Create a Book model with properties like Id, Title, Author, Price, and Category. Create a Category model with properties like Id and Name. Implement GET, POST, PUT, and DELETE for both books and categories. Add a GET /books/{categoryId} to list all books under a specific category.

Here's the folder structure we'll create:

```
markdown
Copy code
/BookManagement
    /Controllers
    /Models
    /Data
    /Program.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. **Create the `Book` model**:
   - Right-click the `Models` folder > **Add** > **Class**.
   - Name it `Book.cs`.

   **Book.cs**:

   ```
   csharp
   Copy code
   namespace BookManagement.Models
   {
       public class Book
       {
           public int Id { get; set; }
           public string Title { get; set; }
           public string Author { get; set; }
           public decimal Price { get; set; }
           public int CategoryId { get; set; }
       }
   }
   ```

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

   **Category.cs**:

   ```
   csharp
   Copy code
   namespace BookManagement.Models
   ```

```
{
    public class Category
    {
        public int Id { get; set; }
        public string Name { get; set; }
    }
}
```

---

## Step 2: Create In-Memory Data Store

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

**DataStore.cs**:

```
csharp
Copy code
using BookManagement.Models;
using System.Collections.Generic;

namespace BookManagement.Data
{
    public static class DataStore
    {
        public static List<Book> Books { get; set; } = new
List<Book>();
        public static List<Category> Categories { get; set; } = new
List<Category>();

        static DataStore()
        {
            // Seed initial data
            Categories.Add(new Category { Id = 1, Name = "Fiction"
});
            Categories.Add(new Category { Id = 2, Name = "Non-
Fiction" });

            Books.Add(new Book { Id = 1, Title = "Book 1", Author =
"Author 1", Price = 10.99M, CategoryId = 1 });
            Books.Add(new Book { Id = 2, Title = "Book 2", Author =
"Author 2", Price = 15.49M, CategoryId = 2 });
        }
    }
}
```

---

## Step 3: Create Controllers

*Books Controller*

1. **Create the `Controllers` folder**:

- Right-click the project > **Add** > **New Folder**.
- Name it **Controllers**.

2. **Create `BookController`:**
   - Right-click the `Controllers` folder > **Add** > **Controller**.
   - Select **API Controller - Empty**.
   - Name it `BookController`.

**BookController.cs:**

```
csharp
Copy code
using Microsoft.AspNetCore.Mvc;
using BookManagement.Data;
using BookManagement.Models;
using System.Linq;

namespace BookManagement.Controllers
{
    [ApiController]
    [Route("api/[controller]")]
    public class BookController : ControllerBase
    {
        [HttpGet]
        public IActionResult GetBooks()
        {
            return Ok(DataStore.Books);
        }

        [HttpGet("{id}")]
        public IActionResult GetBook(int id)
        {
            var book = DataStore.Books.FirstOrDefault(b => b.Id ==
id);
            if (book == null) return NotFound("Book not found.");
            return Ok(book);
        }

        [HttpPost]
        public IActionResult CreateBook([FromBody] Book newBook)
        {
            newBook.Id = DataStore.Books.Count + 1;
            DataStore.Books.Add(newBook);
            return CreatedAtAction(nameof(GetBook), new { id =
newBook.Id }, newBook);
        }

        [HttpPut("{id}")]
        public IActionResult UpdateBook(int id, [FromBody] Book
updatedBook)
        {
            var book = DataStore.Books.FirstOrDefault(b => b.Id ==
id);
            if (book == null) return NotFound("Book not found.");

            book.Title = updatedBook.Title;
            book.Author = updatedBook.Author;
            book.Price = updatedBook.Price;
            book.CategoryId = updatedBook.CategoryId;
```

```
                return NoContent();
            }

            [HttpDelete("{id}")]
            public IActionResult DeleteBook(int id)
            {
                var book = DataStore.Books.FirstOrDefault(b => b.Id ==
id);
                if (book == null) return NotFound("Book not found.");

                DataStore.Books.Remove(book);
                return NoContent();
            }

            // GET /api/book/category/{categoryId}
            [HttpGet("category/{categoryId}")]
            public IActionResult GetBooksByCategory(int categoryId)
            {
                var books = DataStore.Books.Where(b => b.CategoryId ==
categoryId).ToList();
                if (!books.Any()) return NotFound("No books found for the
specified category.");
                return Ok(books);
            }
        }
    }
```

---

### *Categories Controller*

1. **Create `CategoryController`**:
    - Right-click the `Controllers` folder > **Add** > **Controller**.
    - Select **API Controller - Empty**.
    - Name it `CategoryController`.

**CategoryController.cs**:

```
csharp
Copy code
using Microsoft.AspNetCore.Mvc;
using BookManagement.Data;
using BookManagement.Models;
using System.Linq;

namespace BookManagement.Controllers
{
    [ApiController]
    [Route("api/[controller]")]
    public class CategoryController : ControllerBase
    {
        [HttpGet]
        public IActionResult GetCategories()
        {
            return Ok(DataStore.Categories);
        }

