go deploy

Screenshots

Module 3: Configuring Middleware and Services in ASP.NET Core

Lab: Configuring Middleware and Services in ASP.NET Core

Scenario

The Adventure Works company wants to develop a website about ball games. For this, the company needs to perform a survey to determine the popularity of different ball games. As their employee, you are required to create a survey site.

Exercise 1: Working with Static Files

Scenario

To create the poll, the application needs a styled HTML page. The HTML page must post the poll results to the server. To transfer the results to the server you will use an HTML form.

The main tasks for this exercise are the following:

- Create a new project by using the ASP.NET Core Empty project template
- Run the application
- Add an HTML file to the wwwroot folder
- Run the application - the content of the HTML file is not displayed
- Enable working with static files
- Run the application - the content of the HTML file is displayed
- Add an HTML file outside of the wwwroot folder
- Run the application - the content of the HTML file outside the wwwroot folder is not displayed

Task 1: Create a new project by using the ASP.NET Core Empty project template

- [ ] 1. In the **Start Page - Microsoft Visual Studio** window, on the **File** menu, point to **New**, and then click **Project**.
- [ ] 2. In the **Create a new project** dialog box, in the languages dropdown, ensure that **C#** is selected.
- [ ] 3. In the **Create a new project** dialog box, choose **ASP.NET Core Empty** and click **Next**.
- [ ] 4. In the **Project Name** text box, type **PollBall** and click **Next**.

6. In the **Additional information** dialog box, ensure that the **Configure for HTTPS** check box is **not** selected, and leave the other settings as their default values.
7. Click on **Create**.
8. In the **PollBall - Microsoft Visual Studio** window, on the **Debug** menu, click **Start Without Debugging**.
9. In Microsoft Edge, in the address bar, note the port number that appears at the end of the URL **http://localhost:[port]**. You will use the port number during this lab.
10. In Microsoft Edge, click **Close**.
11. In the **PollBall - Microsoft Visual Studio** window, in Solution Explorer, click **Program.cs**.
12. In the **Program.cs** code window, find the following line of code:

```
app.MapGet("/", () => "Hello World!");
```

13. Replace this with the following code:

```
app.Run(async (context) =>
{
      await context.Response.WriteAsync("This text was generated by the app.Run mid
});
```

Task 2: Run the application

1. In the **PollBall - Microsoft Visual Studio** window, on the **File** menu, click **Save All**.
2. In the **PollBall -- Microsoft Visual Studio** window, on the **Debug** menu, click **Start Without Debugging**.

> **Note**: The browser displays **This text was generated by the app.Run middleware.**

3. In Microsoft Edge, click **Close**.

Task 3: Add an HTML file to the wwwroot folder

1. In the **PollBall - Microsoft Visual Studio** window, in Solution Explorer, right-click the **PollBall** project, point to **Add**, and then click **New Folder**.
2. In the **NewFolder** box, enter **wwwroot**, and then press Enter.
3. In the **PollBall - Microsoft Visual Studio** window, in Solution Explorer, right-click **wwwroot**, point

4. In the **NewFolder** box, enter **css**, and then press Enter.
5. In the **PollBall - Microsoft Visual Studio** window, in Solution Explorer, right-click **css**, point to **Add**, and then click **Existing Item...**.
6. In the **Add Existing - PollBall** dialog, go to **D:Allfiles\Mod03\Labfiles\01_PollBall_begin** and choose the file **style.css**, and then click **Add**.
7. Open File Explorer and browse to **D:Allfiles\Mod03\Labfiles\01_PollBall_begin**.
8. Position the File Explorer window so that the Microsoft Visual Studio Solution Explorer pane is visible, and then drag the images folder onto **wwwroot** in Solution Explorer, to copy the images folder and its contents into the project.

   > ⚠ **Note**: Verify that in Solution Explorer, under **wwwroot**, the **images** folder is displayed.

