Tutorial 04
Getting started with Blazor

Objectives

The main objective of this tutorial is to get familiar with Blazor for creating client applications. We will see that Visual Studio offers many different templates and options for creating a new Blazor project.

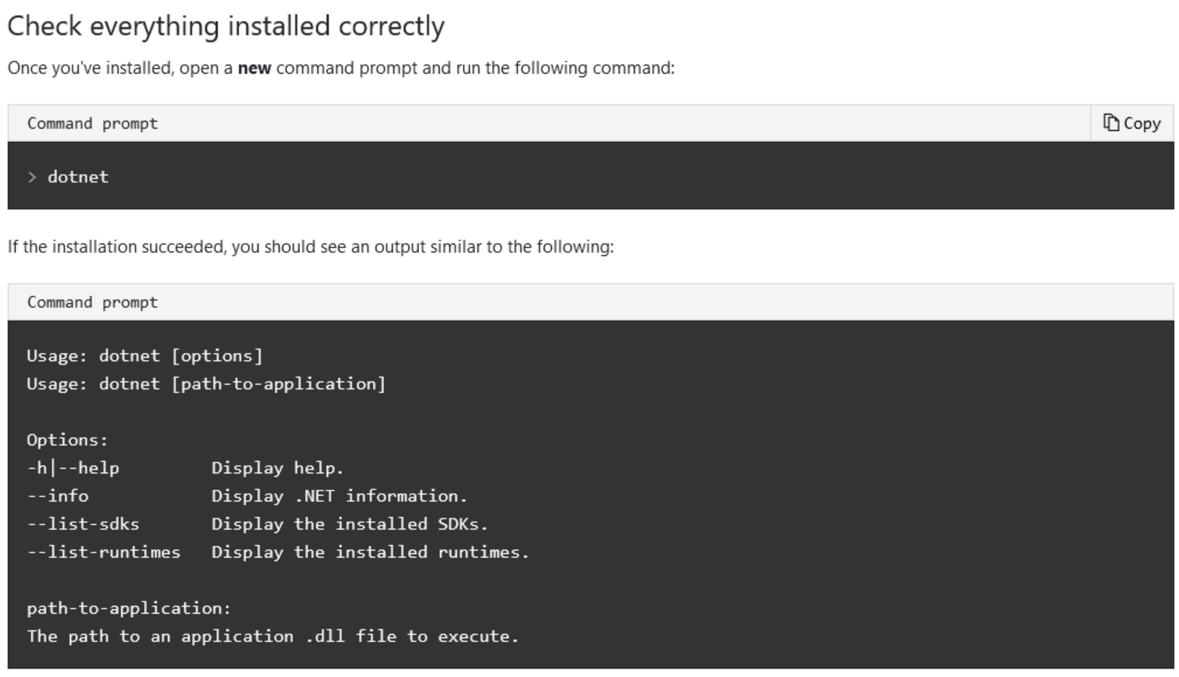
For each option that we select we will explore the structure of the files and projects that Visual Studio generates for us.

The options for creating Blazor projects that we will explore:

1. Blazor Client App (Web Assembly)
2. Blazor Client App (Web Assembly) + ASP.NET Server application
3. Blazor Client App (Web Assembly) + ASP.NET Server application + Identity
4. Blazor Server App

Ensure .NET SDK is installed

https://dotnet.microsoft.com/learn/aspnet/blazor-tutorial/install



Check everything installed correctly

Once you've installed, open a **new** command prompt and run the following command:

Command prompt

```
> dotnet
```

If the installation succeeded, you should see an output similar to the following:

Command prompt

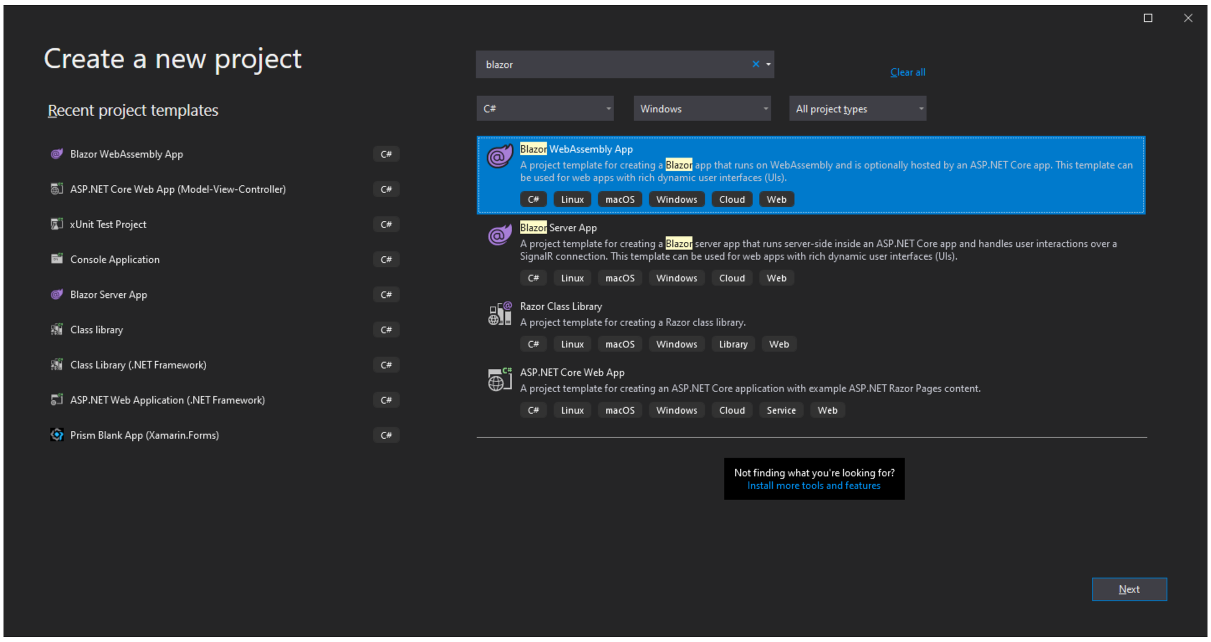
```
Usage: dotnet [options]
Usage: dotnet [path-to-application]

Options:
-h|--help         Display help.
--info            Display .NET information.
--list-sdks       Display the installed SDKs.
--list-runtimes   Display the installed runtimes.

path-to-application:
The path to an application .dll file to execute.
```

