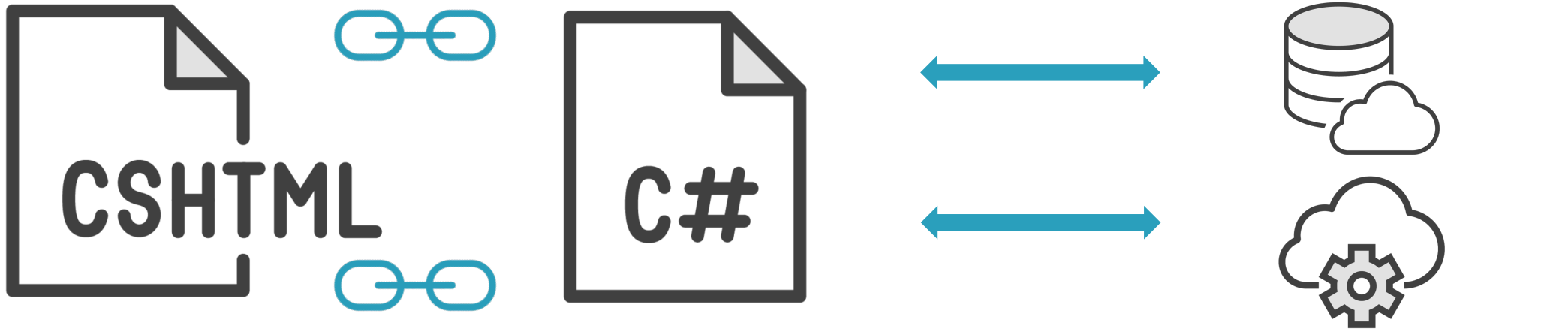
Exploring Pages, Layouts and Routing

Alex Wolf
.NET Developer

www.thecodewolf.com

One Razor Page – Two Files



Razor markup

C# page model class

Databases and services

The Anatomy of Razor Page Files

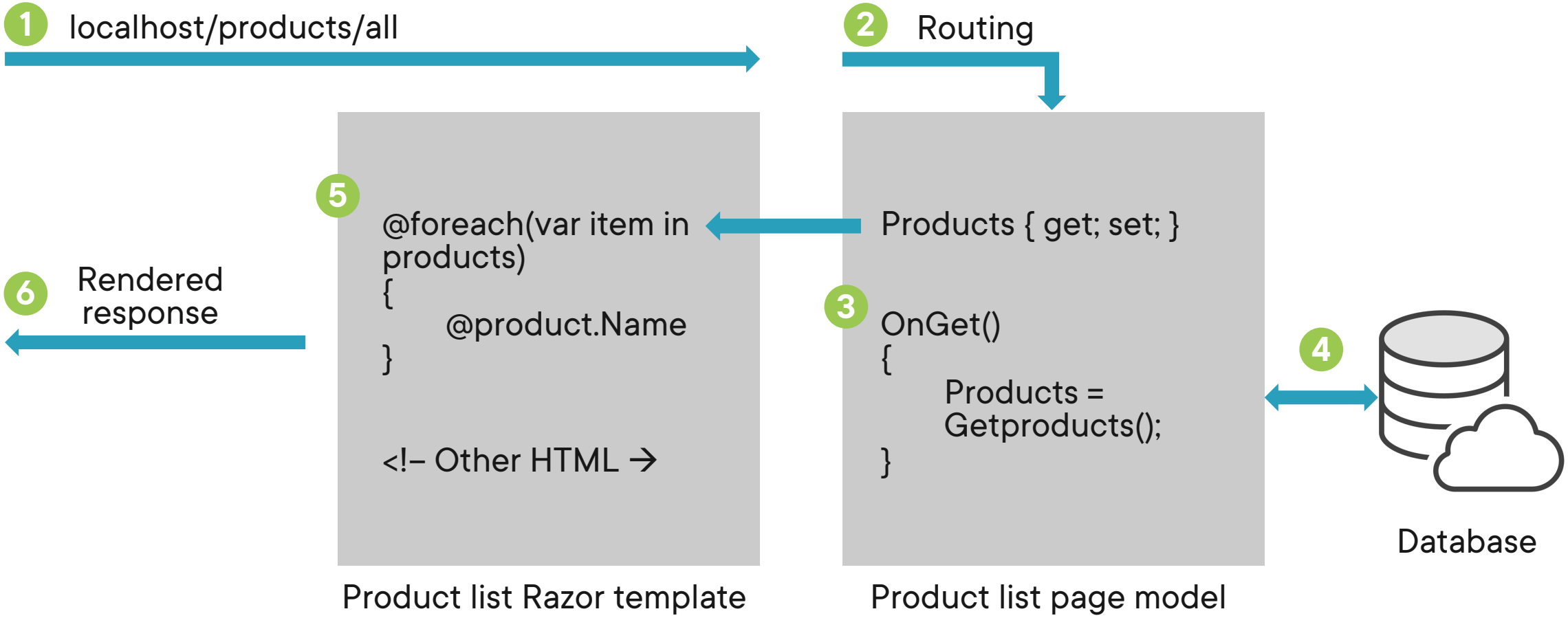
Index.cshtml

```
<!-- Defines a routable Razor Page -->
@page

<!-- Associates a Page Model class -->
@model IndexPageModel

<div>
    <!-- Razor and HTML markup -->
</div>
```

