GETTING STARTED WITH ASP.NET CORE

Kamal Beydoun
Lebanese University – Faculty of Sciences I
Kamal.beydoun@ul.edu.lb

WHAT IS ASP.NET CORE?

- **ASP.NET Core Benefits:**
  - Offers structure, helper functions, and a framework for application development
  - Reduces the need for manual coding, enhancing efficiency
- **Application Flow:**
  - ASP.NET Core framework code manages the underlying structure
  - Your **handlers** are called by the framework, invoking your application's business logic
  - Business logic serves as the **core** of the application
- **Business Logic:**
  - Independent of ASP.NET Core
  - Involves interaction with other services, such as databases or remote APIs

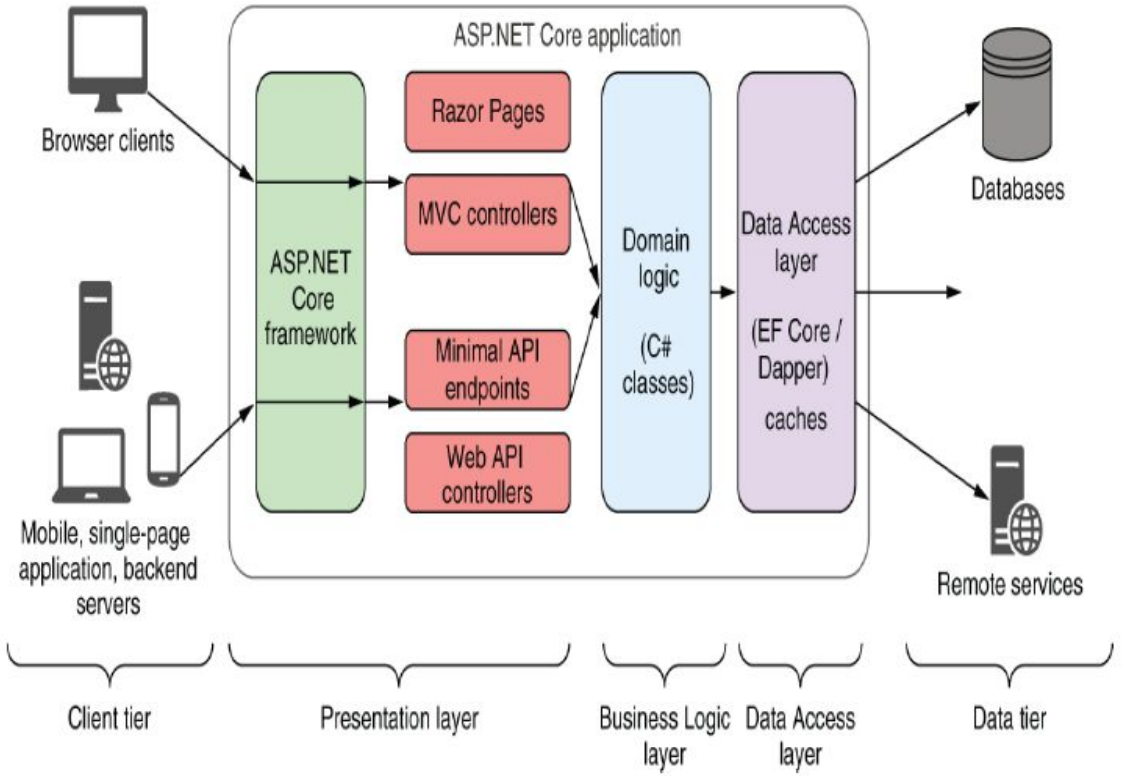
WHAT IS ASP.NET CORE?

ASP.NET Core apps can serve browser-based clients or can provide APIs for mobile and other clients.

The ASP.NET framework code handles the raw requests and calls in to Razor Pages and web API controller handlers.

You write these handlers using primitives provided by the framework. These typically invoke methods in your domain logic.

Your domain can use external services and databases to perform its function and to persist data.



WHAT TYPES OF APPLICATIONS CAN YOU BUILD?

- **API Support:**
  - **Minimal APIs:**
    - Simple HTTP APIs for mobile applications or browser-based single-page apps.
  - **Web APIs:**
    - More structured and feature-rich approach to building HTTP APIs.
  - **gRPC APIs:**
    - Efficient binary APIs for server-to-server communication using the gRPC protocol.
- **Application Paradigms:**
  - **Razor Pages:**
    - Used for building page-based server-rendered applications.
  - **MVC Controllers:**
    - Similar to Razor Pages but without the page-based paradigm.
    - Model-View-Controller (MVC) for server-based applications.

WHAT TYPES OF APPLICATIONS CAN YOU BUILD?

