MVC Design Pattern in C#

What is MVC?

MVC (Model-View-Controller) is a **software design pattern** used for developing web applications by **separating concerns** into three main components:

1. **Model (M)**: Represents the application's data and business logic.
2. **View (V)**: Responsible for displaying UI elements to the user.
3. **Controller (C)**: Handles user input, interacts with the model, and determines the appropriate view.

Advantages of MVC

✔ **Separation of Concerns**: UI, business logic, and data access are separate, making the code modular and maintainable.

✔ **Easier to Test**: Each component can be tested independently.

✔ **Supports Multiple Views**: The same business logic (Model) can be presented in different views (UI).

✔ **Better Control Over HTML**: Allows developers to have full control over the generated HTML.

✔ **Improved Code Reusability**: The separation enables reuse of models and views in different parts of the application.

How to Implement MVC in C#?

Step 1: Create an ASP.NET Core MVC Project

1 Open **Visual Studio → Create a New Project** → Select **ASP.NET Core Web App (Model-View-Controller)**.

2 Choose **.NET Core** and click **Create**.

Step 2: Create Model (Business Logic & Data Handling)

 Example: Create a `Student` Model

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

Step 3: Create Controller (Handles User Requests & Logic)

 Example: Create a `StudentController` to Handle Student Data

```
using Microsoft.AspNetCore.Mvc;
using System.Collections.Generic;

public class StudentController : Controller {
    public IActionResult Index() {
        List<Student> students = new List<Student> {
            new Student { Id = 1, Name = "John", Age = 22 },
            new Student { Id = 2, Name = "Alice", Age = 21 }
        };

        return View(students);
    }
}
```

Step 4: Create View (Displays Data to the User)

 Example: Create a View (`Index.cshtml`) to Show Students

```
@model List<Student>

<h2>Student List</h2>

<table border="1">
    <tr>
        <th>ID</th>
        <th>Name</th>
        <th>Age</th>
    </tr>
    @foreach (var student in Model) {
        <tr>
            <td>@student.Id</td>
            <td>@student.Name</td>
            <td>@student.Age</td>
        </tr>
    }
</table>
```

Passing Data from Controller to View

There are **3 ways** to pass data from Controller to View:

1 Strongly-Typed Data (ViewModel) – Recommended Approach

- Define a **model** and pass it to the view.
- Example:

```
public IActionResult Contact() {
    var student = new Student { Name = "Binod", Age = 22 };
    return View(student);
}
```

- In the View (`Contact.cshtml`):

```
@model Student
<h2>Student Name: @Model.Name</h2>
<h2>Age: @Model.Age</h2>
```

---

2 Weakly-Typed Data (ViewData & ViewBag)

- **ViewData**: Uses **key-value pairs** and requires typecasting.
- **ViewBag**: Uses **dynamic properties** (easier to use).

✔ **Using ViewData in Controller**:

```
public IActionResult SomeAction() {
    ViewData["Greeting"] = "Hello";
    ViewData["Student"] = new Student { Name = "Binod", Age = 22 };
    return View();
}
```

✔ **Using ViewData in View** (`SomeAction.cshtml`):

```
<h2>@ViewData["Greeting"] World!</h2>
@{
    var student = ViewData["Student"] as Student;
}
<p>Name: @student.Name</p>
<p>Age: @student.Age</p>
```

✔ **Using ViewBag in Controller**:

```
public IActionResult About() {
    ViewBag.Message = "Welcome to MVC!";
    return View();
}
```

✔ **Using ViewBag in View** (`About.cshtml`):

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

3□ Using Partial Views

- **Reusing UI Components** (like headers, menus, or footers).
- Example:

```
@Html.Partial("_StudentPartial")
```

Conclusion

□ **MVC improves application structure** by separating logic into Model, View, and Controller.

□ It makes the application **modular, testable, and maintainable**.

□ **ASP.NET Core MVC** simplifies the process of implementing MVC in web applications.

Would you like a **diagram** to illustrate this process? 😊