**Source Code**

Index.html

<!doctype html>

<html lang="en">

<head>

  <meta charset="utf-8">

  <title>EventManagement</title>

  <base href="/">

  <meta name="viewport" content="width=device-width, initial-scale=1">

  <link rel="icon" type="image/x-icon" href="favicon.ico">

  <script src="https://kit.fontawesome.com/a076d05399.js" crossorigin="anonymous"></script>

  <!-- Latest compiled and minified CSS -->

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/css/bootstrap.min.css" rel="stylesheet">

<!-- Latest compiled JavaScript -->

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.1/dist/js/bootstrap.bundle.min.js"></script>

</head>

<body>

  <app-root></app-root>

</body>

</html>

App.component.html

<router-outlet></router-outlet>

<app-login></app-login>

App.component.ts

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

@Component({

  selector: 'app-root',

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

  styleUrls: ['./app.component.css']

})

export class AppComponent {

  title = 'Event-Management';

}

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 { LoginComponent } from './login/login.component';

import { DashboardComponent } from './dashboard/dashboard.component';

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

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

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

import { ApiService } from './service/api.service';

@NgModule({

  declarations: [

    AppComponent,

    LoginComponent,

    DashboardComponent

  ],

  imports: [

    BrowserModule,

    AppRoutingModule,HttpClientModule,RouterModule,ReactiveFormsModule

  ],

  // Insert ApiService for providing service

  providers: [ApiService],

  bootstrap: [AppComponent]

})

export class AppModule { }

app-routing.module.ts

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

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

import { DashboardComponent } from './dashboard/dashboard.component';

import { LoginComponent } from './login/login.component';

const routes: Routes = [

  {path: 'login',component: LoginComponent},

  { path: 'dashboard', component: DashboardComponent }

];

@NgModule({

  imports: [RouterModule.forRoot(routes)],

  exports: [RouterModule]

})

export class AppRoutingModule { }

login.component.html

<!-- Section: Design Block -->

<section class="text-center text-lg-start">

    <!-- Jumbotron -->

    <div class="container py-4 bg-warning">

        <div class="row g-0 align-items-center">

            <div class="col-lg-12 mb-4 mb-lg-0">

                <div class="card cascading-right" style="

            background: hsla(0, 0%, 100%, 0.55);

            backdrop-filter: blur(30px);

          ">

                    <div class="card-body p-5 shadow-5 text-center">

                        <h2 class="fw-bold mb-5">Admin Login</h2>

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

                            <!--Taking Email With Validation -->

                            <div class="form-outline mb-4">

                                <label class="form-label" for="form3Example3">Email Id :</label>

                                <input type="email" formControlName="email"

                                    [ngClass]="{ 'is-invalid': submitted && f.email.errors }" class="form-control" />

                                <div \*ngIf="submitted && f.email.errors" class="invalid-feedback">

                                    <div \*ngIf="f.email.errors.required">Email Id is Required</div>

                                    <div \*ngIf="f.email.errors.email">

                                        Email Id must be a valid Email Id (Enter : Admin@gmail.com)

                                    </div>

                                </div>

                            </div>

                            <!--Taking Password with validation-->

                            <div class="form-outline mb-4">

                                <label class="form-label" for="form3Example4">Password :</label>

                                <input type="password" formControlName="password" class="form-control"

                                    [ngClass]="{ 'is-invalid': submitted && f.password.errors }" />

                                <div \*ngIf="submitted && f.password.errors" class="invalid-feedback">

                                    <div \*ngIf="f.password.errors.required">

                                        Password is Required (Enter : Pass@1234)

