polaris

Polaris and OAuth 2.0 with OpenID Connect Integration Guide

7.4

innovative
Part of Clarivate

Innovative Interfaces Incorporated
3133 W. Frye Road, Suite 400
Chandler, AZ 85226
+1.510.655.6200

Legal Notices

Contents

Introduction

Polaris System Administration (web-based) requires OAuth 2.0 with OpenID Connect and Proof Key for Code Exchange (PKCE). As of version 7.2, Leap supports using OAuth 2.0 with OpenID.

Staff authentication for Polaris System Administration (web-based) is handled by either Active Directory and Active Directory Federation Services (AD FS), or by Azure Active Directory (Azure AD).

Staff authentication for Leap is handled by either Active Directory and AD FS, or by Azure AD.

Review the Minimum Requirements for the configuration you plan to use:

- Polaris System Administration (web-based) with Active Directory and AD FS Authentication
- Polaris System Administration (web-based) with Azure AD authentication
- Leap with Active Directory and AD FS authentication
- Leap with Azure AD authentication

Then, continue to one of the following configuration procedures:

- Configuring Active Directory with AD FS
- Configuring Azure AD

Minimum Requirements

This section discusses minimum requirements for the following configurations:

- Polaris System Administration (Web-Based) with Active Directory and AD FS Authentication
- Polaris System Administration (Web-Based) with Azure AD Authentication
- Leap with Active Directory and AD FS Authentication
- Leap with Azure AD Authentication

Polaris System Administration (Web-Based) with Active Directory and AD FS Authentication

To use Polaris System Administration (web-based) with Active Directory and AD FS authentication, you must have the following installed:

- Windows Server 2019 Standard
    - Polaris requires OAuth 2.0 with PKCE support
    - AD FS on Windows Server 2019 supports PKCE
- Active Directory Domain Services
- SSL Certificate
    - Publicly trusted CA signed certificate
- Polaris 7.1 or later
- Polaris 7.1 or later PolarisAdmin

Polaris System Administration (Web-Based) with Azure AD Authentication

To use Polaris System Administration (web-based) with Azure AD authentication, you must have:

- Access to Microsoft's Azure AD services
- The Polaris 7.3 or later PolarisAdmin installed

Leap with Active Directory and AD FS Authentication

To use Leap with Active Directory and AD FS authentication, you must have the following installed:

- Windows Server 2019 Standard
    - Polaris requires OAuth 2.0 with PKCE support
    - AD FS on Windows Server 2019 supports PKCE
- Active Directory Domain Services
- SSL Certificate
    - Publicly trusted CA signed certificate
- Polaris 7.2 or later
- Polaris 7.2 or later LeapWebApp

Leap with Azure AD Authentication

To use Leap with Azure AD authentication, you must have:

- Access to Microsoft's Azure AD services
- The Polaris 7.3 or later LeapWebApp installed

Configuring Active Directory with AD FS

Important:
The mechanism used to connect an Active Directory user to a Polaris user is the user principal name (UPN) in the format of an email address. For example, user@mydomain.com. During the account verification process, we request the UPN claim from Active Directory. This must return a UPN in the name@domain format. The Polaris.AdminServices (API) can then use that information to map the AD user to a Polaris user.

To configure Polaris OAuth support with Active Directory and AD FS, perform the following tasks:

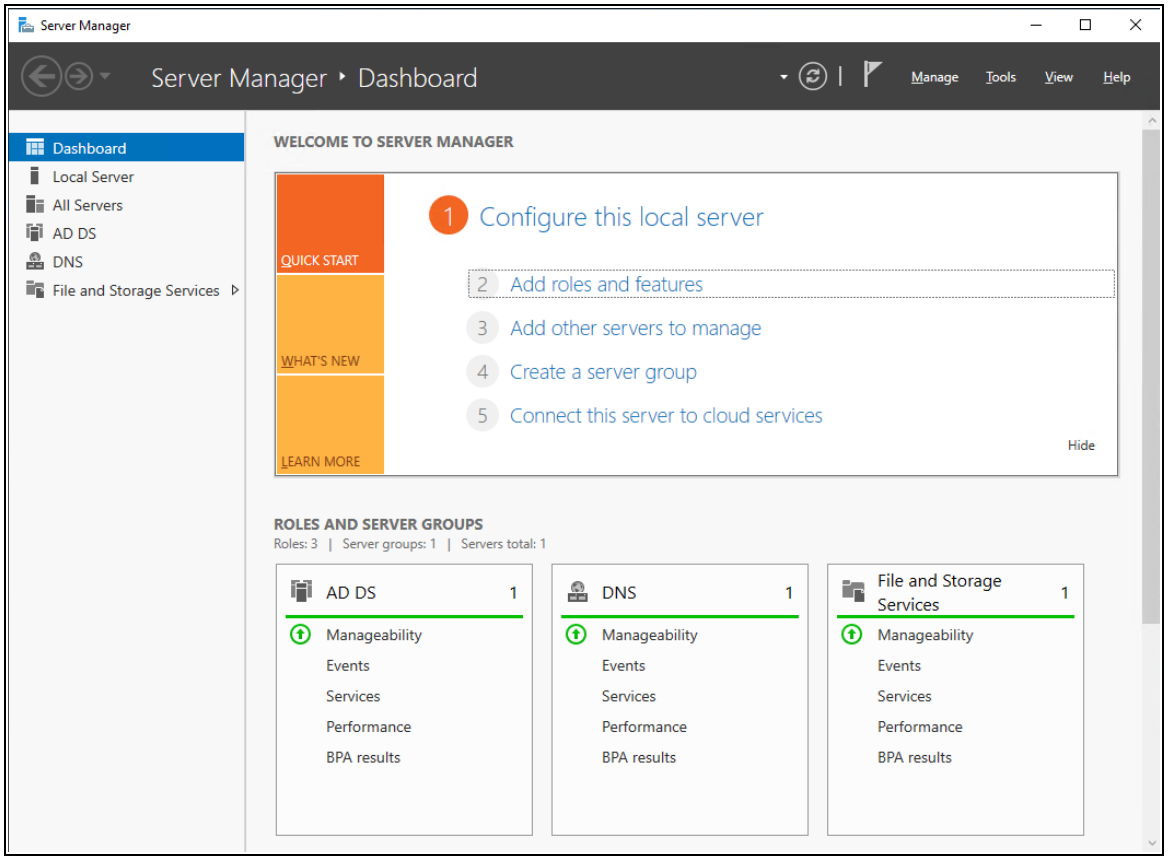
1. Install Active Directory Federation Services.
2. Configure Active Directory Federation Services.
3. Verify that Active Directory Federation Services is running.
4. Verify that OAuth 2.0 is Enabled.
5. Create an Application Group for Polaris LeapWebApp.
6. Configure the AD FS Web Application: Claims and Permitted Scopes.
7. Enable CORS on AD FS to accept requests from Polaris APIs.
8. Set up web services and applications.
9. Enable session storage for LeapWebApp.
10. Customize the AD FS pages.
11. Change the access token lifetime.
12. Bind a new SSL certificate.
13. Troubleshoot.

After you complete these tasks, Add a URL rewrite rule for LeapWebApp.

Install Active Directory Federation Services

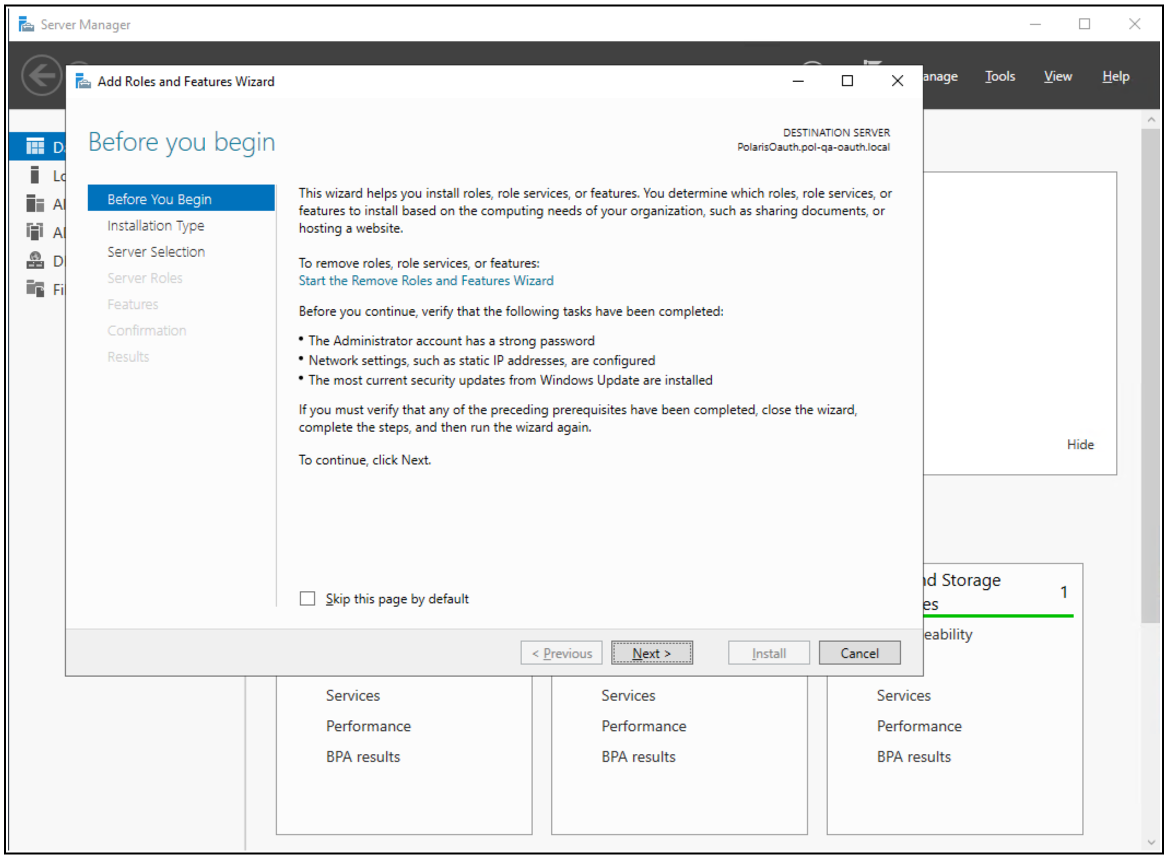
To install AD FS

1. Sign in to Windows Server 2019 with administrative privileges.
2. Start the Server Manager desktop application.



3. On the **Server Manager Dashboard** view, select **Add roles and features**.

   The Add Roles and Features Wizard opens.



This wizard helps you install roles, role services, or features. You determine which roles, role services, or features to install based on the computing needs of your organization, such as sharing documents, or hosting a website.

To remove roles, role services, or features:
Start the Remove Roles and Features Wizard

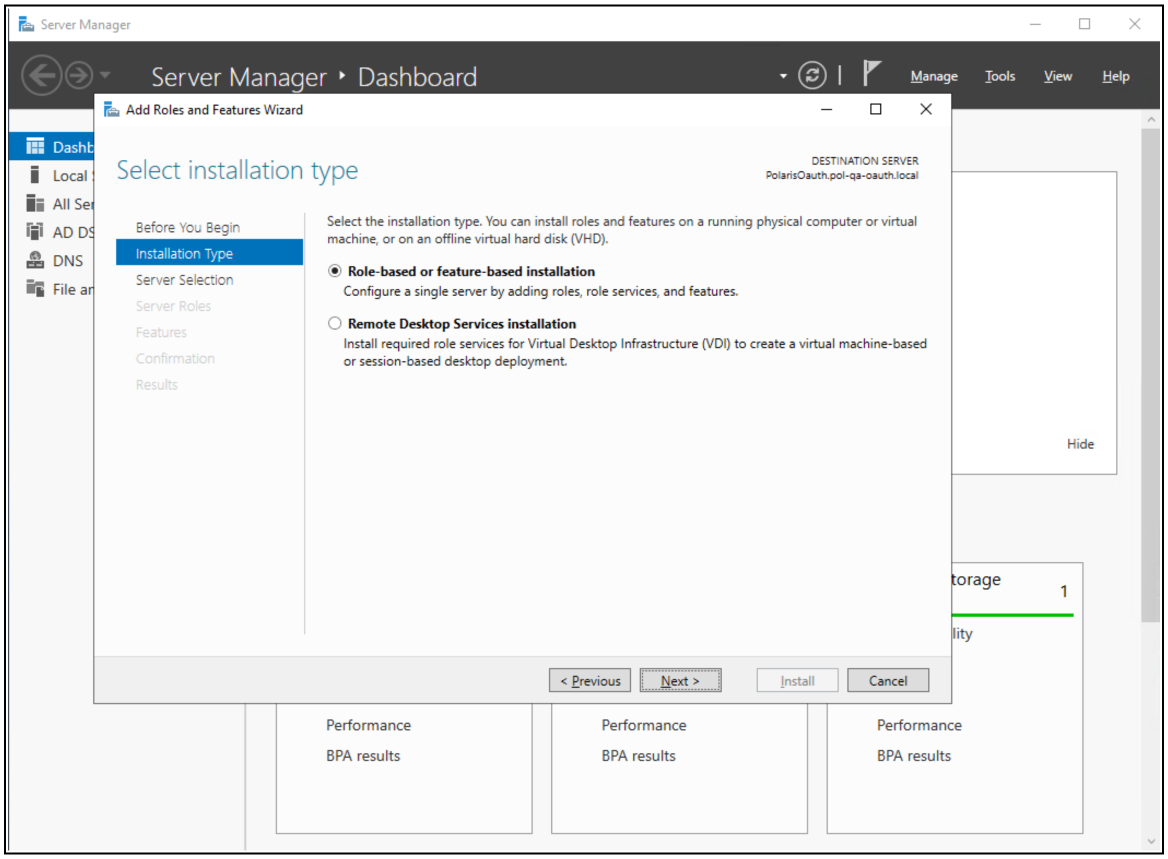
Before you continue, verify that the following tasks have been completed:

- The Administrator account has a strong password
- Network settings, such as static IP addresses, are configured
- The most current security updates from Windows Update are installed

If you must verify that any of the preceding prerequisites have been completed, close the wizard, complete the steps, and then run the wizard again.

To continue, click Next.

