**Source code for Event Management App:**

**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">

  <link

      rel="stylesheet"

      href="https://cdn.jsdelivr.net/npm/bootstrap@4.6.2/dist/css/bootstrap.min.css"

      integrity="sha384-xOolHFLEh07PJGoPkLv1IbcEPTNtaed2xpHsD9ESMhqIYd0nLMwNLD69Npy4HI+N"

      crossorigin="anonymous"

    />

</head>

<body>

  <app-root></app-root>

  <script

      src="https://cdn.jsdelivr.net/npm/jquery@3.5.1/dist/jquery.slim.min.js"

      integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj"

      crossorigin="anonymous"

    ></script>

    <script

      src="https://cdn.jsdelivr.net/npm/bootstrap@4.6.2/dist/js/bootstrap.bundle.min.js"

      integrity="sha384-Fy6S3B9q64WdZWQUiU+q4/2Lc9npb8tCaSX9FK7E8HnRr0Jz8D6OP9dO5Vg3Q9ct"

      crossorigin="anonymous"

    ></script>

</body>

</html>

**Employee-dashboard.component.html:**

<nav class="navbar navbar-light bg-info">

    <div class="container-fluid">

      <h1 style="color: white">Employee Admin Panel</h1>

      <div class="d-fles">

        <button

        (click)="clickAddEmployee()"

          type="button"

          data-toggle="modal"

          data-target="#exampleModal"

          class="btn btn-success"

        >

          Add Employee

        </button>

      </div>

    </div>

  </nav>

  <table class="table mt-3">

    <thead>

      <tr>

        <th scope="col">Employee Id</th>

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

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

        <th scope="col">Email Id</th>

        <th scope="col">Mobile No.</th>

        <th scope="col">Salary</th>

        <th scope="col">Action</th>

      </tr>

    </thead>

    <tbody>

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

        <td>{{ temp.id }}</td>

        <td>{{ temp.firstName }}</td>

        <td>{{ temp.lastName }}</td>

        <td>{{ temp.email }}</td>

        <td>{{ temp.mobile }}</td>

        <td>{{ temp.salary }}</td>

        <td>

          <button

            type="button"

            data-toggle="modal"

            data-target="#exampleModal"

            class="btn btn-info"

            (click)="onEdit(temp)"

          >

            Edit

          </button>

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

            Delete

          </button>

        </td>

      </tr>

    </tbody>

  </table>

  <!-- Button trigger modal -->

  <!-- Modal -->

  <div

    class="modal fade"

    id="exampleModal"

    tabindex="-1"

    role="dialog"

    aria-labelledby="exampleModalLabel"

    aria-hidden="true"

  >

    <div class="modal-dialog" role="document">

      <div class="modal-content">

        <div class="modal-header">

          <h5 class="modal-title" id="exampleModalLabel">Employee Details</h5>

          <button

            type="button"

            class="close"

            data-dismiss="modal"

            aria-label="Close"

          >

            <span aria-hidden="true">&times;</span>

          </button>

        </div>

        <div class="modal-body">

          <form [formGroup]="formValue">

            <div class="form-group">

              <label for="exampleInputEmail1">First Name</label>

              <input

                type="text"

                class="form-control"

                id="exampleInputEmail1"

                aria-describedby="emailHelp"

                formControlName="firstName"

              />

            </div>

            <div class="form-group">

              <label for="exampleInputPassword1">Last Name</label>

              <input

                type="text"

                class="form-control"

                id="exampleInputPassword1"

                formControlName="lastName"

              />

            </div>

            <div class="form-group">

              <label for="exampleInputPassword1">Email</label>

              <input

                type="email"

                class="form-control"

                id="exampleInputPassword1"

                formControlName="email"

              />

            </div>

            <div class="form-group">

              <label for="exampleInputPassword1">Mobile No.</label>

              <input

                type="text"

                class="form-control"

                id="exampleInputPassword1"

                formControlName="mobile"

              />

            </div>

            <div class="form-group">

              <label for="exampleInputPassword1">Salary</label>

              <input

                type="text"

                class="form-control"

                id="exampleInputPassword1"

                formControlName="salary"

              />

            </div>

          </form>

        </div>

        <div class="modal-footer">

          <button

            type="button"

            id="cancel"

            class="btn btn-secondary"

            data-dismiss="modal"

          >

            Close

          </button>

          <button

            type="button"

            (click)="postEmployeeDetails()"

            class="btn btn-primary"

            \*ngIf="showAdd"

          >

            Add

          </button>

          <button

            type="button"

            (click)="updateEmployee()"

            class="btn btn-primary"

            \*ngIf="showUpdate"

          >

            Update

          </button>

        </div>

      </div>

    </div>

  </div>

**Employee-dashboard.component.ts:**

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

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

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

import { EmployeeModel } from './employee-dash board.model';

