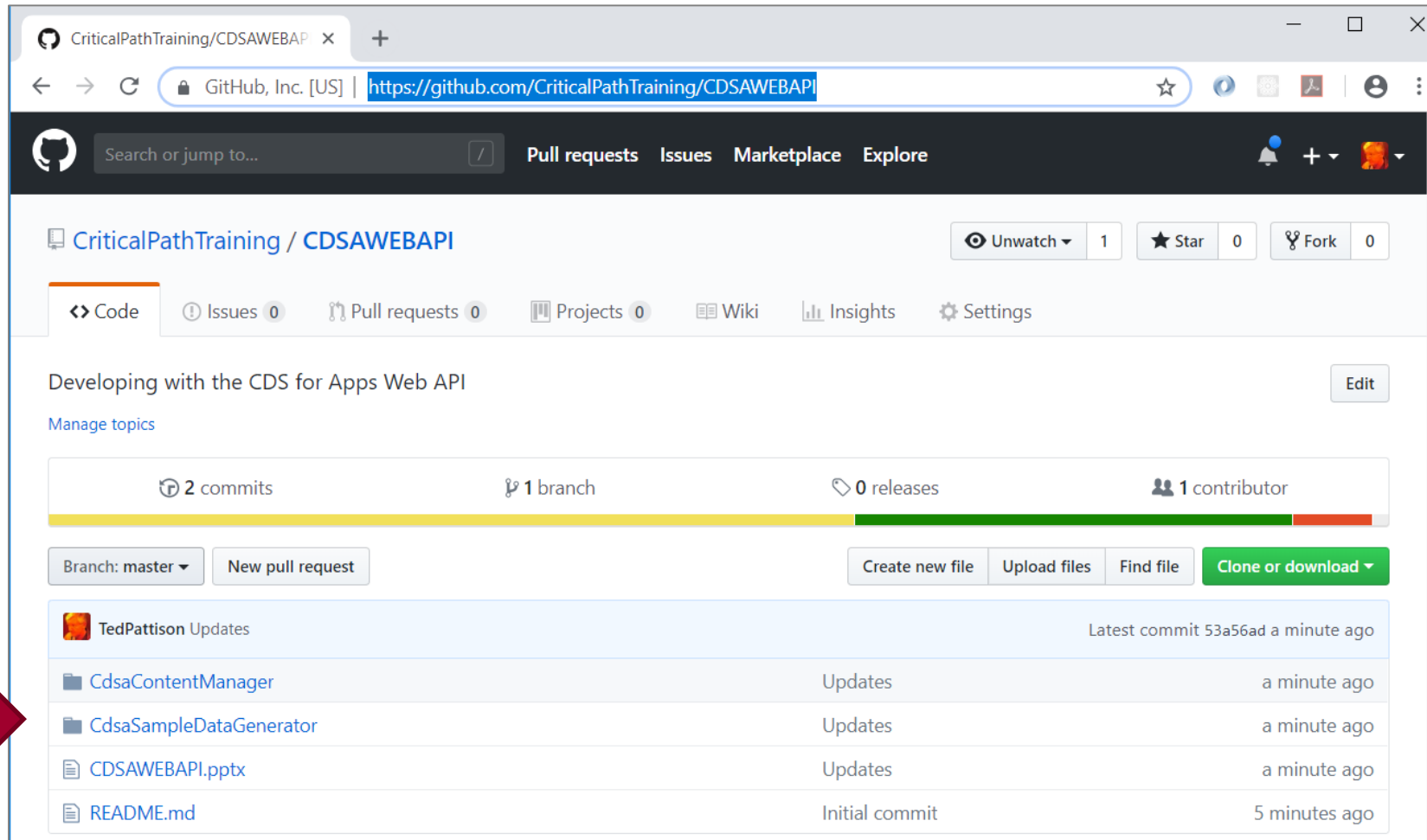


# Developing with the CDS for Apps Web API



# Slides and Code for this Session

- <https://github.com/CriticalPathTraining/CDSAWEBAPI>



The screenshot displays the GitHub repository page for `CriticalPathTraining / CDSAWEBAPI`. The page includes a search bar, navigation links for Pull requests, Issues, Marketplace, and Explore, and a header with repository statistics: 1 commit, 1 branch, 0 releases, and 1 contributor. Below the header, there are tabs for Code, Issues, Pull requests, Projects, Wiki, Insights, and Settings. The main content area shows the repository name, a description "Developing with the CDS for Apps Web API", and a list of files. A red arrow points to the `CdsaContentManager` file in the file list.

File	Commit	Time
<code>CdsaContentManager</code>	Updates	a minute ago
<code>CdsaSampleDataGenerator</code>	Updates	a minute ago
<code>CDSAWEBAPI.pptx</code>	Updates	a minute ago
<code>README.md</code>	Initial commit	5 minutes ago

# Agenda

- CDS for Apps Web API Overview
- Authentication with Azure Active Directory
- Programming the CDS for Apps Web API
- Authentication in a Web App



# Introduction to the CDS for Apps APIs

- You can program CDSA using the Organizational Service
  - .NET-based libraries designed to program with Dynamics 365
  - Handles SOAP-based calls xRM endpoints behind the scenes
  - Lots of existing code is already written this way
- You can program CDSA using the Web API
  - Based on emerging standards (REST, ODATA, OAUTH2, OpenID)
  - You must manage REST-based calls directly to HTTP endpoints
  - Represents where Microsoft's future investment will focus
  - Today, there is no friendly .NET library to assist



# Getting Started

- What you need to get started?
  - Visual Studio 2017
  - Office 365 trial tenant
  - PowerApps Plan2 license
  - Access to Azure portal to create Azure AD applications
- Azure subscription not required!
  - Azure portal used to create Azure AD application
  - Azure subscription helpful to create Azure resources



