

UCI OAuth

Programmering talen: C#, Java, Node.js, Python, PHP, Ruby,

Complexiteit: *) +++++

Verwachte implementatie tijd: 8 u

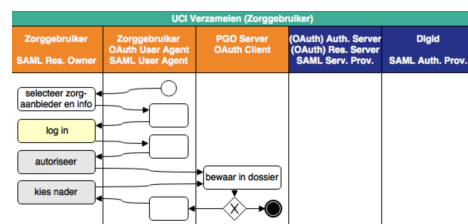
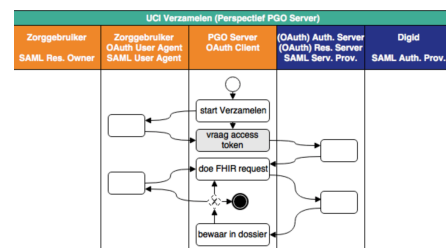
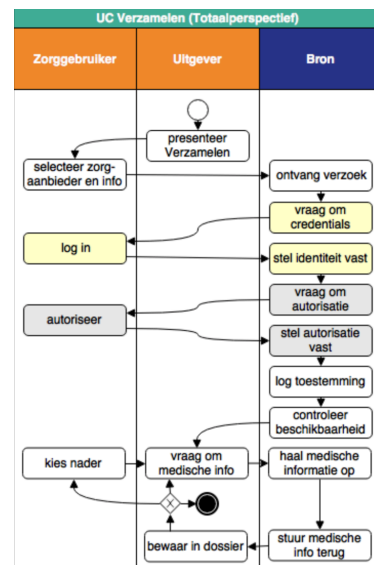
Oplevering: Coding Standard volgens Blauwdruk Design Template (zie Appendix)

Deployment: Library in een van Programmering talen

Testing: WS direct access?

Opdracht context:

- De PGO GW start de flow door in de *User Agent* van de *Zorggebruiker* de mogelijkheid te presenteren om een bepaalde *Gegevensdienst* bij een zekere *Zorgaanbieder* te verzamelen. Uit de *Zorgaanbiederslijst* weet de PGO GW welke *Gegevensdiensten* voor een *Zorgaanbieder* beschikbaar zijn. In de local state-parameter geeft de PGO GW informatie mee aan de ZA GW, waaraan de PGO GW later, bij de redirect, precies weet bij welk verzoek de authorization code hoort.
- De *Zorggebruiker* maakt zijn selectie en laat de *OAuth User Agent* een verzamel-verzoek sturen naar de ZA GW. Het adres van het authorization endpoint komt uit de ZAL. De redirect URI geeft aan waarnaartoe de ZA GW (als *OAuth Authorization Server*) de *OAuth User Agent* verderop moet redirecten (met de authorization code).
- Daarop begint de ZA GW de OAuth-flow (in zijn rol als *OAuth Authorization Server*) door een sessie te creëren.
- De ZA GW controleert alvast of de *Zorgaanbieder* voor de betreffende *Gegevensdienst* überhaupt gezondheidsinformatie van die *Persoon* beschikbaar heeft.
- Zo ja, dan presenteert de ZA GW (nog steeds als *OAuth Authorization Server*) via de browser aan *Zorggebruiker* de vraag of laatstgenoemde hem toestaat de gevraagde persoonlijke gezondheidsinformatie aan de PGO GW (als *OAuth Client*) te sturen. Onder het flow-diagram staat gespecificeerd welke informatie, waarvandaan, de *OAuth Authorization Server* verwerkt in de aan *Zorggebruiker* voor te leggen autorisatievraag.
- Bij akkoord logt de ZA GW dit als toestemming, genereert een authorization code en stuurt dit als ophaalbewijs, door middel van een browser redirect met de in stap 1 ontvangen redirect URI, naar de PGO GW. De ZA GW stuurt daarbij de local state-informatie mee die hij in de eerste stap van de PGO GW heeft gekregen. Laatstgenoemde herkent daaraan het verzoek waarmee hij de authorization code moet associëren.
- De PGO GW vat niet alleen deze authorization code op als ophaalbewijs, maar leidt er ook van af dat de toestemming is gegeven en logt deze toestemming.
- Met dit ophaalbewijs wendt de PGO GW zich weer tot de ZA GW, maar nu zonder tussenkomst van de *OAuth User Agent*, voor een access token.
- Daarop genereert de ZA GW een access token en stuurt deze naar de PGO GW.
- Nu is de PGO GW gereed om het verzoek om de gezondheidsinformatie naar de ZA GW te sturen. Het adres van het resource endpoint haalt hij uit de ZAL. Hij plaatst het access token in het bericht en zorgt ervoor dat in het bericht geen BSN is opgenomen.