9. In the **PollBall - Microsoft Visual Studio** window, in Solution Explorer, right-click **wwwroot**, point to **Add**, and then click **New Item...**.
10. In the **Add New Item -- PollBall** dialog box, choose **HTML Page**.
11. In the **Add New Item -- PollBall** dialog box, In the **Name** box, enter **poll-questions**, and then click **Add**.
12. In the **poll-questions.html** code window, in the **BODY** element, enter the following code.

```
<p>
   <h1>Favorite Ball Game Poll</h1>
   Please select your favorite ball game, and then press Submit Poll.
</p>
```

13. In the **BODY** element, below the **P** element, enter the following code:

go deploy

```
            <div class="main-batch1">
                  <div class="item">
                     <div class="img-item"><img src="images\basketball.png" /></div>
                     <div class="input-item"><input type="radio" name="favorite" value="B
                  </div>
                  <div class="item">
                     <div class="img-item"><img src="images\football.png" /></div>
                     <div class="input-item"><input type="radio" name="favorite" value="F
                  </div>
                  <div class="item">
                     <div class="img-item"><img src="images\rugby.png" /></div>
                     <div class="input-item"><input type="radio" name="favorite" value="R
                  </div>
                  <div class="item">
                     <div class="img-item"><img src="images\volleyball.png" /></div>
                     <div class="input-item"><input type="radio" name="favorite" value="V
                  </div>
            </div>
            <div class="main-batch2">
                  <div class="item">
                     <div class="img-item"><img src="images\billiard.png" /></div>
                     <div class="input-item"><input type="radio" name="favorite" value="B
                  </div>
                  <div class="item">
                     <div class="img-item"><img src="images\golf.png" /></div>
                     <div class="input-item"><input type="radio" name="favorite" value="G
                  </div>
                  <div class="item">
                     <div class="img-item"><img src="images\hockey.png" /></div>
                     <div class="input-item"><input type="radio" name="favorite" value="H
                  </div>
                  <div class="item">
                     <div class="img-item"><img src="images\tennis.png" /></div>
                     <div class="input-item"><input type="radio" name="favorite" value="T
                  </div>
            </div>
         </div>
         <div class="submit-batch">
            <input type="submit" value="Submit Poll" />
         </div>
      </form>
```

Task 4: Run the application -- content of the HTML file is not displayed

- [ ] 1. In the **PollBall - Microsoft Visual Studio** window, on the **File** menu, click **Save All**.
- [ ] 2. In the **PollBall -- Microsoft Visual Studio** window, on the **Debug** menu, click **Start Without Debugging**.

go deploy

> **Note**: The browser displays **This text was generated by the app.Run middleware.** and not the content of the **poll-questions.html** file.

- [ ] 4. In Microsoft Edge, click **Close**.

Task 5: Enable working with static files

- [ ] 1. In the **PollBall - Microsoft Visual Studio** window, in Solution Explorer, click **Program.cs**.
- [ ] 2. In the **Program.cs** code window, before the line **app.Run(async (context) =>**, add the following code:

```
app.UseStaticFiles();
```

Task 6: Run the application -- content of the HTML file is displayed

- [ ] 1. In the **PollBall - Microsoft Visual Studio** window, on the **File** menu, click **Save All**.
- [ ] 2. In the **PollBall - Microsoft Visual Studio** window, on the **Debug** menu, click **Start Without Debugging**.
- [ ] 3. In Microsoft Edge, in the address bar, enter http://localhost:[port]/poll-questions.html, and then press Enter.

> **Note**: The browser displays the **poll-questions.html** file content, but the HTML content is not styled by a CSS file yet (e.g. no background, image size is wrong).

- [ ] 4. In Microsoft Edge, click **Close**.
- [ ] 5. In Solution Explorer, under **wwwroot**, click **poll-questions.html**.
- [ ] 6. In the **poll-questions.html** code window, in the **HEAD** element, below the **title** element, enter the following code:

```
<link type="text/css" rel="stylesheet" href="css/style.css" />
```

- [ ] 7. In the **PollBall - Microsoft Visual Studio** window, on the **File** menu, click **Save All**.
- [ ] 8. In the **PollBall -- Microsoft Visual Studio** window, on the **Debug** menu, click **Start Without Debugging**.
- [ ] 9. In Microsoft Edge the content should have been updated. If not, in the address bar, enter http://localhost:[port]/poll-questions.html, and then press Enter.

formatted an styled by using the **style.css** file.