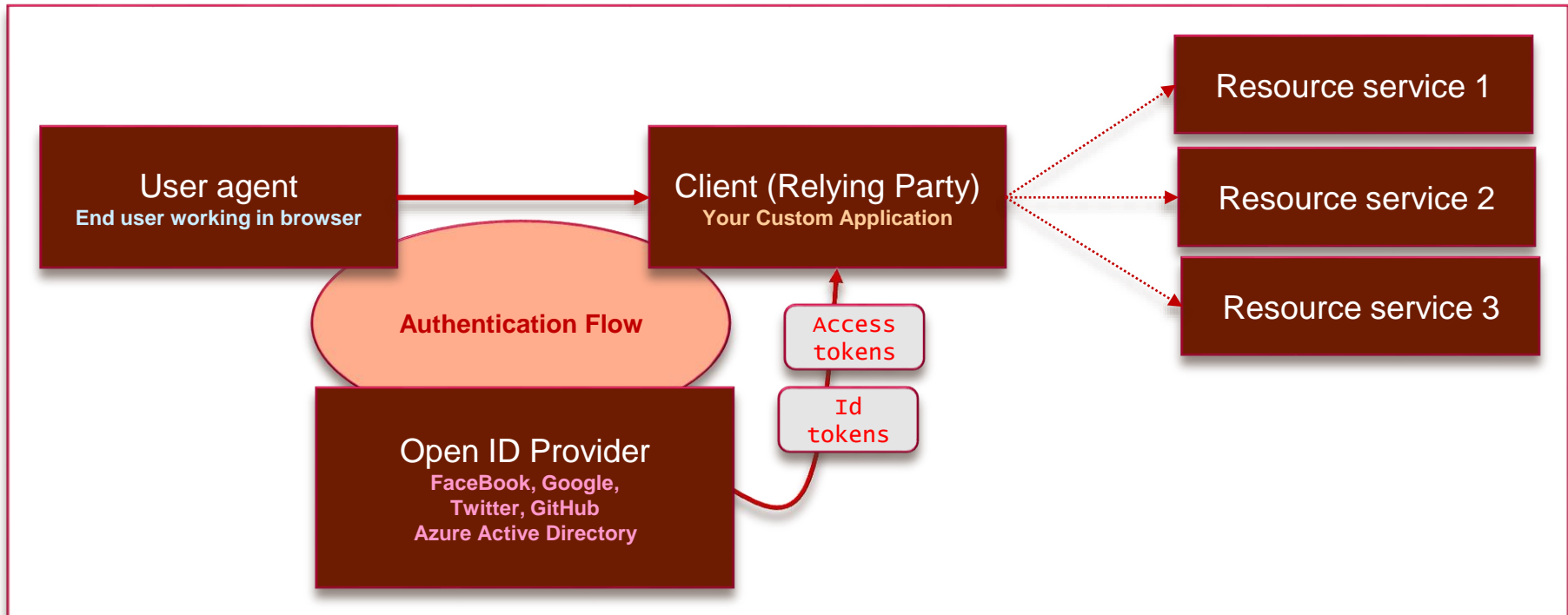
# Agenda

- ✓ CDS for Apps Web API Overview
- Authentication with Azure Active Directory
  - Programming the CDS for Apps Web API
  - Authentication in a Web App



# OAuth2 and Open ID Connect

- Power BI Service requires authentication with OAuth2
  - Your application must implement an authentication flow
  - Authentication flow used to acquire an access token
  - Access token required whenever calling Power BI Service API





# Client Application Registration

- Application must be registered with authorization server
  - Authorization server tracks each client with unique Client ID
  - Client should be registered with one or more Reply URLs
  - Reply URL should be fixed endpoint on Internet
  - Reply URL used to transmit security tokens to clients
  - Client registration tracks permissions and other attributes





# Authentication Flows

- Authorization Code Grant Flow (*confidential client*)
  - Client first obtains authorization code then access token
  - Server-side application code never sees user's password
- Implicit Grant Flow (*public client*)
  - Used in SPAs built with JavaScript and AngularJS
  - Application obtains access token w/o acquiring authorization code
- User Credentials Flow (*public client*)
  - Used in Native clients to obtain access code
  - Requires passing user name and password
- Client Credentials Grant Flow (*confidential client*)
  - Authentication based on SSL certificate with public-private key pair
  - Used to obtain access token when using app-only permissions



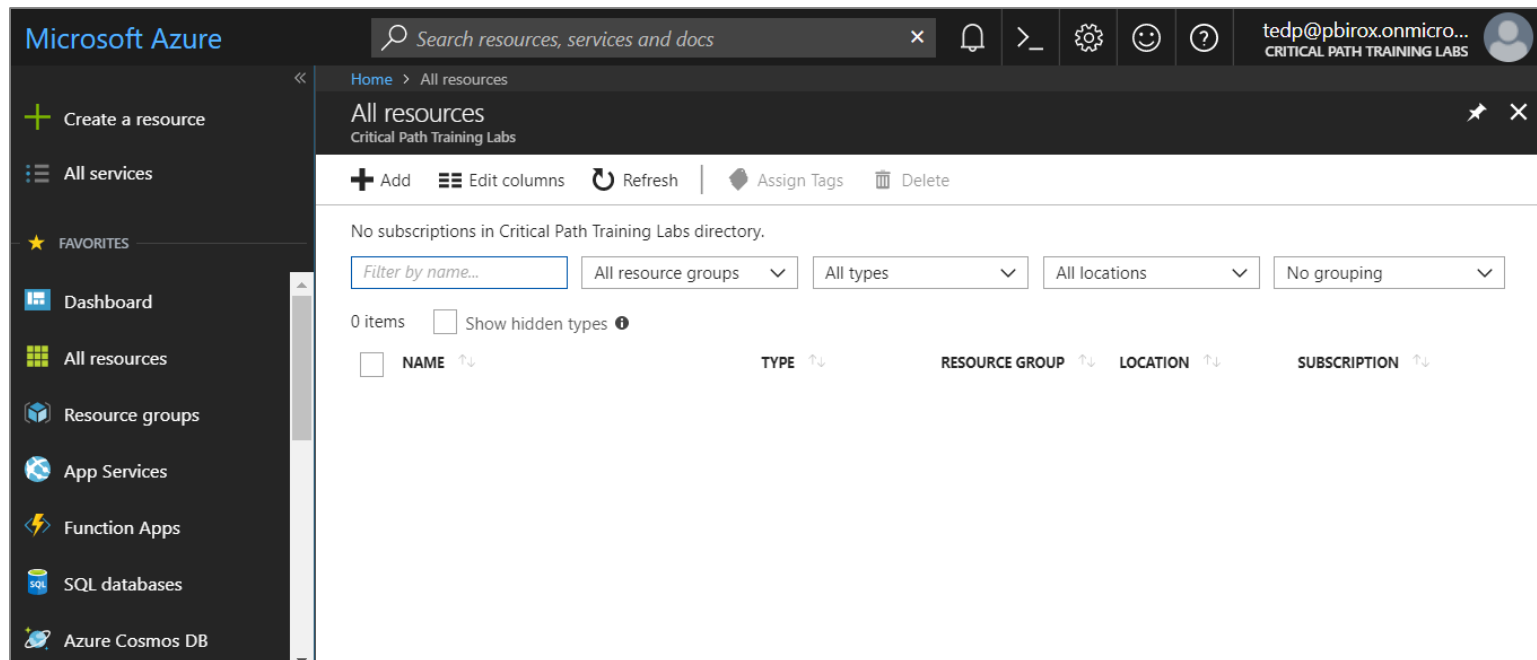
# Azure Active Directory (AAD)

- AAD plays role of an OpenID Connect Provider
  - Creates access tokens based on OAuth 2.0
  - Creates id tokens based on OpenID Connect 1.0
- AAD provides authentication & authorization for...
  - Office 365, Dynamics 365 and SharePoint Online
  - Power BI Service API and Microsoft Graph API
  - Custom Web Applications and Web Services