                                    </div>

                                </div>

                            </div>

                            <!-- Submit button -->

                            <button type="submit" class="btn btn-primary btn-block mb-4">

                                Login

                            </button>

                        </form>

                    </div>

                </div>

            </div>

        </div>

    </div>

    <!-- Jumbotron -->

</section>

<!-- Section: Design Block -->

Login.component.ts

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

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

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

@Component({

  selector: 'app-login',

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

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

})

export class LoginComponent implements OnInit {

  showme: boolean = false;

  loginForm!: FormGroup;

  submitted = false;

  constructor(private frombulider: FormBuilder, private router: Router) { }

  ngOnInit() {

    this.loginForm = this.frombulider.group({

      email: ['',[Validators.required,Validators.email,Validators.pattern('Admin@gmail.com')]],

      password: ['',[Validators.required, Validators.pattern('Pass@1234')]],

    });

  }

  onSubmit() {

    this.submitted = true;

    if (this.loginForm.invalid) {

      return;

    }

    this.router.navigate([`/dashboard`]);

  }

  get f() {

    return this.loginForm.controls;

  }

  hideComponent() {

    this.showme = !this.showme;

  }

}

Employee.model.ts

export class EmployeeModel {

    id: number = 0;

    firstName: string = '';

    LastName: string = '';

    email: string = '';

  }

/\*

This class will be used as object to create new employee.

\*/

Api.service.ts

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

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

import { map } from 'rxjs';

@Injectable({

  providedIn: 'root'

})

export class ApiService {

  //'http://localhost:3000/posts' = API URL

  constructor(private http: HttpClient) {}

  // Creating New Employee

  postEmployee(data: any) {

    return this.http.post<any>('http://localhost:3000/posts', data).pipe(

      map((res: any) => {

        return res;

      })

    );

  }

//Display All Employee

  getEmployee() {

    return this.http.get<any>('http://localhost:3000/posts').pipe(

      map((res: any) => {

        return res;

      })

    );

  }

//Update Employee With Id

  updateEmployee(data: any, id: number) {

    return this.http.put<any>('http://localhost:3000/posts/' + id, data).pipe(

      map((res: any) => {

        return res;

      })

    );

  }

//Delete Employee With Id

  deleteEmployee(id: number) {

    return this.http.delete<any>('http://localhost:3000/posts/' + id).pipe(

      map((res: any) => {

        return res;

      })

    );

  }

}

Dashboard.component.html

<!-- Navbar -->

<nav class="navbar navbar-dark bg-dark" style="background-color: #4c0bce !important">

    <div class="container-fluid">

        <a class="navbar-brand" href="#">Employee Management Dashboard</a>

        <div class="d-flex">

            <button type="button" data-bs-toggle="modal" data-bs-target="#exampleModal" class="btn btn-success" (click)="clickAddEmployee()">

        Add employee

      </button>

        </div>

    </div>

</nav>

<table class="table mt-3">

    <thead>

        <tr>

            <th scope="col">ID</th>

            <th scope="col">First Name</th>

            <th scope="col">Last Name</th>

            <th scope="col">Email</th>

            <th scope="col">Action</th>

        </tr>

    </thead>

    <tbody>

        <tr \*ngFor="let row of employeeData">

            <td>{{ row.id }}</td>

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

            <td>{{ row.LastName }}</td>

            <td>{{ row.email }}</td>

            <td>

                <button (click)="onEdit(row)" style="color: aliceblue" class="btn btn-warning" data-bs-toggle="modal" data-bs-target="#exampleModal">

          Edit

        </button>

                <button (click)="deleteEmployee(row)" class="btn btn-danger mx-3">

          Delete

        </button>

            </td>

        </tr>

    </tbody>

</table>

<!--Add OR Update Employee --><!--Used Modal Display out Employee Form-->

<div class="modal fade" id="exampleModal" tabindex="-1" aria-labelledby="exampleModalLabel" aria-hidden="true">

    <div class="modal-dialog">

        <div class="modal-content">

            <div class="modal-header">

                <h5 class="modal-title" id="exampleModalLabel">Add OR Update Employee</h5>

                <button type="button" class="btn-close" data-bs-dismiss="modal" aria-label="Close"></button>