1. Blazor Client project (Web Assembly)

- Select Blazor WebAssembly App.



Create a new project

Recent project templates

- Blazor WebAssembly App — C#
- ASP.NET Core Web App (Model-View-Controller) — C#
- xUnit Test Project — C#
- Console Application — C#
- Blazor Server App — C#
- Class library — C#
- Class Library (.NET Framework) — C#
- ASP.NET Web Application (.NET Framework) — C#
- Prism Blank App (Xamarin.Forms) — C#

blazor

Clear all

C# | Windows | All project types

Blazor WebAssembly App
A project template for creating a Blazor app that runs on WebAssembly and is optionally hosted by an ASP.NET Core app. This template can be used for web apps with rich dynamic user interfaces (UIs).
C# Linux macOS Windows Cloud Web

Blazor Server App
A project template for creating a Blazor server app that runs server-side inside an ASP.NET Core app and handles user interactions over a SignalR connection. This template can be used for web apps with rich dynamic user interfaces (UIs).
C# Linux macOS Windows Cloud Web

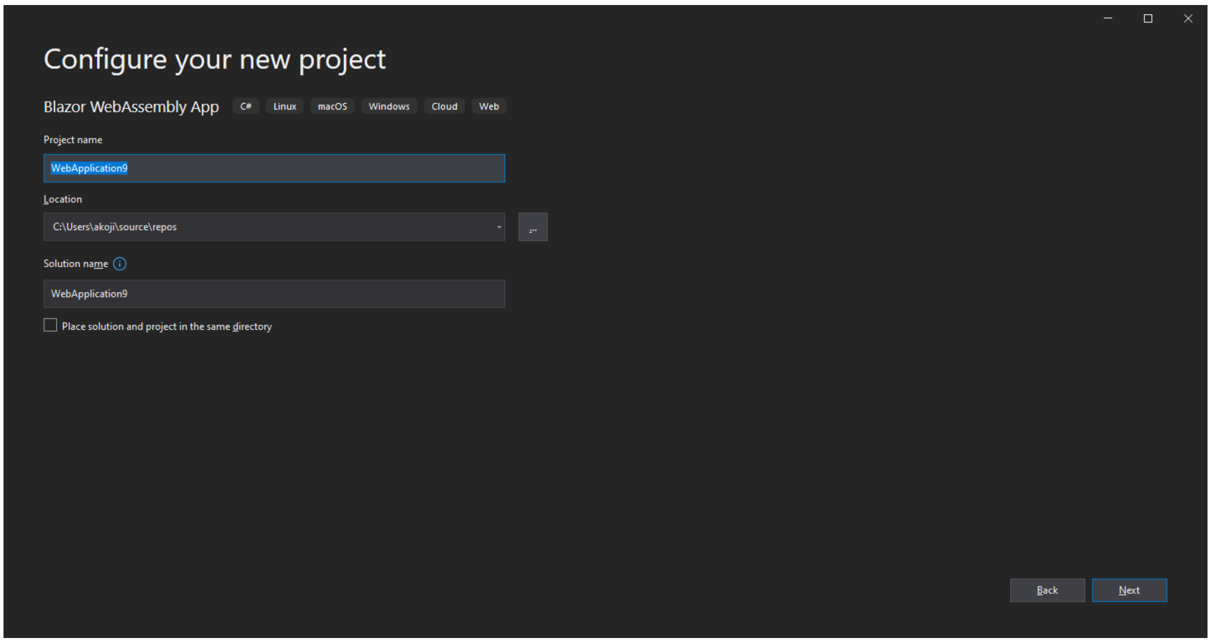
Razor Class Library
A project template for creating a Razor class library.
C# Linux macOS Windows Library Web

ASP.NET Core Web App
A project template for creating an ASP.NET Core application with example ASP.NET Razor Pages content.
C# Linux macOS Windows Cloud Service Web

Not finding what you're looking for?
Install more tools and features

Next

- Give your project a name.