- [ ] 10. In Microsoft Edge, click **Close**.

Task 7: Add an HTML file outside the wwwroot folder

- [ ] 1. In the **PollBall - Microsoft Visual Studio** window, in Solution Explorer, right-click the **PollBall** project, point to **Add**, and then click **Existing Item...**.
- [ ] 2. In the **Add Existing - PollBall** dialog, go to **D:Allfiles\Mod03\Labfiles\01_PollBall_begin**, set the file type filter in the dialog to **All Files (*.*)**, and choose the file **test.html**, and then click **Add**.

Task 8: Run the application -- content of the HTML file outside the wwwroot folder is not displayed

- [ ] 1. In the **PollBall -- Microsoft Visual Studio** window, on the **Debug** menu, click **Start Without Debugging**.
- [ ] 2. In Microsoft Edge, in the address bar, enter http://localhost:[port]/test.html, and then press Enter.

> **Note**: The browser displays **This text was generated by the app.Run middleware**. By default, the **UseStaticFiles** middleware cannot display static files that are outside the **wwwroot** directory.

- [ ] 3. In Microsoft Edge, click **Close**.

> **Result**: At the end of this exercise, you are able to work with static files inside a Microsoft ASP.NET Core project.

Exercise 2: Creating Custom Middleware

Scenario

The server needs to handle the client's request. You have been asked to find which ball game was chosen by the user. To do this, you will create a middleware component.

The main tasks for this exercise are the following:

- Create the middleware
- Run the application
- Change the order of the middleware

go deploy

2. In the **Program.cs** code window, locate the following code:

```
app.UseStaticFiles();
```

3. **Before** this line of code, enter the following:

```
app.Use(async (context, next) =>
{
   if (context.Request.Query.ContainsKey("favorite"))
   {
      string selectedValue = context.Request.Query["favorite"];
      await context.Response.WriteAsync("Selected value is: " + selectedValue);
   }
   else
   {
      await next.Invoke();
   }
});
```

Task 2: Run the application

1. In the **PollBall - Microsoft Visual Studio** window, on the **File** menu, click **Save All**.
2. In the **PollBall - Microsoft Visual Studio** window, on the **Debug** menu, click **Start Without Debugging**.
3. In Microsoft Edge, in the address bar, enter http://localhost:[port]/poll-questions.html, and then press Enter.
4. In Microsoft Edge, click **Basketball**, and then click **Submit Poll**.

> **Note**: The browser displays **Selected value is: Basketball**, which is generated by the **app.Use** middleware.

5. In Microsoft Edge, click **Close**.

Task 3: Change the order of middleware

1. In the **PollBall -- Microsoft Visual Studio** window, in Solution Explorer, click **Program.cs**.
2. In the **Program.cs** code window, select the following code:

```
app.UseStaticFiles();
```

4. In the **Program.cs** code window, place the cursor before the following code:

```
app.Use(async (context, next) =>
{
```

5. Right-click at the cursor location, click **Paste**, and then press Enter.
6. In the **PollBall - Microsoft Visual Studio** window, on the **File** menu, click **Save All**.
7. In the **PollBall - Microsoft Visual Studio** window, on the **Debug** menu, click **Start Without Debugging**.
8. In Microsoft Edge, in the address bar, enter http://localhost:[port]/poll-questions.html, and then press Enter.
9. In Microsoft Edge, click **Basketball**, and then click **Submit Poll**.

> **Note**: The poll no longer works. The browser displays the **poll-questions.html** file content located under **wwwroot** folder because the request was captured by the **UseStaticFiles** middleware before the **app.Use** middleware had an opportunity to be executed.

10. In Microsoft Edge, click **Close**.
11. In the **Program.cs** code window, restore the code by using undo (Ctrl-Z) until the **app.UseStaticFiles** line follows the **app.Use** block again.
12. In the **PollBall - Microsoft Visual Studio** window, on the **Debug** menu, click **Start Without Debugging**.
13. In Microsoft Edge, in the address bar, enter http://localhost:[port]/poll-questions.html, and then press Enter.
14. Verify that the poll is working correctly again.
15. In Microsoft Edge, click **Close**.