- **Blazor Framework:**
  - **Blazor WebAssembly:**
    - Browser-based single-page app framework using WebAssembly standard.
    - Comparable to JavaScript frameworks like Angular, React, and Vue.
  - **Blazor Server:**
    - Used for building stateful applications rendered on the server.
    - Combines the ease of development of a server-rendered application with the interactivity of client-side SPAs.

ASP.NET Core allows combining various paradigms within a single application.

WHAT TYPES OF APPLICATIONS CAN YOU BUILD?

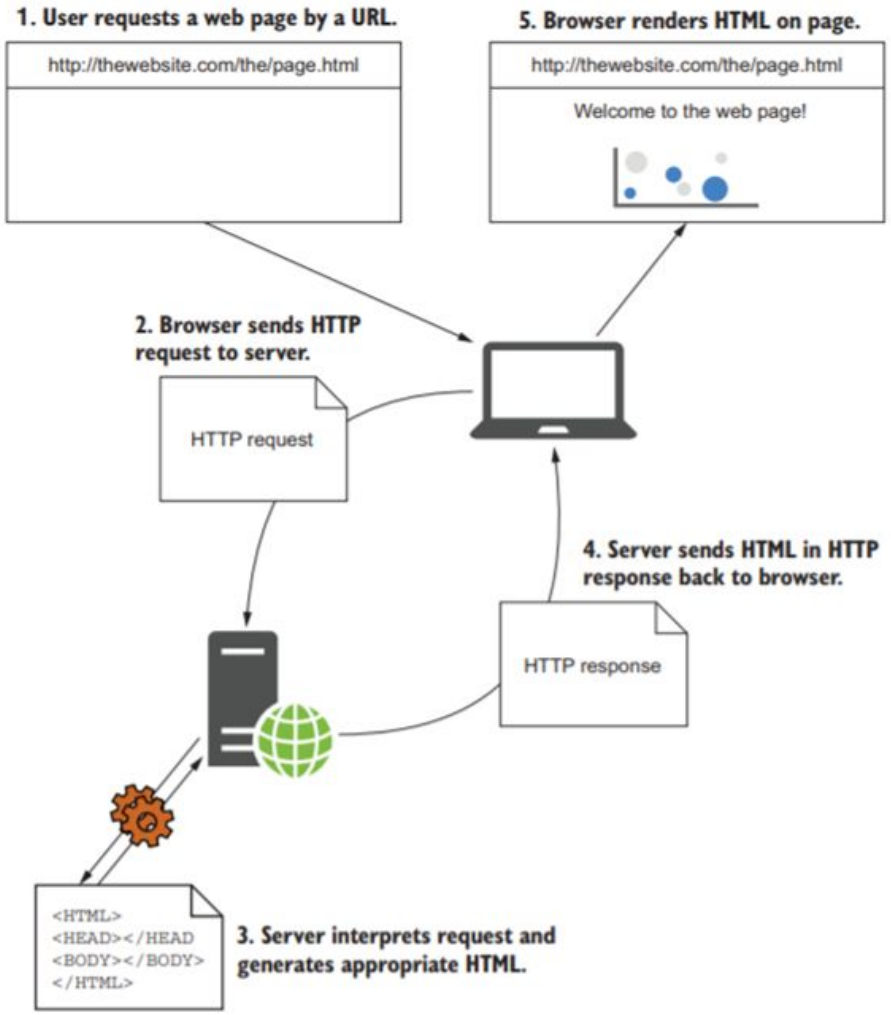
- **Shared Building Blocks:**
  - All ASP.NET Core paradigms leverage the same foundational elements, including configuration and logging libraries.
- **Factors Influencing Paradigm Choice:**
  - **API Requirements:**
    - Choose based on the specific needs of your APIs.
  - **Integration with Existing Applications:**
    - Consider details of interactions with existing applications.
  - **Customer Environment:**
    - Assess browser and operating environment details of your customers.
  - **Scalability and Uptime:**
    - Consider requirements for scalability and uptime.

CHOOSING ASP.NET CORE

- **ASP.NET Core has many selling points compared with other cross-platform web frameworks:**
  - It's a modern, high-performance, open-source web framework.
  - It uses familiar design patterns and paradigms.
  - C# is a great language (but you can use VB.NET or F# if you prefer).
  - You can build and run on any platform.

HOW DOES ASP.NET CORE WORK?

- **How does an HTTP web request work?**



HOW DOES ASP.NET CORE WORK?

- **How does ASP.NET Core process a request?**

Kestrel is responsible for receiving raw requests and constructing an internal representation of the data, an **HttpContext** object, which can be used by the rest of the application.