Configure your new project

Blazor WebAssembly App C# Linux macOS Windows Cloud Web

Project name
WebApplication9

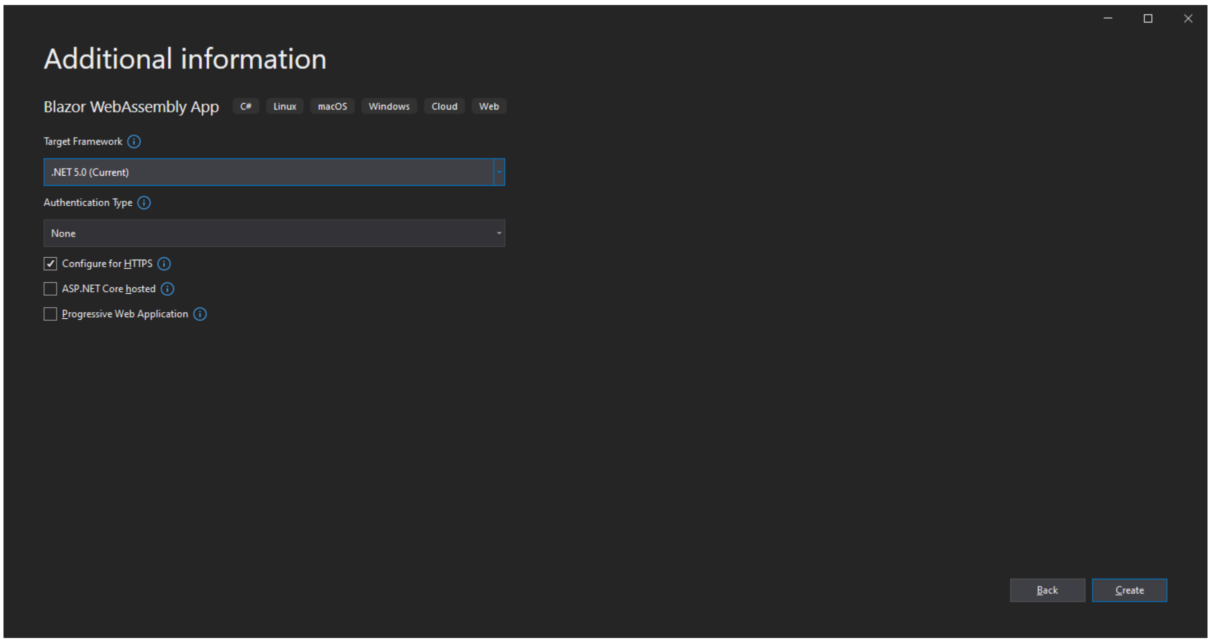
Location
C:\Users\akoji\source\repos

Solution name
WebApplication9

☐ Place solution and project in the same directory

Back Next

- Select .NET 5 framework and ignore other options for now.



Additional information

Blazor WebAssembly App

Target Framework: .NET 5.0 (Current)

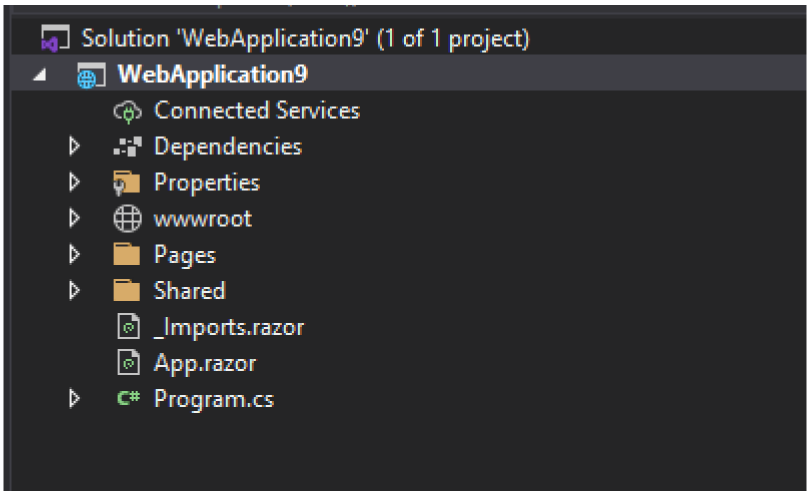
Authentication Type: None

[✓] Configure for HTTPS

[ ] ASP.NET Core hosted

[ ] Progressive Web Application

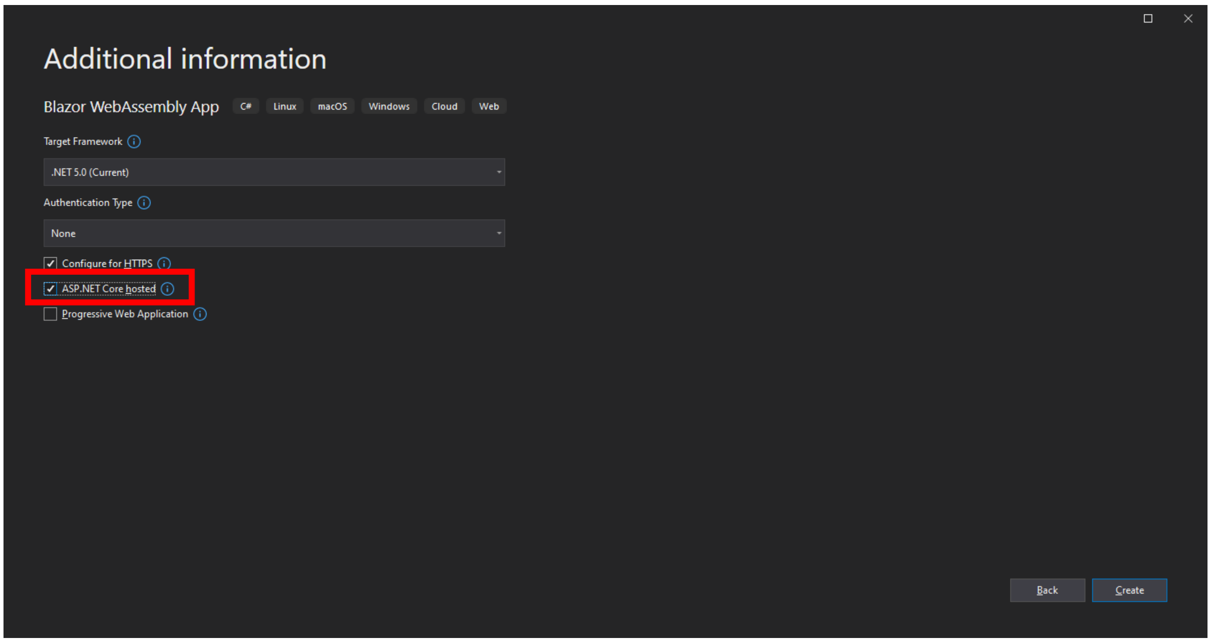
- Once you click on Create, this is what you should see.



