Task – 5

Htmlfile:

<!DOCTYPE html>

<html>

<head>

    <title>Client Creation</title>

    <link rel="stylesheet" type="text/css" href="styles.css">

</head>

<body>

    <div class="container">

        <h1>Create a Client</h1>

        <form action="clientForm" method="post">

            <form action="#" method="post">

            <p>Please fill in this form to create a client</p>

            <br>

            <b>Name</b>

            <br>

            <input type="text" class="form-control" name="txtName" placeholder="Enter Name" required>

            <br>

            <b>Email</b>

            <br>

            <input type="text" class="form-control" name="txtEmail" placeholder="Enter Email" required>

            <br>

            <b>Address</b>

            <br>

            <input type="text" class="form-control" name="txtAddress" placeholder="Enter Address" required>

            <br>

            <b>Password</b>

            <br>

            <input type="password" class="form-control" name="txtPassword" placeholder="Enter Password" required>

            <br>

            <b>Repeat Password</b>

            <br>

            <input type="password" class="form-control" name="txtPassword" placeholder="Repeat Password" required>

            <br>

            <p>Create a Meeting Schedule</p>

            <br>

            <b>Meeting Topic</b>

            <br>

            <input type="text" name="txtName" placeholder="Enter Meeting Topic" required>

            <br>

            <b>Number of People</b>

            <br>

            <input type="text" name="txtName" placeholder="Enter Number of People" required>

            <br>

            <b>Start Time</b>

            <input type="datetime-local" id="meeting-time" name="meeting-time" />

            <br>

            <button type="submit">Register</button>

            <!-- <input type="submit" value="Register" > -->

        </form>

       </div>

</body>

</html>

Cssfile:

body {

    font-family: Arial, sans-serif;

    background-color: #f0f0f0;

    margin: 0;

    padding: 0;

}

.container {

    max-width: 600px;

    margin: 50px auto;

    background-color: #fff;

    padding: 20px;

    box-shadow: 0 0 10px rgba(0, 0, 0, 0.1);

}

h1 {

    text-align: center;

}

form {

    display: flex;

    flex-direction: column;

}

label {

    margin: h5 col text-right;

}

input {

    padding: 10px;

    margin-bottom: 20px;

    border: 1px solid #ccc;

    border-radius: 5px;

}

button {

    padding: 10px;

    background-color: #28a745;

    color: #fff;

    border: none;

    border-radius: 5px;

    cursor: pointer;

}

button:hover {

    background-color: #218838;

}

label {

    display: block;

    font:

      1rem 'Fira Sans',

      sans-serif;

  }

input,

label {

    margin: 0.4rem 0;

  }

Task -6,9

App.module.ts

import { NgModule } from '@angular/core';

import { BrowserModule } from '@angular/platform-browser';

import { AppRoutingModule } from './app-routing.module';

import { AppComponent } from './app.component';

import { ClientListComponent } from './client-list/client-list.component';

import { ClientDetailsComponent } from './client-details/client-details.component';

import { ClientCreateComponent } from './client-create/client-create.component';

import { ClientEditComponent } from './client-edit/client-edit.component';

import { FormsModule } from '@angular/forms';

@NgModule({

  declarations: [

    AppComponent,

    ClientListComponent,

    ClientDetailsComponent,

    ClientCreateComponent,

    ClientEditComponent

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,

    FormsModule

  ],

  providers: [],

  bootstrap: [AppComponent]

})

export class AppModule { }

client.service.ts

import { Injectable } from '@angular/core';

import { Observable, of } from 'rxjs';

export interface Client {

  id: number;

  name: string;

  email: string;

  phone: string;

}

@Injectable({

  providedIn: 'root'

})

export class ClientService {

  private clients: Client[] = [

    { id: 1, name: 'John Doe', email: 'john@example.com', phone: '1234567890' },

    { id: 2, name: 'Jane Smith', email: 'jane@example.com', phone: '0987654321' }

  ];

  constructor() { }

  getClients(): Observable<Client[]> {

    return of(this.clients);

  }

  getClient(id: number): Observable<Client | undefined> {

    return of(this.clients.find(client => client.id === id));