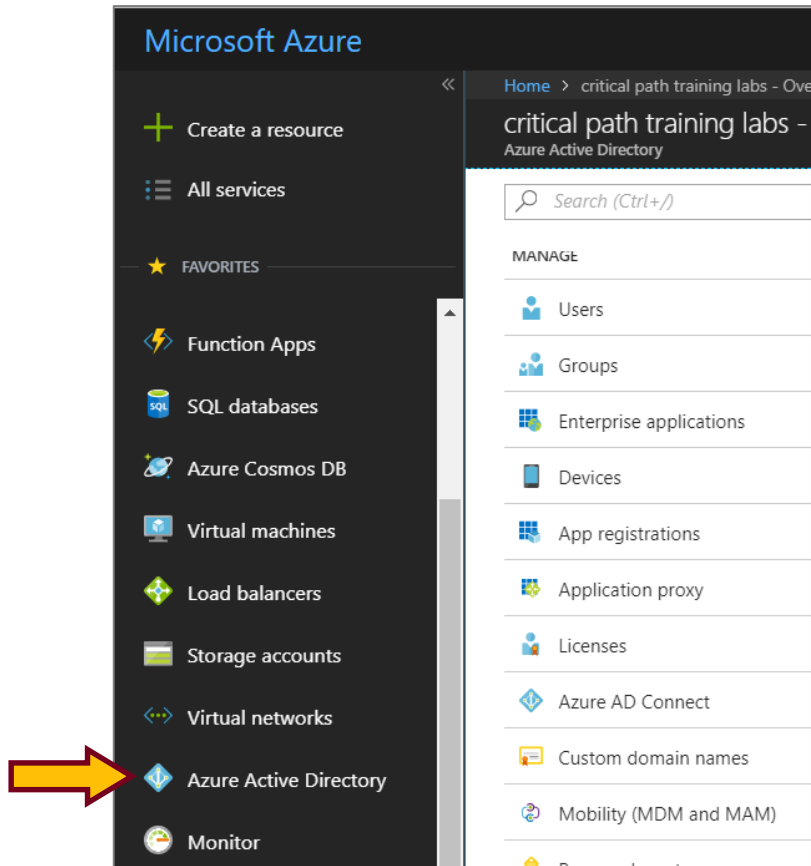
# The Azure Portal

- Azure portal allows to create application
  - Azure Portal accessible at <https://portal.azure.com>
  - Azure subscription required to create resources (e.g. Web Apps, VMs)
  - No Azure subscription required to manage users, groups and applications



# Azure Active Directory

- Azure portal access to Access Azure Active Directory
  - Provides ability to configure users, groups and application



# Managing Users and Groups

- You can manage users and assign licenses
- You can create groups and populate members

Home > critical path training labs > Users - All users

Users - All users  
critical path training labs - Azure Active Directory

Search (Ctrl+/)

+ New user + New guest user Reset password Delete user Multi-Factor Authentication

Name  
Search by name or email

Show  
All users

NAME	USER NAME	USER TYPE
AP Austin Powers	AustinP@pbiox.onMicrosoft.com	Member
CM Carrie Mathison	CarrieM@pbiox.onMicrosoft.com	Member
EP Emma Peel	EmmaP@pbiox.onMicrosoft.com	Member
JB Jack Bauer	JackB@pbiox.onMicrosoft.com	Member
JR Jack Ryan	JackR@pbiox.onMicrosoft.com	Member
JB James Bond	JamesB@pbiox.onMicrosoft.com	Member
JB Jason Bourne	JasonB@pbiox.onMicrosoft.com	Member
MS Maxwell Smart	MaxwellS@pbiox.onMicrosoft.com	Member
TP Ted Pattison	tedp@pbiox.onmicrosoft.com	Member

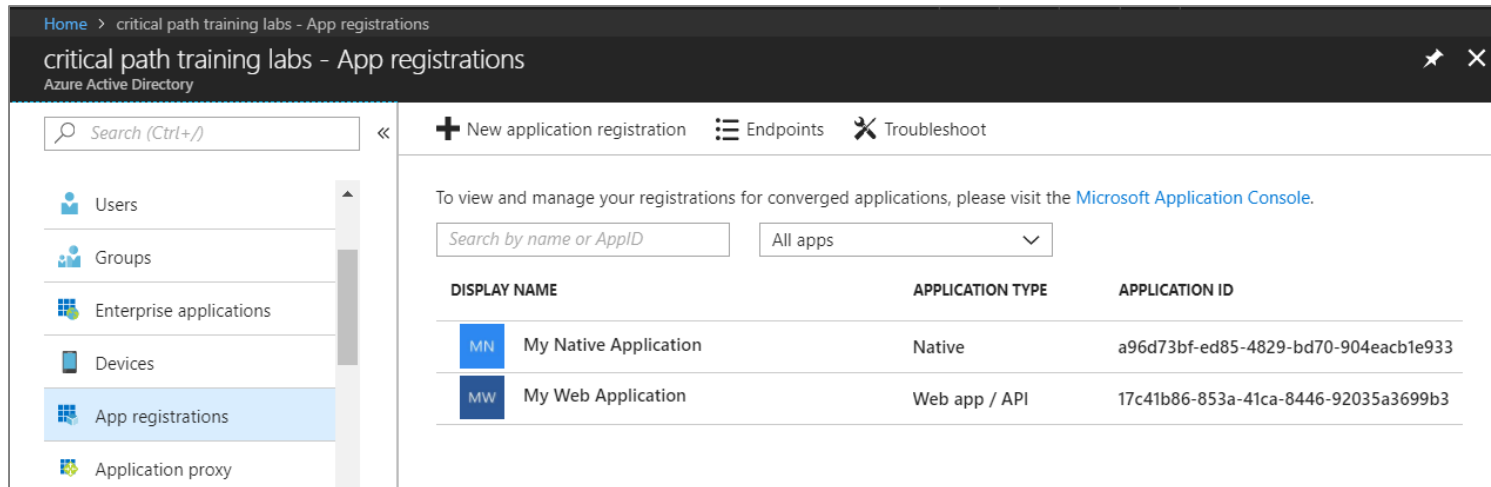
Left sidebar navigation:

- All users
- Deleted users
- Password reset
- User settings
- ACTIVITY
  - Sign-ins
  - Audit logs
- TROUBLESHOOTING + SUPPORT
  - Troubleshoot
  - New support request



# Azure AD Applications

- Creating applications required for AAU authentication
  - Applications are as Native application or Web Applications
  - Application identified using GUID known as **application ID**
  - Application ID often referred to as **client ID** or **app ID**



Home > critical path training labs - App registrations

critical path training labs - App registrations

Azure Active Directory

Search (Ctrl+/)

Users

Groups

Enterprise applications

Devices

App registrations

Application proxy

+ New application registration

Endpoints

Troubleshoot

To view and manage your registrations for converged applications, please visit the [Microsoft Application Console](#).

Search by name or AppID

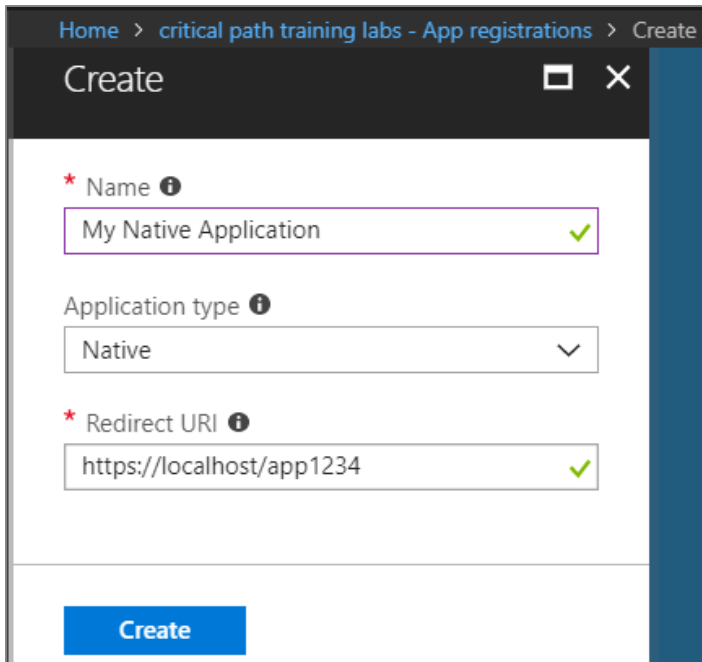
All apps

	DISPLAY NAME	APPLICATION TYPE	APPLICATION ID
MN	My Native Application	Native	a96d73bf-ed85-4829-bd70-904eacb1e933
MW	My Web Application	Web app / API	17c41b86-853a-41ca-8446-92035a3699b3



# Creating a Native Application

- Power BI supports Native applications
  - Can be used for desktop applications and Console applications
  - Used for third party embedding (known as **App Owns Data** model)
  - Application type should be configured as **Native**
  - Requires Redirect URI with unique string - not an actual URL



The screenshot shows a 'Create' dialog box with a dark header bar containing the text 'Create' and window control icons. The breadcrumb path at the top reads 'Home > critical path training labs - App registrations > Create'. The form contains three fields, each with a red asterisk and an information icon:

- Name**: A text input field containing 'My Native Application' with a green checkmark to its right.
- Application type**: A dropdown menu showing 'Native' with a downward arrow.
- Redirect URI**: A text input field containing 'https://localhost/app1234' with a green checkmark to its right.

