Visual Studio LIVE! | Las Vegas

EXPERT SOLUTIONS FOR ENTERPRISE DEVELOPERS

W15 - Creating Business Applications using Blazor

Michael Washington
Programmer
BlazorHelpWebsite.com

Level: Intermediate

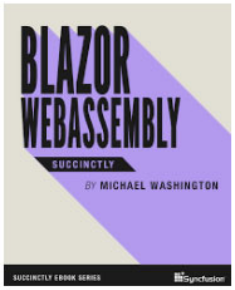
#VSLIVE

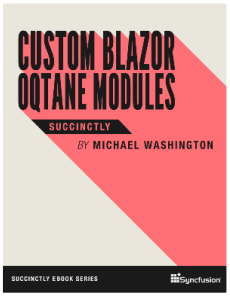
Your Code Powers the World.
Our Training Powers You.

About Michael Washington

Microsoft Most Valuable Professional Award

Blazor Books By Michael Washington

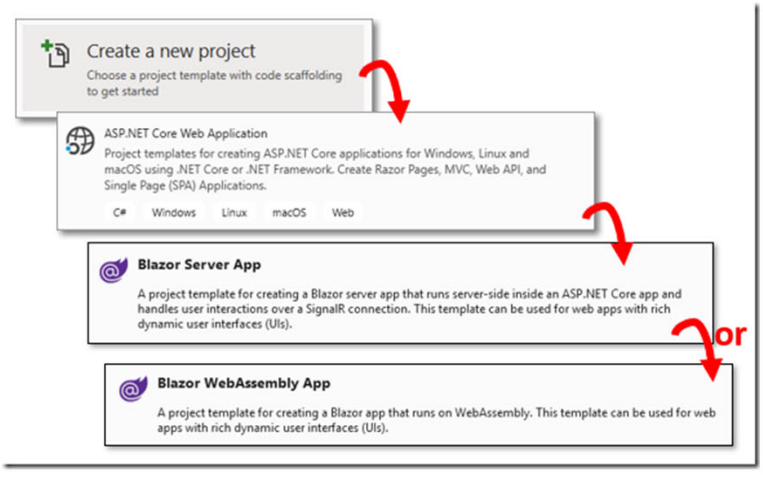




Agenda

- Blazor basics
- Creating A Step-By-Step End-To-End Database Server-Side Blazor Application
- Creating A Step-By-Step End-To-End Database Client-Side (WebAssembly) Blazor Application
- Creating A Blazor Server Azure B2C App

What is Blazor?



WebAssembly

The client-s

- Has n
- Has a
- Fully l
- Offloa
- Suppo

There are d

- Restri
- Requi
- Has a
- Has le

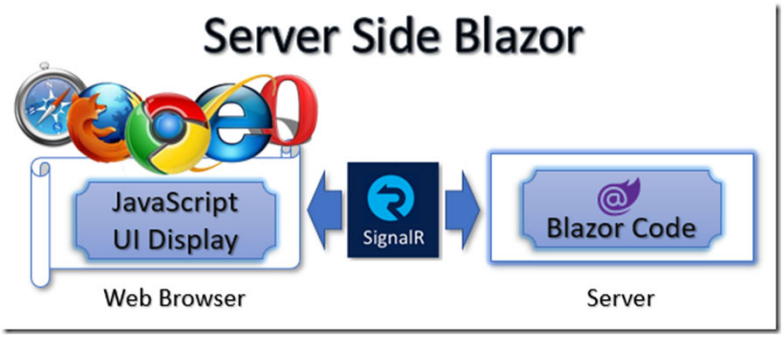
suppo

dard

ion

The server-side hosting model offers several benefits:

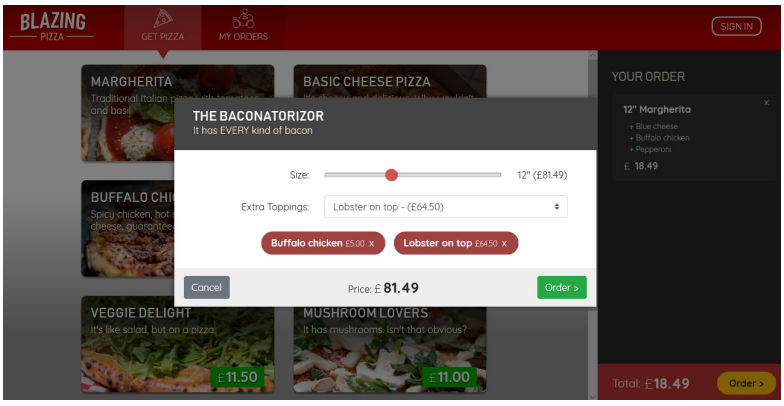
Server Side Blazor



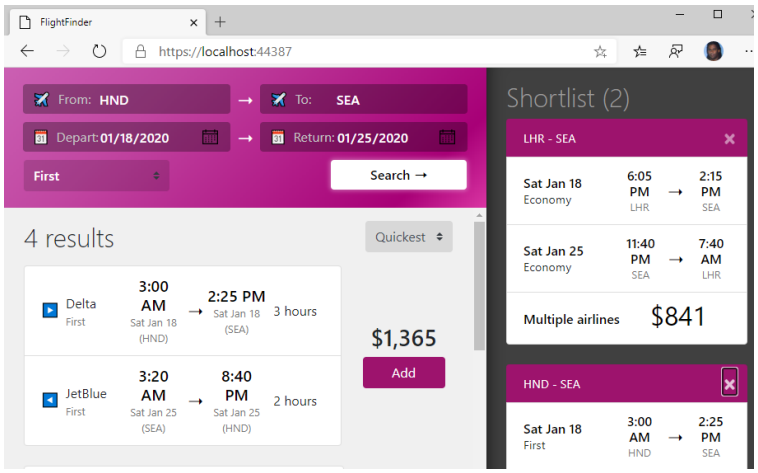
- Requires an ASP.NET Core server to serve the app. Deployment without a server (for example, from a CDN) isn't possible.

