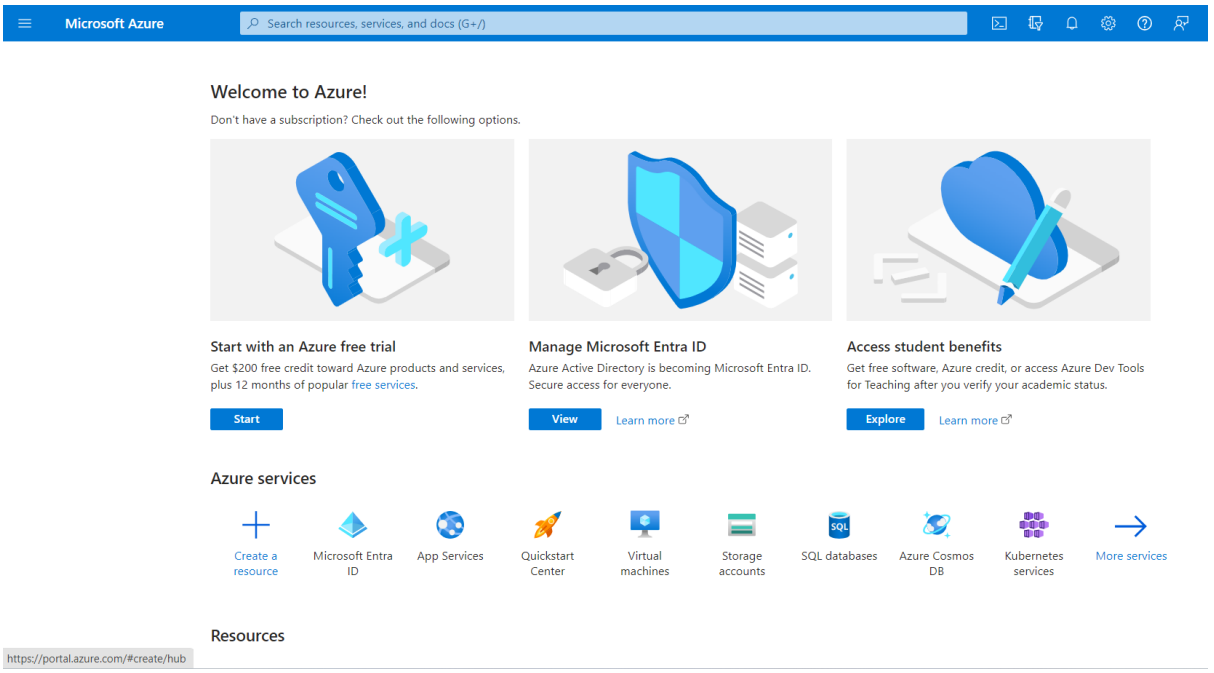
POC for SSO Integration with Azure using Java and React JS using

Microsoft Azure Single Sign-On (SSO) is a cloud-based identity and access management service that enables users to securely sign in once and gain access to various applications and services without the need to re-authenticate. It simplifies user access to a multitude of applications, enhancing productivity and providing a seamless and secure authentication experience.

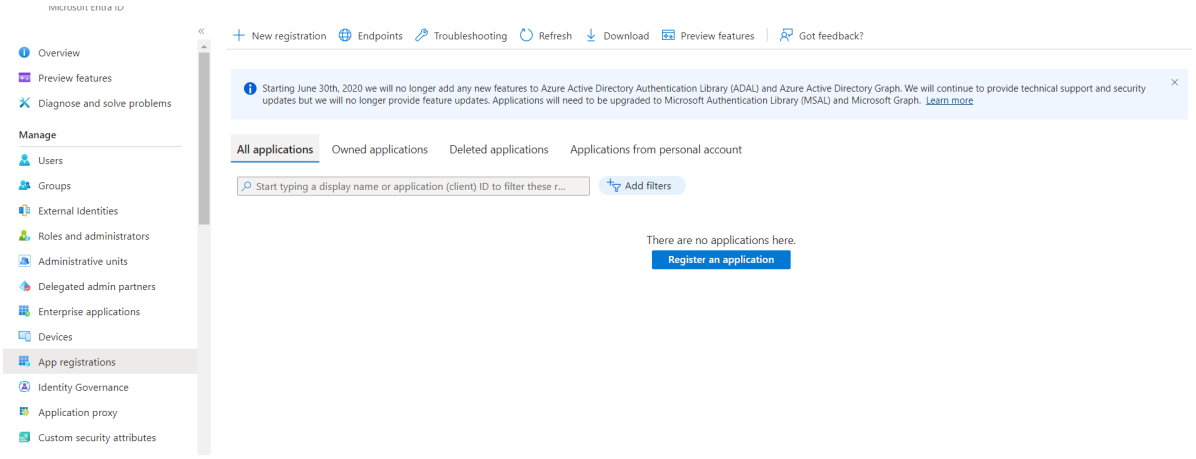
Here are few steps which required to setup the azure account