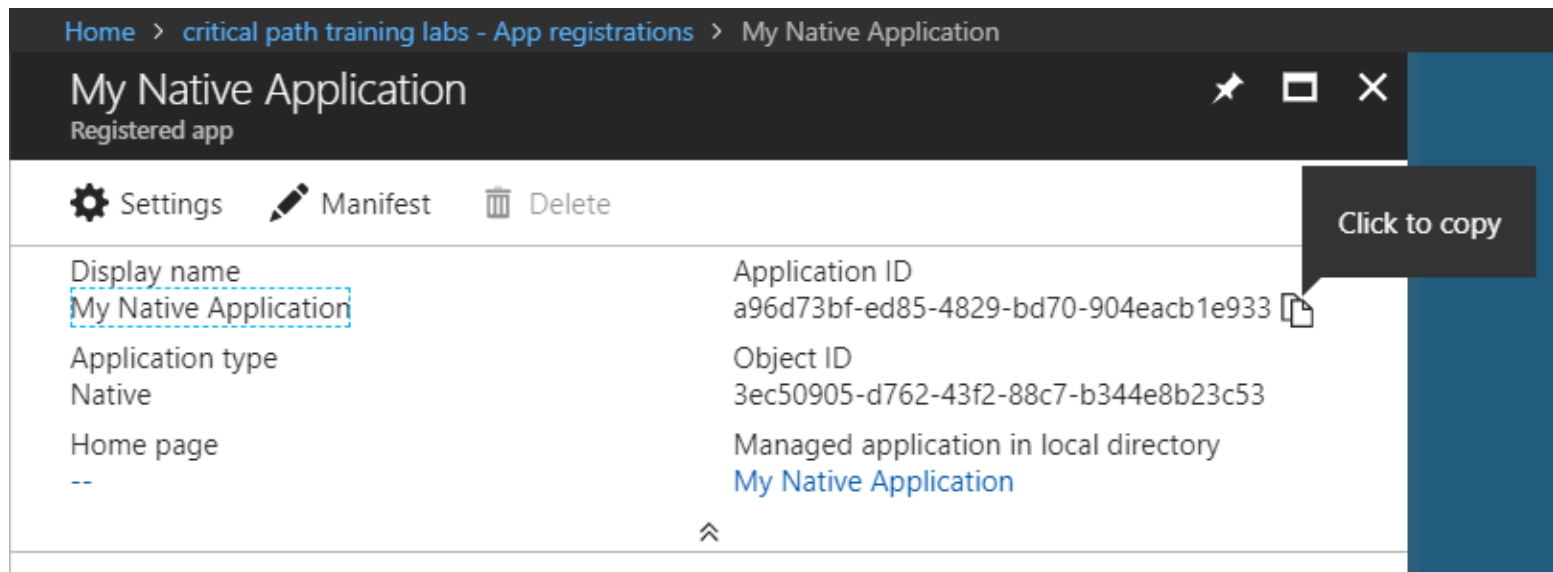
A blue 'Create' button is located at the bottom left of the dialog.





# Copying the Application ID

- Each new application created with Application ID
  - You cannot supply your own GUID for application ID
  - Azure AD will always create this GUID
  - You can copy the application IS from the azure portal



Home > critical path training labs - App registrations > My Native Application

## My Native Application

Registered app

Settings Manifest Delete

Display name	Application ID
My Native Application	a96d73bf-ed85-4829-bd70-904eacb1e933
Application type	Object ID
Native	3ec50905-d762-43f2-88c7-b344e8b23c53
Home page	Managed application in local directory
--	<a href="#">My Native Application</a>

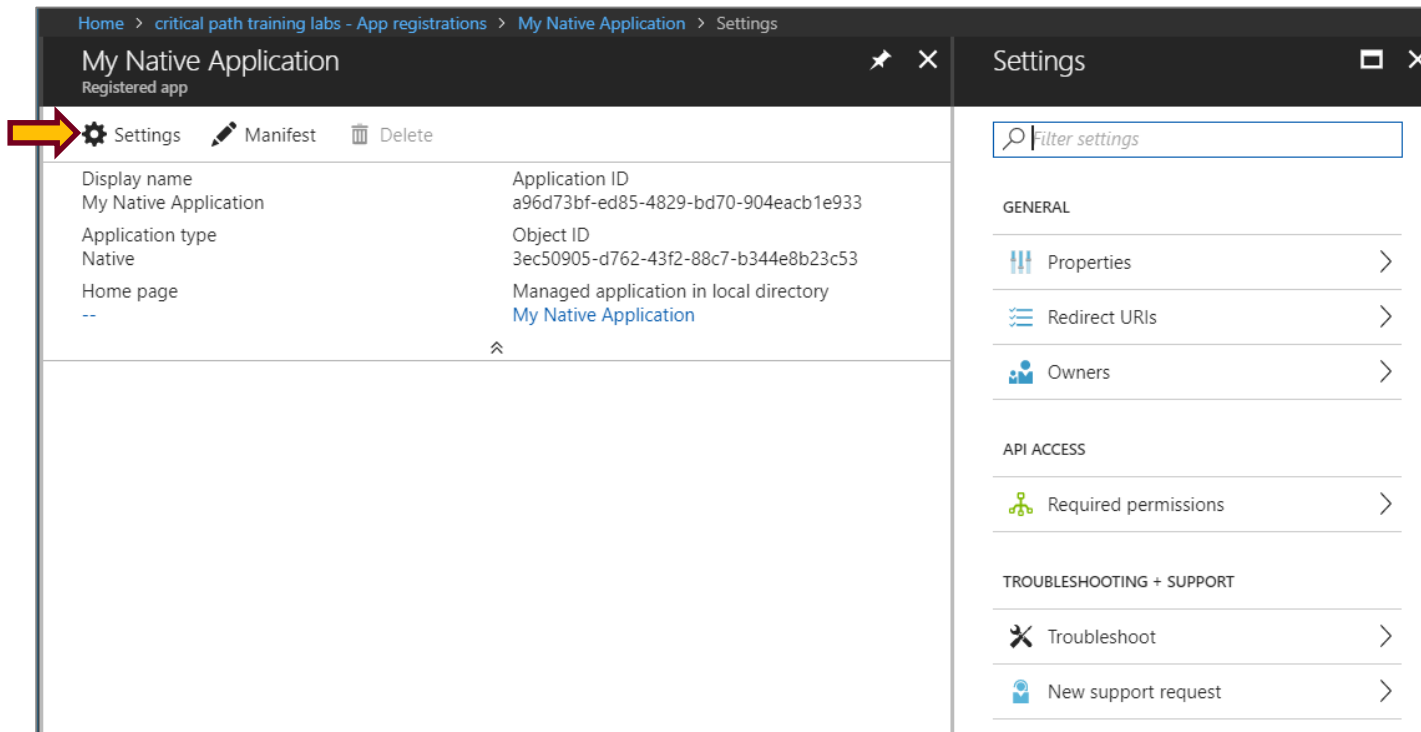
⌵

Click to copy



# Native Application Settings

- Properties
- Redirect URLs
- Owners
- Required Permissions



The screenshot displays the Azure portal interface for managing a native application. The breadcrumb navigation at the top reads: Home > critical path training labs - App registrations > My Native Application > Settings. The main header shows the application name 'My Native Application' and its status as a 'Registered app'. Below the header, there are three tabs: 'Settings' (highlighted with a red arrow), 'Manifest', and 'Delete'. The 'Settings' tab is active, showing a table of application properties and a right-hand pane with a list of settings categories.

