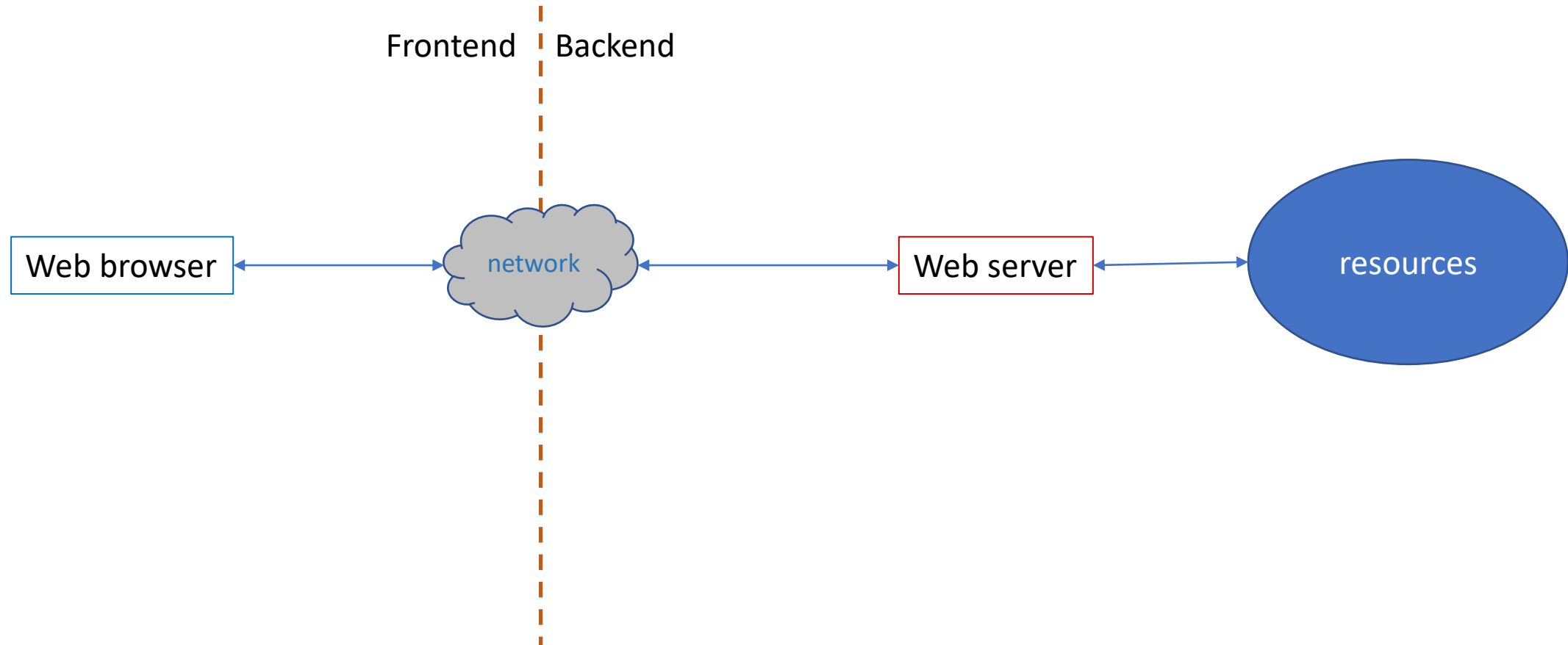


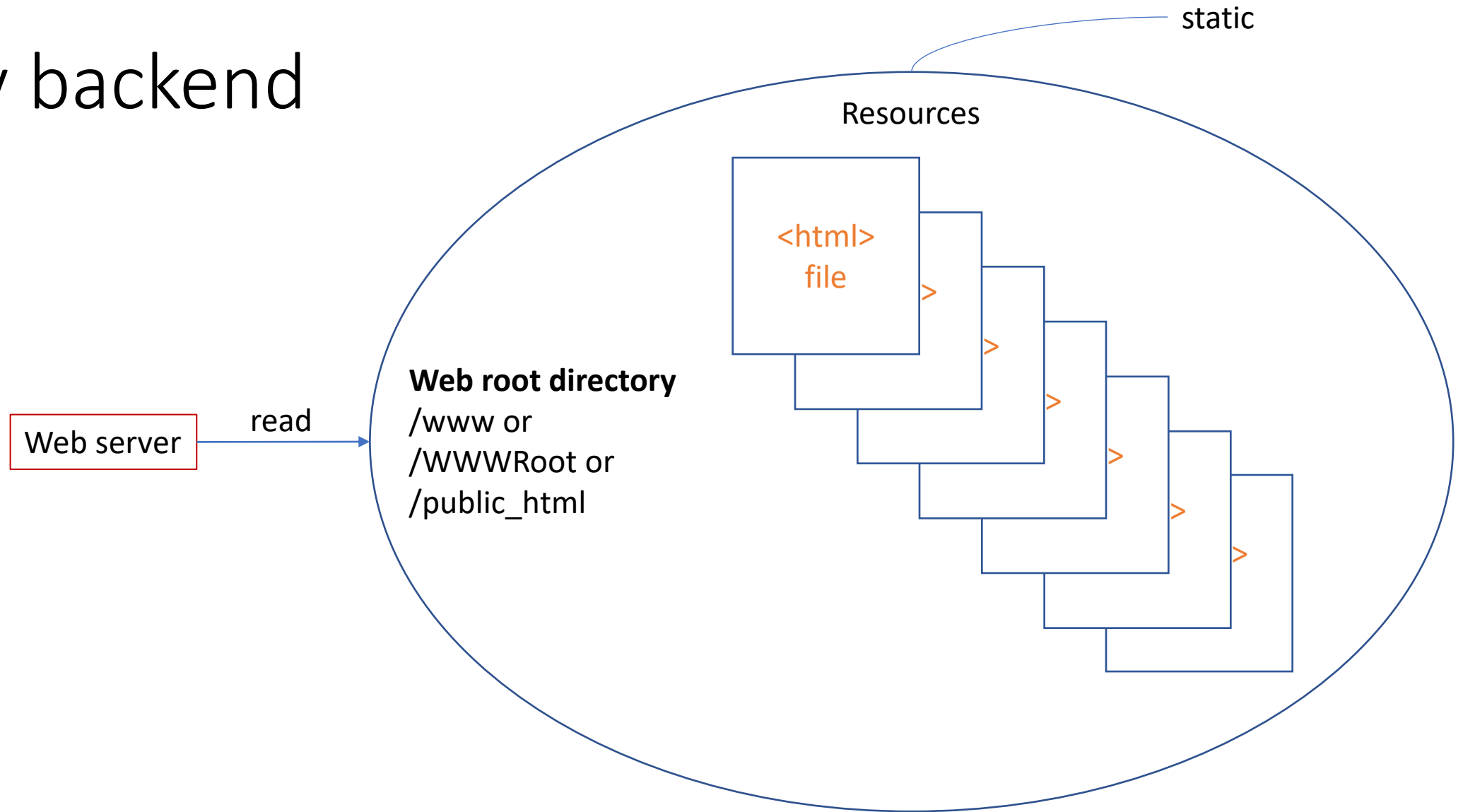
Backend

History & ASP.Net Core Introduction

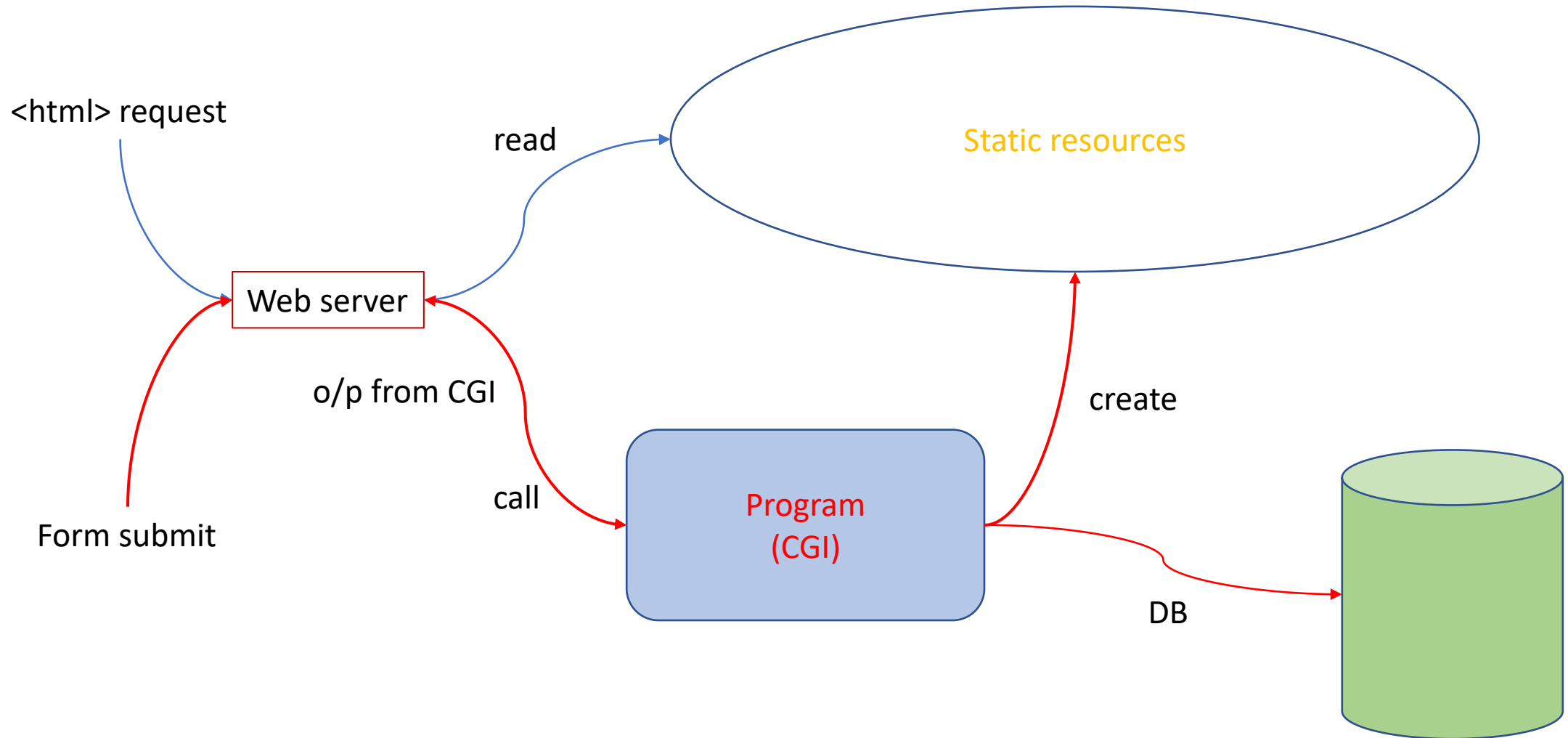
Web in general



Early backend

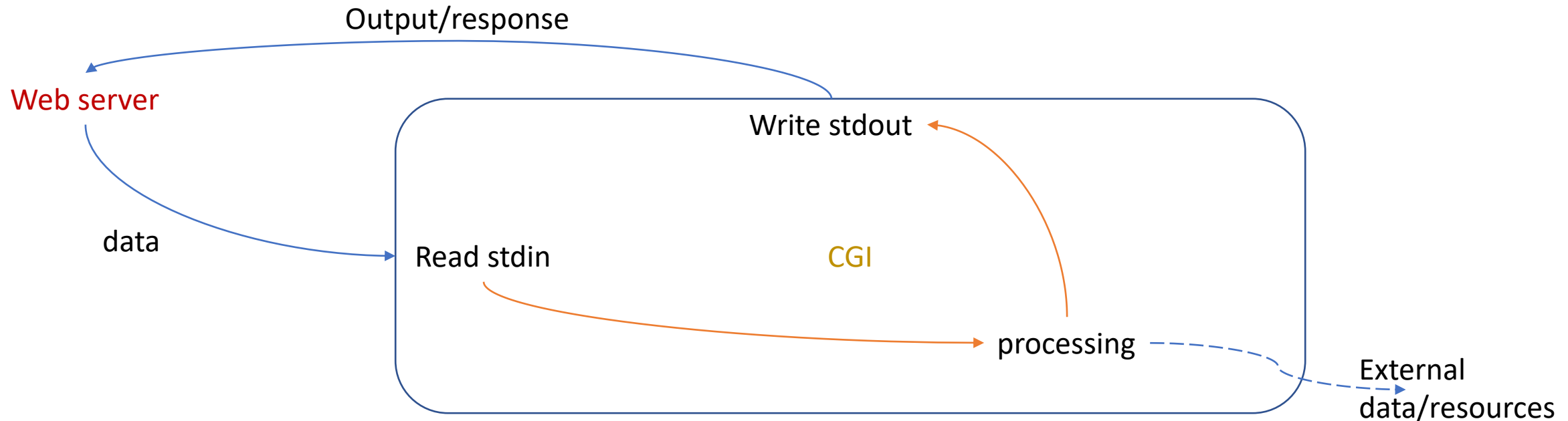


Early backend with data handling

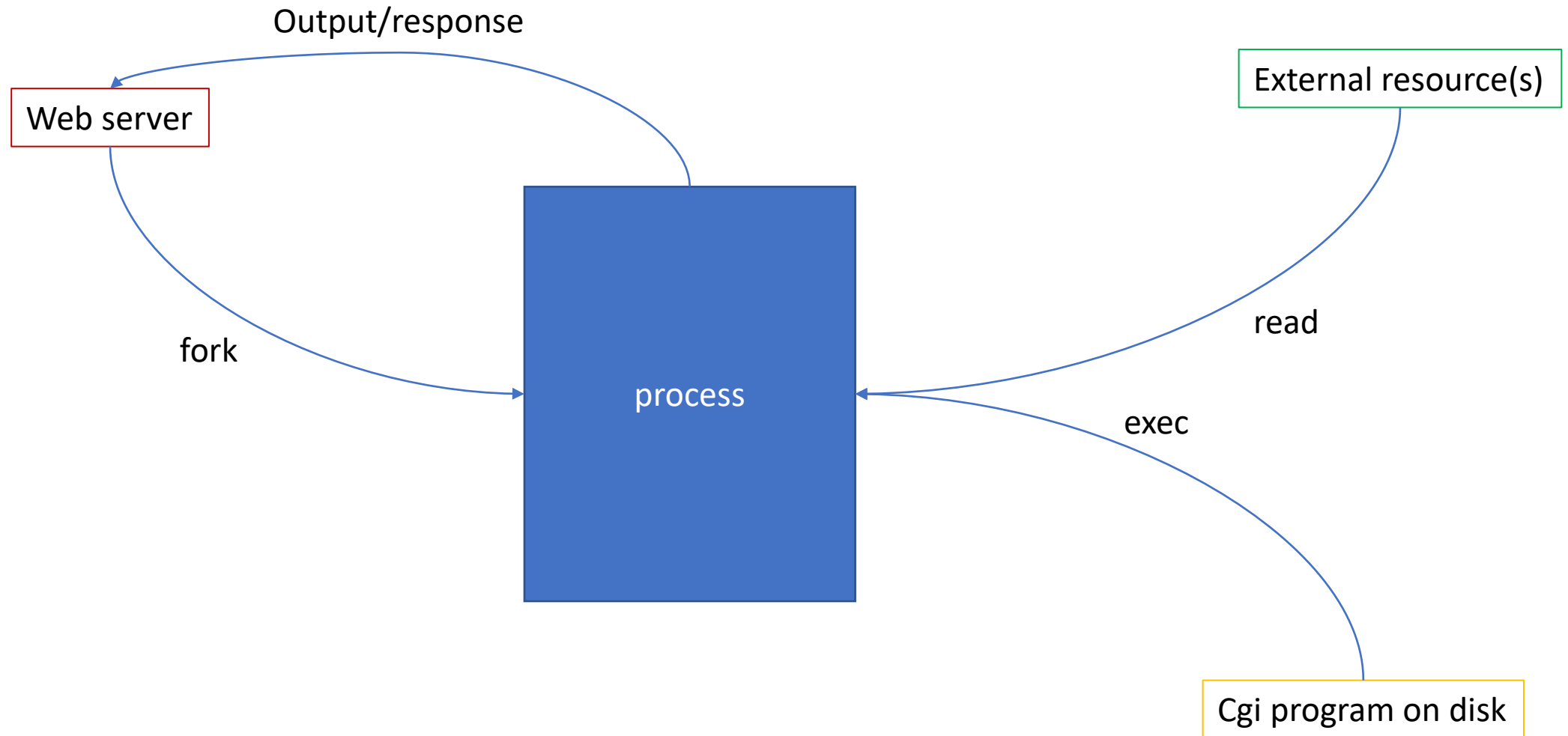


Early CGI

- Develop with language such as C/C++, perl, etc.
- Data transfer from web server to CGI via standard input
- Data transfer from CGI to web server via standard output



