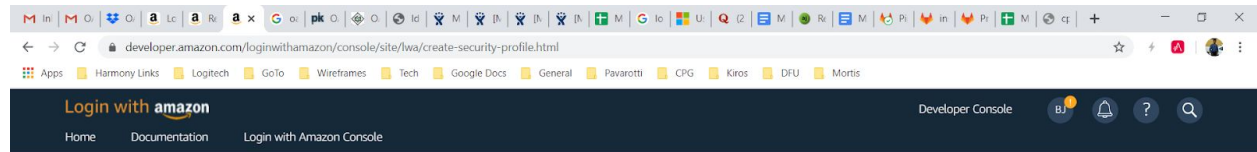


<https://developer.amazon.com/docs/login-with-amazon/minitoc-lwa-websites.html>

Register the App in Amazon Developer Console



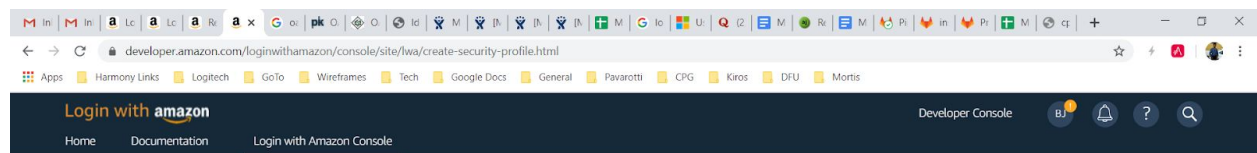
Security Profile Management

Name your new Security Profile

Choose a name for this security profile. You can create multiple security profiles. You will associate a security profile with one or more apps. Apps that use the same security profile can share some types of data (for example, a "My App - Free" and a "My App - HD" could share data). For a shared security profile, choose a name that applies to all the apps that will use it (for example, "My App profile"). [Learn More](#)

* Indicates a required field

Security Profile Name *	<input type="text" value="Boney-App"/>
Security Profile Description *	<input type="text" value="Boney's website"/>
Consent Privacy Notice URL *	<input type="text" value="https://www.codingsoldier.com"/>
Consent Logo Image	<div>UPLOAD IMAGE</div>



Login with Amazon

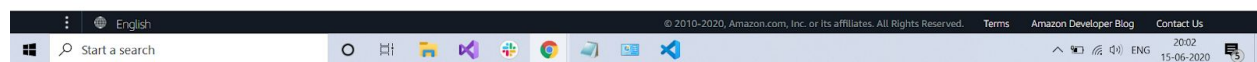
Login with Amazon allows users to login to registered third party websites or apps ('clients') using their Amazon user name and password. Clients may ask the user to share some personal information from their Amazon profile, including name, email address, and zip code. To get started, select an existing Security Profile or create a new Security Profile. [Learn More](#)

[Create a New Security Profile](#)

✔ Login with Amazon successfully enabled for Security Profile. Click to manage Security Profile.

Login with Amazon Configurations

Security Profile Name	OAuth2 Credentials	Manage
Boney-App	Client ID: amzn1.application-oa2-client.b37e161d1c4d457ebacc44d9dcd00ad8 Client Secret: 3ce03d1b02c3f562fd1323b044ec92102fa49da49488fb9c2236aa857c74ae26	<input type="button" value="⚙"/>



Note the generated ClientId and ClientSecret

ClientID:amzn1.application-oa2-client.b37e161d1c4d457ebacc44d9dcd00ad8

ClientSecret:3ce03d1b02c3f562fd1323b044ec92102fa49da49488fb9c2236aa857c74ae26

Add Allowed Origins and Allowed Return URLs

In the web settings under Security Profile, I add the Allowed Origins and Allowed Return URLs. The origin is an ngrok domain, which means it is localhost available in public via the ngrok domain. (See the [Ngrok settings](#) below)

Security Profile Management



Boney-App - Security Profile

General

Web Settings

Android/Kindle Settings

iOS Settings

TVs and Other Devices Settings

To use Login with Amazon with a website, you must specify either an allowed JavaScript origin (for the Implicit grant) c

Client ID	amzn1.application-oa2-client.b37e161d1c4d457ebacc44d9dcd00ad8
-----------	---

Client Secret	Show Secret
---------------	-------------

Allowed Origins ?	https://ba136f528fcf.ngrok.io
-------------------	-------------------------------

Allowed Return URLs ?	https://ba136f528fcf.ngrok.io/handle_callback
-----------------------	---

Ngrok settings

```
C:\Windows\system32\cmd.exe - ngrok http 3000
```

ngrok by @inconshreveable

```
Session Status      online
Session Expires    7 hours, 59 minutes
Version             2.3.35
Region             United States (us)
Web Interface       http://127.0.0.1:4040
Forwarding          http://ba136f528fcf.ngrok.io -> http://localhost:3000
Forwarding          https://ba136f528fcf.ngrok.io -> http://localhost:3000

Connections        ttl    opn    rt1    rt5    p50    p90
                   0      0      0.00   0.00   0.00   0.00
```

So I have a client and a server.