Solution 'WebApplication9' (1 of 1 project)
WebApplication9
Connected Services
Dependencies
Properties
wwwroot
Pages
Shared
_Imports.razor
App.razor
Program.cs

2. Blazor Client project (Web Assembly) + ASP.NET Server application

- Repeat the steps from the previous section until you get to this page. **Ensure ASP.NET Core hosted is selected in order to create a server project as well.**



Additional information

Blazor WebAssembly App

Target Framework: .NET 5.0 (Current)

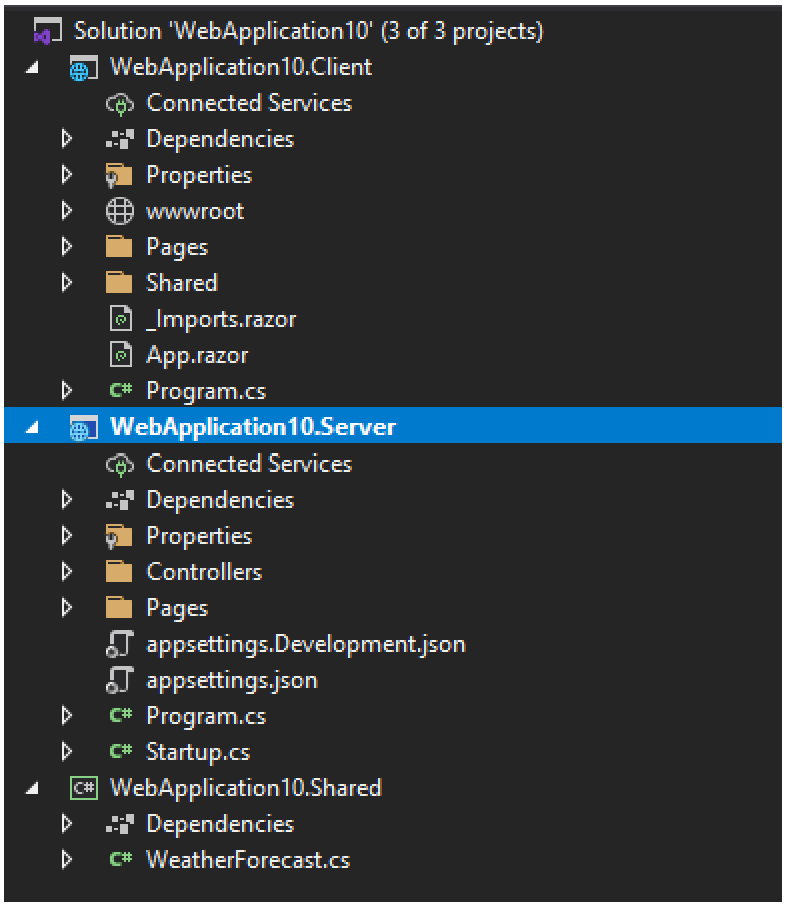
Authentication Type: None

[✓] Configure for HTTPS

[✓] ASP.NET Core hosted

[ ] Progressive Web Application

- This is what the solution in Visual Studio should look like.



```
Solution 'WebApplication10' (3 of 3 projects)
◢ WebApplication10.Client
      Connected Services
   ▷  Dependencies
   ▷  Properties
   ▷  wwwroot
   ▷  Pages
   ▷  Shared
      _Imports.razor
      App.razor
   ▷  Program.cs
◢ WebApplication10.Server
      Connected Services
   ▷  Dependencies
   ▷  Properties
   ▷  Controllers
   ▷  Pages
      appsettings.Development.json
      appsettings.json
   ▷  Program.cs
   ▷  Startup.cs
◢ WebApplication10.Shared
   ▷  Dependencies
   ▷  WeatherForecast.cs
```

Please note, now we have got three projects created.