Sample Applications

Blazing Pizza

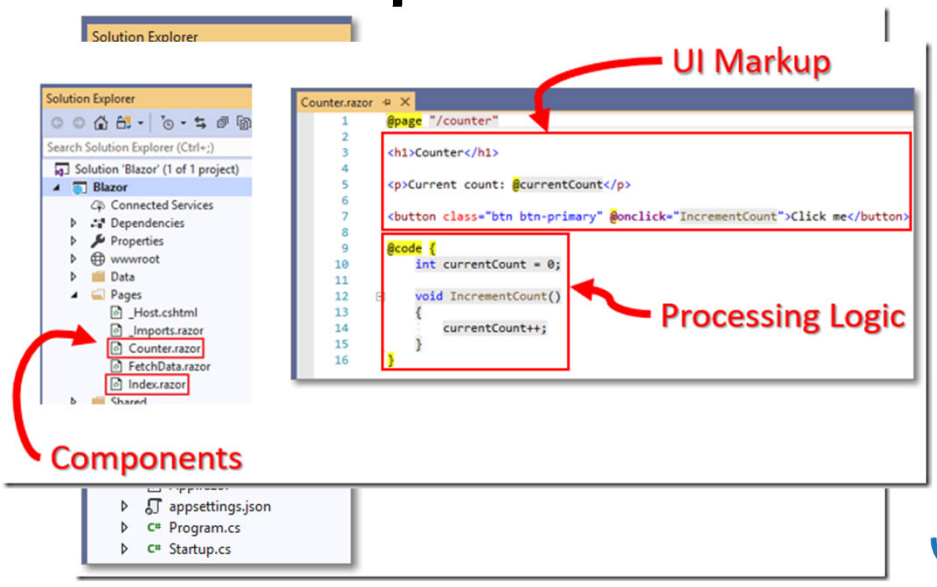


Flight Finder

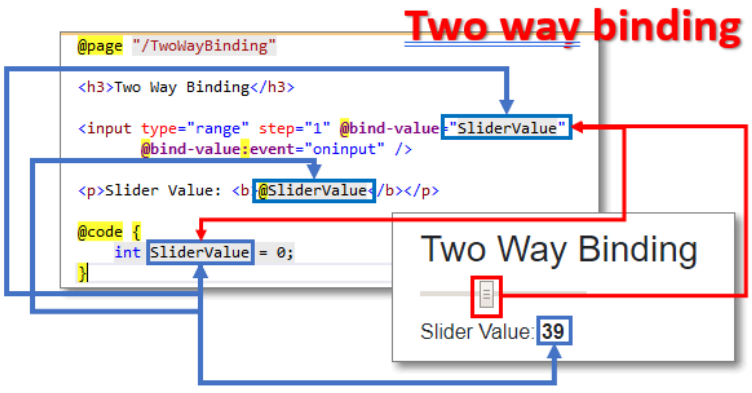


Features

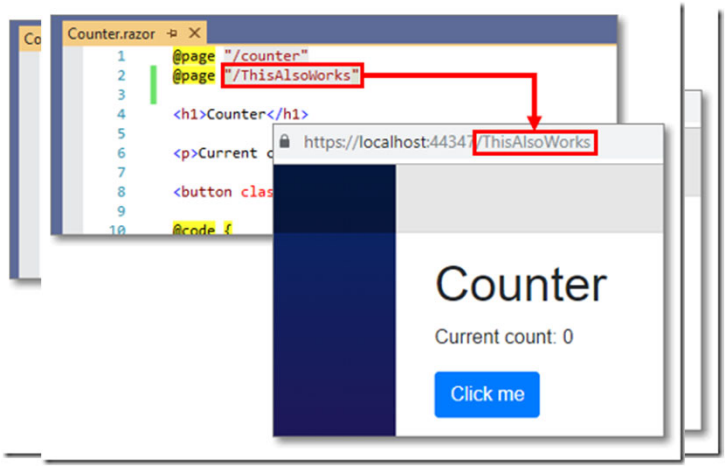
Components



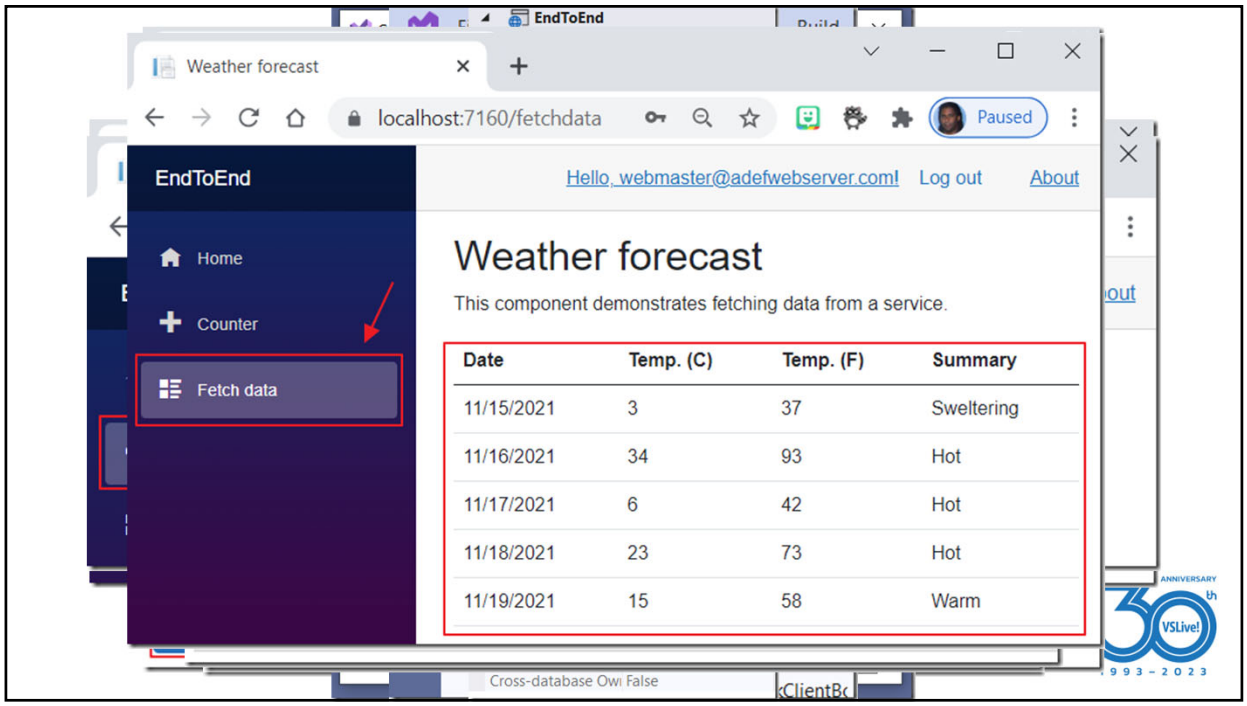
Data Binding

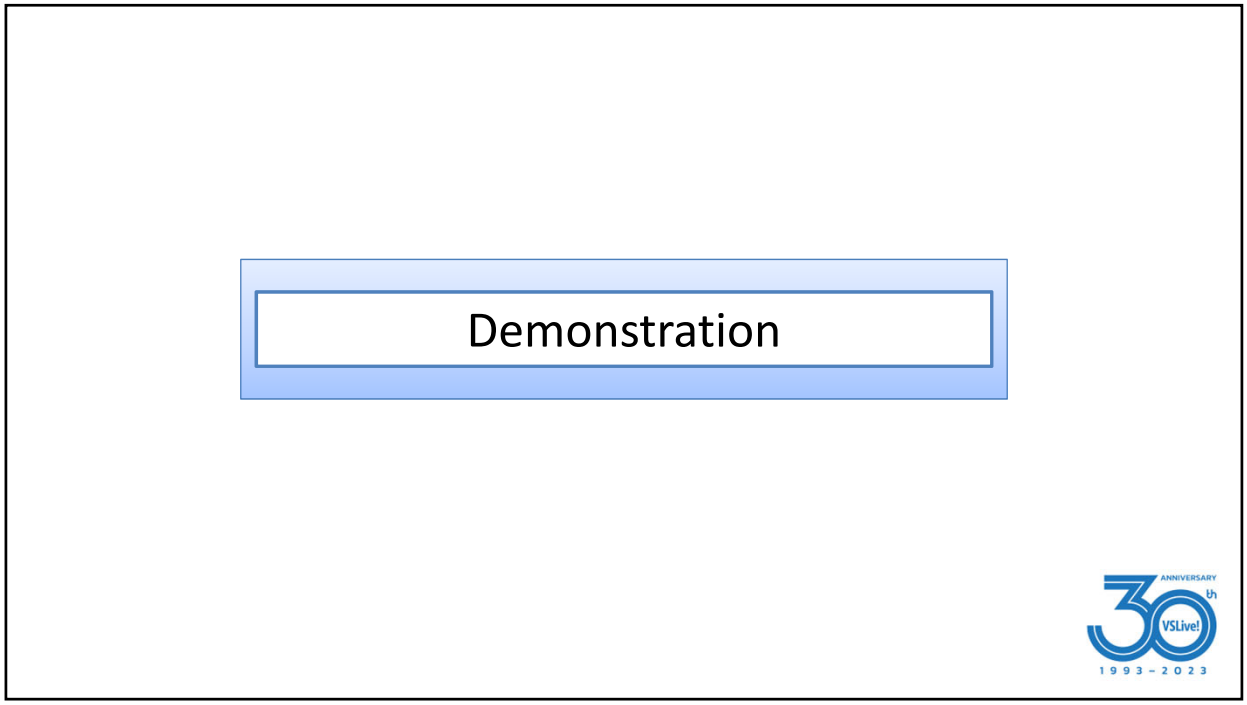


Routing

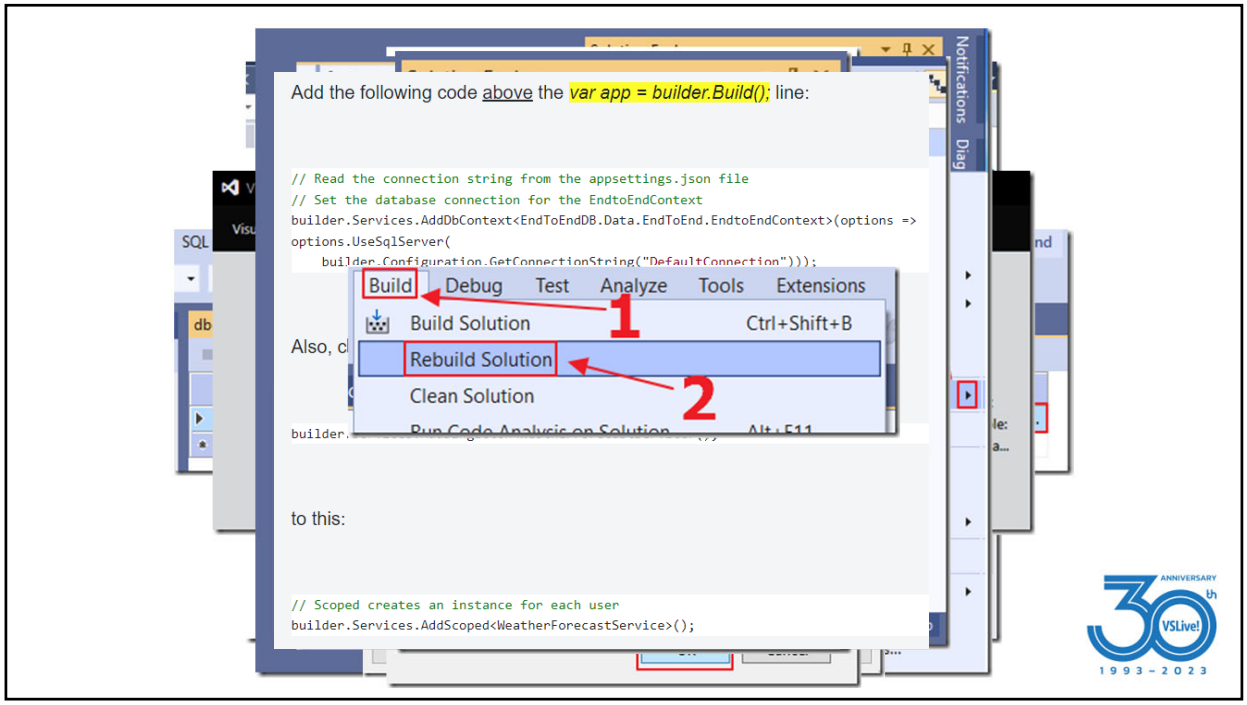


An End-To-End Sample
Server Side Blazor


EndToEnd
Weather forecast
localhost:7160/fetchdata
Paused
EndToEnd
Hello, webmaster@adefwebserver.com! Log out About
Home
Counter
Fetch data
Weather forecast
This component demonstrates fetching data from a service.
Date Temp. (C) Temp. (F) Summary
11/15/2021 3 37 Sweltering
11/16/2021 34 93 Hot
11/17/2021 6 42 Hot
11/18/2021 23 73 Hot
11/19/2021 15 58 Warm
Cross-database Owi False

