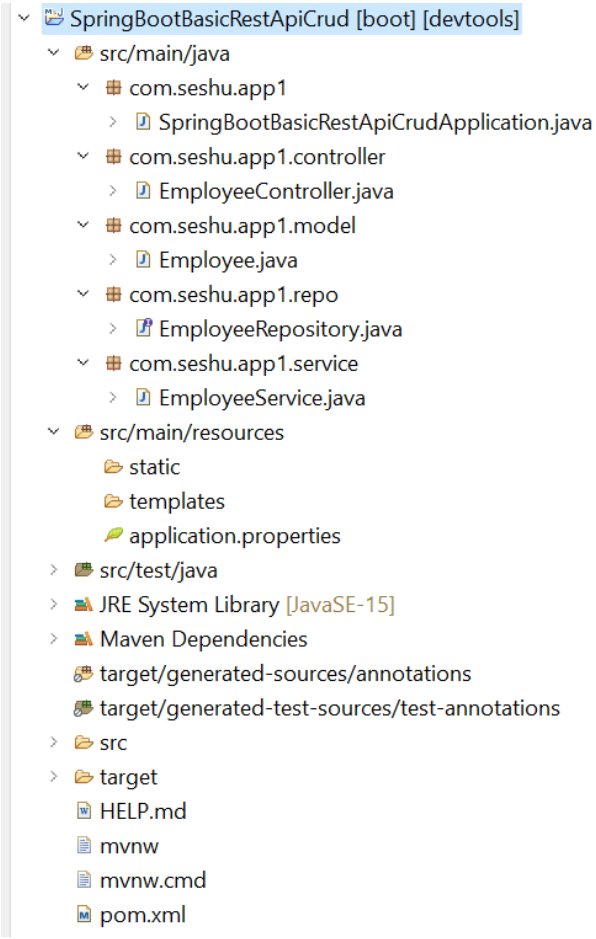
Angular and Spring Boot Rest API Crud Operation

Create Spring Starter Project as **SpringBootBasicRestApiCrud** with fallowing dependencies;

Spring Web,
Spring Boot DevTools,
Mysql Driver,
Spring Data JPA



application.properties

```
server.port=8181
server.servlet.context-path=/myapp

spring.datasource.url=jdbc:mysql://localhost:3306/adidb
spring.datasource.username=root
spring.datasource.password=seshu

spring.jpa.hibernate.ddl-auto=update
spring.jpa.show-sql=true
spring.jpa.database-platform=org.hibernate.dialect.MySQL8Dialect
```

Employee.java

```
package com.seshu.app1.model;

import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.Id;
import javax.persistence.Table;

@Entity
@Table(name = "emp001")
public class Employee {
	@Id
	@GeneratedValue
	private Integer employeeId;
	private String employeeName;
	private Double employeeSalary;
	private String employeeEmail;
	private Long employeeMobile;

	public Employee() {
		super();
	}

	public Employee(Integer employeeId, String employeeName, Double
employeeSalary, String employeeEmail,
			Long employeeMobile) {
		super();
		this.employeeId = employeeId;
		this.employeeName = employeeName;
		this.employeeSalary = employeeSalary;
		this.employeeEmail = employeeEmail;
		this.employeeMobile = employeeMobile;
	}

	public Integer getEmployeeId() {
		return employeeId;
	}

	public void setEmployeeId(Integer employeeId) {
		this.employeeId = employeeId;
	}
```

```
	public String getEmployeeName() {
		return employeeName;
	}

	public void setEmployeeName(String employeeName) {
		this.employeeName = employeeName;
	}

	public Double getEmployeeSalary() {
		return employeeSalary;
	}

	public void setEmployeeSalary(Double employeeSalary) {
		this.employeeSalary = employeeSalary;
	}

	public String getEmployeeEmail() {
		return employeeEmail;
	}

	public void setEmployeeEmail(String employeeEmail) {
		this.employeeEmail = employeeEmail;
	}

	public Long getEmployeeMobile() {
		return employeeMobile;
	}

	public void setEmployeeMobile(Long employeeMobile) {
		this.employeeMobile = employeeMobile;
	}

	@Override
	public String toString() {
		return "Employee [employeeId=" + employeeId + ", employeeName=" + 
employeeName + ", employeeSalary="
			+ employeeSalary + ", employeeEmail=" + employeeEmail + ", 
employeeMobile=" + employeeMobile + "]";
	}
}
```

EmployeeRepository.java

```
package com.seshu.app1.repo;

import org.springframework.data.jpa.repository.JpaRepository;

import com.seshu.app1.model.Employee;

public interface EmployeeRepository extends JpaRepository<Employee, Integer>{

}
```

EmployeeService.java

```
package com.seshu.app1.service;

import java.util.List;
import java.util.Optional;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

import com.seshu.app1.model.Employee;
import com.seshu.app1.repo.EmployeeRepository;

@Service
public class EmployeeService {
	@Autowired
	private EmployeeRepository repo;

	public Employee saveEmployee(Employee e) {
		return repo.save(e);
	}

	public List<Employee> getEmployees() {
		return repo.findAll();
	}

	public Optional<Employee> getEmployee(Integer id) {
		return repo.findById(id);
	}
```

```
	public void updateEmployee(Employee e) {
		repo.save(e);

	}

	public void deleteEmployee(Integer id) {
		repo.deleteById(id);
	}
}
```

EmployeeController.java

```
package com.seshu.app1.controller;

import java.util.List;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.http.HttpStatus;
import org.springframework.http.ResponseEntity;
import org.springframework.web.bind.annotation.CrossOrigin;
import org.springframework.web.bind.annotation.DeleteMapping;
import org.springframework.web.bind.annotation.GetMapping;
import org.springframework.web.bind.annotation.PathVariable;
import org.springframework.web.bind.annotation.PostMapping;
import org.springframework.web.bind.annotation.PutMapping;
import org.springframework.web.bind.annotation.RequestBody;
import org.springframework.web.bind.annotation.RequestMapping;
import org.springframework.web.bind.annotation.RestController;

import com.seshu.app1.model.Employee;
import com.seshu.app1.service.EmployeeService;

@CrossOrigin(origins = "http://localhost:4200")
@RestController
@RequestMapping("/api")
public class EmployeeController {
	@Autowired
	private EmployeeService service;

	@PostMapping("/employee")
	public ResponseEntity<Employee> save(@RequestBody Employee employee) {
		Employee newEmployee = service.saveEmployee(employee);
		return new ResponseEntity<>(newEmployee, HttpStatus.CREATED);
```

```
	}

	@GetMapping("/employee")
	public ResponseEntity<List<Employee>> getAllEmployees() {
		List<Employee> employeeList = service.getEmployees();
		return new ResponseEntity<>(employeeList, HttpStatus.OK);
	}

	@GetMapping("/employee/{id}")
	public ResponseEntity<?> getEmployeeById(@PathVariable Integer id) {
		Employee employee = service.getEmployee(id).get();
		return new ResponseEntity<Employee>(employee, HttpStatus.OK);
	}

	@PutMapping("/employee")
	public ResponseEntity<?> update(@RequestBody Employee employee){
		service.updateEmployee(employee);
		return new ResponseEntity<>(HttpStatus.OK);
	}

	@DeleteMapping("/employee/{id}")
	public ResponseEntity<?> delete(@PathVariable Integer id){
		service.deleteEmployee(id);
		return new ResponseEntity<>(HttpStatus.OK);
	}
}
```

SpringBootBasicRestApiCrudApplication.java

```
package com.seshu.app1;

import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

@SpringBootApplication
public class SpringBootBasicRestApiCrudApplication {

	public static void main(String[] args) {
		SpringApplication.run(SpringBootBasicRestApiCrudApplication.class,
args);
	}
}
```