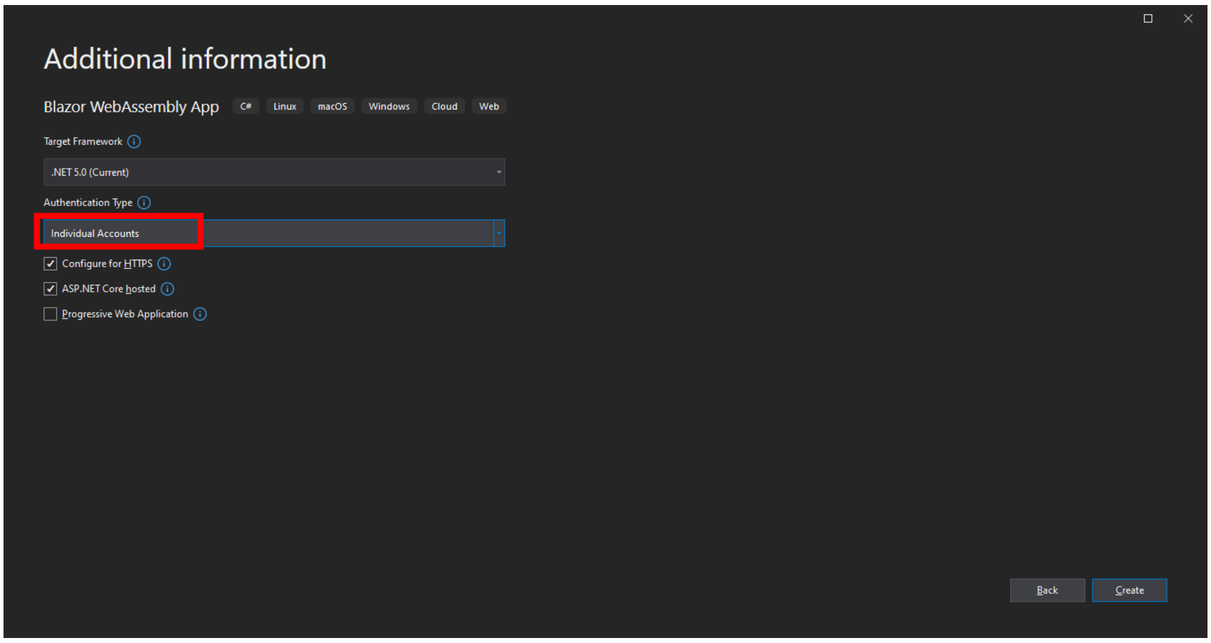
- Client project
- Server project
- Shared project for shared classes

Even if we follow the Option 1 for the Blazor project creation, the server project (Web APIs) can be added separately by going right click on a solution > Add > New project > ASP.NET Core Web API project.

This means we can get from Option 1 to Option 2 but this would require some work from our side. With Option 2 we just tell Visual Studio to set everything up for us.

3. Blazor Client project (Web Assembly) + ASP.NET Server application + Identity Server

- Repeat all the steps from the previous section. When you get to this page, ensure **Authentication Type** is set to **Individual Accounts.**



Additional information

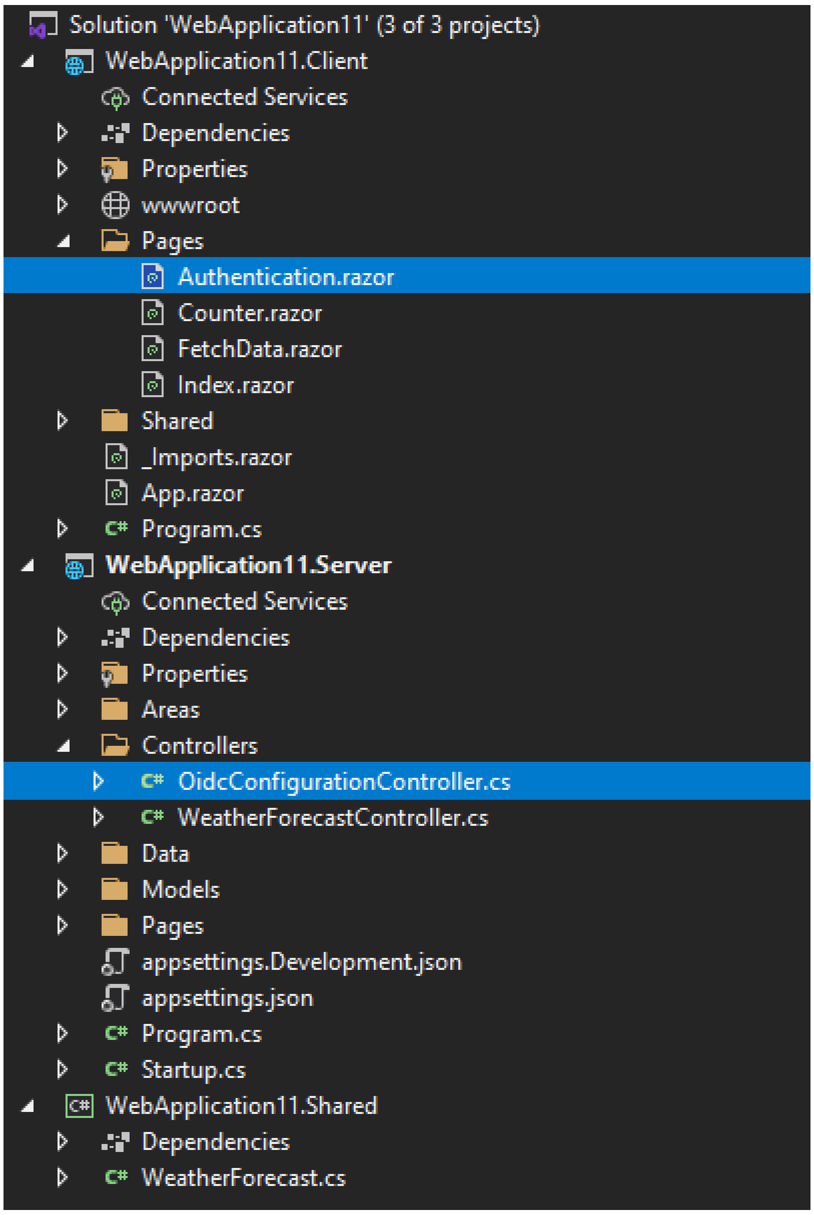
Blazor WebAssembly App

Target Framework: .NET 5.0 (Current)

Authentication Type: Individual Accounts

☑ Configure for HTTPS
☑ ASP.NET Core hosted
☐ Progressive Web Application

- The final project should look like this.



```
Solution 'WebApplication11' (3 of 3 projects)
  WebApplication11.Client
    Connected Services
    Dependencies
    Properties
    wwwroot
    Pages
      Authentication.razor
      Counter.razor
      FetchData.razor
      Index.razor
    Shared
    _Imports.razor
    App.razor
    Program.cs
  WebApplication11.Server
    Connected Services
    Dependencies
    Properties
    Areas
    Controllers
      OidcConfigurationController.cs
      WeatherForecastController.cs
    Data
    Models
    Pages
    appsettings.Development.json
    appsettings.json
    Program.cs
    Startup.cs
  WebApplication11.Shared
    Dependencies
    WeatherForecast.cs
```

What is Identity (server)?