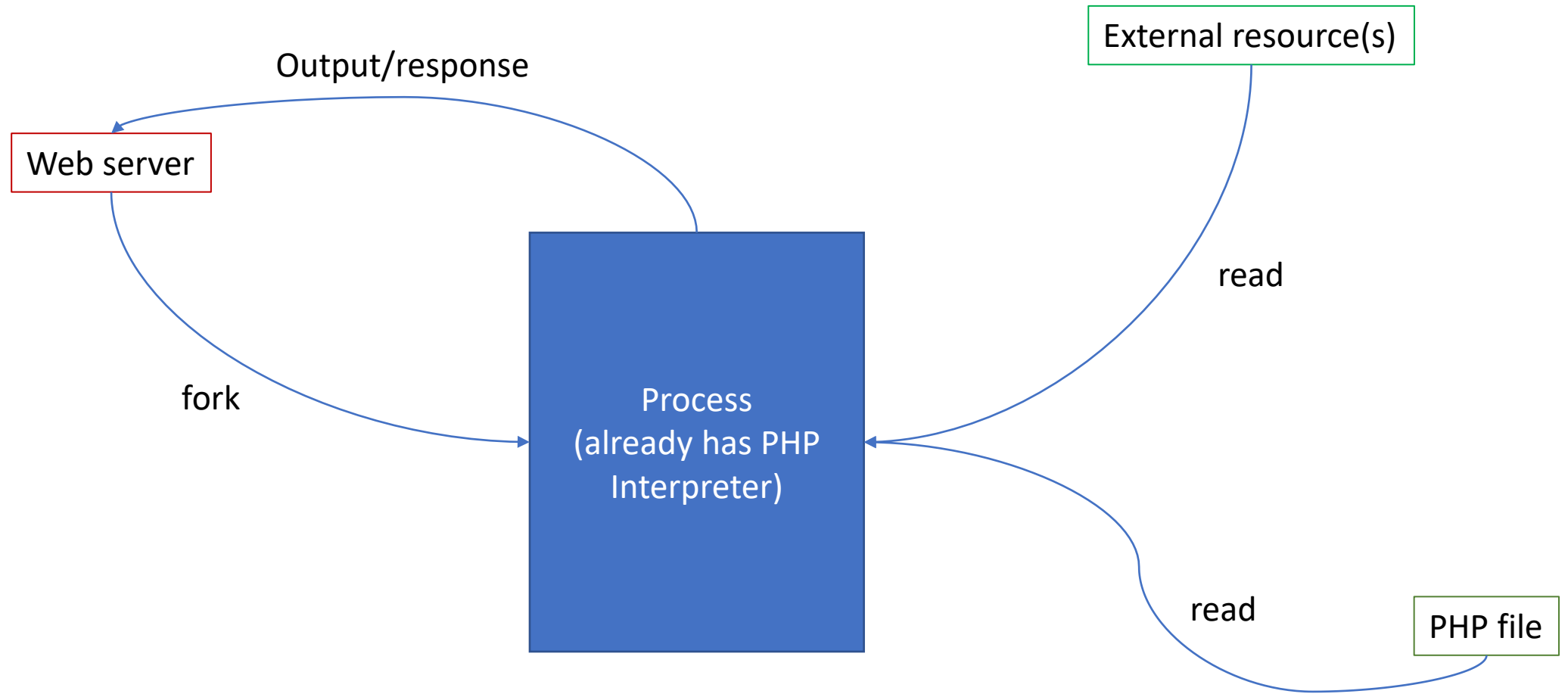
Execution



PHP (ASP)

- Easier data input handling
 - Automatic generate \$_GET or \$_POST
- Easier coding
 - Coding in html file
 - <?php ?>
- Integrates PHP interpreter into web server (option)
 - Speed up execution

Execution



ASP

- Classic ASP
- Active Server Page
- Developed by Microsoft
- Same concepts as PHP
- ASP command written within `<% ... %>`
- Based on VBScript syntax

Example: ASP code

```
<!DOCTYPE html>  
<html>  
<body>
```

```
<p>ASP example: output HTML tag</p>
