Practical No 6

1. Create an MVC application to demonstrate ViewBag, ViewData, TempData, Layouts and partial views.

HomeController.cs

```
using System.Diagnostics; using
Microsoft.AspNetCore.Mvc; using
MyFirstMVC.Models;


namespace MyFirstMVC.Controllers

{

   public class HomeController : Controller     {

      public ViewResult Index()        {

         int hour=DateTime.Now.Hour;            string msg = hour < 12 ?
"Good Morning" : "Good Afternoon";            return View("MyView",msg);

      }

      public IActionResult ViewBagDemo()        {

         ViewBag.Message = "Hello from ViewBag";

         ViewBag.CurrentDate=DateTime.Now.ToLongDateString();


         return View();

      }

      public IActionResult ViewDataDemo()

      {

         ViewData["Message"] = "Hello From ViewData";

         ViewData["Date"]=DateTime.Now.ToLongDateString();


         var colors=new List<string> { "Maroon", "Green" ,"Red"};

         ViewData["colors"]=colors;


         return View();

      }

      public IActionResult Submit()

      {

         TempData["SuccesMessage"] = "Form submitted succesfully !";
```

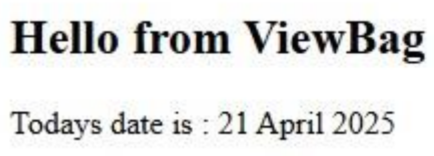
```
        return RedirectToAction("Confirmation");
        }
        public IActionResult Confirmation()
        {           return
View();
        }    } }
```

ViewBagDemo.cshtml

```
@{
    Layout = null;
}

<!DOCTYPE html>

<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <title>ViewBagDemo</title>

</head>
<body>
    <h2>@ViewBag.Message</h2>
    <p>Todays date is : @ViewBag.CurrentDate</p>
</body>
</html>
```

Program.cs

```
namespace MyFirstMVC
{
    public class Program
    {
        public static void Main(string[] args)
        {
            var builder = WebApplication.CreateBuilder(args);

            // Add services to the container.
            builder.Services.AddControllersWithViews();

            var app = builder.Build();

            // Configure the HTTP request pipeline.
if (!app.Environment.IsDevelopment())
            {
                app.UseExceptionHandler("/Home/Error");
                // The default HSTS value is 30 days. You may want to change this for production scenarios,
see https://aka.ms/aspnetcore-hsts.
                app.UseHsts();
            }

            app.UseHttpsRedirection();
app.UseStaticFiles();
            app.UseRouting();
            app.UseAuthorization();
            app.MapControllerRoute(
    name: "default",
 pattern: "{controller=Home}/{action=ViewBagDemo}/{id?}");
            app.Run();
        }
    }
}
```



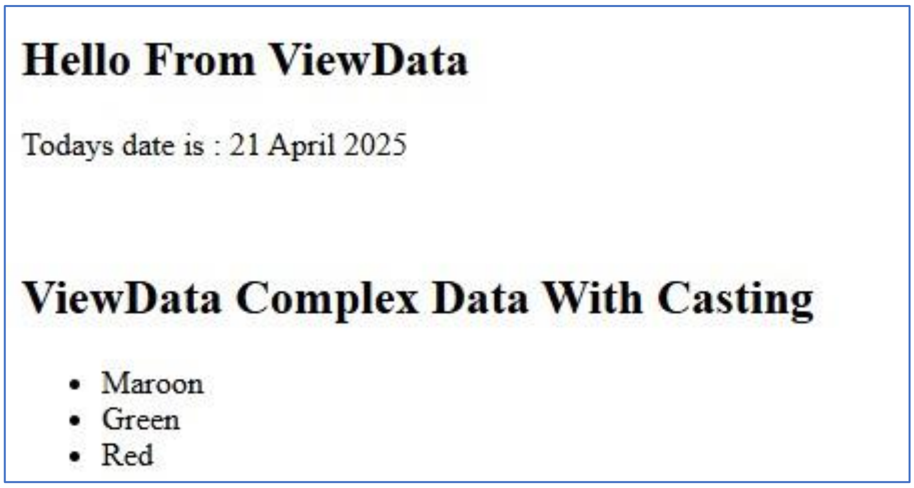
```
Hello from ViewBag

Todays date is : 21 April 2025
```

ViewDataDemo.cshtml

```
@{
    ViewData["Title"] = "ViewData Demo";
    Layout = "~/Views/Shared/_Layout1.cshtml";
}

<!DOCTYPE html>

<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <title>ViewDataDemo</title>
</head>
<body>
    <h2>@ViewData["Message"]</h2>
    <p>Todays date is : @ViewData["Date"]</p>
    <br />
    <h2>ViewData Complex Data With Casting</h2>
    @{        var colors = ViewData["colors"] as
List<string>;
    }
    <ul>
        @foreach (var color in colors)
        {
            <li>@color</li>
        }
    </ul>
</body>
</html>
```

Program.cs

```
namespace MyFirstMVC
{
   public class Program
   {
      public static void Main(string[] args)
      {
         var builder = WebApplication.CreateBuilder(args);

         // Add services to the container.
         builder.Services.AddControllersWithViews();

         var app = builder.Build();

         // Configure the HTTP request pipeline.
if (!app.Environment.IsDevelopment())
         {
            app.UseExceptionHandler("/Home/Error");
            // The default HSTS value is 30 days. You may want to change this for production scenarios,
see https://aka.ms/aspnetcore-hsts.                app.UseHsts();
         }

         app.UseHttpsRedirection();
app.UseStaticFiles();

         app.UseRouting();

         app.UseAuthorization();

         app.MapControllerRoute(                name: "default",                pattern:
"{controller=Home}/{action=ViewDataDemo}/{id?}");

         app.Run();
      }    } }
```



