front office staff will have a unique user name and password. There is no screen to manage the users, the data will be managed in the user table directly in the database. The database table will have the below data for all the front office staff.

* User name (unique, no special characters)
* Password

Screen 1: Login

This screen will prompt the user to enter their user name and password. Both fields are mandatory and to be validated. The page will also have a Company logo

The user name and password will be validated at the server side against the database table – if the details match, the users will be redirected to the Register and if not, they will get the same login page with the error message “You’ve entered an incorrect user name or password”.

Screen 2: Add Employee

Users will use this screen to register the Employee information to the system. Screen will have the below fields,

* Employee Id (auto generated number by the system, not editable by the users)
* Employee First Name (No special characters allowed)
* Employee Last Name (No special characters allowed)
* Sex (Choices - M/F/Others)
* Employee Address
* Employee Contact Number
* Employee Boarding point

When the users submit the data, validate and save to the database.

Screen 3: Show Route

Users will use this screen to Identify the Route. Screen will have the below fields,

* Route Numbers
* Number Of Stops (No special characters allowed)
* Stop 1
* Stop 2
* stop 3

When the users selects the route, validate and Redirect to Vehicle Page

Screen 4: Fixing the Vehicle

Users will use this screen to see the vehicles available and book that vehicle. If Vehicle one is full of capacity , then vehicle two will be given in the screen. Screen will have the below fields,

* VehicleNumber
* Capacity
* IsAC
* Available seats
* IsOperational

This Page will show all the vehicles availabity. If the vehicle one is of full capacity then this page shoud show vehicle unavailakle and Redirect to vehicle two and so on.

Screen 5: Cancel Ride

Users if want to cancel the ride if he or she faces any difficulties ,he can cancel the ride which he or she has booked.Screen will have the below details.

* Reason for Cancellation

And user hits Cancellation it should be redirect to the Register Screen.

General Requirements

* All screens will have a button .“Go to Home” to take the user back to the Register screen
* Whenever an operation is successful, the users should be shown a message as such. E.g., if the vehicle has been successfully allocated, it will display “vehicle successfully allocated. Enjoy Journey”.
* On failures, the users should be shown appropriate friendly error message (not exceptions).

Technical Requirements

* Tables should be designed appropriately (proper data types, table/column names, constraints), use a SQL Server database
* The tables should be created only using SQL Files and these should be available in the repo
* Front end – ASP.NET Core MVC, HTML5, CSS, Bootstrap, Javascript/Jquery
* Business Logic – Web API/Service in MVC
* Data Access – EF Core (DB/Model first approach)
* All names used in the code (classes, variables, properties etc.,) should be meaningful and follow consistent naming approach (Camel Case, Pascal Case etc.,)
* Classes/methods/properties should have proper accessibility modifiers
* Use code formatting to ensure the code is readable, write comments where required so that the reviewers can understand your code

IMPORTANT

* Create a new repo “ProjectA” in your account and use that for this exercise
* Push the code to the remote repository often or at least before end of each day.
* Repo should have a folder “DB Script” and it will have all the SQL Files (tables, stored procedure etc.,)
* Repo should have a folder “src” and it will have your ASP.NET Core Projects