Index.cshtml.cs

```
public class IndexPageModel : PageModel
{

    // Page Model logic
}
```

Razor Pages Workflow In-depth



Razor Pages Handler Methods

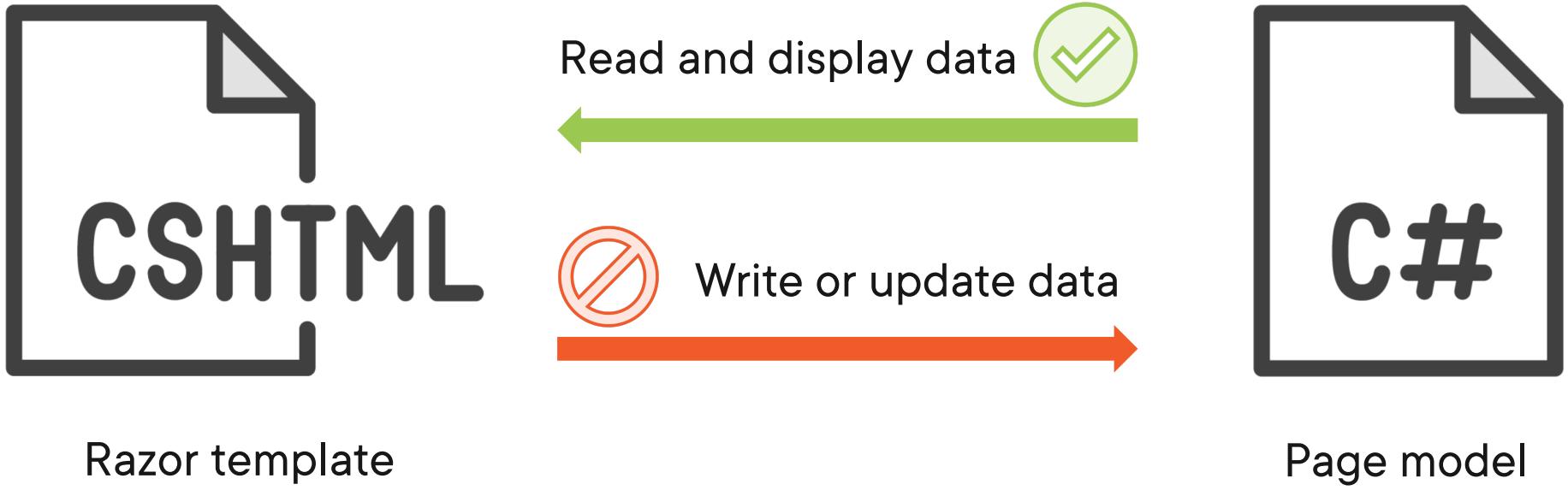
OnGet()

OnPost()

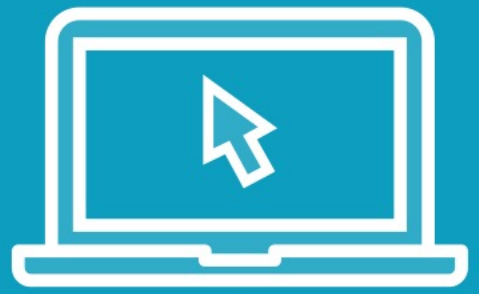
OnPut()

OnDelete()

