Name: Kaustubh Wade ER No.: 160410116050 Class: TYIT-1 Batch-C

Beyond The Syllabus Practical

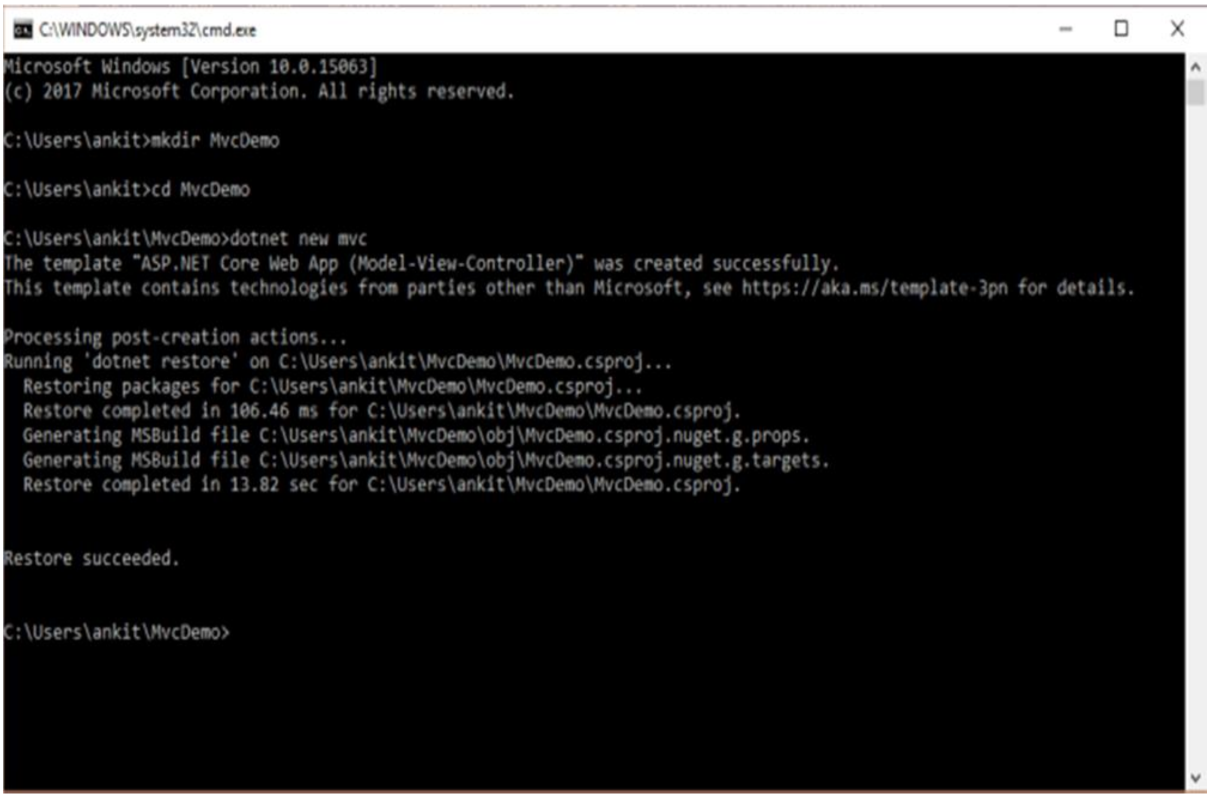
AIM: Create an application by using MVC framework which performs CRUD operation.

Type the following Commands. It will create our MVC application, "MvcDemo":

mkdir MvcDemo

cd MvcDemo

dotnet new mvc



```
Microsoft Windows [Version 10.0.15063]
(c) 2017 Microsoft Corporation. All rights reserved.

C:\Users\ankit>mkdir MvcDemo

C:\Users\ankit>cd MvcDemo

C:\Users\ankit\MvcDemo>dotnet new mvc
The template "ASP.NET Core Web App (Model-View-Controller)" was created successfully.
This template contains technologies from parties other than Microsoft, see https://aka.ms/template-3pn for details.

Processing post-creation actions...
Running 'dotnet restore' on C:\Users\ankit\MvcDemo\MvcDemo.csproj...
  Restoring packages for C:\Users\ankit\MvcDemo\MvcDemo.csproj...
  Restore completed in 106.46 ms for C:\Users\ankit\MvcDemo\MvcDemo.csproj.
  Generating MSBuild file C:\Users\ankit\MvcDemo\obj\MvcDemo.csproj.nuget.g.props.
  Generating MSBuild file C:\Users\ankit\MvcDemo\obj\MvcDemo.csproj.nuget.g.targets.
  Restore completed in 13.82 sec for C:\Users\ankit\MvcDemo\MvcDemo.csproj.


Restore succeeded.


C:\Users\ankit\MvcDemo>
```

Employees.cs

```
using System.;

using System.ComponentModel.DataAnnotations;

namespace Mvc.Models

{       public class Employees

        {       public int Id

                {       get;    set;    }

                [Required]

                public string Name
                {       get;    set;    }
                [Required]
                public string City
                {       get;    set;    }
                [Required]
                public  string  Department
                {       get;    set;    }
                [Required]
                public  int  Salary
                {       get;    set;    }
        }
}
```

