

As we have learned about \$http services in the previous sessions, now we will perform CRUD operations for employee and department.

Open **Visual Studio 2015** → go to **File** → **New** → **Project** → Expand **Visual C#** → **Web** → **Create ASP.NET Web Application** → Check **WebApi and MVC**

Step1: Creating tables for Employee and Department in SQL Server

Table - Employee		Table - Department	
EmpId	Int (PK)	PKDeptId	Int (PK)
EmpName	Varchar(50)	DeptName	Varchar(50)
EmpSalary	Money		
FKDeptId	Int (FK)		

Step2: Create ADO.NET Entity Data model & Create API's

Solution Explorer → go to **Models** → right click and click **Add** → **New Item** → **Visual C#** → **Data** → **ADO.NET Entity Data Model**

Step3: Creating API's using entity framework

File: **ManageDepartmentController.cs**

```
public class ManageDepartmentController : ApiController
{
    EmpDeptDBEntities context = new EmpDeptDBEntities();
    // GET: api/ManageDepartment
    public IEnumerable<Department> Get()
    {
        return context.Departments;
    }
    // GET: api/ManageDepartment/5
    public Department Get(int id)
    {
        return context.Departments.Find(id);
    }
    // POST: api/ManageDepartment
    public IEnumerable<Department> AddDepartment([FromBody]Department dept)
    {
        context.Departments.Add(dept);
        context.SaveChanges();
        return Get();
    }
}
```

```
// PUT: api/ManageDepartment/5
public IEnumerable<Department> Put([FromBody]Department dept)
{
    Department oldDept = context.Departments.Find(dept.DeptId);
    context.Entry(oldDept).CurrentValues.SetValues(dept);
    context.SaveChanges();
    return Get();
}

// DELETE: api/ManageDepartment/5
public IEnumerable<Department> Delete(int id)
{
    context.Departments.Remove(Get(id));
    context.SaveChanges();
    return Get();
}
}
```

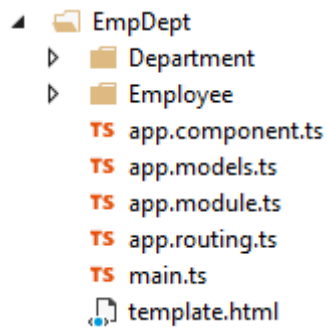
File: **ManageEmployeeController.cs**

```
public class ManageEmployeeController : ApiController
{
    EmpDeptDBEntities context = new EmpDeptDBEntities();
    public List<Employee> Get()
    {
        List<Employee> lstEmp = context.Employees.ToList();
        return lstEmp;
    }
    public Employee Get(int id)
    {
        return context.Employees.Find(id);
    }
    public List<Employee> AddEmployee(Employee emp)
    {
        context.Employees.Add(emp);
        context.SaveChanges();
        return Get();
    }
    public List<Employee> Put([FromBody]Employee emp)
    {
        Employee oldEmp = Get(emp.EmpId);
```

```
context.Entry(oldEmp).CurrentValues.SetValues(emp);
context.SaveChanges();
return Get();
}
public List<Employee> Delete(int id)
{
    context.Employees.Remove(Get(id));
    context.SaveChanges();
    return Get();
}
}
```

Step4: Setup the application with quick start files from github, for installation please refer introduction.

Step5: Create directory structure for angular application



```
EmpDept
├── Department
├── Employee
├── TS app.component.ts
├── TS app.models.ts
├── TS app.module.ts
├── TS app.routing.ts
├── TS main.ts
└── template.html
```

Step6: Creating models for Employee and Department

File: **app.models.ts**

```
export class Department {
    DeptId: number;
    DeptName: string;
}

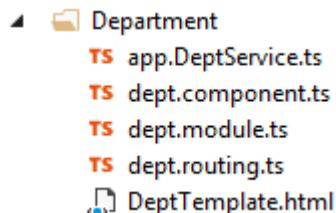
export class Employee {
    EmpId: number;
    EmpName: string;
    EmpSalary: number;
    FKDeptId: number;
    Department: Department;
    constructor() {
    }
}
```