> **Result**: At the end of this exercise, you have created custom middleware and used it to receive submitted form data.

Exercise 3: Using Dependency Injection

Scenario

go deploy

The main tasks for this exercise are as follows:

- Define an interface for a service
- Define an implementation for the service
- Use dependency injection
- Run the application

Task 1: Define an interface for a service

1. In the **PollBall - Microsoft Visual Studio** window, in Solution Explorer, right-click **PollBall**, point to **Add**, and then click **New Folder**.
2. In the **NewFolder** box, enter **Services**, and then press Enter.
3. In the **PollBall - Microsoft Visual Studio** window, in Solution Explorer, right-click **Services**, point to **Add**, and then click **Class...**.
4. In the **Add New Item - PollBall** dialog box, in the **Name** box, enter **SelectedGame**, and then click **Add**.
5. In the **SelectedGame.cs** code window, select the following code:

```
public class SelectedGame
```

6. Replace the selected code with the following code:

```
public enum SelectedGame
```

7. Place the cursor within the **SelectedGame** enum code block, press Enter, and then enter the following code:

```
Basketball,
Football,
Rugby,
Volleyball,
Billiards,
Hockey,
Golf,
Tennis
```

8. In the **PollBall - Microsoft Visual Studio** window, in Solution Explorer, right-click **Services**, point to **Add**, and then click **New Item...**.
9. In the **Add New Item -- PollBall** dialog box, click **Interface**.

11. In the **IPollResultsService.cs** code window, verify that the interface is declared public, as follows:

```
public interface IPollResultsService
```

12. Place the cursor within the **IPollResultsService** interface code block, press Enter, and then enter the following code:

```
void AddVote(SelectedGame game);
SortedDictionary<SelectedGame, int> GetVoteResult();
```

Task 2: Define an implementation for the service

1. In the **PollBall - Microsoft Visual Studio** window, in Solution Explorer, right-click **Services**, point to **Add**, and then click **Class...**.
2. In the **Add New Item - PollBall** dialog box, in the **Name** box, enter **PollResultsService**, and then click **Add**.
3. In the **PollResultsService.cs** code window, locate the following code:

```
public class PollResultsService
```

4. Append the **IPollResultsService** interface to the class definition so it looks like the following:

```
public class PollResultsService : IPollResultsService
```

5. In the body of the **PollResultsService** class, enter the following code:

```
private Dictionary<SelectedGame, int> _selectionVotes;

public PollResultsService()
{
   _selectionVotes = new Dictionary<SelectedGame, int>();
}
public void AddVote(SelectedGame game)
{

}
```

6. Place the cursor within the **AddVote** method code block, and then enter the following code:

```
            _selectionVotes[game]++;
        }
        else
        {
            _selectionVotes.Add(game, 1);
        }
```

- [ ] 7. Place the cursor at the end of the **AddVote** method, press Enter twice, and then enter the following code:

```
public SortedDictionary<SelectedGame, int> GetVoteResult()
{
    return new SortedDictionary<SelectedGame, int>(_selectionVotes);
}
```

Task 3: Use dependency injection

- [ ] 1. In the **PollBall - Microsoft Visual Studio** window, in Solution Explorer, click **Program.cs**.
- [ ] 2. Locate the following line of code:

```
var app = builder.Build();
```

- [ ] 3. On the preceding line, enter the following code and press Enter:

```
builder.Services.AddSingleton<IPollResultsService, PollResultsService>();
```

> ⚠ Note: you must configure the services collection before the **builder.Build()** method is called.