MvcDemo.csproj

```
<Project Sdk="Microsoft.NET.Sdk.Web">
<PropertyGroup>
<TargetFramework>netcoreapp2.0</TargetFramework>
</PropertyGroup>  <ItemGroup>        <PackageReference Include="Microsoft.AspNetCore.All" Version="2.0.0" />
<PackageReference Include="Microsoft.VisualStudio.Web.CodeGeneration.Design" Version="2.0.0" /> </ItemGroup>
<ItemGroup>
<DotNetCliToolReference Include="Microsoft.VisualStudio.Web.CodeGeneration.Tools" Version="2.0.0" />
<DotNetCliToolReference Include="Microsoft.EntityFrameworkCore.Tools.DotNet" Version="2.0.0" />
</ItemGroup>
</Project>
```

MVCEmployeeContext.cs

```
using  Microsoft.EntityFrameworkCore; namespace  MvcDemo.Models
{       public  class  MvcEmployeeContext  :  DbContext
```

```
        {       public  MvcEmployeeContext  (DbContextOptions<MvcEm ployeeContext>  options)
: base(options)
                { }
                public  DbSet<MvcDemo.Models.Employees>  Employee
                {       get;    set;    }
        }
}
```

Startup.cs

```
using System;
using  System.Collections.Generic; using  System.Linq;
using  System.Threading.Tasks;
using  Microsoft.AspNetCore.Builder; using  Microsoft.AspNetCore.Hosting;
using  Microsoft.Extensions.Configuration;
using  Microsoft.Extensions.DependencyInjection; using  Microsoft.EntityFrameworkCore;
using  MvcDemo.Models; namespace  MvcDemo
{       public  class  Startup
        {       public  Startup(IConfiguration  configuration)
                {       Configuration  =  configuration;  }
                public  IConfiguration  Configuration
                {       get;    }
                public  void  ConfigureServices(IServiceCollectionservices)
                {       services.AddMvc(); services.AddDbContext<MvcEmployeeContext>(opt
ions  =>options.UseSqlite("Data  Source=MvcEmployee.db"));
        }
// This method gets called by the runtime. Use th is method to configure the HTTP request
pipeline.
        public  void  Configure(IApplicationBuilder  app,  IH ostingEnvironment  env)
        {       if(env.IsDevelopment())
                {       app.UseDeveloperExceptionPage();        }
                else
                {       app.UseExceptionHandler("/Home/Error");         }
```

```
                    app.UseStaticFiles();

                    app.UseMvc(routes  =>

                    {           routes.MapRoute( name:  "default", template:
                    "{controller=Home}/{action==>{Index}/{id?}");

                    }

          }

}
```

Run the following commands:

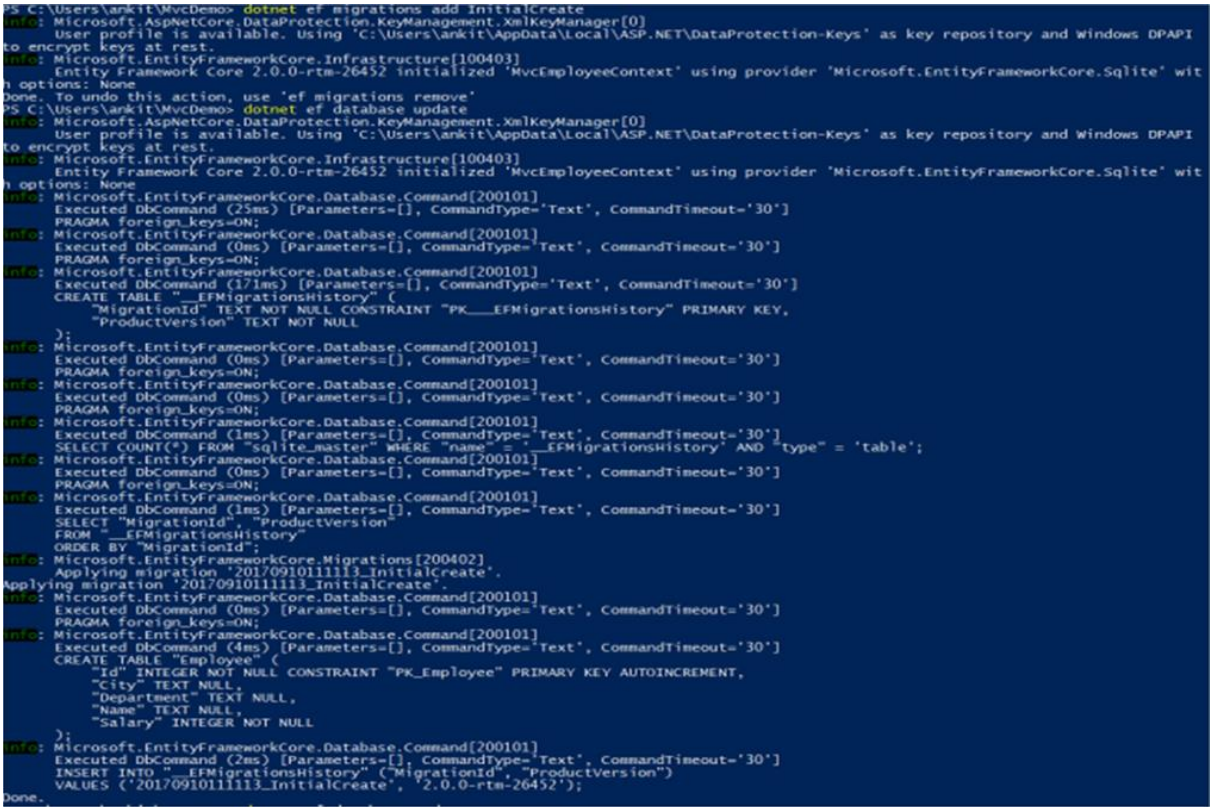
- dotnet restore

dotnet aspnet-codegenerator controller -name EmployeeCont roller -m Employees -dc MvcEmployeeContext --relativeFold erPath Controllers --useDefaultLayout – referenceScriptLibraries

Run the following commands:

- dotnet ef migrations add InitialCreate
- dotnet ef database update

The dotnet ef migrations add InitialCreate command generates code to create the initial database schema. The schema is based on the model specified in the DbContext (in the Models/MVCEmployeeContext.cs file).



```
PS C:\Users\ankit\MvcDemo> dotnet ef migrations add InitialCreate
info: Microsoft.AspNetCore.DataProtection.KeyManagement.XmlKeyManager[0]
      User profile is available. Using 'C:\Users\ankit\AppData\Local\ASP.NET\DataProtection-Keys' as key repository and Windows DPAPI
to encrypt keys at rest.
info: Microsoft.EntityFrameworkCore.Infrastructure[100403]
      Entity Framework Core 2.0.0-rtm-26452 initialized 'MvcEmployeeContext' using provider 'Microsoft.EntityFrameworkCore.Sqlite' wit
h options: None
Done. To undo this action, use 'ef migrations remove'
PS C:\Users\ankit\MvcDemo> dotnet ef database update
info: Microsoft.AspNetCore.DataProtection.KeyManagement.XmlKeyManager[0]
      User profile is available. Using 'C:\Users\ankit\AppData\Local\ASP.NET\DataProtection-Keys' as key repository and Windows DPAPI
to encrypt keys at rest.
info: Microsoft.EntityFrameworkCore.Infrastructure[100403]
      Entity Framework Core 2.0.0-rtm-26452 initialized 'MvcEmployeeContext' using provider 'Microsoft.EntityFrameworkCore.Sqlite' wit
h options: None
info: Microsoft.EntityFrameworkCore.Database.Command[200101]
      Executed DbCommand (25ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      PRAGMA foreign_keys=ON;
info: Microsoft.EntityFrameworkCore.Database.Command[200101]
      Executed DbCommand (0ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      PRAGMA foreign_keys=ON;
info: Microsoft.EntityFrameworkCore.Database.Command[200101]
      Executed DbCommand (171ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      CREATE TABLE "__EFMigrationsHistory" (
          "MigrationId" TEXT NOT NULL CONSTRAINT "PK___EFMigrationsHistory" PRIMARY KEY,
          "ProductVersion" TEXT NOT NULL
      );
info: Microsoft.EntityFrameworkCore.Database.Command[200101]
      Executed DbCommand (0ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      PRAGMA foreign_keys=ON;
info: Microsoft.EntityFrameworkCore.Database.Command[200101]
      Executed DbCommand (0ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      PRAGMA foreign_keys=ON;
info: Microsoft.EntityFrameworkCore.Database.Command[200101]
      Executed DbCommand (1ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      SELECT COUNT(*) FROM "sqlite_master" WHERE "name" = '__EFMigrationsHistory' AND "type" = 'table';
info: Microsoft.EntityFrameworkCore.Database.Command[200101]
      Executed DbCommand (0ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      PRAGMA foreign_keys=ON;
info: Microsoft.EntityFrameworkCore.Database.Command[200101]
      Executed DbCommand (1ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      SELECT "MigrationId", "ProductVersion"
      FROM "__EFMigrationsHistory"
      ORDER BY "MigrationId";
info: Microsoft.EntityFrameworkCore.Migrations[200402]
      Applying migration '20170910111113_InitialCreate'.
Applying migration '20170910111113_InitialCreate'.
info: Microsoft.EntityFrameworkCore.Database.Command[200101]
      Executed DbCommand (0ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      PRAGMA foreign_keys=ON;
info: Microsoft.EntityFrameworkCore.Database.Command[200101]
      Executed DbCommand (4ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      CREATE TABLE "Employee" (
          "Id" INTEGER NOT NULL CONSTRAINT "PK_Employee" PRIMARY KEY AUTOINCREMENT,
          "City" TEXT NULL,
          "Department" TEXT NULL,
          "Name" TEXT NULL,
          "Salary" INTEGER NOT NULL
      );
info: Microsoft.EntityFrameworkCore.Database.Command[200101]
      Executed DbCommand (2ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      INSERT INTO "__EFMigrationsHistory" ("MigrationId", "ProductVersion")
      VALUES ('20170910111113_InitialCreate', '2.0.0-rtm-26452');
Done.
```

After running the second command you will get a message at end "Done." And that's it. We have created our first ASP.NET Core MVC application.

