Development of Login and Registration Feature using Auth0 and Microsoft Identity

1.Introduction:-

This Report aims to demonstrate the integration of Auth0 with Microsoft Identity to implement a login and registration feature in a web application. By leveraging Auth0's authentication platform and Microsoft Identity's trusted mechanism, users can securely log in and register using their Microsoft accounts. This POC underscores the importance of seamless authentication and highlights the benefits of utilizing industry-leading identity management solutions for enhanced security and user experience in web applications.

2.Research on auth0 and Microsoft Identity:-

Auth0:

Auth0 stands as a leading authentication and authorization platform, providing developers with robust solutions to implement secure and seamless user authentication in their applications. With Auth0, developers can offload the complexity of authentication workflows, including social login, multi-factor authentication, and single sign-on (SSO), while ensuring compliance with industry standards like OAuth 2.0 and OpenID Connect. By abstracting away authentication complexities, Auth0 empowers developers to focus on building features and delivering value, all while maintaining the highest standards of security and user experience.

Key Features of Auth0:-

1. User Authentication: Auth0 helps your app verify who your users are, so only authorized individuals can access it.
2. Social Login: Users can log in using their existing accounts from popular social platforms like Google, Facebook, or Microsoft, making the login process faster and easier.
3. Single Sign-On (SSO): Once users log in to one of your apps, they don't need to enter their credentials again when accessing other connected apps, streamlining their experience.
4. Multi-Factor Authentication (MFA): Adds an extra layer of security by requiring users to provide two or more forms of verification, like a password and a code sent to their phone.
5. Customizable Login Pages: You can design login and registration pages that match your app's look and feel, providing a seamless user experience.
6. Authorization: Auth0 helps control what users can and cannot do within your app, ensuring that only authorized users can access certain features or data.
7. Scalability: Whether you have a handful of users or millions, Auth0 scales with your app, handling authentication traffic efficiently and reliably.
8. Security: Auth0 follows best practices for security, protecting user data with encryption and regularly updating its systems to defend against new threats.
9. Identity Management: Easily manage user identities, including user profiles, permissions, and roles, all from one centralized dashboard.

10.Extensibility: Customize and extend Auth0's functionality with custom rules, hooks, and extensions, allowing you to tailor authentication to your specific needs.

Use Case: Secure Authentication for a Web Application

Imagine you're developing a web application that requires users to create accounts, log in, and access personalized content. You want to ensure that the authentication process is secure, user-friendly, and scalable as your application grows. Auth0 provides a comprehensive solution for implementing secure authentication with minimal development effort.

Microsoft Identity:

Microsoft Identity is a comprehensive identity platform offered by Microsoft, designed to simplify and enhance the authentication and authorization process for applications and services. Leveraging industry-leading technologies and standards, Microsoft Identity provides developers with a suite of tools and services to manage user identities securely across various platforms and devices.

With Microsoft Identity, developers can implement robust authentication mechanisms, including single sign-on (SSO), multi-factor authentication (MFA), and social login, enabling users to access applications and services with ease while maintaining security and compliance standards.

Key Features of Microsoft Identity:

1. Single Sign-On (SSO): Allows users to sign in once and access multiple applications and services without needing to enter their credentials repeatedly.
2. Multi-Factor Authentication (MFA): Adds an extra layer of security by requiring users to provide additional verification methods, such as SMS codes or biometric authentication, during the sign-in process.
3. Social Login Integration: Enables users to authenticate using their existing social media accounts, such as Microsoft, Google, or Facebook, simplifying the login experience and increasing user engagement.
4. Identity Management: Provides a centralized platform for managing user identities, including user profiles, permissions, and access control policies, ensuring consistent and secure identity management across applications.
5. Developer Tools and SDKs: Offers a range of developer tools, libraries, and software development kits (SDKs) to simplify the integration of Microsoft Identity into various applications and platforms, including web, mobile, and desktop.
6. Compliance and Security: Ensures compliance with industry standards and regulations, such as OAuth 2.0 and OpenID Connect, while implementing robust security measures to protect user data and credentials against unauthorized access and attacks.
7. Scalability and Reliability: Provides scalable and reliable identity services, capable of handling millions of authentication requests and user identities, with built-in redundancy and high availability.

8. Integration with Microsoft Services: Seamlessly integrates with other Microsoft services and platforms, such as Azure Active Directory (Azure AD) and Microsoft 365, to provide a unified identity and access management solution for organizations.

