Asp.Net Core

Asp.Net Core is a cross-platform, high-performance, open-source framework for building modern, cloud-enabled web applications and services.

**Cross-platform**

Asp.Net Core apps can be hosted on Windows, LINUX and Mac.

**Can be hosted on different servers**

Supports Kestrel, IIS, Nginx, Docker, Apache

**Open-source**

Contributed by over 1000+ contributors on GitHub

https://github.com/dotnet/aspnetcore

**Cloud-enabled**

Out-of-box support for Microsoft Azure

Modules

**Asp.Net Core Mvc**

For creating medium to complex web applications

**Asp.Net Core Web API**

For creating RESTful services for all types of client applications.

**Asp.Net Core Razor Pages**

For creating simple & page-focused web applications

**Asp.Net Core Blazor**

For creating web applications with C# code both on client-side and server-side

Asp.Net Web Forms [vs] Asp.Net Mvc [vs] Asp.Net Core

**Asp.Net Web Forms**

* 2002
* Performance issues due to server events and view-state.
* Windows-only
* Not cloud-friendly
* Not open-source
* Event-driven development model.

**Asp.Net Mvc**

* 2009
* Performance issues due to some dependencies with asp.net (.net framework)
* Windows-only
* Slightly cloud-friendly
* Open source
* Model-view-controller (MVC) pattern

**Asp.Net Core**

* 2016
* Faster performance
* Cross-platform
* Cloud-friendly
* Open-source
* Model-view-controller (MVC) pattern

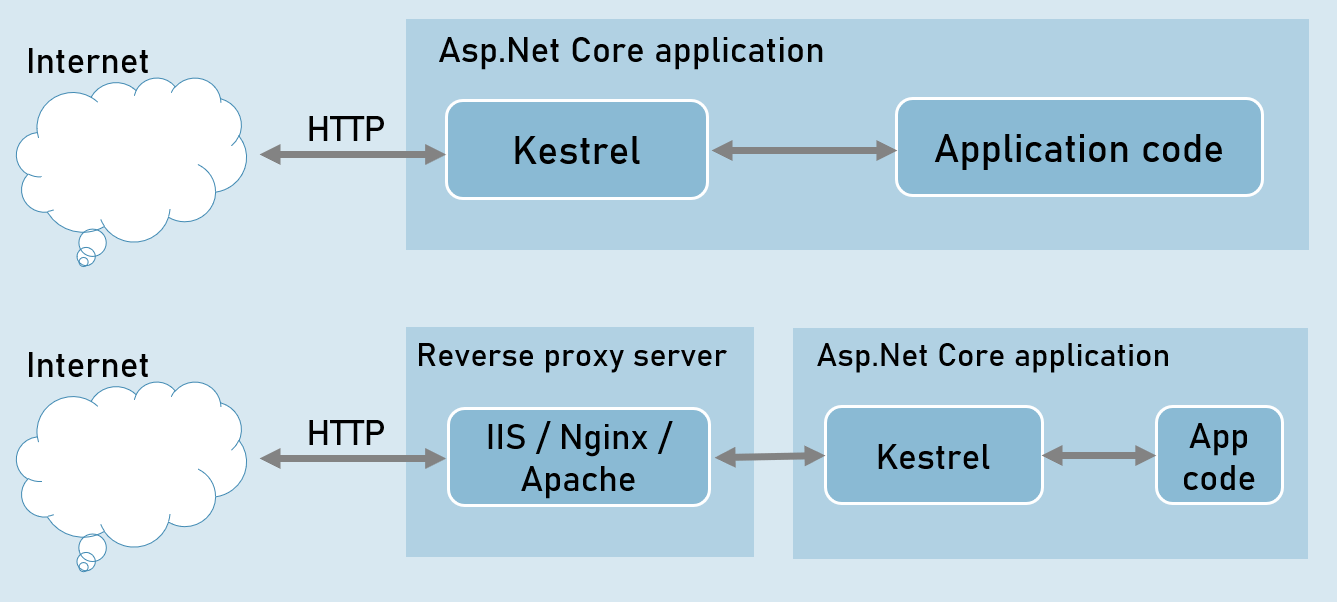
Kestrel and Other Servers

**Application Servers**

* Kestrel

**Reverse Proxy Servers**

* IIS
* Nginx
* Apache



**Benefits of Reverse Proxy Servers**

* Load Balancing
* Caching
* URL Rewriting
* Decompressing the requests
* Authentication
* Decryption of SSL Certificates

**IIS express**

* HTTP access logs
* Port sharing
* Windows authentication
* Management console
* Process activation
* Configuration API
* Request filters
* HTTP redirect rules

**Mülakat Soruları**

Answer the following questions as you are in an interview room.

