**Event Management App.**

**Source code:**

**Admin.component.html:**

<div class="container">

<div class="row">

<div class="col-25" \*ngIf="showRegistration">

<h2>Registration Form</h2>

<form [formGroup]="registrationForm" (ngSubmit)="register()">

<div class="form-group">

<label for="name">Name</label>

<input type="text" id="name" formControlName="name" required>

</div>

<div class="form-group">

<label for="email">Email</label>

<input type="email" id="email" formControlName="email" required>

</div>

<div class="form-group">

<label for="password">Password</label>

<input type="password" id="password" formControlName="password" required>

</div>

<button type="submit" [disabled]="!registrationForm.valid">Register</button>

<p>Already have an account? <a (click)="toggleForm()">Login</a></p>

</form>

</div>

<div class="col-25" \*ngIf="!showRegistration">

<h2>Login Form</h2>

<form [formGroup]="loginForm" (ngSubmit)="login()">

<div class="form-group">

<label for="email">Email</label>

<input type="email" id="email" formControlName="email" required>

</div>

<div class="form-group">

<label for="password">Password</label>

<input type="password" id="password" formControlName="password" required>

</div>

<button type="submit" [disabled]="!loginForm.valid" [routerLink]="['/employee']">Login</button>

<p>Don't have an account? <a (click)="toggleForm()">Register Here</a></p>

</form>

</div>

<div \*ngIf="isLoggedIn">

<button (click)="logout()">Logout</button>

</div>

</div>

</div>

<router-outlet></router-outlet>

**Admin.component.ts:**

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

import { FormBuilder, FormGroup, Validators } from '@angular/forms';

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

import { HttpClient } from '@angular/common/http';

@Component({

selector: 'app-admin',

templateUrl: './admin.component.html',

styleUrls: ['./admin.component.css']

})

export class AdminComponent {

registrationForm: FormGroup;

loginForm: FormGroup;

isLoggedIn: boolean = false;

showRegistration: boolean = false;

constructor(private formBuilder: FormBuilder, private router: Router, private http: HttpClient) {

this.registrationForm = this.formBuilder.group({

name: ['', Validators.required],

email: ['', [Validators.required, Validators.email]],

password: ['', [Validators.required, Validators.minLength(6)]]

});

this.loginForm = this.formBuilder.group({

email: ['', [Validators.required, Validators.email]],

password: ['', Validators.required]

});

}

register() {

if (this.registrationForm.valid) {

const registrationData = this.registrationForm.value;

// Send registration data to JSON server

this.http.post('http://localhost:3000/users', registrationData).subscribe(

(response) => {

console.log('Registration successful:', response);

this.showRegistration = false;

},

(error) => {

console.error('Registration failed:', error);

}

);

}

}

login() {

if (this.loginForm.valid) {

const enteredEmail = this.loginForm.value.email;

const enteredPassword = this.loginForm.value.password;

if (enteredEmail === this.registrationForm.value.email && enteredPassword === this.registrationForm.value.password) {

this.isLoggedIn = true;

this.router.navigate(['/employee']);

} else {

console.log('Invalid login credentials');

}

}

}

logout() {

this.isLoggedIn = false;

}

toggleForm() {

this.showRegistration = !this.showRegistration;

this.registrationForm.reset();

this.loginForm.reset();

}

}

**Admin.component.css:**

.container {

background-image: url('https://www.w3schools.com/Css/img\_mountains.jpg');

background-repeat: no-repeat;

background-size: cover;

background-position: center;

height: 90vh;

display: flex;

justify-content: center;

align-items: center;

}

.container .row {

margin-bottom: 50px;

}

.container h2 {

margin-bottom: 10px;

}

.container form {

padding: 20px;

background-color: rgba(240, 233, 233, 0.8);

border-radius: 5px;

width: 400px;

height: 300px;

display: flex;

flex-direction: column;

justify-content: center;

align-items: center;

}

.container form .form-group {

margin-bottom: 10px;

}

.container form label {

display: block;

font-weight: bold;

}

.container form input[type="text"],

.container form input[type="email"],

.container form input[type="password"] {

width: 100%;

padding: 8px;

border: 1px solid #ccc;

border-radius: 4px;

}

.container form button {

padding: 8px 16px;

background-color: #4CAF50;

color: white;

border: none;

border-radius: 4px;

cursor: pointer;