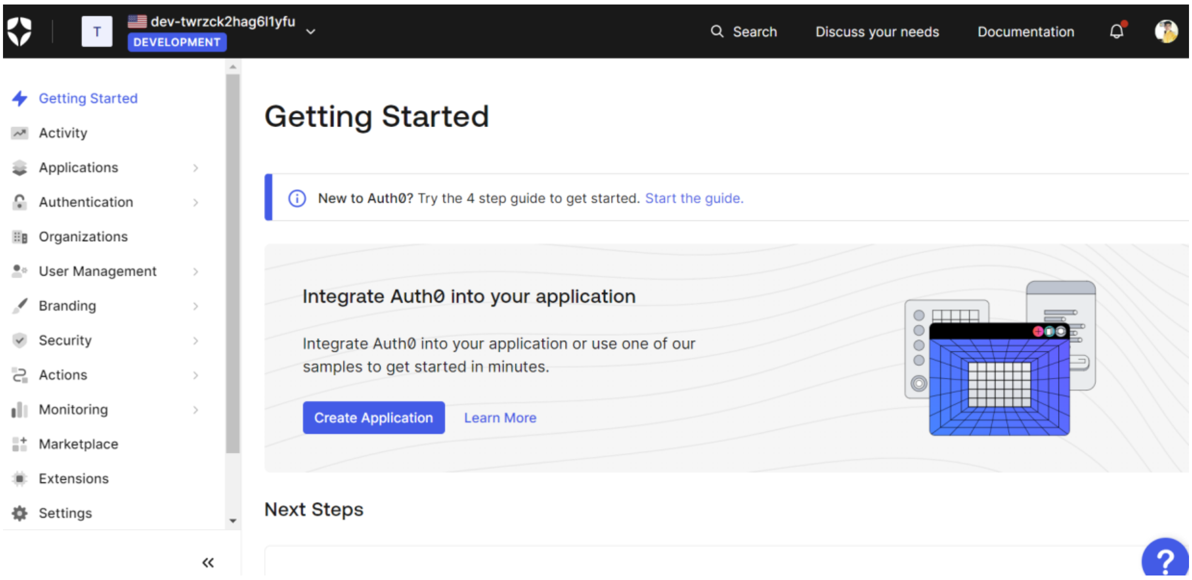
Use Case: Enterprise Application Single Sign-On (SSO) with Microsoft Identity

Scenario: A large enterprise with multiple internal applications, such as HR management systems, document repositories, and project management tools, wants to implement a seamless Single Sign-On (SSO) solution for its employees. The goal is to improve user experience, increase security, and simplify identity management across the organization.

3.Set Up and Configuration:

Setup Process of Auth0:

1. **Create an Auth0 Account:** If you don't have an Auth0 account, sign up for one at https://auth0.com/.
2. **Login to Auth0 Dashboard:** After creating an account, login to the Auth0 Dashboard.



Getting Started

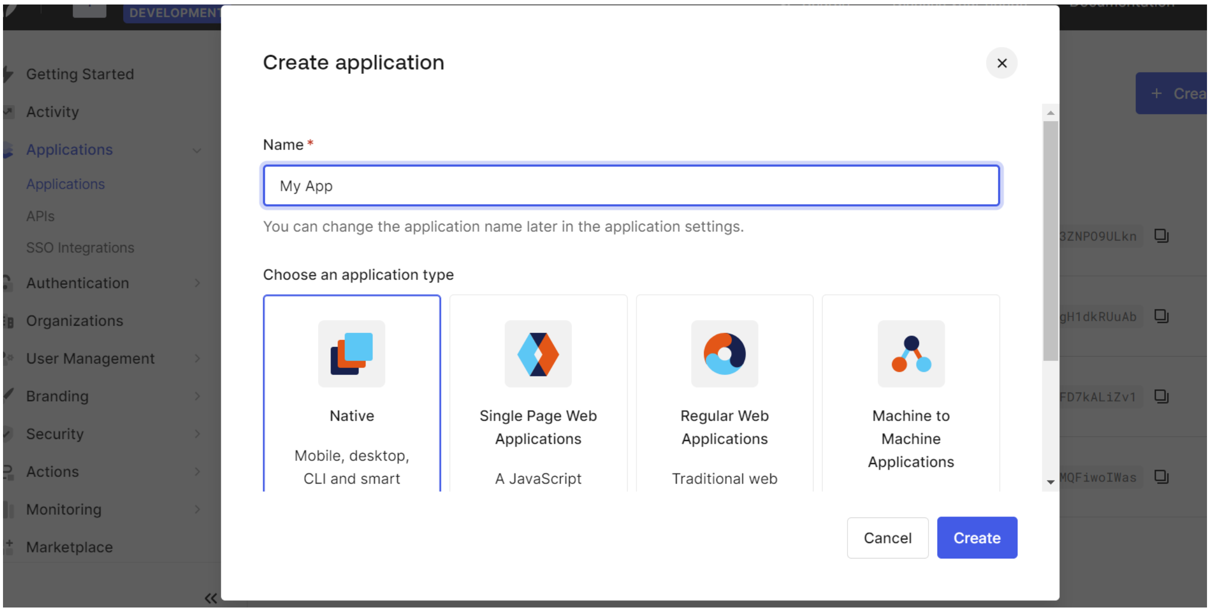
New to Auth0? Try the 4 step guide to get started. Start the guide.