- [ ] 4. Verify that Visual Studio has automatically added a reference to the **PollBall.Services** namespace at the beginning of **Program.cs**:

```
using PollBall.Services;
```

- [ ] 5. In the **Program.cs** code window, select the following code:

```
await context.Response.WriteAsync("This text was generated by the app.Run middlewar
```

- [ ] 6. Replace the selected code with the following code:

```
await context.Response.WriteAsync("This text was generated by the app.Run middlewar
```

```
app.Use(async (context, next) =>
```

8. Add the following code before the located statement:

```
using (var serviceScope = app.Services.CreateScope())
{
   var services = serviceScope.ServiceProvider;
   var pollResults = services.GetRequiredService<IPollResultsService>();
```

9. Add a **}** (closing brace) sign on a new line after the end of the app.Use section (i.e. after the line containing **});**).

10. In the **Program.cs** code window, locate the following code:

```
string selectedValue = context.Request.Query["favorite"];
```

11. Place the cursor at the end of the located code, press Enter, and then enter the following code:

```
SelectedGame selectedGame = (SelectedGame)Enum.Parse(typeof(SelectedGame), selected
pollResults.AddVote(selectedGame);
```

12. In the **Program.cs** code window, select the following code:

```
await context.Response.WriteAsync("Selected value is: " + selectedValue);
```

13. Replace the selected code with the following code:

```
SortedDictionary<SelectedGame, int> gameVotes = pollResults.GetVoteResult();
foreach (KeyValuePair<SelectedGame,int> currentVote in gameVotes)
{
   await context.Response.WriteAsync($"<div> Game name: {currentVote.Key}. Votes: {
}
```

Task 4: Run the application

1. In the **PollBall - Microsoft Visual Studio** window, on the **File** menu, click **Save All**.

2. In the **PollBall - Microsoft Visual Studio** window, on the **Debug** menu, click **Start Without Debugging**.

> **Note**: The browser displays **This text was generated by the app.Run middleware. wwwroot folder path: [local path of your wwwroot folder]**.

3. In Microsoft Edge, in the address bar, enter http://localhost:[port]/poll-questions.html, and then

- [ ] 4. In Microsoft Edge, click **Basketball**, and then click **Submit Poll**.

> **Note**: The browser displays: **"Game name: Basketball. Votes: 1"**

- [ ] 5. In Microsoft Edge, in the address bar, enter http://localhost:[port]/poll-questions.html, and then press Enter (or use the browser's back button).
- [ ] 6. In Microsoft Edge, click **Football**, and then click **Submit Poll**.

> **Note**: The browser displays: **Game name: Basketball. Votes: 1 Game name: Football. Votes: 1**

- [ ] 7. In Microsoft Edge, in the address bar, enter http://localhost:[port]/poll-questions.html, and then press Enter.
- [ ] 8. In Microsoft Edge, click **Basketball**, and then click **Submit Poll**.

> **Note**: The browser displays: **Game name: Basketball. Votes: 2 Game name: Football. Votes: 1**

- [ ] 9. In Microsoft Edge, click **Close**.

> ✓ **Result**: At the end of this exercise, you have registered and used a service called **PollResultsService** by using dependency injection.

Exercise 4: Injecting a Service into a Controller

go deploy

In this exercise, you will create an ASP.NET Core MVC controller to display the poll results. Although normally you would scaffold a complete MVC project automatically using Visual Studio, in this exercise you will do some of the configuration of MVC manually.

The main tasks for this exercise are the following:

- Enable working with MVC
- Add a controller
- Run the application
- Use dependency injection in a controller
- Run the application

Task 1: Enable working with MVC

1. In the **Program.cs** code window, locate the following code:

```
builder.Services.AddSingleton<IPollResultsService, PollResultsService>();
```

2. Place the cursor at the end of the located code, press Enter, and then enter the following code:

```
builder.Services.AddMvc();
```

3. In the **Program.cs** code window, locate the following code:

```
app.Run(async (context) =>
{
   await context.Response.WriteAsync("This text was generated by the app.Run middle
});
```

4. Replace the located code with the following code:

```
app.UseRouting();
app.MapDefaultControllerRoute();
```

Task 2: Add a controller

1. In the **PollBall - Microsoft Visual Studio** window, in Solution Explorer, right-click **PollBall**, point to **Add**, and then click **New Folder**.
2. In the **NewFolder** box, enter **Controllers**, and then press Enter.
3. In the **PollBall - Microsoft Visual Studio** window, in Solution Explorer, right-click **Controllers**, point to **Add**, and then click **Controller...**.

go deploy

5. In the **Add New Item - PollBall** dialog, in the **Name:** textbox, enter **HomeController**, and then click **Add** (if this is the first controller, the name may be pre-filled as HomeController).

