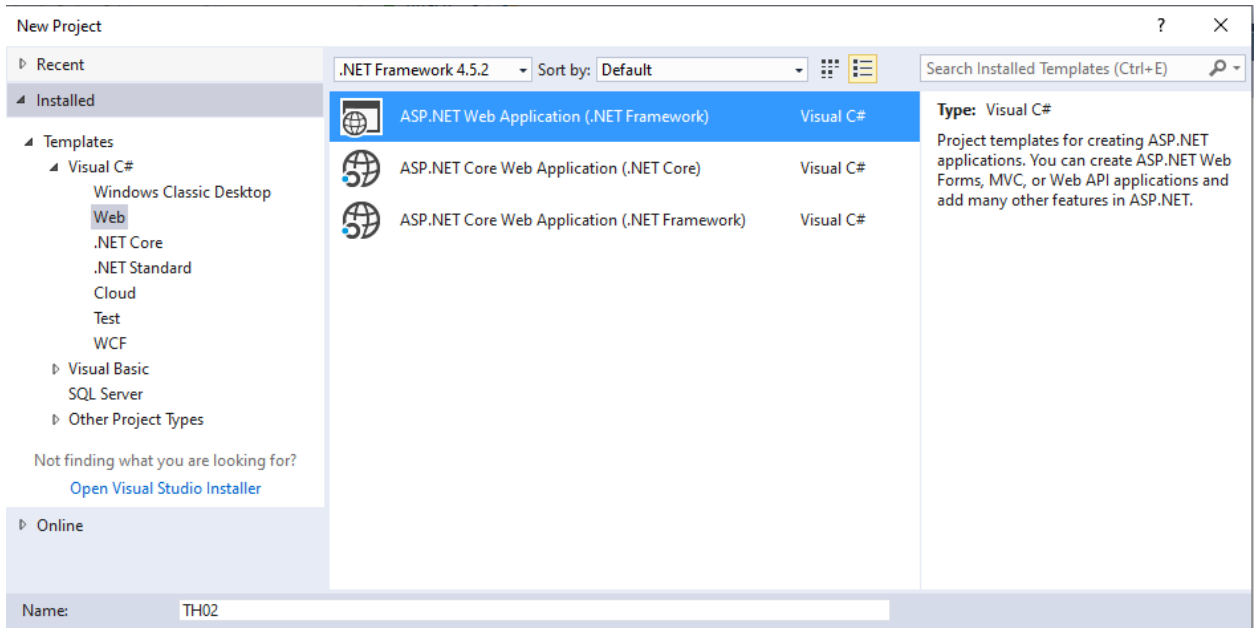


THỰC HÀNH LẬP TRÌNH WEB NÂNG CAO

BUỔI 02

I. Truyền dữ liệu từ Controller vào View

Đầu tiên ta tạo Project MVC, đặt tên là “TH02”



Tiếp theo vào thư mục Models, tạo model sau:

```
0 references
public class Employee
{
    0 references | 0 exceptions
    public string FirstName { get; set; }
    0 references | 0 exceptions
    public string LastName { get; set; }
    0 references | 0 exceptions
    public int Salary { get; set; }
}
```

Tạo EmployeeController để kiểm tra truyền data với ViewData

0 references

```
public class EmployeeController : Controller
{
    // GET: Employee
    0 references | 0 requests | 0 exceptions
    public ActionResult GetView()
    {
        Employee emp = new Employee();
        emp.FirstName = "Sukesh";
        emp.LastName = "Marla";
        emp.Salary = 20000;
        ViewData["Employee"] = emp;
        return View();
    }
}
```

Nhớ lưu ý dùng model Employee thì phải using:

```
using TH02.Models;
```

Ứng với ActionMethod trên, tạo View tương ứng như sau:

```

@using TH02.Models
@{
    ViewBag.Title = "GetView";
    Layout = "~/Views/Shared/_Layout.cshtml";
}

<div>
    @{
        Employee emp = (Employee) ViewData["Employee"];
    }
    <b>Employee Details </b><br />
    Employee Name : @emp.FirstName @emp.LastName <br />
    Employee Salary: @emp.Salary.ToString("C")|
</div>

```

Kết quả:

Employee Details

Employee Name : Sukesh Marla

Employee Salary: \$20,000.00

© 2020 - My ASP.NET Application

Bài tập 1: Thay vì dùng ViewData, hãy thử dùng ViewBag và xem kết quả
 Hãy nhận xét khuyết điểm của việc sử dụng ViewData và ViewBag (lưu thành file Word)

II. Sử dụng Strongly-Typed View

Ta có thể truyền trực tiếp model vào trong View

```
0 references
public class EmployeeController : Controller
{
    // GET: Employee
    0 references | 0 requests | 0 exceptions
    public ActionResult GetView()
    {
        Employee emp = new Employee();
        emp.FirstName = "Sukesh";
        emp.LastName = "Marla";
        emp.Salary = 20000;
        return View(emp);
    }
}
```

Sửa lại thông tin View hiển thị:

```
@model TH02.Models.Employee
@{
    ViewBag.Title = "GetView";
    Layout = "~/Views/Shared/_Layout.cshtml";
}

<hr/>
<div>
    <b>Employee Details </b><br />
    Employee Name : @Model.FirstName @Model.LastName <br />
    Employee Salary: @Model.Salary.ToString("C")
</div>
```

Một số điểm lưu ý

- Phải khai báo @model đầu file để khai báo Strongly-Typed View
- Dùng @Model để lấy giá trị các fields của model truyền vào

