1. What are the responsibilities of each layer of the MVC architecture and how are they connected?

Model- klasy, które reprezentują dane (struktury i powiązania) aplikacji. Zazwyczaj instancje modelu pobierają i przechowują dane w bazie danych.
View - komponenty odpowiedzialne za wyświelanie interfejsu użytkownika
Controller - klasy, które obsługują zapytania, pobierają dane z modelu i wywołują "view templates" zwracające odpowiedź

2. What are the naming conventions for models, controllers, controller actions, views folders and views themselves?

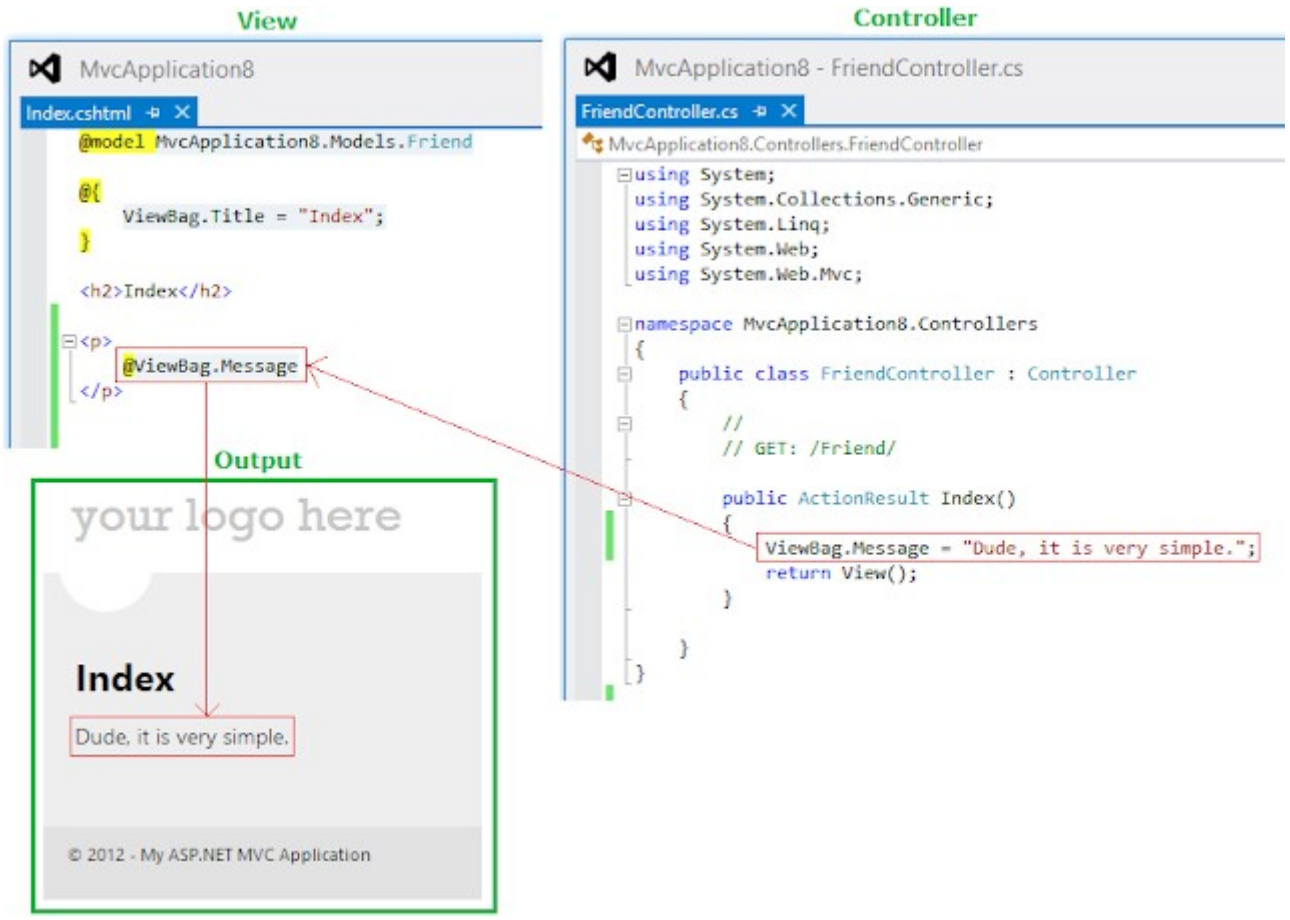
-nazwa kontrolera musi kończyć się na "controller"
-nazwa modelu powinna być w liczbie pojedynczej i powinna odpowiadać nazwie tabeli w bazie danych, z którą model jest powiązany
-nazwa folderu zawierającego widoki powinna być w liczbie mnogiej