Client is the index.html, served by the server running in Node.js.

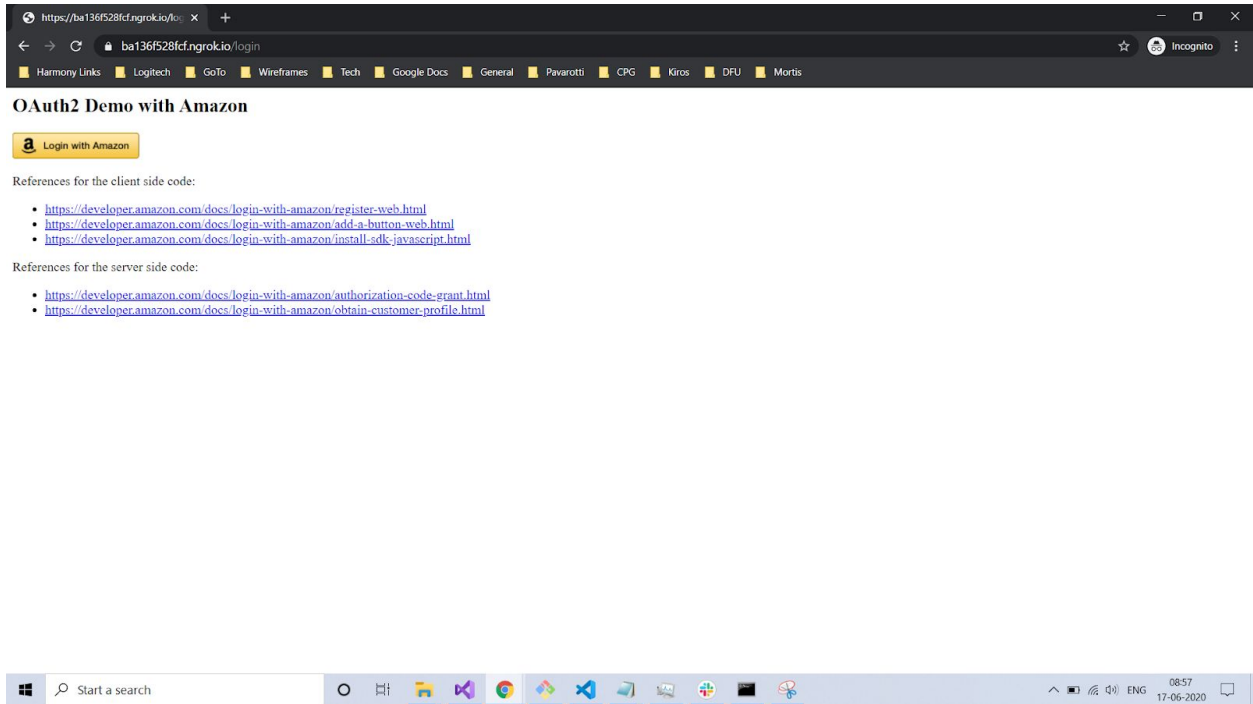
On to the index.html page, we add the html/javascript provided in the documentation [here](#) and [here](#). Or you can directly the index.html file from the code I have in [GitHub](#).

Note that I have set some additional properties on the “options” object. I have made the **response_type** to “code”. By default the value is “implicit”.

