Session 17

Advanced Features of ASP.NET Core 2.1

Session Overview

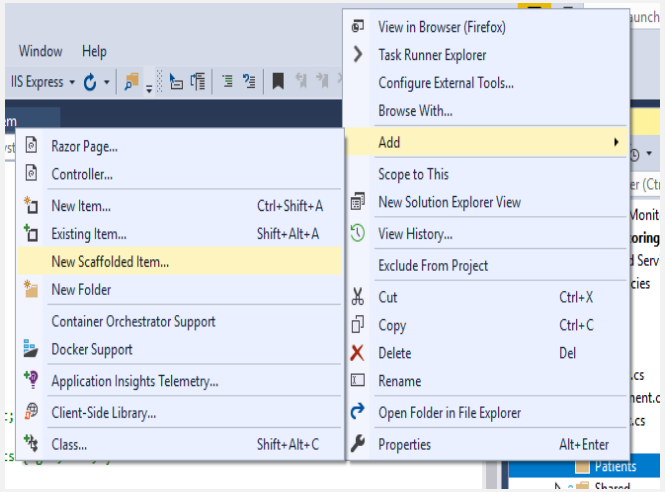
- Describe Data Access with ASP.NET Core 2.1 and EF Core
- Explain creation of Razor Pages application using Entity Framework (EF) Core
- Describe deploying ASP.NET Core 2.1 applications

Data Access with ASP.NET Core 2.1 and EF Core

- Razor Pages in ASP.NET Core 2.1 have several benefits including better organized structure of the project files.
- EF Core is a lightweight, extensible, and cross-platform version of Entity Framework.
- ASP.NET Core Razor Pages together with EF Core can be used to build Web apps for Create, Read, Update, and Delete (CRUD) operations, sorting, filtering, and so on.

Creating Razor Pages Application Using EF Core (1-5)

1. Click **File → New**. In the New Project dialog box, click .NET Core in the left pane and ASP.NET Core Web Application in the right pane.
2. Specify the name **PatientMonitoringSystem** and click **OK**.
3. In the subsequent dialog box, select **ASP.NET Core 2.1** in the top drop-down and **Web Application** as the template.
4. Click **OK**. The project will be created.
5. Next, create a new folder named *Models* and within it, a class file named Patient.*cs* having code as given in Code Snippet 1.
6. Similarly, create other entities such as Doctor and PatientEnrollment as shown in Code Snippets 2 and 3 respectively.
7. Edit *_Layout.cshtml* to delete default content. Add the line that is bold in Code Snippet 4.
8. In Solution Explorer, right-click *Pages/Patients* folder and select **Add → New Scaffolded Item**.

