Source code for EventManagementApp:

1. **Admin.component.html:**

```html
 <div class = "row" style="padding-top: 30px;">
  <div class="col">
    <h2> Employee List</h2>
  </div>
  <div class="col">
    <button class="btn btn-success btn-sm float-right" (click) = "add()">
      <a routerLink="user" routerLinkActive="active" class="nav-link" style="color: white;" >Add Employee</a>
    </button>
  </div>

 </div>

 <table class = "table table-striped table-bordered text-center">
  <thead>
      <tr>
          <th>Id</th>
          <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.id }} </td>
          <td> {{ employee.firstName }} </td>
          <td> {{ employee.lastName }} </td>
          <td> {{ employee.emailId }} </td>
          <td>
              <button (click) = "employeeDetails(employee.id)" class = "btn btn-primary btn-sm" style="margin-left: 10px"> View Details</button>
          </td>
      </tr>
  </tbody>
 </table>
```

2. **admin.component.ts:**

```typescript
import { Component, OnInit } from '@angular/core';
import { CrudService } from '../crud.service';
import { Employee } from '../Employee';
import { Router } from '@angular/router';
@Component({
  selector: 'app-admin',
  templateUrl: './admin.component.html',
  styleUrls: ['./admin.component.css']
})
export class AdminComponent implements OnInit {

  employees:Employee[];
  constructor(private crudService:CrudService ,private router:Router ) { }
ngOnInit(): void {
  this.getEmployees();
}

private getEmployees(){
  this.crudService.getEmployeesList().subscribe(data => {
    this.employees = data;
  });
}

add(){
  this.router.navigate(['user']);
}
employeeDetails(id: number){
  this.router.navigate(['details', id]);
}

updateEmployee(id: number){
  this.router.navigate(['edit', id]);
}

deleteEmployee(id: number){
  this.crudService.deleteEmployee(id).subscribe( data => {
    console.log(data);
    this.getEmployees();
  })
}
}
```

3. **user.component.html:**

```
<h3 class="text-center" style="padding-top: 30px;"> Create Employee </h3>

<div class="row">
 <div class="card col-md-6 offset-md-3 offset-md-3">
     <div class="row">
         <hr />
         <div class="card-body">
             <form [formGroup]="registerForm" (ngSubmit)="onSubmit()">

                 <div class="form-group">
                     <label> First Name</label>
                     <input type="text" class="form-control" id="firstName" [(ngModel)]="employee.firstName"
                         name="firstName" formControlName="firstName" [ngClass]="{'is-invalid': submitted && f['firstName'].errors}">
                         <div *ngIf="submitted && f['firstName'].errors">
                           <div *ngIf="submitted && f['firstName'].errors['required']">
                               FirstName  is required
                           </div>
                       </div>
                       </div>
```

```
            <div class="form-group">
              <label> Last Name</label>
              <input type="text" class="form-control" id="lastName" [(ngModel)]="employee.lastName"
                name="lastName" formControlName="lastName" [ngClass]="{'is-invalid': submitted && f['lastName'].errors}">
                <div *ngIf="submitted && f['lastName'].errors">
                 <div *ngIf="submitted && f['lastName'].errors['required']">
                   LastName  is required
                 </div>
               </div>
            </div>

            <div class="form-group">
              <label> Email Id</label>
              <input type="email" class="form-control" id="emailId" [(ngModel)]="employee.emailId"
                name="emailId" formControlName="emailId" [ngClass]="{'is-invalid': submitted && f['emailId'].errors}">
                <div *ngIf="submitted && f['emailId'].errors">
                 <div *ngIf="submitted && f['emailId'].errors['required']">
                   Email  is required
                 </div>
               </div>
               </div>

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

          </form>
        </div>
      </div>
  </div>
</div>
```