III. Một số thao tác với Strongly-Typed View:

Tạo một ActionMethod khác trong EmployeeController

```
0 references | 0 requests | 0 exceptions
public ActionResult Display()
{
    Employee emp = new Employee();
    emp.FirstName = "John";
    emp.LastName = "Cena";
    emp.Salary = 100000;
    return View(emp);
}
```

Tạo View tương ứng với action trên với nội dung như sau:

```
@{
    ViewBag.Title = "Display";
    Layout = "~/Views/Shared/_Layout.cshtml";
}
<hr/>
<b>Employee Details</b><br />
Employee Name : @Model.FirstName @Model.LastName <br />
@if (Model.Salary > 15000)
{
    <span style="background-color:yellow">
        Employee Salary: @Model.Salary.ToString("C")
    </span>
}
else
{
    <span style="background-color:green">
        Employee Salary: @Model.Salary.ToString("C")
    </span>
}
```

Chạy và xem kết quả:

Employee Details

Employee Name : John Cena

Employee Salary: \$100,000.00

Bài tập 2: Việc xử lý if-else trong View ở ví dụ trên là nên hạn chế. Hãy suy nghĩ để đoạn code xử lý màu sắc nằm hoàn toàn trong Controller, bên View chỉ hiển thị màu mà thôi

VD:

```
@{
    ViewBag.Title = "Display";
    Layout = "~/Views/Shared/_Layout.cshtml";
}
<hr/>
Hello @Model.UserName
<hr />
<div>
    <b>Employee Details</b><br />
    Employee Name : @Model.EmployeeName <br />
    <span style="background-color:@Model.SalaryColor">
        Employee Salary: @Model.Salary
    </span>
</div>
```

IV. View với Collection

Bổ sung thêm ActionMethod GetEmployees như hình:

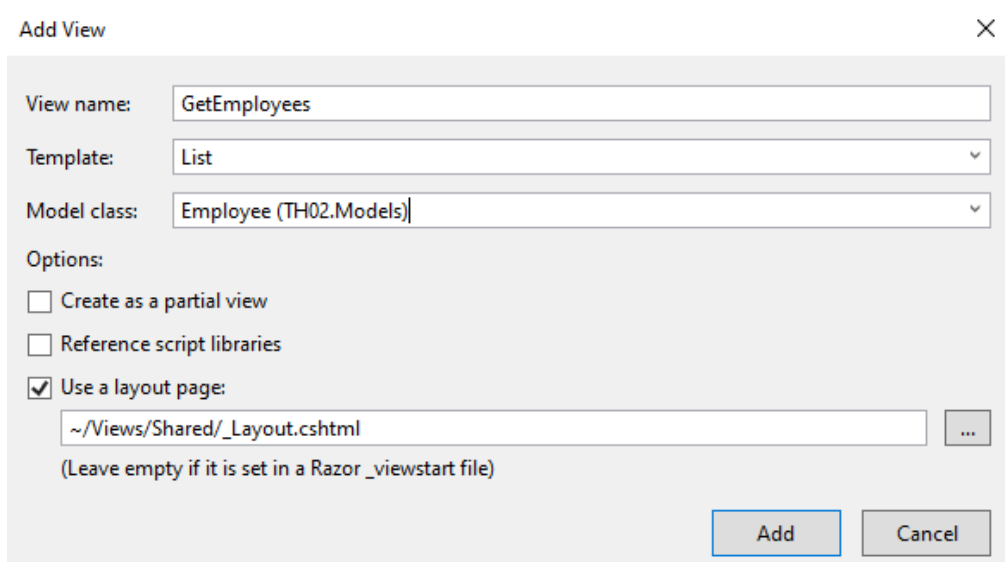
```
0 references | 0 requests | 0 exceptions
public ActionResult GetEmployees()
{
    List<Employee> employees = new List<Employee>();
    Employee emp = new Employee();
    emp.FirstName = "johnson";
    emp.LastName = "fernandes";
    emp.Salary = 14000;
    employees.Add(emp);

    emp = new Employee();
    emp.FirstName = "michael";
    emp.LastName = "jackson";
    emp.Salary = 16000;
    employees.Add(emp);

    emp = new Employee();
    emp.FirstName = "robert";
    emp.LastName = "pattinson";
    emp.Salary = 20000;
    employees.Add(emp);

    return View(employees);
}
```

View ta tạo sẽ nhận danh sách (collection) các nhân viên, ở đây lưu ý chọn Template là List và Model class là “Employee”



The screenshot shows the 'Add View' dialog box with the following configuration:

- View name: GetEmployees
- Template: List
- Model class: Employee (TH02.Models)
- Options:
 - ☐ Create as a partial view
 - ☐ Reference script libraries
 - ☒ Use a layout page: ~/Views/Shared/_Layout.cshtml

Buttons: Add, Cancel

Kết quả:

GetEmployees

[Create New](#)

FirstName	LastName	Salary	
johnson	fernandes	14000	Edit Details Delete
michael	jackson	16000	Edit Details Delete
robert	pattinson	20000	Edit Details Delete

Bài tập 3:

Dựa vào kiến thức đã học, hãy

- Tạo ứng dụng quản lý thông tin sản phẩm Product (gồm ID, ProductName, ProductType, ProductPrice)
- Tạo danh sách gồm 5 sản phẩm khác nhau (do sinh viên tự điền thông tin)
- Tạo View hiển thị thông tin các sản phẩm và dùng Bootstrap 3 để chỉnh sửa các hiển thị cho đẹp (bonus điểm)

Bài tập về nhà

- Lưu trữ hình của sản phẩm (tạo thư mục Images) và tìm cách để hiển thị hình của sản phẩm trong bản thông tin tất cả sản phẩm (chỉnh hình kích thước nhỏ lại)

HẾT