Identity provides us with the logic for:

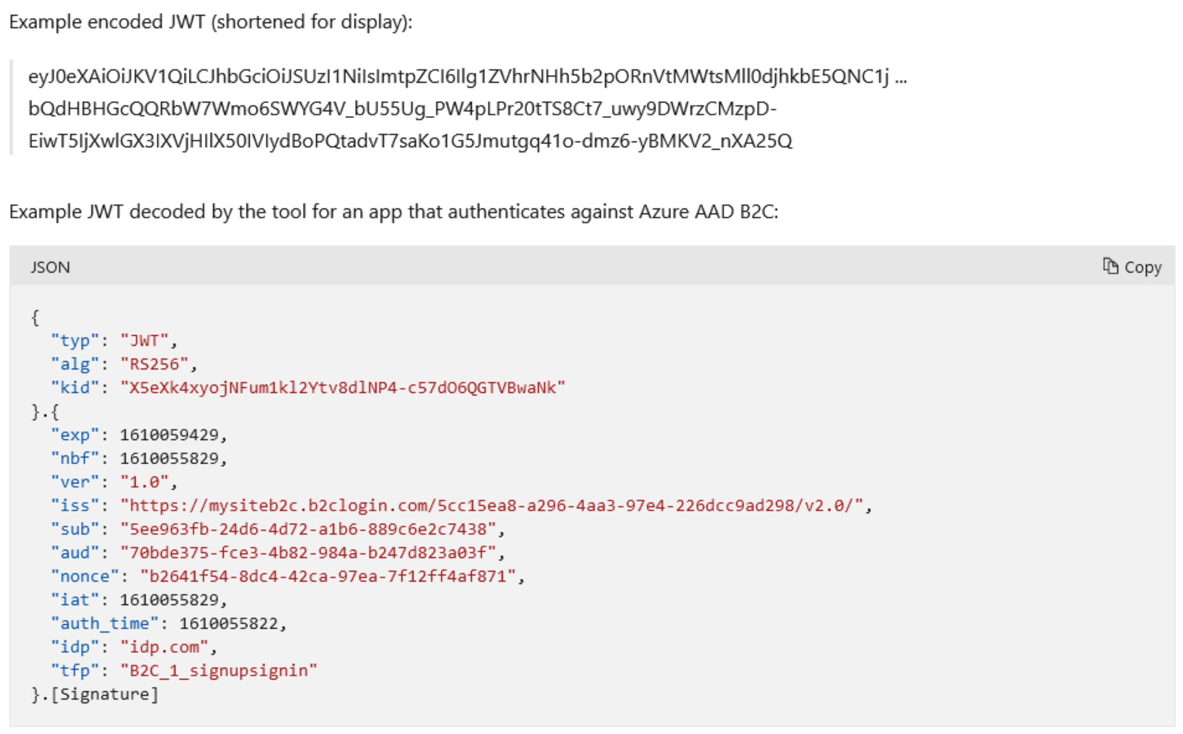
- Login
- Registration
- Authentication

This is done using OpenID Connect. More info on this link:

https://docs.microsoft.com/en-us/aspnet/core/blazor/security/webassembly/hosted-with-identity-server?view=aspnetcore-5.0&tabs=visual-studio

JSON Web Token (JWT)

OpenID Connect generates a token which is called JSON Web Token or just JWT.



Example encoded JWT (shortened for display):

> eyJ0eXAiOiJKV1QiLCJhbGciOiJSUzI1NiIsImtpZCI6Ilg1ZVhrNHh5b2pORnVtMWtsMll0djhkbE5QNC1j ...
> bQdHBHGcQQRbW7Wmo6SWYG4V_bU55Ug_PW4pLPr20tTS8Ct7_uwy9DWrzCMzpD-
> EiwT5IjXwlGX3IXVjHIlX50IVIydBoPQtadvT7saKo1G5Jmutgq41o-dmz6-yBMKV2_nXA25Q

Example JWT decoded by the tool for an app that authenticates against Azure AAD B2C:

JSON

Copy

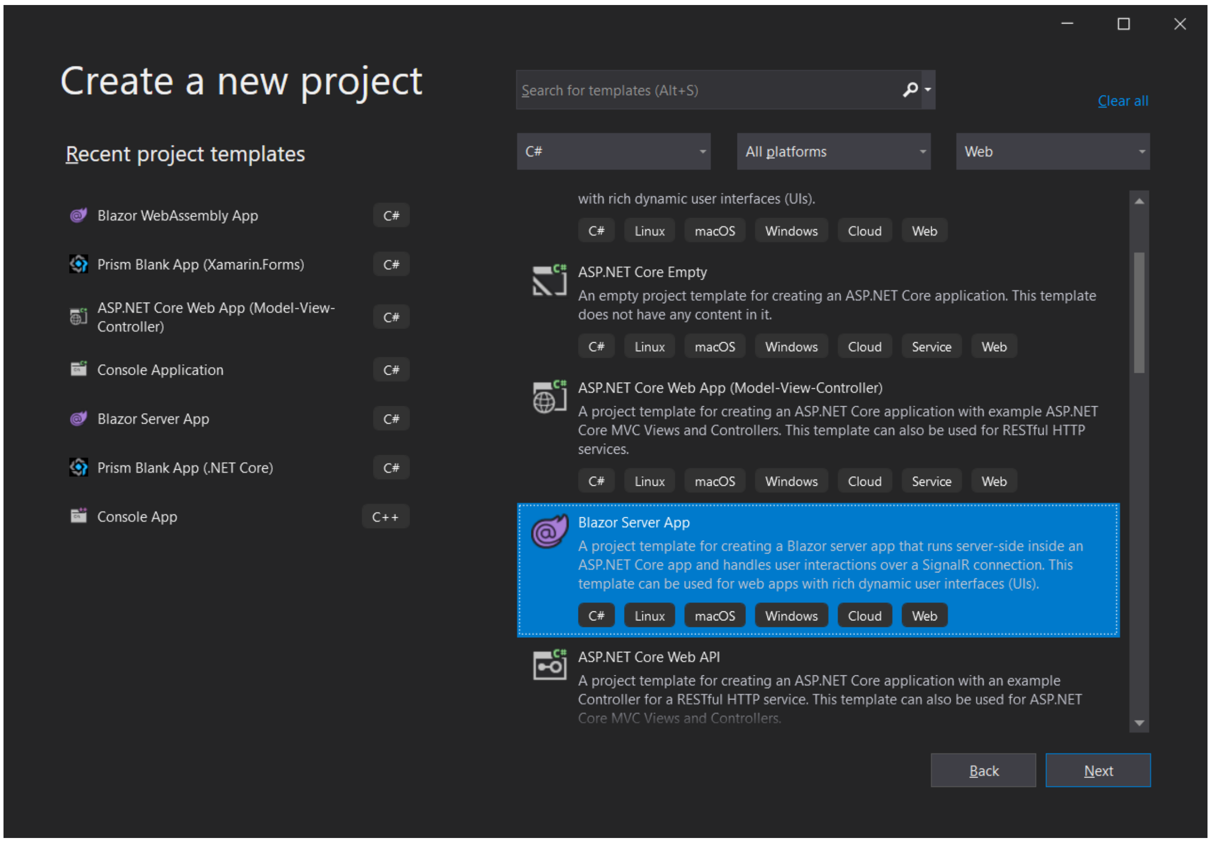
```json
{
  "typ": "JWT",
  "alg": "RS256",
  "kid": "X5eXk4xyojNFum1kl2Ytv8dlNP4-c57dO6QGTVBwaNk"
}.{
  "exp": 1610059429,
  "nbf": 1610055829,
  "ver": "1.0",
  "iss": "https://mysiteb2c.b2clogin.com/5cc15ea8-a296-4aa3-97e4-226dcc9ad298/v2.0/",
  "sub": "5ee963fb-24d6-4d72-a1b6-889c6e2c7438",
  "aud": "70bde375-fce3-4b82-984a-b247d823a03f",
  "nonce": "b2641f54-8dc4-42ca-97ea-7f12ff4af871",
  "iat": 1610055829,
  "auth_time": 1610055822,
  "idp": "idp.com",
  "tfp": "B2C_1_signupsignin"
}.[Signature]
```

4. Blazor Server App

This option for creating a Blazor project is completely different from the first three because we will not create a client application anymore, instead we will create an application that encapsulates both client and server. This application uses SignalR and web sockets to establish a communication between the client and server.

In the upcoming tutorials we will exclusively focus on building Blazor client applications but it is important to understand that this option also exists.

- Select Blazor Server App.



Create a new project

Recent project templates

- Blazor WebAssembly App — C#
- Prism Blank App (Xamarin.Forms) — C#
- ASP.NET Core Web App (Model-View-Controller) — C#
- Console Application — C#
- Blazor Server App — C#
- Prism Blank App (.NET Core) — C#
- Console App — C++

Search for templates (Alt+S)

Clear all

C# | All platforms | Web

with rich dynamic user interfaces (UIs).
C# Linux macOS Windows Cloud Web

ASP.NET Core Empty
An empty project template for creating an ASP.NET Core application. This template does not have any content in it.
C# Linux macOS Windows Cloud Service Web