4. On the **Before You Begin** tab, select **Next**.



Select installation type

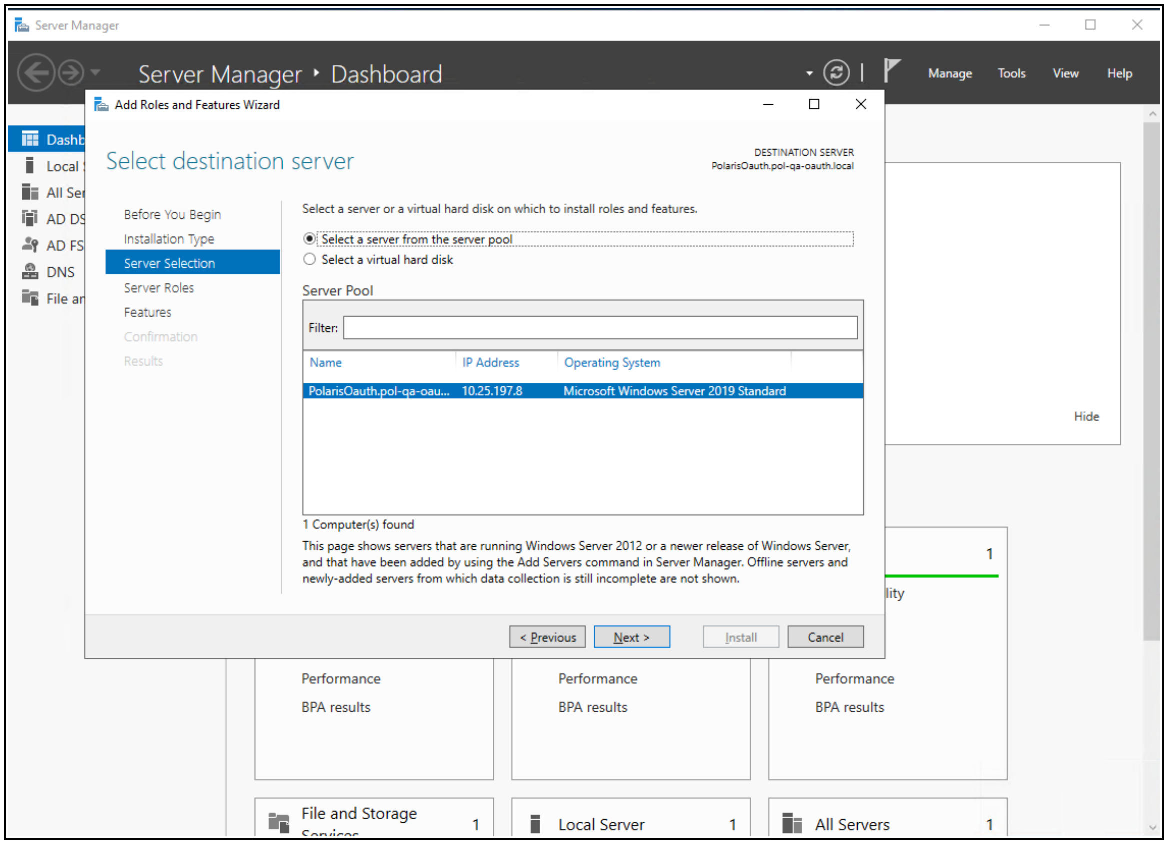
DESTINATION SERVER
PolarisOauth.pol-qa-oauth.local

Select the installation type. You can install roles and features on a running physical computer or virtual machine, or on an offline virtual hard disk (VHD).

● **Role-based or feature-based installation**
Configure a single server by adding roles, role services, and features.

○ **Remote Desktop Services installation**
Install required role services for Virtual Desktop Infrastructure (VDI) to create a virtual machine-based or session-based desktop deployment.

5. On the **Installation Type** tab, select **Role-based or feature-based installation**, and then select **Next**.



Select destination server

DESTINATION SERVER
PolarisOauth.pol-qa-oauth.local

Select a server or a virtual hard disk on which to install roles and features.

(●) Select a server from the server pool
( ) Select a virtual hard disk

Server Pool

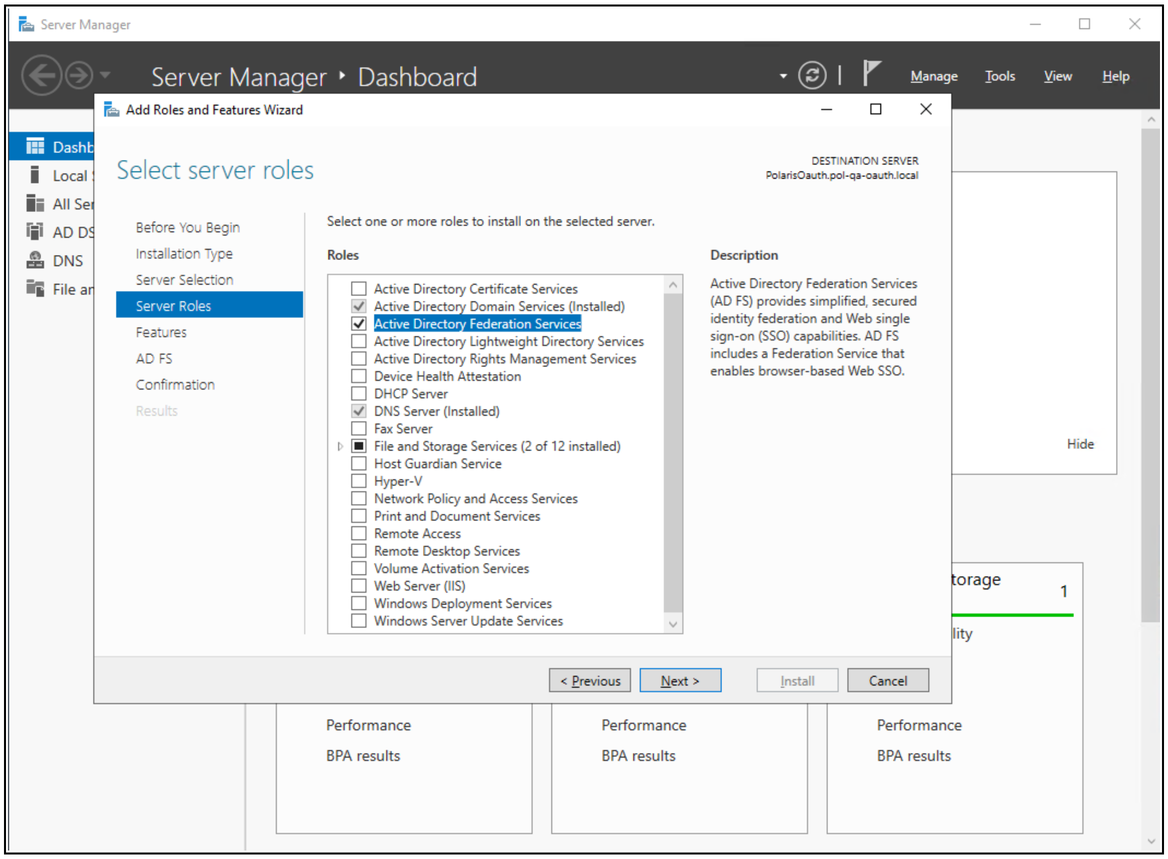
Filter:

Name	IP Address	Operating System
PolarisOauth.pol-qa-oau...	10.25.197.8	Microsoft Windows Server 2019 Standard

1 Computer(s) found

This page shows servers that are running Windows Server 2012 or a newer release of Windows Server, and that have been added by using the Add Servers command in Server Manager. Offline servers and newly-added servers from which data collection is still incomplete are not shown.

6. On the **Server Selection** tab, select the server, and then select **Next**.



Select one or more roles to install on the selected server.

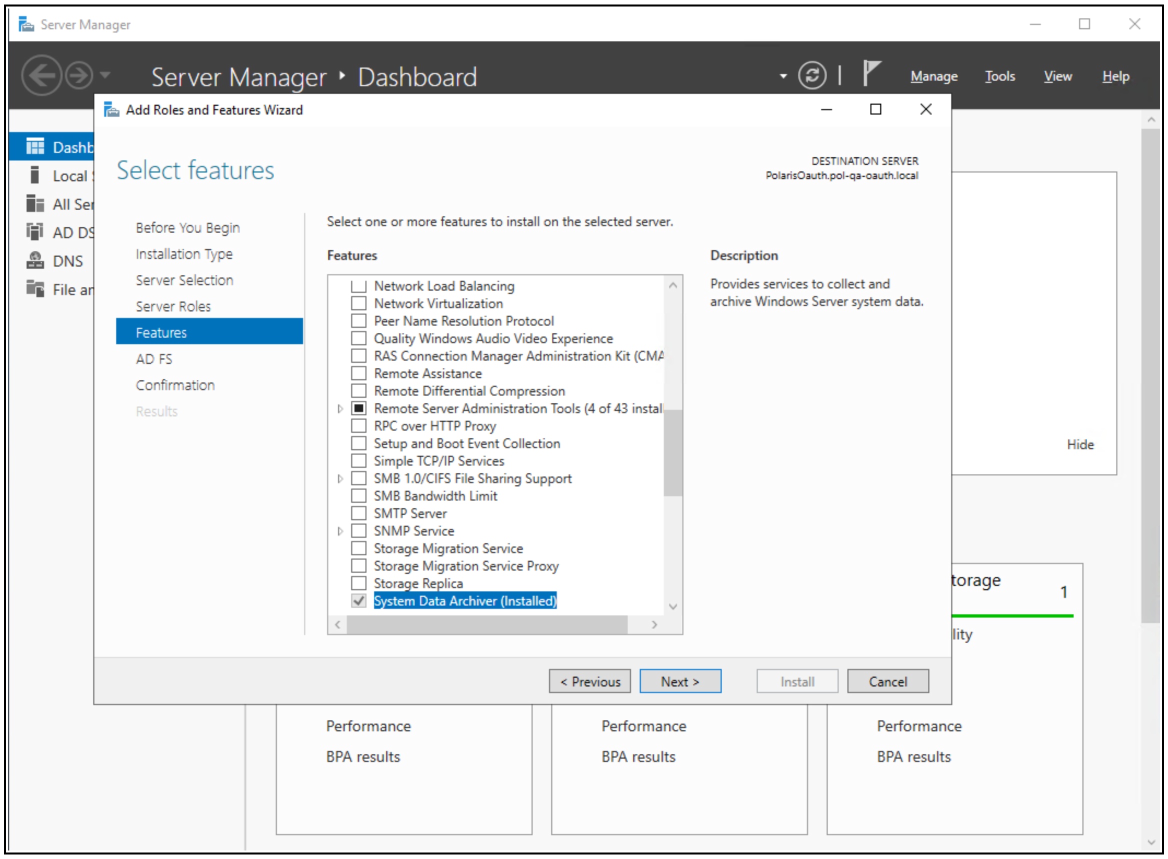
Roles

- [ ] Active Directory Certificate Services
- [✓] Active Directory Domain Services (Installed)
- [✓] Active Directory Federation Services
- [ ] Active Directory Lightweight Directory Services
- [ ] Active Directory Rights Management Services
- [ ] Device Health Attestation
- [ ] DHCP Server
- [✓] DNS Server (Installed)
- [ ] Fax Server
- [■] File and Storage Services (2 of 12 installed)
- [ ] Host Guardian Service
- [ ] Hyper-V
- [ ] Network Policy and Access Services
- [ ] Print and Document Services
- [ ] Remote Access
- [ ] Remote Desktop Services
- [ ] Volume Activation Services
- [ ] Web Server (IIS)
- [ ] Windows Deployment Services
- [ ] Windows Server Update Services

Description

Active Directory Federation Services (AD FS) provides simplified, secured identity federation and Web single sign-on (SSO) capabilities. AD FS includes a Federation Service that enables browser-based Web SSO.

7. On the **Server Roles** tab, do the following:
   a. Verify that **Active Directory Domain Services** are installed.
   b. Select the **Active Directory Federation Services** role.
   c. Select **Next**.



Select features

DESTINATION SERVER
PolarisOauth.pol-qa-oauth.local

Select one or more features to install on the selected server.

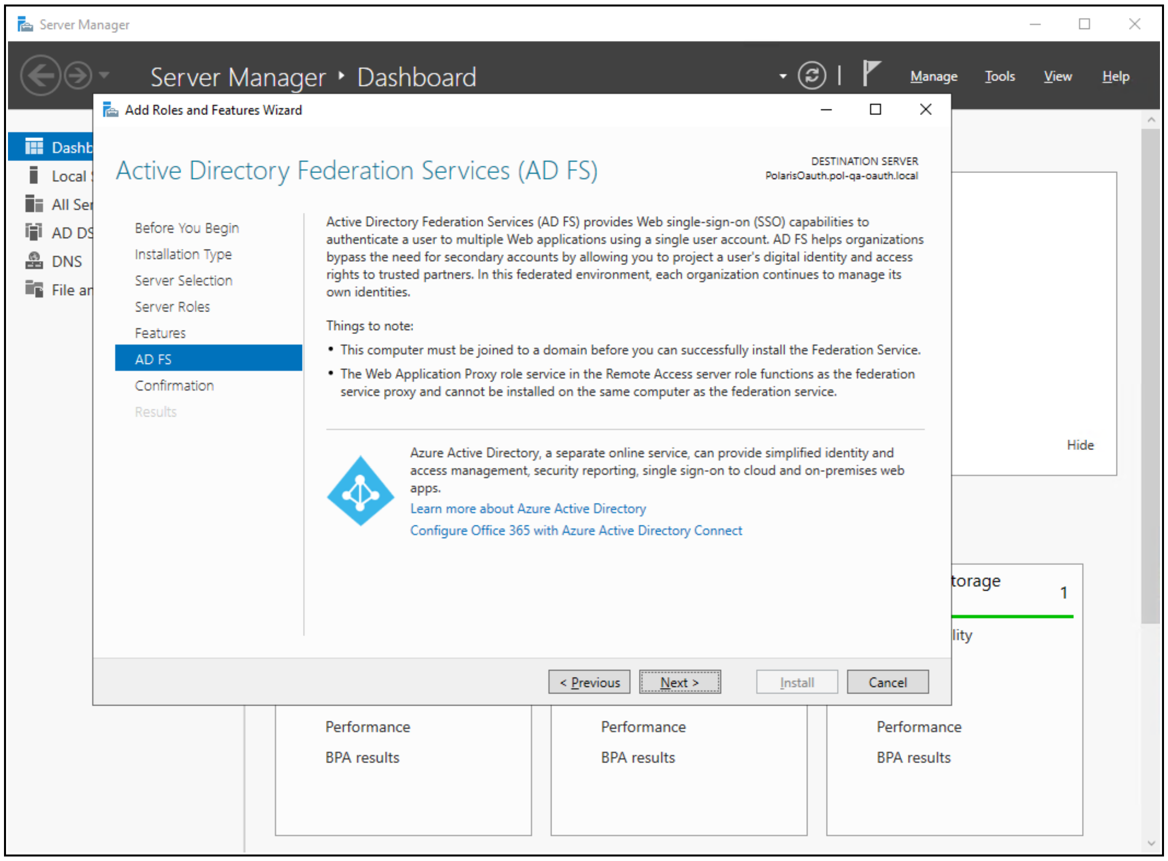
Features

- [ ] Network Load Balancing
- [ ] Network Virtualization
- [ ] Peer Name Resolution Protocol
- [ ] Quality Windows Audio Video Experience
- [ ] RAS Connection Manager Administration Kit (CMA
- [ ] Remote Assistance
- [ ] Remote Differential Compression
- ■ Remote Server Administration Tools (4 of 43 instal
- [ ] RPC over HTTP Proxy
- [ ] Setup and Boot Event Collection
- [ ] Simple TCP/IP Services
- [ ] SMB 1.0/CIFS File Sharing Support
- [ ] SMB Bandwidth Limit
- [ ] SMTP Server
- [ ] SNMP Service
- [ ] Storage Migration Service
- [ ] Storage Migration Service Proxy
- [ ] Storage Replica
- [✓] System Data Archiver (Installed)

Description

Provides services to collect and archive Windows Server system data.