```

```
<%  
response.write("<h1>Hello World!</h1>")  
%>
```

```
</body>  
</html>
```

Example: what browser receive

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<p>ASP example: output HTML tag</p>
```

```
<h1>Hello World!</h1>
```

```
</body>
```

```
</html>
```

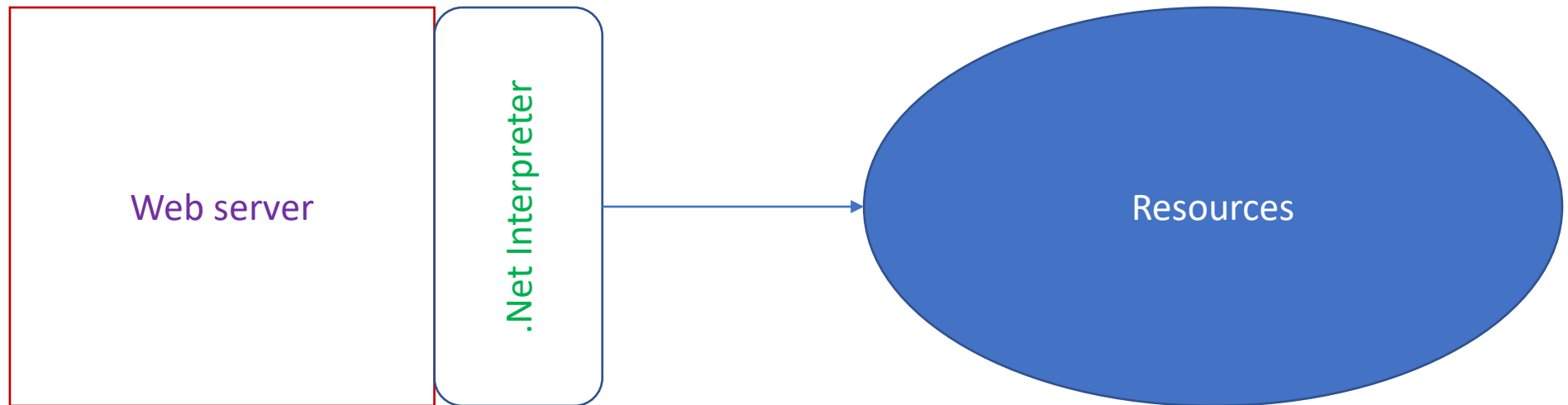
ASP.Net

- Release in 2002 as a successor to ASP
- Normally used c# syntax
- Uses .aspx as an extension
- Tutorial: https://www.w3schools.com/asp/webpages_intro.asp

Example:

```
<!DOCTYPE html>
<html>
<body>
@{
if (IsPost)
{
string companyname = Request["CompanyName"];
string contactname = Request["ContactName"];
<p>You entered: <br>
Company Name: @companyname <br>
Contact Name: @contactname </p>
}
else
{
<form method="post" action="">
Company Name:<br>
<input type="text" name="CompanyName" value=""><br>
Contact Name:<br><br>
<input type="text" name="ContactName" value=""><br><br>
<input type="submit" value="Submit" class="submit">
</form>
}
}
</body>
</html>
```

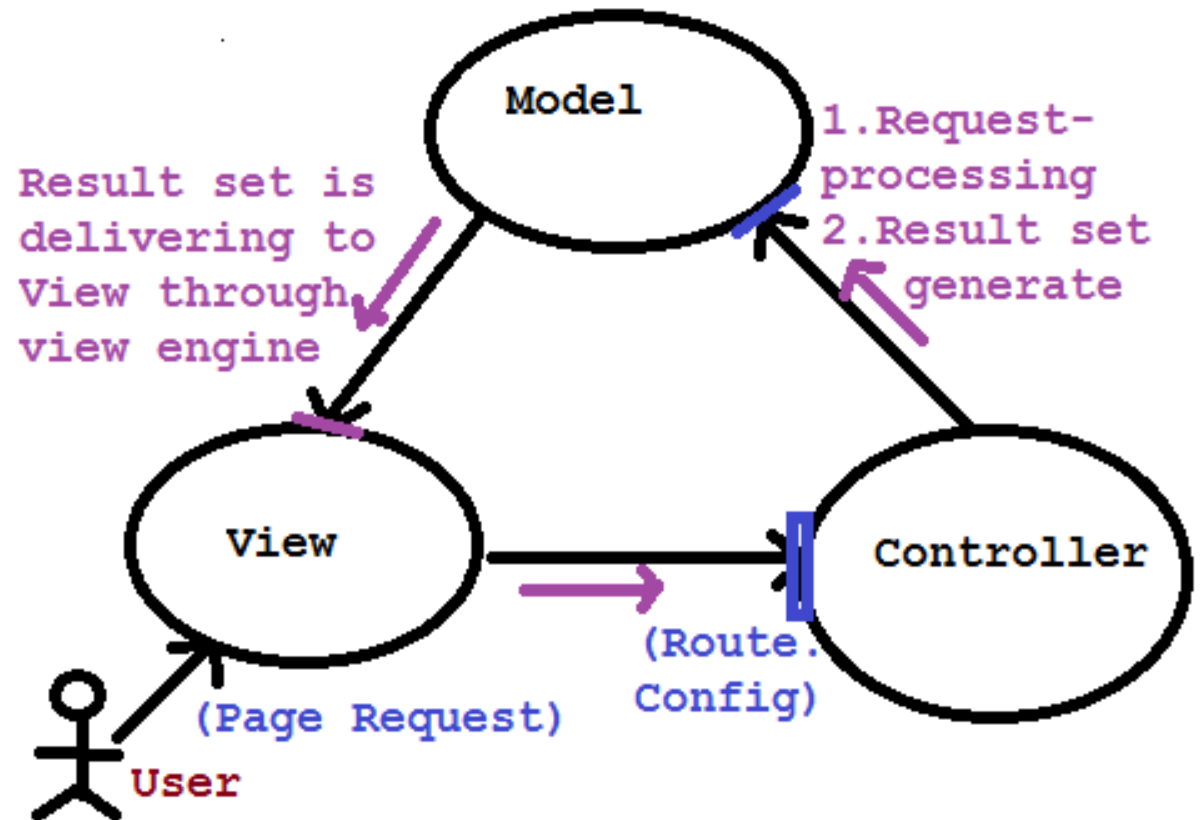
ASP/ASP.Net



ASP.Net MVC → ASP.Net Core MVC (2016)

- MVC = Model View Controller
 - Model represent data
 - View is UI
 - Controller is the request handler

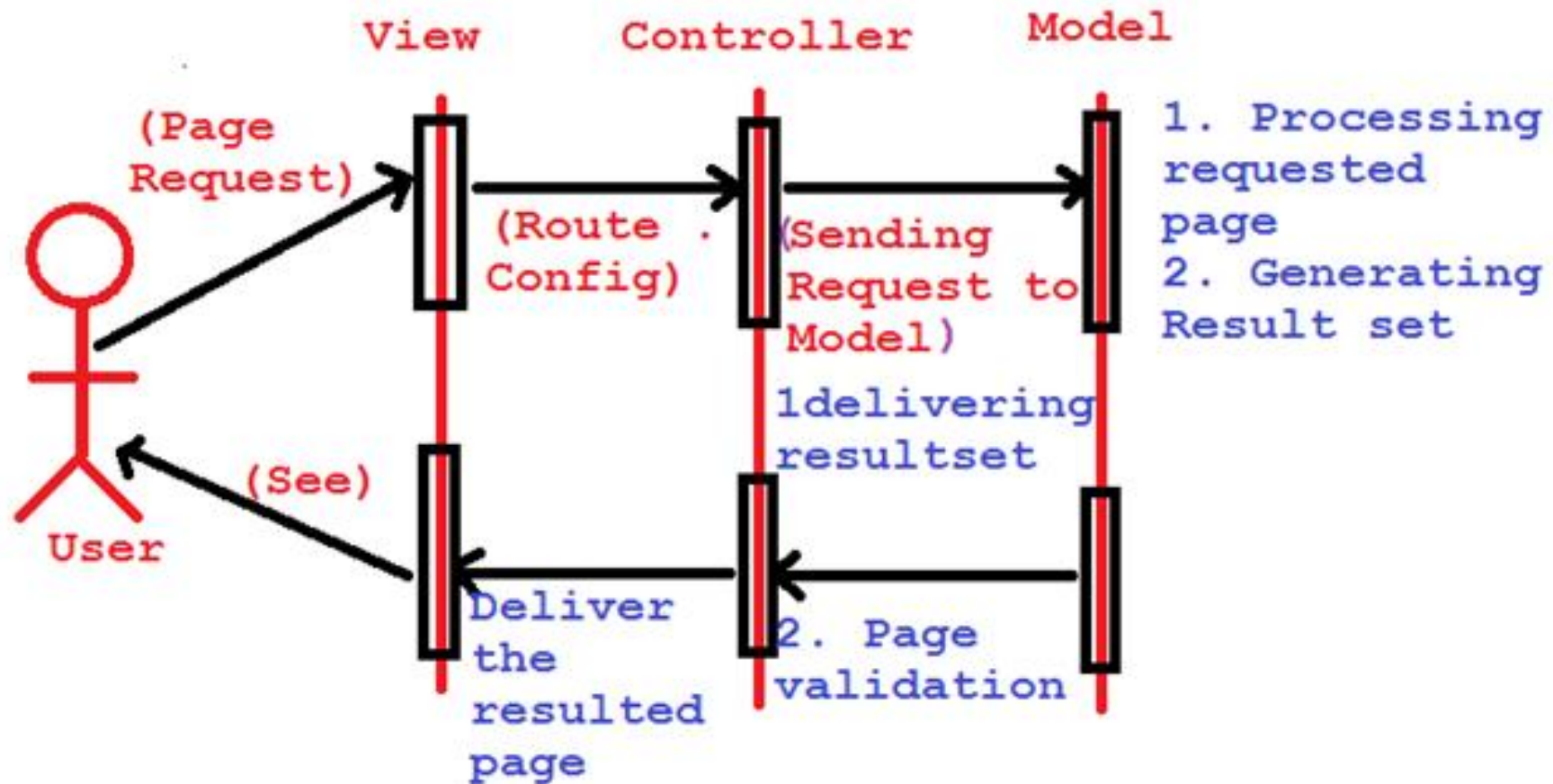
MVC Architecture



