Creating the Required Models:

First, create a class file with the name **Student.cs** within the Models folder of your application. This is the model that is going to represent the basic information of a student such as a **Name, DateOfBirth, Gender**, etc. Once you create the **Student.cs** class file, then copy and paste the following code in it.

```
namespace CodeBitsPress.Models
{
public class Student
{
public int StudentId { get; set; }
public string Name { get; set; }
public string DateOfBirth { get; set; }
public string Gender { get; set; }
}
}
```

Next, we need to create the Address model which is going to represent the Student Address such as **City, State, Country,** etc. So, create a class file with the name Address.cs within the Models folder and then copy and paste the following code in it.

```
namespace CodeBitsPress.Models
{
public class Address
{
public int StudentId { get; set; }
public string City { get; set; }
public string State { get; set; }
public string Country { get; set; }
}
}
```

Creating the View Model:

Now we need to create the View Model which will store the required data that is required for a particular view. In our case its student's Details view. This View Model is going to represent the Student Model + Student Address Model + Some additional data like page title and page header.

You can create the View Models anywhere in your application, but it is recommended to create all the View Models within a **folder** called **ViewModels** to keep the things organized.

So first create a folder at the root directory of your application with the name **ViewModels** and then create a class file with the name **StudentDetailsViewModel.cs** within the

ViewModels folder. Once you create the **StudentDetailsViewModel.cs** class file, then copy and paste the following code in it.

```
using CodeBitsPress.Models;
namespace CodeBitsPress.ViewModels
{
public class StudentDetailsViewModel
{
public Student Student { get; set; }
public Address Address { get; set; }
public string Title { get; set; }
public string Header { get; set; }
}
}
```

We named the **ViewModel** class as **StudentDetailsViewModel**. Here the word **Student** represents the **Controller** name, the word **Details** represent the **action method name** within the Student Controller. As it is a view model so we prefixed the word **ViewModel**. Although it is not mandatory to follow this naming convention, I personally prefer it to follow this naming convention to organize view models.

Creating Student Controller:

Right-click on the Controllers folder and then add a new class file with the name StudentController.cs and then copy and paste the following code in it.

```
using CodeBitsPress.Models;
using CodeBitsPress.ViewModels;
using Microsoft.AspNetCore.Mvc;

namespace CodeBitsPress.Controllers
{
public class StudentController : Controller
{
public ViewResult Details()
{
ViewBag.Title = "Student Details Page";
ViewBag.Header = "Student Details";
//Student Basic Details
Student student = new Student()
{
StudentId = 100,
Name = "Abayomi",
DateOfBirth = "20-10-1986",
Gender = "Male"
};
//Student Address
```

```
Address address = new Address()
{
StudentId = 100,
City = "Ota",
State = "Ogun",
Country = "Nigeria",
};

//Creating the View model
====================
StudentDetailsViewModel studentDetailsViewModel = new StudentDetailsViewModel()
{
Student = student,
Address = address,
Title = "Student Details Page",
Header = "Student Details",
};
//Pass the studentDetailsViewModel to the view
return View(studentDetailsViewModel);
}
}
}
```

As you can see, now we are passing the view model as a parameter to the view. This is the view model that contains all the data required by the Details view. As you can notice, now we are not using any ViewData or ViewBag to pass the Page Title and Header to the view instead they are also part of the ViewModel which makes it a strongly typed view.

Creating the Details View:

First, add a folder with the name **Student** within the **Views** folder your project. Once you add the Student Folder, then you need to add a razor view file with the name **Details.cshtml** within the Student folder. Once you add the **Details.cshtml** view then copy and paste the following code in it.

```
@model CodeBitsPress.ViewModels.StudentDetailsViewModel
<head>
<title>@Model.Title</title>
</head>
<body>
<h1>@Model.Header</h1>
<div>
StudentId : @Model.Student.StudentId
</div>
<div>
Name : @Model.Student.Name
</div>
```

```
<div>
Branch : @Model.Student.Branch
</div>
<div>
Section : @Model.Student.Section
</div>
<div>
Gender : @Model.Student.Gender
</div>
<h1>Student Address</h1>
<div>
City : @Model.Address.City
</div>
<div>
State : @Model.Address.State
</div>
<div>
Country : @Model.Address.Country
</div>
<div>
Pin : @Model.Address.Pin
</div>
</body>
</html>
```

Now, the Details view has access to the **StudentDetailsViewModel** object that we passed from the controller action method using the View() extension method. By using the **@model** directive, we set **StudentDetailsViewModel** as the Model for the **Details** view. Then we access **Student, Address, Title,** and **Header** using **@Model** property.

Now run the application, and navigate to the "**/Student/Details**" URL and you will see the output as expected on the webpage as shown in the below image.