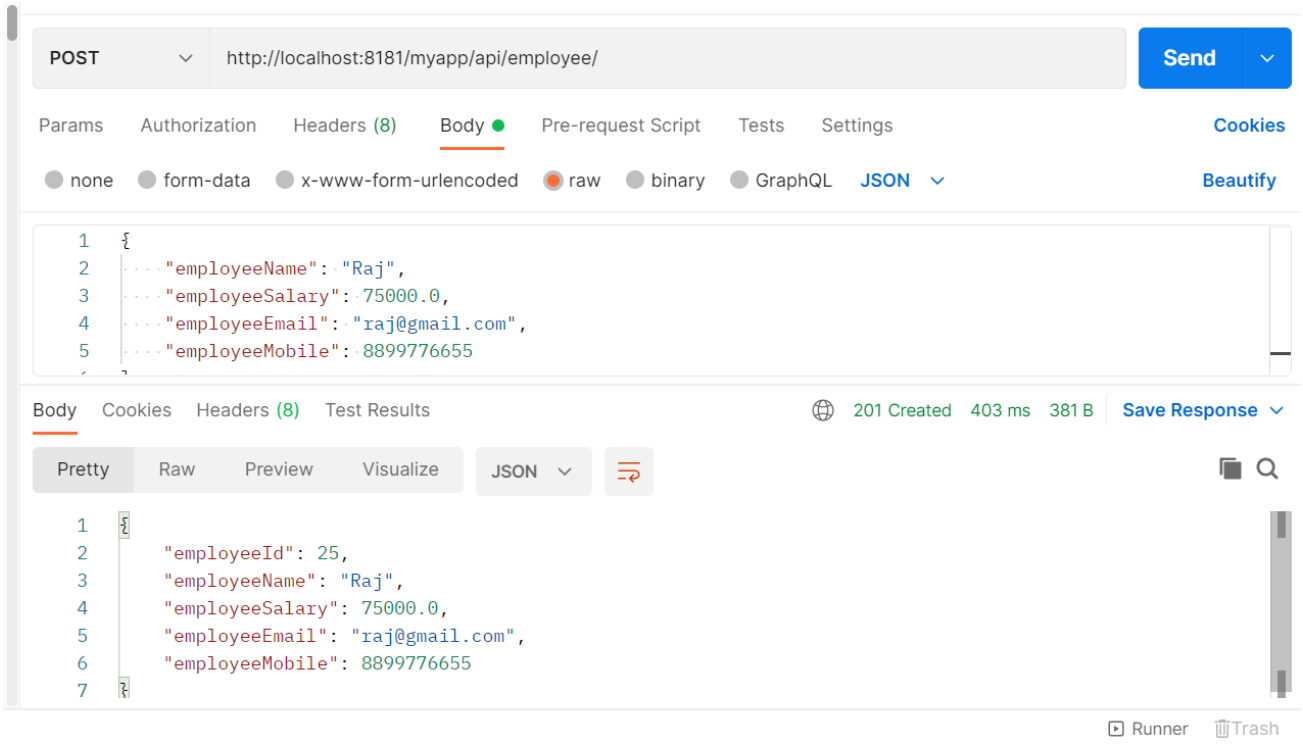
Right click on SpringBootBasicRestApiCrudApplication.java -> Run As -> Spring Boot App

Test the Employee API service using Postman.

POST Request;

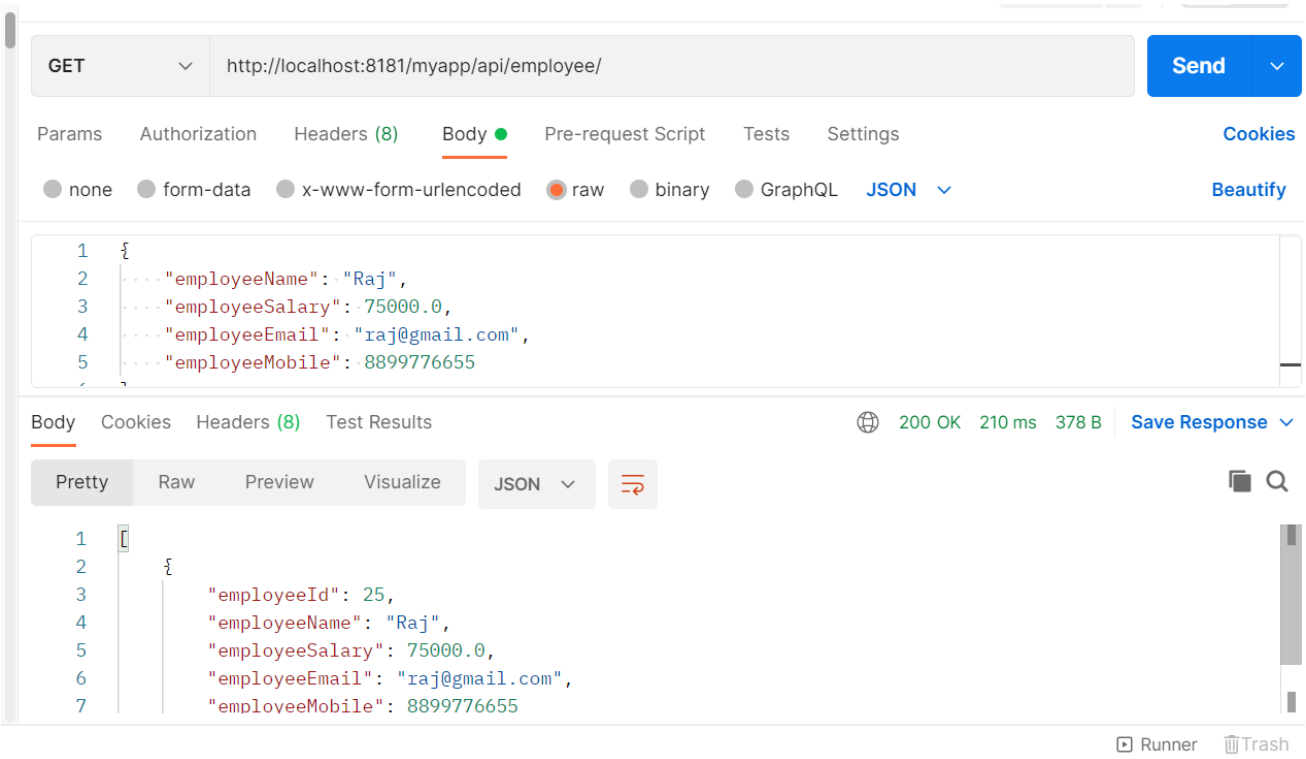
http://localhost:8181/myapp/api/employee/

```
{
   "employeeName": "Raj",
   "employeeSalary": 75000.0,
   "employeeEmail": "raj@gmail.com",
   "employeeMobile": 8899776655
}
```



GET REQUEST;

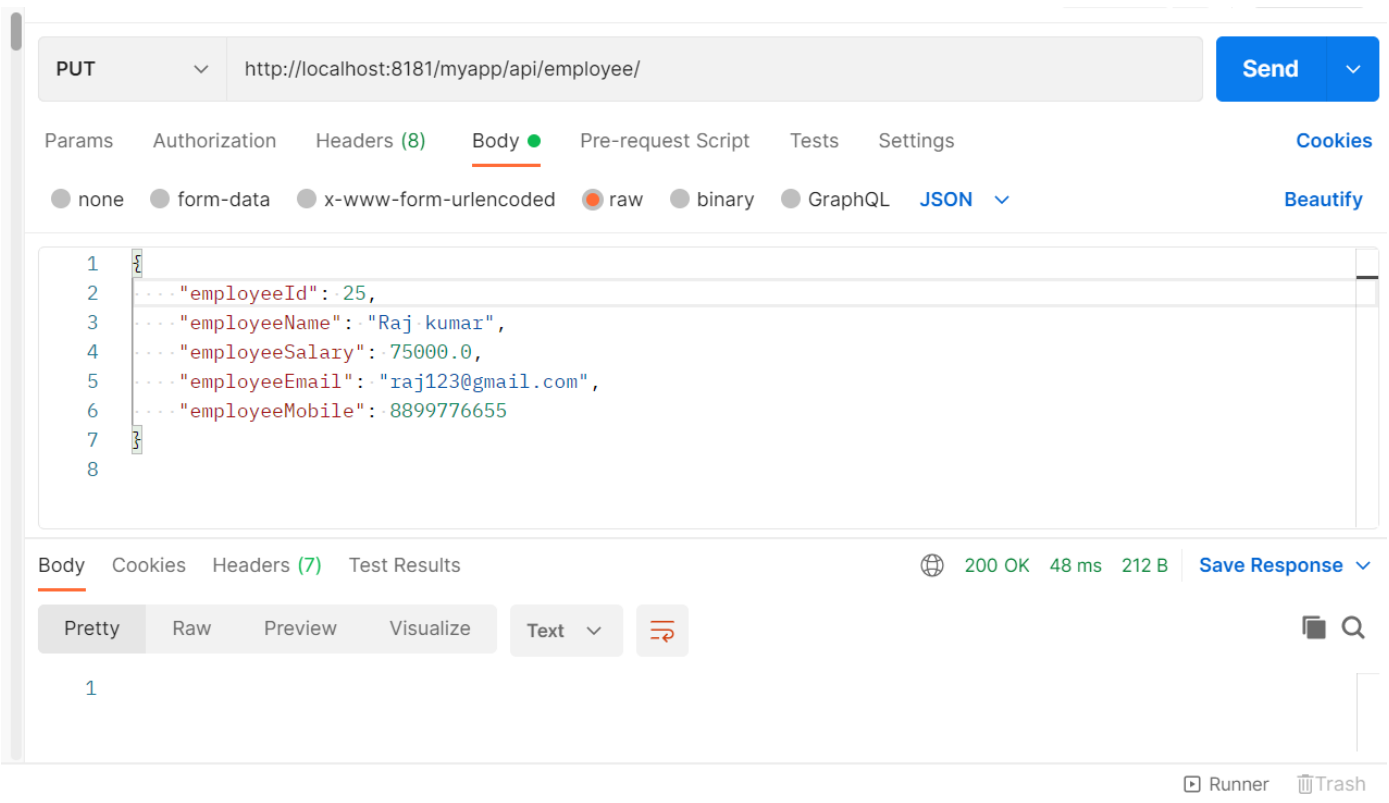
http://localhost:8181/myapp/api/employee/