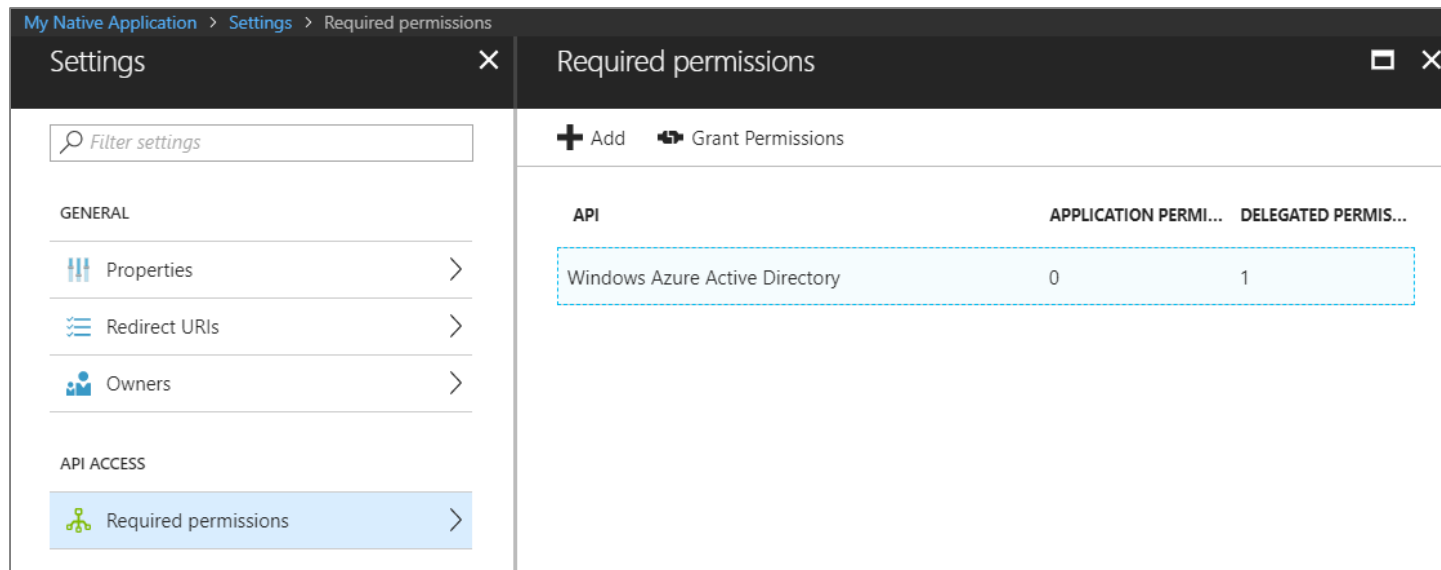
Property	Value
Display name	My Native Application
Application ID	a96d73bf-ed85-4829-bd70-904eacb1e933
Application type	Native
Object ID	3ec50905-d762-43f2-88c7-b344e8b23c53
Home page	Managed application in local directory
--	<a href="#">My Native Application</a>

The right-hand pane, titled 'Settings', contains a search bar labeled 'Filter settings'. Below it, the settings are organized into sections: GENERAL, API ACCESS, and TROUBLESHOOTING + SUPPORT. Each section lists specific settings with icons and right-pointing arrows.

- GENERAL**
  - Properties
  - Redirect URLs
  - Owners
- API ACCESS**
  - Required permissions
- TROUBLESHOOTING + SUPPORT**
  - Troubleshoot
  - New support request

# Configuring Required Permissions

- Application configured with permissions
  - Default permissions allows user authentication – but that's it
  - To use APIs, you must assign permissions to the application



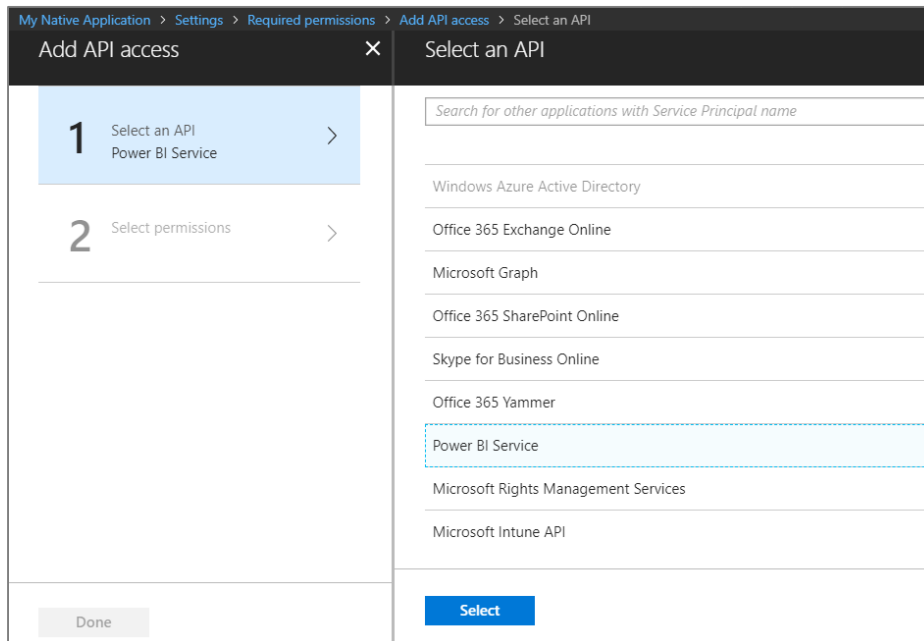
The screenshot shows the 'Required permissions' settings window for a native application. The left sidebar contains a search bar and a list of settings categories: GENERAL (Properties, Redirect URIs, Owners) and API ACCESS (Required permissions). The main panel displays the 'Required permissions' table, which lists the API 'Windows Azure Active Directory' with 0 application permissions and 1 delegated permission. The table is highlighted with a dashed blue border.