Integrate Auth0 into your application

Integrate Auth0 into your application or use one of our samples to get started in minutes.

Next Steps

3. **Create a New Application:** In the Auth0 Dashboard, navigate to the Applications section and click on the "Create Application" button. Choose the type of application you want to create.



Create application

Name *

My App

You can change the application name later in the application settings.

Choose an application type

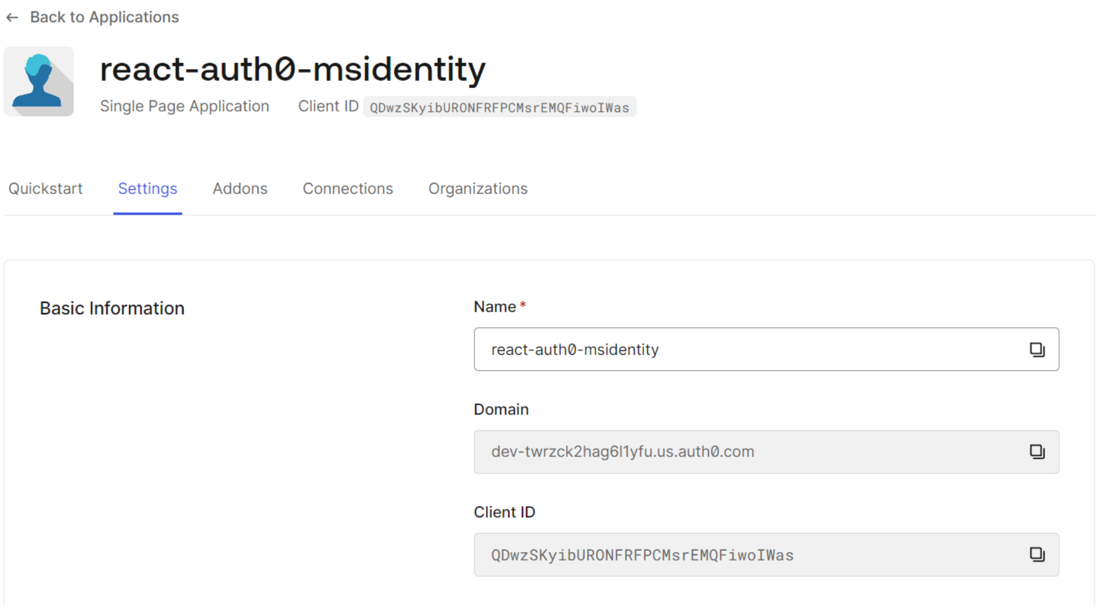
Native — Mobile, desktop, CLI and smart

Single Page Web Applications — A JavaScript

Regular Web Applications — Traditional web

Machine to Machine Applications

4. **Configure Application Settings:** Configure the settings for your application, such as the name, allowed callback URLs, allowed logout URLs, etc.