PUT REQUEST;

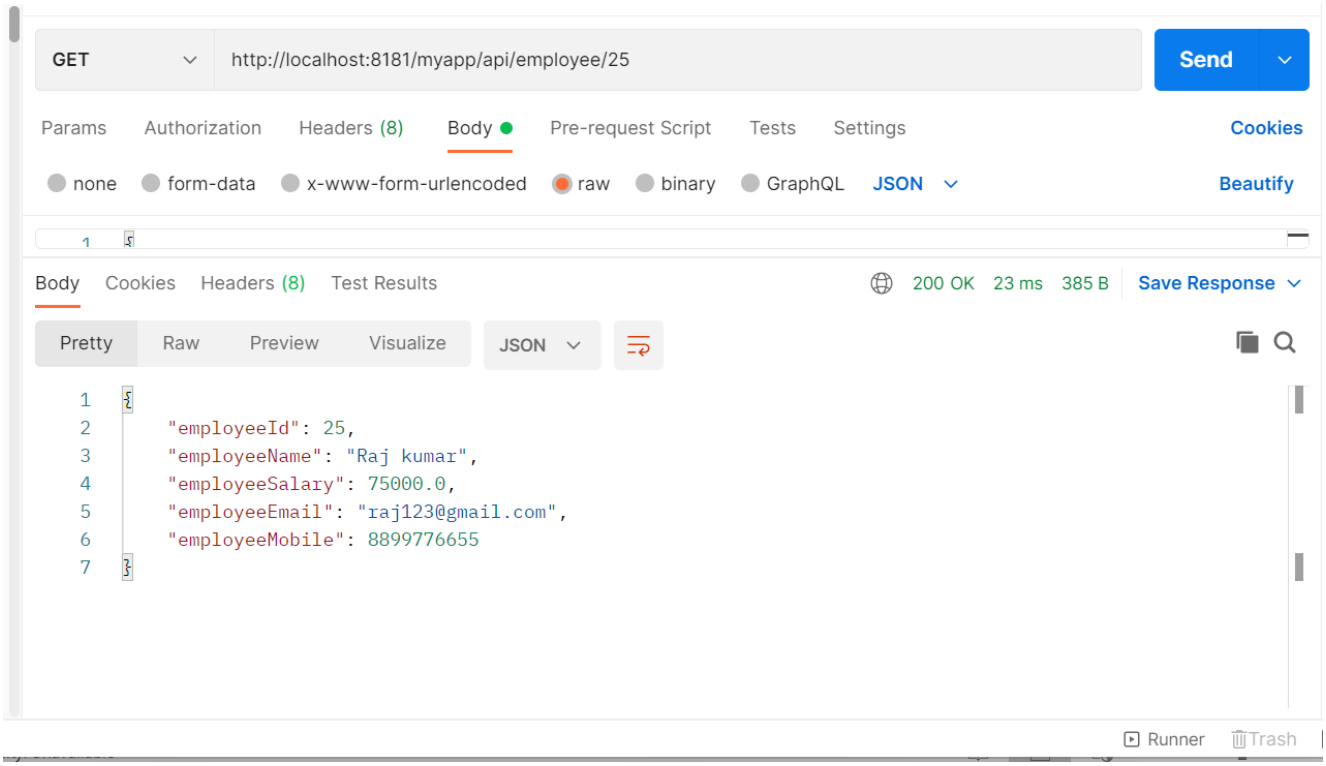
http://localhost:8181/myapp/api/employee/

```
{
   "employeeId": 25,
   "employeeName": "Raj kumar",
   "employeeSalary": 75000.0,
   "employeeEmail": "raj123@gmail.com",
   "employeeMobile": 8899776655
}
```



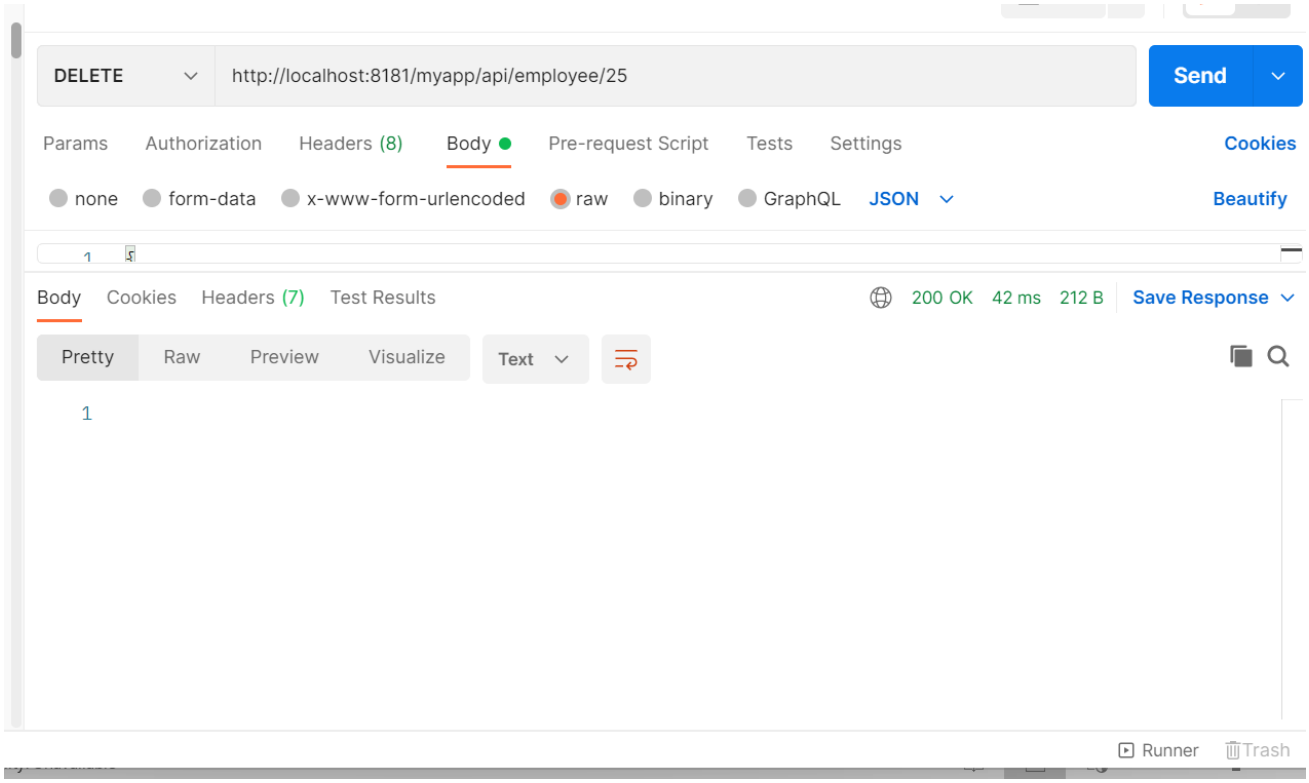
GET REQUEST;

http://localhost:8181/myapp/api/employee/25



DELETE REQUEST;

http://localhost:8181/myapp/api/employee/25



Handling Rest API in Angular

HttpClient:

- Most front-end applications need to communicate with a server over the **HTTP protocol**, in order to download or upload data and access other back-end services.

- Angular provides a simplified client HTTP API for Angular applications, the **HttpClient** service class in **@angular/common/http.**

- The **HttpClient** service calls the server side service over HTTP.
  The HTTP verb that is sent with each request to the server, specifies what we want to do with the resource on the server.

HTTP Verb	Purpose
GET	To get data from the server
POST	To create new item on the server
DELETE	To delete data
PUT	To update data

Http service vs HttpClient service:

Http	HttpClient
Available only upto Angular Version < 4.3.x	Available from Angular Version 4.3.x and later
Present in import { Http } from '@angular/http';	Present in import { HttpClient } from '@angular/common/http';
Depricated from Anguler 5 version	It is available in latest Angular 14 version also.

Note: Since we are using **Angular Version 14** we will go with **HttpClient** service only.