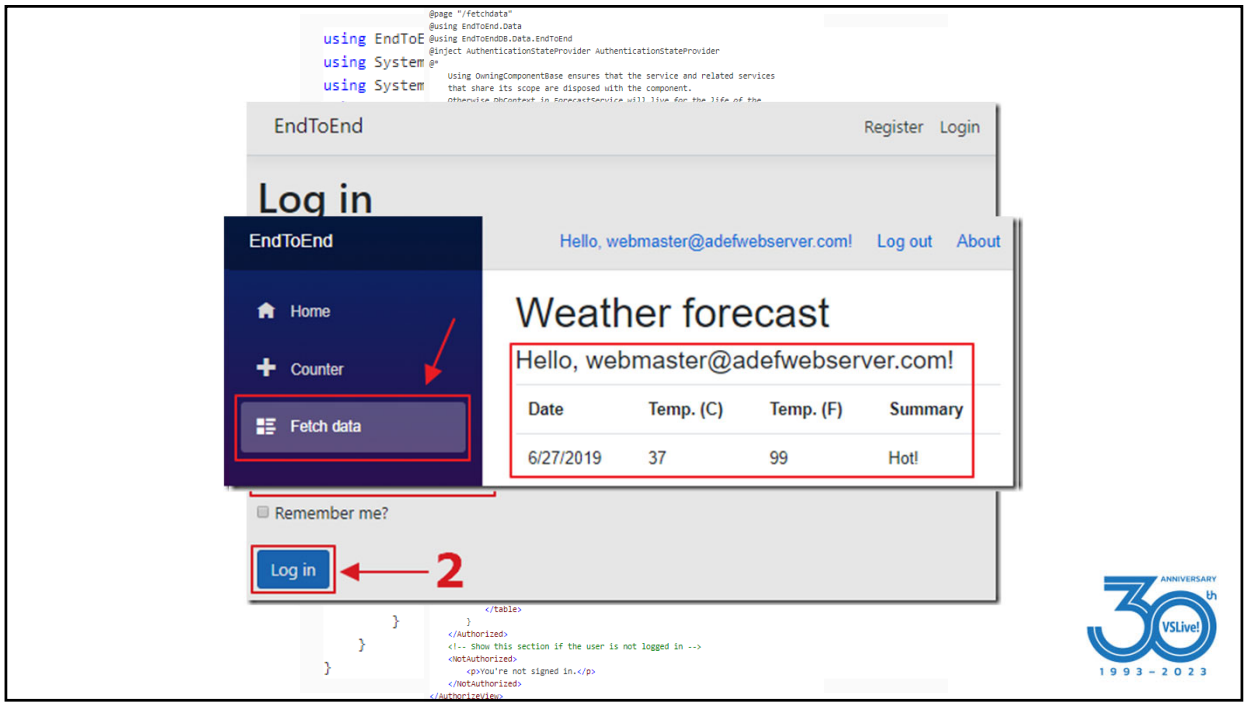

Demonstration

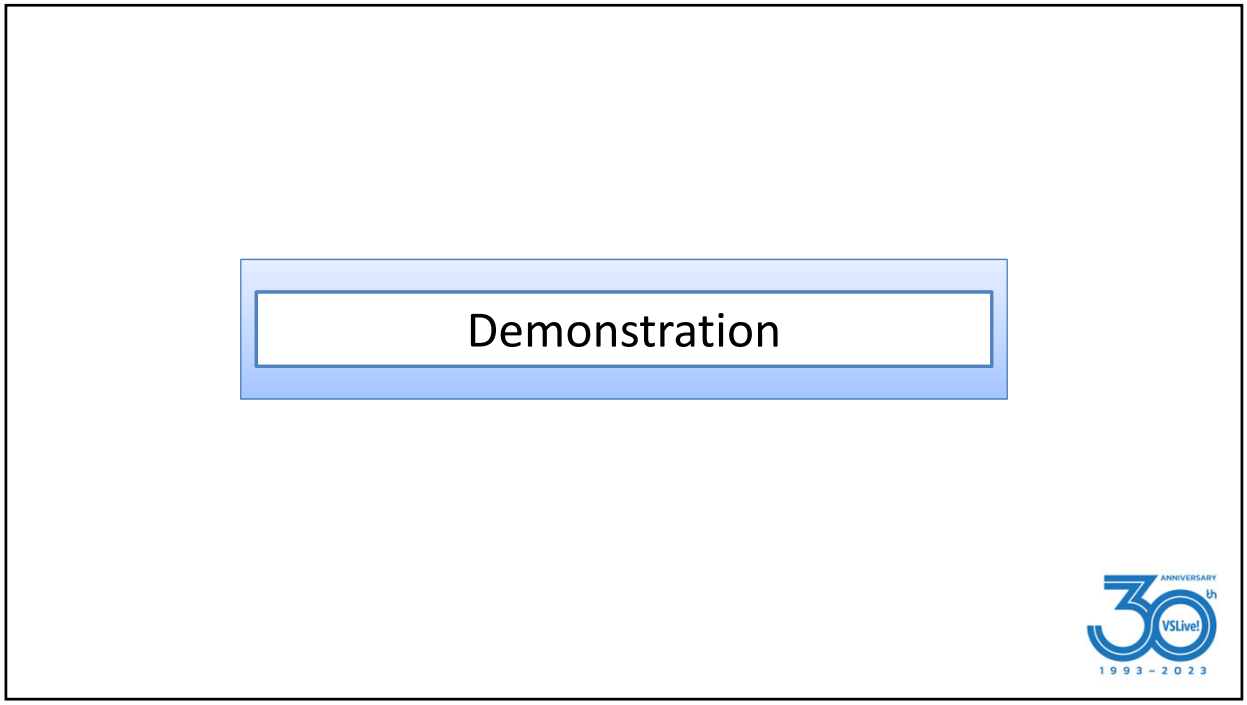
Create The Database



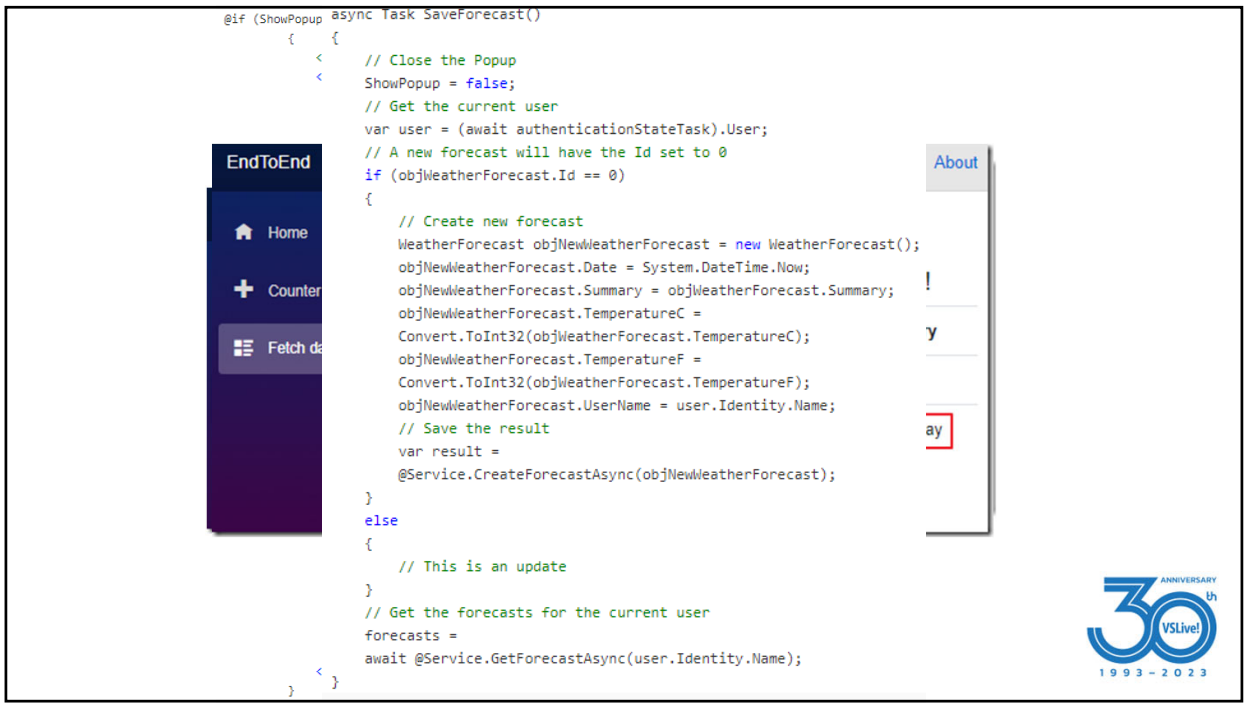
Demonstration

Read From The Database


EndToEnd
Register
Login
Log in
EndToEnd
Hello, webmaster@adefwebserver.com!
Log out
About
Home
Counter
Fetch data
Weather forecast
Hello, webmaster@adefwebserver.com!
Date
Temp. (C)
Temp. (F)
Summary
6/27/2019
37
99
Hot!
Remember me?
Log in
2