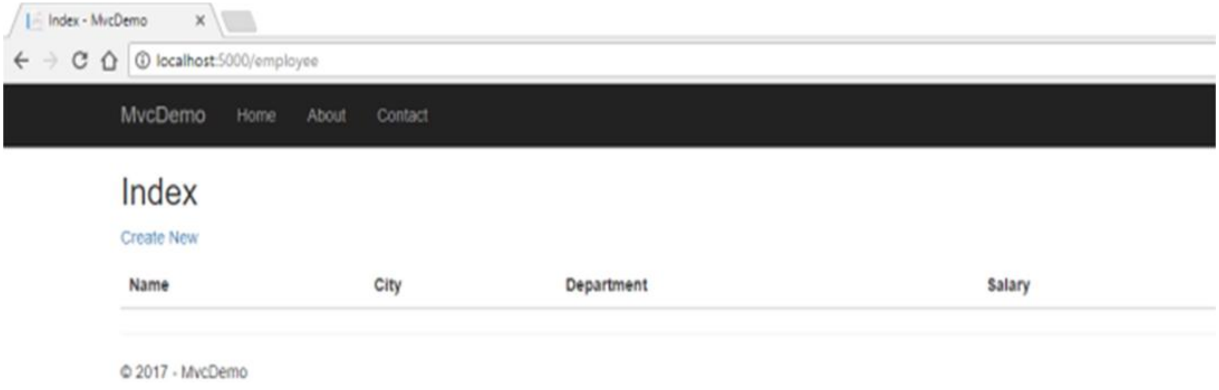
Before running the application, open launch.json and make sure that 'Program' path is set correctly:

"program": "${workspaceRoot}/bin/Debug/netcoreapp2.0/MvcD emo.dll"

Now your launch.json will look like this:

```
{
        // Use IntelliSense to find out which attributes exist for C# debugging

        // Use hover for the description of the existing attri butes

        // For further information visit https://github.com/OmniSharp/omnisharp-
vscode/blob/master/debugger-launchjson. md

"version": "0.2.0", "configurations": [

        "name": ".NET Core Launch (web)",

        "type": "coreclr",

        "request": "launch", "preLaunchTask": "build",

// If you have changed target frameworks, mak e sure to update the program path.

        "program": "${workspaceRoot}/bin/Debug/netcor eapp2.0/MvcDemo.dll", "args": [],
            "cwd":  "${workspaceRoot}", "stopAtEntry":  false, "internalConsoleOptions":
"openOnSessionStart", "launchBrowser":  { "enabled": true, "args": "${auto-detect-url}", "windows":
{           "command":  "cmd.exe",

            "args":  "/C  start  ${auto-detect-url}"

},

"osx":

{           "command":  "open"     },

"linux":

{           "command":  "xdg-open"           }

},

"env":

            "ASPNETCORE_ENVIRONMENT":  "Development"

            },

            "sourceFileMap":

            {           "/Views":  "${workspaceRoot}/Views"    }

            },

            {           "name":  ".NET  Core  Attach", "type": "coreclr",

                        "request": "attach",

                        "processId":  "${command:pickProcess}"

            }

            }

}
```

Output:



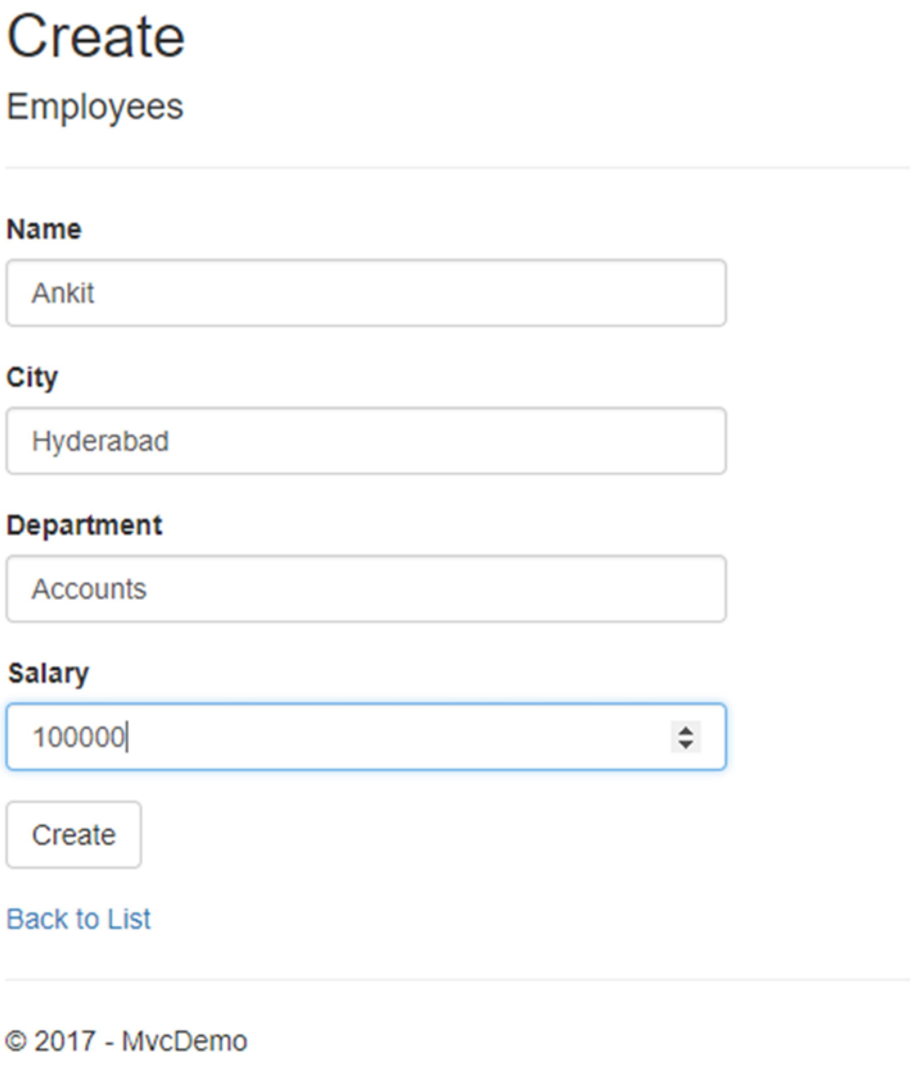
Index

Name	City	Department	Salary

© 2017 - MvcDemo

Now we will proceed with our CRUD operations.

Click on CreateNew to create a new Employee record. Add a new Employee record as shown in the image below.



Create

Employees

Name

Ankit

City

Hyderabad

Department

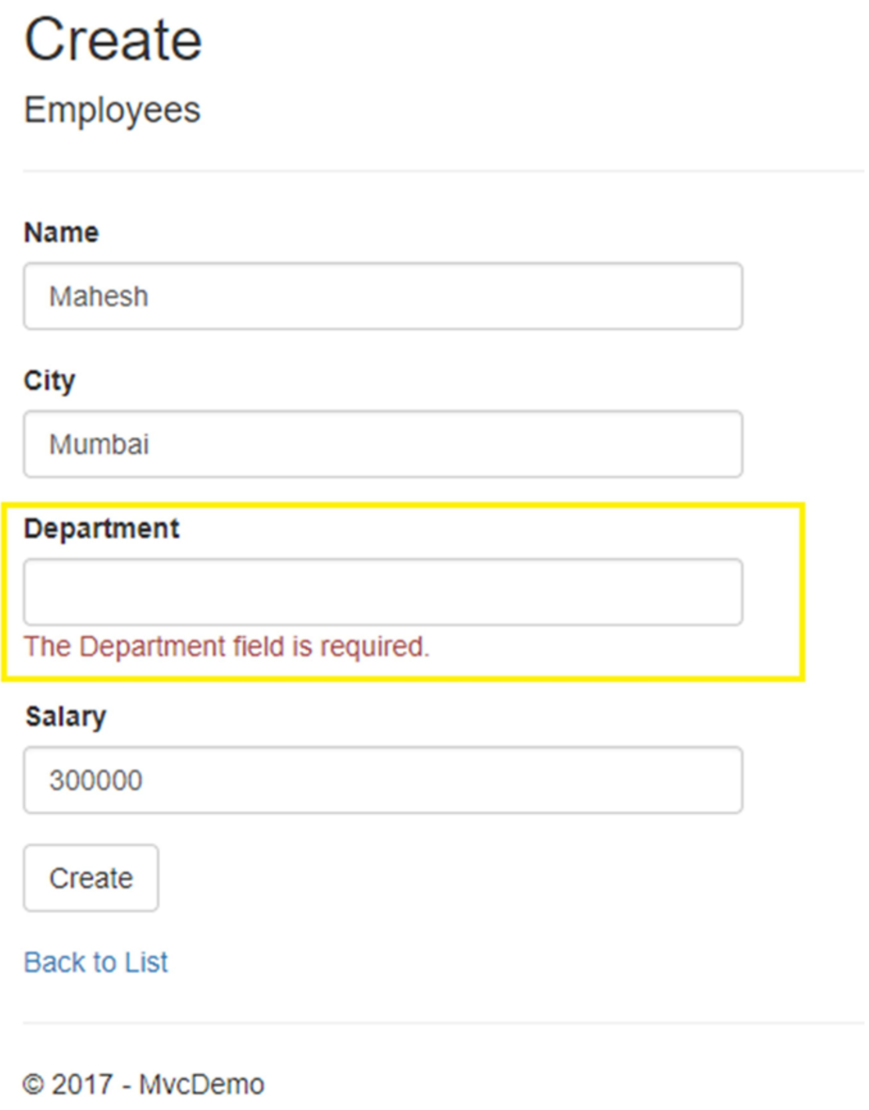
Accounts

Salary

100000

© 2017 - MvcDemo

If we miss data in any fields then we will get a required field validation message.