Generate a new project;

>ng new employee-mgt-app

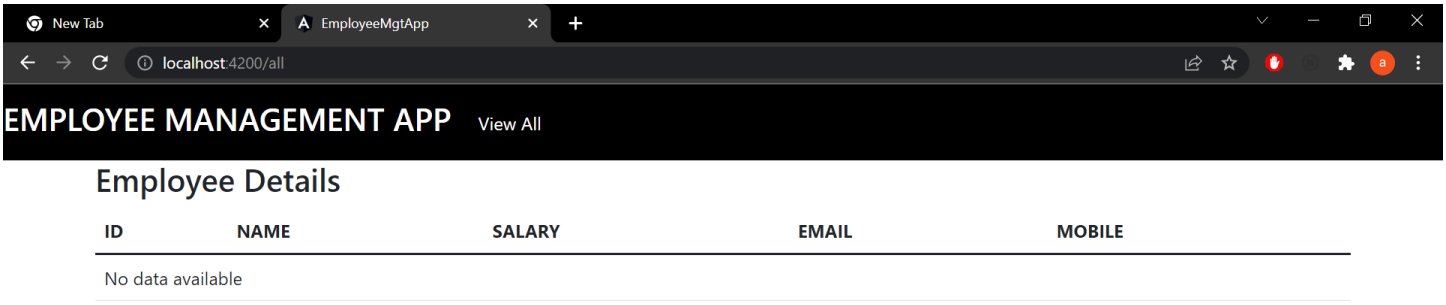
Update *tsconfig.json* with fallowing

"strictPropertyInitialization": false

```
/* To learn more about this file see: https://angular.io/config/tsconfig. */
{
  "compileOnSave": false,
  "compilerOptions": {
    "strictPropertyInitialization": false,
    "baseUrl": "./",
    "outDir": "./dist/out-tsc",
    "forceConsistentCasingInFileNames": true,
    "strict": true,
    "noImplicitOverride": true,
    "noPropertyAccessFromIndexSignature": true,
    "noImplicitReturns": true,
    "noFallthroughCasesInSwitch": true,
    "sourceMap": true,
    "declaration": false,
    "downlevelIteration": true,
    "experimentalDecorators": true,
    "moduleResolution": "node",
    "importHelpers": true,
    "target": "es2017",
    "module": "es2020",
    "lib": [
      "es2020",
      "dom"
    ]
  },
  "angularCompilerOptions": {
    "enableI18nLegacyMessageIdFormat": false,
    "strictInjectionParameters": true,
    "strictInputAccessModifiers": true,
    "strictTemplates": true
  }
}
```

Handling Http GET request (Getting data from server)

USE CASE:



employee-mgt-app>ng g class model/employee

employee-mgt-app >ng g s services/employee

employee-mgt-app >ng g c components/employee-list

Add the bootstrap CDN links in index.html

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="utf-8" />
    <title>EmployeeMgtApp</title>
    <base href="/" />
    <meta name="viewport" content="width=device-width, initial-scale=1" />
    <link rel="icon" type="image/x-icon" href="favicon.ico" />
    <link
      href="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/css/bootstrap.min.css"
      rel="stylesheet"
      integrity="sha384-1BmE4kWBq78iYhFldvKuhfTAU6auU8tT94WrHftjDbrCEXSU1oBoqyl2QvZ6jIW3"
      crossorigin="anonymous"
    />
    <script
      src="https://cdn.jsdelivr.net/npm/bootstrap@5.1.3/dist/js/bootstrap.bundle.min.js"
      integrity="sha384-ka7Sk0Gln4gmtz2MlQnikT1wXgYsOg+OMhuP+IlRH9sENBO0LRn5q+8nbTov4+1p"
      crossorigin="anonymous"
    ></script>
  </head>
  <body>
    <app-root></app-root>
  </body>
</html>
```

Add *HttpClientModule* and *FormsModule* and ReactiveFormsModule in *app.module.ts*

src\app\app.module.ts

```
import { NgModule } from '@angular/core';
import { FormsModule, ReactiveFormsModule } from '@angular/forms';
import { BrowserModule } from '@angular/platform-browser';

import { AppRoutingModule } from './app-routing.module';
import { AppComponent } from './app.component';
import { EmployeeListComponent } from './components/employee-list/employee-list.component';
import { HttpClientModule } from '@angular/common/http';

@NgModule({
  declarations: [AppComponent, EmployeeListComponent],
  imports: [
    BrowserModule,
    AppRoutingModule,
    FormsModule,
    ReactiveFormsModule,
    HttpClientModule,
  ],
  providers: [],
  bootstrap: [AppComponent],
})
export class AppModule {}
```

Add the fallowing Employe properties in employee.ts
src\app\model\employee.ts

```
export class Employee {
  employeeId: number;
  employeeName: string;
  employeeSalary: number;
  employeeEmail: string;
  employeeMobile: number;
}
```

Import "HttpClient" service and add dependence in Constructor in "YourService":

```
import { HttpClient } from "@angular/common/http";

constructor(private http : HttpClient){ }
```

Send "get" request to server:

```
this.http.get<ModelClassName>("url");
```

src\app\services\employee.service.ts

```
import { HttpClient } from '@angular/common/http';
import { Injectable } from '@angular/core';
import { Observable } from 'rxjs';
import { Employee } from '../model/employee';

@Injectable({
  providedIn: 'root',
})
export class EmployeeService {
  basePath: string = 'http://localhost:8181/myapp/api/employee';

  constructor(private http: HttpClient) {}

  getEmployees(): Observable<Employee[]> {
    return this.http.get<Employee[]>(this.basePath);
  }
}
```

Use EmployeeService in EmployeeListComponent

src\app\components\employee-list\employee-list.component.ts

```
import { Component, OnInit } from '@angular/core';
import { Employee } from 'src/app/model/employee';
import { EmployeeService } from 'src/app/services/employee.service';

@Component({
  selector: 'app-employee-list',
  templateUrl: './employee-list.component.html',
  styleUrls: ['./employee-list.component.css'],
})
export class EmployeeListComponent implements OnInit {
  employees: Employee[] = [];

  constructor(private service: EmployeeService) {}

  ngOnInit(): void {
    this.getEmployees();
  }

  getEmployees() {
    this.service.getEmployees().subscribe((data) => {
      this.employees = data;
      console.log(data);
    });
  }
}
```

Display the Employee Details in template.

src\app\components\employee-list\employee-list.component.html

```
<div class="container">
  <h3>Employee Details</h3>
  <table class="table">
    <thead>
      <tr>
        <th>ID</th>
        <th>NAME</th>
        <th>SALARY</th>
        <th>EMAIL</th>
        <th>MOBILE</th>
      </tr>
    </thead>
    <tbody>
      <tr *ngFor="let employee of employees">
        <td>{{ employee.employeeId }}</td>
        <td>{{ employee.employeeName }}</td>
        <td>{{ employee.employeeSalary }}</td>
        <td>{{ employee.employeeEmail }}</td>
        <td>{{ employee.employeeMobile }}</td>
      </tr>
      <tr *ngIf="!employees || employees.length == 0">
        <td colspan="5">No data available</td>
      </tr>
    </tbody>
  </table>
</div>
```

Add routes path for EmployeeListComponent in app-routing.module.ts file

src\app\app-routing.module.ts

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { EmployeeListComponent } from './components/employee-list/employee-list.component';

const routes: Routes = [{ path: 'all', component: EmployeeListComponent }];

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

Finally add route url in app.component.html

app1\src\app\app.component.html

```
<nav class="navbar navbar-expand-sm navbar-light bg-black">
  <a class="navbar-brand text-white" href="#">
    <h3>EMPLOYEE MANAGEMENT APP</h3>
  </a>
  <button
    class="navbar-toggler"
    type="button"
    data-toggle="collapse"
    data-target="#navbarSupportedContent"
    aria-controls="navbarSupportedContent"
    aria-expanded="false"
    aria-label="Toggle navigation"
  >
    <span class="navbar-toggler-icon"></span>
  </button>