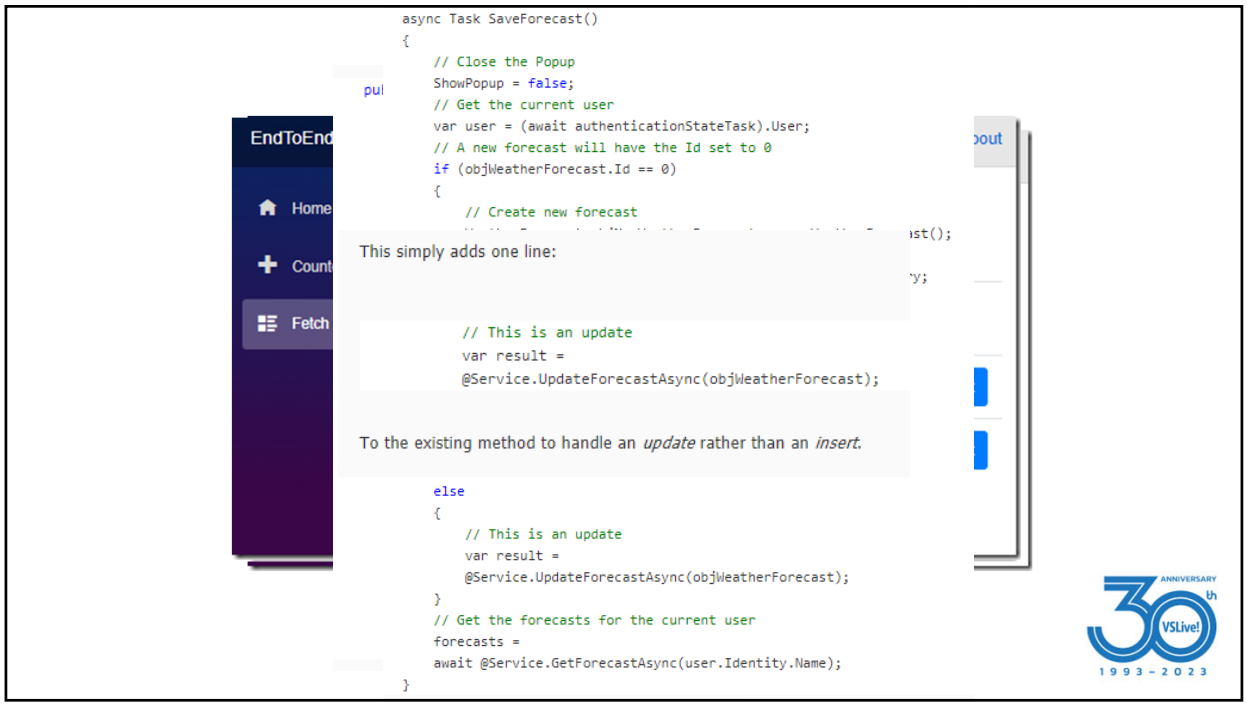

Demonstration

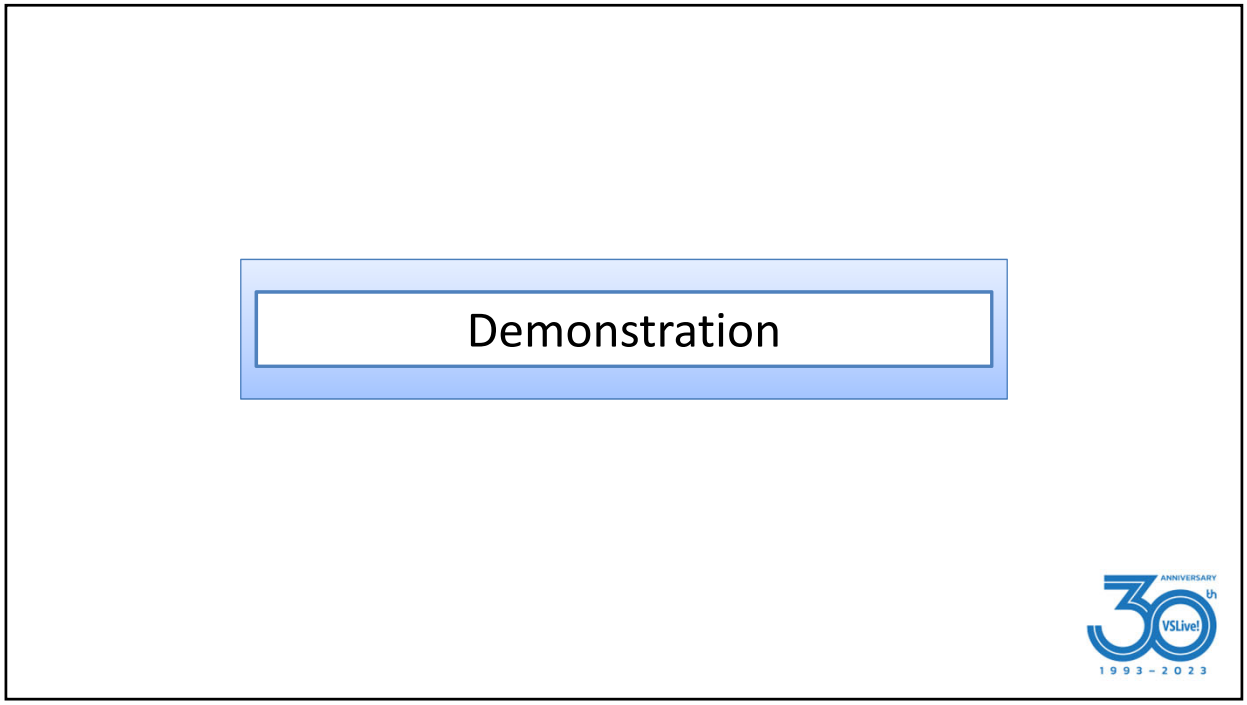
Inserting Data Into The Database



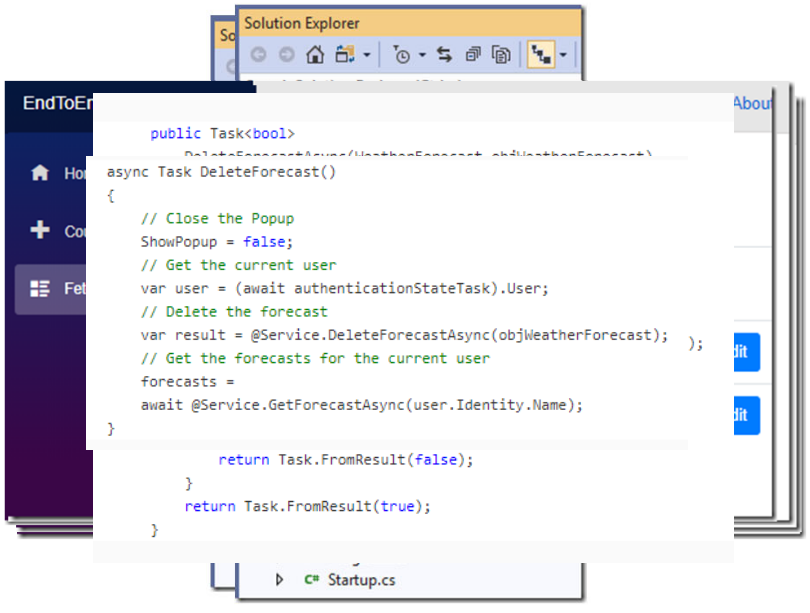
Demonstration

Updating The Data


async Task SaveForecast()
{
// Close the Popup
ShowPopup = false;
// Get the current user
var user = (await authenticationStateTask).User;
// A new forecast will have the Id set to 0
if (objWeatherForecast.Id == 0)
{
// Create new forecast
EndToEnd
Home
Fetch
This simply adds one line:
// This is an update
var result =
@Service.UpdateForecastAsync(objWeatherForecast);
To the existing method to handle an update rather than an insert.
else
{
// This is an update
var result =
@Service.UpdateForecastAsync(objWeatherForecast);
}
// Get the forecasts for the current user
forecasts =
await @Service.GetForecastAsync(user.Identity.Name);
}

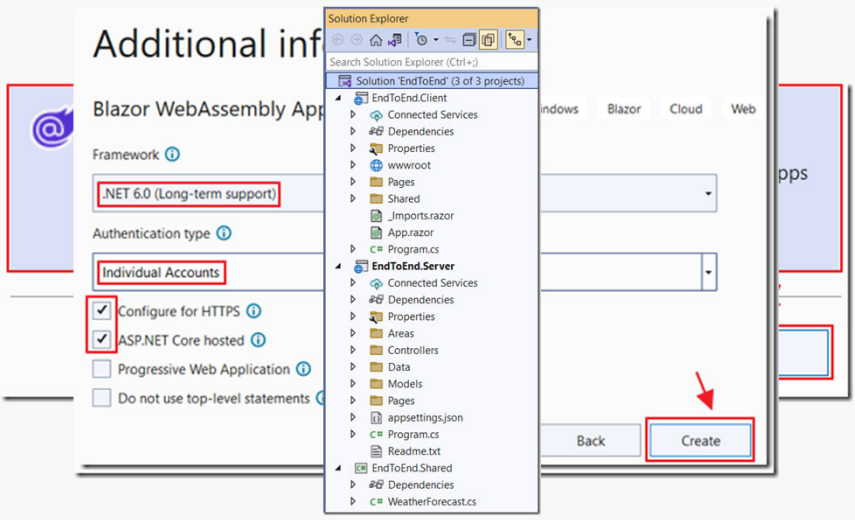

Demonstration

Deleting The Data

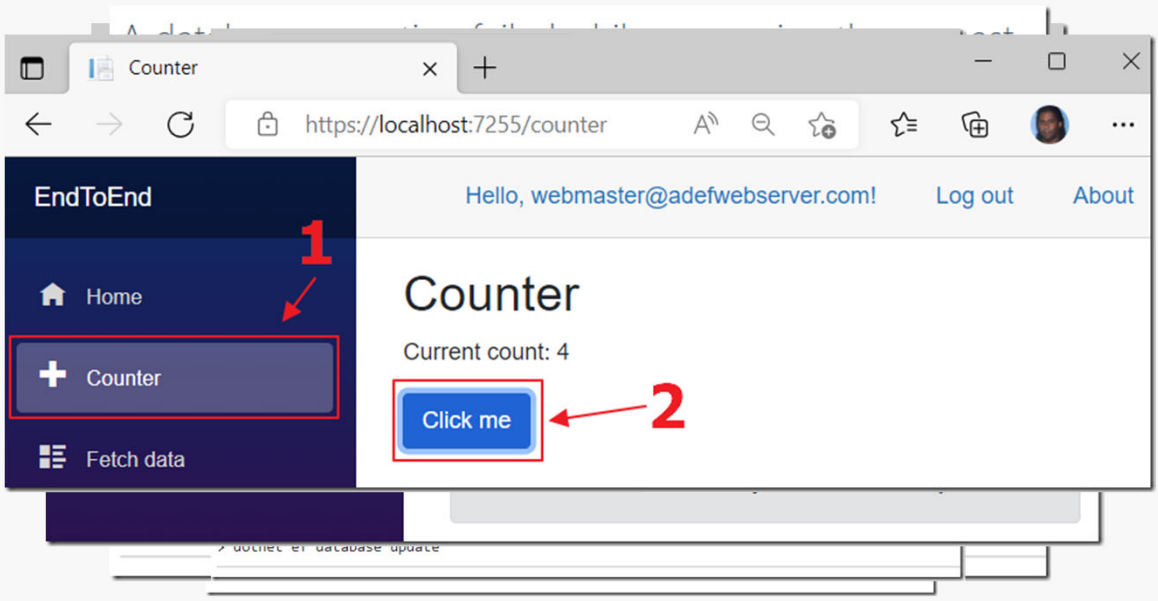


An End-To-End Sample
Client Side (WebAssembly) Blazor

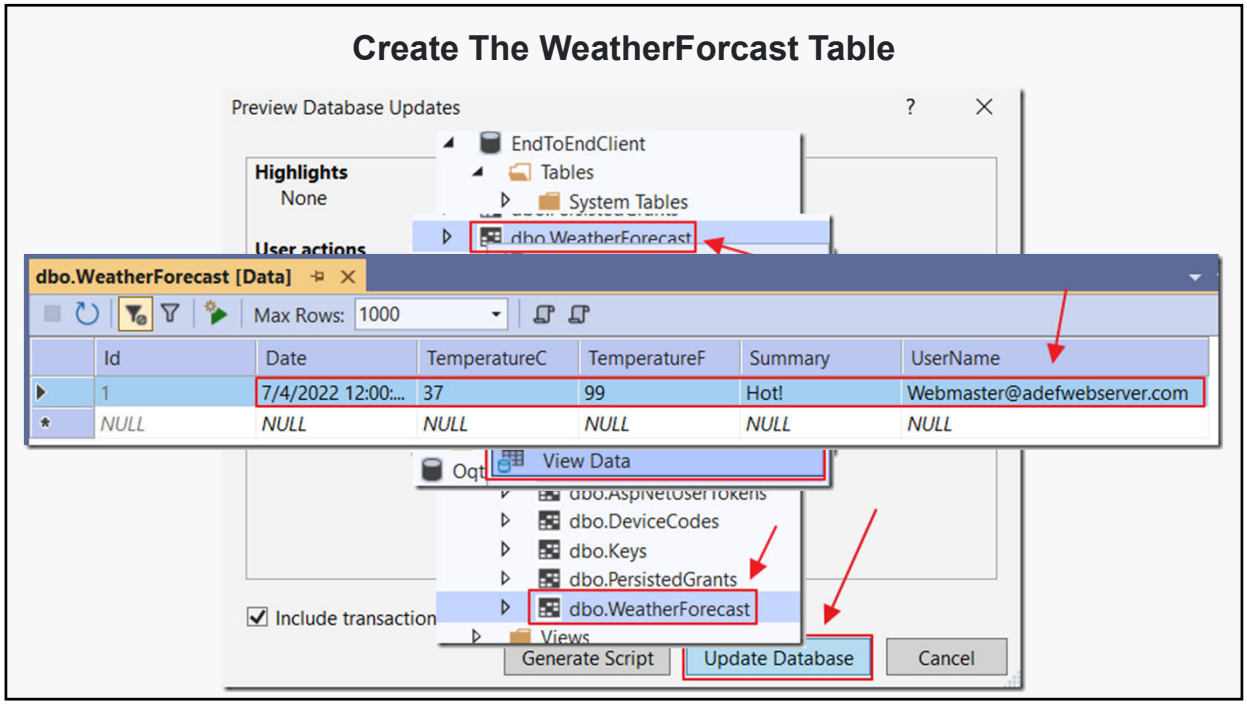
Create The Project



Create The Database



Create The WeatherForcast Table



Create The Data Context

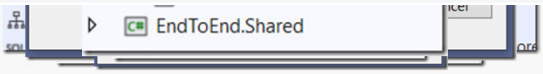
```
// Read the connection string from the appsettings.json file
// Set the database connection for the EndtoEndContext
builder.Services.AddDbContext<EndToEndDB.Data.EndToEndContext>(options =>
options.UseSqlServer(
    builder.Configuration.GetConnectionString("DefaultConnection")));
```