  }

  addClient(client: Client): void {

    this.clients.push(client);

  }

  updateClient(client: Client): void {

    const index = this.clients.findIndex(c => c.id === client.id);

    if (index !== -1) {

      this.clients[index] = client;

    }

  }

  deleteClient(id: number): void {

    this.clients = this.clients.filter(client => client.id !== id);

  }

}

App-routing.module.ts

import { NgModule } from '@angular/core';

import { RouterModule, Routes } from '@angular/router';

import { ClientListComponent } from './client-list/client-list.component';

import { ClientDetailsComponent } from './client-details/client-details.component';

import { ClientCreateComponent } from './client-create/client-create.component';

import { ClientEditComponent } from './client-edit/client-edit.component';

const routes: Routes = [

  { path: '', redirectTo: '/client-list', pathMatch: 'full' },

  { path: 'client-list', component: ClientListComponent },

  { path: 'client-details/:id', component: ClientDetailsComponent },

  { path: 'client-create', component: ClientCreateComponent },

  { path: 'client-edit/:id', component: ClientEditComponent }

];

@NgModule({

  imports: [RouterModule.forRoot(routes)],

  exports: [RouterModule]

})

export class AppRoutingModule { }

client-list.component.ts

import { Component, OnInit } from '@angular/core';

import { ClientService, Client } from '../client.service';

@Component({

selector: 'app-client-list',

templateUrl: './client-list.component.html',

styleUrls: ['./client-list.component.css']

})

export class ClientListComponent implements OnInit {

clients: Client[] = [];

constructor(private clientService: ClientService) { }

ngOnInit(): void {

this.clientService.getClients().subscribe(clients => this.clients = clients);

}

}

client-list.component.html

<h2>Client List</h2>

<ul>

<li \*ngFor="let client of clients">

{{ client.name }} - {{ client.email }} - {{ client.phone }}

<button [routerLink]="['/client-details', client.id]">View</button>

<button [routerLink]="['/client-edit', client.id]">Edit</button>

<button (click)="deleteClient(client.id)">Delete</button>

</li>

</ul>

<button [routerLink]="['/client-create']">Add New Client</button>

client-details.component.ts

import { Component, OnInit } from '@angular/core';

import { ActivatedRoute } from '@angular/router';

import { ClientService, Client } from '../client.service';

@Component({

selector: 'app-client-details',

templateUrl: './client-details.component.html',

styleUrls: ['./client-details.component.css']

})

export class ClientDetailsComponent implements OnInit {

client: Client | undefined;

constructor(

private route: ActivatedRoute,

private clientService: ClientService

) { }

ngOnInit(): void {

const id = +this.route.snapshot.paramMap.get('id')!;

this.clientService.getClient(id).subscribe(client => this.client = client);

}

}

client-details.component.html

<div \*ngIf="client">

<h2>{{ client.name }} Details</h2>

<div><span>Id: </span>{{ client.id }}</div>

<div><span>Email: </span>{{ client.email }}</div>

<div><span>Phone: </span>{{ client.phone }}</div>

</div>

client-create.component.ts

import { Component } from '@angular/core';

import { Router } from '@angular/router';

import { ClientService, Client } from '../client.service';

@Component({

selector: 'app-client-create',

templateUrl: './client-create.component.html',

styleUrls: ['./client-create.component.css']

})

export class ClientCreateComponent {

client: Client = { id: 0, name: '', email: '', phone: '' };

constructor(private clientService: ClientService, private router: Router) { }

createClient(): void {

this.clientService.addClient(this.client);

this.router.navigate(['/client-list']);

}

}

client-create.component.html

<h2>Create Client</h2>

<form (ngSubmit)="createClient()">

<label>Name:

<input [(ngModel)]="client.name" name="name">

</label>

<label>Email:

<input [(ngModel)]="client.email" name="email">

</label>

<label>Phone:

<input [(ngModel)]="client.phone" name="phone">

</label>

<button type="submit">Create</button>

</form>

client-edit.component.ts

import { Component, OnInit } from '@angular/core';

import { ActivatedRoute, Router } from '@angular/router';

import { ClientService, Client } from '../client.service';

