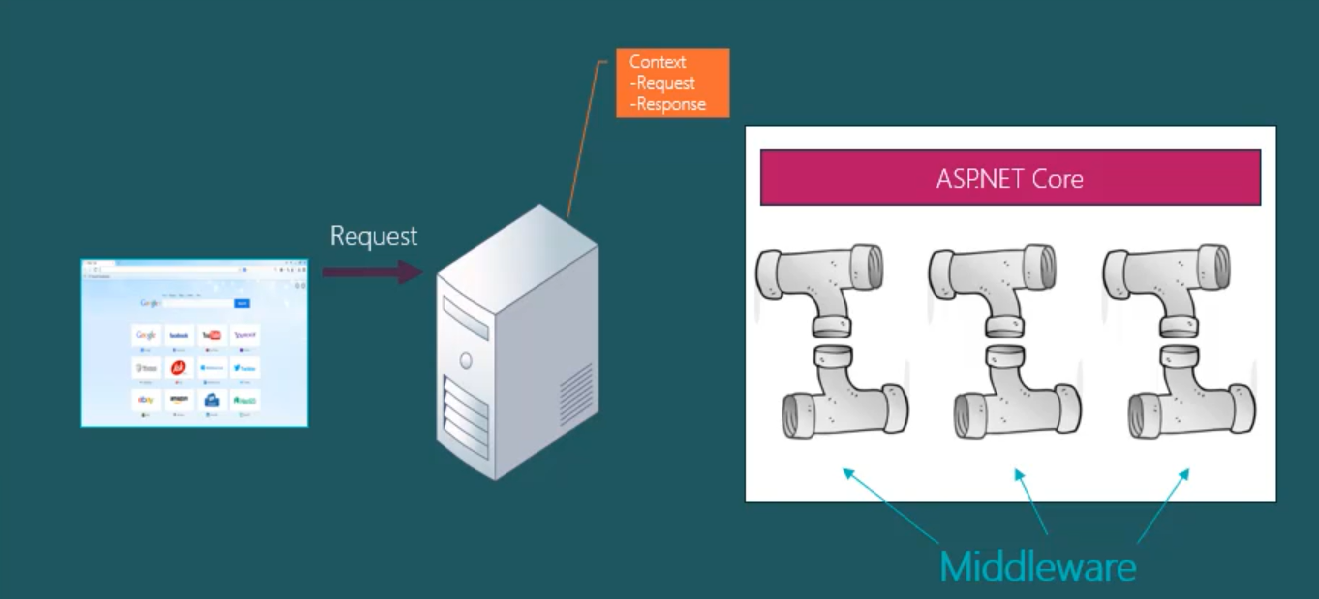
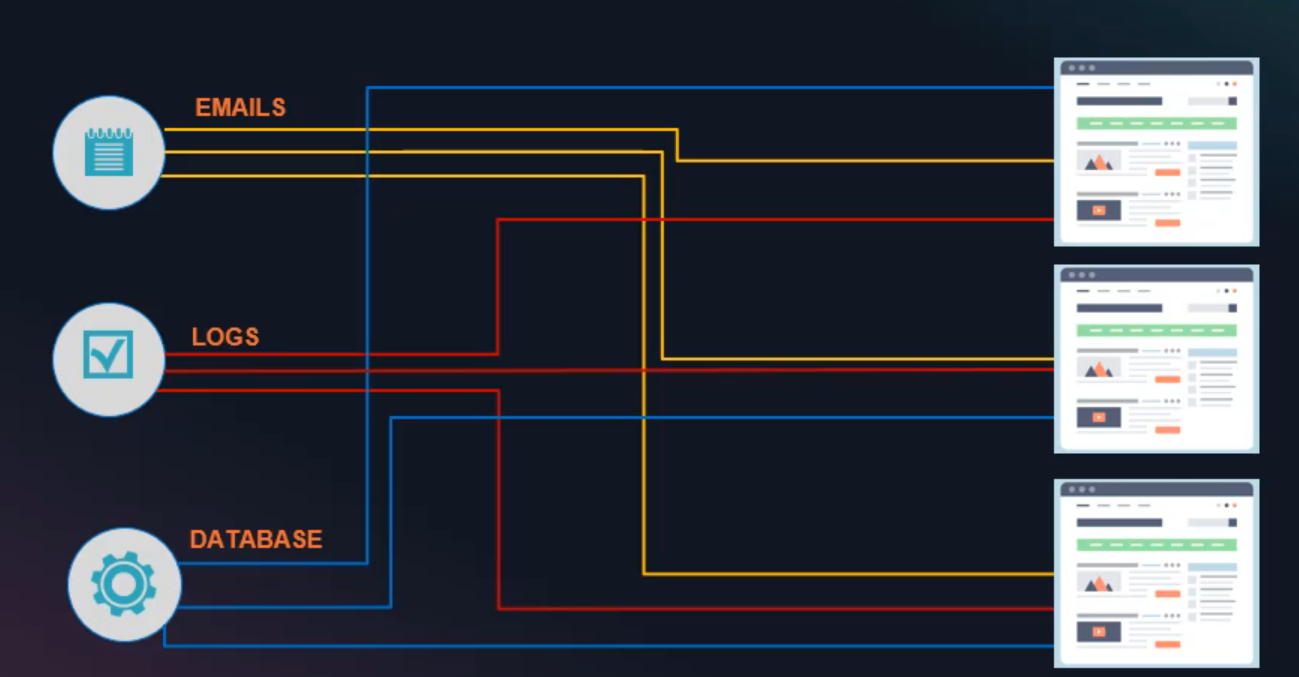
# Middleware en Request Pipeline in ASP.NET Core