To allow the user name to be readable in the Controller methods (to be created later), change the folliwng line:

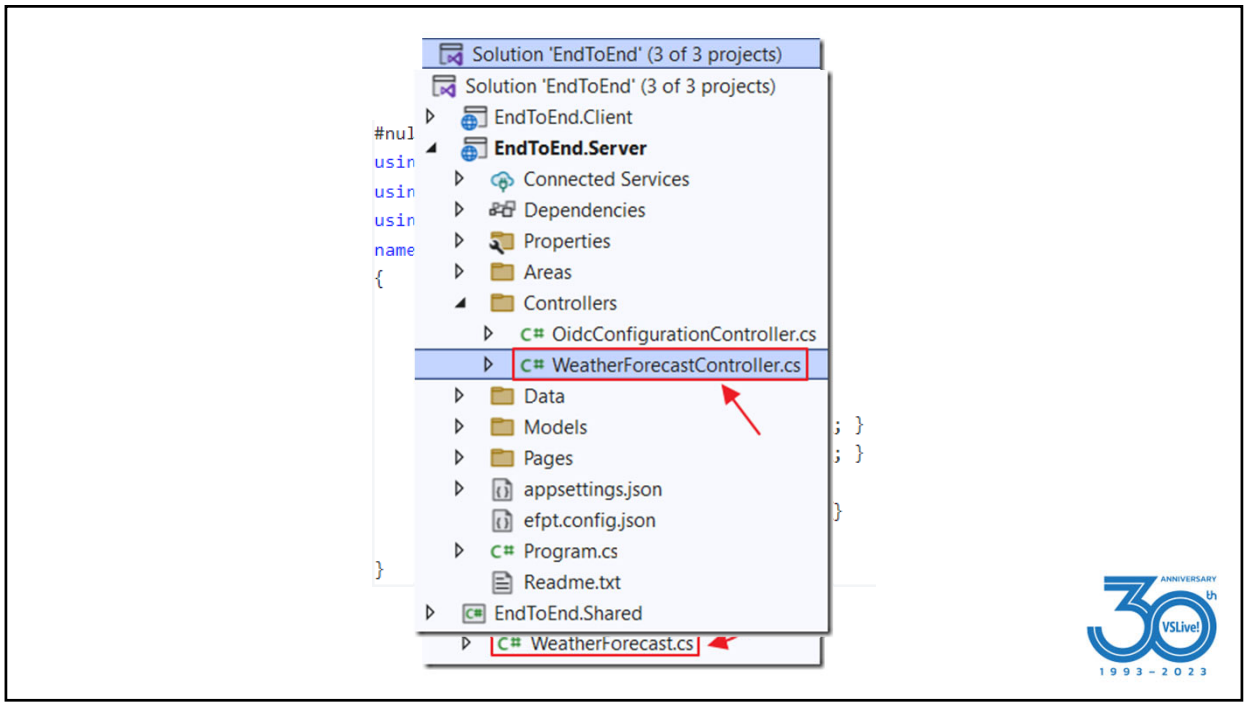
```
builder.Services.AddIdentityServer()
   .AddApiAuthorization<ApplicationUser, ApplicationDbContext>();
```

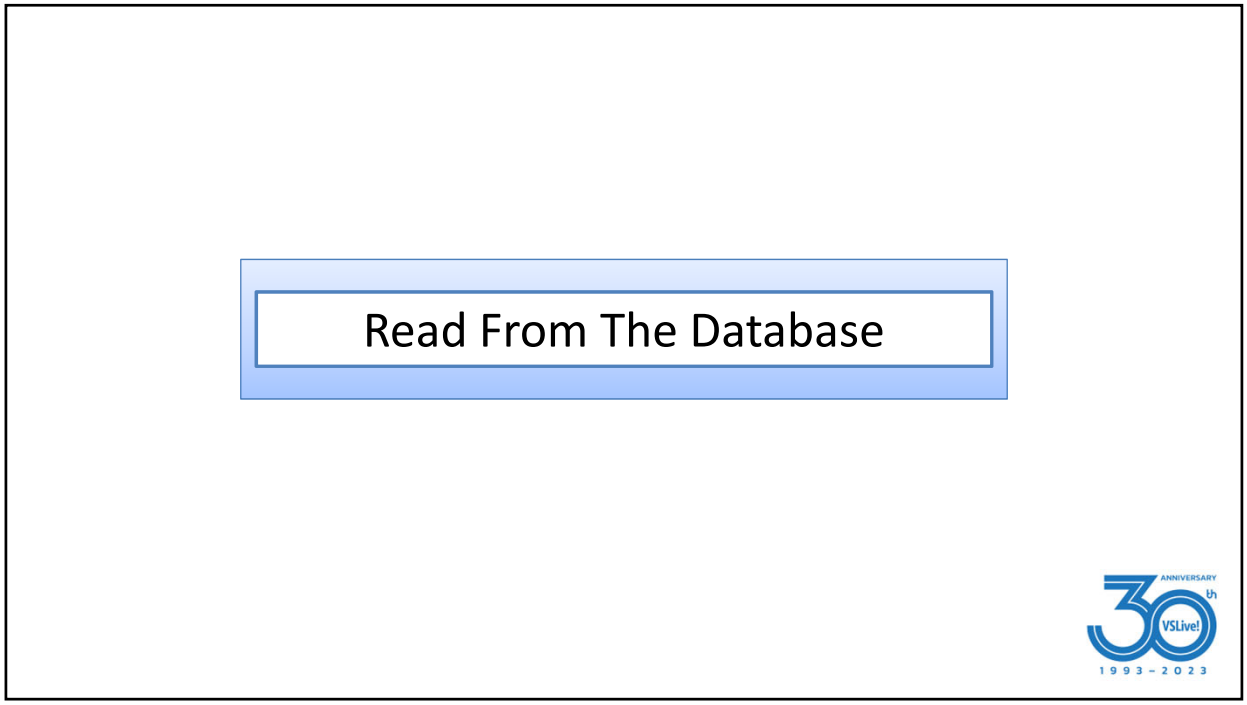
To:

```
// This allows the username (email address) of the currrently
// logged in user to be read in the server side controller
builder.Services.AddIdentityServer()
    .AddApiAuthorization<ApplicationUser, ApplicationDbContext>(options => {
        options.IdentityResources["openid"].UserClaims.Add("name");
        options.ApiResources.Single().UserClaims.Add("name");
    });
```



Client Side (WebAssembly) Blazor CRUD Methods


Solution 'EndToEnd' (3 of 3 projects)
Solution 'EndToEnd' (3 of 3 projects)
EndToEnd.Client
EndToEnd.Server
Connected Services
Dependencies
Properties
Areas
Controllers
OidcConfigurationController.cs
WeatherForecastController.cs
Data
Models
Pages
appsettings.json
efpt.config.json
Program.cs
Readme.txt
EndToEnd.Shared
WeatherForecast.cs


Read From The Database

```
[HttpGet]
[Route("api/WeatherForecast/GetAsync")]
public async Task<List<WeatherForecastDTO>>
    GetAsync()
{
    // Get the current user
    string strCurrentUser = this.User.Claims
        .Where(x => x.Type ==
        "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name")
        .FirstOrDefault().Value;

    // Get Weather Forecasts
    var result = await _context.WeatherForecast
         // Only get entries for the current logged in user
         .Where(x => x.UserName == strCurrentUser)
         // Use AsNoTracking to disable EF change tracking
         // Use ToListAsync to avoid blocking a thread
         .AsNoTracking().ToListAsync();
    // Collection to return
    List<WeatherForecastDTO> ColWeatherForecast =
        new List<WeatherForecastDTO>();
    // Loop through the results
    foreach (var item in result)
    {
        // Create a new WeatherForecast instance
        WeatherForecastDTO objWeatherForecastDTO =
            new WeatherForecastDTO();
        // Set the values for the WeatherForecast instance
        objWeatherForecastDTO.Id =
            item.Id;
        objWeatherForecastDTO.Date =
            item.Date;
        objWeatherForecastDTO.UserName =
            item.UserName;
        objWeatherForecastDTO.TemperatureF =
            item.TemperatureF;
        objWeatherForecastDTO.TemperatureC =
            item.TemperatureC;
        objWeatherForecastDTO.Summary =
            item.Summary;
        // Add the WeatherForecast instance to the collection
        ColWeatherForecast.Add(objWeatherForecastDTO);
    }
    // Return the final collection
    return ColWeatherForecast;
}
```

Inserting Data

```
[HttpPost]
[Route("api/WeatherForecast/Post")]
public void
    Post([FromBody] WeatherForecastDTO paramWeatherForecast)
{
    // Create a new WeatherForecast instance
    WeatherForecast objWeatherForecast =
        new WeatherForecast();

    // Get the current user
    string strCurrentUser = this.User.Claims
        .Where(x => x.Type ==
        "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name")
        .FirstOrDefault().Value;

    // Set the values for the WeatherForecast instance
    objWeatherForecast.Id =
        paramWeatherForecast.Id;
    objWeatherForecast.Date =
        paramWeatherForecast.Date;
    objWeatherForecast.UserName =
        strCurrentUser;
    objWeatherForecast.TemperatureF =
        paramWeatherForecast.TemperatureF;
    objWeatherForecast.TemperatureC =
        paramWeatherForecast.TemperatureC;
    objWeatherForecast.Summary =
        paramWeatherForecast.Summary;
    // Save to the database
    _context.WeatherForecast.Add(objWeatherForecast);
    _context.SaveChanges();
}
```

Updating Data

```
[HttpPut]
[Route("api/WeatherForecast/Put")]
public void
       Put([FromBody] WeatherForecastDTO objWeatherForecast)
{
    // Get the current user
    string strCurrentUser = this.User.Claims
        .Where(x => x.Type ==
        "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name")
        .FirstOrDefault().Value;

    // Find the existing Weather Forecast record
    // using the id
    var ExistingWeatherForecast =
        _context.WeatherForecast
        .Where(x => x.Id == objWeatherForecast.Id)
        .Where(x => x.UserName == strCurrentUser)
        .FirstOrDefault();
    if (ExistingWeatherForecast != null)
    {
        // Update the values
        ExistingWeatherForecast.Date =
            objWeatherForecast.Date;
        ExistingWeatherForecast.Summary =
            objWeatherForecast.Summary;
        ExistingWeatherForecast.TemperatureC =
            objWeatherForecast.TemperatureC;
        ExistingWeatherForecast.TemperatureF =
            objWeatherForecast.TemperatureF;
        // Save to the database
        _context.SaveChanges();
    }
}
```