4. user.component.ts:

```
<h3 class="text-center" style="padding-top: 30px;"> Create Employee </h3>

<div class="row">
  <div class="card col-md-6 offset-md-3 offset-md-3">
      <div class="row">
        <hr />
        <div class="card-body">
          <form [formGroup]="registerForm" (ngSubmit)="onSubmit()">

            <div class="form-group">
              <label> First Name</label>
              <input type="text" class="form-control" id="firstName" [(ngModel)]="employee.firstName"
                name="firstName" formControlName="firstName" [ngClass]="{'is-invalid': submitted && f['firstName'].errors}">
                <div *ngIf="submitted && f['firstName'].errors">
                 <div *ngIf="submitted && f['firstName'].errors['required']">
                   FirstName  is required
                 </div>
            </div>
            </div>

         <div class="form-group">
            <label> Last Name</label>
            <input type="text" class="form-control" id="lastName" [(ngModel)]="employee.lastName"
              name="lastName" formControlName="lastName" [ngClass]="{'is-invalid': submitted && f['lastName'].errors}">
              <div *ngIf="submitted && f['lastName'].errors">
                <div *ngIf="submitted && f['lastName'].errors['required']">
                  LastName  is required
                </div>
            </div>
         </div>

         <div class="form-group">
            <label> Email Id</label>
            <input type="email" class="form-control" id="emailId" [(ngModel)]="employee.emailId"
              name="emailId" formControlName="emailId" [ngClass]="{'is-invalid': submitted && f['emailId'].errors}">
              <div *ngIf="submitted && f['emailId'].errors">
                <div *ngIf="submitted && f['emailId'].errors['required']">
                  Email  is required
                </div>
            </div>
            </div>

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

       </form>
     </div>
   </div>
 </div>
</div>
```

5. **signin.component.html:**

```
 <div class="container-fluid" style="padding-top: 40px;">
   <div class="container">

      <div class="row">
         <div class="col-md-6 offset-md-3 ">

            <form [formGroup]="registerForm" (ngSubmit)="OnSubmit()">
               <h2 style="text-align:center">Sign in</h2>
               <!--email-->
               <div class="form-group">
                  <label>Email</label>
                  <input type="email" formControlName="email" class="form-control" [ngClass]="{'is-invalid': submitted &&
f['email'].errors}"
                        placeholder="Email" id="email" name="email"/>
                  <!--Custom Validations-->
```

```
                    <div *ngIf="submitted && f['email'].errors">
                        <div *ngIf="submitted && f['email'].errors['required']">
                            Email  is required
                        </div>
                    </div>
                </div>

                <!--Password-->

                <div class="form-group">
                    <label>Password</label>
                    <input type="password" formControlName="password" class="form-control" [ngClass]="{'is-invalid': submitted &&
f['password'].errors}"
                            placeholder="Password" name="password" id="password"/>
                    <!--Custom Validations-->
                    <div *ngIf="submitted && f['password'].errors">
                        <div *ngIf="submitted && f['password'].errors['required']">
                            Password must be minimum 5 character long
                        </div>
                    </div>
                </div>

                <div class="form-group text-center">
                    <button class="btn btn-primary">Login</button>
                </div>
            </form>
        </div>
    </div>
  </div>
</div>
```

6. **signin.component.ts:**

```
import { Component, OnInit } from '@angular/core';
import { FormGroup,FormBuilder,Validators } 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;
  submitted:boolean=false;
  constructor(private builder:FormBuilder,private router:Router) { }

  ngOnInit(): void {
    this.registerForm= this.builder.group(
      {
        email:["",[Validators.required,Validators.email]],
        password:["",[Validators.required,Validators.minLength(5)]]
      }
    );
  }
  OnSubmit(){
    this.submitted=true;
    if(this.registerForm.invalid)
      return;
    else
    {
    const email:any = document.getElementById('email') as HTMLInputElement | null;
    let password:any = document.getElementById('password') as HTMLInputElement | null;
    if(email.value=="vamsi@gmail.com" && password.value=="vamsi")
    {
      this.router.navigate(['/admin']);
    }
    else{
      alert("Wrong email or password");
    this.router.navigate(['/signin']);
    }
    }
  }

  get f(){
    return this.registerForm.controls;
  }
  check(){

  }
}
```

