Table of Contents

[1. Detailed Design 2](#_Toc25583009)

[1.1 Overview 2](#_Toc25583010)

[1.2 Design Specification for Configured Provider & Consumer 2](#_Toc25583011)

[1.3 Design Specification for Dynamic Client Registration 4](#_Toc25583012)

[1.4 Design Specification for Authorization Server 6](#_Toc25583013)

[1.5 Design Specification for Open Badges Assertions 8](#_Toc25583014)

[1.6 Design Specification for Open Badges Profile 8](#_Toc25583015)

# Detailed Design

## Overview

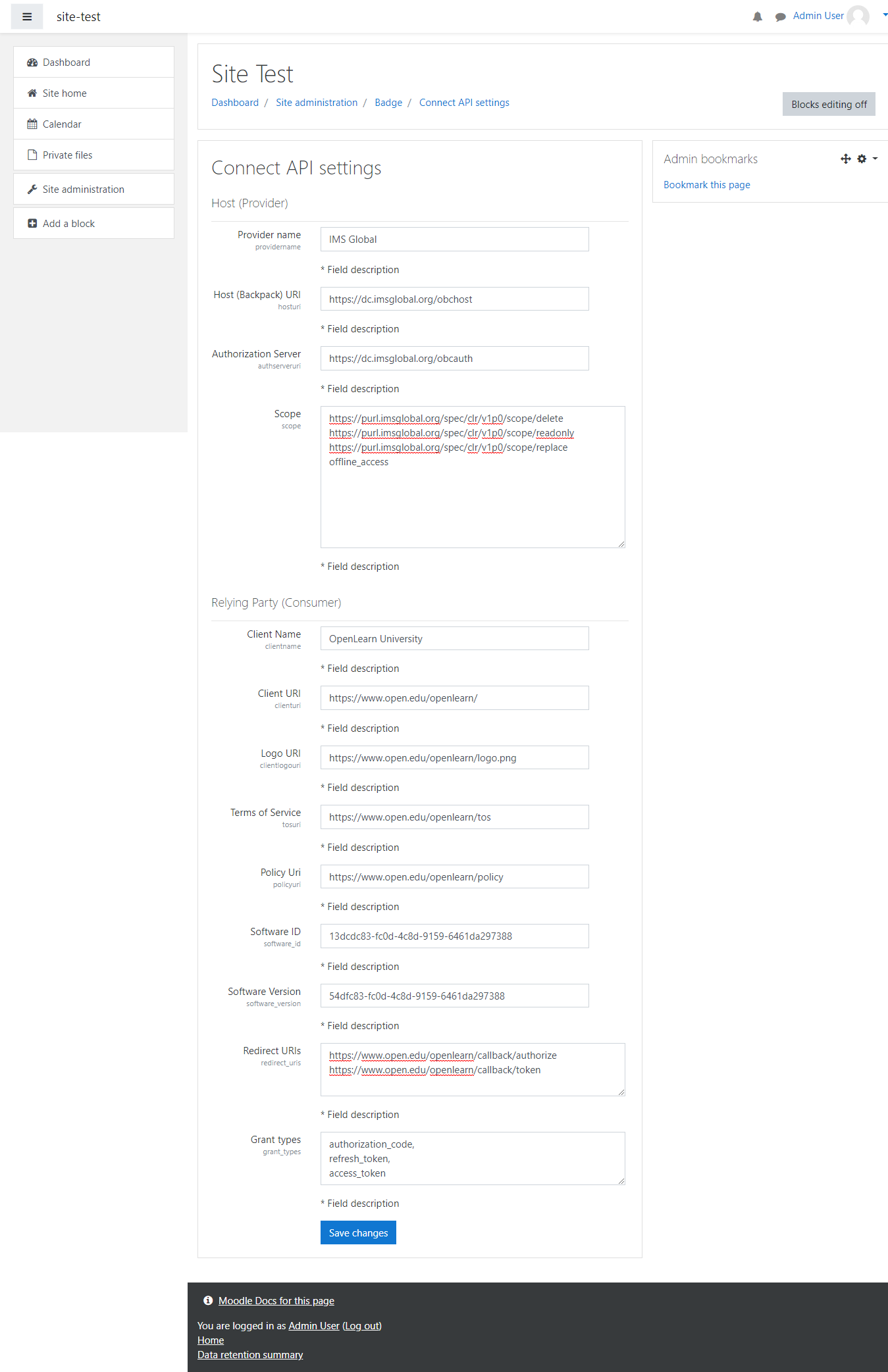
This section of the document will explain how to get started with a relying party application for Badge Connect. Moodle application needs to go through several common processes before the workflows for their specific needs diverge. Here we will explain these steps, including:

1. Backpack (**Host**) Selection: Configured Provider & Consumer
2. Dynamic Client Registration
3. Authorization server: Authorization Code & Token Exchange & Refresh token
4. Access Open Badges OpenAPI Assertions
5. Access Open Badges OpenAPI Profile

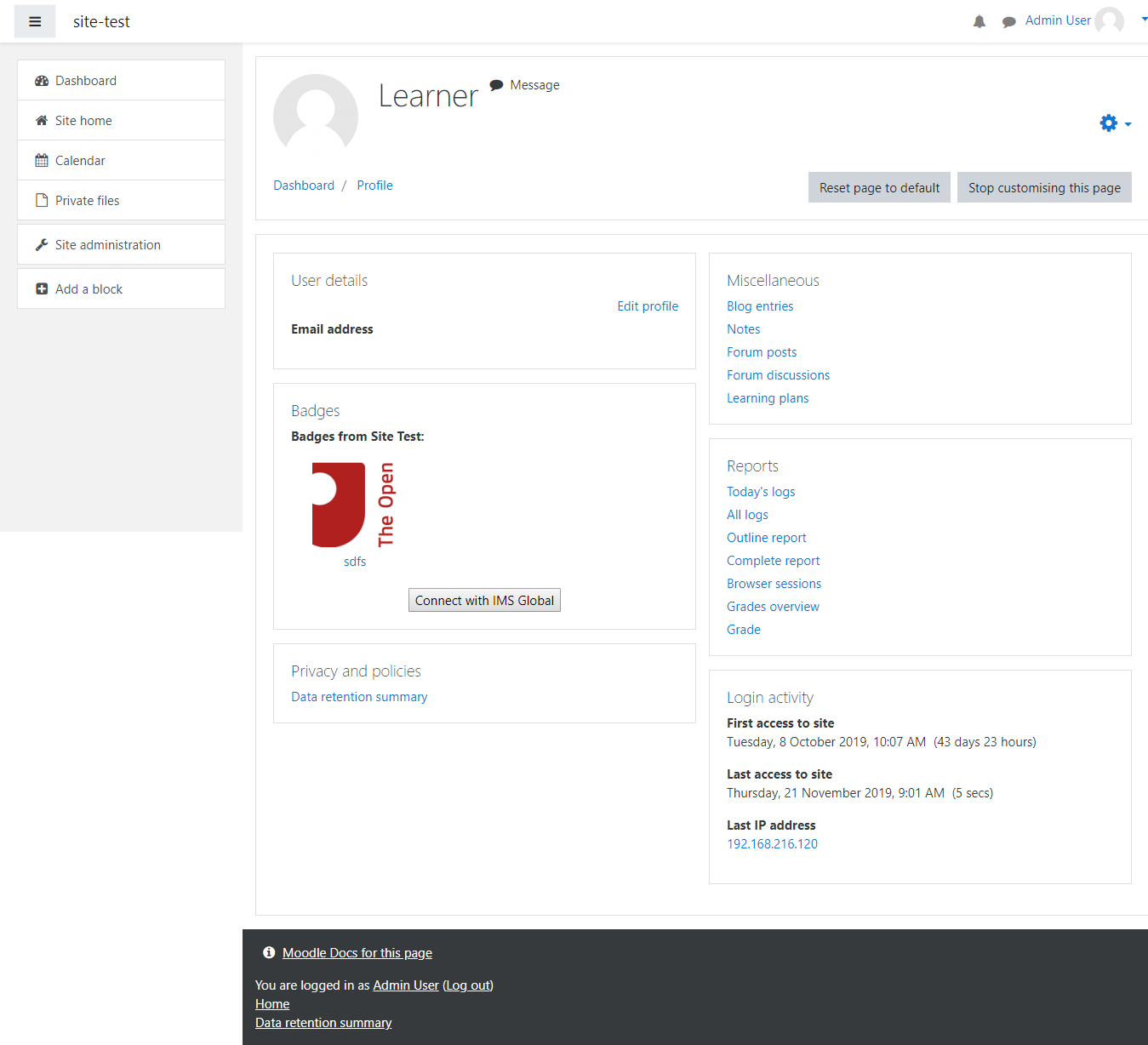
## Design Specification for Configured Provider & Consumer

In Moodle 3.7 onwards, settings in the Site administration enable admins to configure the Open Badges 2.1