Setup Azure account with default directory

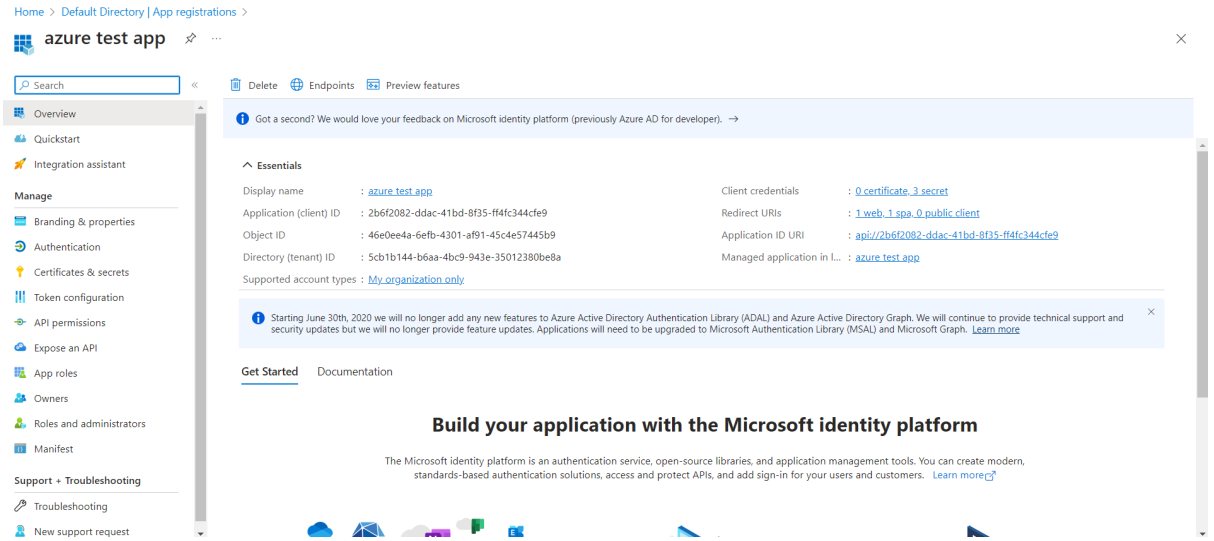
Sign in to azure portal by clicking this https://portal.azure.com/. After signing in, the home page will be displayed as shown in the image, and after that click on **Manage Microsoft Entra ID**.



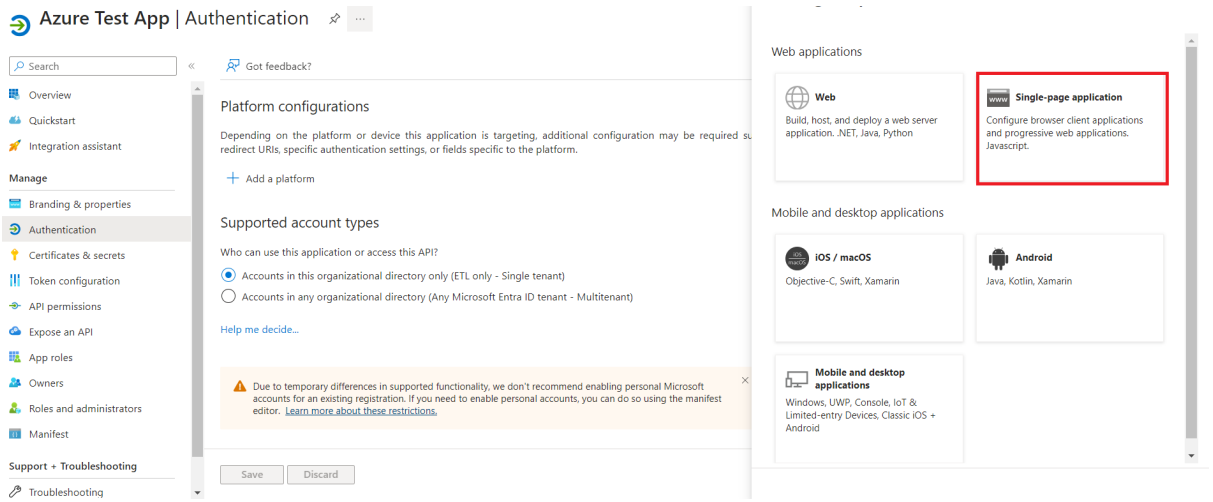
After clicking on Microsoft Entra Id, the Default directory will be displayed. Now go to **App Registration -> New Registration**



After successful application registration, the application overview page will be displayed. And application information will displayed like Client Id, Tenant Id