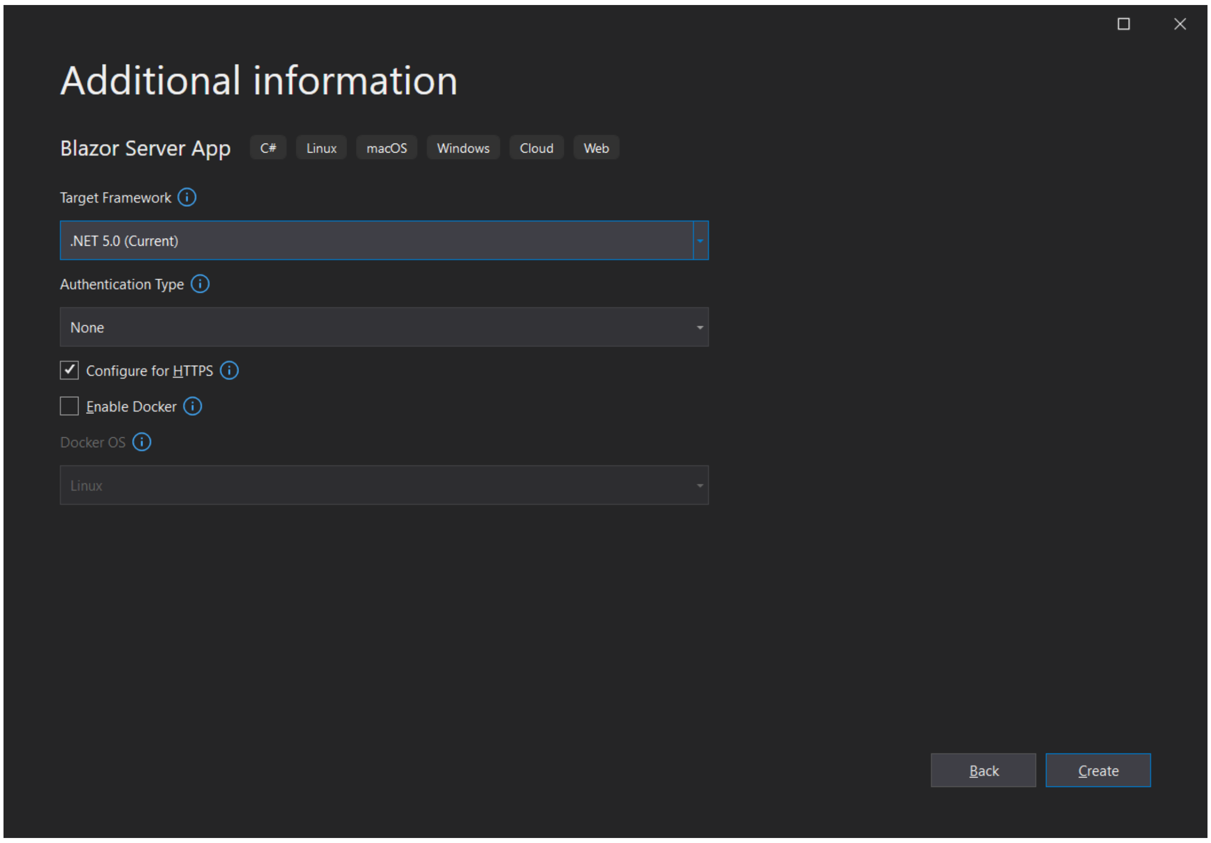
ASP.NET Core Web App (Model-View-Controller)
A project template for creating an ASP.NET Core application with example ASP.NET Core MVC Views and Controllers. This template can also be used for RESTful HTTP services.
C# Linux macOS Windows Cloud Service Web

Blazor Server App
A project template for creating a Blazor server app that runs server-side inside an ASP.NET Core app and handles user interactions over a SignalR connection. This template can be used for web apps with rich dynamic user interfaces (UIs).
C# Linux macOS Windows Cloud Web

ASP.NET Core Web API
A project template for creating an ASP.NET Core application with an example Controller for a RESTful HTTP service. This template can also be used for ASP.NET Core MVC Views and Controllers.

Back | Next

- Next step looks very similar like in the client app. We can select the authentication type or leave the default option.



Additional information

Blazor Server App — C# Linux macOS Windows Cloud Web

Target Framework: .NET 5.0 (Current)

Authentication Type: None

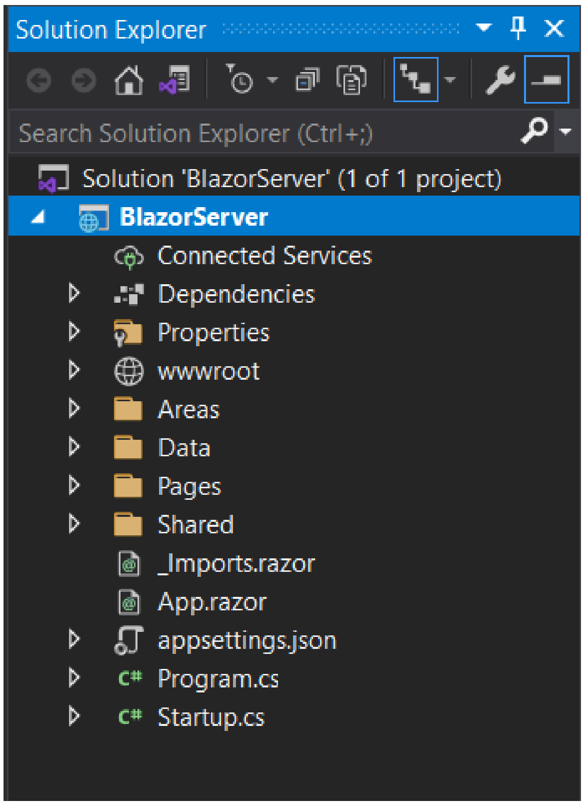
[✓] Configure for HTTPS

[ ] Enable Docker

Docker OS: Linux

Back | Create

- This is what the final solution should look like.



```
Solution 'BlazorServer' (1 of 1 project)
◢ BlazorServer
      Connected Services
   ▷  Dependencies
   ▷  Properties
   ▷  wwwroot
   ▷  Areas
   ▷  Data
   ▷  Pages
   ▷  Shared
      _Imports.razor
      App.razor
   ▷  appsettings.json
   ▷  Program.cs
   ▷  Startup.cs
```

Learn more

- https://docs.microsoft.com/en-us/aspnet/core/blazor/?view=aspnetcore-5.0
- https://docs.microsoft.com/en-us/aspnet/core/security/authentication/scaffold-identity?view=aspnetcore-5.0&tabs=visual-studio#standalone-or-hosted-blazor-webassembly-apps
- https://docs.microsoft.com/en-us/aspnet/core/blazor/security/?view=aspnetcore-5.0