**Bölüm1**

#### Questions for this assignment

**1-What is Asp.Net Core?**

ASP.NET Core is a free, open-source, and cross-platform framework for building cloud-based applications, such as web apps, IoT apps, and mobile backends. It is designed to run on the cloud as well as on-premises.

ASP.NET Core is not an upgraded version of ASP.NET. ASP.NET Core is completely rewriting that work with the .net Core framework. It is much faster, configurable, modular, scalable, extensible, and has cross-platform support. It is best suitable for developing cloud-based such as web applications, mobile applications, and IoT applications.

**2-What are the features of Asp.Net Core?**

Following are the core features that are provided by the ASP.NET Core

* Built-in supports for Dependency Injection
* Built-in supports for the logging framework and it can be extensible
* Introduced a new, fast and cross-platform web server - Kestrel. So, a web application can run without IIS, Apache, and Nginx.
* Multiple hosting platforms are supported
* It supports modularity, so the developer needs to include the module required by the application.
* Command-line supports to creating, building, and running of the application
* There is no web .config file. We can store the custom configuration into an appsettings.json file
* It has good support for asynchronous programming

**3-What are the advantages of ASP.NET Core over ASP.NET (.NET Framework)?**

There are the following advantages of ASP.NET Core over ASP.NET:

* It offers faster performance due to its minimalistic design
* It is cross-platform, so it can be run on Windows, Linux, and Mac.
* It is open-source
* There is no dependency on framework installation because all the required dependencies are shipped with our application to the production server
* Multiple deployment platforms available with ASP.NET Core

**4-What is Asp.Net Core meta package?**

The ASP.NET Core shared framework (Microsoft.AspNetCore.App) contains assemblies that are developed and supported by Microsoft. Microsoft.AspNetCore.App is installed when the .NET Core 3.0 or later SDK is installed. The shared framework is the set of assemblies (.dll files) that are installed on the machine and includes a runtime component and a targeting pack.

<Project Sdk="Microsoft.NET.Sdk.Web">

<PropertyGroup>

<TargetFramework>net6.0</TargetFramework>

</PropertyGroup>

...

</Project>

**5-When do you choose classic ASP.NET MVC over ASP.NET Core?**

Though Asp.Net Core is a better choice in almost all the aspects, you don’t have to switch to ASP.NET Core if you are maintaining a legacy ASP.NET application that you are happy with and that is no longer actively developed.

ASP.NET MVC is a better choice if you:

* Don’t need cross-platform support for your Web app.
* The existing team is already working on an existing app and extending its functionality.
* The existing developers needs a learning curve to upgrade themselves to Asp.Net Core

**6-What is a web application framework, and what are its benefits?**

Learning to build a modern web application can be daunting. Most of the web applications have a standard set of functionality such as:

* Build a dynamic response that corresponds to an HTTP request.
* Allow users to log into the application and manage their data.
* Store the data in the database.
* Handle database connections and transactions.
* Route URLs to appropriate methods.
* Supporting sessions, cookies, and user authorization.
* Format output (e.g. HTML, JSON, XML)
* Improve security.

Frameworks help developers to write, maintain and scale applications. They provide tools and libraries that simplify the above recurring tasks, eliminating a lot of unnecessary complexity.

**Bölüm2**

#### Questions for this assignment

**1-What is Kestrel and what are advantages of Kestrel in Asp.Net Core?**

Asp.Net Core application uses Kestrel by default.

Kestrel is an event-driven, I/O-based, open-source, cross-platform, and asynchronous server which hosts .NET applications. It is provided as a default server for .NET Core; therefore, it is compatible with all the platforms and their versions which .NET Core supports.

It is a listening server with a command-line interface.

It can be used to reverse proxy servers such as IIS, Nginx etc.

