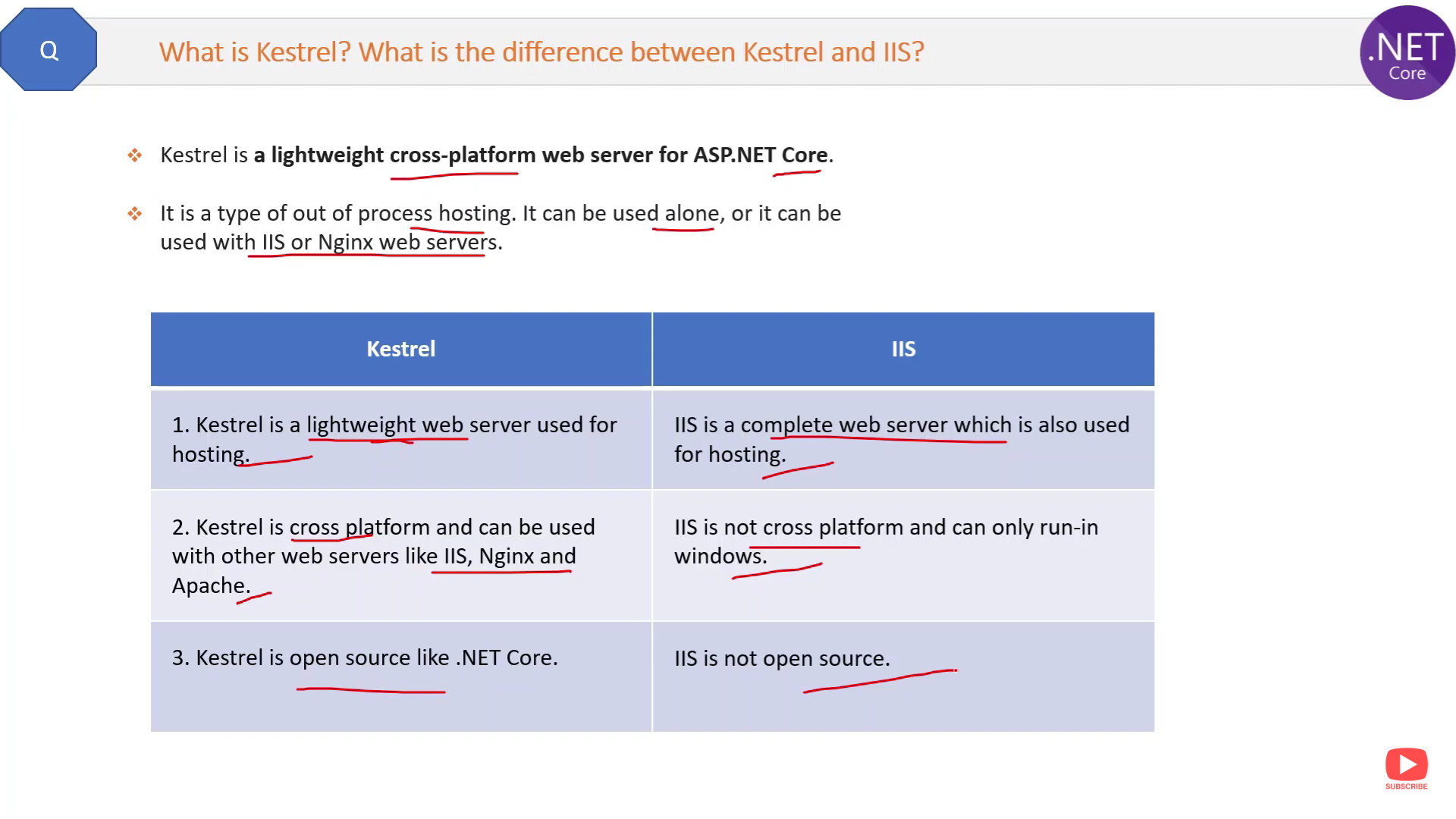
**Asp.Net core 7 (.net 7)**

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

We can create rest API services which act as backend for web and mobile applications.

**Features:**

1. **Cross-platform**- we can host our asp .net core apps in linux,mac,windows. generally in most of the cases we use Windows or Mac on the developer machine and Linux on the server mission
2. **Can be hosted on different servers**- supports kestrel,IIS,Nginx,Docker,Apache.  there should be a software that can receive the request and can provide the response which is also called **as reverse proxies**. asp.net core supports Kestrel by default as application server and the remaining as reverse proxies that is IIs, nginx and docker where the docker itself supports various operating systems you can run Windows or Linux on the docker. our asp.net core application can run on any of these servers, it is open source.(visual studio by default uses iis to run. But if we use CLI(command line interface) then it uses kestrel as web server. )



1. **Cloud-enabled** – out -of-box support for Microsoft azure

\* 4 parts of asp.net core

1.**Asp.net core MVC**- for creating medium to complex applications.

2.**Asp.net core web api**-(if we have model,controllers but no view ) to

create restful services so that it can receive the request and provide the data as response only the data but not the views so that the front end(reacr,angular-web app, mobile-xamarin,ionic) can be created by using any one of the web application or mobile application Frameworks.

3.**Asp.net core Razor pages**- for creating simple and page-focused.

4.**Asp.net core Blazor-** for creating with c# code both on client and server side .

**Asp.net weforms**(from -2002yr)

asp.net web forms mainly it has the performance drawbacks it offers slower performance due to server events and view state.

because asp.net web forms itself tries to make the web as stateful instead of stateless that means every page has to store its state and that state will be stored in the form of view state and that viewstate object has to be transferred from the client to server and server to client for each request. for simple web applications it may be okay but for medium to larger applications

it's the biggest drawback because the viewstate weight gets increased and that heavyweight viewstate has to be transferred for each request between the client and server so overall it is the slower performance.

and also in asp.net web forms for every request server page lifecycle has to be executed on the server that means there is a series of server events that executes for each request it is too complex and heavyweight so it is another reason of slower performance and

also there are other disadvantages such as unit testing is difficult

so with all these problems kept in mind asp.net mvc was introduced in 2009.

**Asp.net MVC(2009 yr)**

asp.net mvc uses model view and controller pattern that means it offers clean separation of concerns so that you can test your models views and controllers independently so you can unit test your controllers models and viewers but it has its own drawback because asp.net mvc is built on the top of some of the components that are already developed for asp.net web forms earlier for example system.web.dll so somewhere it offers slower performance partially and lack of support for cross-platform

that means it is quite difficult or impossible to host your asp.net mvc applications on other operating systems other than windows because asp.net mvc being a part of dotnet framework or at least it is tightly coupled with some of the components of dotnet

framework internally it makes it problematic to host on other operating systems it is not designed for that so we have to improve the performance and we have to enable the cloud friendliness and it should be cross platform these are the design goals of asp.net core asp.net core first version was released in 2016.

Asp.net webforms and .net mvc works on Microsoft.

**Asp.net core 6-**

cloud platform,cross-platform, support for clouds such as microsoft azure so that you will be able to host your application on the microsoft azure without having any server infrastructure on your development center.

**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**

ASP.NET framework is a highly testable and powerful framework. It's a web form based framework. You can create applications in ASP.NET (.aspx) or Razor page.

* 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
* **Asp.net core meta package** refers to Nuget package that serves a high level package containing reference to multiple related packages. these meta packages simplify the processs of adding dependencies to project by including collection of common libraries and components.
* we often choose ASP.NET MVC over ASP.NET Core. When we create an enterprise application, mvc is tested and proven for a decade. Don’t need a cross-platform support for your Web app. Need a stable environment to work in.