Opdracht beschrijving:

Test case 1:

UCI Verzamelen 1	De <i>Zorggebruiker</i> maakt zijn selectie en laat de <i>OAuth User Agent</i> een verzamel-verzoek sturen naar de <i>ZA GW</i> . Het adres van het authorization endpoint komt uit de <i>ZAL</i> . De redirect URI geeft aan waarnaartoe de <i>ZA GW</i> (als <i>OAuth Authorization Server</i>) de <i>OAuth User Agent</i> verderop moet redirecten	Conform OAuth 2.0
---------------------	--	-----------------------------------

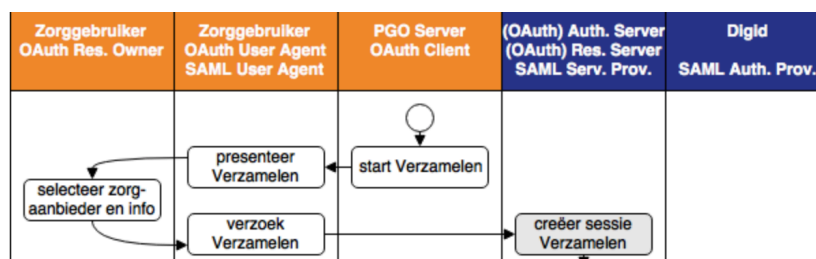
Exceptie handling

UCI Verzamelen 1	Authorization Server vindt het ontvangen verzoek ongeldig.	Authorization Server informeert <i>Zorggebruiker</i> over deze uitzondering. <i>Zorggebruiker</i> laat <i>PGO Server</i> de flow afbreken.	conform OAuth 2.0-specificatie, par. 4.1.2.1, error code invalid_request, met in de error description de oorzaak
---------------------	--	--	--

UCI Verzamelen 2	<i>Authorization Server</i> kan de identiteit van de <i>Zorggebruiker</i> niet vaststellen.	<i>Authorization Server</i> informeert <i>PGO Server</i> over deze uitzondering. <i>PGO Server</i> informeert daarop <i>Zorggebruiker</i> hierover.	conform OAuth 2.0-specificatie, par. 4.1.2.1, error code unauthorized_client
---------------------	---	---	--

UCI Verzamelen 3	<i>Authorization Server</i> stelt vast dat van <i>Persoon</i> bij <i>Zorgaanbieder</i> geen gezondheidsinformatie voor die <i>Gegevensdiens</i> beschikbaar is.	<i>Authorization Server</i> informeert <i>PGO Server</i> over deze uitzondering. <i>PGO Server</i> informeert daarop <i>Zorggebruiker</i> hierover.	conform OAuth 2.0-specificatie, par. 4.1.2.1, error code access denied, met in de error description "No such resources."
---------------------	---	---	--

Uitleg



```
URI - /oauth?response_type=code&client_id=CLIENT_ID
&redirect_uri=REDIRECT_URI&scope=?&state=1233xxx
```

- URI – provided from the definition in the ZAL
- client_id=CLIENT_ID – PGO NODE Hostname
- redirect_uri=REDIRECT_URI - Indicates the URI to return the user to after authorization is complete, such as `http://authorize`
- scope= 1... (zie <https://afsprakenstelsel.medmij.nl/display/PUBLIC/Gegevenscatalogus>) - ONE scope value indicating which parts of the user's account you wish to access
- state=1234zyx - A random string generated by your application, which you'll verify later
- **Note:** *Noch in de authorization code, noch in het access token wordt betekenisvolle informatie opgenomen. Dat zorgt er ook voor dat er een minimale afhankelijkheid wordt gecreëerd tussen de PGO GW en de ZA GW, zodat principe P1 maximaal wordt nageleefd en interne complexiteit en implementatiekeuzes van de ZA GW niet doorschemeren in, of invloed uitoefenen op, de implementatie van de PGO GW*
- **Note:** De OAuth-rol Client biedt aan de Authorization Servers slechts redirect URI's aan die volledig (full) zijn én verwijzen naar een HTTPS-beschermd endpoint. Authorization Servers redirecten niet naar een URI die niet aan deze eisen voldoet.