dodatkowo:

//źródło: jeremybytes.blogspot.com//
The default route pattern is [Controller Name] / [Action Name] / [ID (optional parameter)].
Controllers are located in the "Controllers" folder.
Controllers are named with the suffix "Controller".
An Action is a function with the same name in the Controller class.
Views are in the "Views" folder with sub-folders based on the Controller name.
Views are named after the corresponding Action in the Controller.

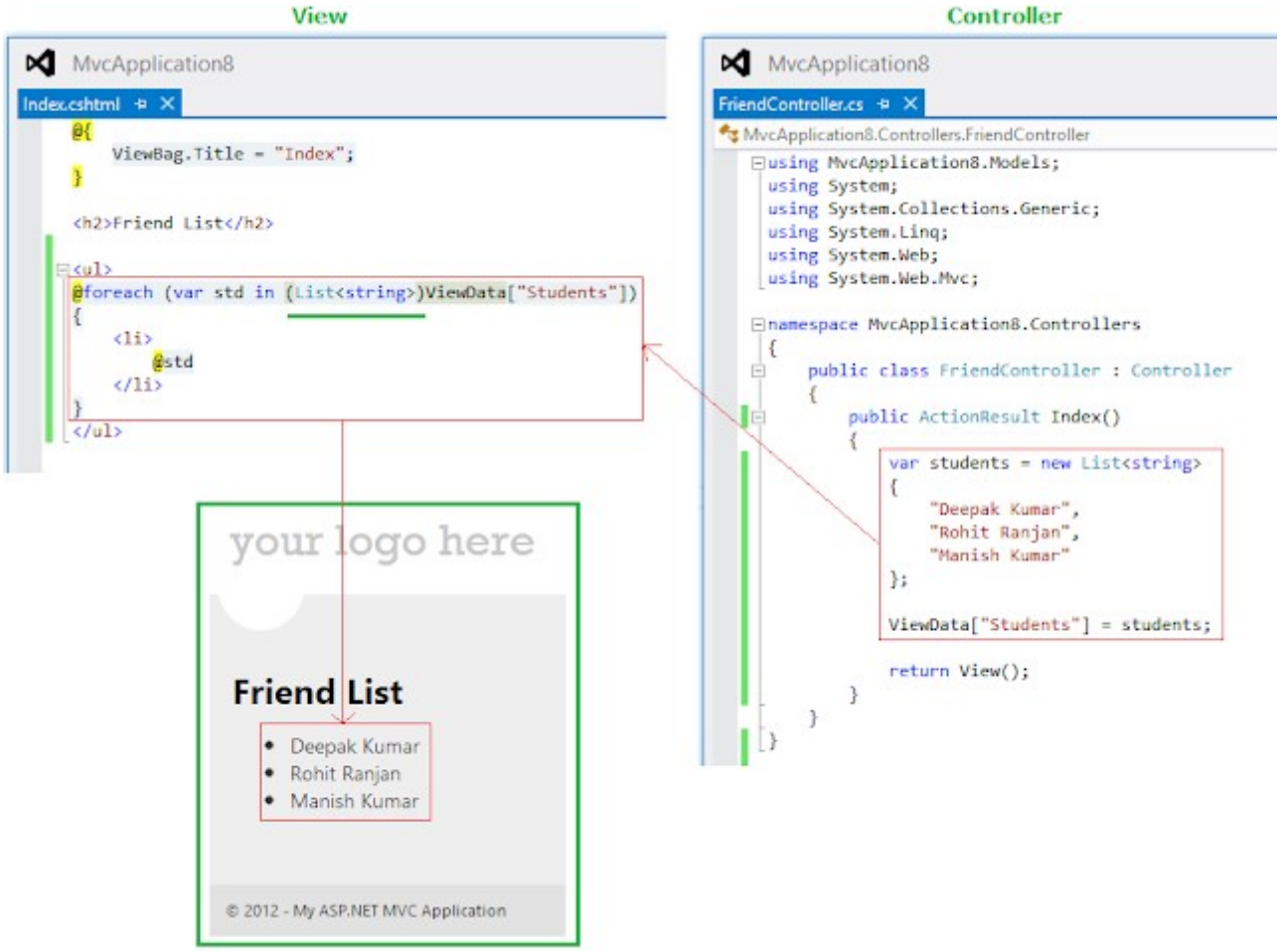
3. How to pass data from controllers to views (2 options)?
ViewBag, ViewData, TempData