        [HttpGet("{id}")]
        public IActionResult GetCategory(int id)
        {
            var category = DataStore.Categories.FirstOrDefault(c =>
c.Id == id);
            if (category == null) return NotFound("Category not
found.");
            return Ok(category);
        }

        [HttpPost]
        public IActionResult CreateCategory([FromBody] Category
newCategory)
        {
            newCategory.Id = DataStore.Categories.Count + 1;
            DataStore.Categories.Add(newCategory);
            return CreatedAtAction(nameof(GetCategory), new { id =
newCategory.Id }, newCategory);
        }

        [HttpPut("{id}")]
        public IActionResult UpdateCategory(int id, [FromBody]
Category updatedCategory)
        {
            var category = DataStore.Categories.FirstOrDefault(c =>
c.Id == id);
            if (category == null) return NotFound("Category not
found.");

            category.Name = updatedCategory.Name;
            return NoContent();
        }

        [HttpDelete("{id}")]
        public IActionResult DeleteCategory(int id)
        {
            var category = DataStore.Categories.FirstOrDefault(c =>
c.Id == id);
            if (category == null) return NotFound("Category not
found.");

            DataStore.Categories.Remove(category);
            return NoContent();
        }
    }
}
```

---

## Step 4: Configure Program.cs

Make sure the `Program.cs` file is properly configured:

csharp
Copy code

```
var builder = WebApplication.CreateBuilder(args);

builder.Services.AddControllers();

var app = builder.Build();

app.UseHttpsRedirection();
app.MapControllers();
app.Run();
```

**Create ASP.Net MVC Web application for Online Mobile Shop with Master Page and minimum 4 Pages.**

## 2. Folder and File Structure

Here's the folder structure we'll use:

```
css
Copy code
/OnlineMobileShop
    /Controllers
    /Models
    /Views
        /Shared
        /Home
        /Products
        /Contact
        /About
    /Content
    /Scripts
```

---

## 3. Create the Master Page (Layout)

### Step 1: Create Layout Page

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

```
html
Copy code
<!DOCTYPE html>
<html>
<head>
    <title>@ViewData["Title"] - Online Mobile Shop</title>
    <link href="~/Content/site.css" rel="stylesheet" />
</head>
<body>
    <header>
        <h1>Online Mobile Shop</h1>
        <nav>
            <ul>
                <li>@Html.ActionLink("Home", "Index", "Home")</li>
                <li>@Html.ActionLink("Products", "Index",
"Products")</li>
                <li>@Html.ActionLink("About", "About", "Home")</li>
                <li>@Html.ActionLink("Contact", "Contact",
"Home")</li>
            </ul>
        </nav>
    </header>
    <main>
        @RenderBody()
    </main>
    <footer>
        <p>© 2024 Online Mobile Shop</p>
    </footer>
</body>
</html>
```

---

## 4. Create Controllers

### Step 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 OnlineMobileShop.Controllers
{
    public class HomeController : Controller
    {
        public ActionResult Index()
        {
            return View();
        }

        public ActionResult About()
        {
            ViewBag.Message = "About the Online Mobile Shop.";
            return View();
        }

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

---

### Step 2: ProductsController

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

```
csharp
Copy code
```

```
using System.Collections.Generic;
using System.Web.Mvc;
using OnlineMobileShop.Models;

namespace OnlineMobileShop.Controllers
{
    public class ProductsController : Controller
    {
        public ActionResult Index()
        {
            var products = new List<Product>
            {
                new Product { Id = 1, Name = "iPhone 14", Price =
999, Category = "Smartphones" },
                new Product { Id = 2, Name = "Samsung Galaxy S23",
Price = 899, Category = "Smartphones" },
                new Product { Id = 3, Name = "OnePlus 11", Price =
799, Category = "Smartphones" }
            };

            return View(products);
        }
    }
}
```

---

## 5. Create Models

### Step 1: Create Product Model

1. **Right-click the `Models` folder** > **Add** > **Class**.
   - Name it `Product.cs`.
2. **Define the `Product` model**:

```
csharp
Copy code
namespace OnlineMobileShop.Models
{
    public class Product
    {
        public int Id { get; set; }
        public string Name { get; set; }
        public decimal Price { get; set; }
        public string Category { get; set; }
    }
}
```

---

## 6. Create Views

### Step 1: Home Views

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

     ```
     html
     ```

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

<h2>Welcome to Online Mobile Shop</h2>
<p>Explore our latest mobile products!</p>
```

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

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

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

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

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

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

---

## Step 2: Products View

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

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

<h2>Our Products</h2>
<table class="table">
    <thead>
        <tr>
            <th>Name</th>
            <th>Price</th>
            <th>Category</th>
        </tr>
    </thead>
    <tbody>
```

```
        @foreach (var product in Model)
        {
            <tr>
                <td>@product.Name</td>
                <td>@product.Price</td>
                <td>@product.Category</td>
            </tr>
        }
    </tbody>
</table>
```

---

## 7. 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 }
        );
    }
}
```