}

.container form button:not(:last-child) {

margin-right: 10px;

}

.container form button:disabled {

background-color: #ccc;

cursor: not-allowed;

}

.container button {

padding: 8px 16px;

background-color: #4CAF50;

color: white;

border: none;

border-radius: 4px;

cursor: pointer;

}

.container button:not(:last-child) {

margin-right: 10px;

}

**Employees.component.html:**

<nav class="navbar navbar-expand-lg navbar-light bg-light">

<div class="collapse navbar-collapse" id="navbarTogglerDemo02">

<ul class="navbar-nav mr-auto mt-2 mt-md-0">

<li class="nav-item active">

<a class="nav-link" routerLink="/admin">Admin</a>

</li>

<li class="nav-item ">

<a class="nav-link" routerLink="/home">Home </a>

</li>

<li class="nav-item">

<a class="nav-link" routerLink="/employee">Employee</a>

</li>

</ul>

</div>

</nav>

<div class="add">

<button (click)="!showAddForm = true">Add Employee</button>

</div>

<div class="table-container">

<table class="custom-table">

<thead>

<tr>

<th>First Name</th>

<th>Last Name</th>

<th>Email</th>

<th>Actions</th>

</tr>

</thead>

<tbody>

<tr \*ngFor="let employee of employees">

<td>{{ employee.firstName }}</td>

<td>{{ employee.lastName }}</td>

<td>{{ employee.email }}</td>

<td>

<button (click)="toggleDetails(employee)">View Details</button>

</td>

</tr>

</tbody>

</table>

</div>

<div \*ngIf="showDetails && selectedEmployee">

<div class="employee-details">

<h3>Employee Details</h3>

<p><strong>First Name:</strong> {{ selectedEmployee?.firstName }}</p>

<p><strong>Last Name:</strong> {{ selectedEmployee?.lastName }}</p>

<p><strong>Email:</strong> {{ selectedEmployee?.email }}</p>

<div class="button-container">

<button (click)="editEmployee(selectedEmployee)">Edit</button>

<button (click)="deleteEmployee(selectedEmployee)">Delete</button>

</div>

</div>

</div>

<div \*ngIf="isEditMode" class="employee-details">

<h3>Edit Employee</h3>

<form [formGroup]="employeeForm" (ngSubmit)="updateEmployee()" >

<div>

<label for="firstName">First Name:</label>

<input type="text" id="firstName" formControlName="firstName">

</div>

<div>

<label for="lastName">Last Name:</label>

<input type="text" id="lastName" formControlName="lastName">

</div>

<div>

<label for="email">Email:</label>

<input type="email" id="email" formControlName="email">

</div>

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

<button type="button" (click)="cancelEdit()">Cancel</button>

</form>

</div>

<div \*ngIf="showAddForm" class="add-employee">

<h3>Add Employee</h3>

<form [formGroup]="employeeForm" (ngSubmit)="addEmployee()">

<div>

<label for="firstName">First Name:</label>

<input type="text" id="firstName" formControlName="firstName">

</div>

<div>

<label for="lastName">Last Name:</label>

<input type="text" id="lastName" formControlName="lastName">

</div>

<div>

<label for="email">Email:</label>

<input type="email" id="email" formControlName="email">

</div>

<button type="submit">Add Employee</button>

</form>

</div>

**Employees.component.ts:**

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

import { FormBuilder, FormGroup, Validators } from '@angular/forms';

import { Employee } from '../employee.model';

import { HttpClient } from '@angular/common/http';

@Component({

selector: 'app-employees',

templateUrl: './employees.component.html',

styleUrls: ['./employees.component.css']

})

export class EmployeesComponent implements OnInit {

employeeForm: FormGroup;

employees: Employee[] = [];

selectedEmployee: Employee | null = null;

isEditMode: boolean = false;

showAddForm = false;

showDetails: boolean = false;

constructor(private formBuilder: FormBuilder, private http: HttpClient) {

this.employeeForm = this.formBuilder.group({

firstName: ['', Validators.required],

lastName: ['', Validators.required],

email: ['', [Validators.required, Validators.email]]

});

}

ngOnInit() {

this.fetchEmployees();

}

fetchEmployees() {

this.http.get<Employee[]>('http://localhost:3000/employees').subscribe(

(data) => {

this.employees = data;

},

(error) => {

console.error('Error fetching employee data:', error);