@Component({

  selector: 'app-employee-dashboard',

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

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

})

export class EmployeeDashboardComponent implements OnInit {

  formValue !: FormGroup;

  employeeModelObj : EmployeeModel = new EmployeeModel();

  employeeData : any;

  showAdd !: boolean;

  showUpdate !: boolean;

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

  ngOnInit(): void {

    this.formValue = this.formbuiler.group({

      firstName : [''],

      lastName : [''],

      email : [''],

      mobile : [''],

      salary : ['']

    })

    this.getAllEmployee();

  }

  clickAddEmployee()

  {

    this.formValue.reset();

    this.showAdd= true;

    this.showUpdate= false;

  }

  postEmployeeDetails()

  {

    this.employeeModelObj.firstName = this.formValue.value.firstName;

    this.employeeModelObj.lastName = this.formValue.value.lastName;

    this.employeeModelObj.email = this.formValue.value.email;

    this.employeeModelObj.mobile = this.formValue.value.mobile;

    this.employeeModelObj.salary = this.formValue.value.salary;

    this.api.postEmployee(this.employeeModelObj)

    .subscribe(res=>{

      console.log(res);

      alert("employee Added")

      let ref = document.getElementById('cancel')

      ref?.click();

      this.formValue.reset();

      this.getAllEmployee();

    },

    err=>{

      alert("Something went wrong")

    })

  }

  getAllEmployee(){

    this.api.getEmployee()

    .subscribe(res=>{

      this.employeeData = res;

    })

  }

  deleteEmployee(temp : any){

    this.api.deleteEmployee(temp.id)

    .subscribe(res=>{

      alert("employee Deleted");

      this.getAllEmployee();

    })

  }

  onEdit(temp : any)

  {

    this.showAdd= false;

    this.showUpdate= true;

    this.employeeModelObj.id = temp.id;

    this.formValue.controls['firstName'].setValue(temp.firstName);

    this.formValue.controls['lastName'].setValue(temp.lastName);

    this.formValue.controls['email'].setValue(temp.email);

    this.formValue.controls['mobile'].setValue(temp.mobile);

    this.formValue.controls['salary'].setValue(temp.salary);

  }

  updateEmployee()

  {

    this.employeeModelObj.firstName = this.formValue.value.firstName;

    this.employeeModelObj.lastName = this.formValue.value.lastName;

    this.employeeModelObj.email = this.formValue.value.email;

    this.employeeModelObj.mobile = this.formValue.value.mobile;

    this.employeeModelObj.salary = this.formValue.value.salary;

    this.api.updateEmployee(this.employeeModelObj, this.employeeModelObj.id)

    .subscribe(res=>{

      alert("Updated Successfully")

      let ref = document.getElementById('cancel')

      ref?.click();

      this.formValue.reset();

      this.getAllEmployee();

    })

  }

}

**Signin.component.html:**

<div class="pos-f-t">

    <nav class="navbar navbar-dark bg-success">

      <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarToggleExternalContent" aria-controls="navbarToggleExternalContent" aria-expanded="false" aria-label="Toggle navigation">

        <span style="font-size: 40px ;">Event Management App</span>

      </button>

    </nav>

  </div>

  <div class="container mt-5 w-50">

    <form [formGroup]="registerForm" (ngSubmit)="OnSubmit()">

      <div class="form-group pt-5">

        <label for="exampleInputEmail1"><i class="fa-solid fa-envelope"></i> Email</label>

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

        <small id="emailHelp" class="form-text text-muted">We'll never share your email with anyone else.</small>

      </div>

      <div class="form-group mt-5">

        <label for="exampleInputPassword1"><i class="fa-sharp fa-solid fa-key"></i> Password </label>

        <input type="password" formControlName="upassword" class="form-control" id="exampleInputPassword1" placeholder="Password">

      </div >

      <div class="text-center">

        <button type="submit" class="btn btn-primary ">Submit</button>

      </div>

        </form>

  </div>

**Signin.component.ts:**

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

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

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

@Component({

  selector: 'app-signin',

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

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

})

export class SigninComponent implements OnInit {

  registerForm!: FormGroup;

  uemail: string = '';

  upassword: string = '';

  submitted: boolean = false;

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

  ngOnInit(): void {

    this.registerForm = this.builder.group({

      uemail: [''],

      upassword: [''],

    });

  }

  OnSubmit() {

    this.submitted = true;

    this.uemail = this.registerForm.get('uemail')?.value;

    this.upassword = this.registerForm.get('upassword')?.value;

    if (this.uemail == 'master@gmail.com' && this.upassword == '123') {

     // alert("Form Submitted Successfully"+this.upassword);

      this.router.navigate(['dashboard']);

    } else {

      alert('wrong credential');

    }

  }

}

**Db.json:**

{

  "posts": [],

  "comments": [],

  "profile": {

    "name": "typicode"