API	APPLICATION PERMI...	DELEGATED PERMIS...
Windows Azure Active Directory	0	1



# Choosing APIs

- There are lots of APIs to choose from
  - Office 365 Exchange Online
  - Microsoft Graph
  - Office 365 SharePoint Online
  - Power BI Service



# Delegated Permissions vs Application Permissions

- Permissions categorized into two basic types
  - **Delegated permissions** are (app + user) permissions
  - **Application permissions** are app-only permissions (*far more powerful*)
  - Not all application types and APIs support application permissions
  - Power BI Service API does not yet support application permissions
- Example permissions for Office 365 SharePoint Online
  - Note that some delegated permissions requires administrative permissions

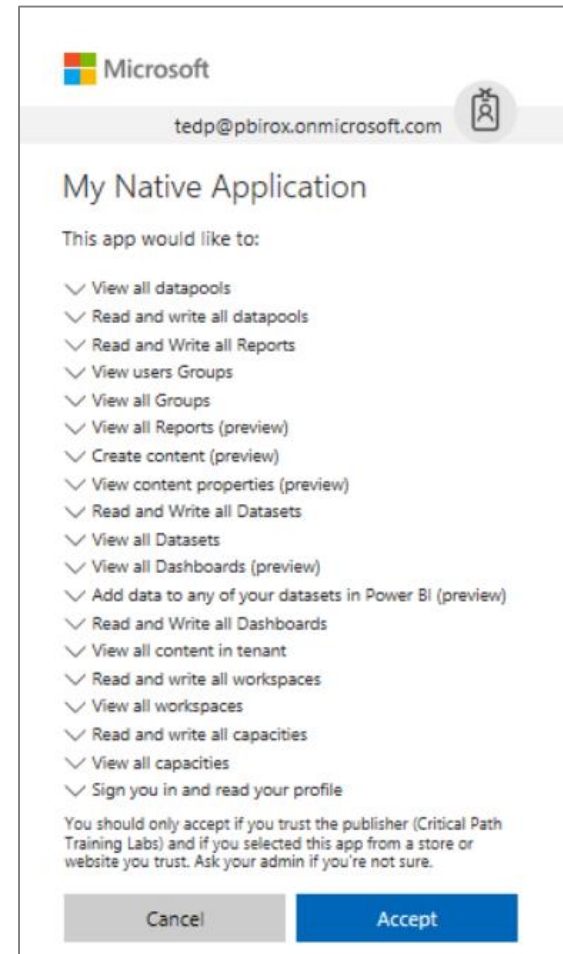
<input type="checkbox"/> DELEGATED PERMISSIONS	REQUIRES ADMIN
Run search queries as a user	✓ Yes
Read user profiles	✓ Yes
<input checked="" type="checkbox"/> Read user files	✗ No
Read managed metadata	✓ Yes
<input checked="" type="checkbox"/> Read items in all site collections	✗ No
Read and write user profiles	✓ Yes
<input checked="" type="checkbox"/> Read and write user files	✗ No
Read and write managed metadata	✓ Yes
<input checked="" type="checkbox"/> Read and write items in all site collections	✗ No
<input checked="" type="checkbox"/> Read and write items and lists in all site collections	✗ No
Have full control of all site collections	✓ Yes

APPLICATION PERMISSIONS	REQUIRES ADMIN
Read user profiles	✓ Yes
Read and write user profiles	✓ Yes
Read and write managed metadata	✓ Yes
Read managed metadata	✓ Yes
Read and write items and lists in all site collections	✓ Yes
Have full control of all site collections	✓ Yes
Read items in all site collections	✓ Yes
Read and write items in all site collections	✓ Yes



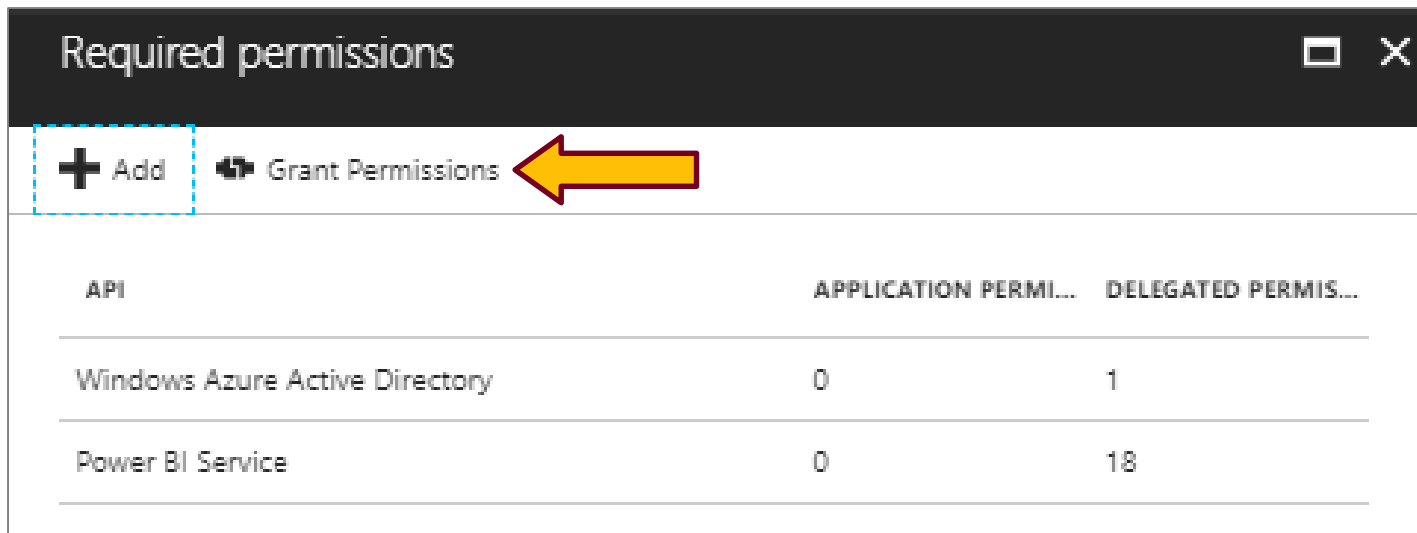
# Interactive Consent for Delegated Permissions

- Users must consent to delegated permissions
  - User prompted during first log in
  - User must click **Accept**
  - Only occurs once for each user



# Granting Delegated Permissions

- It can be helpful to Grant Permissions in Azure portal
  - Prevents the need for interactive granting of application by user
  - Might be required when authenticating in non-interactive fashion



The screenshot shows a 'Required permissions' dialog box. At the top, there are two buttons: '+ Add' and 'Grant Permissions'. A yellow arrow points to the 'Grant Permissions' button. Below the buttons is a table with three columns: 'API', 'APPLICATION PERMI...', and 'DELEGATED PERMIS...'. The table lists two APIs: 'Windows Azure Active Directory' and 'Power BI Service'.

API	APPLICATION PERMI...	DELEGATED PERMIS...
Windows Azure Active Directory	0	1
Power BI Service	0	18





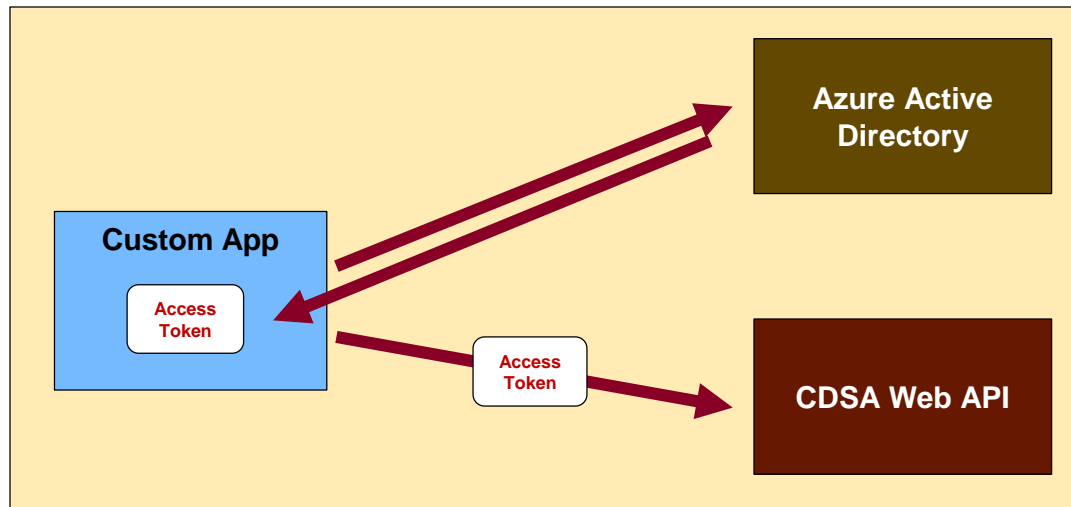
# Agenda

- ✓ CDS for Apps Web API Overview
- ✓ Authentication with Azure Active Directory
- Programming the CDS for Apps Web API
  - Authentication in a Web App