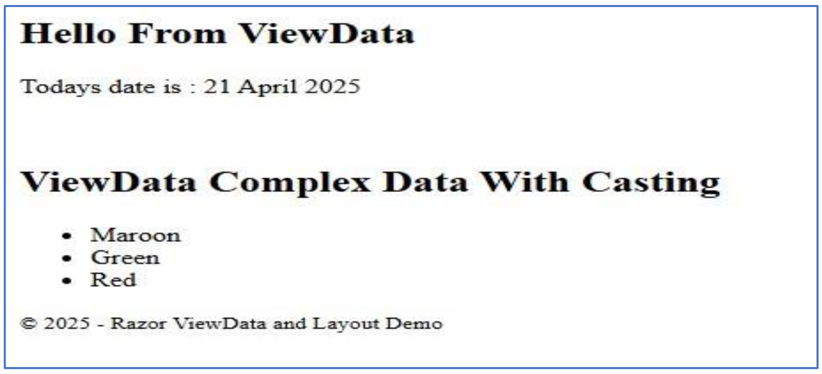
```
Hello From ViewData

Todays date is : 21 April 2025


ViewData Complex Data With Casting

 • Maroon
 • Green
 • Red
```

Layout.cshtml

```
<!DOCTYPE html>

<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <title>@ViewData["title"] and Layout Demo</title>
</head>
<body>
    <div class="container mt-4">
        @RenderBody()
    </div>
    <footer>
        <small>&copy; @DateTime.Now.Year - Razor ViewData and Layout Demo</small>
</footer>
</body>
</html>
```

ViewDataDemo.cshtml

```
@{
    ViewData["Title"] = "ViewData Demo";
    Layout = "~/Views/Shared/_Layout1.cshtml";
}

<!DOCTYPE html>

<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <title>ViewDataDemo</title>
</head>
<body>
    <h2>@ViewData["Message"]</h2>
    <p>Todays date is : @ViewData["Date"]</p>
    <br />
    <h2>ViewData Complex Data With Casting</h2>
    @{          var colors = ViewData["colors"] as
List<string>;
    }
    <ul>
        @foreach (var color in colors)
        {
            <li>@color</li>
        }
    </ul>
</body>
</html>
```


Hello From ViewData

Todays date is : 21 April 2025

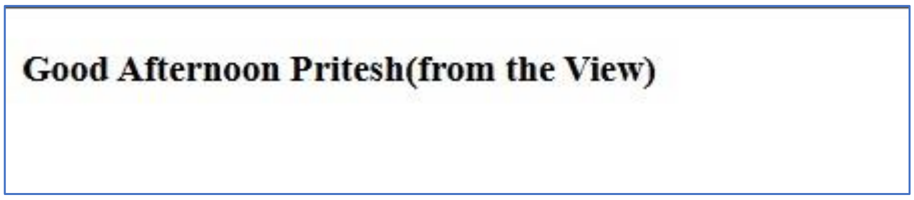
ViewData Complex Data With Casting

- Maroon
- Green
- Red

© 2025 - Razor ViewData and Layout Demo

MyView.cshtml

```
@model string

@{
    Layout = null;
}

<!DOCTYPE html>

<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <title>MyView</title>
</head>
<body>
    <h3>@Model Pritesh(from the View)</h3>
</body>
</html>
```


Good Afternoon Pritesh(from the View)

2. Create an MVC application to accept Customer details and display the same using views. Use automatically implemented properties, Tag helpers and apply validation.

HomeControllers.cs

```
using Microsoft.AspNetCore.Mvc;
using CustomerMVC.Models;
namespace CustomerMVC.Controllers
{
  public class HomeController : Controller
  {
    [HttpGet]
    public IActionResult Create()
    {
      return View();
    }
    [HttpPost]
    public IActionResult Create(Customer customer)
    {
      if (ModelState.IsValid)
      {
        return View("Details", customer);
      }
      return View(customer);
    }

    public IActionResult Details(Customer customer)
    {
      return View(customer);
    }

    public IActionResult Index()
    {
            return View();
        }
    }
}
```

Customers.cs

```
using System.ComponentModel.DataAnnotations
namespace CustomerMVC.Models
{
    public class Customer
    {
        public int Id { get; set;
        [Required(ErrorMessage = "Name is required")]
        [StringLength(50)]
        public string Name { get; set;
        [Required(ErrorMessage = "Email is required")]
        [EmailAddress]
        public string Email { get; set;
        [Required(ErrorMessage = "Phone number is required")]
        [Phone]
        public string Phone { get; set;
        [Range(18, 100, ErrorMessage = "Age must be between 18 and 100")]
        public int Age { get; set; }
    }
}
```

Create.cshtml

```
@model CustomerMVC.Models.Customer
@{
    Layout = null;
}
<!DOCTYPE html>
```

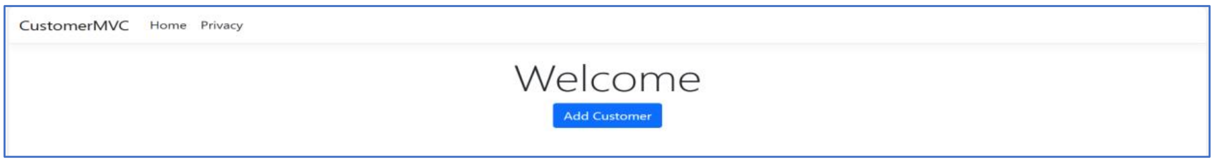
```
<html>
<head>
  <meta name="viewport" content="width=device-width" />
  <title>Create</title>
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet" />
</head>
<body>
  <div class="container mt-4">
    <h2>Add Customer</h2>
    <form asp-action="Create" method="post">
      <div asp-validation-summary="ModelOnly" class="text-danger"></div>
      <div class="form-group mb-2">
        <label asp-for="Name"></label>
        <input asp-for="Name" class="form-control" />
        <span asp-validation-for="Name" class="text-danger"></span>
      </div
      <div class="form-group mb-2">
        <label asp-for="Email"></label>
        <input asp-for="Email" class="form-control" />
        <span asp-validation-for="Email" class="text-danger"></span>
      </div
      <div class="form-group mb-2">
        <label asp-for="Phone"></label>
        <input asp-for="Phone" class="form-control" />
        <span asp-validation-for="Phone" class="text-danger"></span>
      </div
      <div class="form-group mb-3">
        <label asp-for="Age"></label>
        <input asp-for="Age" class="form-control" />
        <span asp-validation-for="Age" class="text-danger"></span>
      </div>
      <button type="submit" class="btn btn-primary">Submit</button>
```