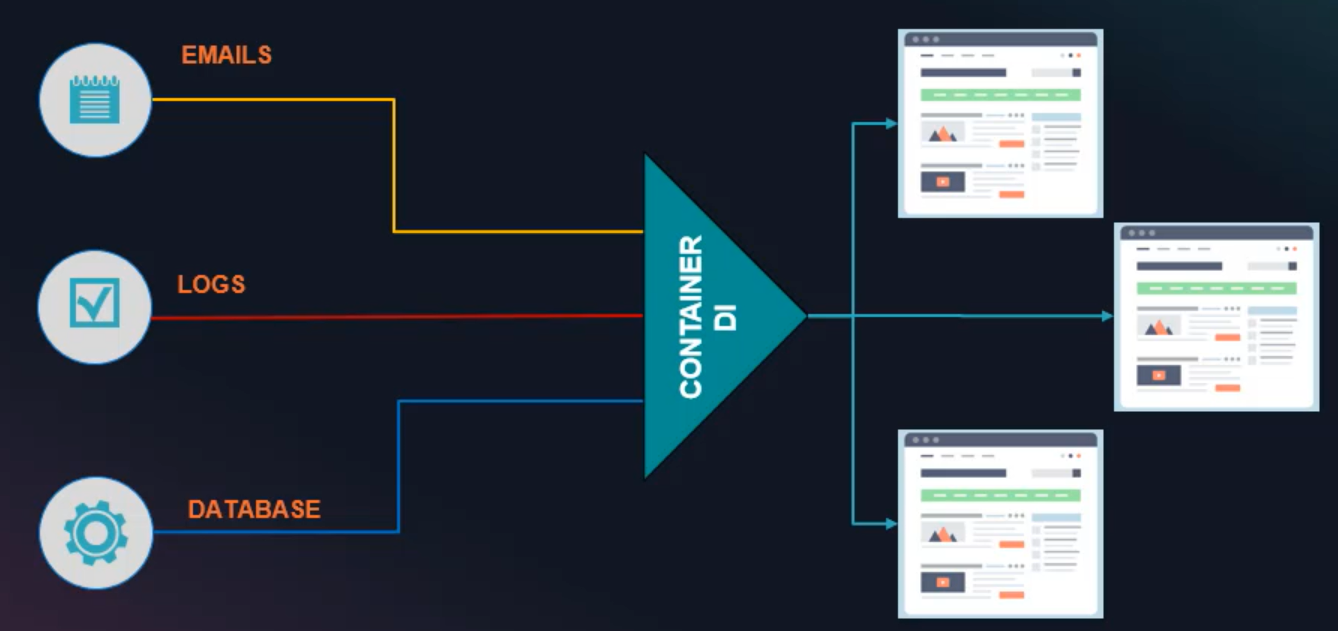


# Dependency Injection (DI)

**Zonder dependency injection**



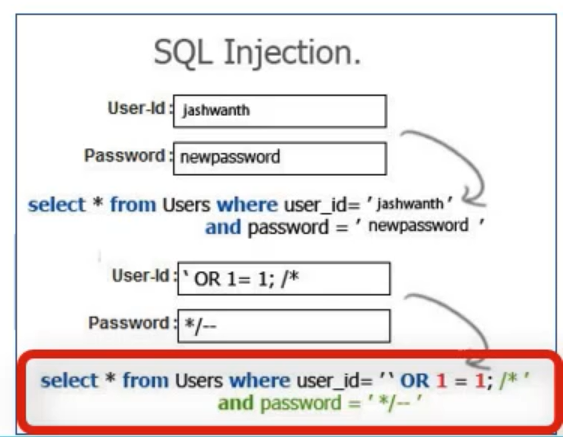
**Met dependency injection**



# DI Container van ASP.NET Core

# 

# SQL Injection attack



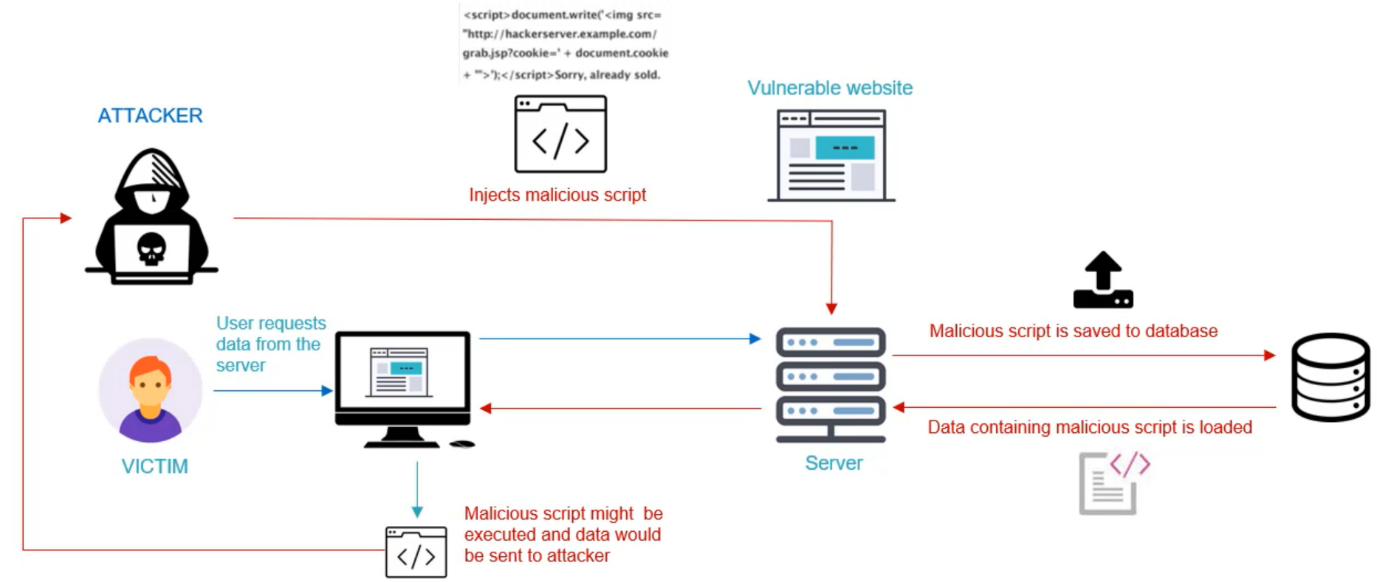
//zeer gevaarlijk om deze soort code te schrijven:!!!