7. **edit.component.html:**

```
<h3 style="text-align: center;padding-top: 30px;"> Update Employee </h3>
<div class="row">
   <div class="card col-md-6 offset-md-3 offset-md-3">
      <div class="row">



         <div class="card-body">
            <form [formGroup]="registerForm" (ngSubmit)="onSubmit()">

               <div class="form-group">
                  <label> First Name</label>
                  <input type="text" class="form-control" id="firstName" [(ngModel)]="employee.firstName"
                     name="firstName"  formControlName="firstName" [ngClass]="{'is-invalid': submitted && f['firstName'].errors}">
                     <div *ngIf="submitted && f['firstName'].errors">
                        <div *ngIf="submitted && f['firstName'].errors['required']">
                           FirstName  is required
                        </div>
```

```
              </div>
          </div>

          <div class="form-group">
            <label> Last Name</label>
            <input type="text" class="form-control" id="lastName" [(ngModel)]="employee.lastName"
              name="lastName" formControlName="lastName" [ngClass]="{'is-invalid': submitted && f['lastName'].errors}">
              <div *ngIf="submitted && f['lastName'].errors">
                <div *ngIf="submitted && f['lastName'].errors['required']">
                  LastName  is required
                </div>
              </div>
          </div>

          <div class="form-group">
            <label> Email Id</label>
            <input type="email" class="form-control" id="emailId" [(ngModel)]="employee.emailId"
              name="emailId" formControlName="emailId" [ngClass]="{'is-invalid': submitted && f['emailId'].errors}">
              <div *ngIf="submitted && f['emailId'].errors">
                <div *ngIf="submitted && f['emailId'].errors['required']">
                  Email  is required
                </div>
              </div>
          </div>
          <button class="btn btn-success" type="submit">Submit</button>

        </form>
      </div>
    </div>
  </div>
</div>
```

8. edit.component.ts:

```
import { Component, OnInit } from '@angular/core';
import { CrudService } from '../crud.service';
import { ActivatedRoute, Router } from '@angular/router';
import { Employee} from '../Employee';
import { FormGroup,FormBuilder,Validators } from '@angular/forms';

@Component({
  selector: 'app-edit',
  templateUrl: './edit.component.html',
  styleUrls: ['./edit.component.css']
})
export class EditComponent implements OnInit {


  registerForm: FormGroup;
  submitted:boolean=false;
  id:number;
employee: Employee = new Employee();

constructor(private crudService: CrudService,private router:Router,
            private route:ActivatedRoute,private builder:FormBuilder) { }

ngOnInit(): void {
  this.id = this.route.snapshot.params['id'];
  this.crudService.getEmployeeById(this.id).subscribe(data => {
    this.employee = data;
  }, error => console.log(error));

  this.registerForm= this.builder.group(
    {
      firstName:["",Validators.required],
      lastName:["",Validators.required],
      emailId:["",[Validators.required,Validators.email]]
    }
  );
}

onSubmit(){
  this.submitted=true;
  if(this.registerForm.invalid)
    return;
  else{
    this.crudService.updateEmployee(this.id, this.employee).subscribe( data =>{
      this.goToEmployeeList();
    }
    , error => console.log(error));
  }
}
get f(){
  return this.registerForm.controls;
}

goToEmployeeList(){
  this.router.navigate(['/admin']);
}
}
```

9. **details.component.html:**

```
<h3 style="margin-top: 30px;">Employee Details</h3>
<div class="jumbotron">
    <div>
        <label> <b> Id: </b></label> {{employee.id}}
    </div>
    <div>
        <label> <b> First Name: </b></label> {{employee.firstName}}
    </div>
```

```
  <div>
    <label> <b> Last Name: </b></label> {{employee.lastName}}
  </div>
  <div>
    <label> <b> Email Id: </b></label> {{employee.emailId}}
  </div>
  <div>
    <button (click)="updateEmployee(employee.id)" class="btn btn-warning" style="margin-right: 3px;">Edit</button>
    <button (click)="deleteEmployee(employee.id)" class="btn btn-danger">Delete</button>
  </div>
</div>
<div style="margin-top:-25px ;">
  <button (click)="goBack()" class="btn btn-primary">Back</button>
</div>
```

10. **details.component.ts:**

```
import { Component, OnInit } from '@angular/core';
import { ActivatedRoute, Router } from '@angular/router';
import { CrudService } from '../crud.service';
import { Employee } from '../Employee';
@Component({
  selector: 'app-details',
  templateUrl: './details.component.html',
  styleUrls: ['./details.component.css']
})
export class DetailsComponent implements OnInit {
  id: number
  employee: Employee

  constructor(private route:ActivatedRoute, private crudService:CrudService,private router:Router) { }

  ngOnInit(): void {
    this.id = this.route.snapshot.params['id'];

    this.employee = new Employee();
    this.crudService.getEmployeeById(this.id).subscribe( data => {
      this.employee = data;
    });
  }
  goBack(){
    this.router.navigate(['admin']);
  }

  updateEmployee(id: number){
    this.router.navigate(['edit', id]);
  }

  deleteEmployee(id: number){
    this.crudService.deleteEmployee(id).subscribe( data => {
      alert("Employee data deleted");
      this.router.navigate(['admin']);
    })

}
}
```