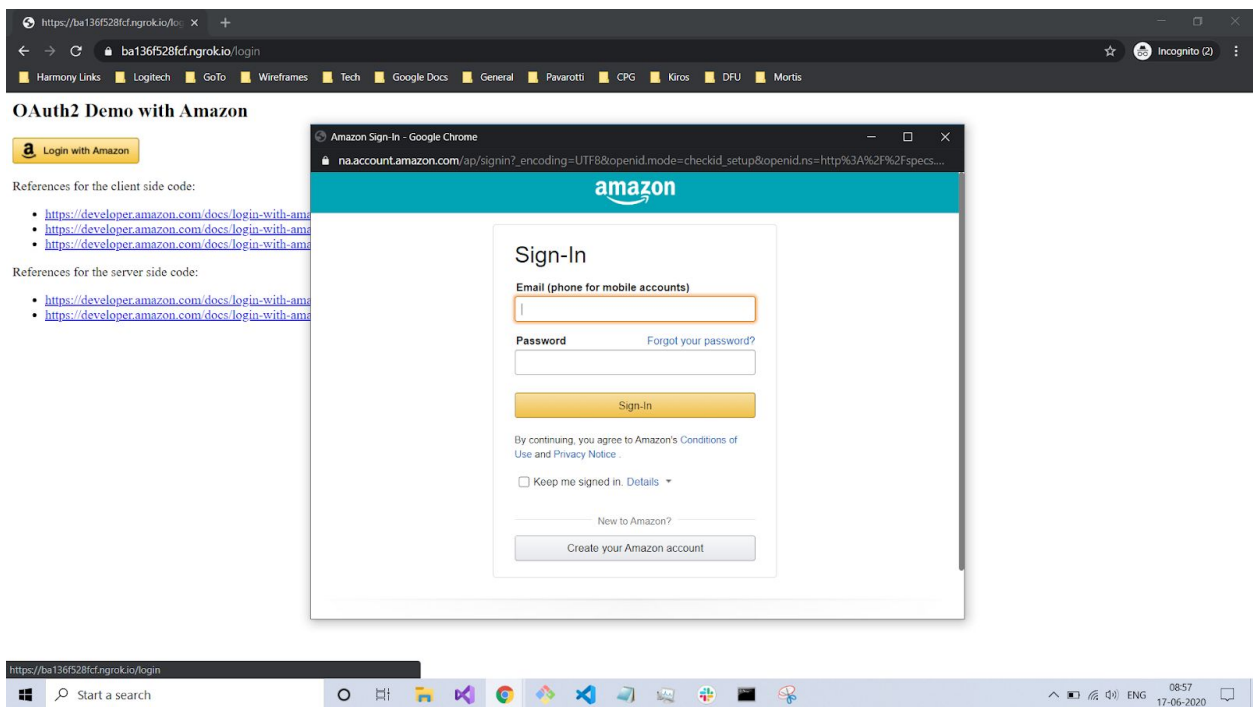
- If value is ‘implicit’, we get the access_token in the callback.
- If the value is ‘code’, in the callback we get the code and then we need to get the access_token using the code. And this is done on the server side. So access_token is not exposed in the client. An extra step for securing the access_token.