Go to *Site administration > Badges > Connect API settings (figure 1)*



*Figure 1 – Badges 2.1 settings*



*Figure 2 – Backpack selection (Connect with IMS global)*

## Design Specification for Dynamic Client Registration

* A user starts on an issuing application and earns a badge. For example, they successfully complete a quiz.
* An Open Badges Assertion is published, linked to a BadgeClass and Issuer Profile, according to the Open Badges 2.0 Specification.
* The user is offered the opportunity to claim the badge by accepting it into a backpack of their choice.
* A choice of backpacks is offered, including perhaps some suggestions known to the Moodle application already as well as an advanced option to select a domain of their choice. (Figure 2)
* If this is the first time the Moodle application has encountered the backpack on the chosen domain, information about the backpack is read from the backpack’s manifest file, and a server to server connection is established using Dynamic Client Registration.

An example registration request may look like:

POST /obcauth/connect/register HTTP/1.1   
Host:dc.imsglobal.org   
Content-Type:application/json   
cache-control:no-cache

{   
   "client\_name":"Open Learn Univeristy",  
   "client\_uri":"https://www.open.edu/openlearn",  
   "logo\_uri":"https://www.open.edu/openlearn/logo.png",  
   "tos\_uri":"https://www.open.edu/openlearn/tos",  
   "policy\_uri":"https://www.open.edu/openlearn/policy",  
   "software\_id":"13dcdc83-fc0d-4c8d-9159-6461da297388",  
   "software\_version":"54dfc83-fc0d-4c8d-9159-6461da297388",  
   "redirect\_uris":[   
      "https://www.open.edu/openlearn/callback/authorize",  
 “https://www.open.edu/openlearn/callback/token”  
   ],  
   "token\_endpoint\_auth\_method":"client\_secret\_basic",  
   "grant\_types":[   
      "authorization\_code",  
      "refresh\_token",  
      "access\_token"  
   ],  
   "response\_types":[   
      "code"  
   ],  
   "scope":"https://purl.imsglobal.org/spec/clr/v1p0/scope/delete https://purl.imsglobal.org/spec/clr/v1p0/scope/readonly https://purl.imsglobal.org/spec/clr/v1p0/scope/replace offline\_access"  
}

Response with a successful result:

HTTP/1.1 201 Created

Content-Type: application/json

Cache-Control: no-store

Pragma: no-cache

{

"client\_id": **3ae65fb887247746**

"client\_secret": **924a54cd6f238118**

"client\_id\_issued\_at": 1558280111,

"client\_secret\_expires\_at": 1558290111,

"client\_name":"Open Learn Univeristy",

"client\_uri":"https://www.open.edu/openlearn",

"logo\_uri":"https://www.open.edu/openlearn/logo.png",

"tos\_uri":"https://www.open.edu/openlearn/tos",

"policy\_uri":"https://www.open.edu/openlearn/policy",

"software\_id":"13dcdc83-fc0d-4c8d-9159-6461da297388",

"software\_version":"54dfc83-fc0d-4c8d-9159-6461da297388",

"redirect\_uris":[

"https://www.open.edu/openlearn/callback/authorize",

"https://www.open.edu/openlearn/callback/authorize",  
 “https://www.open.edu/openlearn/callback/token”  
 ],

"token\_endpoint\_auth\_method":"client\_secret\_basic",

"grant\_types":[

"authorization\_code",

"refresh\_token",

"access\_token"

],

"response\_types":[

"code"

],

"scope":"https://purl.imsglobal.org/spec/clr/v1p0/scope/delete https://purl.imsglobal.org/spec/clr/v1p0/scope/readonly https://purl.imsglobal.org/spec/clr/v1p0/scope/replace offline\_access"

}

## Design Specification for Authorization Server

* The “OAuth Dance” is initiated, directing the user to the backpack for authorization.
* The user accepts the Moodle app’s requested permissions from within the backpack interface. (figure 3 & 4 – requested permissions & approve permissions)
* The user is redirected back to the Moodle application with a code in the query parameters portion of the URL. (yellow highlight in response result)
* The Moodle application exchanges the code for an access token (including a refresh token for later).
* After some time, typically 1 hour, the access token expires. The next time the Moodle app needs to contact the backpack on the behalf of this user, it will need to refresh its access token. At the time of its issue, the Moodle application also obtained a refresh token. The refresh token is used to exchange the expired access token for a new one with an expiration date in the future

