

**AUTH0 IN UMBUY**

Auth is used for connecting to a robust, simple and secure HTTPS API is key to allowing seamless integration with other tools and experiences. **Auth0** allows you to authenticate and authorize apps and APIs with any identity provider running on any stack on any device or cloud.

We have used Auth0 in our web platform as well the android platform to:

* Help new users sign up and use the related information to create a new profile using their umnet (university) email
* Help users log in and out using the credentials that they had signed up with
* Use the related data to create a profile page for users so that users can keep track of the ads they have created and edit or delete accordingly

**Using Auth0**

Got to [auth0.com](https://auth0.com/) using your web browser

Sign in using: [munsifb@yahoo.com](mailto:munsifb@yahoo.com)

Password: Team6best

Dashboard: indicates the users that are allowed to login and their passwords

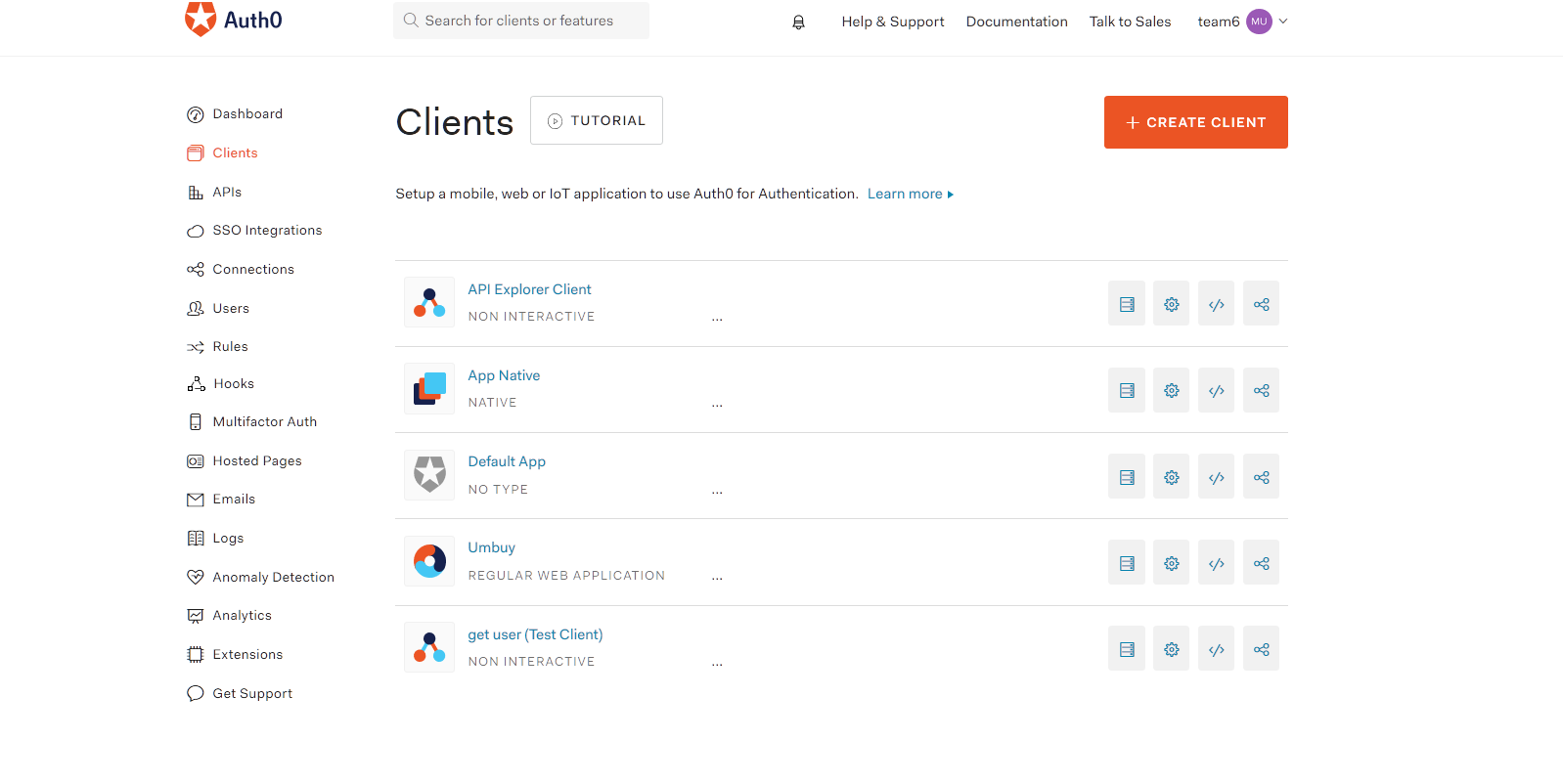
APIs: has the Auth0 Management API and API Audience

The API can be used to authenticate or manage users. The API documentation can be found in the following link.

API Link: <https://auth0.com/docs/api/management/v2>

**Setting up Auth0 with angular:**

Before integrating auth0 to our Angular App we have to create a client in the Auth0 Dashboard. To create a new client click on Clients and click Create Client.



Name the client and click on Single Page Web Application. Now the client is created which can allows us to provide authentication service between our web app and User database in Auth0.

Configuring Callback URL:

Callback url is used to redirect the user to url after the use has been authenticated. If you are running Angular locally set the callback URL to <http://localhost:3000/callback>.

