# Workshop devdays 2022

In deze workshop zullen jullie een applicatie “KVK-checker” bouwen die koppelt met Focus (SB+). Om de workshop uit te kunnen voeren heb je Visual Studio of VSCode nodig. Omdat we beperkte tijd hebben, hebben we alvast wat bestanden voor je klaargezet.

## Fase 1: De webapplicatie

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

* Start een terminal (powershell of command prompt)
* Navigeer naar de folder die je gecloned hebt van github (devdays2022workshop)
* Voer het volgende commando uit:  
  **dotnet new web --name KVKChecker**
* Nu worden de bestanden die nodig zijn voor een .NET web applicatie voor je neergezet.
* Open nu Visual studio of VSCode, dit kan met het volgende commando:  
  **code .**

De volgende stap is het toevoegen van de landingpage. Dit is de pagina waarnaartoe Focus navigeert vanuit het “App Center”. Hierbij wordt de Focus-url doorgegeven. Open de gegenereerde **program.cs** file en voeg de volgende regel toe voor app.Run():

**LandingPage.Register(app);**

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. Omdat elke focus klant zijn eigen url heeft met daarin de tenant (accountant) en scope (klant) is het niet mogelijk om vooraf in de applicatie configuratie de url van Focus vast te leggen.

Voor nu hebben we even genoeg gedaan aan onze webapplicatie. Start de webapplicatie met **F5** of via **dotnet run** in de terminal. Je webapplicatie opent nu in een browser. Onthoud het poortnummer waarop de applicatie draait.

## Fase 2: De registratie in Focus

Focus kent het “App center” waar alle applicaties geregistreerd staan die gekoppeld kunnen worden. Deze lijst wordt door AFAS beheerd, maar gelukkig kunnen we via een “AFAS import” hier jullie KVKChecker aan toevoegen.

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

* Verzin een geheime ClientSecret om te voorkomen dat iemand anders je koppeling misbruikt
* Wijzig het poortnummer in de ClientRedirectUrl en ClientLandingPageUrl naar het poortnummer van jouw applicatie
* Sla deze wijzigingen op

Jullie groep heeft een url gekregen waarop jullie focus applicatie draait. Open deze in een browser, open 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 als het goed is dat jouw import gelukt is. Ga nu via **F3** naar App Center en klik op jouw applicatie. Aan het einde van de url staat nu 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.

## Fase 3: Autoriseren

Nu gaan we de autorisatie toevoegen aan de 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***", "***clientSecret***");**

De **FocusAuthorization.cs** class staat in de folder WorkshopLibrary. Deze bevat gebruikelijke .NET OAuth code die nodig is voor de autorisatie met Focus.

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 voor app.Run()

RunPage.Register(app);

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

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

  await Task.Delay(1000);

  await Log("Done");

});

Deze pagina’s zijn geautoriseerd, dus zodra de gebruiker hier naartoe navigeert, wordt eerst gecontroleerd of er wel gekoppeld is met Focus. Als dat niet het geval is moet je via de appcenter in Focus de koppeling leggen.

Start nu de web applicatie opnieuw en ga verder vanaf de landing page die je nog open had staan. Je komt nu als het goed is op de run pagina terecht waar je ziet dat je script hierboven uitgevoerd wordt.

## Fase 4: Gegevens ophalen uit Focus

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. Normaalgesproken zou de partner dit niet zelf doen, maar doet het Focus team dit (Christiaan Lankman om precies te zijn).

We doen dit als volgt:

* Ga in Focus (via **F3**) naar “Applicatie onderdelen”
* Doe hier Nieuw 🡪 Getconnector
* Vul het veld **url** met **kvkchecker-organisaties**
* Open de **Query voor gegevens**
* Gebruik gegevensverzameling **Organisatie**
* Voeg de velden toe uit onderstaand screenshot
* Sla op
* Wijzig nu het laatste deel van de url in /apidocs en nu zie je als het goed is jouw koppeling terug en je kunt hem ook interactief aanroepen

Graphical user interface, application

Description automatically generated

## Fase 5: Analyse script maken

Nadat de koppeling met focus tot stand is gebracht heeft je webapplicatie een access\_token gekregen waarmee de Focus API’s aangeroepen worden. Normaliter wordt deze opgeslagen in de database van de webapplicatie, maar door de manier waarop je webapplicatie nu geconfigureerd is staan deze gegevens in een cookie.

Hieronder staan wat stukken code waarmee je met behulp van de access\_token de get-connector van Focus kunt aanroepen. Gebruik deze om de lijst met organisaties te downloaden.

Gebruik vervolgens onderstaande stukken code om organisaties die een kvk nummer hebben op te zoeken bij de Kvk. De apikey van de kvk die je nodig hebt heb je via mail van ons ontvangen.

// 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.GetAfasBaseUrl(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));

// 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);

  var name = result!.naam;

  if (result!.handelsnamen != null)

  {

    name = string.Join(  
 ", ",   
 result!.handelsnamen!.OrderBy(h => h.volgorde).Select(h => h.naam)  
 );

  }

  await Log("Found " + name);

}

catch (HttpRequestException e)

{

  if (e.StatusCode == System.Net.HttpStatusCode.NotFound)

  {

    await Log("Not Found");

  }

  else

  {

    throw;

  }

}

## Fase 6: Update connector

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 we via deze update-connector willen versturen willen we de volgende gegevens laten 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. In de devdays2022workshop staat het json-schema die de json hierboven beschrijft.

Graphical user interface, text, application

Description automatically generated

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>>

TODO: extra opdracht bronomschrijving fixen