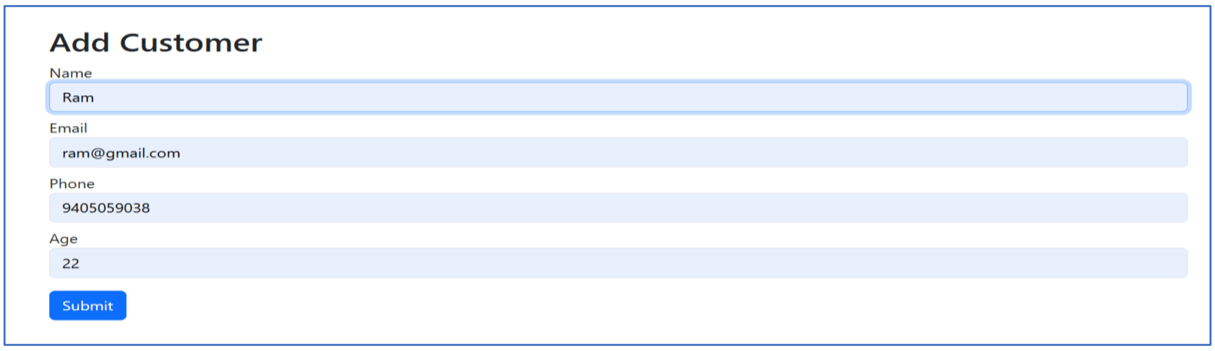
```
        </form>
    </div>
</body>
</html>
```

Details.cshtml

```
@model CustomerMVC.Models.Customer
@{
    Layout = null;
}
<!DOCTYPE html>
<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <title>Details</title>
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.0/dist/css/bootstrap.min.css"
rel="stylesheet" />
</head>
<body>
    <div class="container mt-4">
        <h2>Customer Details</h2>
        <ul class="list-group">
            <li class="list-group-item"><strong>Name:</strong> @Model.Name</li>
            <li class="list-group-item"><strong>Email:</strong> @Model.Email</li>
            <li class="list-group-item"><strong>Phone:</strong> @Model.Phone</li>
            <li class="list-group-item"><strong>Age:</strong> @Model.Age</li>
        </ul>
    </div>
</body>
</html>
```

Output:




Add Customer

Name

Ram

Email

ram@gmail.com

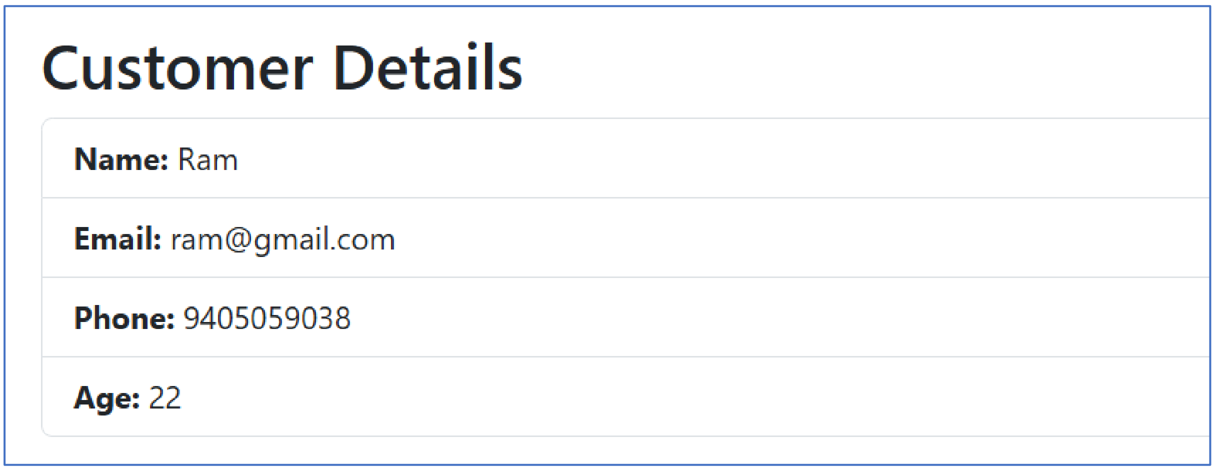
Phone

9405059038

Age

22

Submit


Customer Details

Name: Ram

Email: ram@gmail.com

Phone: 9405059038

Age: 22

3. Create Employee Database application using MVC Core and Entity Framework Core (Code-First

Approach).

Employee.cs

```
using System.ComponentModel.DataAnnotations;

namespace MVCEntityFrameworkCodeFirst.Models

{    public class Employee    {

        public int Id { get; set; }

        [Required]
        public string Name { get; set;
        [Required, EmailAddress]
        public string Email { get; set; }
        public string Department { get; set;
        [Range(10000, 100000)]
        public decimal Salary { get; set; }
    }}
```

AppDbContext.cs

```
using MVCEntityFrameworkCodeFirst.Models;
using Microsoft.EntityFrameworkCore;
namespace MVCEntityFrameworkCodeFirst.Data
{
    public class AppDbContext:DbContext
    {
        public AppDbContext(DbContextOptions<AppDbContext> options): base(options) { }
        public DbSet<Employee> Employees { get; set; }
        protected override void OnModelCreating(ModelBuilder modelBuilder)
        {
            // Disable plural table names globally
            foreach (var entity in modelBuilder.Model.GetEntityTypes())
            {
                entity.SetTableName(entity.DisplayName());
            }       }    }}
```

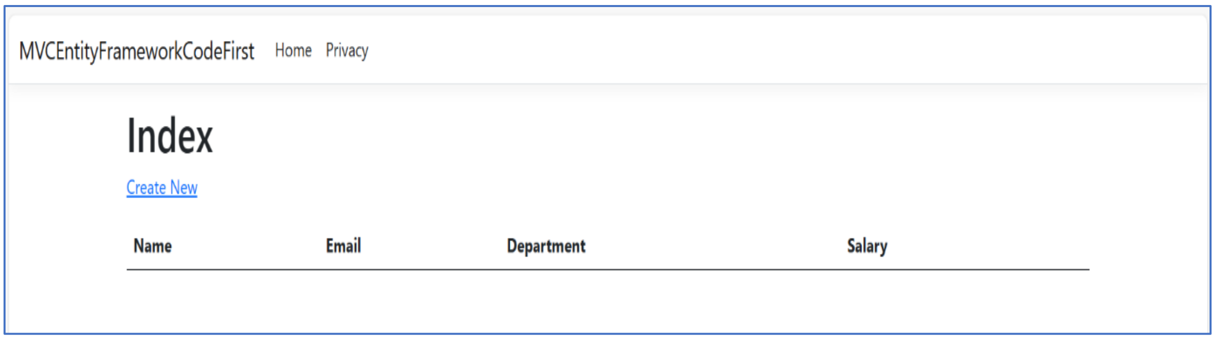
Appsetting.json

```
{
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*",
  "ConnectionStrings": {
    "DefaultConnection":
"Server=(localdb)\\mssqllocaldb;Database=EmployeeDb;Trusted_Connection=True;"
  }
}
```