}

);

}

addEmployee() {

this.showAddForm=false;

this.showDetails=false;

if (this.employeeForm.valid) {

const newEmployee: Employee = {

id: 0,

firstName: this.employeeForm.value.firstName,

lastName: this.employeeForm.value.lastName,

email: this.employeeForm.value.email

};

this.http.post('http://localhost:3000/employees', newEmployee).subscribe(

(response) => {

console.log('Employee added successfully:', response);

this.fetchEmployees();

this.employeeForm.reset();

},

(error) => {

console.error('Error adding employee:', error);

}

);

} else {

console.error('Invalid form data');

}

}

editEmployee(employee: Employee) {

this.selectedEmployee = { ...employee };

this.showAddForm=false;

this.showDetails=false;

this.employeeForm.patchValue({

firstName: employee.firstName,

lastName: employee.lastName,

email: employee.email,

});

this.isEditMode = true;

}

toggleDetails(employee: Employee) {

this.showDetails = !this.showDetails;

this.showAddForm=false;

this.isEditMode=false;

if (this.showDetails) {

this.selectedEmployee = employee;

} else {

this.selectedEmployee = null;

}

}

deleteEmployee(employee: Employee) {

const url = `http://localhost:3000/employees/${employee.id}`;

this.http.delete(url).subscribe(

() => {

console.log('Employee deleted successfully.');

const index = this.employees.findIndex(emp => emp.id === employee.id);

if (index !== -1) {

this.employees.splice(index, 1);

}

this.selectedEmployee = null;

},

(error) => {

console.error('Error deleting employee:', error);

// Handle error scenarios

}

);

}

updateEmployee() {

this.showAddForm=false;

this.showDetails=false;

if (this.employeeForm.valid && this.selectedEmployee) {

this.selectedEmployee.firstName = this.employeeForm.value.firstName;

this.selectedEmployee.lastName = this.employeeForm.value.lastName;

this.selectedEmployee.email = this.employeeForm.value.email;

this.http.put(`http://localhost:3000/employees/${this.selectedEmployee.id}`, this.selectedEmployee).subscribe(

(response) => {

console.log('Employee updated successfully:', response);

this.isEditMode = false;

this.employeeForm.reset();

this.fetchEmployees();

},

(error) => {

console.error('Failed to update employee:', error);

}

);

}

}

cancelEdit() {

this.isEditMode = false;

this.employeeForm.reset(); }

}

**Employee.component.css:**

ul {

list-style-type: none;

margin-left: 50px;

margin-right: 50px;

padding: 0;

overflow: hidden;

background-color: #333;

}

li {

float: left;

}

li a {

display: block;

color: white;

text-align: center;

padding: 14px 16px;

text-decoration: none;

margin-bottom: 0;

}

li a:hover {

background-color: #111;

}

.head {

text-align: center;

background-color: beige;

padding:10px;

height:70px;

margin-left: 50px;

margin-right: 50px;

}

.table-container {

display: flex;

padding:10px;

justify-content: center;

margin-top: 0;

}

.custom-table {

border-collapse: collapse;

text-align: center;

width: 100%;

margin: 50px;

padding:10px;

}

.custom-table th {

font-weight: bold;

padding:5px;

background-color:rgb(42, 42, 33);

color: white;

}

.employee-details {

border: 1px solid #ccc;

background-color: beige;

display: flex;

justify-content: center;

align-items: center;

flex-direction: column;

padding:5px;

margin-left: 50px;

margin-right: 50px;

justify-content: center;

}

.table-row {

border-bottom: 1px solid #ddd;

padding:10px;

margin-bottom: 10px;

justify-content: center;

background-color:beige;

}

.custom-table td {

text-align: center;

padding:2px;

background-color:beige;

}

.add {

margin-left:50px;

margin-right:50px;

background-color:beige;

padding-bottom:0;

}

.add-button {

margin-left: 50px;

margin-top: 10px;

}

.add-form {

margin-left: 50px;

margin-right: 50px;

padding: 5px;

background-color: beige;

padding:10px;

height:70px;

}

.add-employee {

background-color: beige;

display: flex;

flex-direction: column;

align-items: center;

padding: 10px;

margin-top: 10px;

}

.button-container {

display: flex;

gap: 10px;

margin-top: 10px;

}

.employee-details form {

margin-top: 10px;