  <div class="collapse navbar-collapse" id="navbarSupportedContent">
    <ul class="navbar-nav mr-auto">
      <li class="nav-item">
        <a class="nav-link text-white" routerLink="/all">View All</a>
      </li>
    </ul>
  </div>
</nav>
<router-outlet></router-outlet>
```

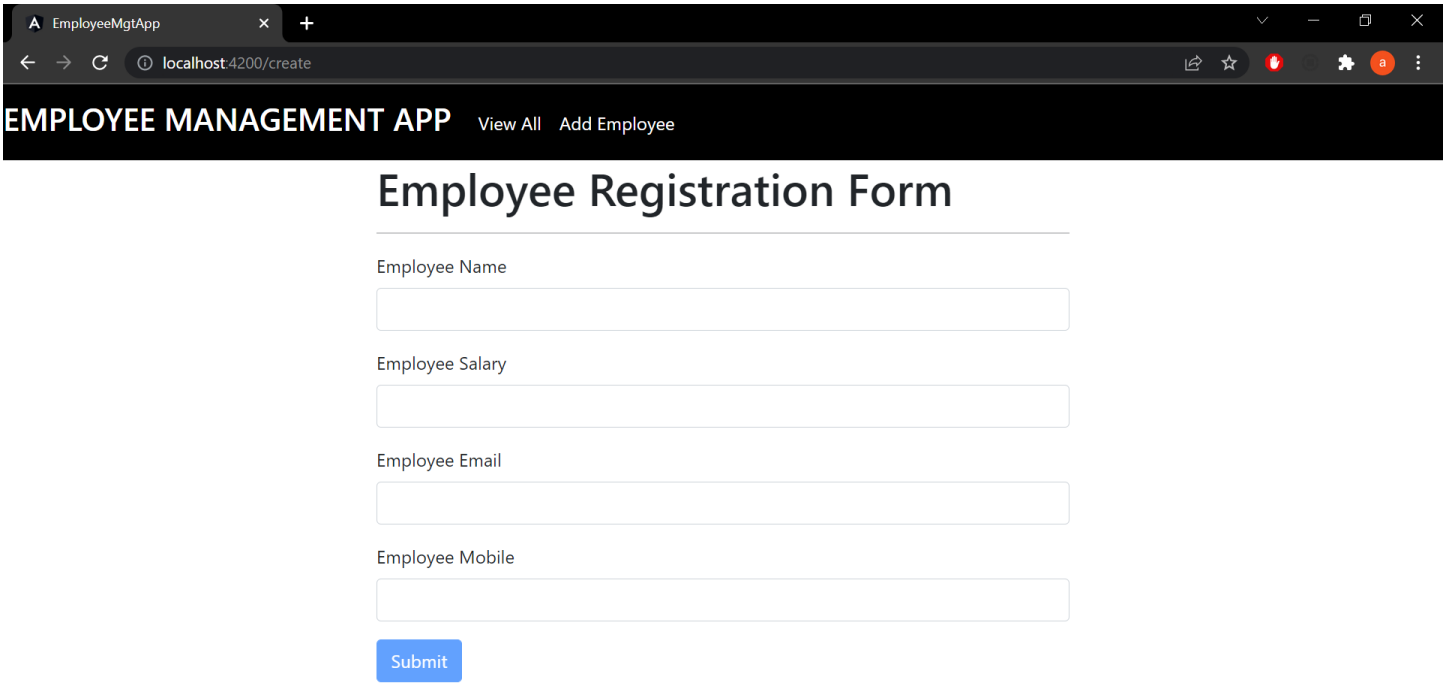
Open another cmd and run the following command.

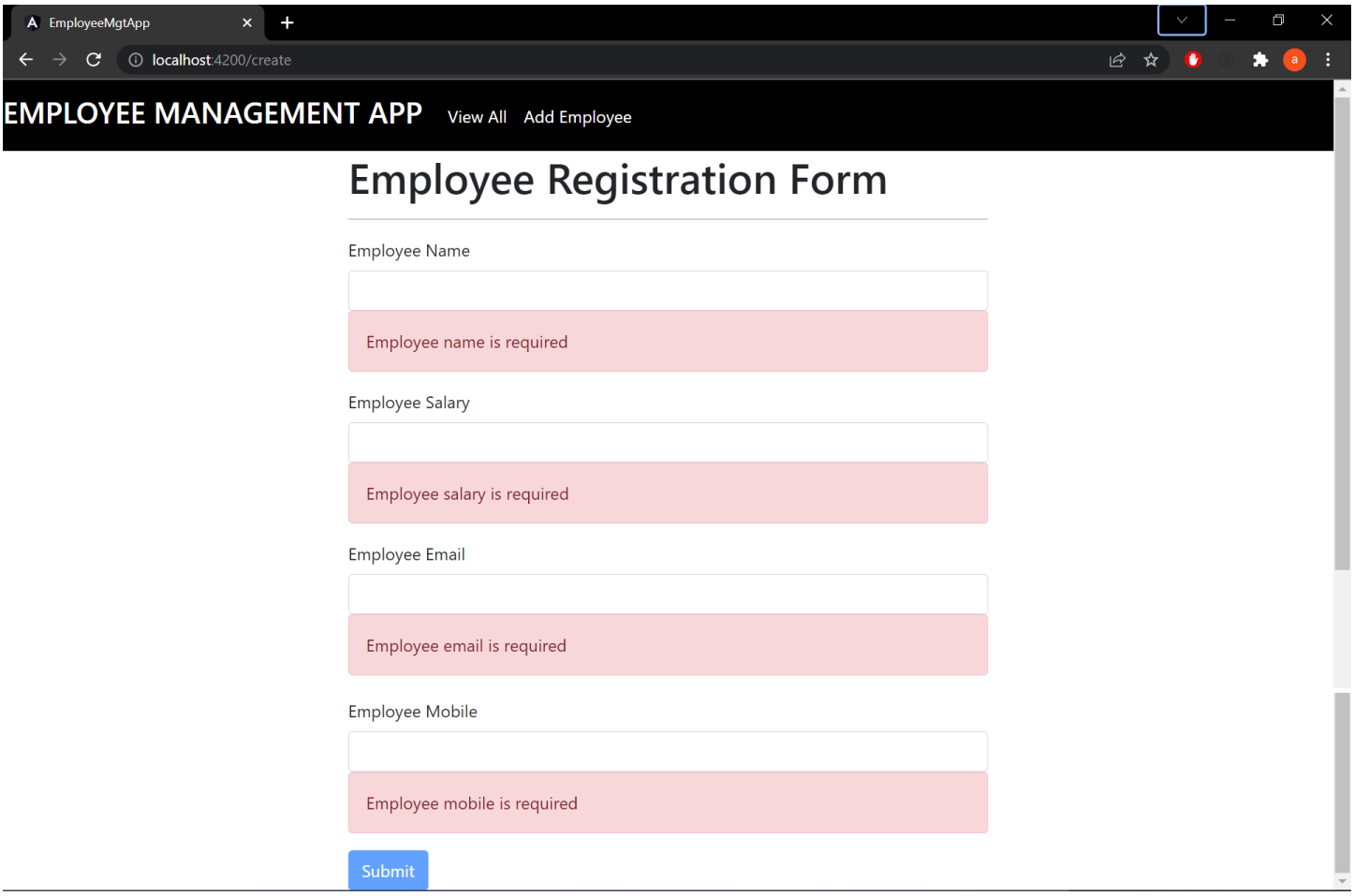
>ng server

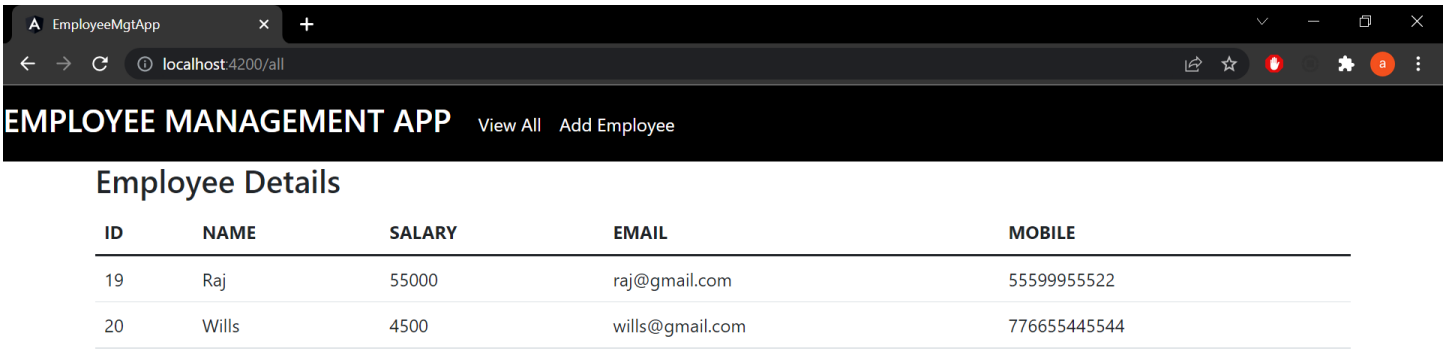
http://localhost:4200/

Handling HTTP POST Request (Sending data to Server):

USE CASE:




EmployeeMgtApp
localhost:4200/create
EMPLOYEE MANAGEMENT APP
View All
Add Employee
Employee Registration Form
Employee Name
Employee name is required
Employee Salary
Employee salary is required
Employee Email
Employee email is required
Employee Mobile
Employee mobile is required
Submit


EmployeeMgtApp
localhost:4200/all
EMPLOYEE MANAGEMENT APP
View All
Add Employee
Employee Details
ID
NAME
SALARY
EMAIL
MOBILE
19
Raj
55000
raj@gmail.com
55599955522
20
Wills
4500
wills@gmail.com
776655445544

Open the CMD and generate employee-create component.

employee-mgt-app>ng g c components/employee-create

Add Route path for EmployeeCreateComponent in app-routing.module.ts file.

app1\src\app\app-routing.module.ts

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { EmployeeCreateComponent } from './components/employee-create/employee-
create.component';
import { EmployeeListComponent } from './components/employee-list/employee-
list.component';

const routes: Routes = [
  { path: 'all', component: EmployeeListComponent },
  { path: 'create', component: EmployeeCreateComponent },
];

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

Add route url.

app1\src\app\app.component.html

