Create a Movie model with properties like Id, Title, Director, and Year. Create a Rating model with properties like MovieId, User, Rating, and Comment. Implement GET /movies to fetch all movies, POST /movies/{id}/ratings to submit a rating, and GET /movies/{id}/ratings to view all ratings for a specific movie. Allow users to rate movies between 1 and 10.

2. Folder Structure

Your folder structure will look like this:

```
markdown
Copy code
/MovieRatingApp
    /Controllers
    /Models
    /Services
    /Program.cs
    /Startup.cs
```

---

3. Step-by-Step Implementation

Step 1: Create Models

1. **Create the `Models` folder**:
   - Right-click the project > **Add** > **New Folder**.
   - Name it **Models**.
2. **Add `Movie.cs`**:
   - Right-click the `Models` folder > **Add** > **Class**.
   - Name it `Movie.cs`.

   ```
   csharp
   Copy code
   namespace MovieRatingApp.Models
   {
       public class Movie
       {
           public int Id { get; set; }
           public string Title { get; set; }
           public string Director { get; set; }
           public int Year { get; set; }
       }
   }
   ```

3. **Add `Rating.cs`**:
   - Right-click the `Models` folder > **Add** > **Class**.
   - Name it `Rating.cs`.

   ```
   csharp
   Copy code
   namespace MovieRatingApp.Models
   {
       public class Rating
       {
   ```

```
        public int MovieId { get; set; }
        public string User { get; set; }
        public int Score { get; set; } // Between 1 and 10
        public string Comment { get; set; }
    }
}
```

Step 2: Create Services for In-Memory Data Storage

1. **Create the `Services` folder**:
   - Right-click the project > **Add** > **New Folder**.
   - Name it **Services**.
2. **Add `MovieService.cs`**:
   - Right-click the `Services` folder > **Add** > **Class**.
   - Name it `MovieService.cs`.

MovieService.cs:

```
csharp
Copy code
using System.Collections.Generic;
using System.Linq;
using MovieRatingApp.Models;

namespace MovieRatingApp.Services
{
    public class MovieService
    {
        private readonly List<Movie> _movies = new();
        private readonly List<Rating> _ratings = new();

        public IEnumerable<Movie> GetAllMovies()
        {
            return _movies;
        }

        public Movie GetMovieById(int id)
        {
            return _movies.FirstOrDefault(m => m.Id == id);
        }

        public void AddMovie(Movie movie)
        {
            _movies.Add(movie);
        }

        public void AddRating(Rating rating)
        {
            _ratings.Add(rating);
        }

        public IEnumerable<Rating> GetRatingsByMovieId(int movieId)
        {
            return _ratings.Where(r => r.MovieId == movieId);
        }
    }
}
```

Step 3: Create Controllers

1. **Create the `Controllers` folder**:
   - Right-click the project > **Add** > **New Folder**.
   - Name it **Controllers**.
2. **Add `MoviesController.cs`**:
   - Right-click the `Controllers` folder > **Add** > **Controller**.
   - Choose **API Controller - Empty**.
   - Name it `MoviesController`.

MoviesController.cs:

```
csharp
Copy code
using Microsoft.AspNetCore.Mvc;
using MovieRatingApp.Models;
using MovieRatingApp.Services;

[ApiController]
[Route("api/[controller]")]
public class MoviesController : ControllerBase
{
    private readonly MovieService _movieService;

    public MoviesController(MovieService movieService)
    {
        _movieService = movieService;
    }

    [HttpGet]
    public IActionResult GetAllMovies()
    {
        return Ok(_movieService.GetAllMovies());
    }

    [HttpPost("{id}/ratings")]
    public IActionResult AddRating(int id, [FromBody] Rating rating)
    {
        if (rating.Score < 1 || rating.Score > 10)
        {
            return BadRequest("Rating must be between 1 and 10.");
        }

        var movie = _movieService.GetMovieById(id);
        if (movie == null)
        {
            return NotFound($"Movie with ID {id} not found.");
        }

        rating.MovieId = id;
        _movieService.AddRating(rating);
        return Ok("Rating added successfully.");
    }

    [HttpGet("{id}/ratings")]
    public IActionResult GetRatingsForMovie(int id)
```

```
        {
            var movie = _movieService.GetMovieById(id);
            if (movie == null)
            {
                return NotFound($"Movie with ID {id} not found.");
            }

            var ratings = _movieService.GetRatingsByMovieId(id);
            return Ok(ratings);
        }
    }
```

Step 4: Register Services in `Program.cs`

1. Open **`Program.cs`** and add dependency injection for the `MovieService`.

   Program.cs:

   ```
   csharp
   Copy code
   var builder = WebApplication.CreateBuilder(args);

   // Add services to the container.
   builder.Services.AddControllers();
   builder.Services.AddSingleton<MovieService>(); // Register
   MovieService as a singleton

   var app = builder.Build();

   // Configure the HTTP request pipeline.
   app.UseHttpsRedirection();
   app.UseAuthorization();
   app.MapControllers();

   app.Run();
   ```

Step 5: Test the Application

1. **Run the Application** by pressing **F5**.
2. Use **Postman** or **Swagger** to test the following endpoints:

Endpoints

1. **GET /api/movies**
   Fetch all movies.
2. **POST /api/movies/{id}/ratings**
   Submit a rating for a specific movie.

   Request Body:

   ```
   json
   ```

```
Copy code
{
    "User": "John Doe",
    "Score": 8,
    "Comment": "Great movie!"
}
```

3. **GET /api/movies/{id}/ratings**
   View all ratings for a specific movie.

Expected Responses

GET /api/movies

```
json
Copy code
[
   {
       "id": 1,
       "title": "Inception",
       "director": "Christopher Nolan",
       "year": 2010
   },
   {
       "id": 2,
       "title": "The Matrix",
       "director": "The Wachowskis",
       "year": 1999
   }
]
```

POST /api/movies/1/ratings

```
json
Copy code
"Rating added successfully."
```

GET /api/movies/1/ratings

```
json
Copy code
[
   {
       "movieId": 1,
       "user": "John Doe",
       "score": 8,
       "comment": "Great movie!"
   }
]
```

Create ASP.Net MVC Web application for School with Master Page and minimum 4 Pages.

2. Folder Structure

The basic folder structure after creating the project will look like this:

```
vbnet
Copy code
/SchoolManagement
    /Controllers
    /Models
    /Views
        /Shared
        /Home
        /Students
        /Teachers
        /Courses
    /Content
    /Scripts
    /App_Start
    /Global.asax
```

---

3. Step-by-Step Implementation

Step 1: Create the Master Page (Layout)

1. **Right-click the `Views/Shared` folder** > **Add** > **New Item**.
    - Select **MVC Layout Page (Razor)**.
    - Name it `Layout.cshtml`.
2. **Edit `Layout.cshtml`:**

    ```
    html
    Copy code
    <!DOCTYPE html>
    <html>
    <head>
        <title>@ViewData["Title"] - School Management System</title>
        <link href="~/Content/site.css" rel="stylesheet" />
    </head>
    <body>
        <header>
            <h1>School Management System</h1>
            <nav>
                <ul>
                    <li>@Html.ActionLink("Home", "Index", "Home")</li>
                    <li>@Html.ActionLink("Students", "Index",
    "Students")</li>
                    <li>@Html.ActionLink("Teachers", "Index",
    "Teachers")</li>
                    <li>@Html.ActionLink("Courses", "Index",
    "Courses")</li>
                </ul>
            </nav>
        </header>
        <main>
            @RenderBody()
        </main>
        <footer>
            <p>© 2024 School Management</p>
        </footer>
    </body>
    </html>
    ```

Step 2: Create Controllers

1. HomeController

1. **Right-click the `Controllers` folder** > **Add** > **Controller**.
   - Choose **MVC 5 Controller - Empty**.
   - Name it `HomeController`.
2. **Edit `HomeController.cs`**:

```
csharp
Copy code
using System.Web.Mvc;

namespace SchoolManagement.Controllers
{
    public class HomeController : Controller
    {
        public ActionResult Index()
        {
            return View();
        }

        public ActionResult About()
        {
            ViewBag.Message = "About the School Management System.";
            return View();
        }

        public ActionResult Contact()
        {
            ViewBag.Message = "Contact us for more information.";
            return View();
        }
    }
}
```

2. StudentsController

1. **Right-click the `Controllers` folder** > **Add** > **Controller**.
   - Choose **MVC 5 Controller - Empty**.
   - Name it `StudentsController`.
2. **Edit `StudentsController.cs`**:

```
csharp
Copy code
using System.Web.Mvc;

namespace SchoolManagement.Controllers
{
    public class StudentsController : Controller
    {
        public ActionResult Index()
        {
            return View();
        }

        public ActionResult Details(int id)
        {
            ViewBag.StudentId = id;
            return View();
        }
    }
}
```