react-auth0-msidentity

Single Page Application Client ID `QDwzSKyibURONFRFPCMsrEMQFiwoIWas`

Basic Information

Name *

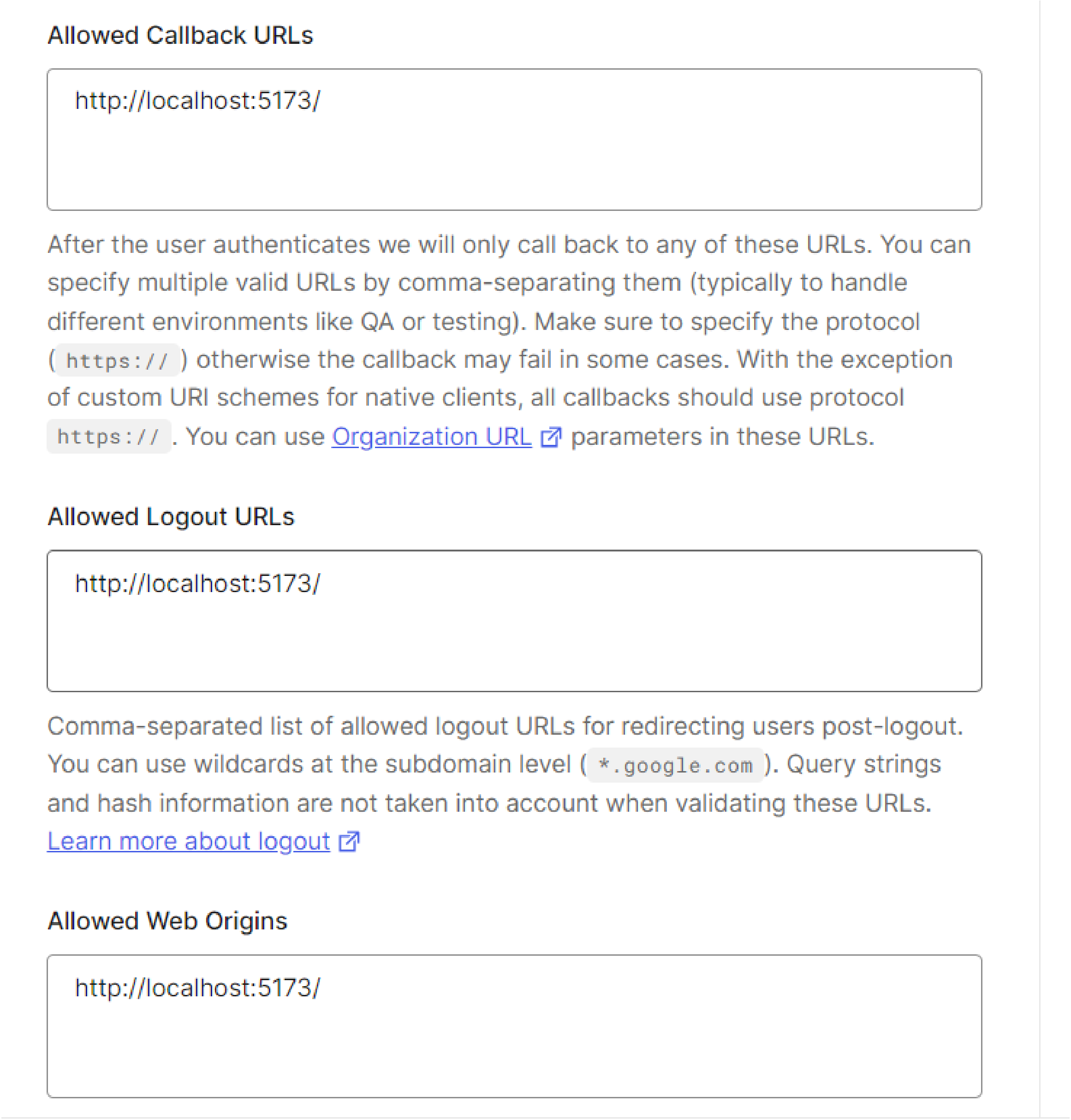
react-auth0-msidentity

Domain

dev-twrzck2hag6l1yfu.us.auth0.com

Client ID

`QDwzSKyibURONFRFPCMsrEMQFiwoIWas`



Allowed Callback URLs

http://localhost:5173/

After the user authenticates we will only call back to any of these URLs. You can specify multiple valid URLs by comma-separating them (typically to handle different environments like QA or testing). Make sure to specify the protocol (`https://`) otherwise the callback may fail in some cases. With the exception of custom URI schemes for native clients, all callbacks should use protocol `https://` . You can use Organization URL parameters in these URLs.