String Sqlquery= “select \* from users where user\_id =” + txtuser + “ and password=’” +txtpwd

**Veiliger**: met SqlParameters of via Stored procs of via EF Core via entity classes (Categories…. Met LINQ)

# Cross-site Scripting (XSS)

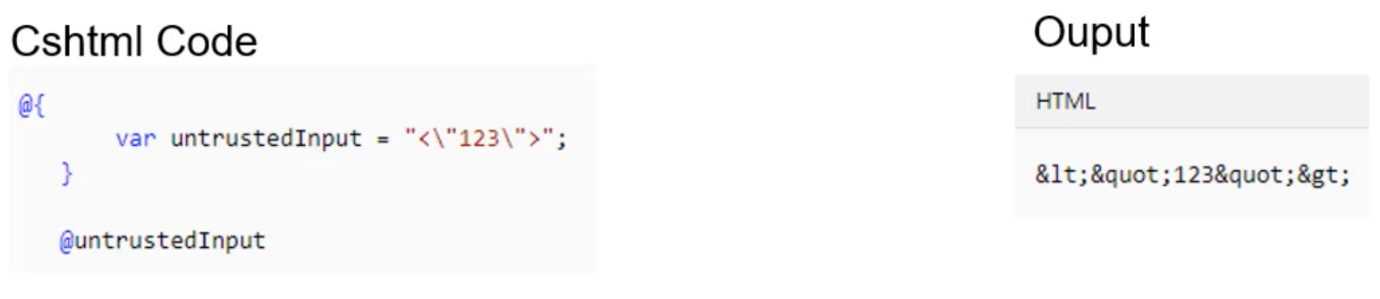
Is bedoeld om de eindgebruiker te hacken