Create

Employees

Name

Mahesh

City

Mumbai

Department

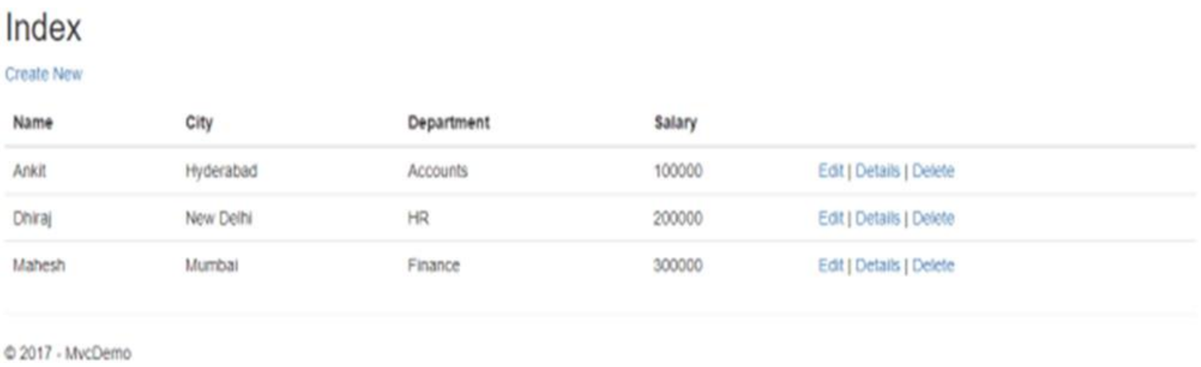
The Department field is required.

Salary

300000

© 2017 - MvcDemo

After You click on the Create button, in Create View it will redirect us to the Index view where we can see all the employees added by us. Here, we can also see the action methods Edit, Details, and Delete.

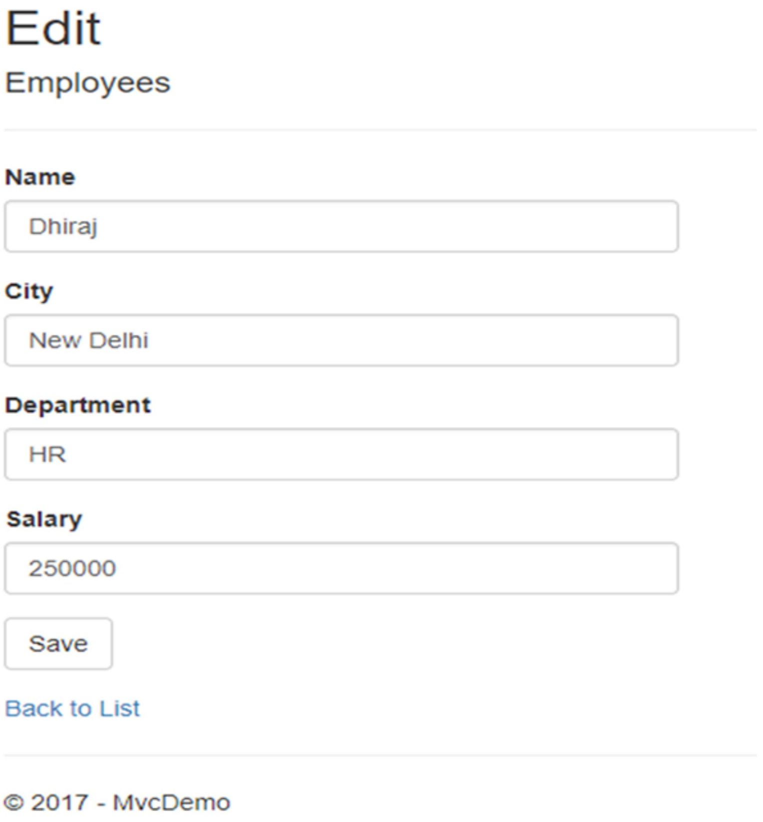


Index

Name	City	Department	Salary	
Ankit	Hyderabad	Accounts	100000	Edit | Details | Delete
Dhiraj	New Delhi	HR	200000	Edit | Details | Delete
Mahesh	Mumbai	Finance	300000	Edit | Details | Delete

© 2017 - MvcDemo

If we want to edit any existing employee records, then click the Edit action link. This will open the Edit view, as shown below, where we can change the employee data.