Allowed Logout URLs

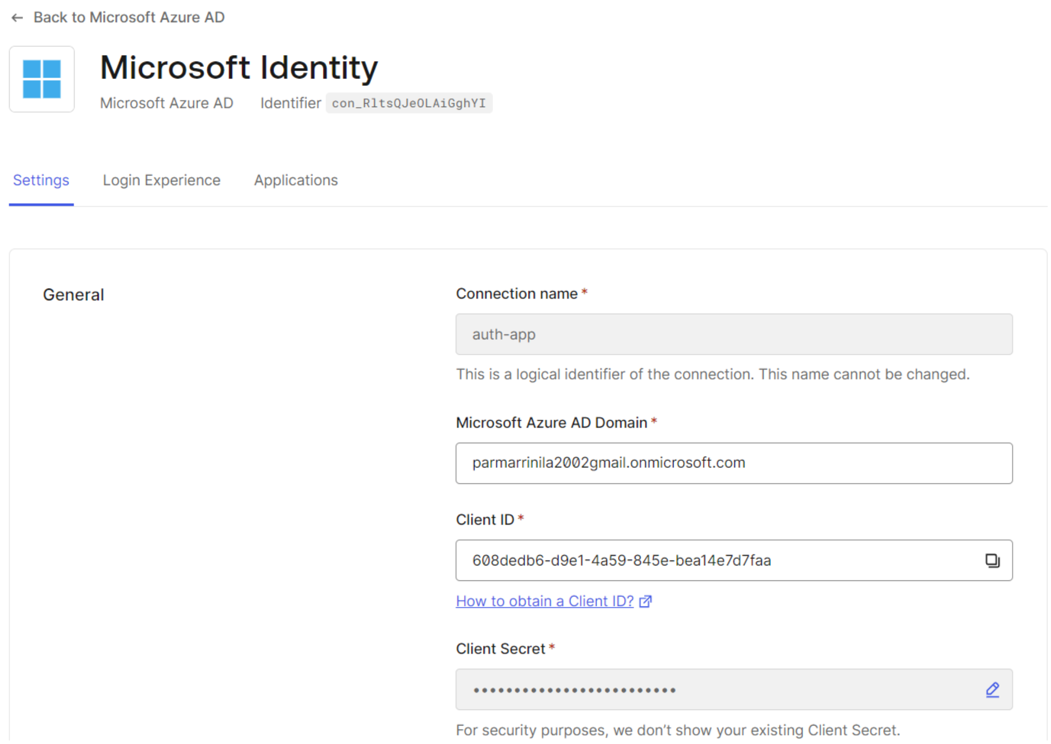
http://localhost:5173/

Comma-separated list of allowed logout URLs for redirecting users post-logout. You can use wildcards at the subdomain level (`*.google.com`). Query strings and hash information are not taken into account when validating these URLs. Learn more about logout

Allowed Web Origins

http://localhost:5173/

5. **Enable Microsoft Identity Provider:** In the Auth0 Dashboard, go to the Connections > Enterprise section and enable Microsoft Identity as a connection. And configure it. For this we need to create new application in Azure.



← Back to Microsoft Azure AD

Microsoft Identity

Microsoft Azure AD Identifier con_RltsQJeOLAiGghYI

Settings Login Experience Applications

General

Connection name *

auth-app

This is a logical identifier of the connection. This name cannot be changed.

Microsoft Azure AD Domain *

parmarrinila2002gmail.onmicrosoft.com

Client ID *

608dedb6-d9e1-4a59-845e-bea14e7d7faa

How to obtain a Client ID?