@Component({

selector: 'app-client-edit',

templateUrl: './client-edit.component.html',

styleUrls: ['./client-edit.component.css']

})

export class ClientEditComponent implements OnInit {

client: Client | undefined;

constructor(

private route: ActivatedRoute,

private clientService: ClientService,

private router: Router

) { }

ngOnInit(): void {

const id = +this.route.snapshot.paramMap.get('id')!;

this.clientService.getClient(id).subscribe(client => this.client = client);

}

updateClient(): void {

if (this.client) {

this.clientService.updateClient(this.client);

this.router.navigate(['/client-list']);

}

}

}

client-edit.component.html

<h2>Edit Client</h2>

<div \*ngIf="client">

<form (ngSubmit)="updateClient()">

<label>Name:

<input [(ngModel)]="client.name" name="name">

</label>

<label>Email:

<input [(ngModel)]="client.email" name="email">

</label>

<label>Phone:

<input [(ngModel)]="client.phone" name="phone">

</label>

<button type="submit">Update</button>

</form>

</div>

App.component.html

<app-client-create></app-client-create>

<app-client-edit></app-client-edit>

<app-client-list></app-client-list>

<app-client-details></app-client-details>

Task-7

CREATE DATABASE IF NOT EXISTS ClientMeetingDB;

USE ClientMeetingDB;

CREATE TABLE IF NOT EXISTS Clients (

    ClientID INT AUTO\_INCREMENT PRIMARY KEY,

    FirstName VARCHAR(50) NOT NULL,

    LastName VARCHAR(50) NOT NULL,

    Email VARCHAR(100),

    PhoneNumber VARCHAR(20),

    Address VARCHAR(255),

    City VARCHAR(50),

    State VARCHAR(50),

    ZipCode VARCHAR(10),

    Country VARCHAR(50),

    CreatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE IF NOT EXISTS Meetings (

    MeetingID INT AUTO\_INCREMENT PRIMARY KEY,

    MeetingDate DATETIME NOT NULL,

    Subject VARCHAR(255) NOT NULL,

    Location VARCHAR(255),

    Notes TEXT,

    CreatedAt TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE IF NOT EXISTS Clients\_Meetings (

    ClientID INT,

    MeetingID INT,

    PRIMARY KEY (ClientID, MeetingID),

    FOREIGN KEY (ClientID) REFERENCES Clients(ClientID),

    FOREIGN KEY (MeetingID) REFERENCES Meetings(MeetingID)

);

Task-8

Feature: Client and Client Meeting Scenarios

# Scenario 1: Scheduling a Client Meeting

Scenario: Scheduling a meeting with a client

Given I am logged in as a project manager

When I navigate to the "Client Meetings" section

And I click on "Schedule a Meeting"

And I fill in the client details

| Client Name | Contact Information |

| Acme Corp | acme@corp.com |

And I select a date and time for the meeting

| Date | Time |

| 2024-08-15 | 10:00 AM |

And I add a meeting agenda

| Agenda Item |

| Project Kick-off Call |

And I click "Save"

Then the meeting should be scheduled successfully

And the client should receive a notification email

# Scenario 2: Updating a Client Meeting

Scenario: Updating the details of a scheduled meeting

Given I am logged in as a project manager

And I have an existing meeting scheduled with Acme Corp on 2024-08-15 at 10:00 AM

When I navigate to the "Client Meetings" section

And I select the meeting with Acme Corp

And I click "Edit"

And I change the meeting date to 2024-08-16

And I update the agenda to "Project Kick-off and Budget Discussion"

And I click "Save"

Then the meeting details should be updated successfully

And the client should receive an updated notification email

# Scenario 3: Cancelling a Client Meeting

Scenario: Cancelling a scheduled meeting with a client

Given I am logged in as a project manager

And I have an existing meeting scheduled with Acme Corp on 2024-08-15 at 10:00 AM

When I navigate to the "Client Meetings" section

And I select the meeting with Acme Corp

And I click "Cancel"

And I confirm the cancellation

Then the meeting should be cancelled successfully

And the client should receive a cancellation notification email