# Workshop DevDays 2022

In deze workshop zullen jullie een app “KVK-checker” bouwen die koppelt met Focus (SB+). Omdat we beperkte tijd hebben, hebben we alvast wat bestanden voor je klaargezet.

## Fase 1: De webapplicatie (App developer)

Je begint met een nieuwe .NET webapplicatie te maken op de volgende manier:

* Start de terminal in VSCode
* Voer het volgende commando uit:  
  **dotnet new web**
* Nu worden de bestanden die nodig zijn voor een .NET web applicatie voor je gegenereerd.

De volgende stap is het toevoegen van de landing page. Dit is de pagina waar Focus naartoe navigeert zodra je vanuit het “Focus App Center” de app onboarding start. Open de zojuist gegenereerde **Program.cs** file en voeg de onderstaande regel toe boven de **app.Run();** regel:

**LandingPage.Register(app);**

Intussen heeft vscode als het goed is een prompt getoond of je “required assets for build and debug” wilt toevoegen, doe dit, zodat er een .vscode folder ontstaat.

Als je **LandingPage.cs** bekijkt (deze staat in de directory WorkshopLibrary), dan zie je dat de pagina een focus-url ontvangt om mee te communiceren. Voor nu hebben we even genoeg gedaan aan onze webapplicatie.

Start de webapplicatie met **F5**. Je webapplicatie opent nu in een browser. Zo niet dan zie je in de terminal https://localhost:[1234] die je dan kan opstarten. Onthoud het poortnummer waarop de applicatie draait. Mocht je een melding krijgen dat de certificaat niet vertrouwd is, draai dan het volgende commando in een terminal: dotnet dev-certs https --trust

## Fase 2: De registratie in Focus (AFAS)

Focus kent het “App center” waar alle applicaties geregistreerd staan die gekoppeld kunnen worden. Deze lijst wordt door AFAS beheerd. Via een “AFAS import” kan je hier de KVKChecker aan toevoegen.

In de import folder staan de bestanden die we nu gaan importeren. Open .\import\AppClient.csv. Maak nu de volgende aanpassingen in deze csv file:

* Wijzig het poortnummer 2 keer in de ClientRedirectUrl en ClientLandingPageUrl naar het poortnummer van jouw applicatie
* Mocht je met meerdere mensen in jouw groepje de workshop willen doen, wijzig dan ook de “Naam” KVKChecker in iets unieks
* Sla deze wijzigingen op

Open de url <https://ddtech00.testafasfocus.ad.afas.nl:401/> en wijzig dan 00 naar jouw groepsnummer (met voorloopnul). Het kan zijn dat jij geen toegang hebt, vraag dan even de begeleiders om dit dit op te lossen. Open nu de spotlight via **F3** en ga naar **Import**. Ga dan in de lijst-switcher naar ‘Uitgevoerde AFAS imports’ en klik hier op Meer 🡪 Importeer AFAS data.

Importeer nu de bestanden die in de import map staan en geef een unieke omschrijving

Graphical user interface, application

Description automatically generated

Na een paar seconden zie je dat jouw import gelukt is. Ga nu in Focus via **F3** naar App Center en klik op KVKChecker (of de naam die jij gegeven hebt in de import). Aan het einde van de url nu in je browserbalk staat vind je een guid. Dit is de clientId die je verderop nodig hebt, dus zet deze even op je clipboard. Klik nu op “Activeren”. Als het allemaal goed gaat is zie je nu jouw landing page. Laat deze open staan en klik nog niet op ja.   
  
Text

Description automatically generated

## Fase 3: Onboarding (App developer)

Nu gaan we de app authenticatie toevoegen aan je KVKChecker webapplicatie. Stop je webserver en voeg in program.cs ***voor*** builder.Build() de volgende regel toe (je vult hier je clientId en clientSecret uit fase 2 in):

**FocusAuthorization.SetUp(builder, "**[clientId]**", "your\_workshop\_secret");**  
  
Voeg ook aan Program.cs ***na*** Builder.Build() de volgende regel toe:

app.UseAuthentication();

app.UseAuthorization();

Nu gaan we een webpagina toevoegen die de functionaliteit gaat uitvoeren. Omdat dit een langlopend proces is gebruiken we een websocket. Voeg daarvoor de volgende regels toe aan program.cs ***voor*** app.Run()

RunPage.Register(app);

RunScriptPage.Register(app, async (context, Log) => {

  await Log("Executing KVKChecker script version 0.0");

  await Task.Delay(1000);

  await Log("Done");

});

Start nu de web applicatie opnieuw en ga verder vanaf de landing page die je nog open had staan (of start opnieuw op “Activeren” in Focus). Nu start je de oAuth onboarding vanuit je KvkChecker App.  
  
Hierna komt je op de run pagina terecht waar je ziet dat je het bovenstaande script uitgevoerd wordt. Dit wil zeggen dat de koppeling is gelegd en kan er gecommuniceerd kan worden met Focus.

## Fase 4: Maken van de get-connector (AFAS)

Voor de koppeling gaan we organisaties ophalen uit Focus om deze vervolgens te vergelijken met de gegevens uit de KVK. Hiervoor maken we een nieuwe get-connector aan in Focus.