Program.cs

```
using Microsoft.EntityFrameworkCore;
using MVCEntityFrameworkCodeFirst.Data;
sing MVCEntityFrameworkCodeFirst.Models;
var builder = WebApplication.CreateBuilder(args);
// Add services to the container.
builder.Services.AddControllersWithViews();
builder.Services.AddDbContext<AppDbContext>(options =>
options.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection")));
var app = builder.Build();
// Configure the HTTP request pipeline.
if (!app.Environment.IsDevelopment())
{
    app.UseExceptionHandler("/Home/Error");
    // The default HSTS value is 30 days. You may want to change this for production scenarios, see
https://aka.ms/aspnetcore-hsts.
    app.UseHsts()
app.UseHttpsRedirection();
app.UseStaticFiles();
app.UseRouting();
app.UseAuthorization()
app.MapControllerRoute(
    name: "default",
    pattern: "{controller=Home}/{action=Index}/{id?}");
app.Run()
```

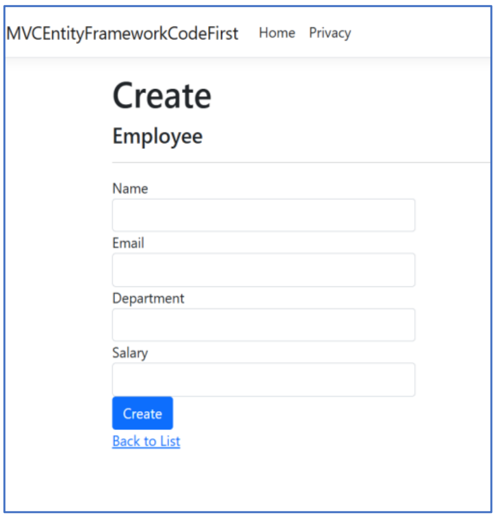
Output


MVCEntityFrameworkCodeFirst Home Privacy

Index

Create New

Name	Email	Department	Salary


MVCEntityFrameworkCodeFirst Home Privacy

Create

Employee

Name

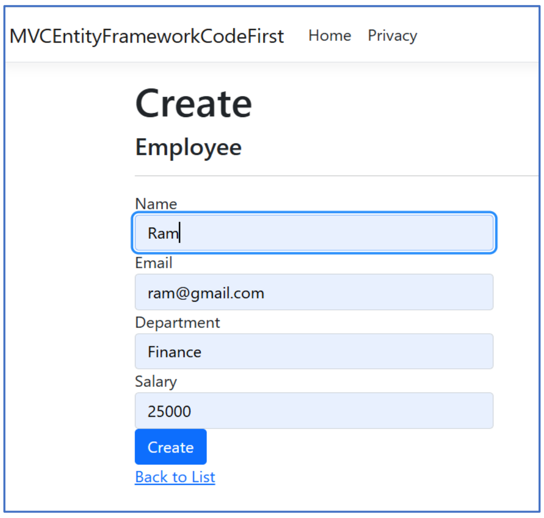
Email

Department

Salary

Create

Back to List


MVCEntityFrameworkCodeFirst Home Privacy

Create

Employee

Name: Ram

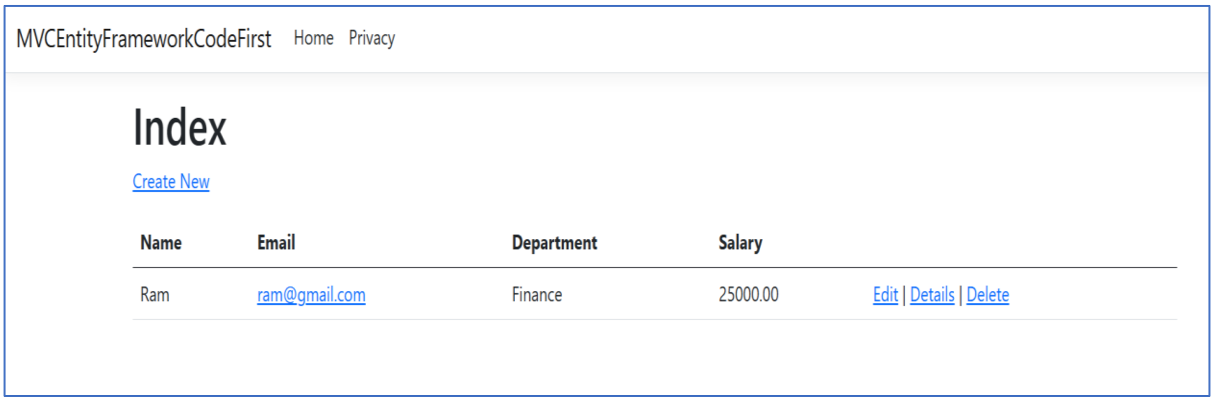
Email: ram@gmail.com

Department: Finance

Salary: 25000

Create

Back to List


MVCEntityFrameworkCodeFirst Home Privacy

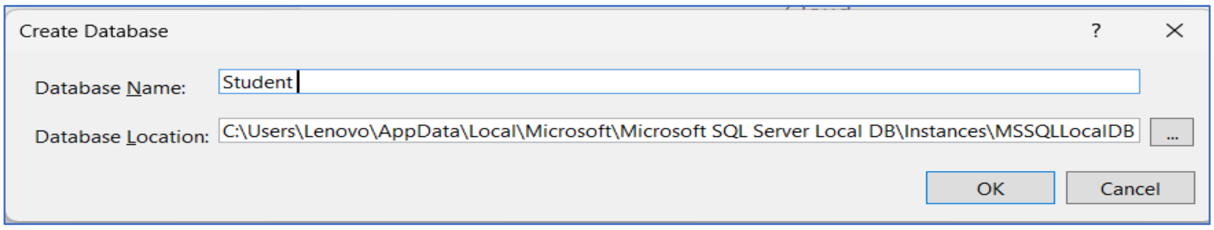
Index

Create New

Name	Email	Department	Salary	
Ram	ram@gmail.com	Finance	25000.00	Edit | Details | Delete