            </div>

            <div class="modal-body">

                <form [formGroup]="fromValue">

                    <div class="mb-3">

                        <label for="exampleInputEmail1" class="form-label">First Name</label

            >

            <input

              type="text"

              class="form-control"

              formControlName="firstname"

              id="exampleInputEmail1"

              aria-describedby="emailHelp"

            />

          </div>

          <div class="mb-3">

            <label for="exampleInputPassword1" class="form-label"

              >Last name</label

            >

            <input

              type="Lastname"

              class="form-control"

              formControlName="lastname"

              id="exampleInputPassword1"

            />

          </div>

          <div class="mb-3">

            <label for="exampleInputEmail1" class="form-label">Email</label>

                        <input type="email" class="form-control" formControlName="email" id="exampleInputEmail1" aria-describedby="emailHelp" />

                    </div>

                </form>

            </div>

            <div class="modal-footer">

                <button type="button" id="cancel" class="btn btn-secondary" data-bs-dismiss="modal">

          Close

        </button>

                <button type="button" class="btn btn-primary" \*ngIf="showAdd" (click)="postEmployeeDetails()">

          Add

        </button>

                <button type="button" class="btn btn-warning" \*ngIf="showUpdate" (click)="updateEmployeeDetails()">

          Update

        </button>

            </div>

        </div>

    </div>

</div>

Dashboard.component.ts

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

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

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

import { ApiService } from '../service/api.service';

@Component({

  selector: 'app-dashboard',

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

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

})

export class DashboardComponent implements OnInit {

  fromValue!: FormGroup;

  employeemodelobj: EmployeeModel = new EmployeeModel();

  employeeData!: any;

  showAdd!: boolean;

  showUpdate!: boolean;

  constructor(private formbuilder: FormBuilder, private api: ApiService) { }

  ngOnInit(): void {

    this.fromValue = this.formbuilder.group({

      firstname: [''],

      lastname: [''],

      email: [''],

    });

    this.getAllEmployee();

  }

//submit employee form after fillig or updating

  postEmployeeDetails() {

    this.employeemodelobj.firstName = this.fromValue.value.firstname;

    this.employeemodelobj.LastName = this.fromValue.value.lastname;

    this.employeemodelobj.email = this.fromValue.value.email;

    this.api.postEmployee(this.employeemodelobj).subscribe(

      (res) => {

        console.log(res);

        alert('Employee added Successfully');

        this.fromValue.reset();

        let ref = document.getElementById('cancel');

        ref?.click();

        this.getAllEmployee();

      },

      (err) => {

        alert('something worng happen');

      }

    );

  }

//dispaly all employee

  getAllEmployee() {

    this.api.getEmployee().subscribe((res) => {

      this.employeeData = res;

    });

  }

//delete an employee

  deleteEmployee(row: any) {

    this.api.deleteEmployee(row.id).subscribe((res) => {

      alert('Employee Deleted Sucessfully');

      this.getAllEmployee();

    });

  }

  onEdit(row: any) {

    this.employeemodelobj.id = row.id;

    this.fromValue.controls['firstname'].setValue(row.firstName);

    this.fromValue.controls['lastname'].setValue(row.LastName);

    this.fromValue.controls['email'].setValue(row.email);

    this.showAdd = false;

    this.showUpdate = true;

  }

  //update employee details

  updateEmployeeDetails() {

    this.employeemodelobj.firstName = this.fromValue.value.firstname;

    this.employeemodelobj.LastName = this.fromValue.value.lastname;

    this.employeemodelobj.email = this.fromValue.value.email;

    this.api

      .updateEmployee(this.employeemodelobj, this.employeemodelobj.id)

      .subscribe((res) => {

        alert('update done');

        let ref = document.getElementById('cancel');

        ref?.click();

        this.getAllEmployee();

      });

  }

  clickAddEmployee() {

    this.fromValue.reset();

    this.showAdd = true;

    this.showUpdate = false;

  }

}