Deleting Data

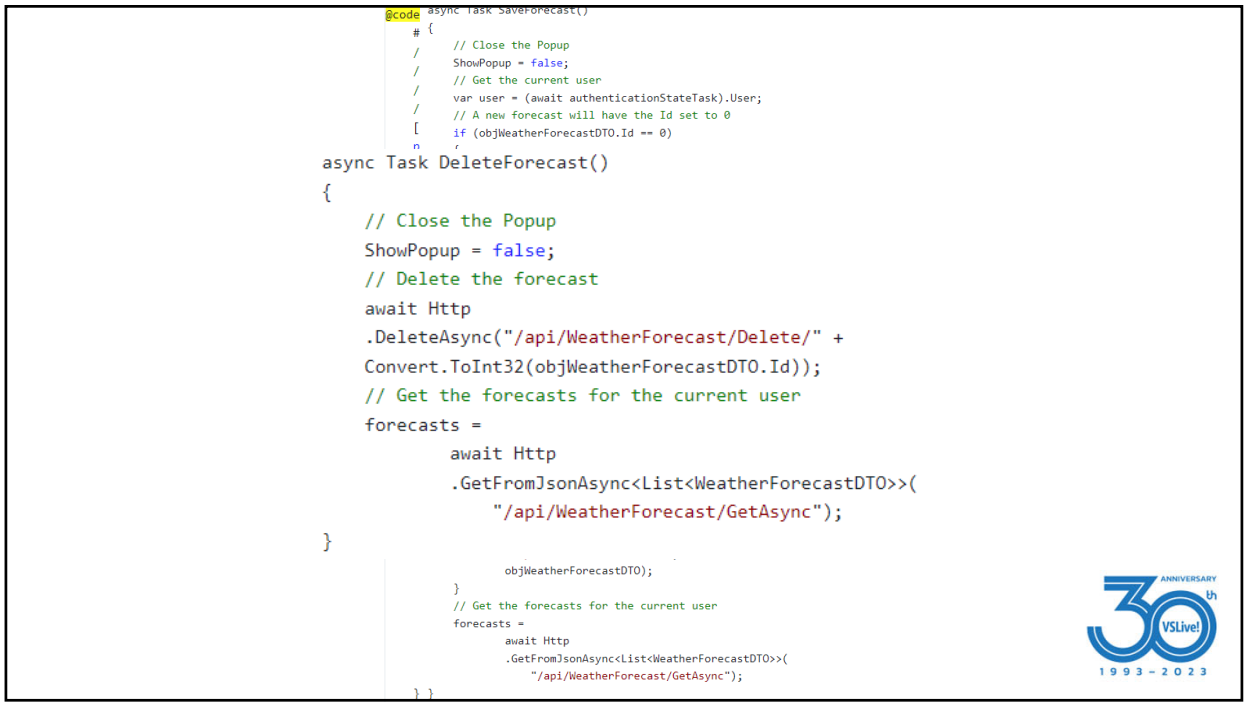
```
[HttpDelete]
[Route("api/WeatherForecast/Delete/{id}")]
public void
    Delete(int id)
{
    // Get the current user
    string strCurrentUser = this.User.Claims
        .Where(x => x.Type ==
        "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/name")
        .FirstOrDefault().Value;

    // Find the existing Weather Forecast record
    // using the id
    var ExistingWeatherForecast =
        _context.WeatherForecast
        .Where(x => x.Id == id)
        .Where(x => x.UserName == strCurrentUser)
        .FirstOrDefault();
    if (ExistingWeatherForecast != null)
    {
        // Remove from the database
        _context.WeatherForecast
            .Remove(ExistingWeatherForecast);
        _context.SaveChanges();
    }
}
```

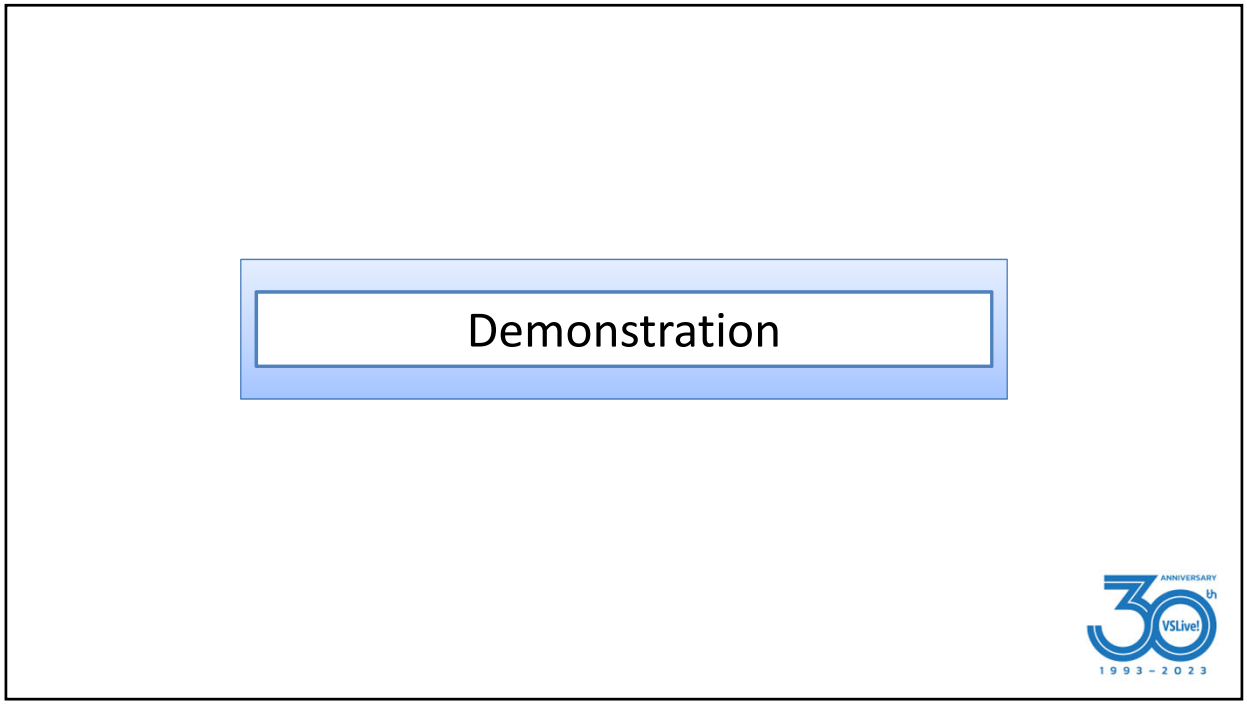
Client Code - Markup

```
<!- @if (ShowPopup)
<ta {
        <!-- This is the popup to create or edit a forecast -->
        <div class="modal" tabindex="-1" style="display:block"
             role="dialog">
            <div class="modal-dialog">
                <div class="modal-content">
                    <div class="modal-header">
                        <h3 class="modal-title">Edit Forecast</h3>
                        <!-- Button to close the popup -->
                        <button type="button" class="close"
                                @onclick="ClosePopup">
                            <span aria-hidden="true">X</span>
                        </button>
                    </div>
                    <!-- Edit form for the current forecast -->
                    <div class="modal-body">
                        <input class="form-control" type="text"
                               placeholder="Celsius forecast"
                               @bind="objWeatherForecastDTO
                                        .TemperatureC" />
                        <input class="form-control" type="text"
                               placeholder="Fahrenheit forecast"
                               @bind="objWeatherForecastDTO
                                        .TemperatureF" />
                        <input class="form-control" type="text"
                               placeholder="Summary"
                               @bind="objWeatherForecastDTO.Summary" />
                        <br />
                        <!-- Button to save the forecast -->
                        <button class="btn btn-primary"                    ">
                                @onclick="SaveForecast">
                            Save
                        </button>
                        <!-- Only show button if not a new record -->
                        @if (objWeatherForecastDTO.Id > 0)
                        {
                            <!-- Button to delete the forecast -->
                            <button class="btn btn-primary"
</t                                  @onclick="DeleteForecast">
<p                              Delete
                            </button>
                        }
                    </div>
                </div>
            </div>
        </div>
</p }
```

Client Code – C# Code

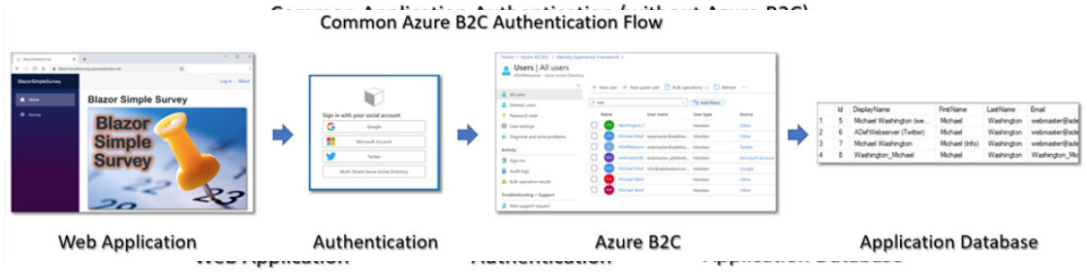


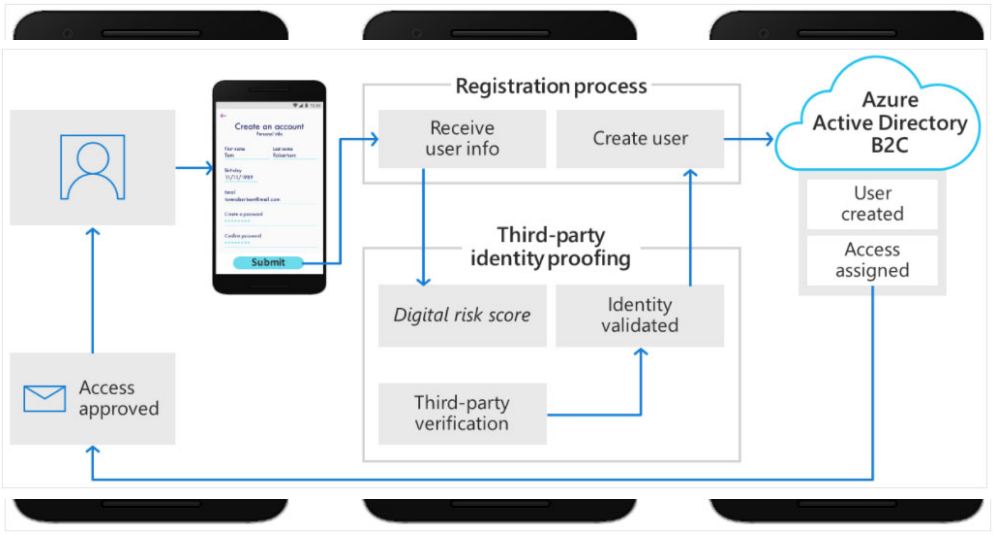
```
async Task DeleteForecast()
{
    // Close the Popup
    ShowPopup = false;
    // Delete the forecast
    await Http
    .DeleteAsync("/api/WeatherForecast/Delete/" +
    Convert.ToInt32(objWeatherForecastDTO.Id));
    // Get the forecasts for the current user
    forecasts =
            await Http
            .GetFromJsonAsync<List<WeatherForecastDTO>>(
                "/api/WeatherForecast/GetAsync");
}
```