  }

}

**Angular.json:**

{

  "$schema": "./node\_modules/@angular/cli/lib/config/schema.json",

  "version": 1,

  "newProjectRoot": "projects",

  "projects": {

    "EventManagement": {

      "projectType": "application",

      "schematics": {},

      "root": "",

      "sourceRoot": "src",

      "prefix": "app",

      "architect": {

        "build": {

          "builder": "@angular-devkit/build-angular:browser",

          "options": {

            "outputPath": "dist/event-management",

            "index": "src/index.html",

            "main": "src/main.ts",

            "polyfills": "src/polyfills.ts",

            "tsConfig": "tsconfig.app.json",

            "assets": [

              "src/favicon.ico",

              "src/assets"

            ],

            "styles": [

              "src/styles.css"

            ],

            "scripts": []

          },

          "configurations": {

            "production": {

              "budgets": [

                {

                  "type": "initial",

                  "maximumWarning": "500kb",

                  "maximumError": "1mb"

                },

                {

                  "type": "anyComponentStyle",

                  "maximumWarning": "2kb",

                  "maximumError": "4kb"

                }

              ],

              "fileReplacements": [

                {

                  "replace": "src/environments/environment.ts",

                  "with": "src/environments/environment.prod.ts"

                }

              ],

              "outputHashing": "all"

            },

            "development": {

              "buildOptimizer": false,

              "optimization": false,

              "vendorChunk": true,

              "extractLicenses": false,

              "sourceMap": true,

              "namedChunks": true

            }

          },

          "defaultConfiguration": "production"

        },

        "serve": {

          "builder": "@angular-devkit/build-angular:dev-server",

          "configurations": {

            "production": {

              "browserTarget": "EventManagement:build:production"

            },

            "development": {

              "browserTarget": "EventManagement:build:development"

            }

          },

          "defaultConfiguration": "development"

        },

        "extract-i18n": {

          "builder": "@angular-devkit/build-angular:extract-i18n",

          "options": {

            "browserTarget": "EventManagement:build"

          }

        },

        "test": {

          "builder": "@angular-devkit/build-angular:karma",

          "options": {

            "main": "src/test.ts",

            "polyfills": "src/polyfills.ts",

            "tsConfig": "tsconfig.spec.json",

            "karmaConfig": "karma.conf.js",

            "assets": [

              "src/favicon.ico",

              "src/assets"

            ],

            "styles": [

              "src/styles.css"

            ],

            "scripts": []

          }

        }

      }

    }

  },

  "cli": {

    "analytics": "cd4b0b75-76b6-4894-9a78-9d7b59d9854d"

  }

}

**Signin.component.spec.ts:**

import { ComponentFixture, TestBed } from '@angular/core/testing';

import { EmployeeDashboardComponent } from './employee-dashboard.component';

describe('EmployeeDashboardComponent', () => {

  let component: EmployeeDashboardComponent;

  let fixture: ComponentFixture<EmployeeDashboardComponent>;

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      declarations: [ EmployeeDashboardComponent ]

    })

    .compileComponents();

    fixture = TestBed.createComponent(EmployeeDashboardComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create', () => {

    expect(component).toBeTruthy();

  });

});

**App.component.spec.ts:**

import { TestBed } from '@angular/core/testing';

import { RouterTestingModule } from '@angular/router/testing';

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

describe('AppComponent', () => {

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      imports: [

        RouterTestingModule

      ],

      declarations: [

        AppComponent

      ],

    }).compileComponents();

  });

  it('should create the app', () => {

    const fixture = TestBed.createComponent(AppComponent);

    const app = fixture.componentInstance;

    expect(app).toBeTruthy();

  });

  it(`should have as title 'EventManagement'`, () => {

    const fixture = TestBed.createComponent(AppComponent);

    const app = fixture.componentInstance;

    expect(app.title).toEqual('EventManagement');

  });

  it('should render title', () => {

    const fixture = TestBed.createComponent(AppComponent);

    fixture.detectChanges();

    const compiled = fixture.nativeElement as HTMLElement;

    expect(compiled.querySelector('.content span')?.textContent).toContain('EventManagement app is running!');

  });

});

**App.component.ts:**

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

@Component({

  selector: 'app-root',

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

  styleUrls: ['./app.component.css']

})

export class AppComponent {

  title = 'EventManagement';

}

**Employee-dashboard.component.spec.ts:**

import { ComponentFixture, TestBed } from '@angular/core/testing';

import { EmployeeDashboardComponent } from './employee-dashboard.component';

describe('EmployeeDashboardComponent', () => {

  let component: EmployeeDashboardComponent;

  let fixture: ComponentFixture<EmployeeDashboardComponent>;

  beforeEach(async () => {

    await TestBed.configureTestingModule({

      declarations: [ EmployeeDashboardComponent ]

    })

    .compileComponents();

    fixture = TestBed.createComponent(EmployeeDashboardComponent);

    component = fixture.componentInstance;

    fixture.detectChanges();

  });

  it('should create', () => {

    expect(component).toBeTruthy();

  });

});