11. app-routing.module.ts:

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { AdminComponent } from './admin/admin.component';
import { DetailsComponent } from './details/details.component';
import { EditComponent } from './edit/edit.component';
import { SigninComponent } from './signin/signin.component';
import { UserComponent } from './user/user.component';

const routes: Routes = [
  {path:"admin",component:AdminComponent},
  {path:"user",component:UserComponent},
  {path:"edit/:id",component:EditComponent},
  {path:'details/:id',component:DetailsComponent},
  {path:"signin",component:SigninComponent}
];


@NgModule({
  imports: [RouterModule.forRoot(routes)],
  exports: [RouterModule]
})
export class AppRoutingModule { }
```

12. app.component.html:

```
<nav class="navbar navbar-expand-sm bg-dark navbar-dark">
  <a href="#" class="navbar-brand">Event Management</a>
  <ul class = "navbar-nav">
      <li class="nav-item">
        <a routerLink="signin" class="nav-link">Signin</a>
      </li>
  </ul>
</nav>
<div class = "container">
  <router-outlet></router-outlet>
</div>
```

13. crud.service.ts:

```
import { Injectable } from '@angular/core';
import { HttpClient} from '@angular/common/http';
import { Observable} from 'rxjs';
import { Employee } from './Employee';
@Injectable({
  providedIn: 'root'
})
export class CrudService {

  constructor(private http: HttpClient) { }

  apiURL: string = 'http://localhost:3000/employees';

  getEmployeesList(): Observable<Employee[]>{
    return this.http.get<Employee[]>(`${this.apiURL}`);
  }

  createEmployee(employee: Employee): Observable<Object>{
    return this.http.post(`${this.apiURL}`, employee);
  }

  getEmployeeById(id: number): Observable<Employee>{
    return this.http.get<Employee>(`${this.apiURL}/${id}`);
  }

  updateEmployee(id: number, employee: Employee): Observable<Object>{
    return this.http.put(`${this.apiURL}/${id}`, employee);
  }

  deleteEmployee(id: number): Observable<Object>{
    return this.http.delete(`${this.apiURL}/${id}`);
  }
}
```

14. **data.json:**

```
{
  "employees": [
    {
      "id": 1,
      "firstName": "Vamsi",
      "lastName": "Pullagura",
      "emailId": "vamsipullagura@gmail.com"
    },
    {
      "id": 2,
      "firstName": "Sirisha",
      "lastName": "Pullagura",
      "emailId": "sirishapullagur@gmail.com"
    },
    {
      "firstName": "Chinna",
      "lastName": "Pullagura",
      "emailId": "chinna@gmail.com",
      "id": 3
    },
    {
      "firstName": "Priya",
      "lastName": "kamineni",
      "emailId": "priya@gmail.com",
      "id": 4
    },
    {
      "firstName": "Vyshanvi ",
      "lastName": "Danda",
      "emailId": "vyshnavi@gmail.com",
      "id": 6
    }
  ]
}
```

15. Employee.ts:

```
export class Employee{
  id:number;
  firstName:string;
  lastName:string;
  emailId:string;
}
```

16. 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>
```

17. **tsconfig.json:**

```
/* To learn more about this file see: https://angular.io/config/tsconfig. */
{
 "compileOnSave": false,
 "compilerOptions": {
  "baseUrl": "./",
  "outDir": "./dist/out-tsc",
  "forceConsistentCasingInFileNames": true,
  "strictPropertyInitialization": false,
  "strict": true,
  "noImplicitOverride": true,
  "noPropertyAccessFromIndexSignature": true,
  "noImplicitReturns": true,
  "noFallthroughCasesInSwitch": true,
  "sourceMap": true,
  "declaration": false,
  "downlevelIteration": true,
  "experimentalDecorators": true,
  "moduleResolution": "node",
  "importHelpers": true,
  "target": "es2020",
  "module": "es2020",
  "lib": [
   "es2020",
   "dom"
  ]
 },
 "angularCompilerOptions": {
  "enableI18nLegacyMessageIdFormat": false,
  "strictInjectionParameters": true,
  "strictInputAccessModifiers": true,
  "strictTemplates": true
 }
}
```