8. On the **Features** tab, select **Next**.



Active Directory Federation Services (AD FS)

DESTINATION SERVER
PolarisOauth.pol-qa-oauth.local

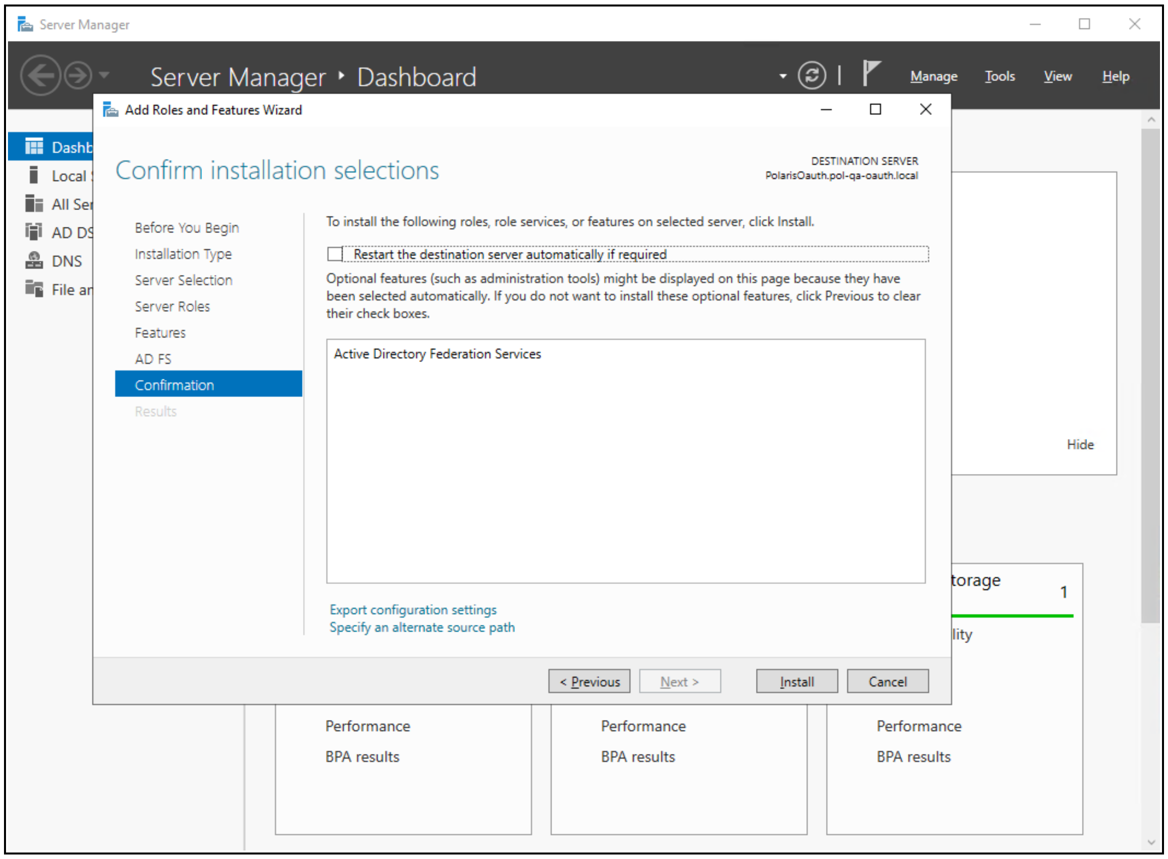
Active Directory Federation Services (AD FS) provides Web single-sign-on (SSO) capabilities to authenticate a user to multiple Web applications using a single user account. AD FS helps organizations bypass the need for secondary accounts by allowing you to project a user's digital identity and access rights to trusted partners. In this federated environment, each organization continues to manage its own identities.

Things to note:

- This computer must be joined to a domain before you can successfully install the Federation Service.
- The Web Application Proxy role service in the Remote Access server role functions as the federation service proxy and cannot be installed on the same computer as the federation service.

Azure Active Directory, a separate online service, can provide simplified identity and access management, security reporting, single sign-on to cloud and on-premises web apps.
Learn more about Azure Active Directory
Configure Office 365 with Azure Active Directory Connect

9. On the **AD FS** tab, read the Active Directory Federation Services (AD FS) information, and then select **Next**.



Confirm installation selections

DESTINATION SERVER
PolarisOauth.pol-qa-oauth.local

To install the following roles, role services, or features on selected server, click Install.

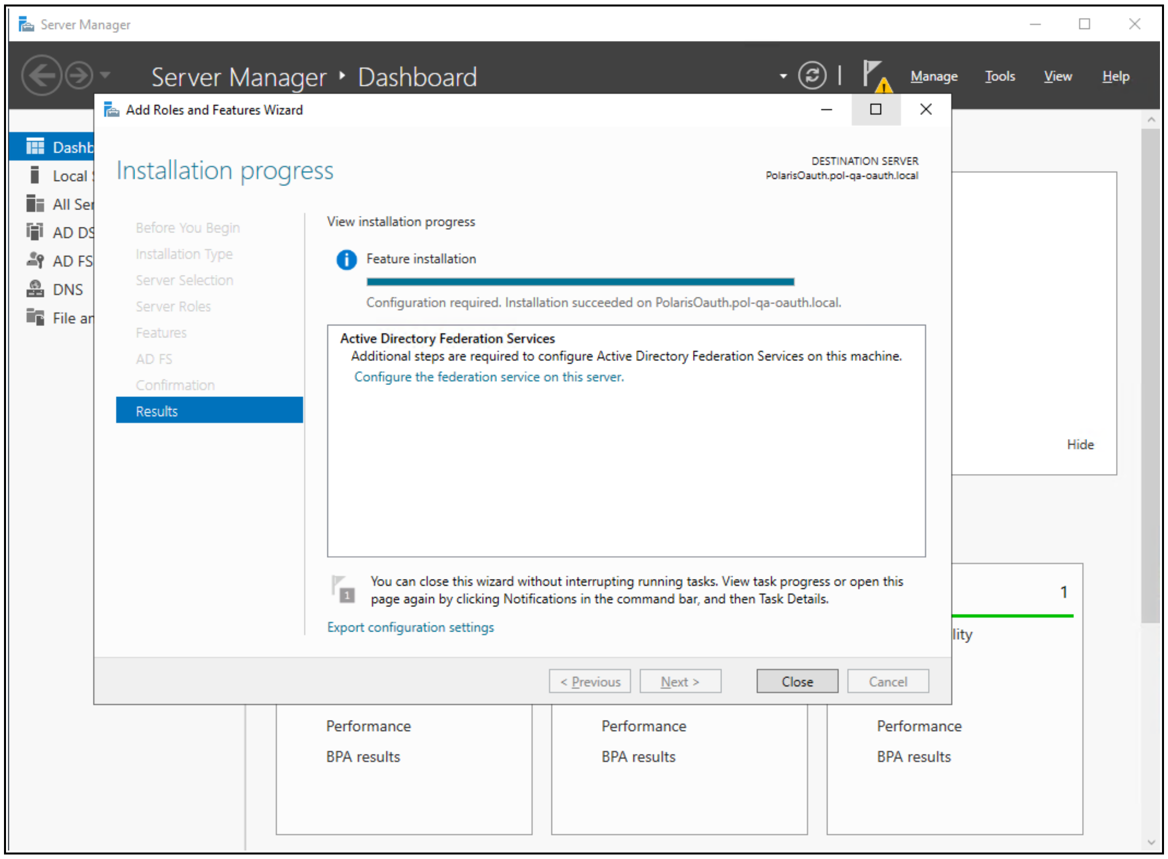
Restart the destination server automatically if required

Optional features (such as administration tools) might be displayed on this page because they have been selected automatically. If you do not want to install these optional features, click Previous to clear their check boxes.

Active Directory Federation Services

Export configuration settings
Specify an alternate source path

10. On the **Confirmation** tab, confirm your selections, and then select **Install**.



Installation progress

DESTINATION SERVER
PolarisOauth.pol-qa-oauth.local

View installation progress

Feature installation

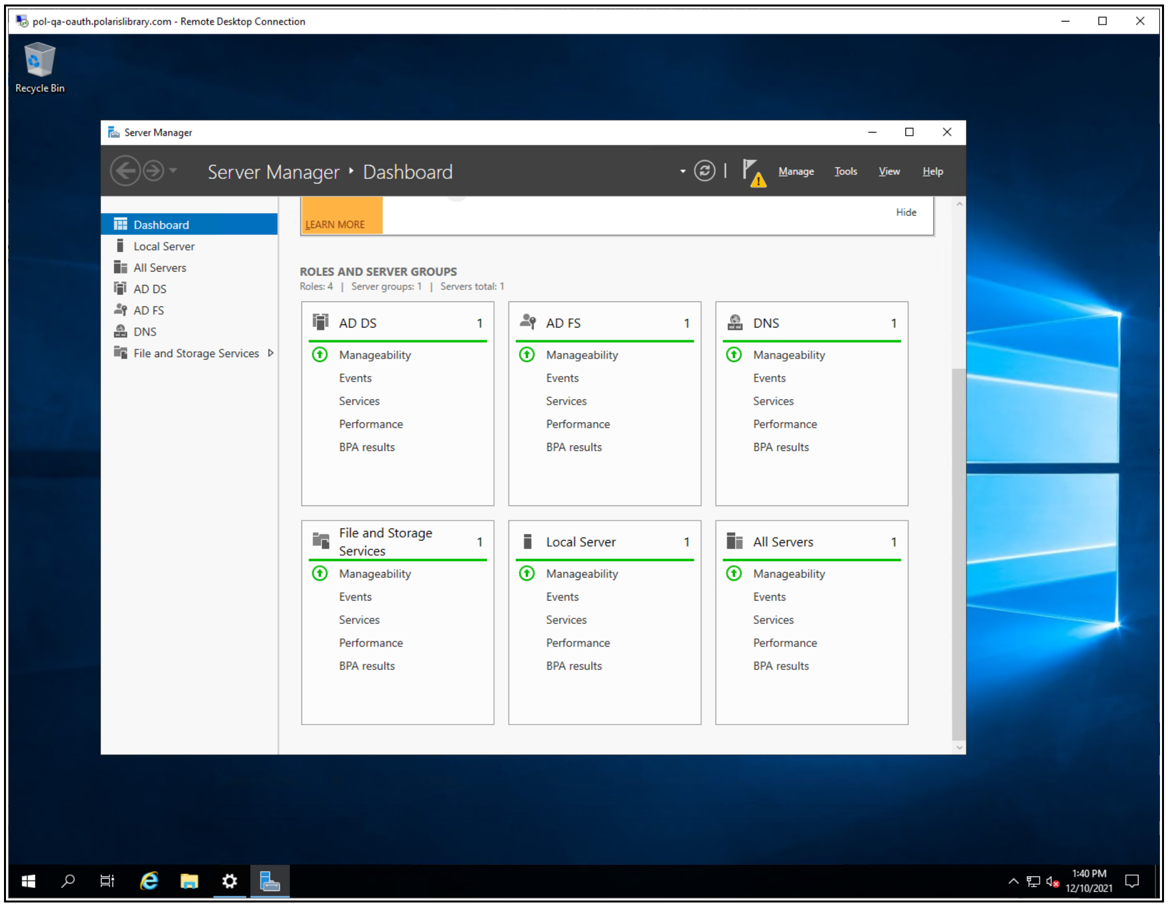
Configuration required. Installation succeeded on PolarisOauth.pol-qa-oauth.local.

Active Directory Federation Services
Additional steps are required to configure Active Directory Federation Services on this machine.
Configure the federation service on this server.

You can close this wizard without interrupting running tasks. View task progress or open this page again by clicking Notifications in the command bar, and then Task Details.

Export configuration settings

11. On the **Results** tab, select **Close** when the installation is complete.



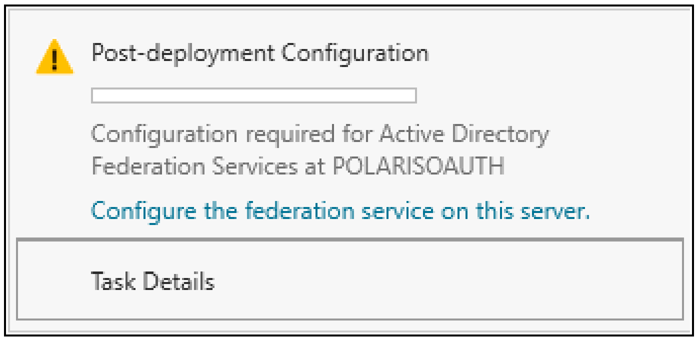
12. On the Server Manager dashboard, verify that AD FS is an installed role.
13. Restart the server.

Configure Active Directory Federation Services

To configure Active Directory Federation Services

1. Start the Server Manager desktop application.

   The system generates a configuration notification.