Understanding One Way Data Binding



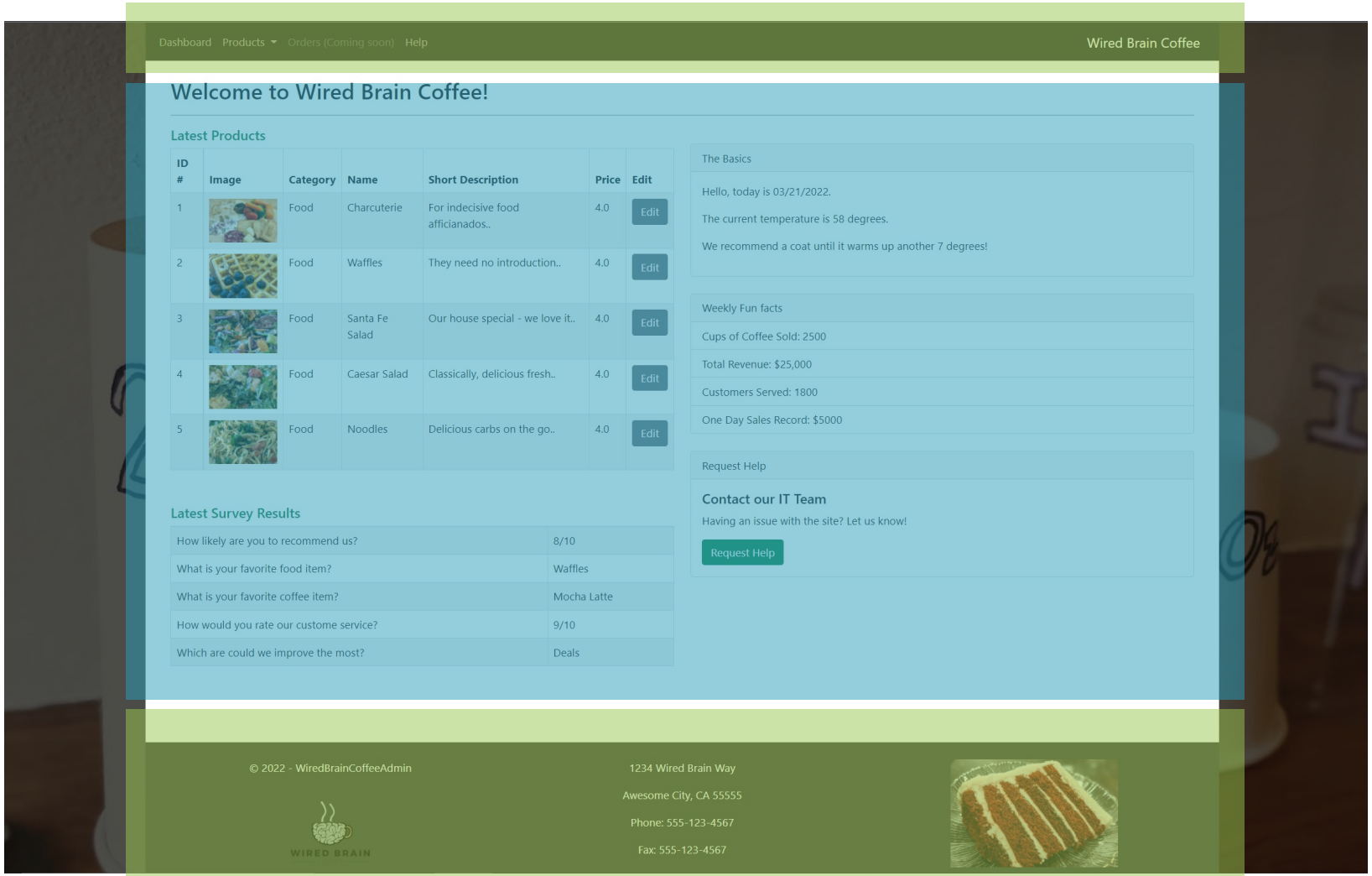
Demo



Working with requests and data binding

Working with Layouts

Defining Application Structure with Layouts



Layout

Page template

```
<html>
    <head>
        <link rel="site.css" />
        <link rel="bootstrap.css" />
    </head>
    <body>
        <nav>
            <!-- Nav items -->
        </nav>
        <div>
            @RenderBody()
        </div>
        <footer>
            <!-- Footer columns -->
        </footer>
    <script src="site.js">
    <script src="jquery.js">
    </body>
</html>
```