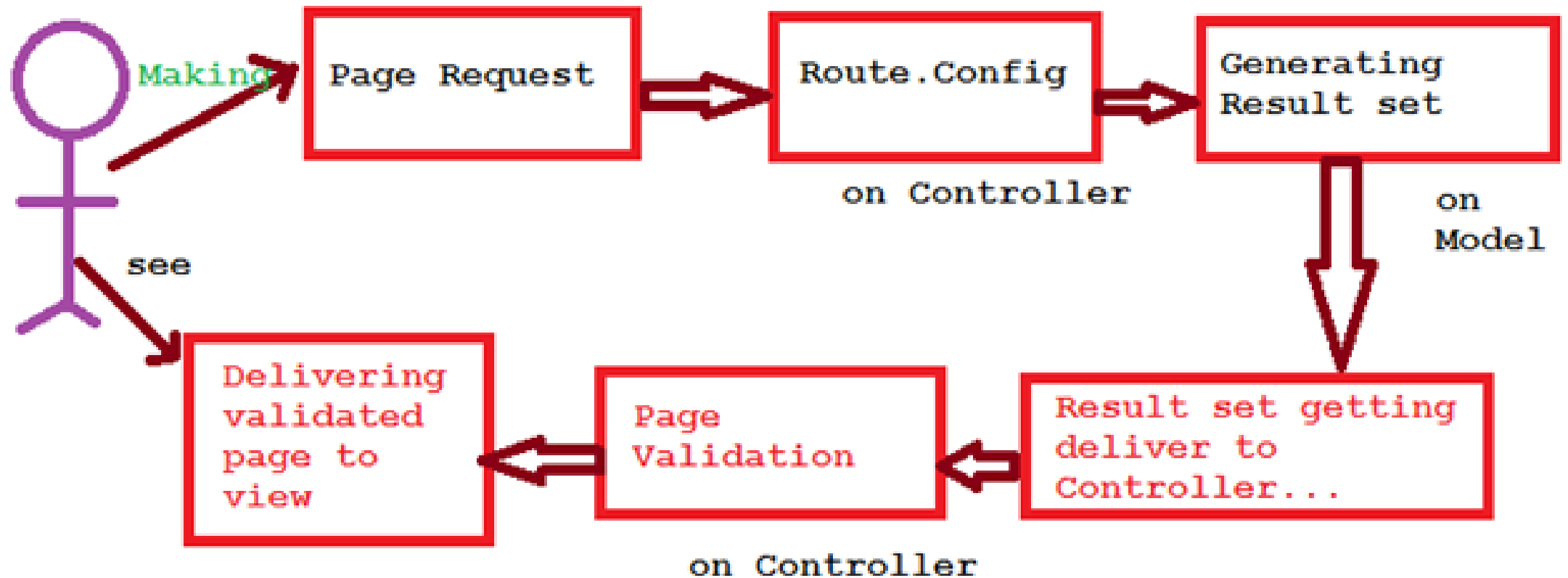
MVC Architecture

By: Sandeep

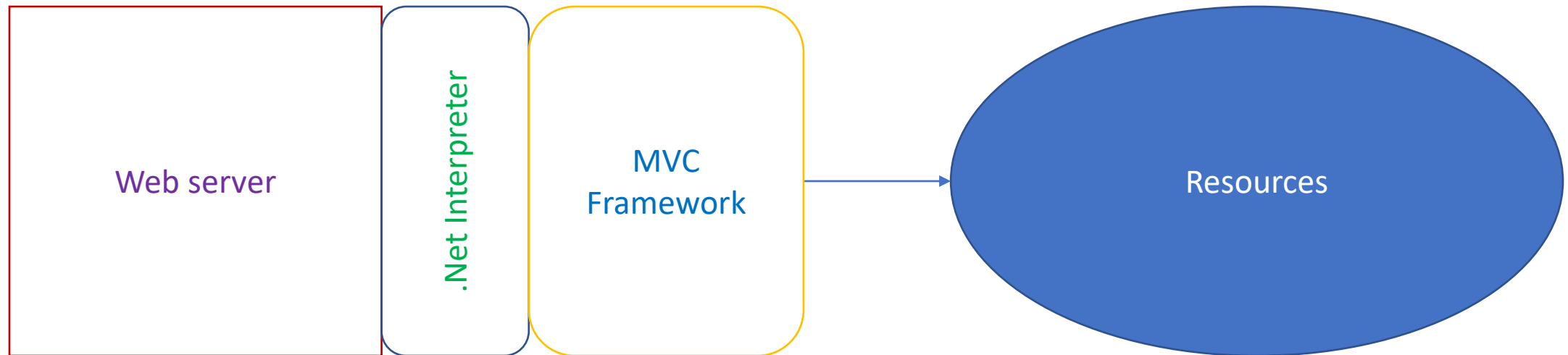
Life cycle of MVC



Live cycle of MVC



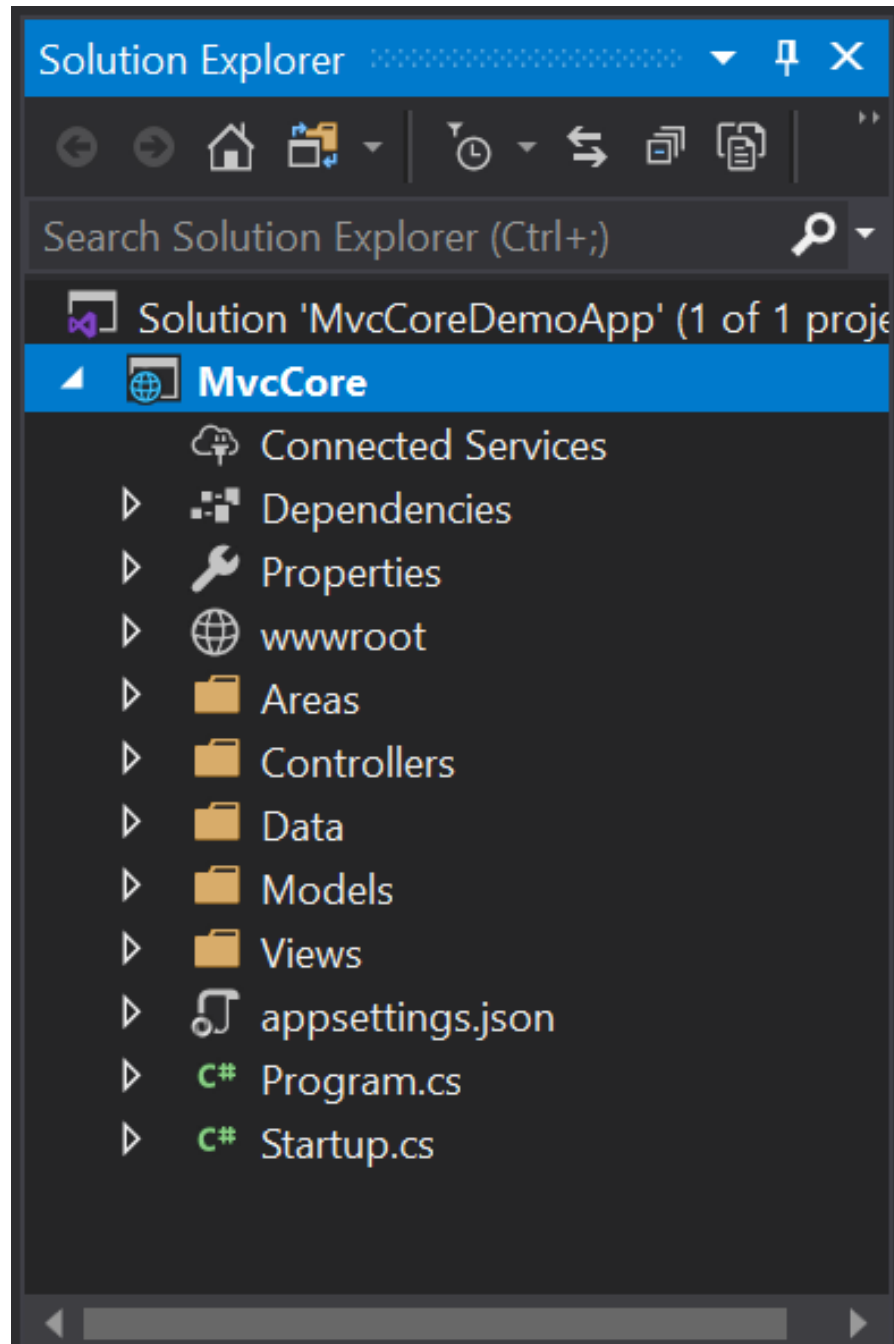
ASP.Net Core MVC



ASP.Net Core MVC Tutorial

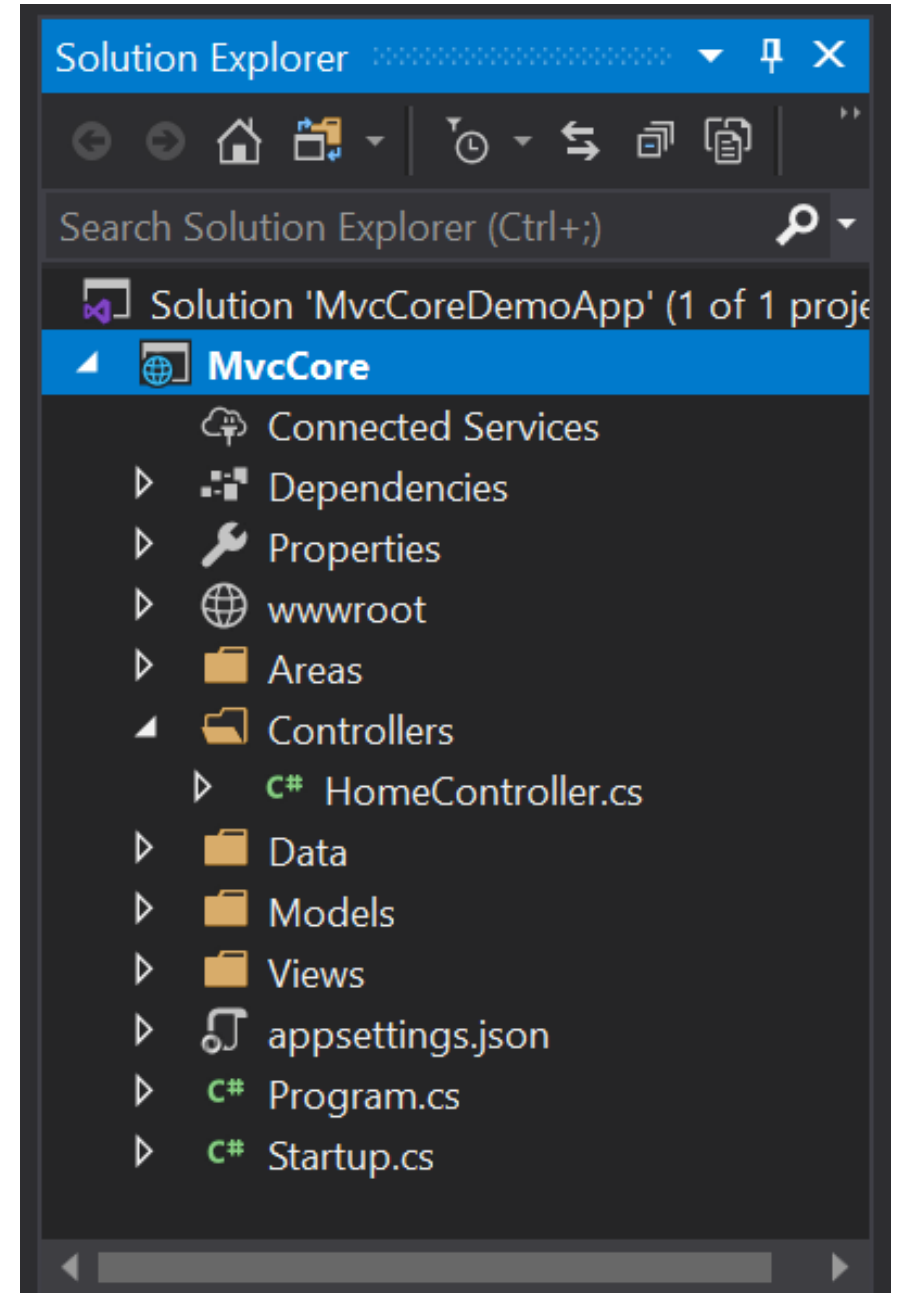
- VDO Clip: ASP.Net Core Introduction:
<https://www.youtube.com/watch?v=1ck9LIBxO14>
- ASP.Net Core Web Tutorial:
 - <https://asp.mvc-tutorial.com/>
 - <https://www.tektutorialshub.com/asp-net-core-tutorial/>

Project Structure



Controller

- Routing to resource
 - From picture there is 1 (virtual) directory in this web site
 - Named: Home
 - `http://host/home`
- In `HomeController.cs` contain methods for rendering pages
 - Each method correspond to 1 page



HomeController.cs

0 references