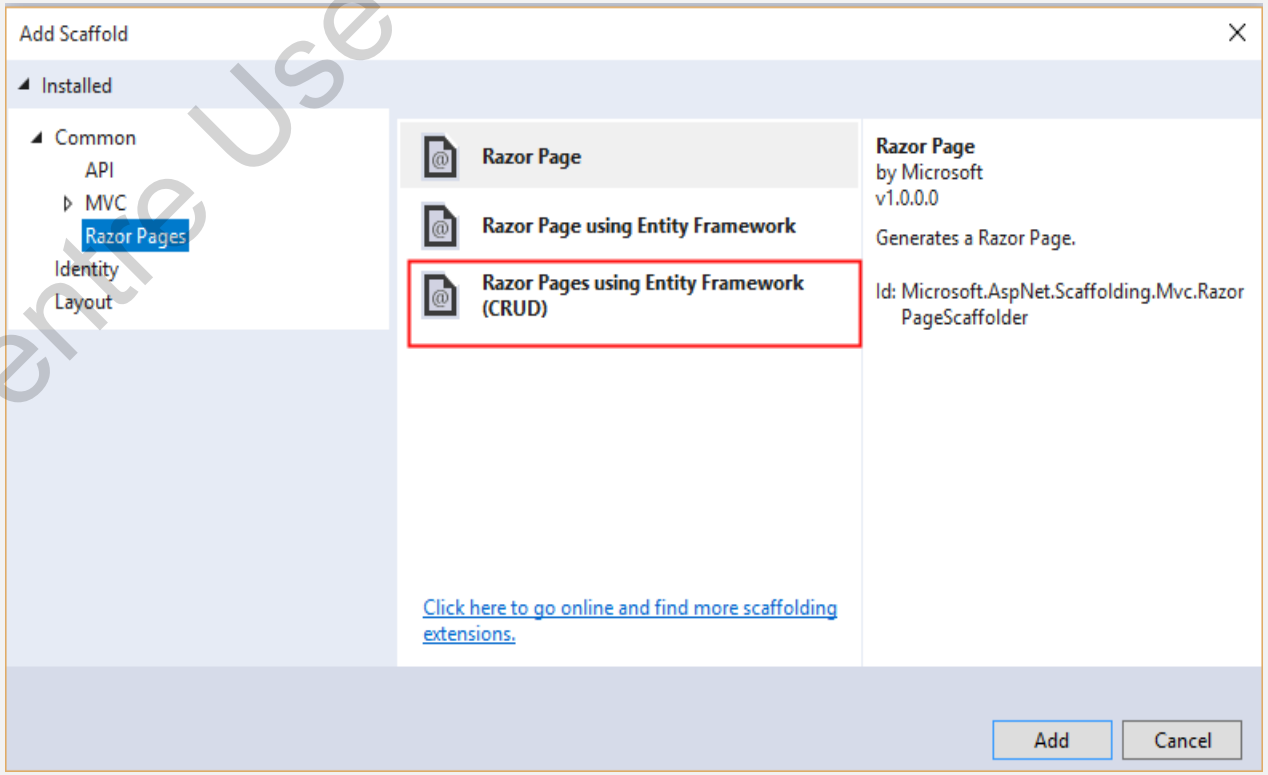


Adding Scaffolded Item

Creating Razor Pages Application Using EF Core (2-5)

9. In the **Add Scaffold** dialog box, select **Razor Pages using Entity Framework (CRUD)** → **Add**.

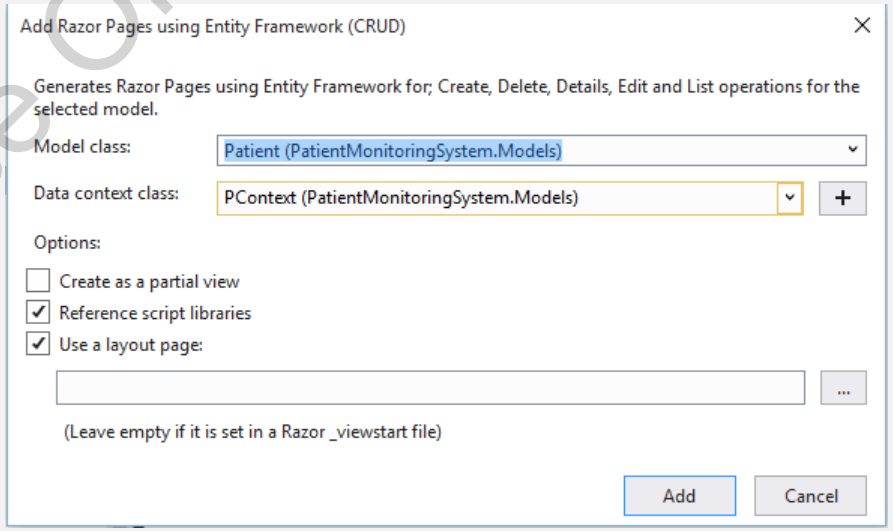
10. In the **Add Razor Pages using Entity Framework (CRUD)** dialog box, under **Model class** drop-down, select **Patient (PatientMonitoringSystem.Models)**.