- Global style resources
- Structural site markup (navigation, footer)
- Inject generated Razor Pages markup
- Global script resources

Associating Layouts with Razor Pages

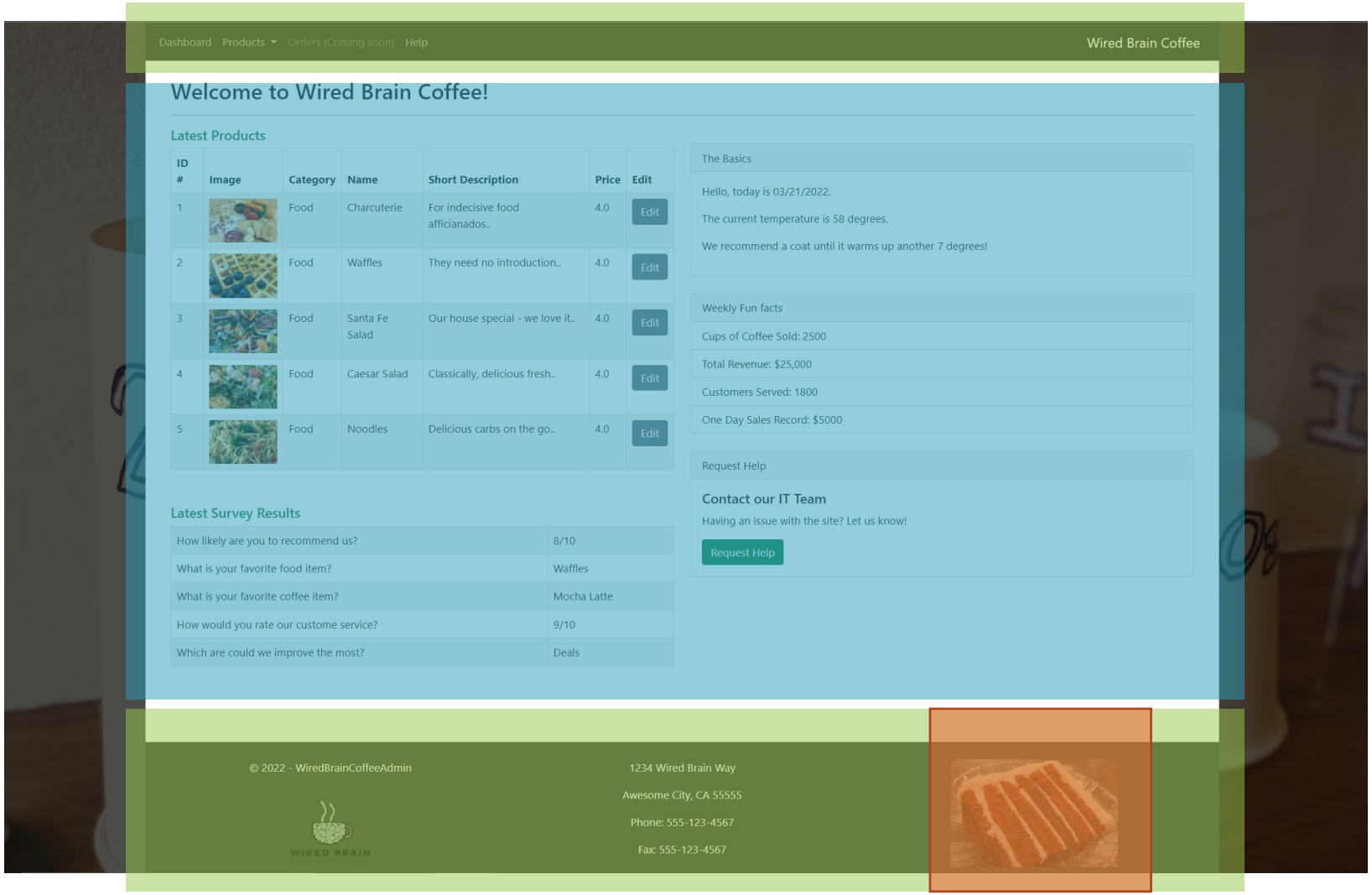
The ViewStart file

Defines a default layout for all pages on the site

The layout property

Allows specific Razor Pages to choose their own layout

Defining Application Structure with Layouts

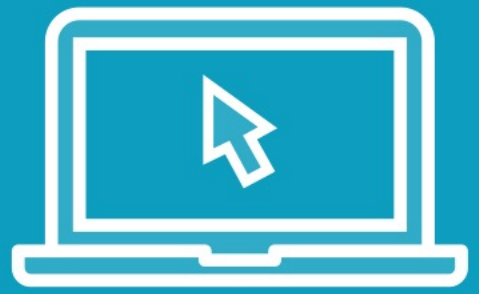


Layout

Page template

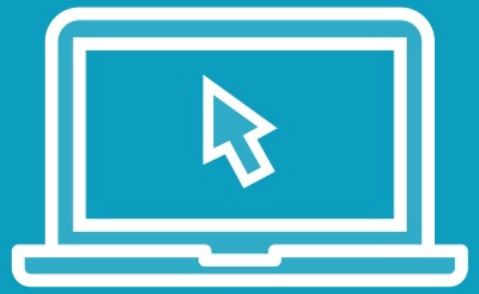
Overridable layout section

Demo



Exploring and configuring layouts

Demo



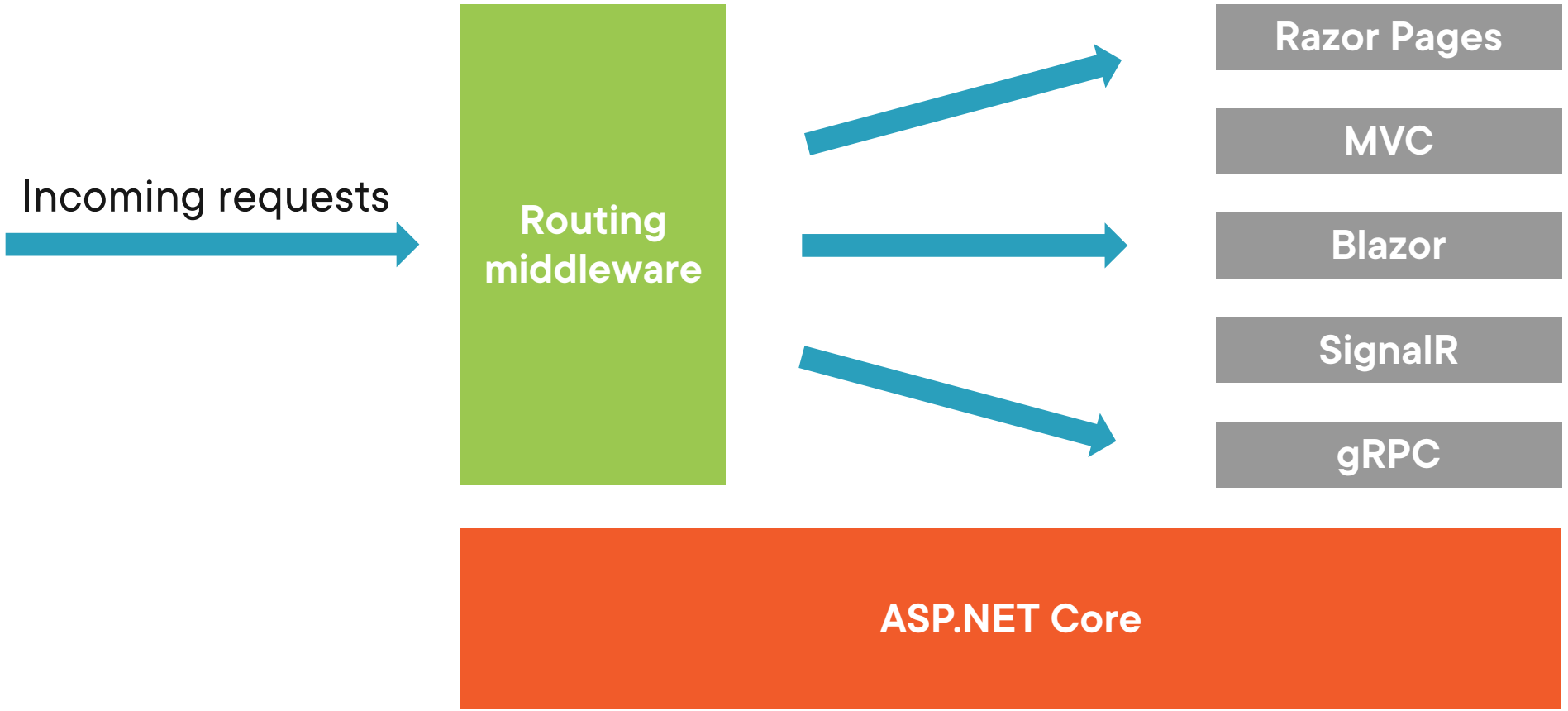
Working with layout sections