```
public IActionResult Index()  
{  
    return View();  
}
```

<http://host/Home> or <http://host/Home/index>

0 references

```
public IActionResult Test()  
{  
    return View();  
}
```

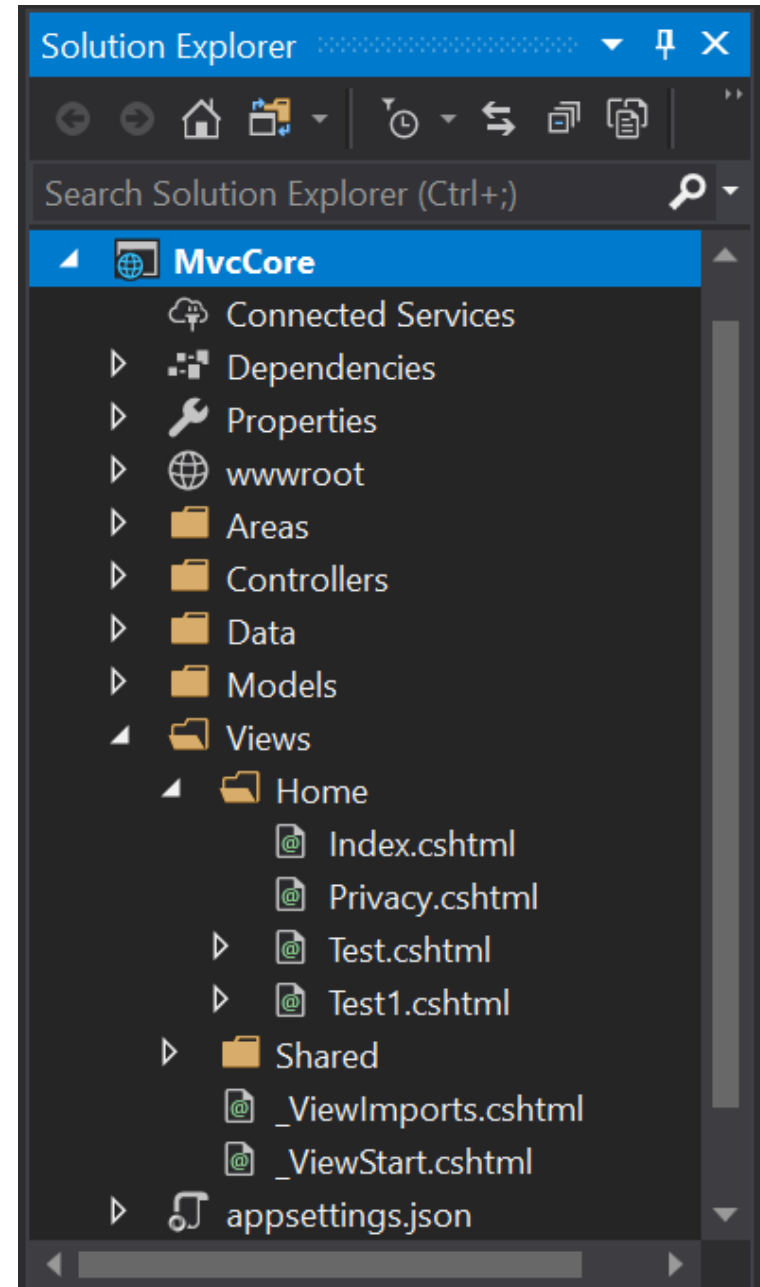
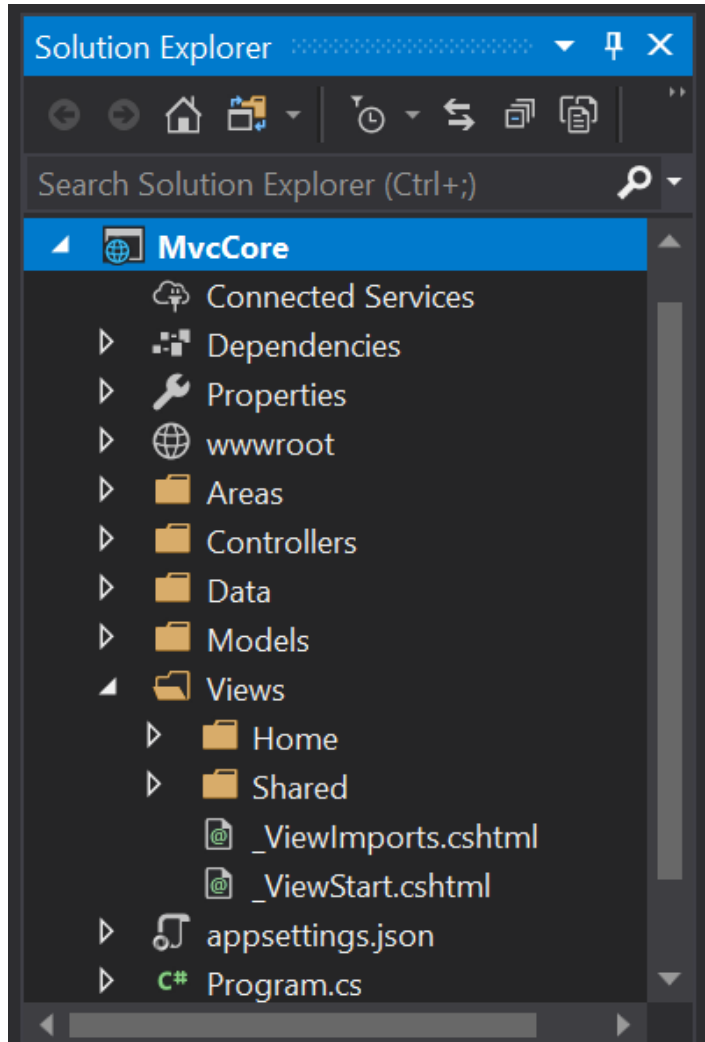
<http://host/Home/Test>

0 references

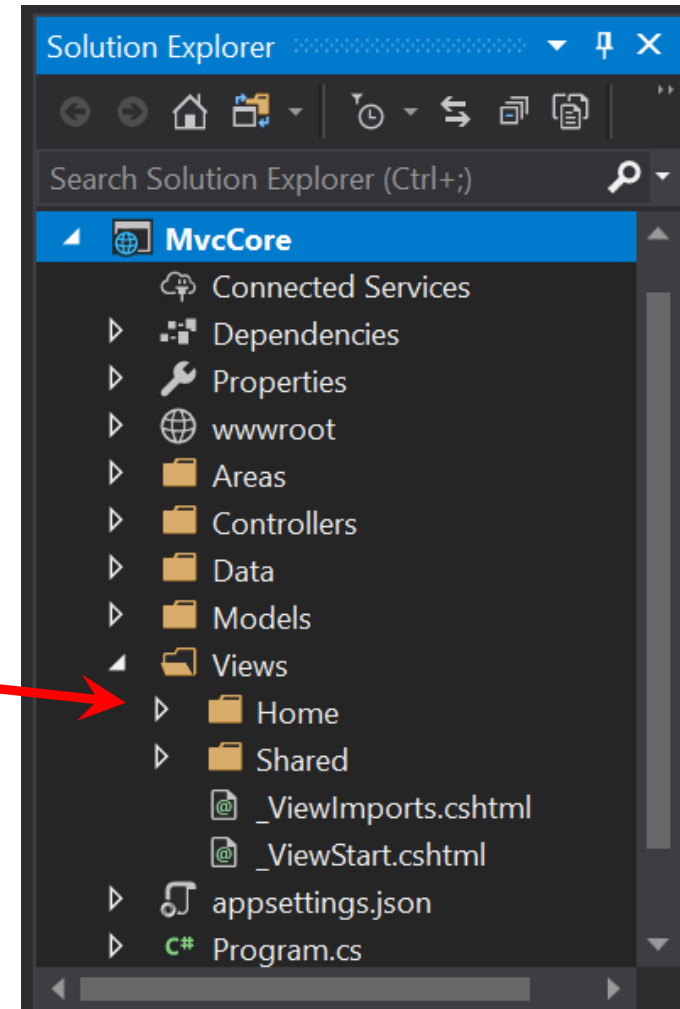