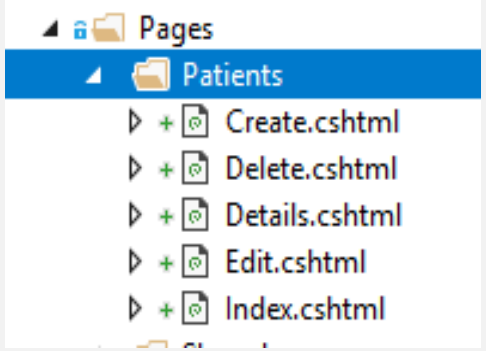
Adding Scaffolded Item for CRUD

Creating Razor Pages Application Using EF Core (3-5)

11. In the **Data context class** row, select the **+** (plus) sign and change the generated name to **PatientMonitoringSystem.Models.PMContext**.



12. Click **Add**. Views will be generated for Create, Delete, Index, Details, and Edit actions.



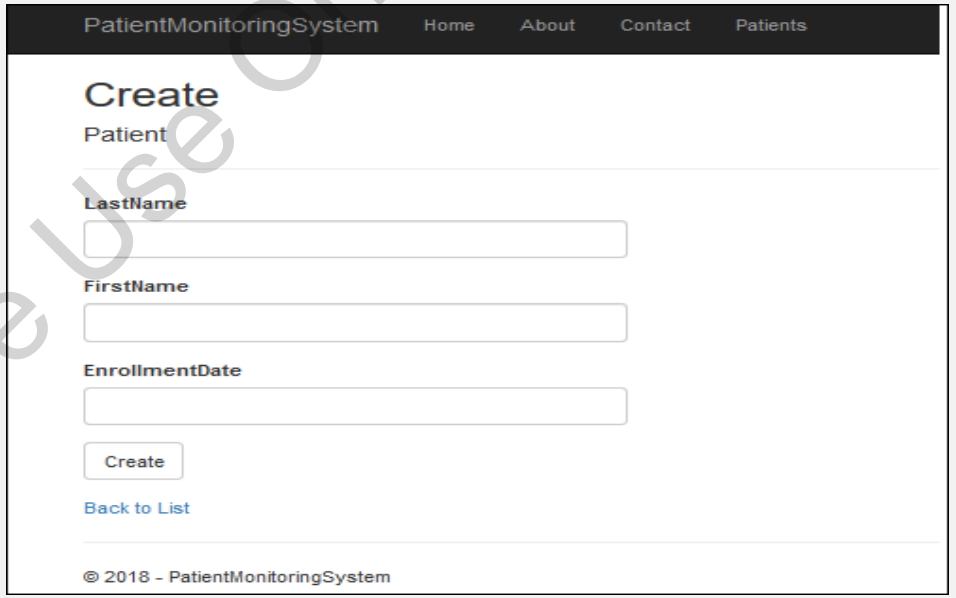
13. Build and execute the application.

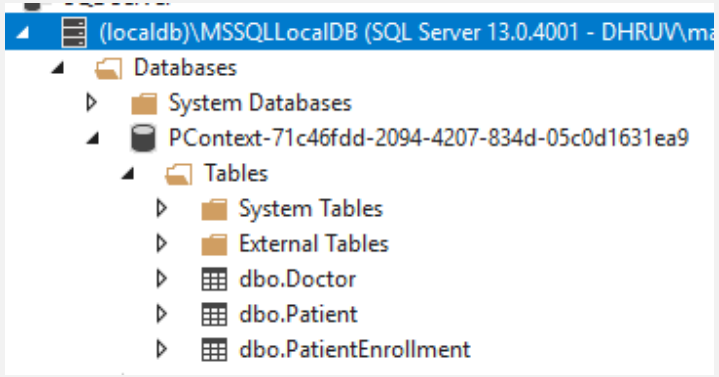
Creating Razor Pages Application Using EF Core (4-5)