Post-deployment Configuration

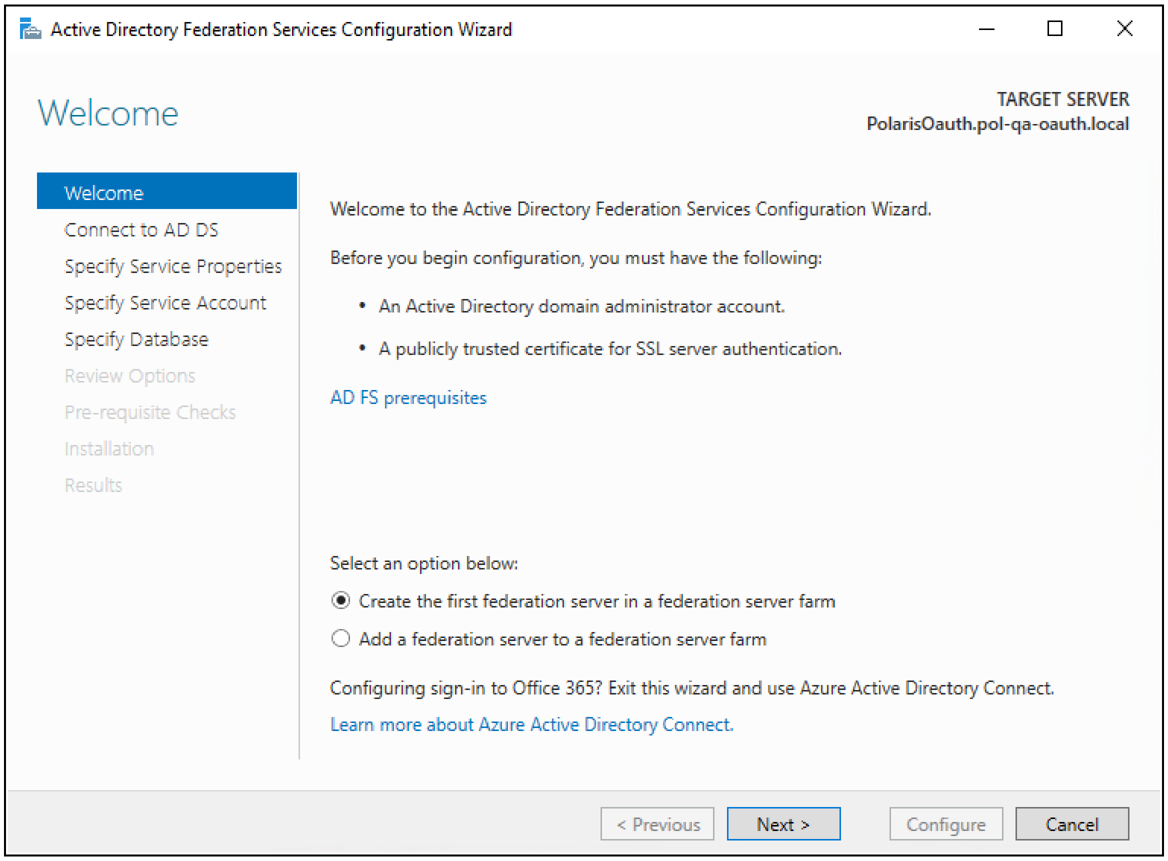
Configuration required for Active Directory Federation Services at POLARISOAUTH

Configure the federation service on this server.

Task Details

2. Open the notification, and select **Configure the federation service on this server**.

   The Active Directory Federation Services Configuration wizard opens.


Active Directory Federation Services Configuration Wizard

Welcome

TARGET SERVER
PolarisOauth.pol-qa-oauth.local

Welcome to the Active Directory Federation Services Configuration Wizard.

Before you begin configuration, you must have the following:

- An Active Directory domain administrator account.
- A publicly trusted certificate for SSL server authentication.

AD FS prerequisites

Select an option below:

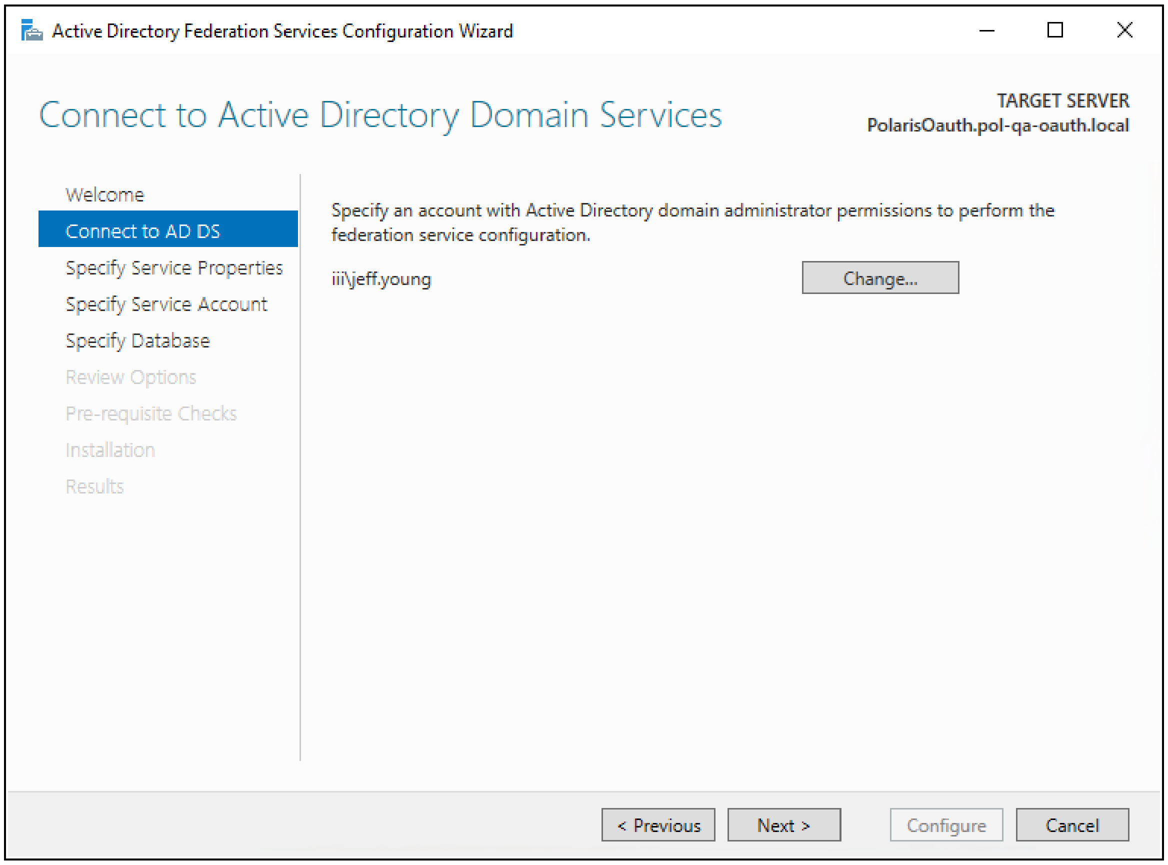
(●) Create the first federation server in a federation server farm

( ) Add a federation server to a federation server farm

Configuring sign-in to Office 365? Exit this wizard and use Azure Active Directory Connect.

Learn more about Azure Active Directory Connect.

3. On the Welcome tab, select **Next**.


Active Directory Federation Services Configuration Wizard

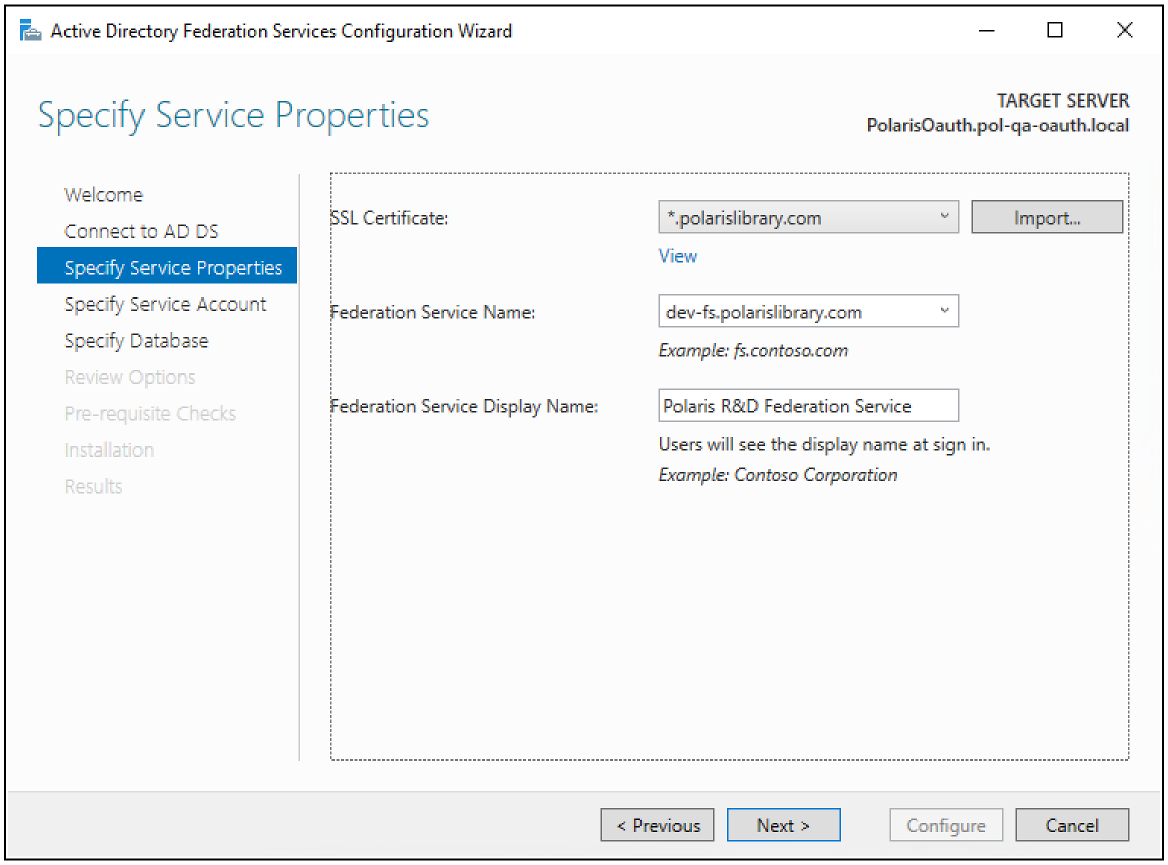
Connect to Active Directory Domain Services

TARGET SERVER
PolarisOauth.pol-qa-oauth.local

Specify an account with Active Directory domain administrator permissions to perform the federation service configuration.

iii\jeff.young

4. Select **Change**, and provide an administrator account. Then select **Next**.


Active Directory Federation Services Configuration Wizard

Specify Service Properties

TARGET SERVER
PolarisOauth.pol-qa-oauth.local

SSL Certificate: *.polarislibrary.com

Federation Service Name: dev-fs.polarislibrary.com

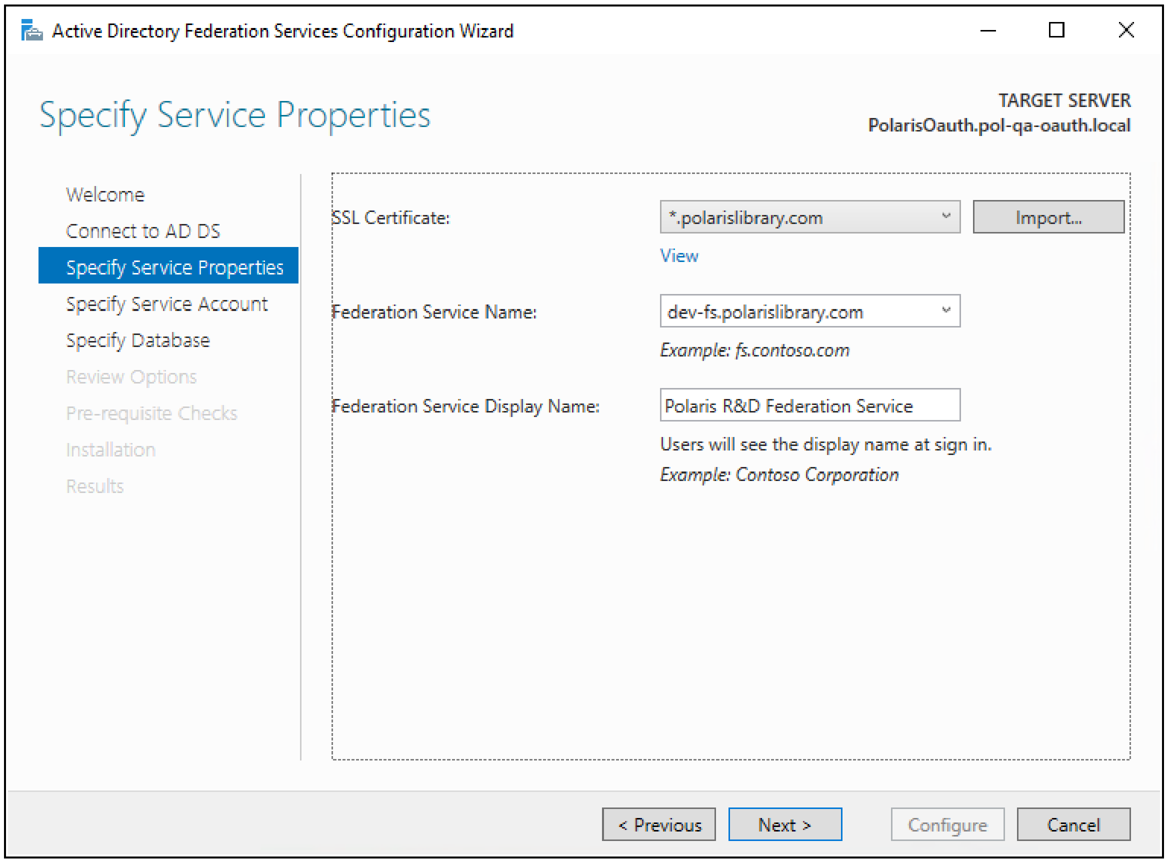
Example: fs.contoso.com

Federation Service Display Name: Polaris R&D Federation Service

Users will see the display name at sign in.