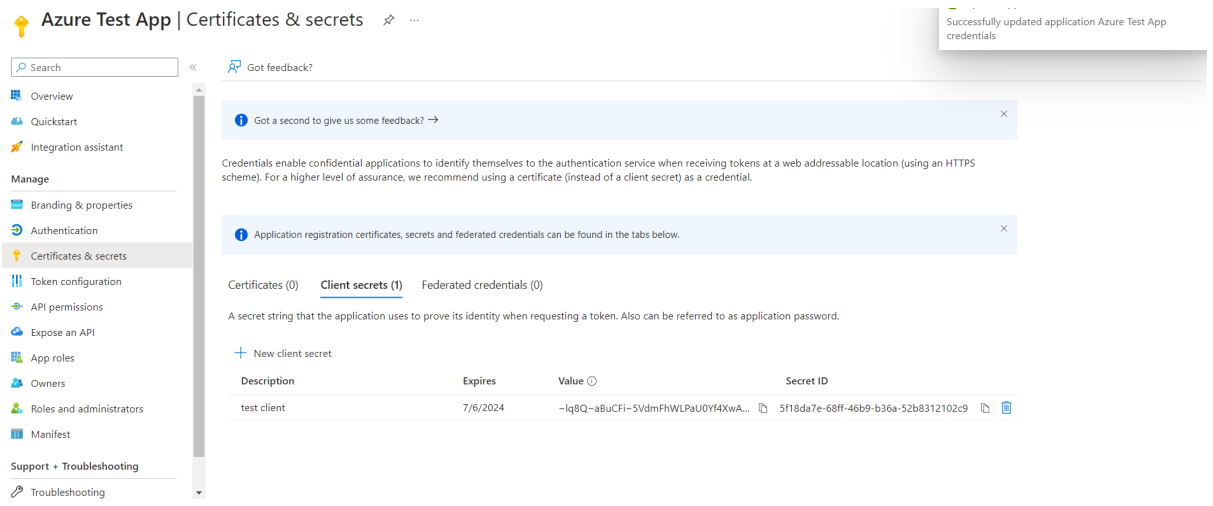


To set up a redirect(Callback) url, click on Authentication -> Add a platform. As we will create authentication from react so we will choose a **single page application** and provide a redirect URL. In this demo, we have provided the URL as http://localhost:3000/. And choose Access tokens for authorization endpoint.



Generate client secret for authentication:

To generate client secret for authentication, go to **Certificates and Secrets** and generate client secret and copy that secret value.



Add Azure Properties in React App:

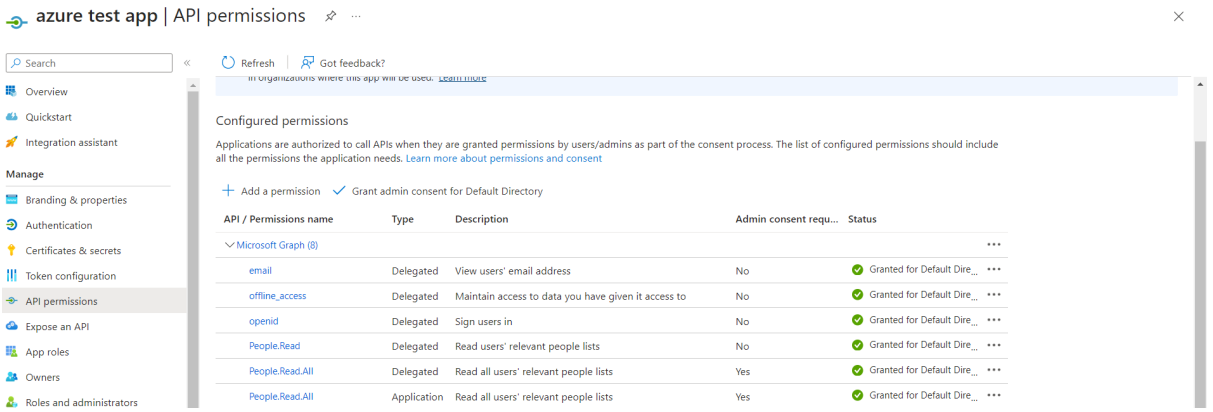
We need to create react project and add azure configuration properties in the React application in the env file.

So by running the react application, we will be able to login using Microsoft Azure and get the **Access Token** of azure.

And with the help of access token, we can access APIs of Microsoft Azure. Here we have used Access Token in Bearer authentication to fetch the profile data of signed in user by using

API Endpoint: https://graph.microsoft.com/v1.0/me
Method: Post
Authorization: Bearer + AccessToken

To access the Azure APIs, we need to provide permissions from the App **Permission** section and provide admin consent to the directory.



Authenticate using Java

To authenticate Microsoft Azure using java, we need to add below dependency to our pom.xml file

```
<dependency>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-oauth2-client</artifactId>
</dependency>
<dependency>
    <groupId>com.azure.spring</groupId>
    <artifactId>spring-cloud-azure-starter-active-directory</artifactId>
</dependency>
```

And add client secret value, redirect URL, tenant id and client Id

```
spring.cloud.azure.active-directory.credential.client-secret=ez-8Q~lMGDL2VYXvkSDm8sUMDkxOuc0OFNuf.
spring.cloud.azure.active-directory.redirect-uri=http://localhost:8080/login/oauth2/code
spring.cloud.azure.active-directory.enabled=true
spring.cloud.azure.active-directory.profile.tenant-id=5cb1b144-b6aa-4bc9-943e-35012380be8a
spring.cloud.azure.active-directory.credential.client-id=2b6f2082-ddac-41bd-8f35-ff4fc344cfe9
```

So by configuring azure properties, we can successfully login to azure using java and react.