Understanding Routing

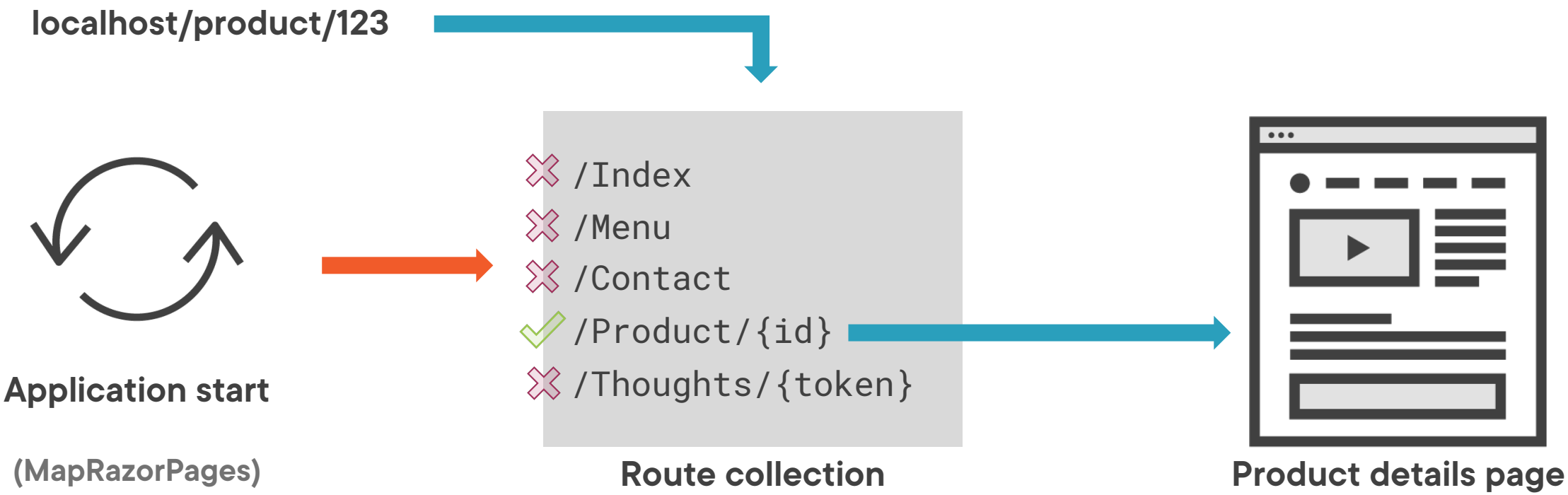
Routing

Matches an incoming HTTP request to a Razor Page using middleware

Routing and ASP.NET Core



Internal Routing Concepts

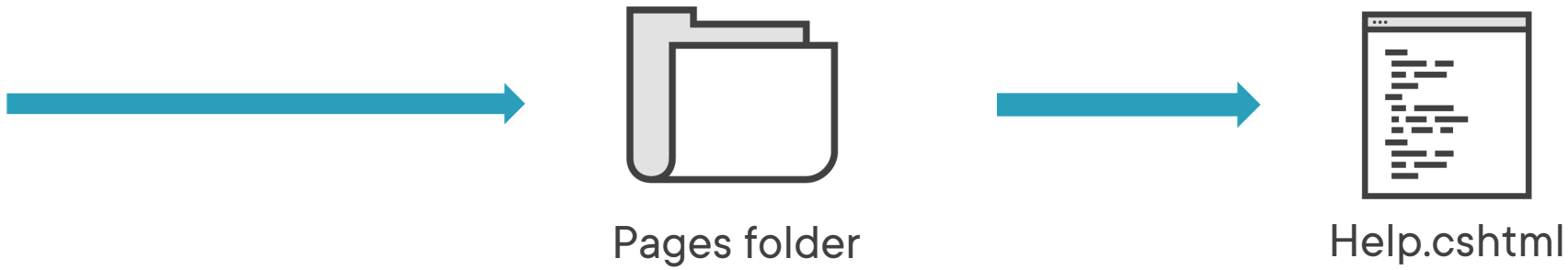


Route Mapping Conventions

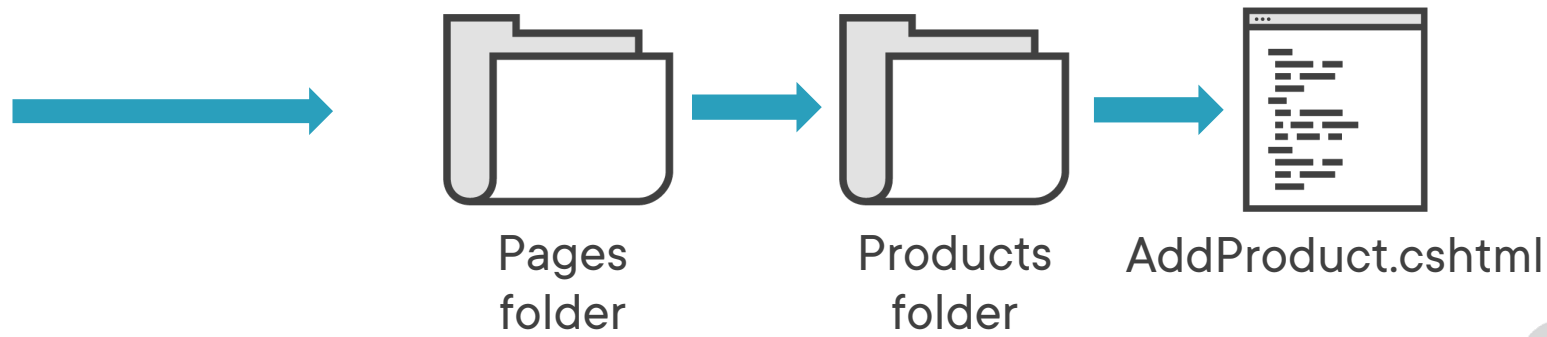
Incoming URL

Matching Project Structure

localhost/help



localhost/products/addproduct



Custom Route Templates

Contact.cshtml

```
<!-- Custom route template -->
@page “/contact-us”

@model ContactPageModel

<div>
    <!-- Razor and HTML markup -->
</div>
```

Products/AddProduct.cshtml

```
<!-- Custom route template -->
@page “/products/add”

