**Integrate Controller, View and Model**

We have already created a [Controller](https://www.tutorialsteacher.com/mvc/mvc-controller), a [model](https://www.tutorialsteacher.com/mvc/mvc-model) and a [view](https://www.tutorialsteacher.com/mvc/mvc-view) in the previous sections. Here, we will integrate them to run the application and see the result.

The following code snippet shows the StudentController, the Student model, and the Index.cshtml view created in the previous sections.

Example: StudentController

public class StudentController : Controller

{

// GET: Student

public ActionResult Index()

{

return View();

}

}

Example: Student Model class

public class Student

{

public int StudentId { get; set; }

public string StudentName { get; set; }

public int Age { get; set; }

}

Example: Index.cshtml View

@model IEnumerable<MVC\_BasicTutorials.Models.Student>

@{

ViewBag.Title = "Index";

Layout = "~/Views/Shared/\_Layout.cshtml";

}

<h2>Index</h2>

<p>

@Html.ActionLink("Create New", "Create")

</p>

<table class="table">

<tr>

<th>

@Html.DisplayNameFor(model => model.StudentName)

</th>

<th>

@Html.DisplayNameFor(model => model.Age)

</th>

<th></th>

</tr>

@foreach (var item in Model) {

<tr>

<td>

@Html.DisplayFor(modelItem => item.StudentName)

</td>

<td>

@Html.DisplayFor(modelItem => item.Age)

</td>

<td>

@Html.ActionLink("Edit", "Edit", new { id=item.StudentId }) |

@Html.ActionLink("Details", "Details", new { id=item.StudentId }) |

@Html.ActionLink("Delete", "Delete", new { id = item.StudentId })

</td>

</tr>

}

</table>

Now, to run it successfully, we need to pass a model object from an action method to a view. As you can see in the above Index.cshtml, it uses IEnumerable<Student> as a model type. So we need to pass it from the Index() action method of the StudentController class, as shown below.

Example: Passing Model from Controller

public class StudentController : Controller

{

static IList<Student> studentList = new List<Student>{

new Student() { StudentId = 1, StudentName = "John", Age = 18 } ,

new Student() { StudentId = 2, StudentName = "Steve", Age = 21 } ,

new Student() { StudentId = 3, StudentName = "Bill", Age = 25 } ,

new Student() { StudentId = 4, StudentName = "Ram" , Age = 20 } ,

new Student() { StudentId = 5, StudentName = "Ron" , Age = 31 } ,

new Student() { StudentId = 4, StudentName = "Chris" , Age = 17 } ,

new Student() { StudentId = 4, StudentName = "Rob" , Age = 19 }

};

// GET: Student

public ActionResult Index()

{

//fetch students from the DB using Entity Framework here

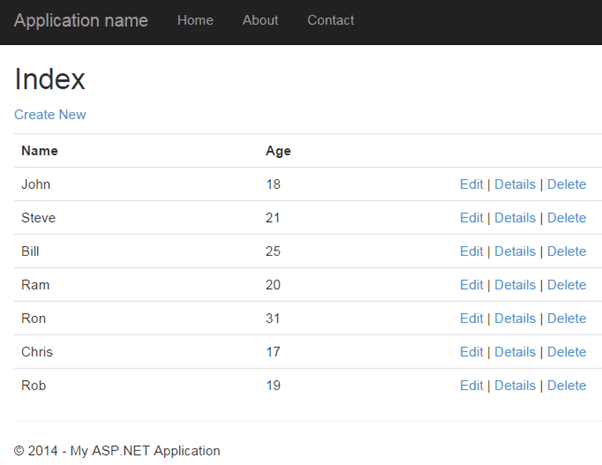
return View(studentList);

}

}

As you can see in the above code, we have created a list of student objects for an example purpose (in real-life application, you can fetch it from the database). We then pass this list object as a parameter in the View() method. The View() method is defined in the base Controller class, which automatically binds a model object to a view.

Now, you can run the MVC project by pressing F5 and navigate to http://localhost/Student. You will see the following view in the browser.

[](https://www.tutorialsteacher.com/Content/images/mvc/index-view.png)