Features of Kestrel are:

* Lightweight and fast.
* Cross-platform and supports all versions of .NET Core.
* Supports HTTP & HTTPS.
* Easy configuration
* Multiple apps on the same port is not supported
* Windows authentication is not supported

**2-What is the difference between IIS and Kestrel? Why do we need two web servers?**

The main difference between IIS and Kestrel is that Kestrel is a cross-platform server. It runs on Windows, Linux, and Mac, whereas IIS only runs on Windows.

Another essential difference between the two is that Kestrel is fully open-source, whereas IIS is closed-source and developed and maintained only by Microsoft.

IIS is very old software and comes with a considerable legacy and bloat. With Kestrel, Microsoft started with high-performance in mind. They developed it from scratch, which allowed them to ignore the legacy/compatibility issues and focus on speed and efficiency.

However, Kestrel doesn’t provide all the rich functionality of a full-fledged web server such as IIS, Nginx, or Apache. Hence, we typically use it as an application server, with one of the above servers acting as a reverse proxy.

**3-What is the purpose of launchSettings.json in asp.net core?**

The launchSettings.json file is used to store the configuration information, which describes how to start the ASP.NET Core application, in Visual Studio.

It mainly contains the run time profiles to configure application urls’ and environment.

The file is used only during the development of the application using Visual Studio. It contains only those settings that required to run the application.

launchSettings.json is only used by Visual Studio.

You don't need launchSettings.json for publishing an app (on production server).

**4-What is generic host or HostBuilder in .NET Core?**

.NET generic host called ‘HostBuilder’ helps us to manage all the below tasks.

* Dependency Injection
* Service lifetime management
* Configuration
* Logging

The generic host was previously present as ‘Web Host’, in .NET Core for web applications. Later, the ‘Web Host’ was deprecated and a generic host was introduced to cater to the web, Windows, Linux, and console applications.

**5-What is the purpose of the .csproj file?**

The project file is one of the most important files in our application. It tells .NET how to build (compile) the project.

The csproj file stores list of package dependencies of the current project, target .net version and other compilation settings.

The .csproj file also contains all the information that .NET tooling needs to build the project. It includes the type of project you are building (console, web, desktop, etc.), the platform this project targets, and any dependencies on other projects or 3rd party libraries.

Here is an example of a .csproj file that lists the NuGet packages and their specific versions.

<Project Sdk="Microsoft.NET.Sdk.Web">

<PropertyGroup>

<TargetFramework>net6.0</TargetFramework>

</PropertyGroup>

<ItemGroup>

<PackageReference Include="PackageName" Version="1.0.0.0" />

</ItemGroup>

</Project>

**6-What is IIS?**

IIS stands for Internet Information Services. It is a powerful web server developed by Microsoft. IIS can also act as a load balancer to distribute incoming HTTP requests to different application servers to allow high reliability and scalability.

It can also act as a reverse proxy, i.e. accept a client’s request, forward it to an application server, and return the client’s response. A reverse proxy improves the security, reliability, and performance of your application.

A limitation of IIS is that it only runs on Windows. However, it is very configurable. You can configure it to suit your application’s specific needs.

**7-What is the “Startup” class in ASP.NET core prior to Asp.Net Core 6?**

The startup class is the entry point of the ASP.NET Core application. Every Asp.NET Core (Asp.Net Core 5 and earlier) application must have this class. It contains the necessary code to bootstrap the application. This class contains two methods Configure() and ConfigureServices().

* Configure(): The Configure() method is used to essential middleware(s) to the application request pipeline.
* ConfigureServices(): The ConfigureServices() method is used to add services to the IoC container.

In Asp.Net Core 6 (.NET 6), Microsoft unifies Startup.cs and Program.cs into a single Program.cs.

In asp.net core 6 (and up), we need to add all such as registering middleware to the application pipeline, adding services to the IoC container, configuring the ‘application configuration’, configuring the logger, authentication and adding DbContext in Program.cs file.

**8-What does WebApplication.CreateBuilder() do?**

This method does the following things.

* Configure the app to use Kestrel as web server.
* Specify to use the current project directory as root directory for the application.
* Setup the configuration sub-system to read setting from appsettings.json and appsettings.{env}.json to environment specific configuration.
* Set Local user secrets storage only for the development environment.
* Configure environment variables to allow for server-specific settings.
* Configure command line arguments (if any).
* Configure logging to read from the Logging section of the appsettings.json file and log to the Console and Debug window.
* Configure integration with IIS
* Configure the default service provider.