4. Create Student Database application using MVC Core and Entity Framework Core (Database-First

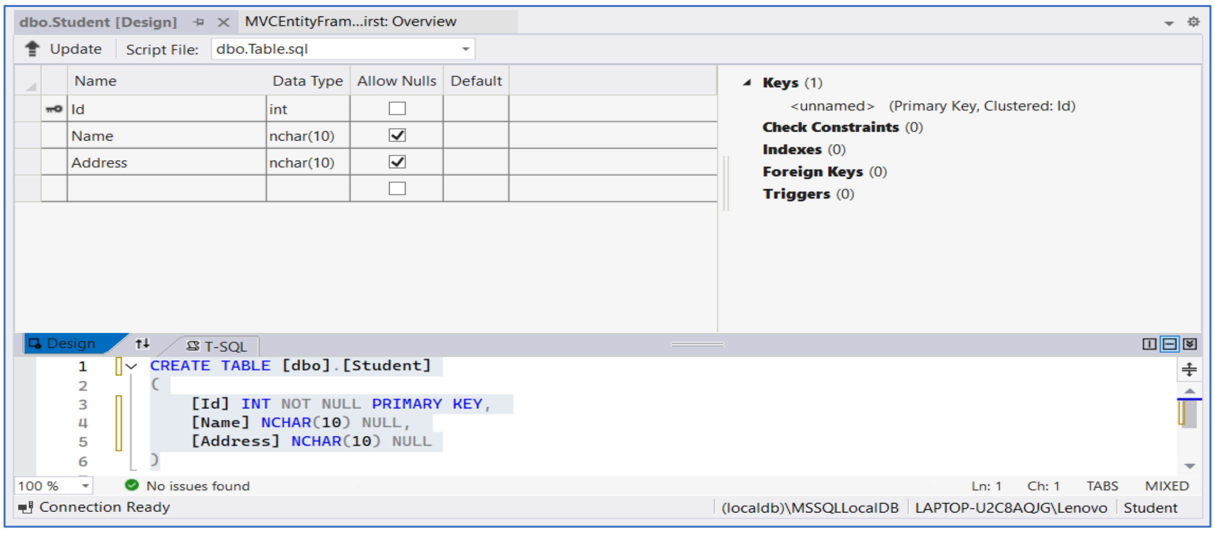
Approach).



Create Database

Database Name: Student

Database Location: C:\Users\Lenovo\AppData\Local\Microsoft\Microsoft SQL Server Local DB\Instances\MSSQLLocalDB



Script File: dbo.Table.sql

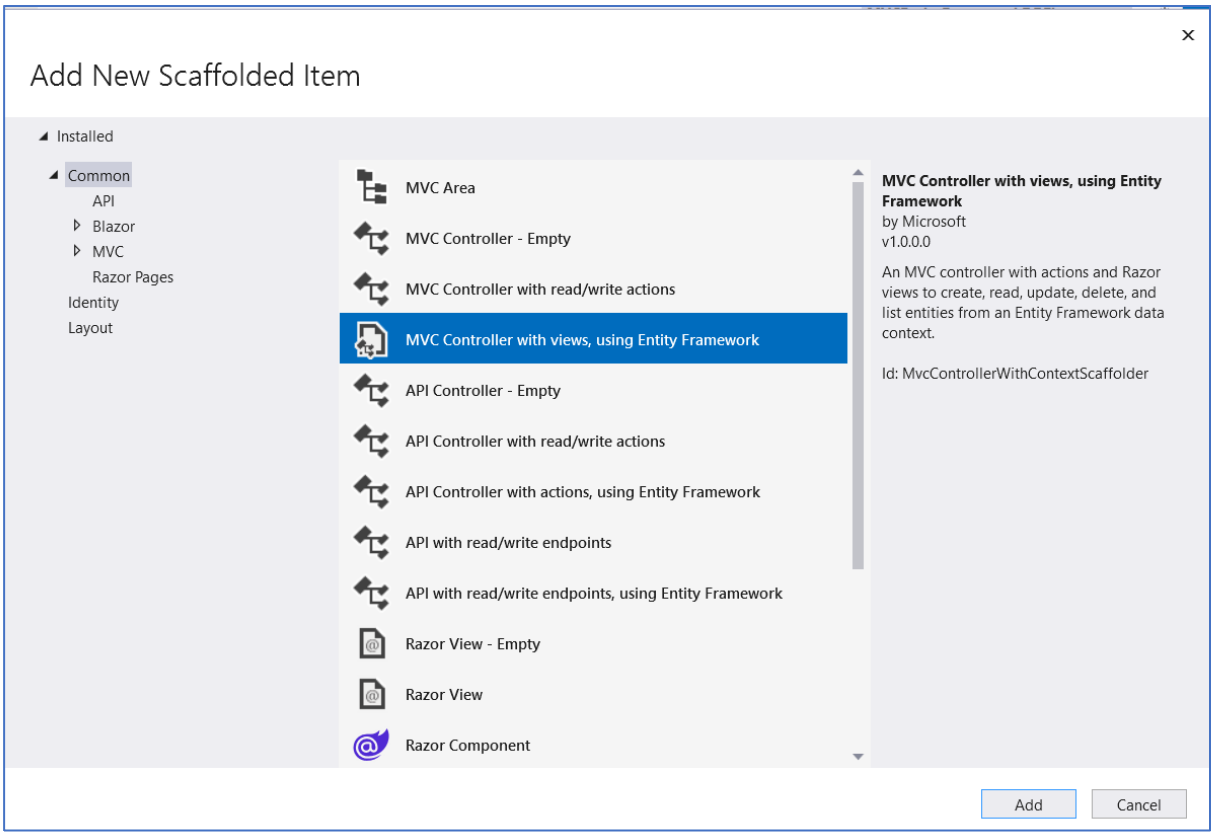
Name	Data Type	Allow Nulls	Default
Id	int	☐	
Name	nchar(10)	✓	
Address	nchar(10)	✓	
		☐	