3. TeachersController

1. **Right-click the `Controllers` folder** > **Add** > **Controller**.
   - Choose **MVC 5 Controller - Empty**.
   - Name it `TeachersController`.
2. **Edit `TeachersController.cs`**:

```
csharp
Copy code
using System.Web.Mvc;

namespace SchoolManagement.Controllers
{
    public class TeachersController : Controller
    {
        public ActionResult Index()
        {
            return View();
        }

        public ActionResult Details(int id)
        {
            ViewBag.TeacherId = id;
            return View();
        }
    }
}
```

4. CoursesController

1. **Right-click the `Controllers` folder** > **Add** > **Controller**.
   - Choose **MVC 5 Controller - Empty**.
   - Name it `CoursesController`.
2. **Edit `CoursesController.cs`**:

```
csharp
Copy code
using System.Web.Mvc;

namespace SchoolManagement.Controllers
{
    public class CoursesController : Controller
    {
        public ActionResult Index()
        {
            return View();
        }

        public ActionResult Details(int id)
        {
            ViewBag.CourseId = id;
            return View();
        }
        }
    }
```

Step 3: Create Views

1. Home Views

1. **Right-click the `Views/Home` folder > Add > View.**
   - Name it `Index.cshtml`.
2. **Edit `Index.cshtml`:**

   ```
   html
   Copy code
   @{
       ViewData["Title"] = "Home";
   }

   <h2>Welcome to the School Management System</h2>
   <p>Manage students, teachers, and courses efficiently.</p>
   ```

3. **Add `About.cshtml`:**
   - Right-click the `Views/Home` folder > **Add** > **View**.
   - Name it `About.cshtml`.

   About.cshtml:

   ```
   html
   Copy code
   @{
       ViewData["Title"] = "About";
   }

   <h2>About</h2>
   <p>@ViewBag.Message</p>
   ```

4. **Add `Contact.cshtml`:**
   - Right-click the `Views/Home` folder > **Add** > **View**.
   - Name it `Contact.cshtml`.

   Contact.cshtml:

   ```
   html
   Copy code
   @{
       ViewData["Title"] = "Contact";
   }

   <h2>Contact</h2>
   <p>@ViewBag.Message</p>
   ```

2. Students Views

1. **Right-click the `Views/Students` folder > Add > View.**
   - Name it `Index.cshtml`.

Index.cshtml:

```
html
Copy code
@{
    ViewData["Title"] = "Students";
}

<h2>Students</h2>
<p>List of all students.</p>
```

2. **Add `Details.cshtml`** for `Students`:
   - Right-click the `Views/Students` folder > **Add** > **View**.
   - Name it `Details.cshtml`.

Details.cshtml:

```
html
Copy code
@{
    ViewData["Title"] = "Student Details";
}

<h2>Student Details</h2>
<p>Details for student with ID: @ViewBag.StudentId</p>
```

3. Teachers Views

1. **Right-click the `Views/Teachers` folder** > **Add** > **View**.
   - Name it `Index.cshtml`.

Index.cshtml:

```
html
Copy code
@{
    ViewData["Title"] = "Teachers";
}

<h2>Teachers</h2>
<p>List of all teachers.</p>
```

2. **Add `Details.cshtml`** for `Teachers`:
   - Right-click the `Views/Teachers` folder > **Add** > **View**.
   - Name it `Details.cshtml`.

Details.cshtml:

```
html
Copy code
@{
    ViewData["Title"] = "Teacher Details";
}

<h2>Teacher Details</h2>
<p>Details for teacher with ID: @ViewBag.TeacherId</p>
```

4. Courses Views

1. **Right-click the `Views/Courses` folder** > **Add** > **View**.
   - Name it `Index.cshtml`.

   Index.cshtml:

   ```
   html
   Copy code
   @{
       ViewData["Title"] = "Courses";
   }

   <h2>Courses</h2>
   <p>List of all courses.</p>
   ```

2. **Add `Details.cshtml`** for `Courses`:
   - Right-click the `Views/Courses` folder > **Add** > **View**.
   - Name it `Details.cshtml`.

   Details.cshtml:

   ```
   html
   Copy code
   @{
       ViewData["Title"] = "Course Details";
   }

   <h2>Course Details</h2>
   <p>Details for course with ID: @ViewBag.CourseId</p>
   ```

Step 4: Configure Routing

1. Open **`RouteConfig.cs`** in the **App_Start** folder:

   ```
   csharp
   Copy code
   public class RouteConfig
   {
       public static void RegisterRoutes(RouteCollection routes)
       {
           routes.IgnoreRoute("{resource}.axd/{*pathInfo}");

           routes.MapRoute(
               name: "Default",
               url: "{controller}/{action}/{id}",
               defaults: new { controller = "Home", action = "Index", id
   = UrlParameter.Optional }
           );
       }
   }
   ```