@model ContactPageModel

<div>
    <!-- Razor and HTML markup -->
</div>
```

```
@page "/product/edit/{id:int}"

<!-- Template markup -->
```

Configuring Route Template Parameters

Define route parameters by wrapping the parameter name in curly braces

Constraints can be applied to enforce valid data types for that parameter

Tag Helpers and Routing

Anchor tag helper

```
<a asp-page="EditProduct" asp-route-id="@Item.Id">Edit</a>
```

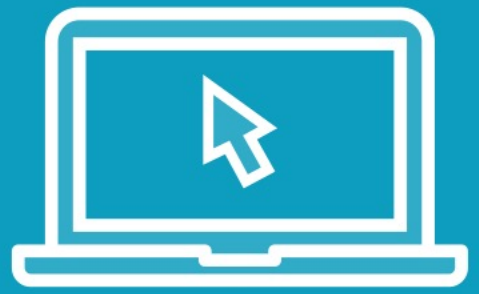
EditProduct page with custom route

/products/edit/{id}

Rendered HTML

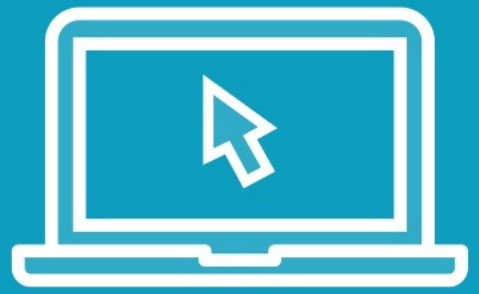
```
<a href="/products/edit/123">Edit</a>
```

Demo



Improving URLs using routing

Demo



Configuring route parameters

Summary

- Razor Pages consist of a Razor markup template and a C# page model class to manage data
- Data binding allows Razor to access and display properties on the page model
- Page models provide built-in handler methods for common HTTP requests, such as OnGet
- Layouts provide structure and consistent styling across a Razor Pages application
- Layout sections are areas of content that can be overridden by individual Razor Pages
- Routing matches an incoming URL to a specific Razor Page using middleware
- The routing system can define custom URL paths and parameters for individual pages