Step6: Creating root component with selectorFile: **app.component.ts**

```
import { Component } from '@angular/core';
@Component({
  selector: 'my-app',
  templateUrl: './template.html',
})
export class AppComponent{ }
```

File: **template.html**

```
<div class="clearfix">&nbsp;</div>
<div class="container">
  <h1>Employee Management</h1>
  <nav>
    <button class="btn btn-warning" routerLink="emp">Employees</button>
    <button class="btn btn-warning" routerLink="dept">Departments</button>
  </nav>
  <router-outlet></router-outlet>
</div>
```

Step7: Creating component,service and template for Department

```
Department
├── app.DeptService.ts
├── dept.component.ts
├── dept.module.ts
├── dept.routing.ts
└── DeptTemplate.html
```

Here we are using lazyloading for department module

File: **app.DeptService.ts**

```
import { Http, Response, RequestOptions, Headers } from '@angular/http';
import { Injectable } from '@angular/core';
import 'rxjs/add/operator/map';
import { Observable } from 'rxjs/observable';
import { Department } from '../app.models';

@Injectable()
export class DeptService {
  departments: Department[];

  constructor(private http: Http) {
```

```
}

GetDepartments(): Observable<Department[]> {
    return this.http.get("/api/ManageDepartment").map((res: Response) => res.json());
}

GetDepartment(deptId: any): Observable<Department> {
    const url = `${'/api/ManageDepartment'}/${deptId}`;
    return this.http.get(url).map((res: Response) => res.json());
}

AddDepartment(dept: Department): Observable<Department[]>{
    let data = JSON.stringify(dept);
    let headers: Headers = new Headers({ "content-type": "application/json" });
    let options: RequestOptions = new RequestOptions({ headers: headers });
    return this.http.post("/api/ManageDepartment/AddDepartment", data, options).map((res: Response) =>
res.json());
}

UpdateDepartment(dept: Department): Observable<Department[]> {
    const url = `${'/api/ManageDepartment'}/${dept.DeptId}`;
    let data = JSON.stringify(dept);
    let headers: Headers = new Headers({ "content-type": "application/json" });
    let options: RequestOptions = new RequestOptions({ headers: headers });
    return this.http.put(url, data, options).map((res: Response) => res.json());
}

DeleteDepartment(deptId: any): Observable<Department[]> {
    debugger;
    const url = `${'/api/ManageDepartment'}/${deptId}`;
    return this.http.delete(url).map((res: Response) => res.json());
}
}
```

File: **dept.component.ts**

```
import { Component, OnInit, Input } from '@angular/core';
import { DeptService } from '../app.DeptService';
import { Department } from '../app.models';
import { CommonModule } from '@angular/common';

@Component({
    templateUrl: './DeptTemplate.html',
    providers: [DeptService]
```

```
})  
export class DeptComponent implements OnInit {  
  departments: Department[];  
  department: Department = new Department();  
  action: string = "Add";  
  constructor(private deptService: DeptService) { }  
  ngOnInit() {  
    this.GetDepartments();  
  }  
  Add() {  
    this.action = "Add";  
    this.department = new Department();  
    this.GetDepartments();  
  }  
  GetDepartments() {  
    this.deptService.GetDepartments().subscribe(depts => this.departments = depts);  
  }  
  GetDepartment(deptId: any) {  
    this.deptService.GetDepartment(deptId).subscribe(dept => this.department = dept);  
  }  
  AddDepartment() {  
    this.deptService.AddDepartment(this.department).subscribe(depts => this.departments = depts);  
  }  
  EditDepartment(deptId: any) {  
    this.GetDepartment(deptId);  
    this.action = "Edit";  
  }  
  UpdateDepartment() {  
    this.deptService.UpdateDepartment(this.department).subscribe(depts => this.departments = depts);  
  }  
  DeleteDepartment(deptId: any) {  
    debugger;  
    this.deptService.DeleteDepartment(deptId).subscribe(depts => this.departments = depts);  
  }  
}
```

File: **dept.routing.ts**

Here we are maintaining separate routing for department to make this module lazy loaded.

```
import { ModuleWithProviders } from '@angular/core';
import { Routes, RouterModule } from '@angular/router';
import { DeptComponent } from './dept.component';
const routes: Routes = [
  { path: '', component: DeptComponent }
];
export const routing: ModuleWithProviders = RouterModule.forChild(routes);
```