schematy poniżej, źródło: www.c-sharpcorner.com


View
Controller
MvcApplication8
Index.cshtml
@model MvcApplication8.Models.Friend
@{
ViewBag.Title = "Index";
}
<h2>Index</h2>

@ViewBag.Message

MvcApplication8 - FriendController.cs
FriendController.cs
MvcApplication8.Controllers.FriendController
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
namespace MvcApplication8.Controllers
{
public class FriendController : Controller
{
//
// GET: /Friend/
public ActionResult Index()
{
ViewBag.Message = "Dude, it is very simple.";
return View();
}
}
}
Output
your logo here
Index
Dude, it is very simple.
© 2012 - My ASP.NET MVC Application


View
Controller
MvcApplication8
Index.cshtml
@{
ViewBag.Title = "Index";
}
<h2>Friend List</h2>

@foreach (var std in (List<string>)ViewData["Students"])
{

@std

}

MvcApplication8
FriendController.cs
MvcApplication8.Controllers.FriendController
using MvcApplication8.Models;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Mvc;
namespace MvcApplication8.Controllers
{
public class FriendController : Controller
{
public ActionResult Index()
{
var students = new List<string>
{
"Deepak Kumar",
"Rohit Ranjan",
"Manish Kumar"
};
ViewData["Students"] = students;
return View();
}
}
}
your logo here
Friend List
Deepak Kumar
Rohit Ranjan
Manish Kumar
© 2012 - My ASP.NET MVC Application

4. How to map URL’s to controller actions?

W kontrolerze definiujemy publiczne metody, które są wywoływane w wyniku zapytań HTTP (HTTP endpoints). Domyśly format foutingu zapytań w MVC to `/[Controller]/[ActionName]/[Parameters]`, w przypadku tego projektu mogłem dostosować je w metodzie Configure w pliku Startup.cs

5. How to restrict controller actions to be executed only via certain HTTP request types (e.g., only via POST)?

Poprzez dodanie odpowiedniej adnotacji, np. HttpPost. Przykładowo:

```
// GET: Movies/Create
public IActionResult Create()
{
    return View();
}

// POST: Movies/Create
[HttpPost]
[ValidateAntiForgeryToken]
public async Task<IActionResult> Create(
    [Bind("ID,Title,ReleaseDate,Genre,Price, Rating")] Movie movie)
{
    if (ModelState.IsValid)
    {
        _context.Add(movie);
        await _context.SaveChangesAsync();
        return RedirectToAction("Index");
    }
    return View(movie);
}
```

6. How to make sure a controller action can only be called through a form on our website and not through some external request?

Wywołać funkcję AntiForgeryToken w widoku oraz adnotację ValidateAntiForgeryToken w akcji kontrolera.

7. Where do you define data validation and how do you ensure it in views and controllers?

Walidację danych zdefiniowałem jedynie w modelu (Movie.cs). Kontrolery i widoki automatycznie przejęły zasady walidacji danych z modelu.