Demonstration

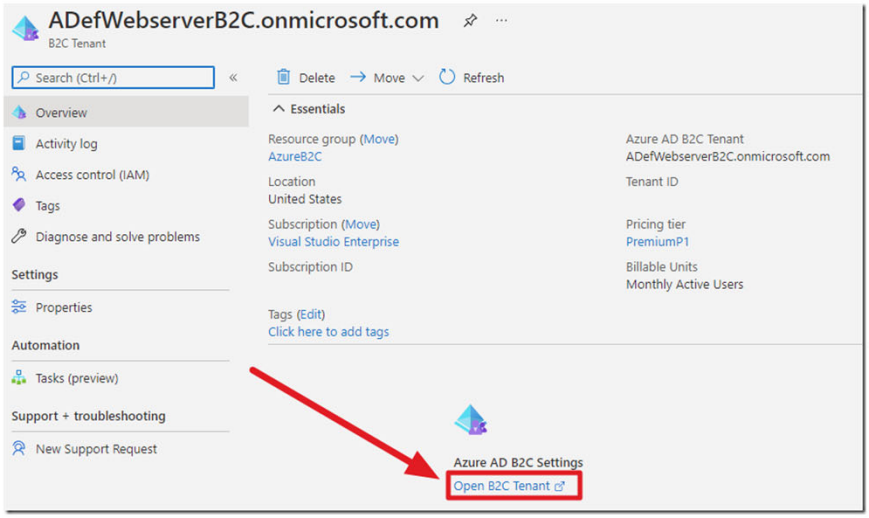
About Azure B2C



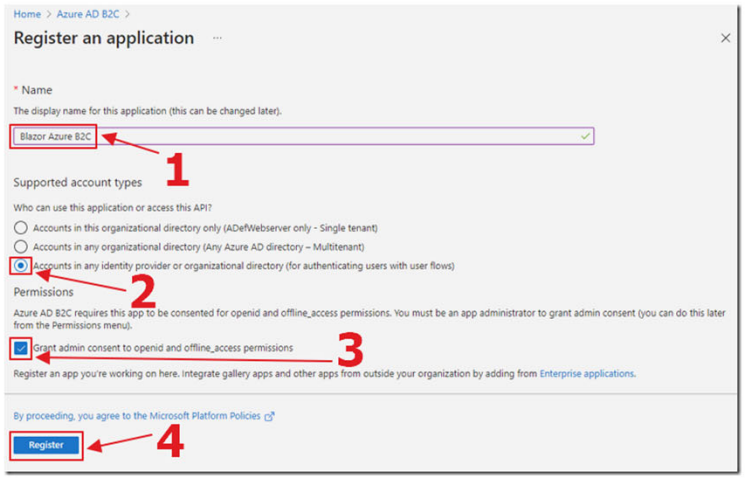


All slides © Microsoft

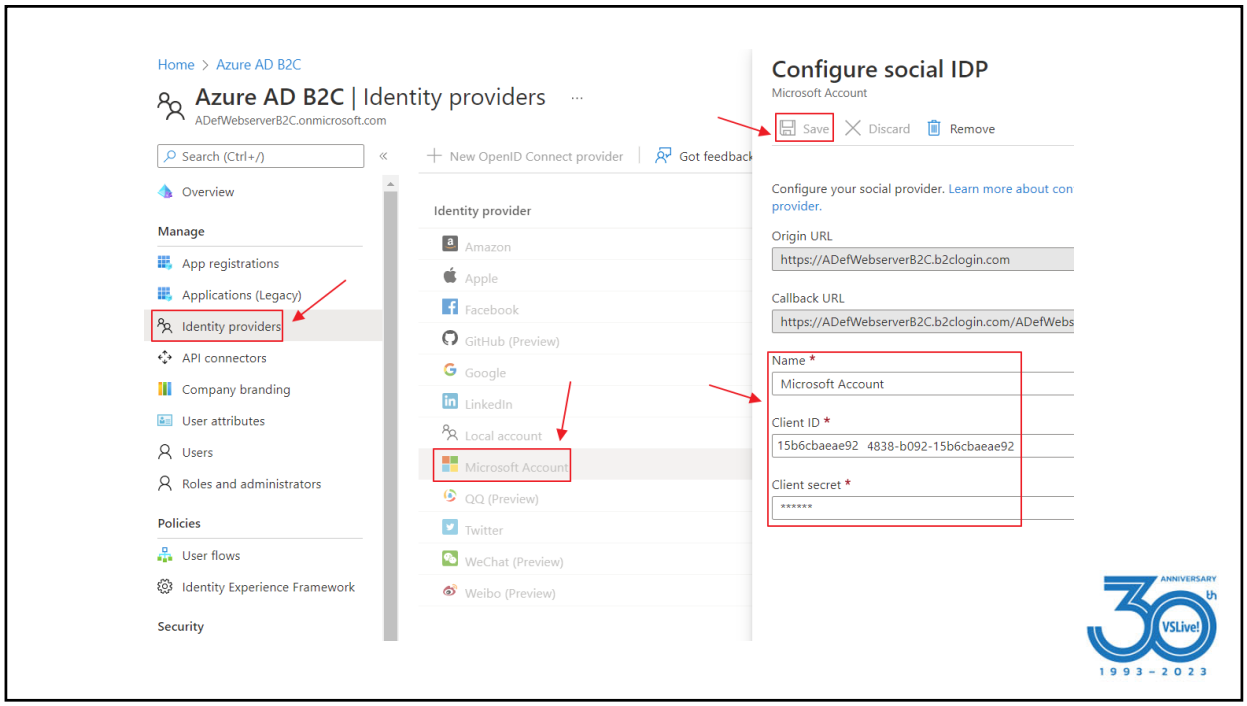
Set-Up An Azure Active Directory B2C Tenant

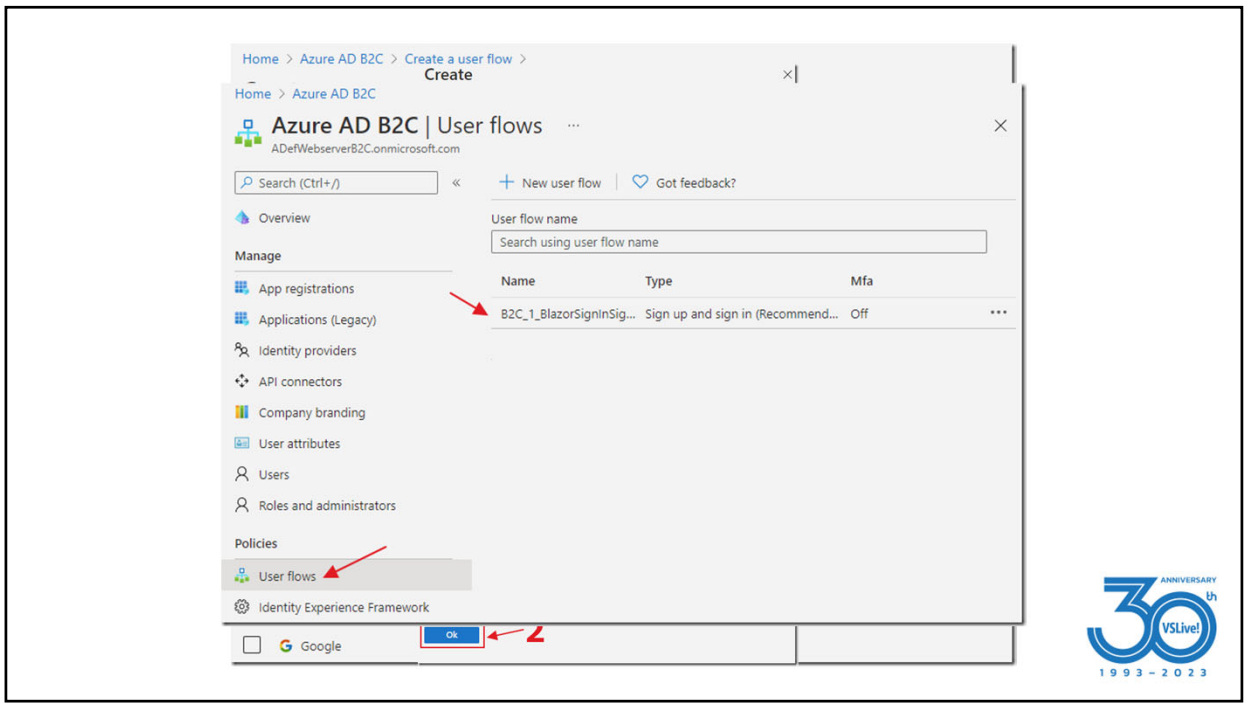


Register App in Azure AD B2C tenant

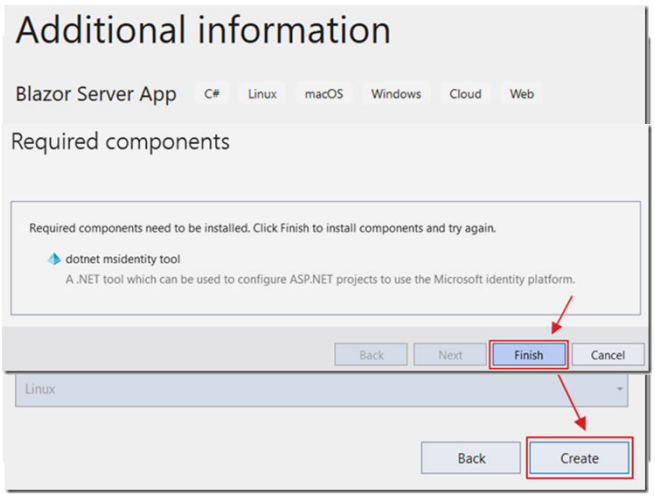


Configure Azure AD B2C Identity Provider & User Flow


Home > Azure AD B2C
Azure AD B2C | Identity providers
ADefWebserverB2C.onmicrosoft.com
Search (Ctrl+/)
Overview
Manage
App registrations
Applications (Legacy)
Identity providers
API connectors
Company branding
User attributes
Users
Roles and administrators
Policies
User flows
Identity Experience Framework
Security
New OpenID Connect provider
Got feedback
Identity provider
Amazon
Apple
Facebook
GitHub (Preview)
Google
LinkedIn
Local account
Microsoft Account
QQ (Preview)
Twitter
WeChat (Preview)
Weibo (Preview)
Configure social IDP
Microsoft Account
Save
Discard
Remove
Configure your social provider. Learn more about con
provider.
Origin URL
https://ADefWebserverB2C.b2clogin.com
Callback URL
https://ADefWebserverB2C.b2clogin.com/ADefWebs
Name *
Microsoft Account
Client ID *
15b6cbaeae92 4838-b092-15b6cbaeae92
Client secret *
******