File: **dept.module.ts**

Here we must import **CommonModule** to use NgModel in lazy loaded module, and no need to import **BrowserModule** because we have already imported that in root **NgModule**.

```
import { NgModule } from '@angular/core';
import { DeptComponent } from './dept.component';
import { HttpClientModule } from '@angular/http';
import { FormsModule } from '@angular/forms';
import { routing } from './dept.routing';
import { CommonModule } from '@angular/common';

@NgModule({
  imports: [CommonModule, HttpClientModule, FormsModule, routing],
  declarations: [DeptComponent],
  bootstrap: [DeptComponent]
})
export class DeptModule { }
```

File: **DeptTemplate.html**

```
<div class="clearfix">&nbsp;</div>
<div class="container">
  <button class="btn btn-primary" data-toggle="modal" data-target="#myModal" (click)="Add()">Add
New</button>
  <div class="clearfix">&nbsp;</div>
  <table class="table table-striped table-bordered" style="width:50%">
    <thead>
      <tr>
        <th>Department Name</th>
        <th></th>
        <th></th>
      </tr>
```

```
</thead>
<tbody>
  <tr *ngFor="let dept of departments">
    <td>{{dept.DeptName}}</td>
    <td><button type="button" class="btn btn-info" (click)="EditDepartment(dept.DeptId)" data-
toggle="modal" data-target="#myModal">Edit</button> </td>
    <td><button type="button" class="btn btn-danger" (click)="GetDepartment(dept.DeptId)" data-
toggle="modal" data-target="#deleteModal">Delete</button> </td>
  </tr>
</tbody>
</table>
<div id="myModal" class="modal fade" role="dialog">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <button type="button" class="close" data-dismiss="modal">&times;</button>
        <h4 class="modal-title">Add/Edit Department</h4>
      </div>
      <div class="modal-body">
        <form name="addEditForm">
          <div class="form-group">
            <label for="empName">Department Name</label>
            <input type="text" class="form-control" name="empName" id="empName"
[(ngModel)]="department.DeptName" required />
          </div>
          <div class="form-group">
            <button type="submit" class="btn btn-info" (click)="AddDepartment()" *ngIf="action=='Add'"
data-dismiss="modal">Add</button>
            <button type="submit" class="btn btn-info" (click)="UpdateDepartment()"
*ngIf="action=='Edit'" data-dismiss="modal">Update</button>
            <button type="button" class="btn btn-default" data-dismiss="modal">Close</button>
          </div>
        </form>
      </div>
    </div>
  </div>
</div>
</div>
```



```
<div id="deleteModal" class="modal fade" role="dialog">
  <div class="modal-dialog">
    <div class="modal-content">
      <div class="modal-header">
        <button type="button" class="close" data-dismiss="modal">&times;</button>
        <h4 class="modal-title">Are you sure want to delete?</h4>
      </div>
      <div class="modal-body">
        Department Name: <b>{{department.DeptName}}</b>
      </div>
      <div class="modal-footer">
        <button type="button" class="btn btn-danger" (click)="DeleteDepartment(department.DeptId)"
data-dismiss="modal">Yes</button>
        <button type="button" class="btn btn-default" data-dismiss="modal">No</button>
      </div>
    </div>
  </div>
</div>
```