Example: Contoso Corporation

5. If not already installed on the server, select **Import** to install an SSL certificate.



Specify Service Properties

TARGET SERVER
PolarisOauth.pol-qa-oauth.local

SSL Certificate: *.polarislibrary.com

Federation Service Name: dev-fs.polarislibrary.com

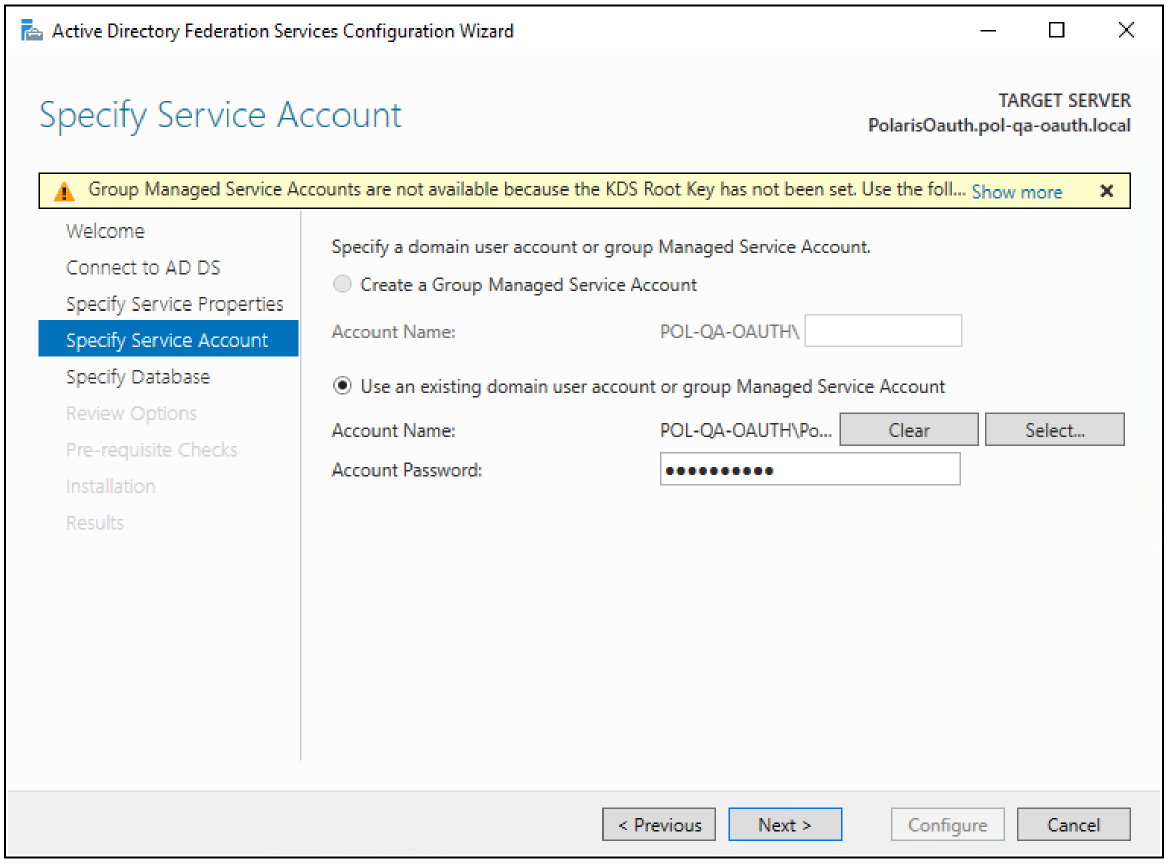
Example: fs.contoso.com

Federation Service Display Name: Polaris R&D Federation Service

Users will see the display name at sign in.

Example: Contoso Corporation

6. Enter the following, and then select **Next**:
   - Federation Service Name
   - Federation Service Display Name


Active Directory Federation Services Configuration Wizard

Specify Service Account

TARGET SERVER
PolarisOauth.pol-qa-oauth.local

Group Managed Service Accounts are not available because the KDS Root Key has not been set. Use the foll... Show more

Specify a domain user account or group Managed Service Account.

( ) Create a Group Managed Service Account

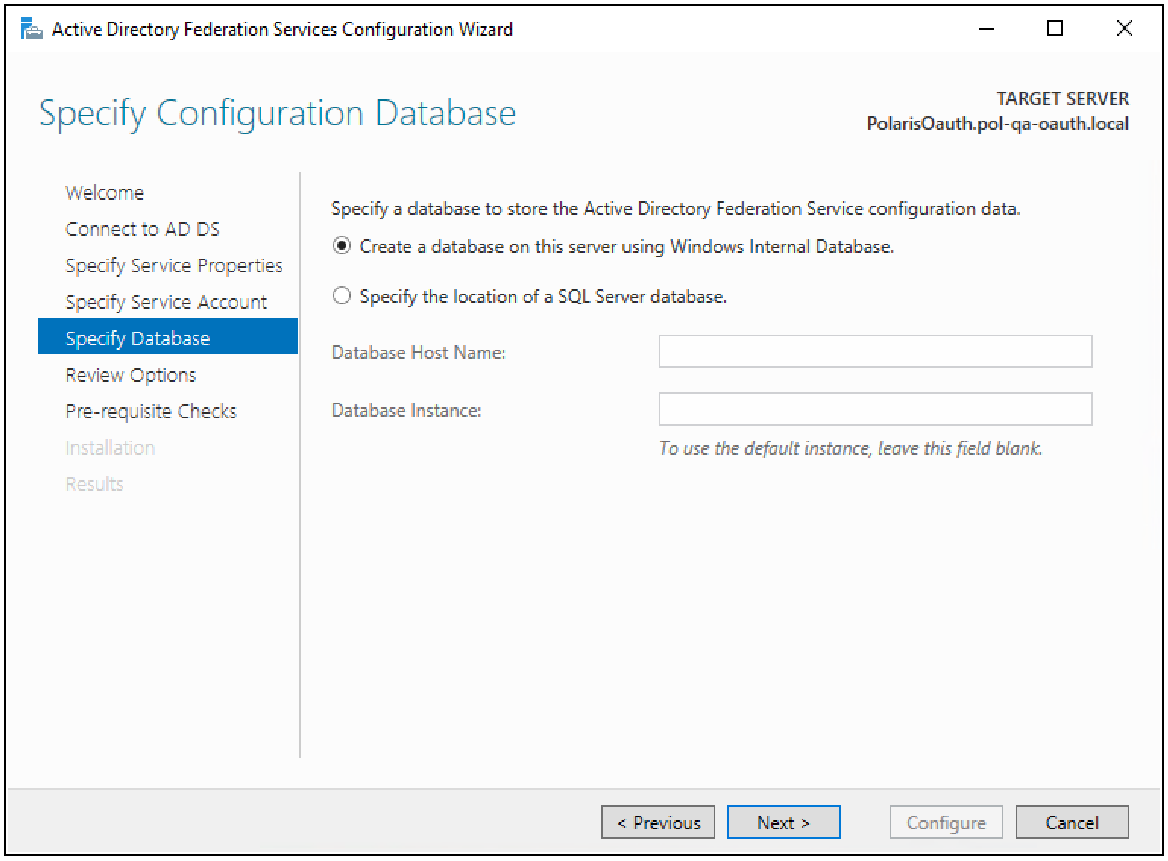
Account Name: POL-QA-OAUTH\

(●) Use an existing domain user account or group Managed Service Account

Account Name: POL-QA-OAUTH\Po...

Account Password: ●●●●●●●●●●●

7. Specify a service account, and then select **Next**.


Active Directory Federation Services Configuration Wizard

Specify Configuration Database

TARGET SERVER
PolarisOauth.pol-qa-oauth.local

Specify a database to store the Active Directory Federation Service configuration data.

[✓ Create a database on this server using Windows Internal Database.]

Specify the location of a SQL Server database.

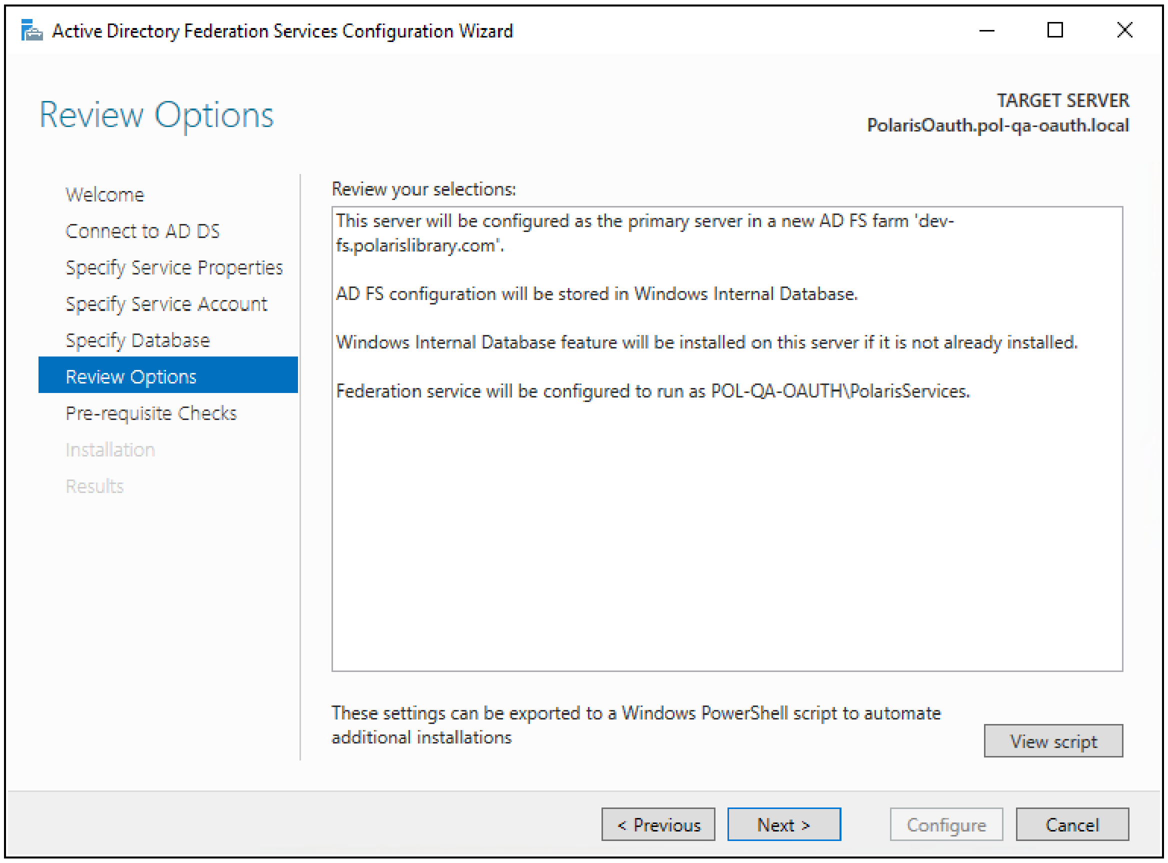
Database Host Name:

Database Instance:

To use the default instance, leave this field blank.

8. Specify the location of the AD FS configuration database, and then select **Next**.

   For simple scenarios, creating the local database is acceptable.



Review your selections:

This server will be configured as the primary server in a new AD FS farm 'dev-fs.polarislibrary.com'.

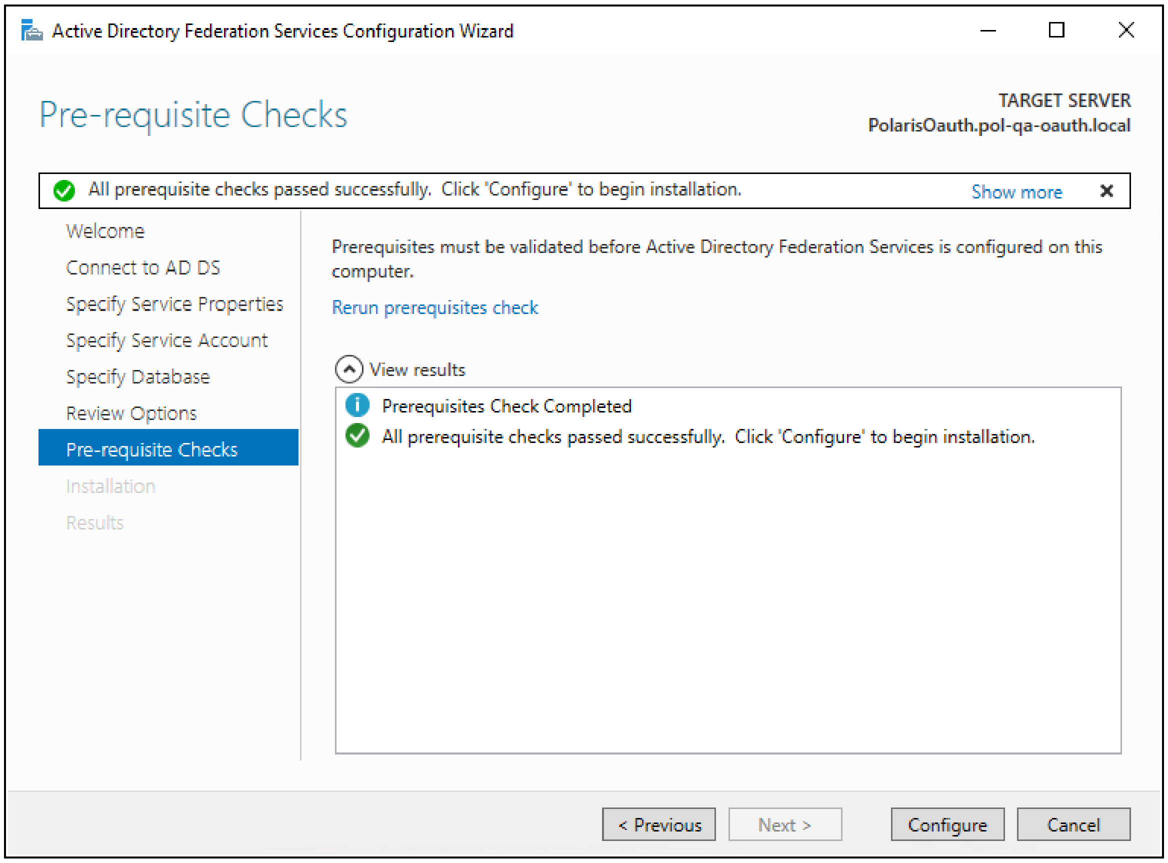
AD FS configuration will be stored in Windows Internal Database.

Windows Internal Database feature will be installed on this server if it is not already installed.

Federation service will be configured to run as POL-QA-OAUTH\PolarisServices.