}

.employee-details label {

display: block;

font-weight: bold;

}

.employee-details input {

width: 100%;

padding: 5px;

margin-bottom: 10px;

}

.employee-details button[type="submit"],

.employee-details button[type="button"] {

margin-top: 10px;

}

**Home.component.html:**

<nav class="navbar navbar-expand-lg navbar-light bg-light">

<div class="collapse navbar-collapse" id="navbarTogglerDemo02">

<ul class="navbar-nav mr-auto mt-2 mt-md-0">

<li class="nav-item active">

<a class="nav-link" routerLink="/admin">Admin</a>

</li>

<li class="nav-item ">

<a class="nav-link" routerLink="/home">Home </a>

</li>

<li class="nav-item">

<a class="nav-link" routerLink="/employee">Employee</a>

</li>

</ul>

</div>

</nav>

<br>

<div class="home">

<p>home works!</p>

</div>

**Home.component.ts:**

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

@Component({

selector: 'app-home',

templateUrl: './home.component.html',

styleUrls: ['./home.component.css']

})

export class HomeComponent {

}

**App-routing.module.ts:**

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

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

import { EmployeesComponent } from './employees/employees.component';

import { AdminComponent } from './admin/admin.component';

import { HomeComponent } from './home/home.component';

const routes: Routes = [

{ path: '', component: AdminComponent },

{ path: 'home', component: HomeComponent },

{ path: 'employee', component: EmployeesComponent },

{ path: '\*\*', redirectTo: '' }

];

@NgModule({

imports: [RouterModule.forRoot(routes)],

exports: [RouterModule]

})

export class AppRoutingModule { }

**App.component.html:**

<div>

<nav>

<ul>

<ng-container \*ngIf="isLoggedIn">

<li><a routerLink="/home">Home</a></li>

<li><a routerLink="/employee">Employee</a></li>

</ng-container>

</ul>

</nav>

<router-outlet></router-outlet> <!-- Make sure you have this router-outlet element -->

</div>

**App.component.ts:**

<div>

<nav>

<ul>

<ng-container \*ngIf="isLoggedIn">

<li><a routerLink="/home">Home</a></li>

<li><a routerLink="/employee">Employee</a></li>

</ng-container>

</ul>

</nav>

<router-outlet></router-outlet> <!-- Make sure you have this router-outlet element -->

</div>

**App.module.ts:**

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

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

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

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

import { HttpClientModule } from '@angular/common/http';

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

import { AdminComponent } from './admin/admin.component';

import { EmployeesComponent } from './employees/employees.component';

import { HomeComponent } from './home/home.component';

@NgModule({

declarations: [

AppComponent,

AdminComponent,

EmployeesComponent,

HomeComponent,

],

imports: [

BrowserModule,

ReactiveFormsModule,

AppRoutingModule,

HttpClientModule // Include AppRoutingModule here

],

providers: [],

bootstrap: [AppComponent]

})

export class AppModule { }

**Employee.model.ts:**

export interface Employee {

id:number;

firstName: string;

lastName: string;

email: string;

}

**Db.json:**

{

"employees": [

{

"id": 2,

"firstName": "Steve",

"lastName": "Palmer",

"email": "[steve@codingthesmartway.com](mailto:steve@codingthesmartway.com)"

},

{

"id": 3,

"firstName": "Ann",

"lastName": "Smith",

"email": "[ann@codingthesmartway.com](mailto:ann@codingthesmartway.com)"

},

{

"id": 4,

"firstName": "Deepika",

"lastName": "veeravalli",

"email": "[deepika@c.com](mailto:deepika@c.com)"

},

{

"id": 5,

"firstName": "abc",

"lastName": "abcd",

"email": "[abcd@c.c](mailto:abcd@c.c)"

}

],

"users": [

{

"name": "deepika",

"email": "[deepika@c.c](mailto:deepika@c.c)",

"password": "12345678",

"id": 1

},

{

"name": "deepika",

"email": "[deepika@c.c](mailto:deepika@c.c)",

"password": "12345678",

"id": 2

},

{

"name": "abcd",

"email": "[abcd@c.c](mailto:abcd@c.c)",

"password": "12345678",

"id": 3

},

{

"name": "abcd",

"email": "[abcd@c.c](mailto:abcd@c.c)",

"password": "12345678",

"id": 4

}

]

}