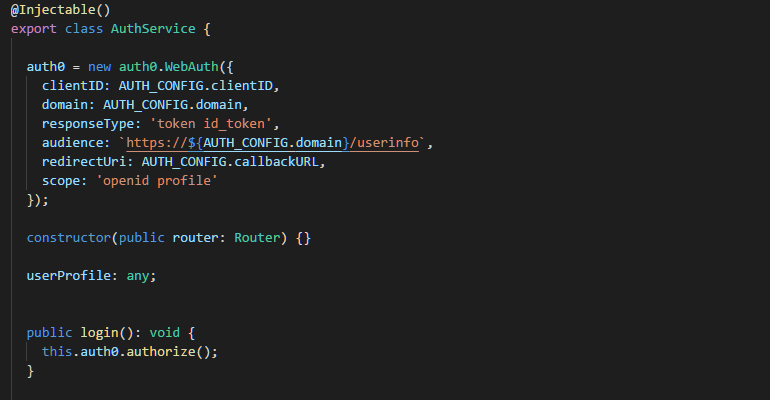
Installing Auth0 library:

We have to install auth0.js library to integrate Auth0 in to our application.

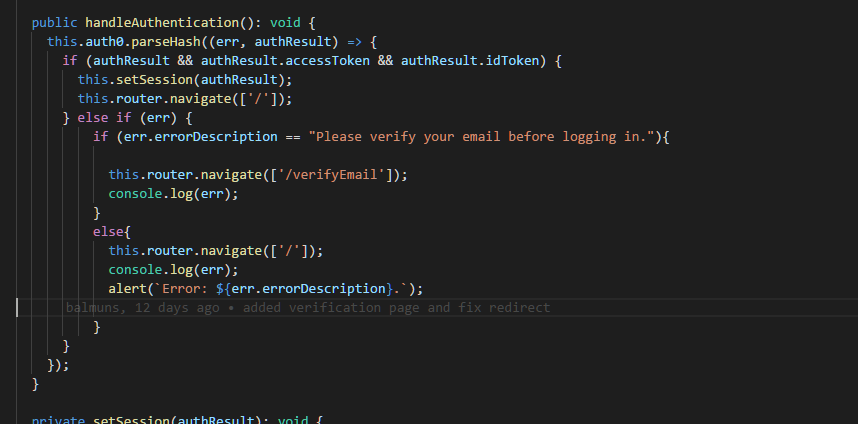
To install auth.js run command: npm install --save auth0-js

Create an Authentication Service that will manage and coordinate user Authentication.

In the service we have to specify an instance of auth0.WebAuth which will contain information about the client we created and add a login function that will call auth0.authorize which will be used to authorize the user.



Create handle Authentication function that will handle what happens after the authentication.



You can add login and Sessions functions by looking at the Documentation provided by Auth0

<https://auth0.com/docs/quickstart/spa/angular2/01-login>

Adding Auth0 to Android:

To integrate Auth0 in android, we need to add the Auth0 depency in our gradle.

Add the following in the gradle dependencies .  
compile 'com.auth0.android:lock:2.7.0'

Now in the manifest placeholder,add the

<!-- Auth0 Lock -->

<activity

android:name="com.auth0.android.lock.LockActivity"

android:label="@string/app\_name"

android:launchMode="singleTask"

android:screenOrientation="portrait"

android:theme="@style/MyTheme" />

<!-- Auth0 Lock End -->

Now, we have the Auth0 dependency in Android we can start creating the LoginActivity.

All the necessary documentation for Android SDK for Auth0 can be found in the following link.

<https://auth0.com/docs/quickstart/native/android/getting-started>

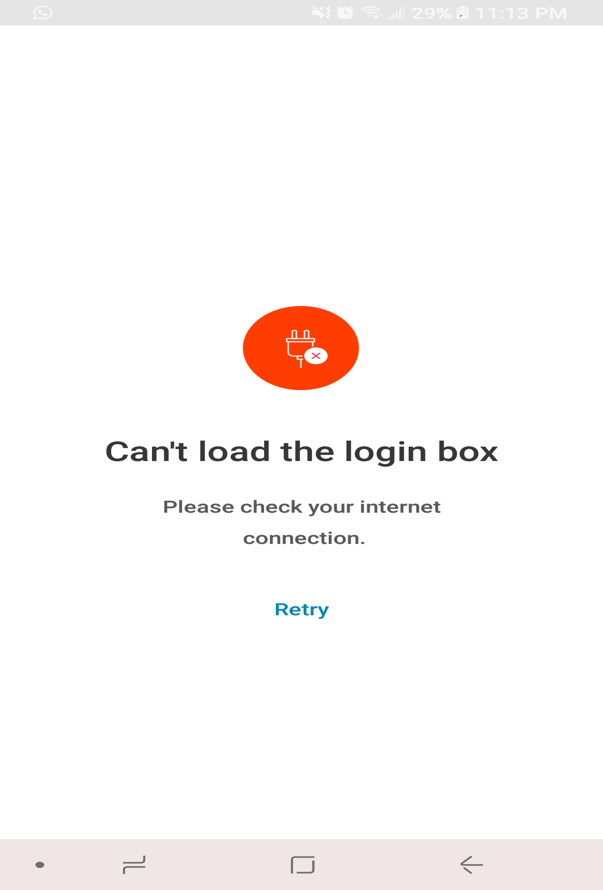
**User Profile:**  
 We get an userProfile as a payload when the authenciation in successful in Android. We are using that to get the user infromation.

**Testing Auth:**

\*We are not testing Auth0 because it is an external dependency and is an external API as discussed with professor.

Issue with Login on Android:

* Auth0 sometimes has issues to do with the DNS and shows the “Can’t load the login box page”. This issue is resolved by using a VPN or changing the DNS. See screenshot below:
* It is not within our control and seems to be an Auth0 provider problem



Auth0 does not recommend using the embedded login anymore. They recommend switching to web based authentication which they say is more secure. If we had more time, we would have switched to the newer authentication system provided by auth0