Edit

Employees

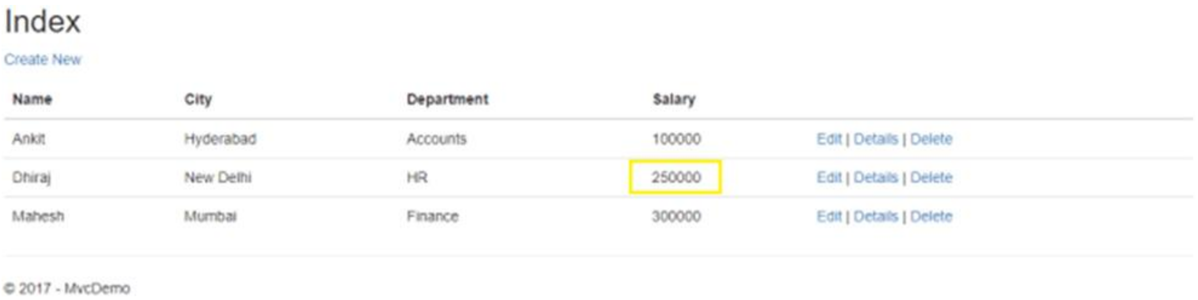
Name: Dhiraj

City: New Delhi

Department: HR

Salary: 250000

Here, we have changed the Salary of the employee with name Dhiraj from 200000 to 250000. Click on Save to return to the Index view and see the updated changes as highlighted in the image below.

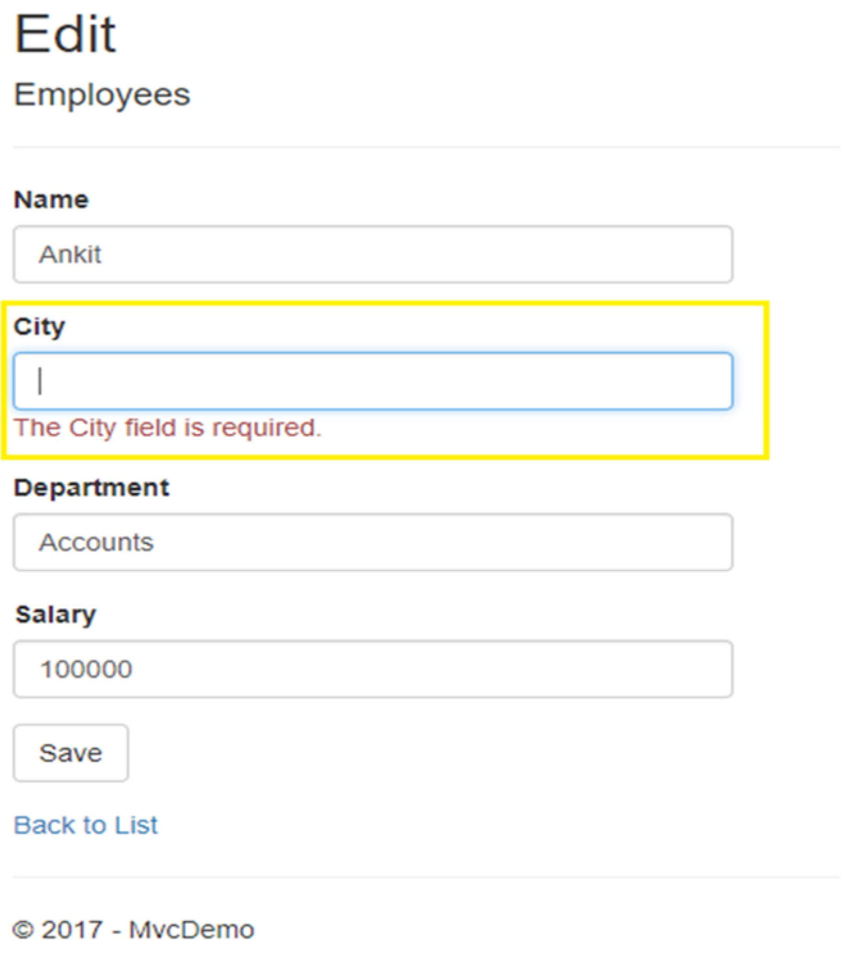


Index

Create New

Name	City	Department	Salary	
Ankit	Hyderabad	Accounts	100000	Edit | Details | Delete
Dhiraj	New Delhi	HR	250000	Edit | Details | Delete
Mahesh	Mumbai	Finance	300000	Edit | Details | Delete

If we miss any fields while editing employee records, then the edit view will also throw the required field validation error message.



Edit

Employees

Name

Ankit

City

The City field is required.

Department

Accounts

Salary

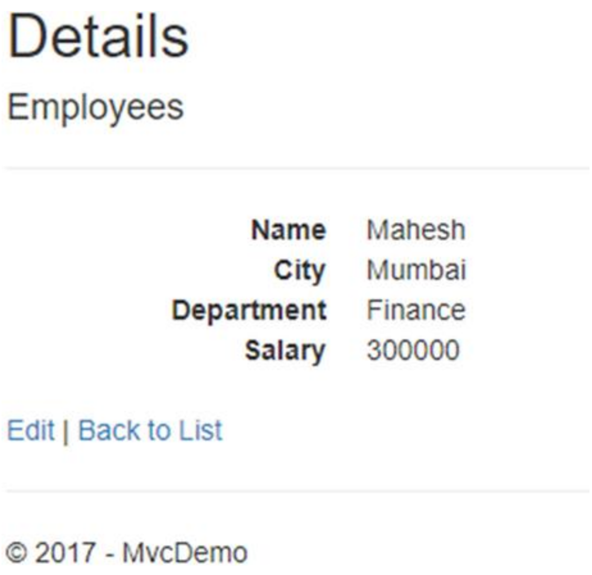
100000

Save

Back to List

© 2017 - MvcDemo

If you want to see the details of any Employee, then click on the Details action link, which will open the Details view as shown in the image below.