```
<nav class="navbar navbar-expand-sm navbar-light bg-black">
  <a class="navbar-brand text-white" href="#">
    <h3>EMPLOYEE MANAGEMENT APP</h3>
  </a>
  <button
    class="navbar-toggler"
    type="button"
    data-toggle="collapse"
    data-target="#navbarSupportedContent"
    aria-controls="navbarSupportedContent"
    aria-expanded="false"
    aria-label="Toggle navigation"
  >
    <span class="navbar-toggler-icon"></span>
  </button>

  <div class="collapse navbar-collapse" id="navbarSupportedContent">
    <ul class="navbar-nav mr-auto">
      <li class="nav-item">
        <a class="nav-link text-white" routerLink="/all">View All</a>
      </li>
      <li class="nav-item">
        <a class="nav-link text-white" routerLink="/create">Add Employee</a>
      </li>
    </ul>
  </div>
</nav>
<router-outlet></router-outlet>
```

Add createEmployee() method to handle http post request in EmployeeService.

src\app\service\employee.service.ts

```
import { HttpClient } from '@angular/common/http';
import { Injectable } from '@angular/core';
import { Observable } from 'rxjs';
import { Employee } from '../model/employee';

@Injectable({
  providedIn: 'root',
})
export class EmployeeService {
  basePath: string = 'http://localhost:8181/myapp/api/employee';

  constructor(private http: HttpClient) {}

  getEmployees(): Observable<Employee[]> {
    return this.http.get<Employee[]>(this.basePath);
  }

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

Call the createEmployee() method of EmployeeService class in EmployeeCreateComponent.

src\app\components\employee-create\employee-create.component.ts

```
import { Component, OnInit } from '@angular/core';
import { FormBuilder, FormGroup, Validators } from '@angular/forms';
import { Router } from '@angular/router';
import { EmployeeService } from 'src/app/services/employee.service';

@Component({
  selector: 'app-employee-create',
  templateUrl: './employee-create.component.html',
  styleUrls: ['./employee-create.component.css'],
})
export class EmployeeCreateComponent implements OnInit {
  myform: FormGroup;

  constructor(
    private formBuilder: FormBuilder,
    private service: EmployeeService,
    private router: Router
  ) {}

  ngOnInit(): void {
    this.myform = this.formBuilder.group({
      employeeName: ['', Validators.required],
      employeeSalary: ['', Validators.required],
      employeeEmail: ['', Validators.required],
      employeeMobile: ['', Validators.required],
    });
  }

  createEmployee() {
    if (this.myform.valid) {
      this.service.createEmployee(this.myform.value).subscribe((data) => {
        console.log(data);
        this.router.navigateByUrl('/all');
      });
    } else {
      console.log(this.myform.value);
    }
  }
}
```

Design Add New Employee Form template.

src\app\components\employee-create\employee-create.component.html

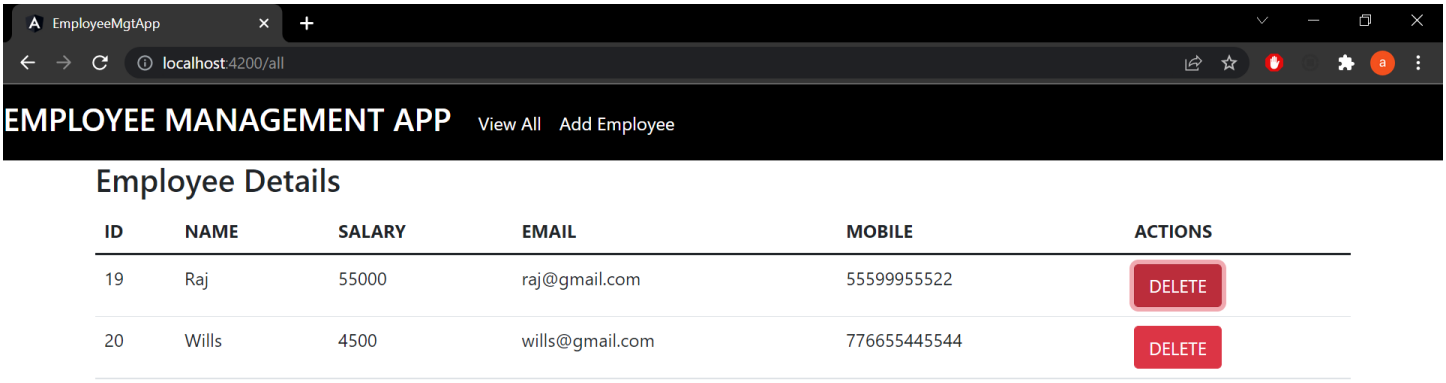
```
<div class="container justify-content-center col-6">
  <h1>Employee Registration Form</h1>
  <hr />
  <form [formGroup]="myform" (ngSubmit)="createEmployee()">
    <div class="mb-3">
      <label class="form-label">Employee Name</label>
      <input type="text" class="form-control" formControlName="employeeName" />
      <div
        *ngIf="
          myform.controls['employeeName'].dirty ||
          myform.controls['employeeName'].touched
        "
      >
        <div
          class="alert alert-danger"
          *ngIf="myform.controls['employeeName'].errors?.['required']"
        >
          Employee name is required
        </div>
      </div>
    </div>

    <div class="mb-3">
      <label class="form-label">Employee Salary</label>
      <input
        type="text"
        class="form-control"
        formControlName="employeeSalary"
      />
      <div
        *ngIf="
          myform.controls['employeeSalary'].dirty ||
          myform.controls['employeeSalary'].touched
        "
      >
        <div
          class="alert alert-danger"
          *ngIf="myform.controls['employeeSalary'].errors?.['required']"
        >
```

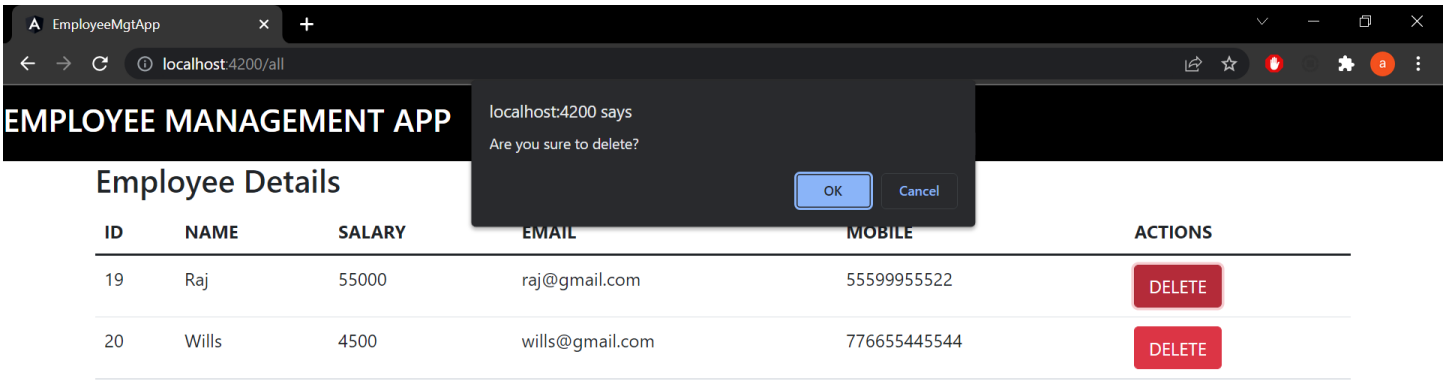
```
          Employee salary is required
        </div>
      </div>
    </div>

    <div class="mb-3">
      <label class="form-label">Employee Email</label>
      <input type="text" class="form-control" formControlName="employeeEmail" />
      <div
        *ngIf="
          myform.controls['employeeEmail'].dirty ||
          myform.controls['employeeEmail'].touched
        "
      >
        <div
          class="alert alert-danger"
          *ngIf="myform.controls['employeeEmail'].errors?.['required']"
        >
          Employee email is required
        </div>
      </div>
    </div>

    <div class="mb-3">
      <label class="form-label">Employee Mobile</label>
      <input
        type="text"
        class="form-control"
        formControlName="employeeMobile"
      />

      <div
        *ngIf="
          myform.controls['employeeMobile'].dirty ||
          myform.controls['employeeMobile'].touched
        "
      >
        <div
          class="alert alert-danger"
          *ngIf="myform.controls['employeeMobile'].errors?.['required']"
        >
          Employee mobile is required
        </div>
```

```
      </div>
    </div>

    <input
      type="submit"
      value="Submit"
      class="btn btn-primary"
      [disabled]="myform.invalid"
    /><br />
  </form>
</div>
```

Handling HTTP DELETE Request.