Step8: Creating component,service and template for Employee

- Employee
 - TS app.EmpService.ts
 - TS emp.component.ts
 - EmpTemplate.html

File: **app.EmpService.ts**

```
import { Http, Response, RequestOptions, Headers } from '@angular/http';
import { Injectable } from '@angular/core';
import 'rxjs/add/operator/map';
import { Observable } from 'rxjs/observable';
import { Employee } from '../app.models';

@Injectable()
export class EmpService {
  employees: Employee[];
  constructor(private http: Http) {
  }
}
```

```
GetEmployees(): Observable<Employee[]> {  
    return this.http.get("/api/ManageEmployee").map((res: Response) => res.json());  
}  
  
GetEmployee(empId: any): Observable<Employee> {  
    const url = `${'/api/ManageEmployee'}/${empId}`;  
    return this.http.get(url).map((res: Response) => res.json());  
}  
  
AddEmployee(emp: Employee): Observable<Employee[]>{  
    let data = JSON.stringify(emp);  
    let headers: Headers = new Headers({ "content-type": "application/json" });  
    let options: RequestOptions = new RequestOptions({ headers: headers });  
    return this.http.post("/api/ManageEmployee/AddEmployee", data, options).map((res: Response) =>  
res.json());  
}  
  
UpdateEmployee(emp: Employee): Observable<Employee[]> {  
    const url = `${'/api/ManageEmployee'}/${emp.EmpId}`;  
    let data = JSON.stringify(emp);  
    let headers: Headers = new Headers({ "content-type": "application/json" });  
    let options: RequestOptions = new RequestOptions({ headers: headers });  
    return this.http.put(url, data, options).map((res: Response) => res.json());  
}  
  
DeleteEmployee(empId: any): Observable<Employee[]> {  
    const url = `${'/api/ManageEmployee'}/${empId}`;  
    return this.http.delete(url).map((res: Response) => res.json());  
}  
}
```

File: **emp.component.ts**

```
import { Component, OnInit, Input } from '@angular/core';  
import { EmpService } from '../app.EmpService';  
import { DeptService } from '../Department/app.DeptService';  
import { Employee, Department } from '../app.models';  
  
@Component({  
    templateUrl: './EmpTemplate.html',  
    providers: [EmpService, DeptService]
```

```
})  
  
export class EmpComponent implements OnInit {  
  employees: Employee[];  
  action: string = "Add";  
  employee: Employee = new Employee();  
  
  departments: Department[];  
  department: Department = new Department();  
  
  constructor(private empService: EmpService, private deptService: DeptService) { }  
  ngOnInit() {  
    this.GetEmployees();  
  }  
  Add() {  
    this.action = "Add";  
    this.employee = new Employee();  
    this.GetDepartments();  
  }  
  GetEmployees() {  
    this.empService.GetEmployees().subscribe(emps => this.employees = emps);  
  }  
  GetDepartments() {  
    this.deptService.GetDepartments().subscribe(depts => this.departments = depts);  
  }  
  GetEmployee(empId: any) {  
    this.empService.GetEmployee(empId).subscribe(emp => this.employee = emp);  
  }  
  AddEmployee() {  
    this.empService.AddEmployee(this.employee).subscribe(emps => this.employees = emps);  
  }  
  EditEmployee(empId: any) {  
    this.GetEmployee(empId);  
    this.GetDepartments();  
    this.action = "Edit";  
  }  
  UpdateEmployee() {  
    this.empService.UpdateEmployee(this.employee).subscribe(emps => this.employees = emps);  
  }  
}
```

```

DeleteEmployee(empId: any) {
    this.empService.DeleteEmployee(empId).subscribe(emps => this.employees = emps);
}
}

```

File: **EmpTemplate.html**

```

<div class="clearfix">&nbsp;</div>
<div class="container">
    <button class="btn btn-primary" data-toggle="modal" data-target="#myModal" (click)="Add()">Add
New</button>
    <div class="clearfix">&nbsp;</div>
    <table class="table table-striped table-bordered" style="width:50%">
        <thead>
            <tr>
                <th>Employee name</th>
                <th>Employee Salary</th>
                <!-- <th>Department</th-->
                <th></th>
                <th></th>
            </tr>
        </thead>
        <tbody>
            <tr *ngFor="let emp of employees">
                <td>{{emp.EmpName}}</td>
                <td>{{emp.EmpSalary}}</td>
                <!--<td>{{emp.Department.DeptName}}</td-->
                <td><button type="button" class="btn btn-info" (click)="EditEmployee(emp.EmpId)" data-
toggle="modal" data-target="#myModal">Edit</button> </td>
                <td><button type="button" class="btn btn-danger" (click)="GetEmployee(emp.EmpId)" data-
toggle="modal" data-target="#deleteModal">Delete</button> </td>
            </tr>
        </tbody>
    </table>
    <div id="myModal" class="modal fade" role="dialog">
        <div class="modal-dialog">
            <div class="modal-content">
                <div class="modal-header">
                    <button type="button" class="close" data-dismiss="modal">&times;</button>