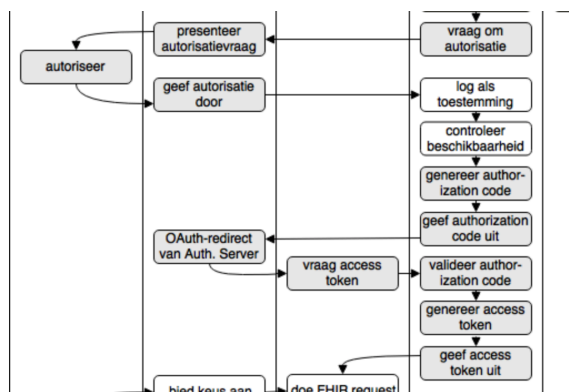
Test case 2:

UCI Verzamelen 1	<p>De ZA GW controleert alvast of de <i>Zorgaanbieder</i> voor de betreffende <i>Gegevensdienst</i> überhaupt gezondheidsinformatie van die <i>Persoon</i> beschikbaar heeft.</p> <p>Zo ja, dan presenteert de ZA GW (nog steeds als <i>OAuth Authorization Server</i>) via de browser aan <i>Zorggebruiker</i> de vraag of laatstgenoemde hem toestaat de gevraagde persoonlijke gezondheidsinformatie aan de PGO GW (als <i>OAuth Client</i>) te sturen. Onder het flow-diagram staat gespecificeerd welke informatie, waarvandaan, de <i>OAuth Authorization Server</i> verwerkt in de aan <i>Zorggebruiker</i> voor te leggen autorisatievraag.</p> <p>Bij akkoord logt de ZA GW dit als toestemming, genereert een authorization code en stuurt dit als ophaalbewijs, door middel van een browser redirect met de in stap 1 ontvangen redirect URI, naar de PGO GW. De ZA GW stuurt daarbij de local state-informatie mee die hij in de eerste stap van de PGO GW heeft gekregen. Laatstgenoemde herkent daaraan het verzoek waarmee hij de authorization code moet associëren.</p>	Conform OAuth 2.0
---------------------	--	--------------------------------------

Exceptie handling

UCI Verzamen 4	De autorisatievraag wordt ontkennend beantwoord.	ZA GW logt de afwijzing en informeert PGO GW hierover. Uitgever informeert daarop Zorggebruiker hierover.	conform OAuth 2.0-specificatie, par. 4.1.2.1, error code access denied, met in de error description "Authorization denied."
UCI Verzamen 5	ZA GW kan de autorisatie niet vaststellen.	ZA GW informeert PGO GW over deze uitzondering. PGO GW informeert daarop Zorggebruiker hierover.	conform OAuth 2.0-specificatie, par. 4.1.2.1, error code access denied, met in de error description "Authorization failed."
UCI Verzamen 6	De validatie van de authorization code door ZA GW faalt.	ZA GW informeert PGO GW over deze uitzondering. PGO GW informeert daarop Zorggebruiker hierover.	conform OAuth 2.0-specificatie, par. 5.2, error code invalid_grant

Uitleg



https://example-app.com/cb?code=AUTH_CODE_HERE&state=1234zyx

- **code** - The server returns the authorization code in the query string
- **state** - The server returns the same state value that you passed

Token Exchange

PGO server exchanges het auth code voor een access token:

```
POST https://api.authorization-server.com/token
grant_type=authorization_code& code=AUTH_CODE_HERE&
```

```
redirect_uri=REDIRECT_URI&    client_id=CLIENT_ID&  
client_secret=CLIENT_SECRET
```

Opdracht resources:

Opdracht acceptatie:

Test case 1: Functionele API coverage

Test case 2: Unit Test executie voor elke API validatie

Test case 3: Integratie Test Met PGO OAuth Client