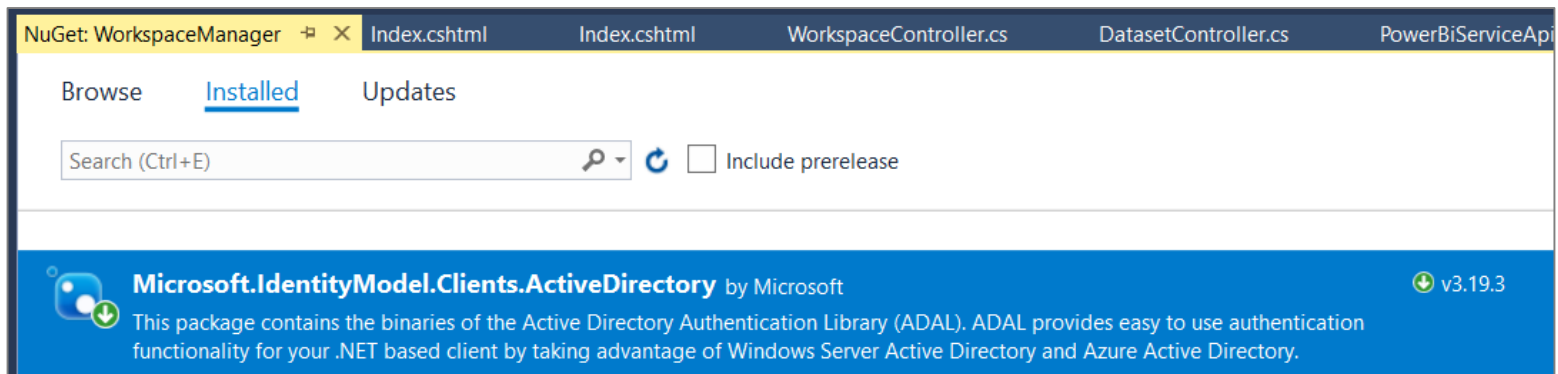
# Authenticating with Azure AD

- User must be authenticated against Azure AD
  - User authentication used to obtain access token
  - Can be accomplished with the Azure AD Authentication Library
  - Access token pass to CDSA Web API in call REST calls



# ADAL for .NET

- Active Directory Authentication Library for .NET
  - Used in Native Clients and in Web Clients
  - Handles authentication flow behind the scenes
  - Provides caching for access tokens and refresh tokens
- ADAL .NET installs as a NuGet Package
  - Package name is `Microsoft.IdentityModel.Clients.ActiveDirectory`



# Access Token Acquisition

```
private static string aadInstance = "https://login.microsoftonline.com/";
private static string resourceUrlPowerBi = "https://analysis.windows.net/powerbi/api";
private static string urlPowerBiRestApiRoot = "https://api.powerbi.com/";

private static string clientId = ConfigurationManager.AppSettings["client-id"];
private static string clientSecret = ConfigurationManager.AppSettings["client-secret"];
private static string redirectUrl = ConfigurationManager.AppSettings["reply-url"];

private static async Task<string> GetAccessTokenAsync() {

    // determine authorization URL for current tenant
    string tenantID = ClaimsPrincipal.Current.FindFirst("http://schemas.microsoft.com/identity/claims/tenantid").Value;
    string tenantAuthority = aadInstance + tenantID;

    // create ADAL cache object
    ApplicationDbContext db = new ApplicationDbContext();
    string signedInUserID = ClaimsPrincipal.Current.FindFirst(ClaimTypes.NameIdentifier).Value;
    ADALTokenCache userTokenCache = new ADALTokenCache(signedInUserID);

    // create authentication context
    AuthenticationContext authenticationContext = new AuthenticationContext(tenantAuthority, userTokenCache);

    // create client credential object using client ID and client Secret";
    ClientCredential clientCredential = new ClientCredential(clientId, clientSecret);

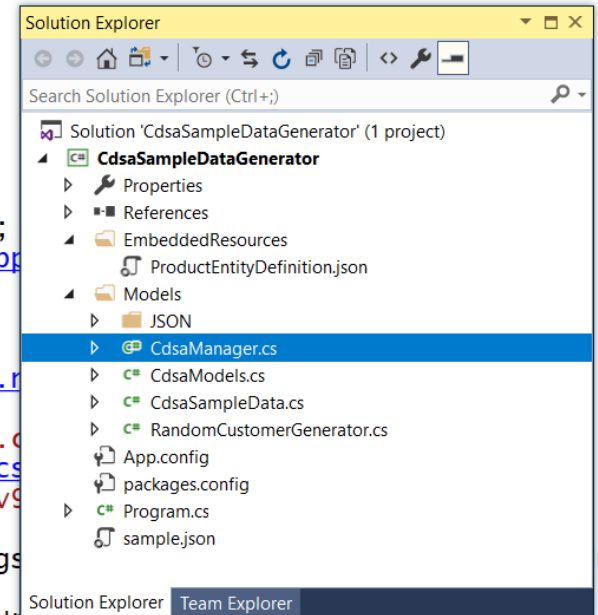
    // create user identifier object for logged on user
    string objectIdentifierId = "http://schemas.microsoft.com/identity/claims/objectidentifier";
    string userObjectID = ClaimsPrincipal.Current.FindFirst(objectIdentifierId).Value;
    UserIdentifier userIdentifier = new UserIdentifier(userObjectID, UserIdentifierType.UniqueId);

    // get access token for Power BI Service API from AAD
    AuthenticationResult authenticationResult =
        await authenticationContext.AcquireTokenSilentAsync(
            resourceUrlPowerBi,
            clientCredential,
            userIdentifier);

    // return access token back to user
    return authenticationResult.AccessToken;
}
```

# Demo Time

```
namespace cdsaSampleDataGenerator.Models {  
    class CdsaManager {  
        //  
        const string cdsaInstanceId = "org4bbc0847";  
  
        const string clientId = "0a36422b-60fc-462b-9cfb-4bbfdefb6147";  
        static readonly Uri redirectUri = new Uri("https://localhost/app");  
  
        #region "Grungy stuff"  
  
        const string aadAuthorizationEndpoint = "https://login.windows.r";  
  
        const string cdsaInstanceUrl = "https://" + cdsaInstanceId + ".c";  
        const string cdsDiscoveryUrl = "https://globaldisco.crm.dynamics";  
        const string cdsaWebApiBaseUrl = cdsaInstanceUrl + "/api/data/v9";  
  
        static readonly JsonSerializerSettings jsonSerializationSettings;  
  
        static string GetAccessToken(string resourceUri = cdsaInstanceUrl) {  
  
            var authContext = new AuthenticationContext(aadAuthorizationEndpoint);  
            var promptBehavior = new PlatformParameters(PromptBehavior.SelectAccount);  
        }  
    }  
}
```