Details

Employees

Name Mahesh

City Mumbai

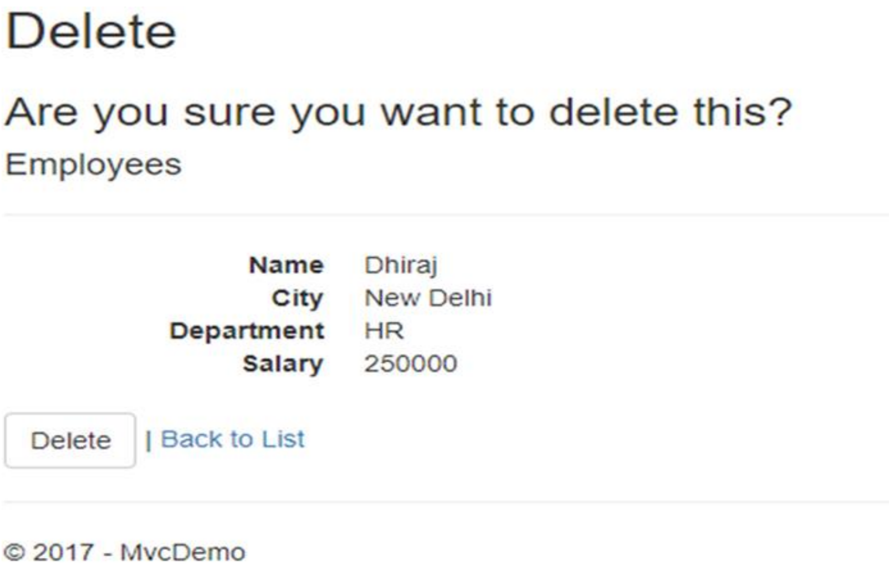
Department Finance

Salary 300000

Edit | Back to List

© 2017 - MvcDemo

Click on Back to List to go back to the Index view. Now, we will perform the Delete operation on the Employee with the name Dhiraj. Click on the Delete action link which will open the Delete view and ask for a confirmation to delete.



Delete

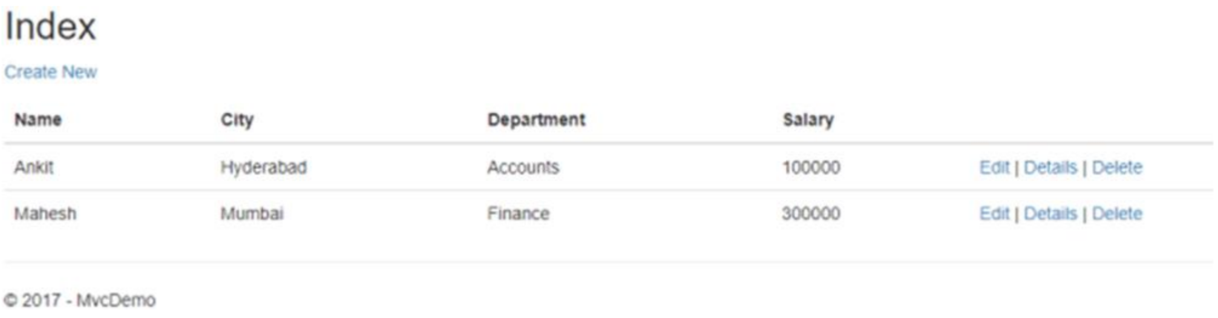
Are you sure you want to delete this?

Employees

Name	Dhiraj
City	New Delhi
Department	HR
Salary	250000

© 2017 - MvcDemo

Once we click on the Delete button, the employee record gets deleted and we will be redirected to the Index view. Here, we can see that the employee with the name Dhiraj has been removed from our record.



Index

Name	City	Department	Salary	
Ankit	Hyderabad	Accounts	100000	Edit | Details | Delete
Mahesh	Mumbai	Finance	300000	Edit | Details | Delete

© 2017 - MvcDemo

EmployeeController.cs

```
// GET: Employee/Delete/5

public async Task<IActionResult> Delete(int? id)

{         if (id == null)

          {         return NotFound();      }

                    var employees = await _context.Employee

                    .SingleOrDefaultAsync(m => m.Id == id); if (employees == null)

                    {         return NotFound();      }

                    return View(employees);

          }
```

```
 // POST: Employee/Delete/5 [HttpPost, ActionName("Delete")] [ValidateAntiForgeryToken]
public async Task<IActionResult> DeleteConfirmed(int id)
{         var employees = await _context.Employee.Single
          OrDefaultAsync(m => m.Id == id);
          _context.Employee.Remove(employees);
          await _context.SaveChangesAsync();
          return RedirectToAction(nameof(Index));
}
```