USE CASE:





Add delete service method in EmployeeService.

employee-mgt-app\src\app\service\employee.service.ts

```
import { HttpClient } from '@angular/common/http';
import { Injectable } from '@angular/core';
import { Observable } from 'rxjs';
import { Employee } from '../model/employee';

@Injectable({
  providedIn: 'root',
})
export class EmployeeService {
  basePath: string = 'http://localhost:8181/myapp/api/employee';

  constructor(private http: HttpClient) {}

  getEmployees(): Observable<Employee[]> {
    return this.http.get<Employee[]>(this.basePath);
  }

  createEmployee(employee: Employee): Observable<Employee> {
    return this.http.post<Employee>(this.basePath, employee);
  }

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

Invoke the delete service method of EmployeeService class.

employee-mgt-app\src\app\components\employee-list\employee-list.component.ts

```
import { Component, OnInit } from '@angular/core';
import { Employee } from 'src/app/model/employee';
import { EmployeeService } from 'src/app/services/employee.service';

@Component({
  selector: 'app-employee-list',
  templateUrl: './employee-list.component.html',
  styleUrls: ['./employee-list.component.css'],
})
export class EmployeeListComponent implements OnInit {
  employees: Employee[] = [];

  constructor(private service: EmployeeService) {}

  ngOnInit(): void {
    this.getEmployees();
  }

  getEmployees() {
    this.service.getEmployees().subscribe((data) => {
      this.employees = data;
      console.log(data);
    });
  }

  deleteEmployee(id: number) {
    if (confirm('Are you sure to delete?')) {
      this.service.deleteEmployee(id).subscribe((data) => {
        console.log(data);
        this.getEmployees();
      });
    }
  }
}
```

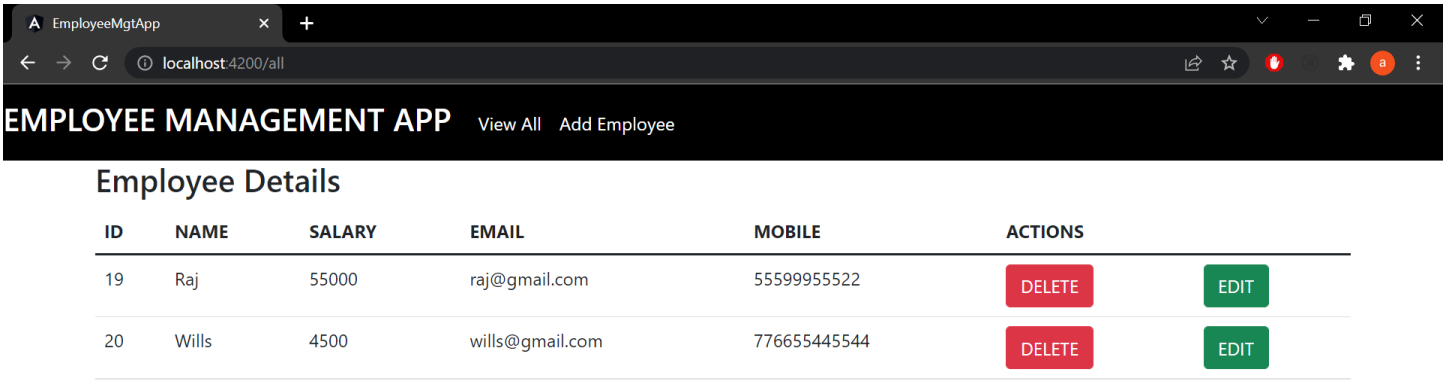
Add a delete button to perform delete event.

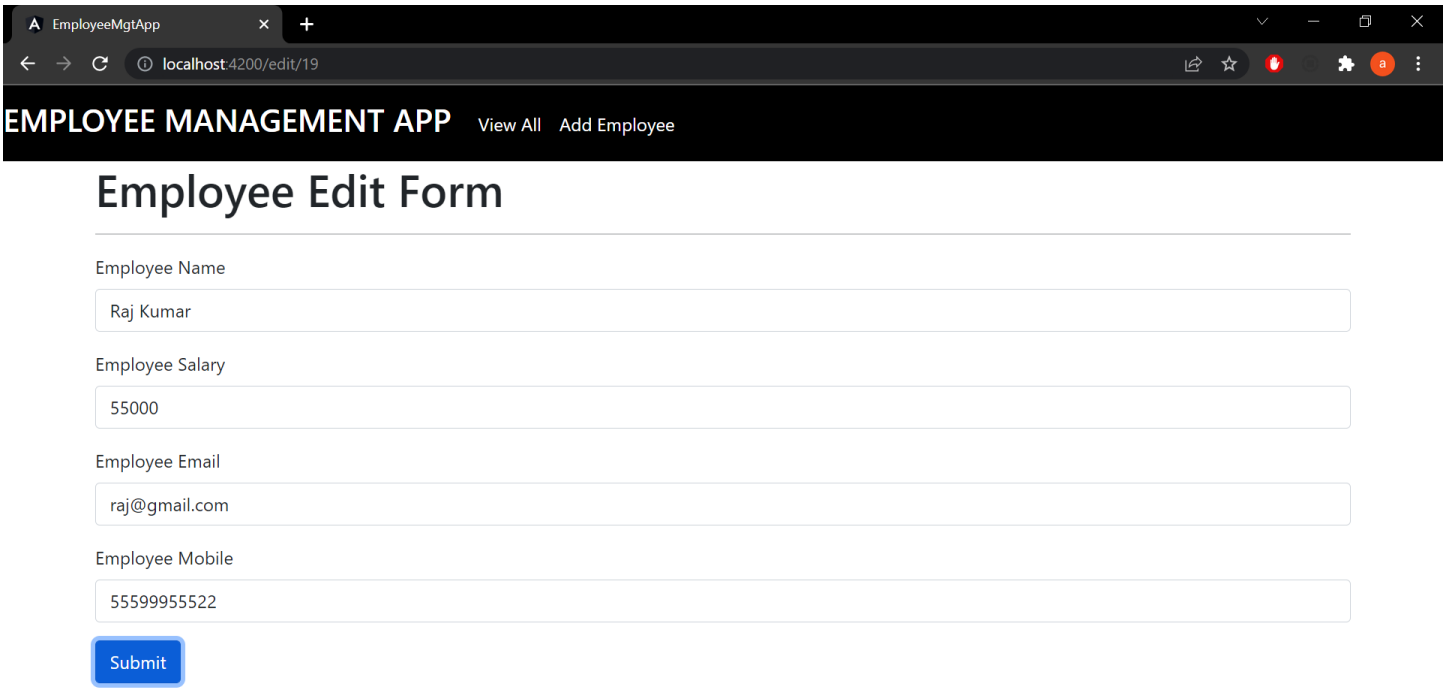
employee-mgt-app\src\app\components\employee-list\employee-list.component.html

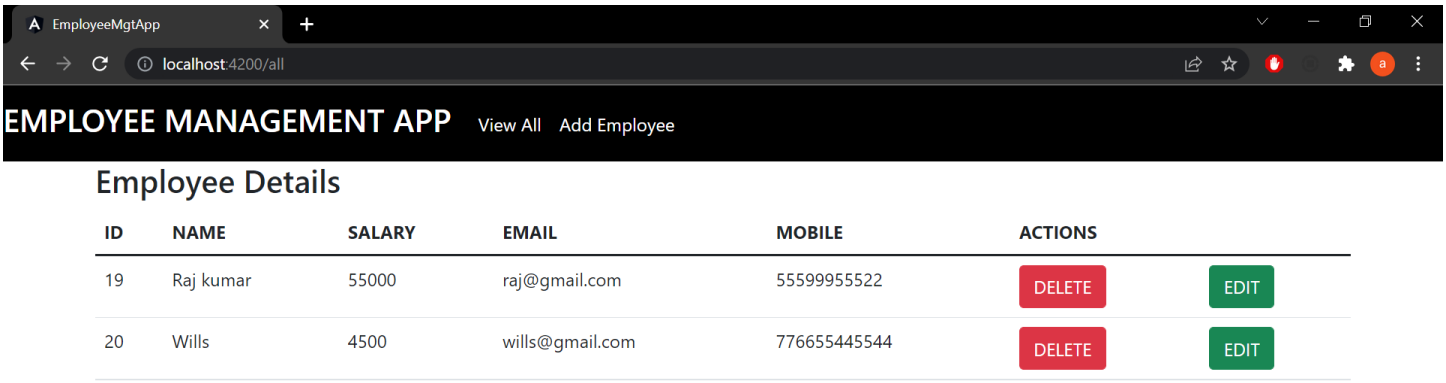
```
<div class="container">
  <h3>Employee Details</h3>
  <table class="table">
    <thead>
      <tr>
        <th>ID</th>
        <th>NAME</th>
        <th>SALARY</th>
        <th>EMAIL</th>
        <th>MOBILE</th>
        <th>ACTIONS</th>
      </tr>
    </thead>
    <tbody>
      <tr *ngFor="let employee of employees">
        <td>{{ employee.employeeId }}</td>
        <td>{{ employee.employeeName }}</td>
        <td>{{ employee.employeeSalary }}</td>
        <td>{{ employee.employeeEmail }}</td>
        <td>{{ employee.employeeMobile }}</td>
        <td>
          <button
            class="btn btn-danger"
            (click)="deleteEmployee(employee.employeeId)"
          >
            DELETE
          </button>
        </td>
      </tr>
      <tr *ngIf="!employees || employees.length == 0">
        <td colspan="5">No data available</td>
      </tr>
    </tbody>
  </table>
</div>
```

Handling HTTP UPDATE Request.

USE CASE:






EmployeeMgtApp
localhost:4200/all
EMPLOYEE MANAGEMENT APP
View All
Add Employee
Employee Details
ID
NAME
SALARY
EMAIL
MOBILE
ACTIONS
19
Raj kumar
55000
raj@gmail.com
55599955522
DELETE
EDIT
20
Wills
4500
wills@gmail.com
776655445544
DELETE
EDIT

Open the CMD and generate employee-edit component.

employee-mgt-app>ng g c components/employee-edit

Add path for EmployeeEditComponent in app-routing.module.ts

app1\src\app\app-routing.module.ts