These settings can be exported to a Windows PowerShell script to automate additional installations

9. Review your selections, and then select **Next**.



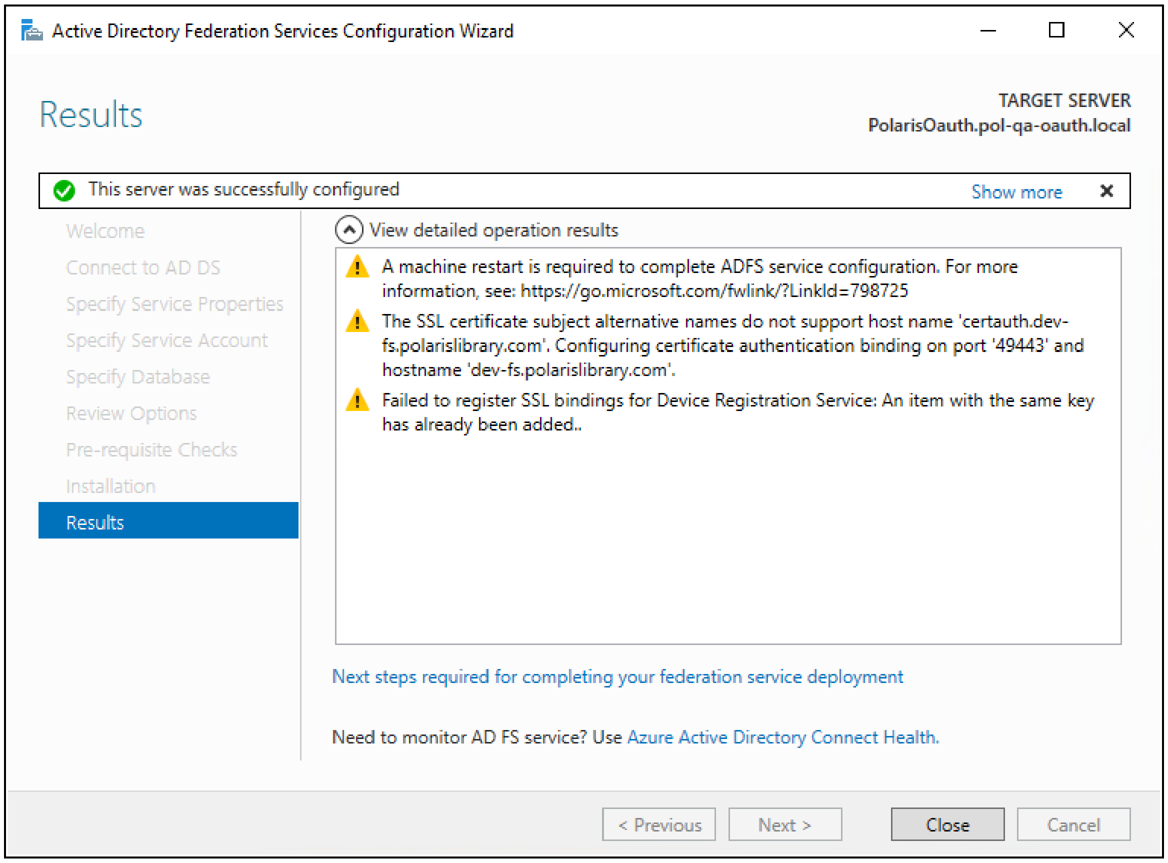
All prerequisite checks passed successfully. Click 'Configure' to begin installation.

Prerequisites must be validated before Active Directory Federation Services is configured on this computer.

Prerequisites Check Completed

All prerequisite checks passed successfully. Click 'Configure' to begin installation.

10. After you complete all pre-requisite checks, select **Configure**.



This server was successfully configured

A machine restart is required to complete ADFS service configuration. For more information, see: https://go.microsoft.com/fwlink/?LinkId=798725

The SSL certificate subject alternative names do not support host name 'certauth.dev-fs.polarislibrary.com'. Configuring certificate authentication binding on port '49443' and hostname 'dev-fs.polarislibrary.com'.

Failed to register SSL bindings for Device Registration Service: An item with the same key has already been added..

Next steps required for completing your federation service deployment

Need to monitor AD FS service? Use Azure Active Directory Connect Health.

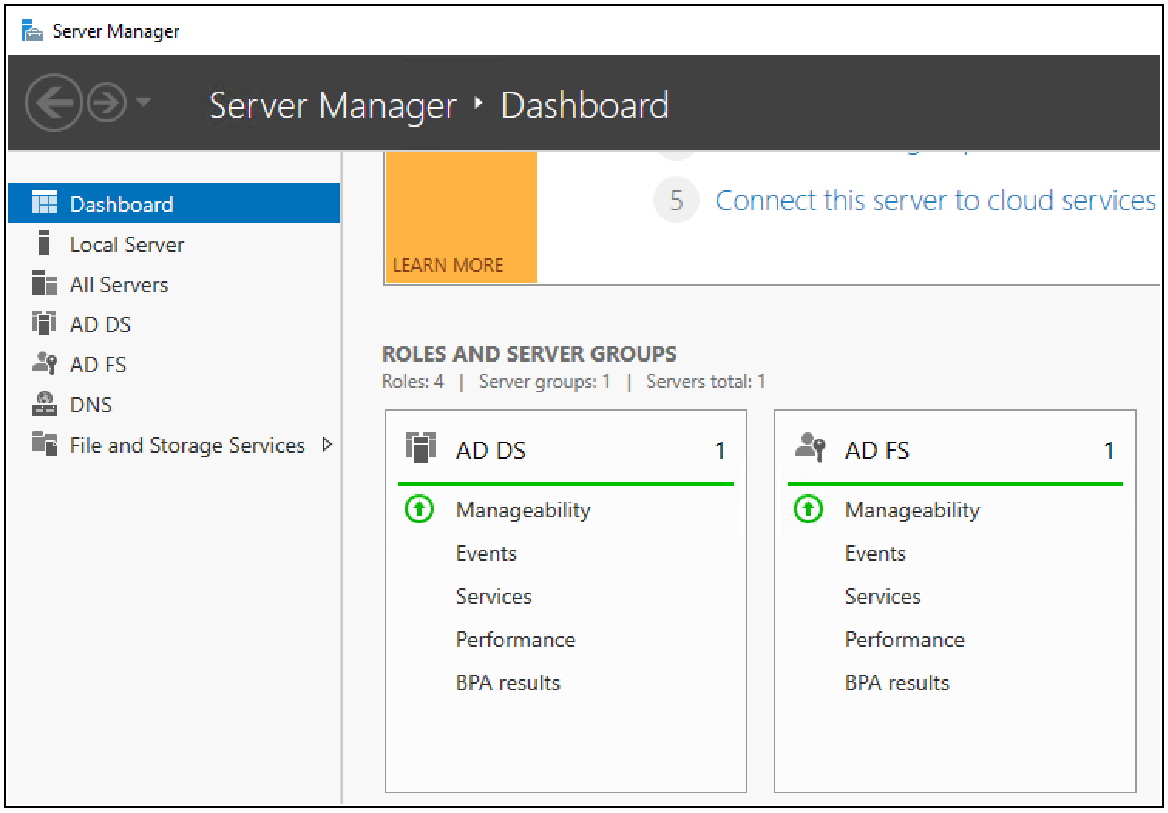
11. When the configuration wizard has completed successfully, select **Close**, and then restart the server.

Verify Active Directory Federation Services Is Running

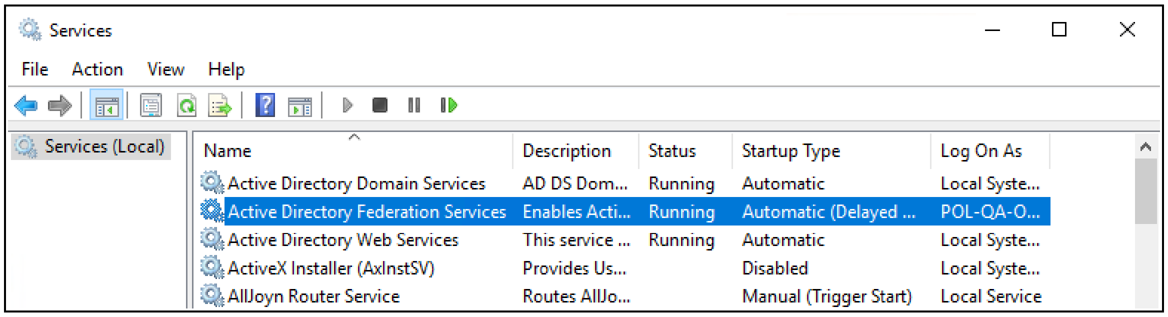
To verify that Active Directory Federation Services is running

1. Start the Server Manager desktop application.

   AD FS should be green.



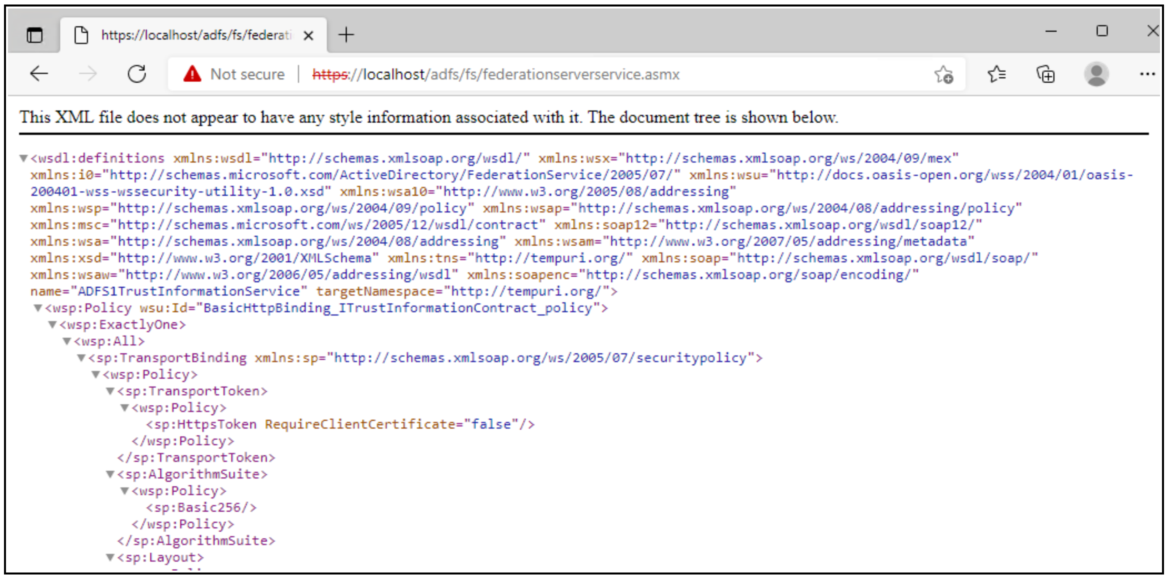
2. Start the Services application and check the status.



Name	Description	Status	Startup Type	Log On As
Active Directory Domain Services	AD DS Dom...	Running	Automatic	Local Syste...
Active Directory Federation Services	Enables Acti...	Running	Automatic (Delayed ...	POL-QA-O...
Active Directory Web Services	This service ...	Running	Automatic	Local Syste...
ActiveX Installer (AxInstSV)	Provides Us...		Disabled	Local Syste...
AllJoyn Router Service	Routes AllJo...		Manual (Trigger Start)	Local Service

3. Open the Edge (or Chrome) web browser and go to https://localhost/adfs/fs/federationserverservice.asmx
   - If you want to ignore certificate errors, select **Advanced**.

A page similar to the following image opens:


https://localhost/adfs/fs/federati
Not secure
~~https~~://localhost/adfs/fs/federationserverservice.asmx

This XML file does not appear to have any style information associated with it. The document tree is shown below.

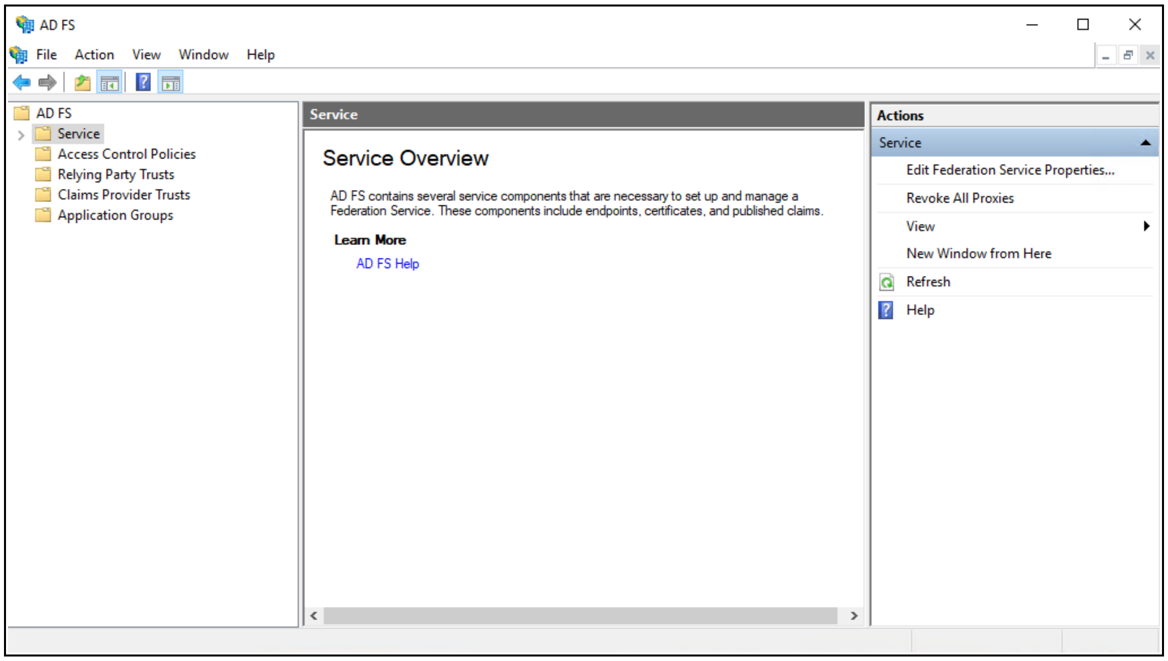
```
▼<wsdl:definitions xmlns:wsdl="http://schemas.xmlsoap.org/wsdl/" xmlns:wsx="http://schemas.xmlsoap.org/ws/2004/09/mex"
 xmlns:i0="http://schemas.microsoft.com/ActiveDirectory/FederationService/2005/07/" xmlns:wsu="http://docs.oasis-open.org/wss/2004/01/oasis-
 200401-wss-wssecurity-utility-1.0.xsd" xmlns:wsa10="http://www.w3.org/2005/08/addressing"
 xmlns:wsp="http://schemas.xmlsoap.org/ws/2004/09/policy" xmlns:wsap="http://schemas.xmlsoap.org/ws/2004/08/addressing/policy"
 xmlns:msc="http://schemas.microsoft.com/ws/2005/12/wsdl/contract" xmlns:soap12="http://schemas.xmlsoap.org/wsdl/soap12/"
 xmlns:wsa="http://schemas.xmlsoap.org/ws/2004/08/addressing" xmlns:wsam="http://www.w3.org/2007/05/addressing/metadata"
 xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:tns="http://tempuri.org/" xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
 xmlns:wsaw="http://www.w3.org/2006/05/addressing/wsdl" xmlns:soapenc="http://schemas.xmlsoap.org/soap/encoding/"
 name="ADFS1TrustInformationService" targetNamespace="http://tempuri.org/">
  ▼<wsp:Policy wsu:Id="BasicHttpBinding_ITrustInformationContract_policy">
    ▼<wsp:ExactlyOne>
      ▼<wsp:All>
        ▼<sp:TransportBinding xmlns:sp="http://schemas.xmlsoap.org/ws/2005/07/securitypolicy">
          ▼<wsp:Policy>
            ▼<sp:TransportToken>
              ▼<wsp:Policy>
                  <sp:HttpsToken RequireClientCertificate="false"/>
                </wsp:Policy>
              </sp:TransportToken>
            ▼<sp:AlgorithmSuite>
              ▼<wsp:Policy>
                  <sp:Basic256/>
                </wsp:Policy>
              </sp:AlgorithmSuite>
            ▼<sp:Layout>
```

This indicates that Active Directory Federation Services is running.