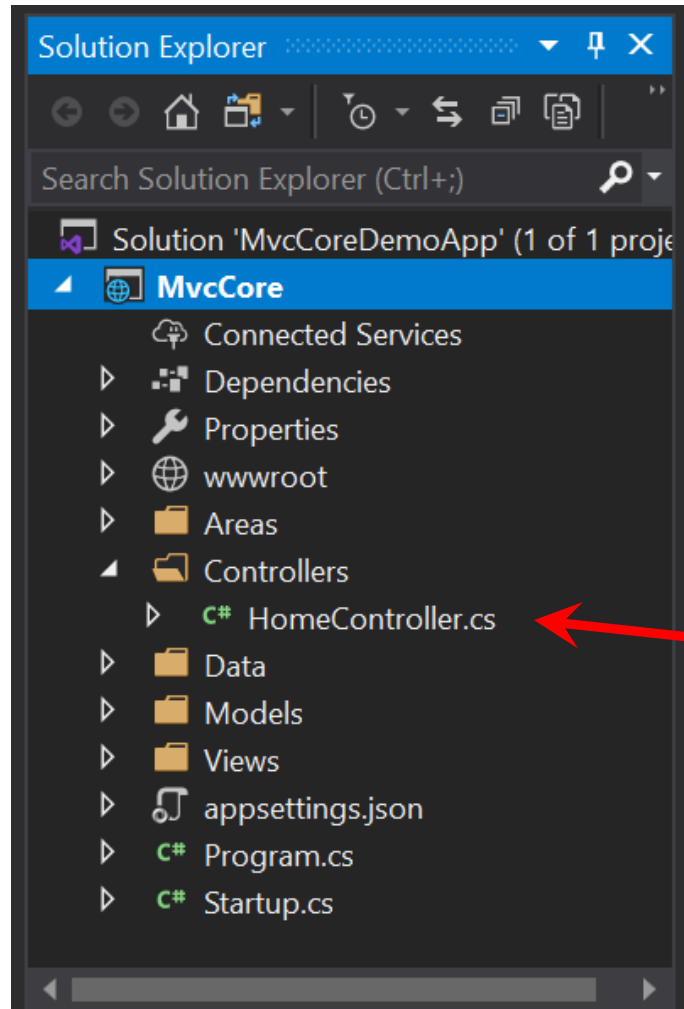
```
public IActionResult Test1()  
{  
    return View();  
}
```

<http://host/Home/Test1>

View



Controller - View



Controller - View

0 references

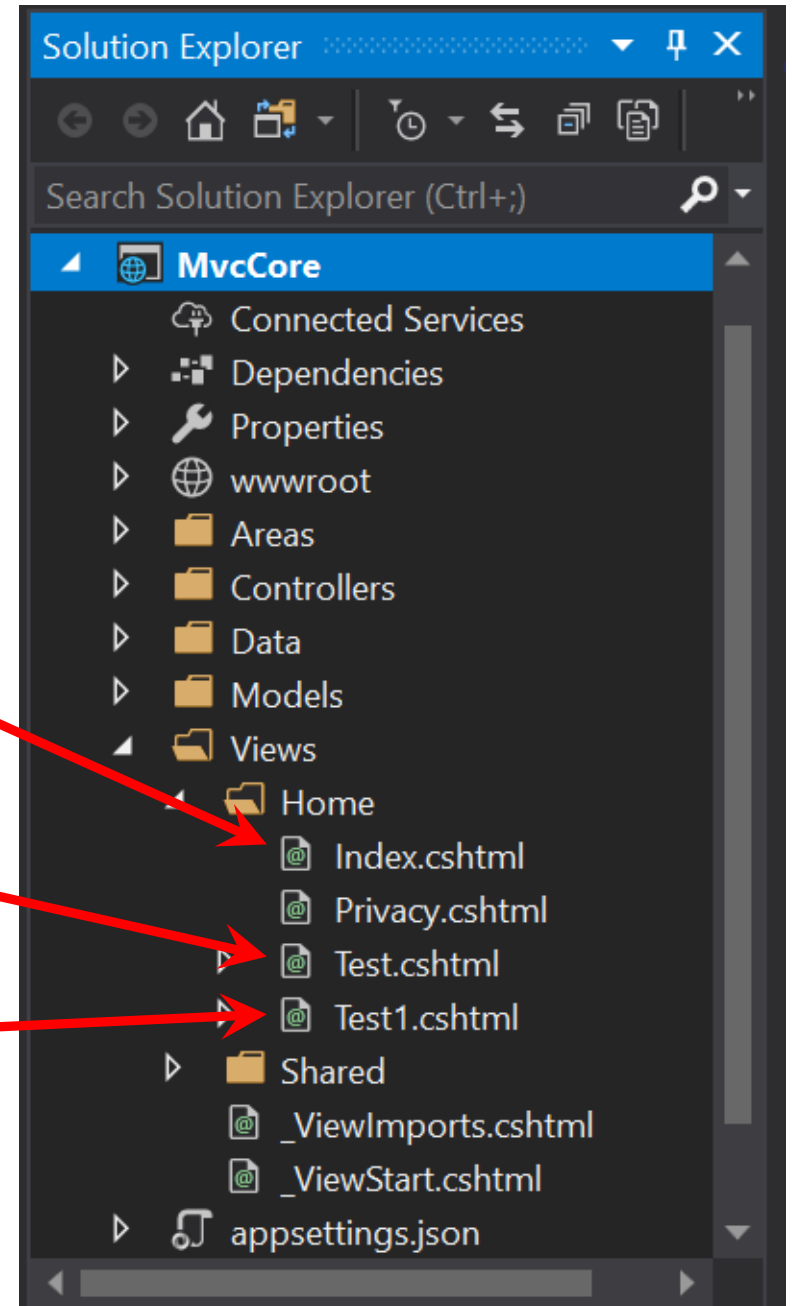
```
public IActionResult Index()  
{  
    return View();  
}
```

0 references