6. In the **HomeController.cs** code window, select the following code:

```
return View();
```

7. Replace the selected code with the following code:

```
return Content("Hello from controller.");
```

Task 3: Run the application

1. In the **PollBall - Microsoft Visual Studio** window, on the **File** menu, click **Save All**.

2. In the **PollBall - Microsoft Visual Studio** window, on the **Debug** menu, click **Start Without Debugging**.

> **Note**: The browser displays **Hello from controller.**

3. In Microsoft Edge, click **Close**.

Task 4: Use dependency injection in a controller

1. In the **PollBall - Microsoft Visual Studio** window, in Solution Explorer, click **Program.cs**.

2. In the **Program.cs** code window, select the following code:

```
SortedDictionary<SelectedGame, int> gameVotes = pollResults.GetVoteResult();

foreach (KeyValuePair<SelectedGame, int> currentVote in gameVotes)
{
   await context.Response.WriteAsync($"<div> Game name: {currentVote.Key}. Votes: {
}
```

3. Replace the selected code with the following code:

```
context.Response.Headers.Add("content-type", "text/html");
await context.Response.WriteAsync("Thank you for submitting the poll. You may look
```

4. In the **PollBall - Microsoft Visual Studio** window, in Solution Explorer, expand **Controllers**, and then click **HomeController.cs**.

5. In the **HomeController.cs** code window, locate the following code:

go deploy

6. Place the cursor after the **{** (opening brace) sign, press Enter, and then enter the following code:

```
private IPollResultsService _pollResults;
```

7. Verify that Visual Studio has automatically added a reference to the **PollBall.Services** namespace at the beginning of **Program.cs**:

```
using PollBall.Services;
```

8. Place the cursor at the end of the **_pollResults** field code, press Enter, enter the following code, and then press Enter.

```
public HomeController(IPollResultsService pollResults)
{
   _pollResults = pollResults;
}
```

> **Note**: This uses constructor injection to give the controller access to the **PollResultsService**.

9. In the **HomeController.cs** code window, select the following code:

```
return Content("Hello from controller.");
```

10. Replace the selected code with the following code:

```
if (Request.Query.ContainsKey("submitted"))
{
   StringBuilder results = new StringBuilder();
   SortedDictionary<SelectedGame, int> voteList = _pollResults.GetVoteResult();

   foreach (var gameVotes in voteList)
   {
         results.Append($"Game name: {gameVotes.Key}. Votes: {gameVotes.Value}{Envi
   }

   return Content(results.ToString());
}
else
{
   return Redirect("poll-questions.html");
}
```

go deploy

2. In the **PollBall -- Microsoft Visual Studio** window, on the **Debug** menu, click **Start Without Debugging**.

> **Note**: The browser displays the **http://localhost:[port]/poll-questions.html** page.

3. In Microsoft Edge, click **Basketball**, and then click **Submit Poll**.

> **Note**: The browser displays **Thank you for submitting the poll. You may look at the poll results Here**.

4. In Microsoft Edge, click **Here**.

> **Note**: The browser displays: **Game name: Basketball. Votes: 1**

5. In Microsoft Edge, open a new window.
6. In the address bar, enter **http://localhost:[port]/poll-questions.html**, and then press Enter.
7. In Microsoft Edge, click **Football**, and then click **Submit Poll**.

> **Note**: The browser displays **Thank you for submitting the poll. You may look at the poll results Here**.

8. In Microsoft Edge, click **Here**.

> **Note**: The browser displays: **Game name: Basketball. Votes: 1 Game name: Football. Votes: 1**

9. Close all the Microsoft Edge windows.
10. In the **PollBall - Microsoft Visual Studio** window, on the **File** menu, click **Exit**.

> ✓ **Result**: At the end of this exercise, you have created a controller, and injected a service into it with dependency injection.