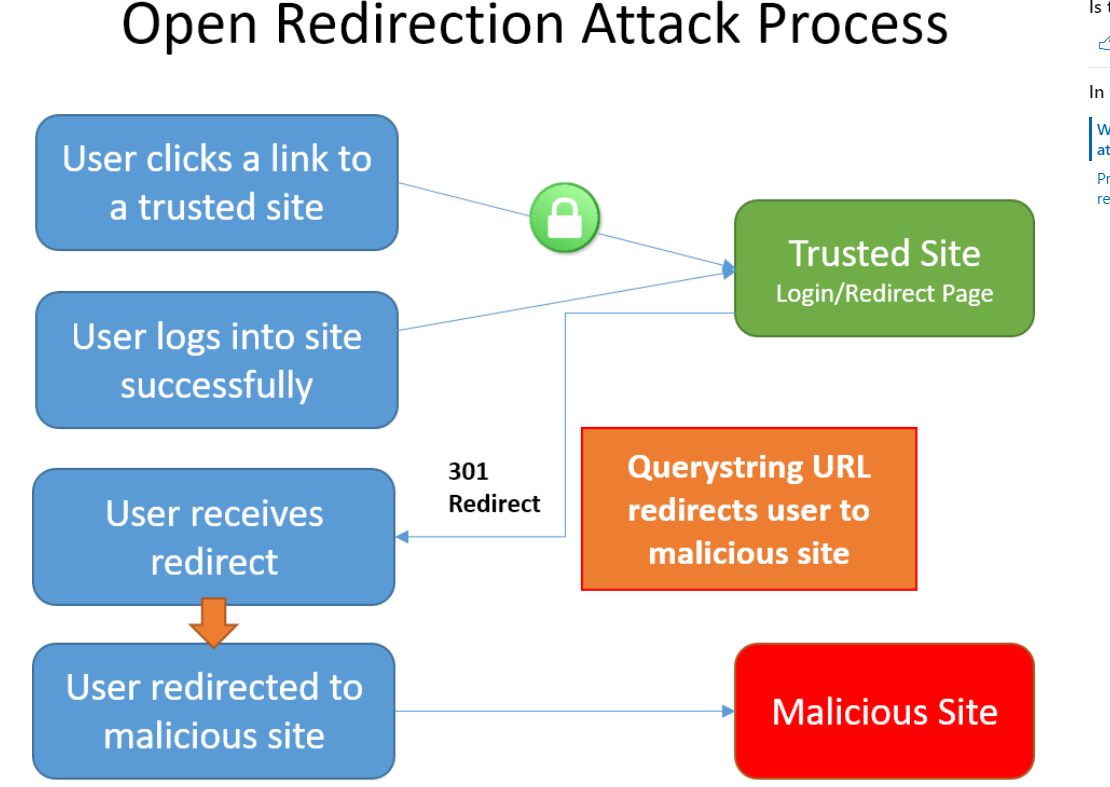


**Hoe voorkomt ASP.NET Core XSS attacks**

-via HTML encoding in Razor pages

-via validaties





# Voorkomen van Open Redirect Attacks in ASP.NET Core MVC

### **IsLocalUrl**: beschermt tegen het redirecten naar een kwaadaardige website

bv

private IActionResult RedirectToLocal(string returnUrl)

{

if (Url.IsLocalUrl(returnUrl))

{

return Redirect(returnUrl);

}

else

{

return RedirectToAction(nameof(HomeController.Index), "Home");

}

}

# Cross-Site Request Forgery (XSRF/CSRF) attacks in ASP.NET Core

<form method="post" asp-antiforgery="true">

...

</form>

Of

<form action="/" method="post">

@Html.AntiForgeryToken()

</form>

**Resultaat**

<input name="\_\_RequestVerificationToken" type="hidden" value="CfDJ8NrAkS ... s2-m9Yw">

<https://docs.microsoft.com/en-us/aspnet/core/security/preventing-open-redirects?view=aspnetcore-3.1>

<https://docs.microsoft.com/en-us/aspnet/core/security/anti-request-forgery?view=aspnetcore-3.1>

<https://docs.microsoft.com/en-us/aspnet/core/security/preventing-open-redirects?view=aspnetcore-3.1>