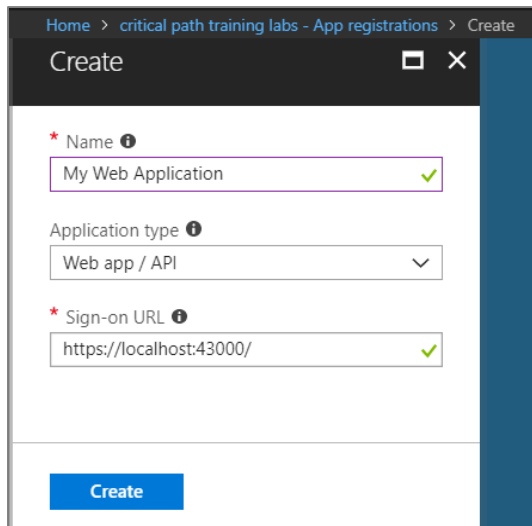
# Agenda

- ✓ CDS for Apps Web API Overview
- ✓ Authentication with Azure Active Directory
- ✓ Programming the CDS for Apps Web API
- Authentication in a Web App



# Creating Applications for Web Applications

- Web applications more secure than native applications
  - Requires Redirect URI which improves security
  - Authentication can be used on client secret (application password)
  - Can use application permissions – Native applications cannot



Home > critical path training labs - App registrations > Create

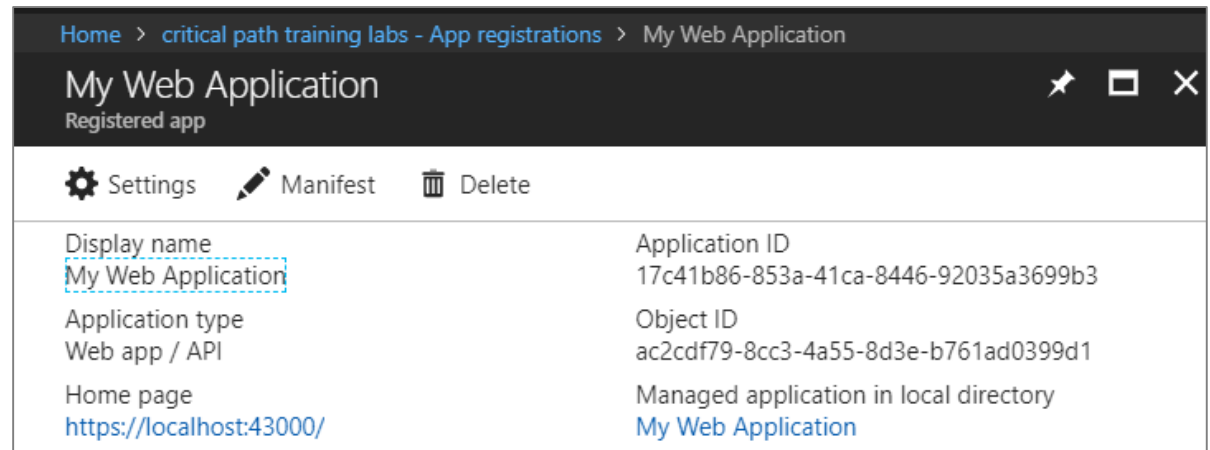
Create

\* Name ⓘ  
My Web Application ✓

Application type ⓘ  
Web app / API ▾

\* Sign-on URL ⓘ  
https://localhost:43000/ ✓

Create



Home > critical path training labs - App registrations > My Web Application

My Web Application  
Registered app

⚙ Settings ✎ Manifest 🗑 Delete






Display name	Application ID
My Web Application	17c41b86-853a-41ca-8446-92035a3699b3
Application type	Object ID
Web app / API	ac2cdf79-8cc3-4a55-8d3e-b761ad0399d1
Home page	Managed application in local directory
https://localhost:43000/	My Web Application





# Reply URLs

- Reply URLs required for web applications
  - Your application must be accessible through the reply URL
  - Provides extra security dimension not available to native apps
  - Application can be configured with multiple reply URLs for single
  - Application must pass Reply URL matching registered Reply URL

Settings	×	Reply URLs
<input type="text" value="Filter settings"/>		 Save  Discard
<b>GENERAL</b>		<input type="text" value="https://localhost:43000/"/>
 Properties >		<input type="text" value="https://MyProductionApp.Azurewebsites.net"/>
 Reply URLs >		<input type="text"/>
 Owners >		



# Creating Keys for Application Passwords

- Web applications authenticate using keys
  - Key acts as application-level password
  - Application requires copy of key value

Settings

×

GENERAL

Properties >

Reply URLs >

Owners >

API ACCESS

Required permissions >

Keys >

Keys

Save Discard Upload Public Key

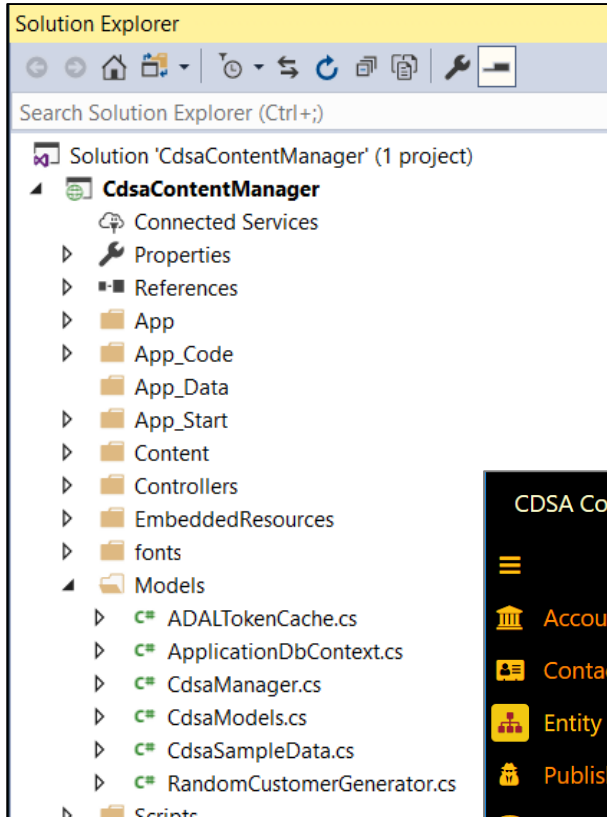
⚠ Copy the key value. You won't be able to retrieve after you leave this blade.

Passwords

DESCRIPTION	EXPIRES	VALUE
My App Password	4/14/2020	O4p6IP2hi5PnB3k92xD9z/qWqCZBz6iMDRDkpx+VWos=
<input type="text" value="Key description"/>	<input type="text" value="Duration"/> <input type="button" value="v"/>	<input type="text" value="Value will be displayed on save"/>



# Demo Time



CDSA Content Manager		Environments		Hello Ted Pattison! Sign out	
	Display Name	LogicalName	EntitySetName	IsCustomEntity	
Accounts	Player	cre1c_player	cre1c_players	True	
Contacts	Solution Component Data Source	msdyn_solutioncomponentdatasource	msdyn_solutioncomponentdatasources	True	
Entity Definitions	OData v4 Data Source	msdyn_odatav4ds	msdyn_odatav4ds	True	
Publishers	Sports Team	cre1c_sportsteam	cre1c_sportsteams	True	
Environments	Solution Component Summary	msdyn_solutioncomponentsummary	msdyn_solutioncomponentsummaries	True	

# Summary

- ✓ CDS for Apps Web API Overview
- ✓ Authentication with Azure Active Directory
- ✓ Programming the CDS for Apps Web API
- ✓ Authentication in a Web App