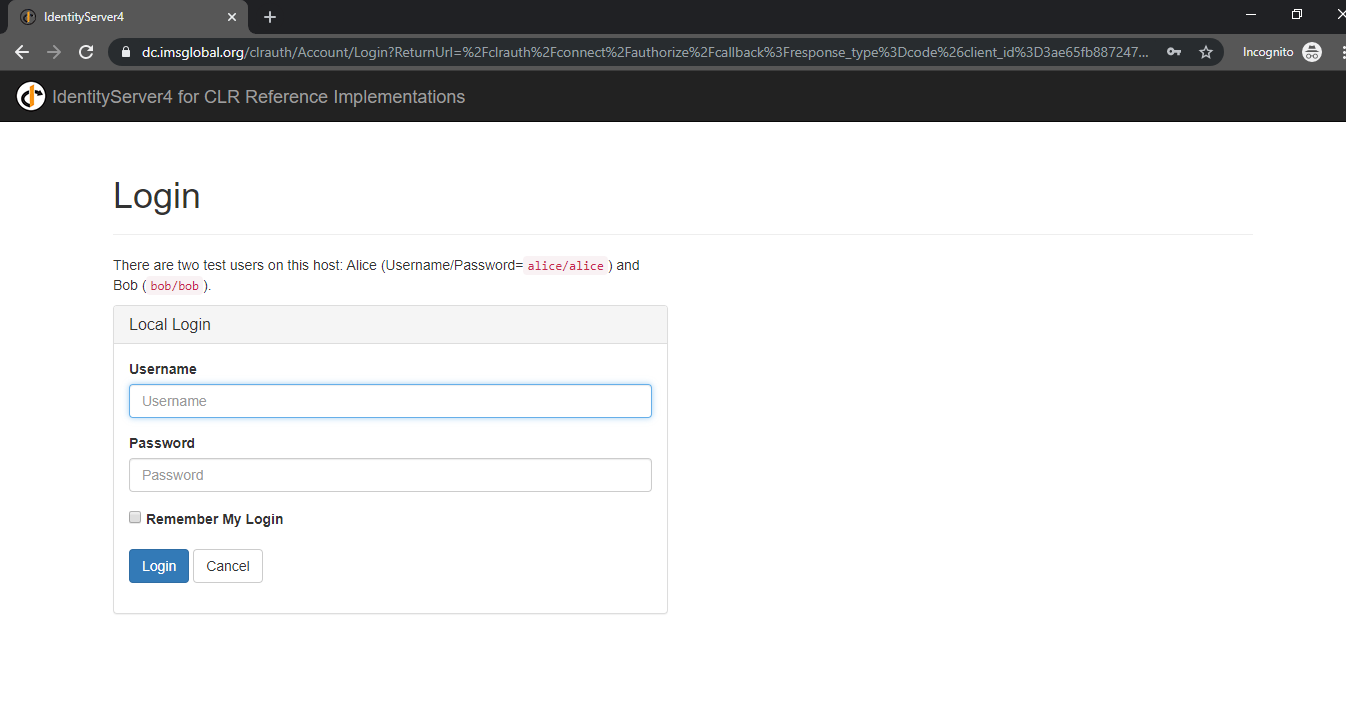


Figure 3 – requested permissions

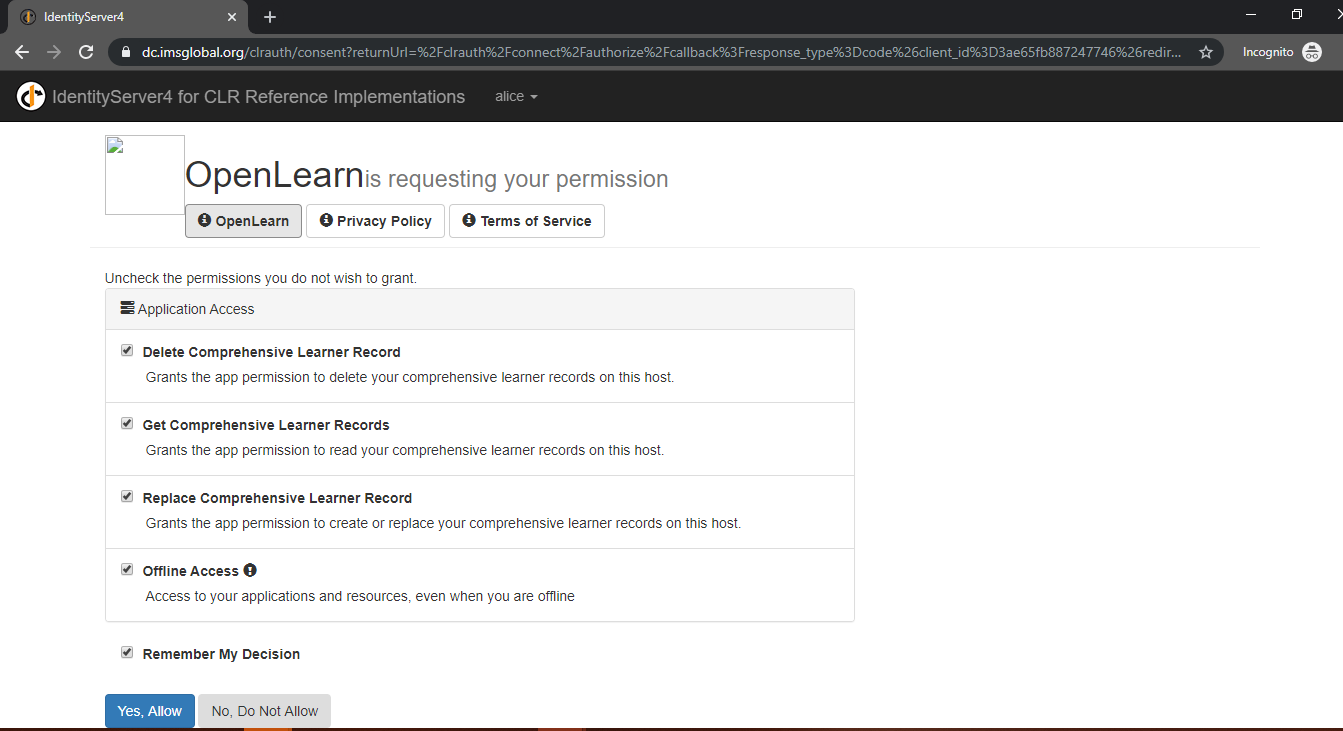


Figure 4 – User accepts permission

Example authorization request:

The HTTP POST request MUST include a Basic authorization header with the **client\_id** and **client\_secret** provided by the response to the registrationUrl endpoint.

GET /obcauth/connect/authorize?response\_type=code&amp; client\_id=**3ae65fb887247746**&amp; redirect\_uri= https://www.open.edu/openlearn/callback/token&amp; state=aAbBcC&amp; scope=https://purl.imsglobal.org/spec/clr/v1p0/scope/delete https://purl.imsglobal.org/spec/clr/v1p0/scope/readonly https://purl.imsglobal.org/spec/clr/v1p0/scope/replace offline\_access&amp; code\_challenge=\_QFaSRqoOx486Dm7Otiwhb-EhzBfrQLSogvGvqwcRkY&amp; code\_challenge\_method=S256 HTTP/1.1

Host: dc.imsglobal.org

cache-control: no-cache

Response with a successful result:

POST /obcauth/connect/token HTTP/1.1

Host: dc.imsglobal.org

cache-control: no-cache

grant\_type=authorization\_code&code=f79609d880067d0a4687f208c882f21bf65917e54cb8beacd4e5a0f4cdf53a63&scope=https%3A%2F%2Fpurl.imsglobal.org%2Fspec%2Fclr%2Fv1p0%2Fscope%2Fdelete+https%3A%2F%2Fpurl.imsglobal.org%2Fspec%2Fclr%2Fv1p0%2Fscope%2Freadonly+https%3A%2F%2Fpurl.imsglobal.org%2Fspec%2Fclr%2Fv1p0%2Fscope%2Freplace+offline\_access&redirect\_uri= https://www.open.edu/openlearn/callback/authorize&code\_verifier=m-6yj0HcC7Uo1f-KPqDydbRB0giV74GQX\_7yKivXsZs

## Design Specification for Open Badges Assertions

* The learner makes a call to the *POST /assertions* endpoint to add the awarded badge to the user’s backpack. The backpack responds with a successful result.
* The learner makes a call to the GET */assertions* endpoint to read the badges in a user’s backpack. The backpack responds with a successful result.
* The Moodle application may now push badges to the user’s backpack at any time.

APIs Endpoint

<https://dc.imsglobal.org/obchost/swagger/index.html>

## Design Specification for Open Badges Profile

* The learner application makes a call to the *POST /profile* endpoint to add their profile to the user’s backpack. The backpack responds with a successful result.
* The learner makes a call to the GET */profile* endpoint to read the badges in a user’s backpack. The backpack responds with a successful result.
* The Moodle application may now push badges to the user’s backpack at any time.

APIs Endpoint

<https://dc.imsglobal.org/obchost/swagger/index.html>