Verify that OAuth 2.0 is Enabled

To verify that OAuth 2.0 is enabled

1. Open the AD FS Management desktop application.



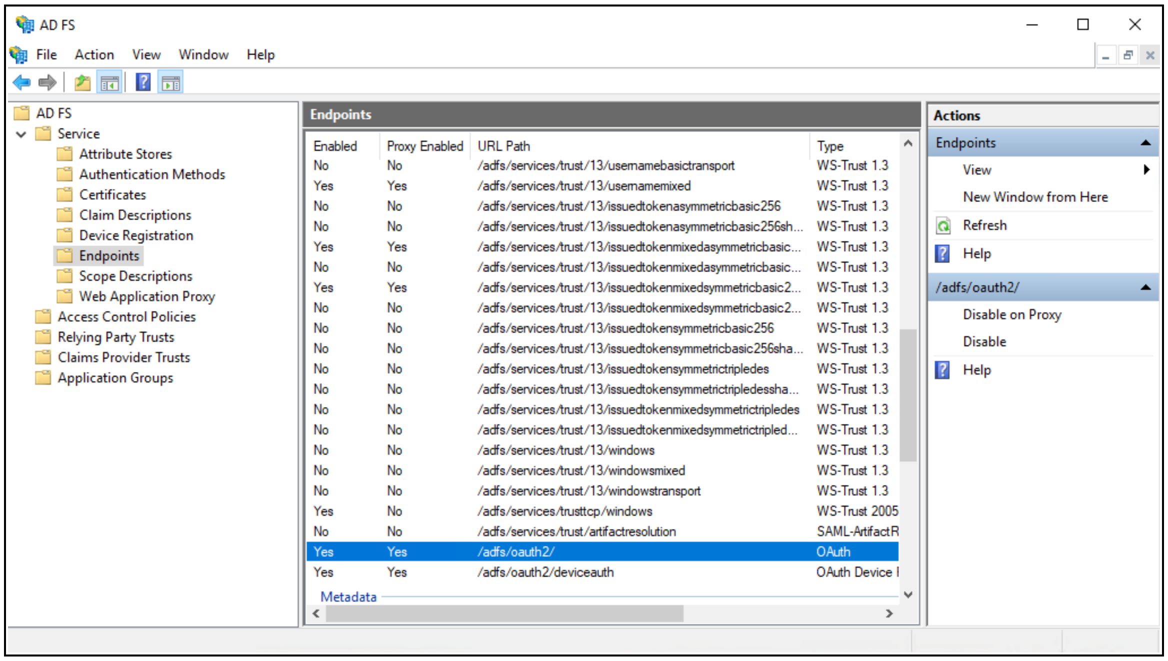
Service Overview

AD FS contains several service components that are necessary to set up and manage a Federation Service. These components include endpoints, certificates, and published claims.

Learn More

AD FS Help

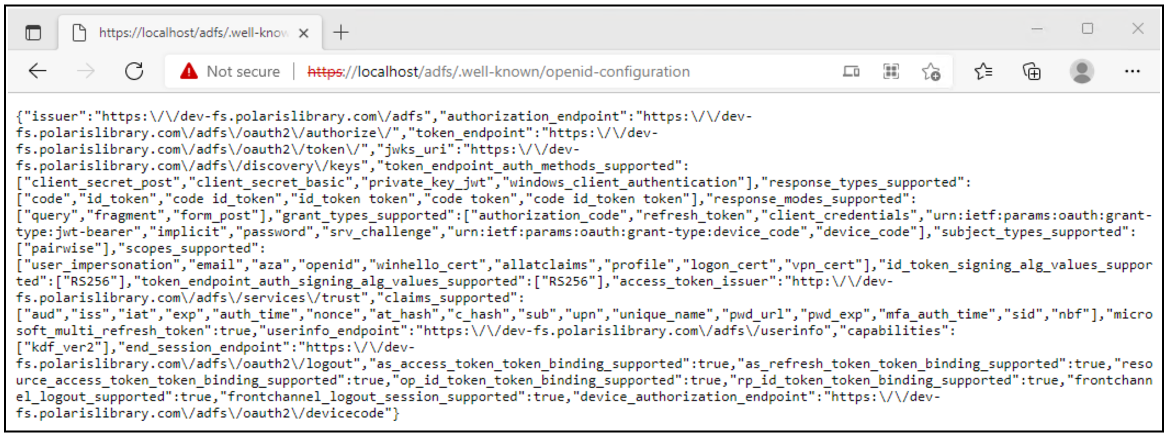
2. Open the **Service** folder, and then select the **Endpoint** folder.



Enabled	Proxy Enabled	URL Path	Type
No	No	/adfs/services/trust/13/usernamebasictransport	WS-Trust 1.3
Yes	Yes	/adfs/services/trust/13/usernamemixed	WS-Trust 1.3
No	No	/adfs/services/trust/13/issuedtokenasymmetricbasic256	WS-Trust 1.3
No	No	/adfs/services/trust/13/issuedtokenasymmetricbasic256sh...	WS-Trust 1.3
Yes	Yes	/adfs/services/trust/13/issuedtokenmixedasymmetricbasic...	WS-Trust 1.3
No	No	/adfs/services/trust/13/issuedtokenmixedasymmetricbasic...	WS-Trust 1.3
Yes	Yes	/adfs/services/trust/13/issuedtokenmixedsymmetricbasic2...	WS-Trust 1.3
No	No	/adfs/services/trust/13/issuedtokenmixedsymmetricbasic2...	WS-Trust 1.3
No	No	/adfs/services/trust/13/issuedtokensymmetricbasic256	WS-Trust 1.3
No	No	/adfs/services/trust/13/issuedtokensymmetricbasic256sha...	WS-Trust 1.3
No	No	/adfs/services/trust/13/issuedtokensymmetrictripledes	WS-Trust 1.3
No	No	/adfs/services/trust/13/issuedtokensymmetrictripledessha...	WS-Trust 1.3
No	No	/adfs/services/trust/13/issuedtokenmixedsymmetrictripledes	WS-Trust 1.3
No	No	/adfs/services/trust/13/issuedtokenmixedsymmetrictripled...	WS-Trust 1.3
No	No	/adfs/services/trust/13/windows	WS-Trust 1.3
No	No	/adfs/services/trust/13/windowsmixed	WS-Trust 1.3
No	No	/adfs/services/trust/13/windowstransport	WS-Trust 1.3
Yes	No	/adfs/services/trusttcp/windows	WS-Trust 2005
No	No	/adfs/services/trust/artifactresolution	SAML-ArtifactR
Yes	Yes	/adfs/oauth2/	OAuth
Yes	Yes	/adfs/oauth2/deviceauth	OAuth Device I

3. Search for the oauth2 path.
4. In either the Edge or Chrome web browser, go to https://localhost/adfs/.well-known/openid-configuration
    - If you want to ignore certificate errors, select **Advanced**.

A page similar to the following image opens:


https://localhost/adfs/.well-know
Not secure
~~https~~://localhost/adfs/.well-known/openid-configuration

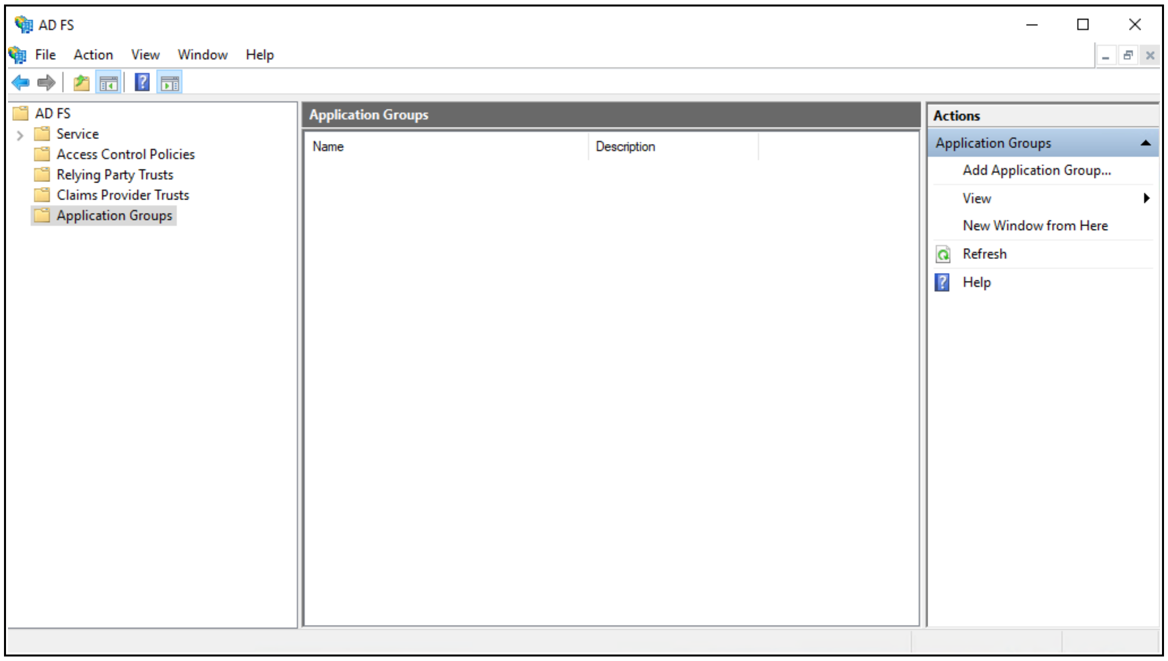
```
{"issuer":"https:\/\/dev-fs.polarislibrary.com\/adfs","authorization_endpoint":"https:\/\/dev-
fs.polarislibrary.com\/adfs\/oauth2\/authorize\/","token_endpoint":"https:\/\/dev-
fs.polarislibrary.com\/adfs\/oauth2\/token\/","jwks_uri":"https:\/\/dev-
fs.polarislibrary.com\/adfs\/discovery\/keys","token_endpoint_auth_methods_supported":
["client_secret_post","client_secret_basic","private_key_jwt","windows_client_authentication"],"response_types_supported":
["code","id_token","code id_token","id_token token","code token","code id_token token"],"response_modes_supported":
["query","fragment","form_post"],"grant_types_supported":["authorization_code","refresh_token","client_credentials","urn:ietf:params:oauth:grant-
type:jwt-bearer","implicit","password","srv_challenge","urn:ietf:params:oauth:grant-type:device_code","device_code"],"subject_types_supported":
["pairwise"],"scopes_supported":
["user_impersonation","email","aza","openid","winhello_cert","allatclaims","profile","logon_cert","vpn_cert"],"id_token_signing_alg_values_suppor
ted":["RS256"],"token_endpoint_auth_signing_alg_values_supported":["RS256"],"access_token_issuer":"http:\/\/dev-
fs.polarislibrary.com\/adfs\/services\/trust","claims_supported":
["aud","iss","iat","exp","auth_time","nonce","at_hash","c_hash","sub","upn","unique_name","pwd_url","pwd_exp","mfa_auth_time","sid","nbf"],"micro
soft_multi_refresh_token":true,"userinfo_endpoint":"https:\/\/dev-fs.polarislibrary.com\/adfs\/userinfo","capabilities":
["kdf_ver2"],"end_session_endpoint":"https:\/\/dev-
fs.polarislibrary.com\/adfs\/oauth2\/logout","as_access_token_token_binding_supported":true,"as_refresh_token_token_binding_supported":true,"reso
urce_access_token_token_binding_supported":true,"op_id_token_token_binding_supported":true,"rp_id_token_token_binding_supported":true,"frontchann
el_logout_supported":true,"frontchannel_logout_session_supported":true,"device_authorization_endpoint":"https:\/\/dev-
fs.polarislibrary.com\/adfs\/oauth2\/devicecode"}
```

This indicates that OAuth 2.0 is available.

Create an Application Group

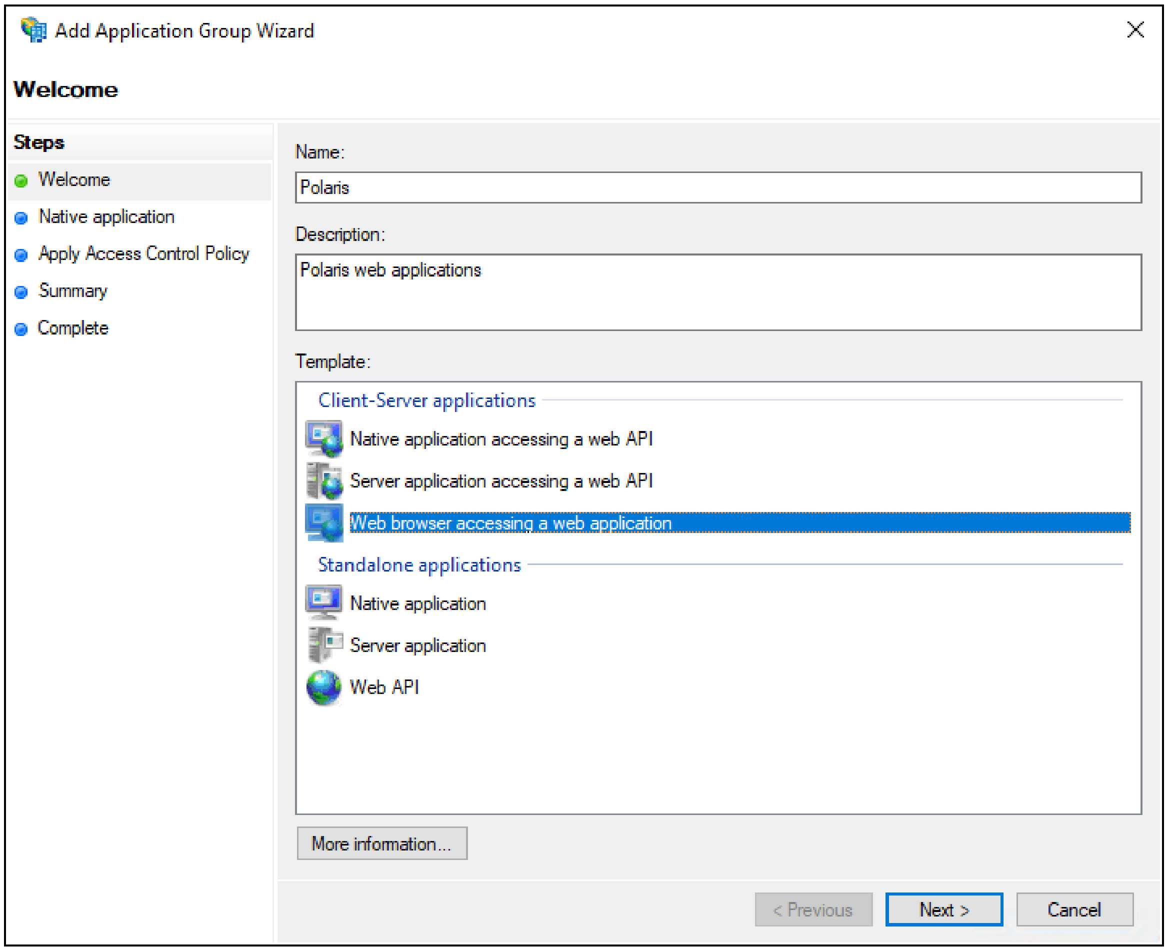
To create an application group for use with Polaris Admin and LeapWebApp

1. Open the AD FS Management desktop application.



2. Select the **Application Groups** folder.
3. Under **Actions**, select **Add Application Group**.

   The Add Application Group wizard opens.