```
public IActionResult Test()  
{  
    return View();  
}
```

0 references

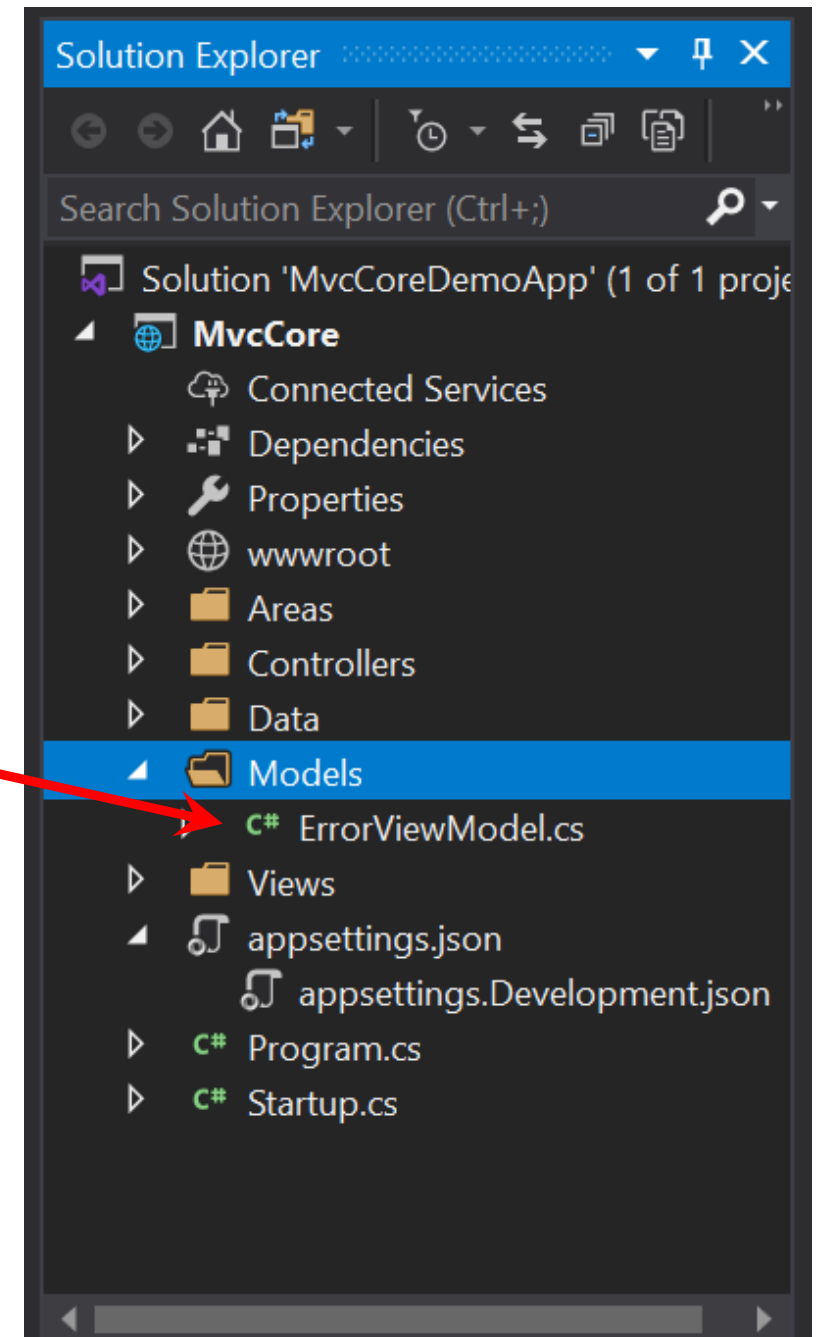
```
public IActionResult Test1()  
{  
    return View();  
}
```



Model

```
namespace MvcCore.Models
{
    4 references
    public class ErrorViewModel
    {
        3 references
        public string RequestId { get; set; }

        1 reference
        public bool ShowRequestId => !string.IsNullOrEmpty(RequestId);
    }
}
```



Controller-Model-View

```
public IActionResult Error()  
{  
    return View(new ErrorViewModel { RequestId = Activity.Current?.Id ?? HttpContext.TraceIdentifier });  
}
```

HomeController.cs (Controller)

```
<h1 class="text-danger">Error.</h1>  
<h2 class="text-danger">An error occurred while processing your request.</h2>  
  
@if (Model.ShowRequestId)  
{  
    <p>  
        <strong>Request ID:</strong> <code>@Model.RequestId</code>  
    </p>  
}
```

Error.cshtml (View)

```
namespace MvcCore.Models  
{  
    4 references  
    public class ErrorViewModel  
    {  
        3 references  
        public string RequestId { get; set; }  
  
        1 reference  
        public bool ShowRequestId => !string.IsNullOrEmpty(RequestId);  
    }  
}
```

ErrorViewModel.cs (Model)