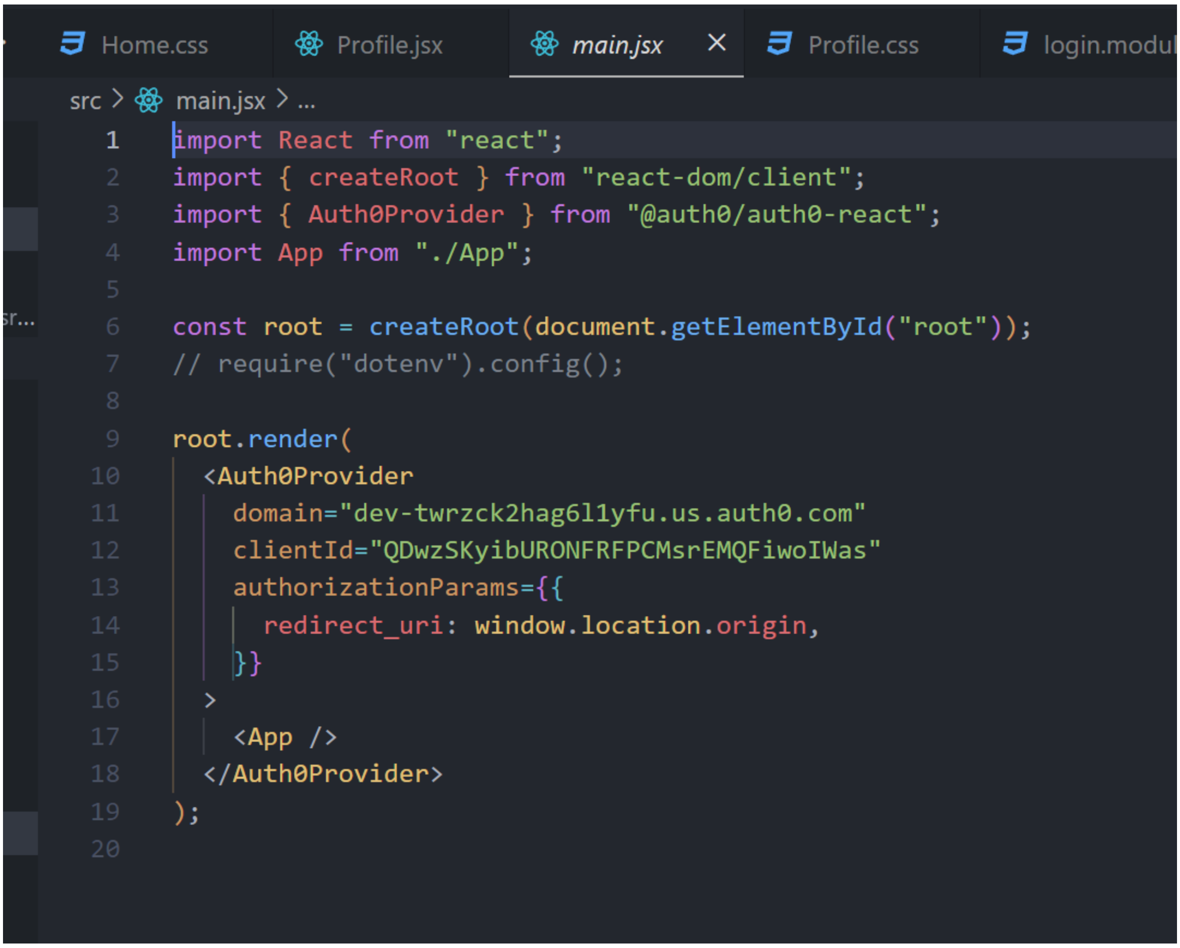
Client Secret *

••••••••••••••••••••••••••

For security purposes, we don't show your existing Client Secret.

6. **Copy Client ID and Client Secret:** After enabling Microsoft Identity, you'll be provided with a Client ID and Client Secret.

   And add to your project code. Wrap App component by auth0provider. Using 'useAuth0()' you can use login of auth0.



```jsx
import React from "react";
import { createRoot } from "react-dom/client";
import { Auth0Provider } from "@auth0/auth0-react";
import App from "./App";

const root = createRoot(document.getElementById("root"));
// require("dotenv").config();

root.render(
  <Auth0Provider
    domain="dev-twrzck2hag6l1yfu.us.auth0.com"
    clientId="QDwzSKyibURONFRFPCMsrEMQFiwoIWas"
    authorizationParams={{
      redirect_uri: window.location.origin,
    }}
  >
    <App />
  </Auth0Provider>
);
```

7. **Show User Profile Information:** Utilize the user property provided by Auth0 React SDK to display user information.

Setup Process of Microsoft Identity:

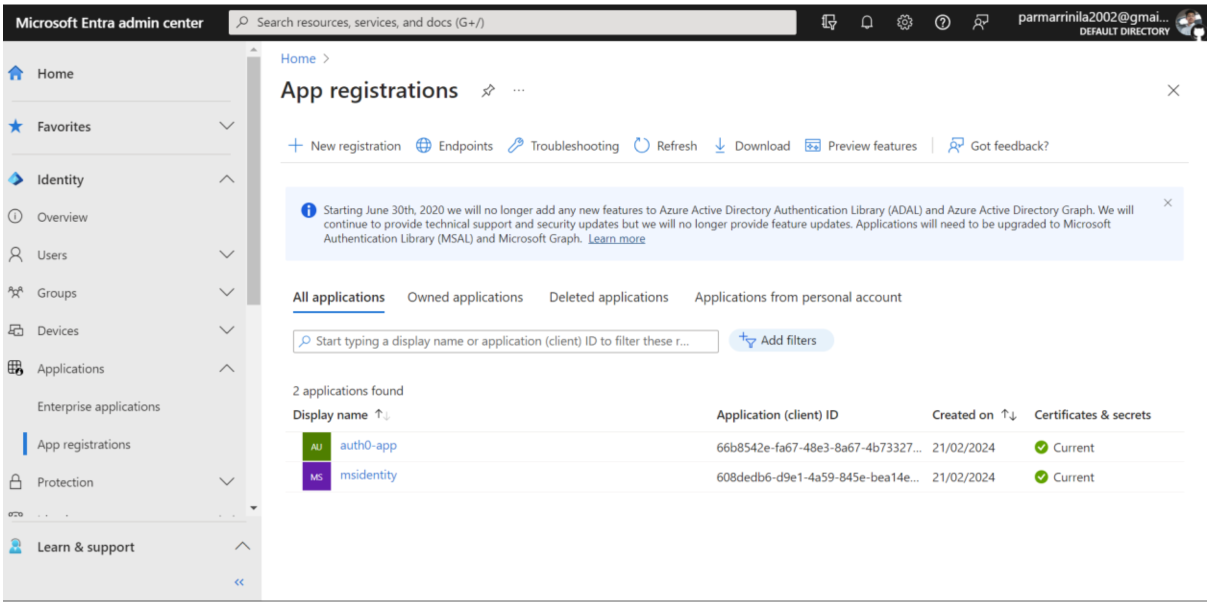
1.Azure Portal Setup

- Navigate to the Azure portal and sign in with your Azure account.

- In the Azure portal, select "Azure Active Directory" from the left-hand navigation.

- Choose "App registrations" and create a new application registration for your webapplication.

- Note down the Application (client) ID and Directory (tenant) ID.

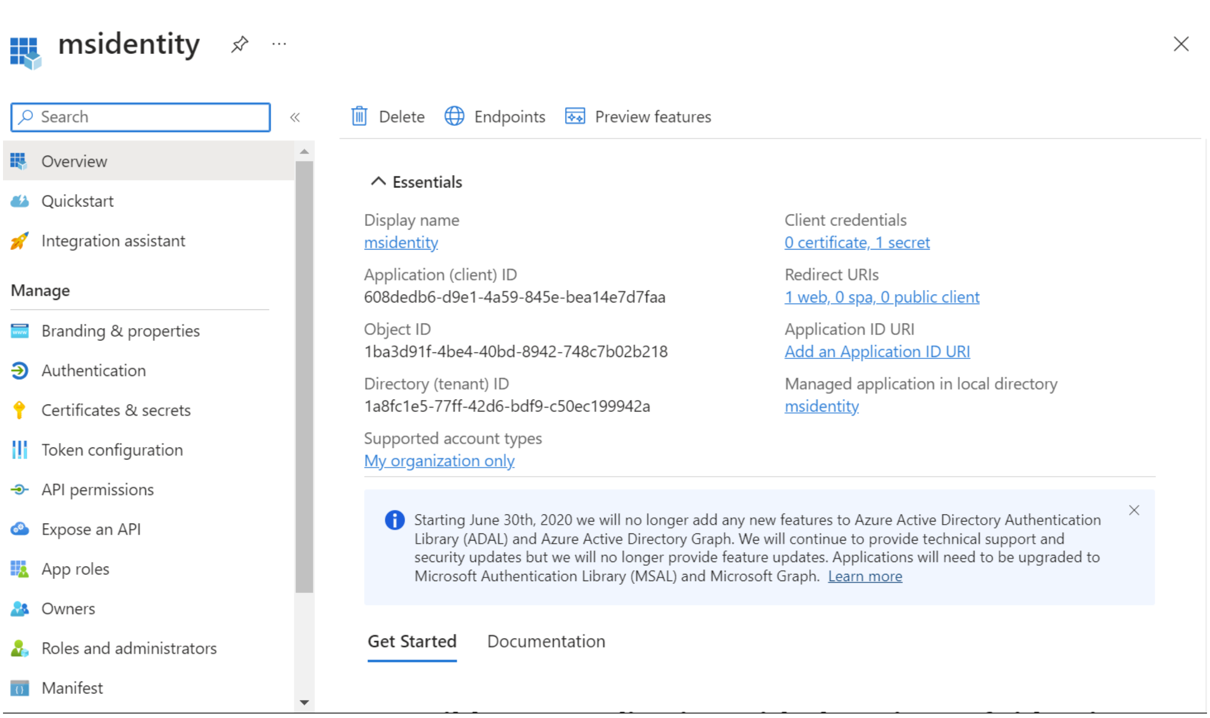


App registrations

Starting June 30th, 2020 we will no longer add any new features to Azure Active Directory Authentication Library (ADAL) and Azure Active Directory Graph. We will continue to provide technical support and security updates but we will no longer provide feature updates. Applications will need to be upgraded to Microsoft Authentication Library (MSAL) and Microsoft Graph. Learn more