```
import { NgModule } from '@angular/core';
import { RouterModule, Routes } from '@angular/router';
import { EmployeeCreateComponent } from './components/employee-create/employee-
create.component';
import { EmployeeEditComponent } from './components/employee-edit/employee-edit.component';
import { EmployeeListComponent } from './components/employee-list/employee-list.component';

const routes: Routes = [
  { path: 'all', component: EmployeeListComponent },
  { path: 'create', component: EmployeeCreateComponent },
  { path: 'edit/:id', component: EmployeeEditComponent },
];

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

Add *getEmployee() and updateEmployee()* service methods in EmployeeService class.

src\app\service\employee.service.ts

```
import { HttpClient } from '@angular/common/http';
import { Injectable } from '@angular/core';
import { Observable } from 'rxjs';
import { Employee } from '../model/employee';

@Injectable({
  providedIn: 'root',
})
export class EmployeeService {
  basePath: string = 'http://localhost:8181/myapp/api/employee';

  constructor(private http: HttpClient) {}

  getEmployees(): Observable<Employee[]> {
    return this.http.get<Employee[]>(this.basePath);
  }

  createEmployee(employee: Employee): Observable<Employee> {
    return this.http.post<Employee>(this.basePath, employee);
  }

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

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

  updateEmployee(employee: Employee): Observable<any> {
    return this.http.put(this.basePath, employee);
  }
}
```

Add *showEdit()* method to display edit form based on the id.

src\app\components\employee-list\employee-list.component.ts

```
import { Component, OnInit } from '@angular/core';
import { Router } from '@angular/router';
import { Employee } from 'src/app/model/employee';
import { EmployeeService } from 'src/app/services/employee.service';

@Component({
  selector: 'app-employee-list',
  templateUrl: './employee-list.component.html',
  styleUrls: ['./employee-list.component.css'],
})
export class EmployeeListComponent implements OnInit {
  employees: Employee[] = [];

  constructor(private service: EmployeeService, private router: Router) {}

  ngOnInit(): void {
    this.getEmployees();
  }

  getEmployees() {
    this.service.getEmployees().subscribe((data) => {
      this.employees = data;
      console.log(data);
    });
  }

  deleteEmployee(id: number) {
    if (confirm('Are you sure to delete?')) {
      this.service.deleteEmployee(id).subscribe((data) => {
        console.log(data);
        this.getEmployees();
      });
    }
  }

  showEdit(id: number) {
    //moving to EmployeeEditComponent
    this.router.navigate(['edit', id]);
  }
```

```
}
```

Add Edit button to perform update operation in student-list template.

src\app\components\employee-list\employee-list.component.html

```
<div class="container">
  <h3>Employee Details</h3>
  <table class="table">
    <thead>
      <tr>
        <th>ID</th>
        <th>NAME</th>
        <th>SALARY</th>
        <th>EMAIL</th>
        <th>MOBILE</th>
        <th>ACTIONS</th>
      </tr>
    </thead>
    <tbody>
      <tr *ngFor="let employee of employees">
        <td>{{ employee.employeeId }}</td>
        <td>{{ employee.employeeName }}</td>
        <td>{{ employee.employeeSalary }}</td>
        <td>{{ employee.employeeEmail }}</td>
        <td>{{ employee.employeeMobile }}</td>
        <td>
          <button
            class="btn btn-danger"
            (click)="deleteEmployee(employee.employeeId)"
          >
            DELETE
          </button>
        </td>
        <td>
          <button
            class="btn btn-success"
            (click)="showEdit(employee.employeeId)"
          >
            EDIT
          </button>
        </td>
      </tr>
```

```
      <tr *ngIf="!employees || employees.length == 0">
        <td colspan="5">No data available</td>
      </tr>
    </tbody>
  </table>
</div>
```

Implement EmployeeEditComponent

src\app\components\employee-edit\employee-edit.component.ts

```
import { Component, OnInit } from '@angular/core';
import { FormBuilder, FormGroup, Validators } from '@angular/forms';
import { ActivatedRoute, Router } from '@angular/router';
import { Employee } from 'src/app/model/employee';
import { EmployeeService } from 'src/app/servicess/employee.service';

@Component({
  selector: 'app-employee-edit',
  templateUrl: './employee-edit.component.html',
  styleUrls: ['./employee-edit.component.css'],
})
export class EmployeeEditComponent implements OnInit {
  myform: FormGroup;
  id: number;
  employee: Employee = new Employee();

  constructor(
    private formBuilder: FormBuilder,
    private service: EmployeeService,
    private router: Router,
    private activatedRoute: ActivatedRoute
  ) {
    this.myform = this.formBuilder.group({
      employeeName: ['', Validators.required],
      employeeSalary: ['', Validators.required],
      employeeEmail: ['', Validators.required],
      employeeMobile: ['', Validators.required],
    });
  }
```

```
  ngOnInit(): void {
    //read ID given by List Component on click Edit
    this.id = this.activatedRoute.snapshot.params['id'];

    //call service and subscribe success data to student
    this.service.getEmployee(this.id).subscribe((data) => {
      this.employee = data;
    });
  }

  updateEmployee() {
    this.service.updateEmployee(this.employee).subscribe((data) => {
      console.log(data);
      this.router.navigate(['all']);
    });
  }
}
```

src\app\components\employee-edit\employee-edit.component.html

app1\src\app\student-edit\student-edit.component.html

```
<div class="container">
  <h1>Employee Edit Form</h1>
  <hr />
  <form [formGroup]="myform" (ngSubmit)="updateEmployee()">
    <div class="mb-3">
      <label class="form-label">Employee Name</label>
      <input
        type="text"
        class="form-control"
        formControlName="employeeName"
        [(ngModel)]="employee.employeeName"
      />
      <div
        *ngIf="
          myform.controls['employeeName'].dirty ||
          myform.controls['employeeName'].touched
        "
      >
        <div
          class="alert alert-danger"
          *ngIf="myform.controls['employeeName'].errors?.['required']"
        >
          Employee name is required
        </div>
      </div>
    </div>

    <div class="mb-3">
      <label class="form-label">Employee Salary</label>
      <input
        type="text"
        class="form-control"
        formControlName="employeeSalary"
        [(ngModel)]="employee.employeeSalary"
      />
      <div
        *ngIf="
          myform.controls['employeeSalary'].dirty ||
          myform.controls['employeeSalary'].touched
```

```
      "
    >
      <div
        class="alert alert-danger"
        *ngIf="myform.controls['employeeSalary'].errors?.['required']"
      >
        Employee salary is required
      </div>
    </div>
  </div>

  <div class="mb-3">
    <label class="form-label">Employee Email</label>
    <input
      type="text"
      class="form-control"
      formControlName="employeeEmail"
      [(ngModel)]="employee.employeeEmail"
    />
    <div
      *ngIf="
        myform.controls['employeeEmail'].dirty ||
        myform.controls['employeeEmail'].touched
      "
    >
      <div
        class="alert alert-danger"
        *ngIf="myform.controls['employeeEmail'].errors?.['required']"
      >
        Employee email is required
      </div>
    </div>
  </div>

  <div class="mb-3">
    <label class="form-label">Employee Mobile</label>
    <input
      type="text"
      class="form-control"
      formControlName="employeeMobile"
      [(ngModel)]="employee.employeeMobile"
    />
```

```
      <div
        *ngIf="
          myform.controls['employeeMobile'].dirty ||
          myform.controls['employeeMobile'].touched
        "
      >
        <div
          class="alert alert-danger"
          *ngIf="myform.controls['employeeMobile'].errors?.['required']"
        >
          Employee mobile is required
        </div>
      </div>
    </div>

    <input
      type="submit"
      value="Submit"
      class="btn btn-primary"
      [disabled]="myform.invalid"
    /><br />
  </form>
</div>
```