Keys (1)
<unnamed> (Primary Key, Clustered: Id)
Check Constraints (0)
Indexes (0)
Foreign Keys (0)
Triggers (0)

```
CREATE TABLE [dbo].[Student]
(
    [Id] INT NOT NULL PRIMARY KEY,
    [Name] NCHAR(10) NULL,
    [Address] NCHAR(10) NULL
)
```

100 % No issues found Ln: 1 Ch: 1 TABS MIXED

Connection Ready (localdb)\MSSQLLocalDB LAPTOP-U2C8AQJG\Lenovo Student



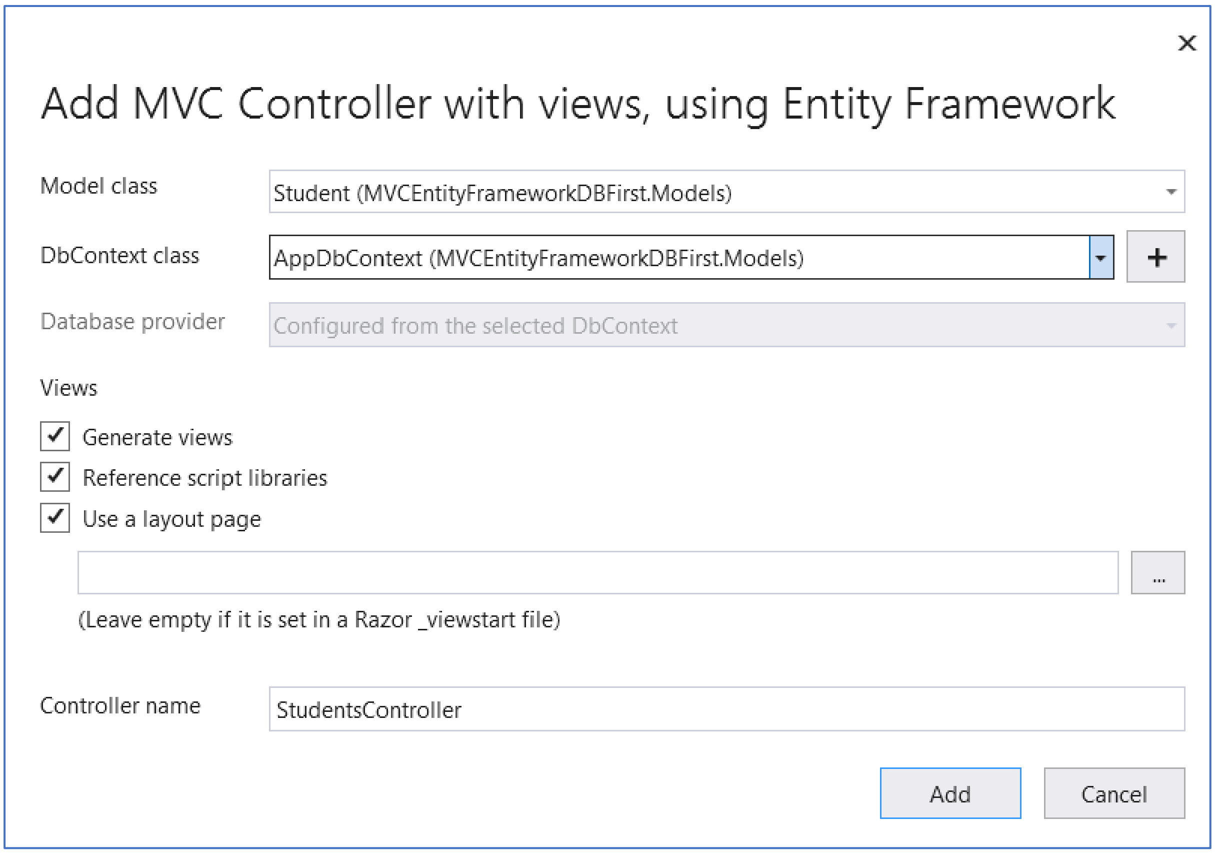
Add New Scaffolded Item

Installed
Common
API
Blazor
MVC
Razor Pages
Identity
Layout
MVC Area
MVC Controller - Empty
MVC Controller with read/write actions
MVC Controller with views, using Entity Framework
API Controller - Empty
API Controller with read/write actions
API Controller with actions, using Entity Framework
API with read/write endpoints
API with read/write endpoints, using Entity Framework
Razor View - Empty
Razor View
Razor Component
Add
Cancel

MVC Controller with views, using Entity Framework
by Microsoft
v1.0.0.0

An MVC controller with actions and Razor views to create, read, update, delete, and list entities from an Entity Framework data context.

Id: MvcControllerWithContextScaffolder


Add MVC Controller with views, using Entity Framework

Model class: Student (MVCEntityFrameworkDBFirst.Models)

DbContext class: AppDbContext (MVCEntityFrameworkDBFirst.Models)

Database provider: Configured from the selected DbContext

Views

[✓] Generate views

[✓] Reference script libraries

[✓] Use a layout page

(Leave empty if it is set in a Razor _viewstart file)

Controller name: StudentsController

Code:

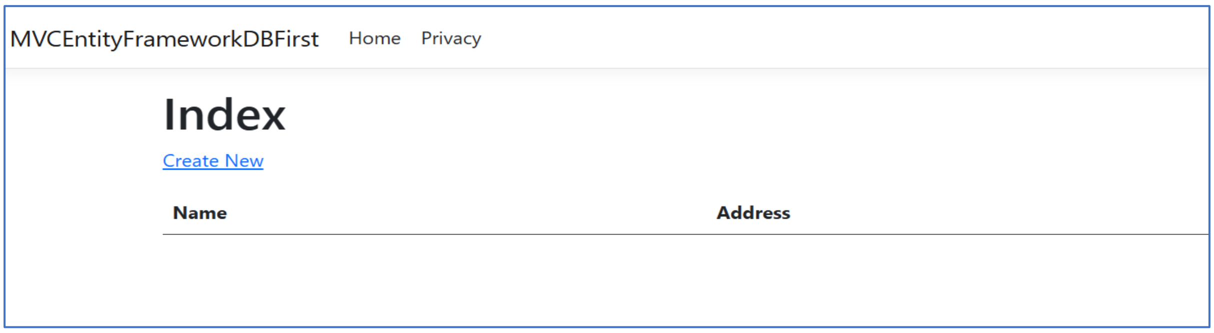
AppDbContext

```
using System;
using System.Collections.Generic;
using Microsoft.EntityFrameworkCore;
namespace MVCEntityFrameworkDBFirst.Models
public partial class AppDbContext : DbContext
{
    public AppDbContext()
    {
    }
    public AppDbContext(DbContextOptions<AppDbContext> options)
        : base(options)
    {
    public virtual DbSet<Student> Students { get; set; }
    protected override void OnConfiguring(DbContextOptionsBuilder optionsBuilder)
#warning To protect potentially sensitive information in your connection string, you should move it out of source code. You can avoid scaffolding the connection string by using the Name= syntax to read it from configuration - see https://go.microsoft.com/fwlink/?linkid=2131148. For more guidance on storing connection strings, see https://go.microsoft.com/fwlink/?LinkId=723263.
        =>
optionsBuilder.UseSqlServer("Server=(localdb)\\MSSQLLocalDB;Database=Student;Trusted_Connection=True;")
    protected override void OnModelCreating(ModelBuilder modelBuilder)
    {
        modelBuilder.Entity<Student>(entity =>
        {
            entity.HasKey(e => e.Id).HasName("PK__Student__3214EC07658FBB1A")
            entity.ToTable("Student")
            entity.Property(e => e.Id).ValueGeneratedNever();
            entity.Property(e => e.Address)
                .HasMaxLength(10)
                .IsFixedLength();
            entity.Property(e => e.Name)
                .HasMaxLength(10)
                .IsFixedLength();
        });
        OnModelCreatingPartial(modelBuilder);
    }
    partial void OnModelCreatingPartial(ModelBuilder modelBuilder);
}
```

Program.cs

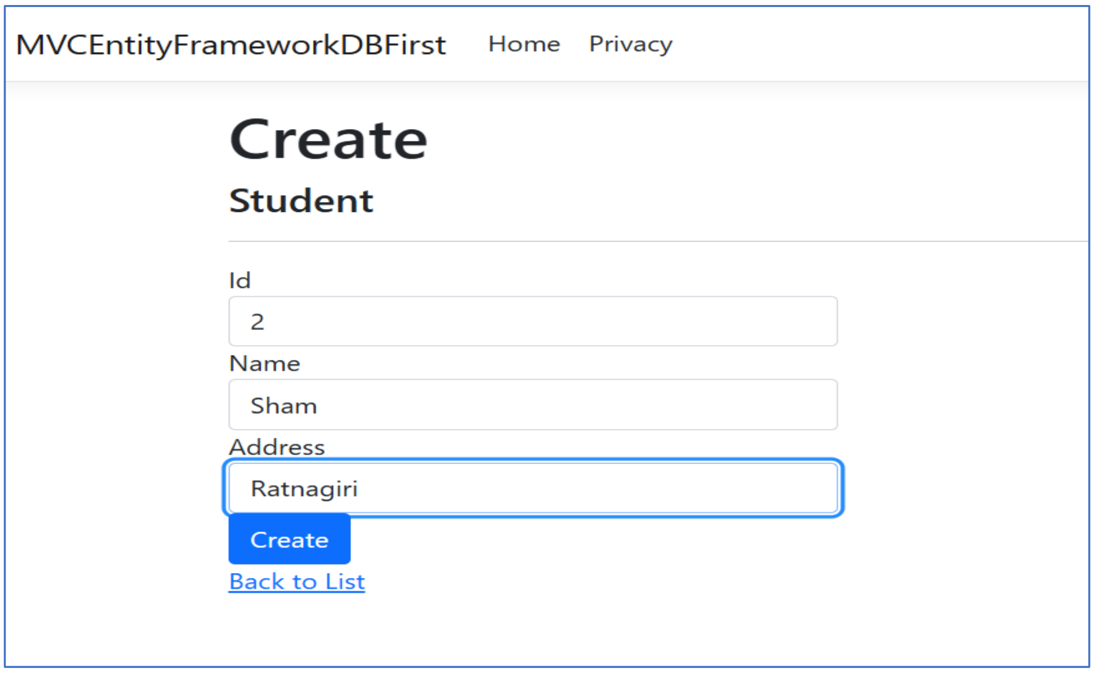
```
using Microsoft.EntityFrameworkCore;
using MVCEntityFrameworkDBFirst.Models;
var builder = WebApplication.CreateBuilder(args);
// Add services to the container.
builder.Services.AddControllersWithViews();
builder.Services.AddDbContext<AppDbContext>(options =>
options.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection")))

var app = builder.Build();

// Configure the HTTP request pipeline.

if (!app.Environment.IsDevelopment())

{

    app.UseExceptionHandler("/Home/Error");

    // The default HSTS value is 30 days. You may want to change this for production scenarios, see https://aka.ms/aspnetcore-hsts.

    app.UseHsts();

}

app.UseHttpsRedirection();

app.UseStaticFiles()

app.UseRouting();

app.UseAuthorization();

app.MapControllerRoute(

    name: "default",

    pattern: "{controller=Home}/{action=Index}/{id?}")

app.Run();
```

Output:



Index

Name	Address



MVCEntityFrameworkDBFirst Home Privacy

Create

Student

Id

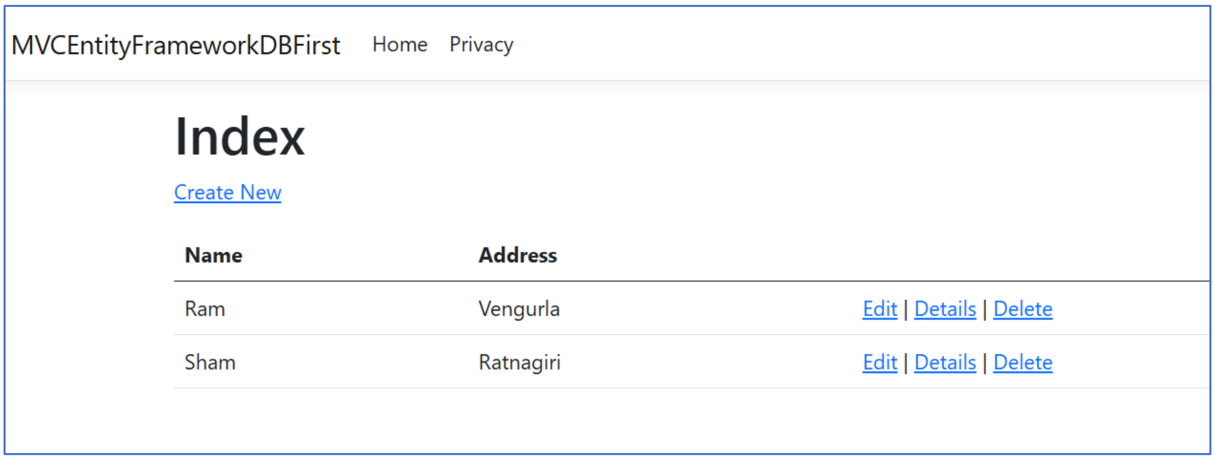
2

Name

Sham

Address

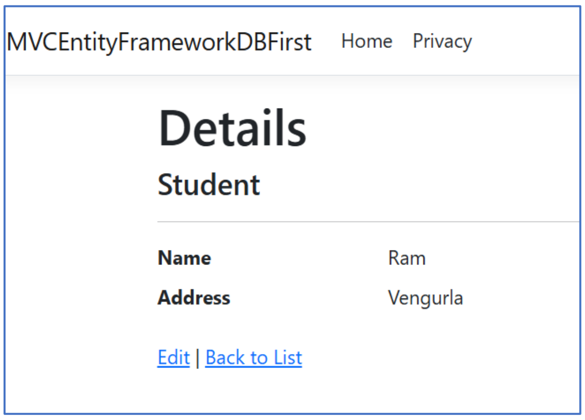
Ratnagiri



MVCEntityFrameworkDBFirst Home Privacy

Index

Name	Address	
Ram	Vengurla	Edit | Details | Delete
Sham	Ratnagiri	Edit | Details | Delete

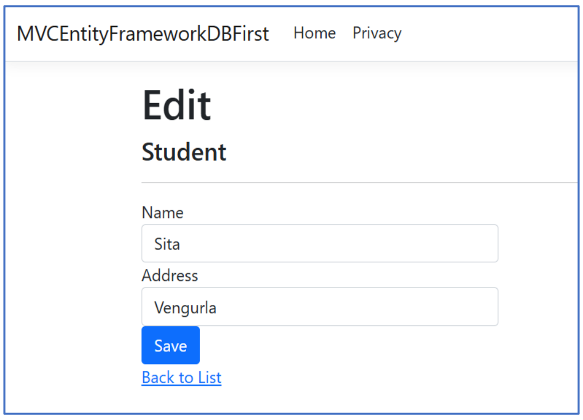


MVCEntityFrameworkDBFirst Home Privacy

Details

Student

Name	Ram
Address	Vengurla



MVCEntityFrameworkDBFirst Home Privacy

Edit

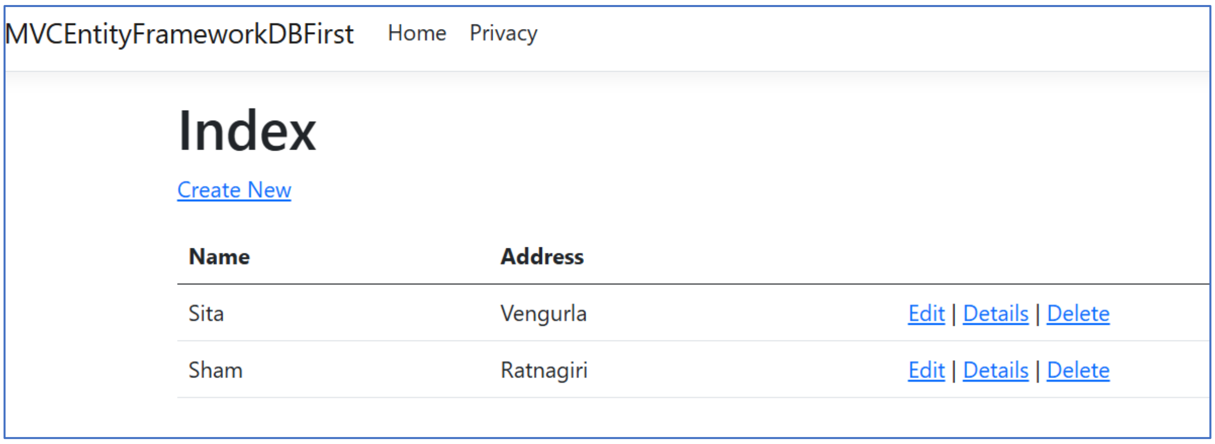
Student

Name

Sita

Address

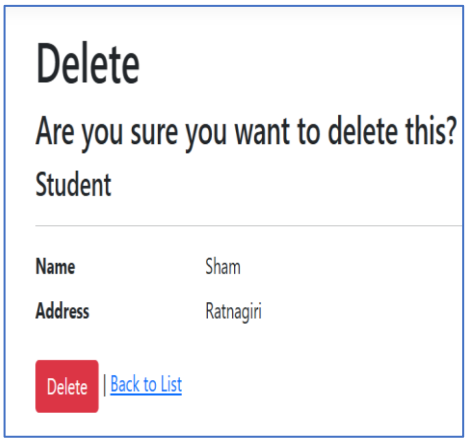
Vengurla


MVCEntityFrameworkDBFirst Home Privacy

Index

Create New

Name	Address	
Sita	Vengurla	Edit | Details | Delete
Sham	Ratnagiri	Edit | Details | Delete

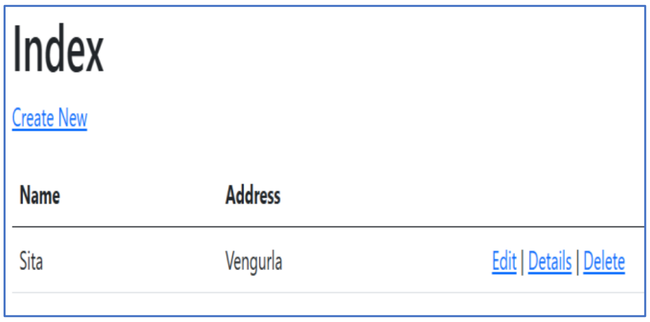

Delete

Are you sure you want to delete this?

Student

Name	Sham
Address	Ratnagiri

Delete | Back to List


Index

Create New

Name	Address	
Sita	Vengurla	Edit | Details | Delete