14. Click **Patients**. Then, click **Create New**.

15. Add a record and click **Create**.

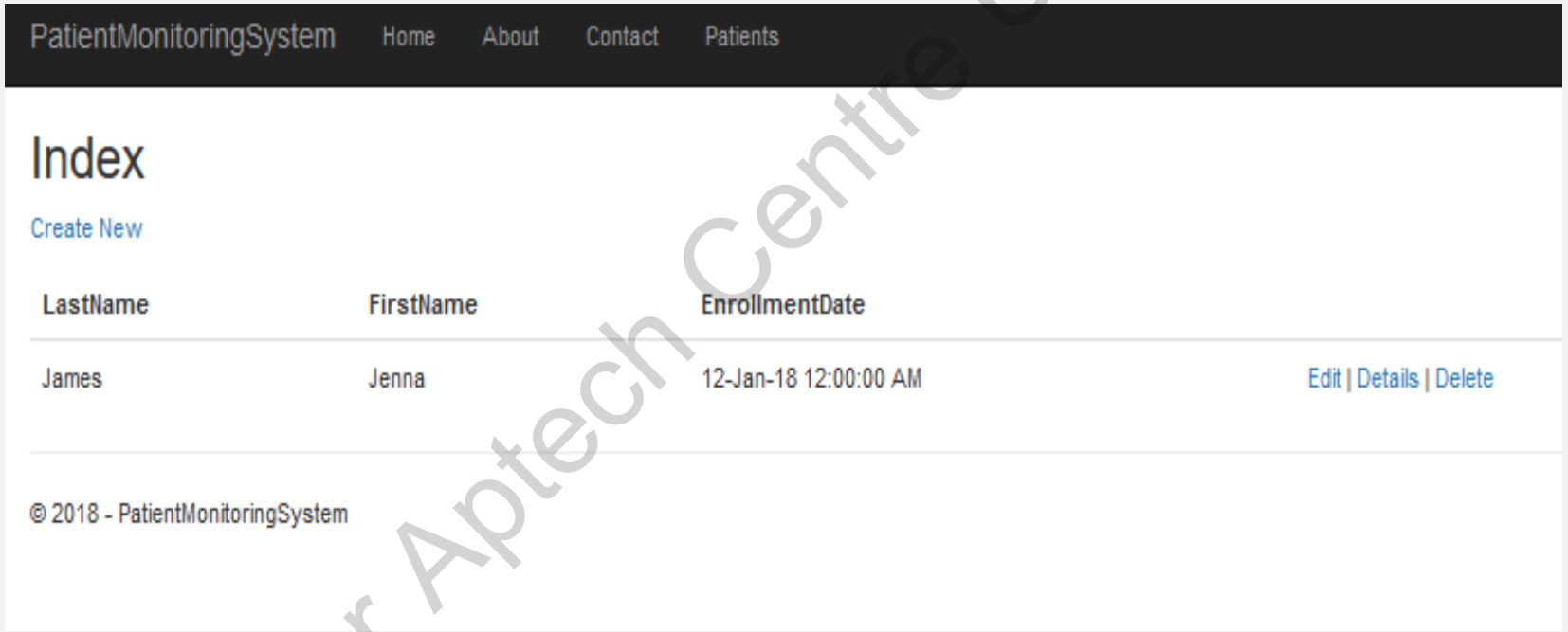
16. Click **View → SQL Server Object Explorer** to see the local database and tables generated.





Creating Razor Pages Application Using EF Core (5-5)

17. Right-click the table **Patient** and select **View Data**. The record entered through the browser page will be displayed in the table.



Deploying ASP.NET Core 2.1 Applications

An ASP.NET Core app can be deployed:

- As a self-contained local application by publishing to a folder
- As a Windows service
- To the cloud via Microsoft Azure
- On IIS

- When publishing to a local folder, publish folder will contain .exe and .dll files for the app, its dependencies, and optionally, the .NET runtime.

- Besides these files, the publish folder for an ASP.NET Core app also contains configuration files, static assets, and MVC views.
-
- Developers can publish ASP.NET Core Web apps to the cloud (on Azure App Service) using Visual Studio 2017 or from the command line.

Summary

- Razor Pages in ASP.NET Core 2.1 have several benefits including better organized structure of the project files.
- EF Core is a lightweight, extensible, and cross-platform version of Entity Framework.
- ASP.NET Core Razor Pages together with EF Core can be used to build Web apps for CRUD operations, sorting, filtering, and so on.
- ASP.NET Core Web apps can be deployed to a local folder, as a Windows service, as a Web application on IIS, or on the cloud on Azure App service.