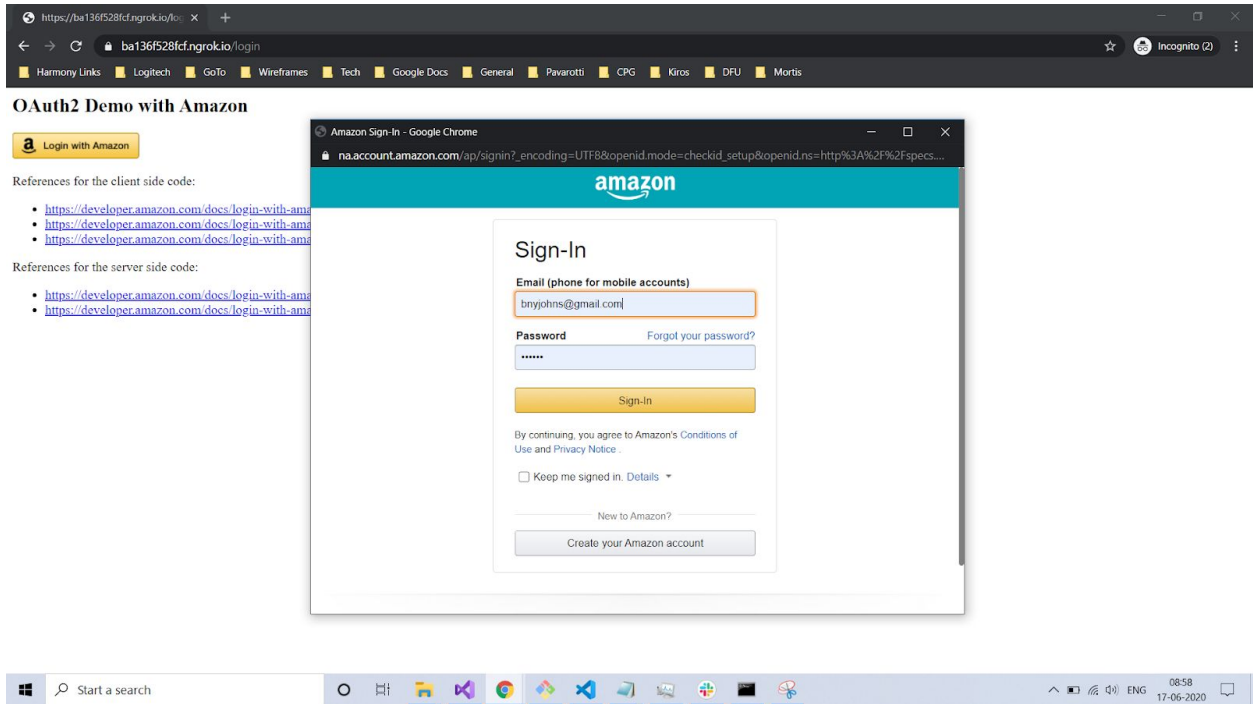
Flow

1. Start the server on port 3000
node index.js
2. Add the ngrok entry
ngrok http 3000
3. Browse to: <https://ba136f528fcf.ngrok.io/login> (replace domain with the ngrok domain that you get when you run the ngrok command)
4. index.html will be served by the server
5. Click on “Login with Amazon” button

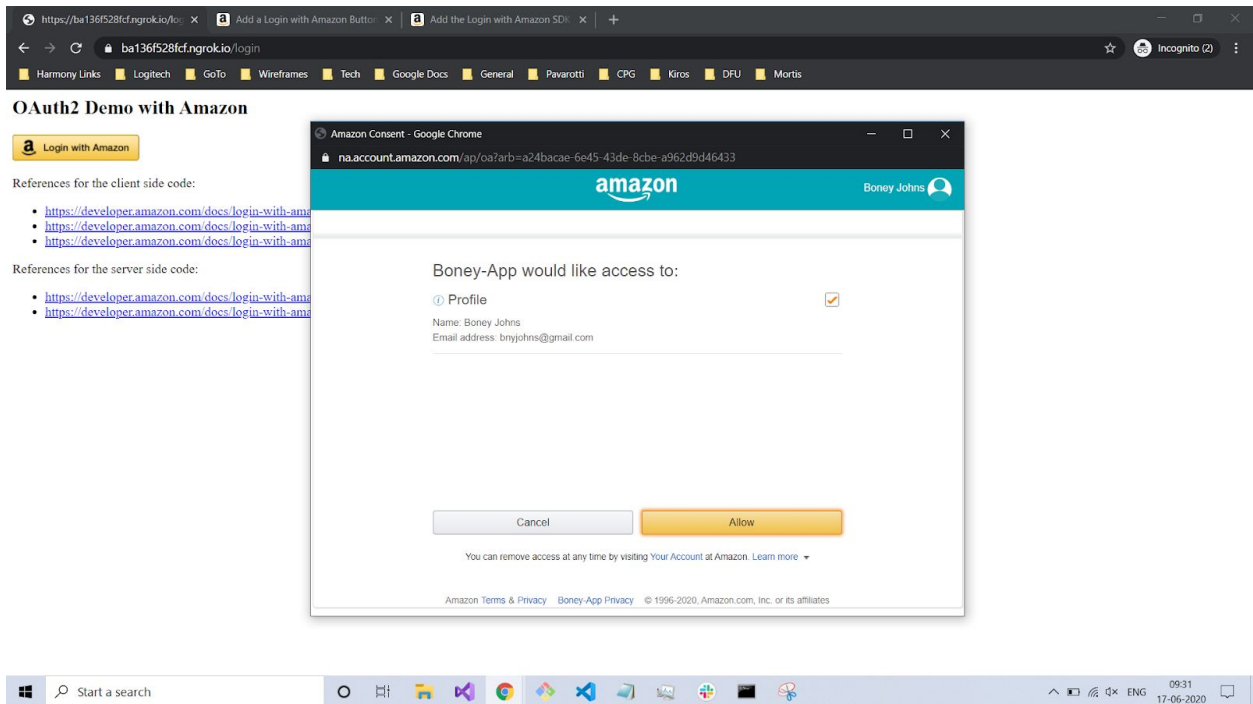


6. It will launch a popup asking for credentials if not already logged into Amazon





7. Now Popup will ask for consent from user to authorize app to access user details from Amazon



8. Clicking Allow will make a get call to:
https://ba136f528cf.ngrok.io/handle_callback?code=ANurVMVaSOlcxiFFoncD&scope=profile with the authorization code in the query string.
9. In the server side,
 - Parse the code from the query string of the request
 - Get access_token using the Authorization code(code)
 - Get the profile(resource) using the access_token
 - Return the name and email(in resource) in the response to the browser