We doen dit als volgt:

* Ga in Focus (via **F3**) naar “Applicatie onderdelen”
* Doe hier “Nieuwe GetConnector”
* Vul het veld **url** met **kvkchecker-organisaties**
  + Als meer mensen uit je groep de workshop doen zorg dan voor een unieke waarde
* Open de **Query voor gegevens**
* Gebruik gegevensverzameling **Organisatie**
* Voeg de velden toe uit onderstaand screenshot
* Sla de query op en sla vervolgens het design op

Graphical user interface, application

Description automatically generated

## Fase 5: KvkChecker functie maken (App developer)

Nu je de koppeling met Focus tot stand hebt gebracht heeft je webapplicatie een tokens gekregen waarmee de Focus API’s aangeroepen worden.

Hieronder staan wat stukken voorbeeldcode waarmee je tokens kunt ophalen, de get-connector van Focus kunt aanroepen en de kvk kunt aanroepen.

Vervang nu de regel **await Task.Delay(1000);** in **Program.cs** door zelf geschreven code die de lijst met organisaties download.

// Getting an access token

var accessToken =

  await Microsoft.AspNetCore.Authentication.AuthenticationHttpContextExtensions

  .GetTokenAsync(context, "access\_token");

// HttpClient to make api calls to Focus with

using var focusClient = new HttpClient();

focusClient.DefaultRequestHeaders.Add("AccessToken", accessToken);  
  
// Query the get-connnector to retrieve organizations  
var getOrganizationsUrl = FocusOAuthHandler.GetFocusUrl(context)   
 + "/api/kvkchecker-organisaties";  
await Log("Querying " + getOrganizationsUrl);  
var organizations = await focusClient  
 .GetFromJsonAsync<List<FocusOrganization>>(getOrganizationsUrl);  
await Log(string.Format("Found {0} results", organizations!.Count));

Als dit gelukt is gebruik je onderstaande stukken code om alle organisaties die een kvk-nummer hebben op te zoeken bij de Kvk. De api key van de KvK die je nodig hebt heb je via mail van ons ontvangen.   
Goed om te weten: doe niet teveel calls met deze key, anders hebben klanten hier last van.

// Querying the kvk to get information about a single organization

using var kvkClient = new HttpClient();

kvkClient.DefaultRequestHeaders.Add("apikey", apiKey);

var url = string.Format(  
 "https://api.kvk.nl/api/v1/basisprofielen/{0}/hoofdvestiging",   
 kvkNummer);

await Log("Querying " + url);

try   
{

  var result = await kvkClient.GetFromJsonAsync<KvkOrganization>(url);

  await Log("Found");

}

catch (Exception)

{

  await Log("Not Found");

}

## Fase 6: Update connector in Focus (AFAS)

Focus kent het concept van signalen, dit zijn berichten van Focus aan de gebruiker die een mogelijk probleem signaleren. Sommige signalen verdwijnen als ze opgelost zijn en andere signalen kun je handmatig afmelden.

Jullie KVKChecker gaat via een update-connector een signaal aanmaken voor elke organisatie die niet bekend is bij de KvK.

De JSON die jullie via deze update-connector willen versturen gaat de volgende gegevens bevatten:  
  
**{**   
 **"organization": "2ad55b99-d3cf-41b8-8126-e443dc1bb0ef",**  
 **"message": "De organisatie \*\*\* is niet bekend bij de KVK"**  
**}**

Ga nu in Focus naar “Applicatie onderdelen” en voeg een nieuwe UpdateConnector toe.

Vul de velden als volgt in:

* Url: **kvkchecker-organisatie-signaal**
  + Als meer mensen uit je groep de workshop doen zorg dan voor een unieke waarde
* Transformatie naar entiteit: zie hieronder
* JSON Schema: het bestand **kvkchecker-organisatie-signaal-schema.json** uit de repository

In het veld “Transformatie naar entiteit” beschrijf je in ATF template formaat ([zie hier](https://dev.azure.com/afassoftware/Anta/_git/templating?path=/docs/ReadMe.md&_a=preview) voor de beschrijving van de mogelijkheden van ATF) hoe je de gegevens uit de json-request gebruikt om een entiteit aan te maken of te updaten. Je hebt hier via autocompletion (ctrl-spatie) de mogelijke entiteiten en velden beschikbaar. Vul dit veld nu met de volgende code:  
  
<<template Template(source) format="json">>

  {

    "Signal": [

      {

        "Kind": "organization-signal",

        "Id": "<<=\message.ToUuid()>>",

        "Bron": {

          "ComponentId": "Organisatie",

          "Id": "<<=\organization>>"

        },

        "BronOmschrijving": "Een organisatie",

        "Message": "<<=\message>>",

        "IsLocked": false

      }

    ]

  }

<</template>>

## Fase 7: Update connector aanroepen (App developer)

Roep nu de connector aan voor elke organisatie die niet gevonden werd bij de kvk met behulp van onderstaande code

var putSignalUrl = FocusOAuthHandler.GetFocusUrl(context)

+ "/api/kvkchecker-organisatie-signaal";

await Log("Calling " + putSignalUrl);

await focusClient.PutAsJsonAsync(

putSignalUrl,

new {

  organization = organization.InstanceId,

    message = $"De organisatie {organization.Omschrijving} is niet bekend bij de KVK"

  });

Als je nu je script uitvoert verschijnen er signalen in Focus waarmee de klant zijn organisaties nu kan verbeteren.

## Extra opdracht

De Bronomschrijving staat nu op “Een organisatie” en deze tekst is ook zichtbaar bij het signaal in Focus bij bronnen. Verbeter dit.   
  
Discussie: Om een signaal te updaten, moet je dan PUT of POST gebruiken? [inspiratie](https://stackoverflow.com/questions/630453/what-is-the-difference-between-post-and-put-in-http)