2 applications found

Display name	Application (client) ID	Created on	Certificates & secrets
AU auth0-app	66b8542e-fa67-48e3-8a67-4b73327...	21/02/2024	✓ Current
MS msidentity	608dedb6-d9e1-4a59-845e-bea14e...	21/02/2024	✓ Current



msidentity

Essentials

Display name
msidentity

Application (client) ID
608dedb6-d9e1-4a59-845e-bea14e7d7faa

Object ID
1ba3d91f-4be4-40bd-8942-748c7b02b218

Directory (tenant) ID
1a8fc1e5-77ff-42d6-bdf9-c50ec199942a

Supported account types
My organization only

Client credentials
0 certificate, 1 secret

Redirect URIs
1 web, 0 spa, 0 public client

Application ID URI
Add an Application ID URI

Managed application in local directory
msidentity

Starting June 30th, 2020 we will no longer add any new features to Azure Active Directory Authentication Library (ADAL) and Azure Active Directory Graph. We will continue to provide technical support and security updates but we will no longer provide feature updates. Applications will need to be upgraded to Microsoft Authentication Library (MSAL) and Microsoft Graph. Learn more

2.Configure Authentication in Auth0

- In the application registration settings, go to the "Authentication" tab.
- Add the appropriate redirect URIs for your application.
- Configure the "Implicit grant" and "ID tokens" settings.
- Save the changes.

3. Set Permissions

- In the application registration settings, go to the "API permissions" tab.

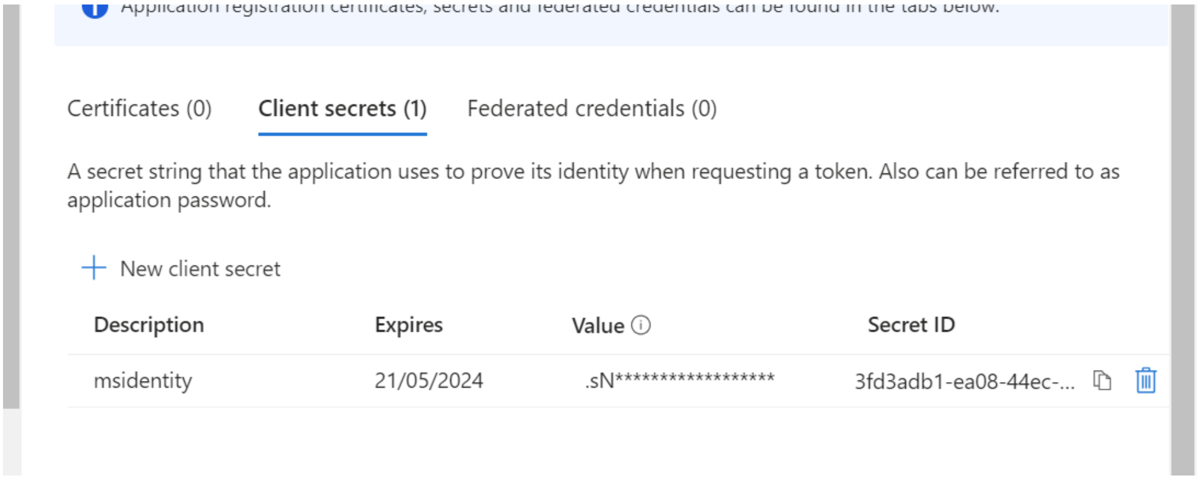
- Add the necessary permissions required for your application, such as user. Read.

- Grant admin consent for the added permissions.

4. Obtain Client Secret

- In the application registration settings, go to the "Certificates & secrets" tab.

- Generate a new client secret and note down the value.



Application registration certificates, secrets and federated credentials can be found in the tabs below.

A secret string that the application uses to prove its identity when requesting a token. Also can be referred to as application password.

Description	Expires	Value ⓘ	Secret ID
msidentity	21/05/2024	.sN******************	3fd3adb1-ea08-44ec-...

5. Now add client Id and secret client into auth0 enterprise azure AD configuration. and test it.

4.Conclusion:

In conclusion, the successful integration of Auth0 or Microsoft Identity into web applications for login pages is pivotal in ensuring a secure and user-friendly authentication experience. Whether leveraging Auth0's comprehensive platform or Microsoft Identity's seamless integration with Azure AD, developers can create login pages that prioritize security, scalability, and user satisfaction. By delivering a frictionless authentication process, developers contribute to the overall success of their applications, fostering trust and enhancing the user experience.

5. References:

For Auth0 Integration:

1. Auth0 Documentation: https://auth0.com/docs
2. Microsoft Identity Platform Documentation: https://docs.microsoft.com/en-us/azure/active-directory/
3. Microsoft Identity Blog: https://techcommunity.microsoft.com/t5/azure-active-directory-identity/bg-p/AzureADIdentity
4. Microsoft Q&A: https://docs.microsoft.com/en-us/answers/topics/azure-active-directory.html
5. Stack Overflow for quick solution of Error while Configuration.