```

```
<h4 class="modal-title">Add/Edit Employee</h4>
</div>
<div class="modal-body">
  <form name="addEditForm">
    <div class="form-group">
      <label for="empName">Employee Name</label>
      <input type="text" class="form-control" name="empName" id="empName"
[(ngModel)]="employee.EmpName" required />
    </div>
    <div class="form-group">
      <label for="empSal">Employee Salary</label>
      <input type="text" class="form-control" name="empSal" id="empSal"
[(ngModel)]="employee.EmpSalary" required />
    </div>
    <div class="form-group">
      <label for="dept">Department</label>
      <select name="selectDepartment" class="form-control" id="dept"
[(ngModel)]="employee.FKDeptId">
        <option *ngFor="let dept of departments"
value="{{dept.DeptId}}">{{dept.DeptName}}</option>
      </select>
    </div>
    <div class="form-group">
      <button type="submit" class="btn btn-info" (click)="AddEmployee()" *ngIf="action=='Add'"
data-dismiss="modal">Add</button>
      <button type="submit" class="btn btn-info" (click)="UpdateEmployee()" *ngIf="action=='Edit'"
data-dismiss="modal">Update</button>
      <button type="button" class="btn btn-default" data-dismiss="modal">Close</button>
    </div>
  </form>
</div>
</div>
</div>
</div>

<div id="deleteModal" class="modal fade" role="dialog">
  <div class="modal-dialog">
```

```
<!-- Modal content-->
<div class="modal-content">
  <div class="modal-header">
    <button type="button" class="close" data-dismiss="modal">&times;</button>
    <h4 class="modal-title">Are you sure want to delete?</h4>
  </div>
  <div class="modal-body">
    Employee Name: <b>{{employee.EmpName}}</b>
  </div>
  <div class="modal-footer">
    <button type="button" class="btn btn-danger" (click)="DeleteEmployee(employee.EmpId)" data-
dismiss="modal">Yes</button>
    <button type="button" class="btn btn-default" data-dismiss="modal">No</button>
  </div>
</div>
</div>
</div>
</div>
```

Step9: Configuring application to use Http service in root NgModule

File: **app.module.ts**

Here default routing is navigated to EmpComponent in **app.routing**

```
import { NgModule } from '@angular/core';
import { BrowserModule } from '@angular/platform-browser';
import { CommonModule } from '@angular/common';
import { AppComponent } from './app.component'; //importing app.component module
import { EmpComponent } from './Employee/emp.component';
import { HttpClientModule } from '@angular/http';
import { FormsModule } from '@angular/forms';
import { routing } from './app.routing';

@NgModule({
  imports: [BrowserModule, HttpClientModule, FormsModule, routing],
  declarations: [AppComponent, EmpComponent],
  bootstrap: [AppComponent]
})
export class AppModule { }
```

File: **app.routing.ts**

As we are considering this app.routing as root module we have configured routes in **forRoot**.

```
import { ModuleWithProviders } from '@angular/core';
import { Routes, RouterModule } from '@angular/router';
import { EmpComponent } from './Employee/emp.component';
import { DeptComponent } from './Department/dept.component';
const routes: Routes = [
  { path: '', redirectTo: 'emp', pathMatch: 'full' },
  { path: 'emp', component: EmpComponent },
  { path: 'dept', loadChildren: './app/EmpDept/Department/dept.module#DeptModule' }
];

export const routing: ModuleWithProviders = RouterModule.forRoot(routes);
```

Step10: Now create html file and use your component selector

```
<script>
  System.import('app/EmpDept/main.js').catch(function (err) { console.error(err); });
</script>
<body>
  <my-app>Loading...</my-app>
</body>
```

Output: You can perform all the crud operations.

Employees

Departments

Add New

Employee name	Employee Salary	Department		
Suresh	12000	Test	Edit	Delete
Muni	12500	Test	Edit	Delete
Phani	17500	Developing	Edit	Delete
kavitha	15001	Developing	Edit	Delete
Yeswanth	12000	Test	Edit	Delete

Employees

Departments

Add New

Department Name		
Test	Edit	Delete
Developing	Edit	Delete