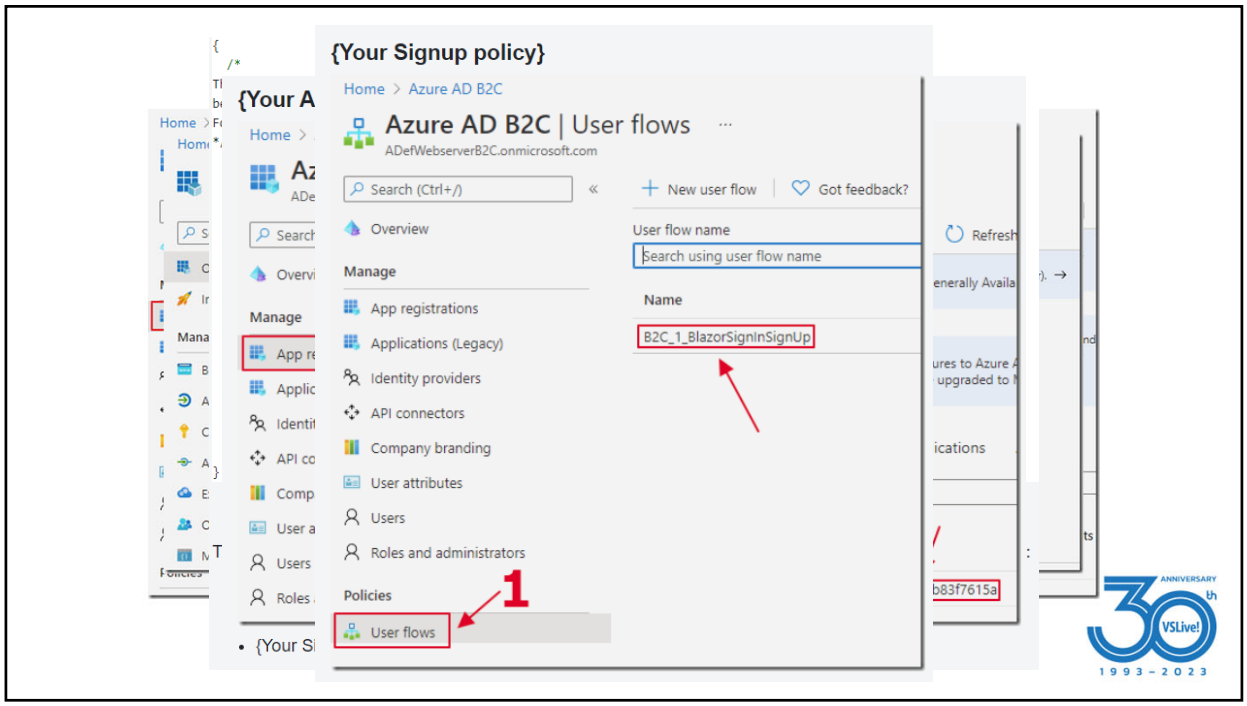

Home > Azure AD B2C > Create a user flow >
Create
Home > Azure AD B2C
Azure AD B2C | User flows
ADefWebserverB2C.onmicrosoft.com
Search (Ctrl+/)
Overview
Manage
App registrations
Applications (Legacy)
Identity providers
API connectors
Company branding
User attributes
Users
Roles and administrators
Policies
User flows
Identity Experience Framework
New user flow
Got feedback?
User flow name
Search using user flow name
Name
Type
Mfa
B2C_1_BlazorSignInSig...
Sign up and sign in (Recommend...
Off
Google
Ok

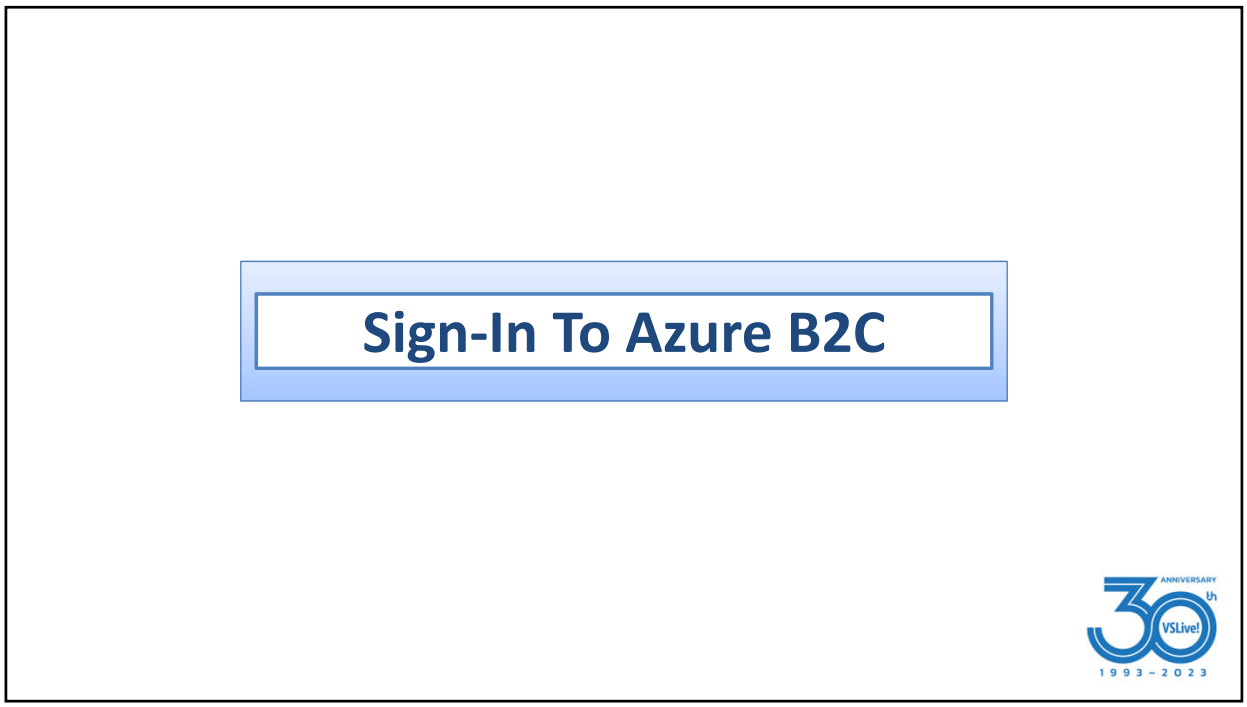
Blazor Server App



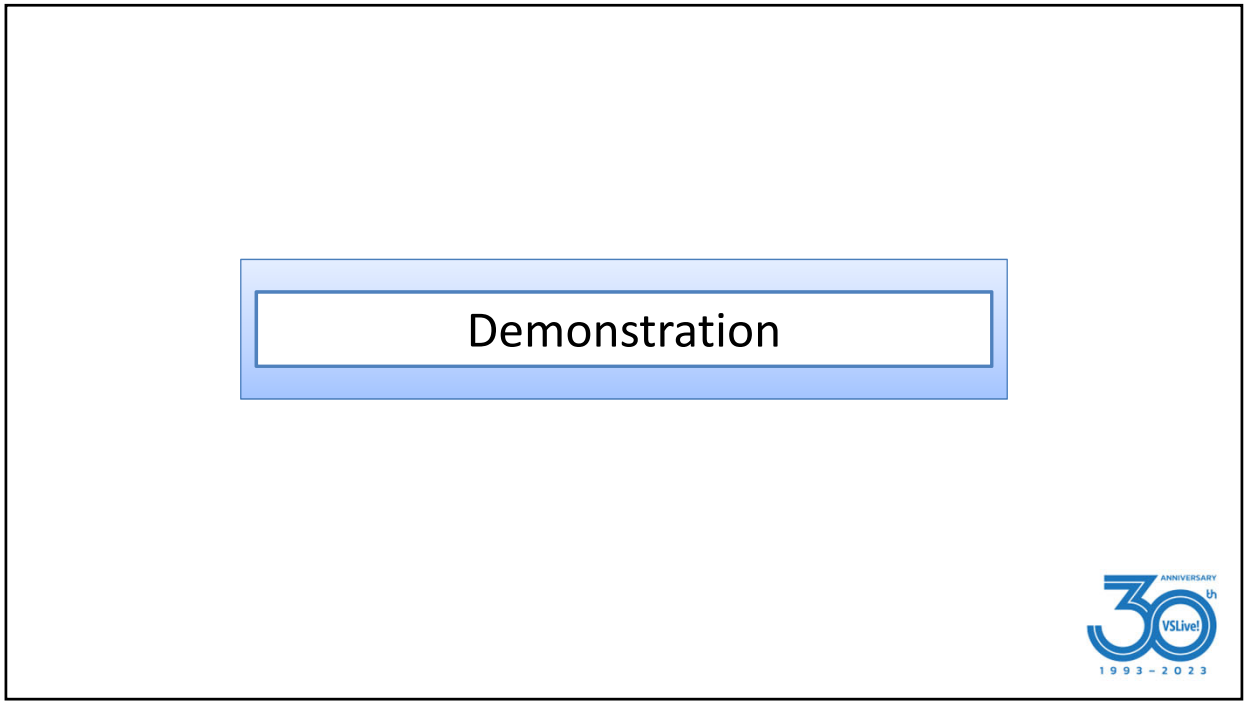
Demonstration

Add the Redirect URI in the Azure App Registration


{Your Signup policy}
Home > Azure AD B2C
Azure AD B2C | User flows
ADefWebserverB2C.onmicrosoft.com
Search (Ctrl+/)
New user flow
Got feedback?
Overview
Manage
App registrations
Applications (Legacy)
Identity providers
API connectors
Company branding
User attributes
Users
Roles and administrators
Policies
User flows
1
User flow name
Search using user flow name
Name
B2C_1_BlazorSignInSignUp


Sign-In To Azure B2C


Cancel
DefWebserver
Test User
BlazorAzureB2C
Hello, Test User!
Log out
About
Home
Counter
Fetch data
Hello, world!
Welcome to your new app.
How is Blazor working for you? Please take our brief survey and tell us what you think.
User
Continue
2


Demonstration

Closing

All Resources
Files / Tutorials

Blazor HelpWebsite
http://BlazorHelpWebsite.com

Thank You!