Name: Polaris

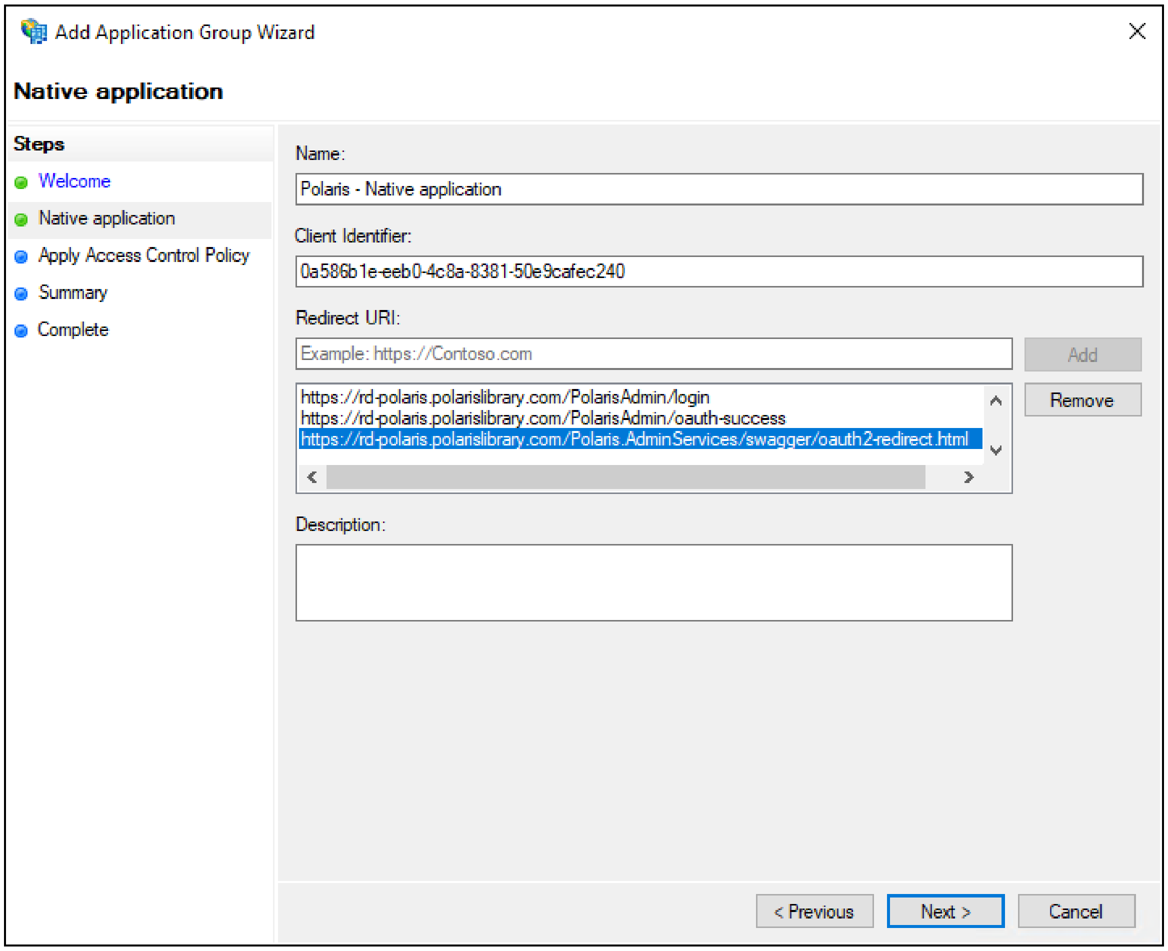
Description: Polaris web applications

Template:

Client-Server applications
- Native application accessing a web API
- Server application accessing a web API
- Web browser accessing a web application

Standalone applications
- Native application
- Server application
- Web API

4. On the **Welcome** tab, do the following:
   a. In the **Name** box, enter **Polaris**.
   b. In the **Description** box, enter **Polaris web applications**.
   c. From the Template section, select **Web browser accessing a web application**.
5. Select **Next**.



Name: Polaris - Native application

Client Identifier: 0a586b1e-eeb0-4c8a-8381-50e9cafec240

Redirect URI: Example: https://Contoso.com

https://rd-polaris.polarislibrary.com/PolarisAdmin/login
https://rd-polaris.polarislibrary.com/PolarisAdmin/oauth-success
https://rd-polaris.polarislibrary.com/Polaris.AdminServices/swagger/oauth2-redirect.html

Description:

6. On the **Native application** tab, in the **Redirect URI** box, enter the following URIs:
   - https://*server address*/PolarisAdmin/
   - https://*server address*/PolarisAdmin/login
   - https://*server address*/PolarisAdmin/oauth-success
   - https://*server address*/Polaris.AdminServices/swagger/oauth2-redirect.html
   - https://*server address*/LeapWebApp/signin-oidc
   - https://*server address*/LeapWebApp/signin-override-oidc
   - https://*server address*/LeapWebApp/signout-callback-oidc
   - https://*server address*/Polaris.ApplicationServices/swagger/oauth2-redirect.html

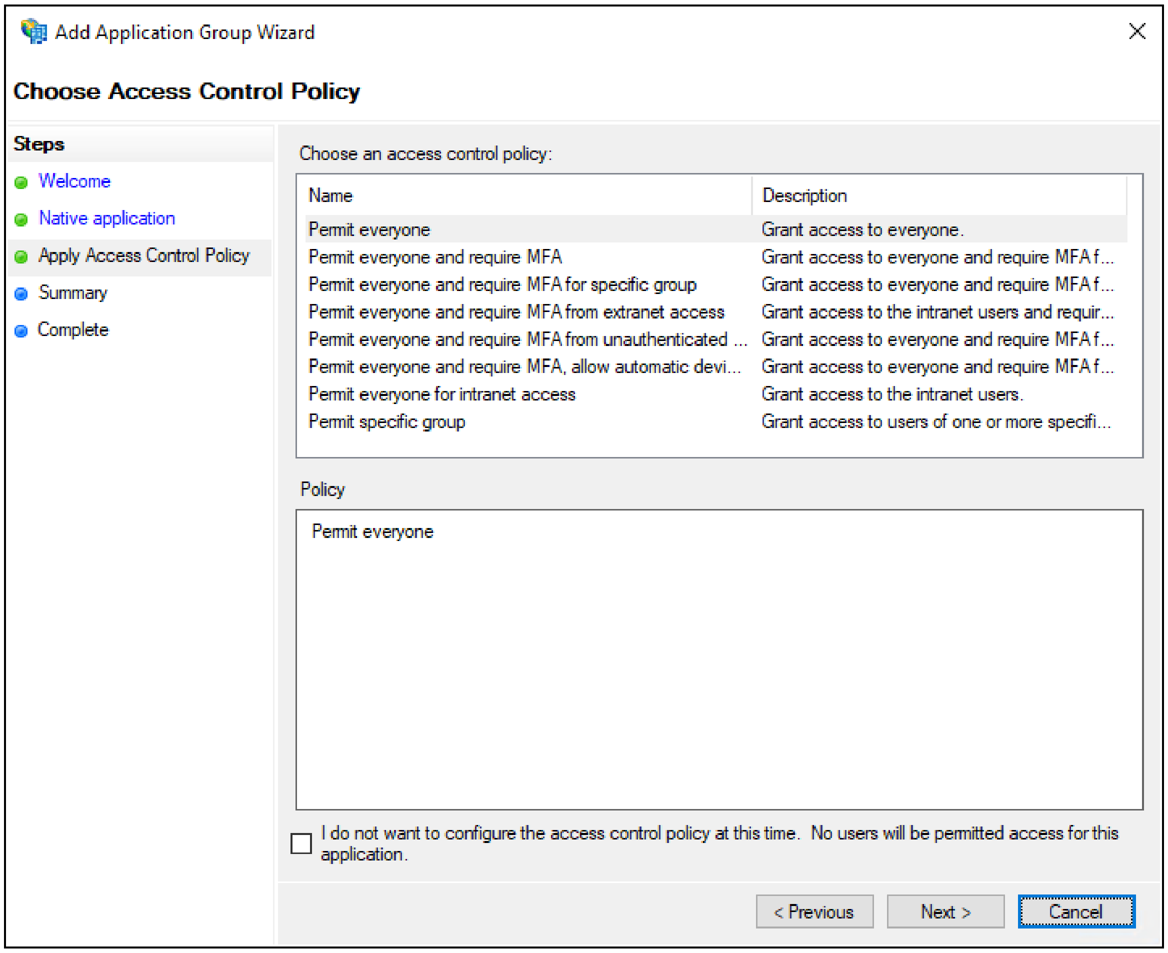
   Note:
   Replace *server address* with the FQDN that matches your Polaris

System Administration (web-based) or Leap server name and certificate.

7. Copy the value in the **Client Identifier** box to Notepad.

   You'll need this when you set up PolarisAdmin's appsettings.user.json.

8. Select **Next**.


Add Application Group Wizard

Choose Access Control Policy

Choose an access control policy:

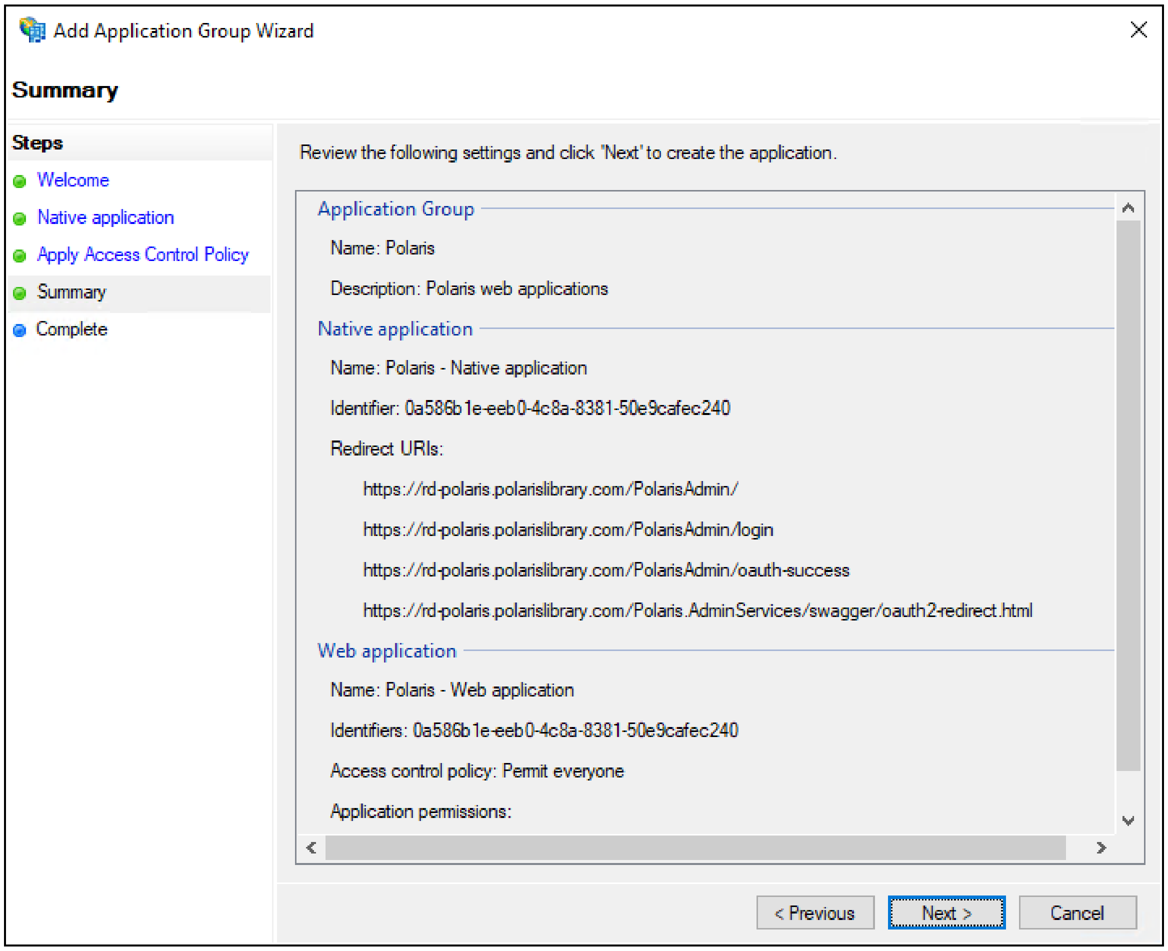
Name	Description
Permit everyone	Grant access to everyone.
Permit everyone and require MFA	Grant access to everyone and require MFA f...
Permit everyone and require MFA for specific group	Grant access to everyone and require MFA f...
Permit everyone and require MFA from extranet access	Grant access to the intranet users and requir...
Permit everyone and require MFA from unauthenticated ...	Grant access to everyone and require MFA f...
Permit everyone and require MFA, allow automatic devi...	Grant access to everyone and require MFA f...
Permit everyone for intranet access	Grant access to the intranet users.
Permit specific group	Grant access to users of one or more specifi...

Policy

Permit everyone

☐ I do not want to configure the access control policy at this time. No users will be permitted access for this application.

9. On the **Apply Access Control Policy** tab, select an access control policy, and then select **Next**.



Review the following settings and click "Next" to create the application.

Application Group

Name: Polaris

Description: Polaris web applications

Native application

Name: Polaris - Native application

Identifier: 0a586b1e-eeb0-4c8a-8381-50e9cafec240

Redirect URIs:

https://rd-polaris.polarislibrary.com/PolarisAdmin/

https://rd-polaris.polarislibrary.com/PolarisAdmin/login

https://rd-polaris.polarislibrary.com/PolarisAdmin/oauth-success

https://rd-polaris.polarislibrary.com/Polaris.AdminServices/swagger/oauth2-redirect.html

Web application

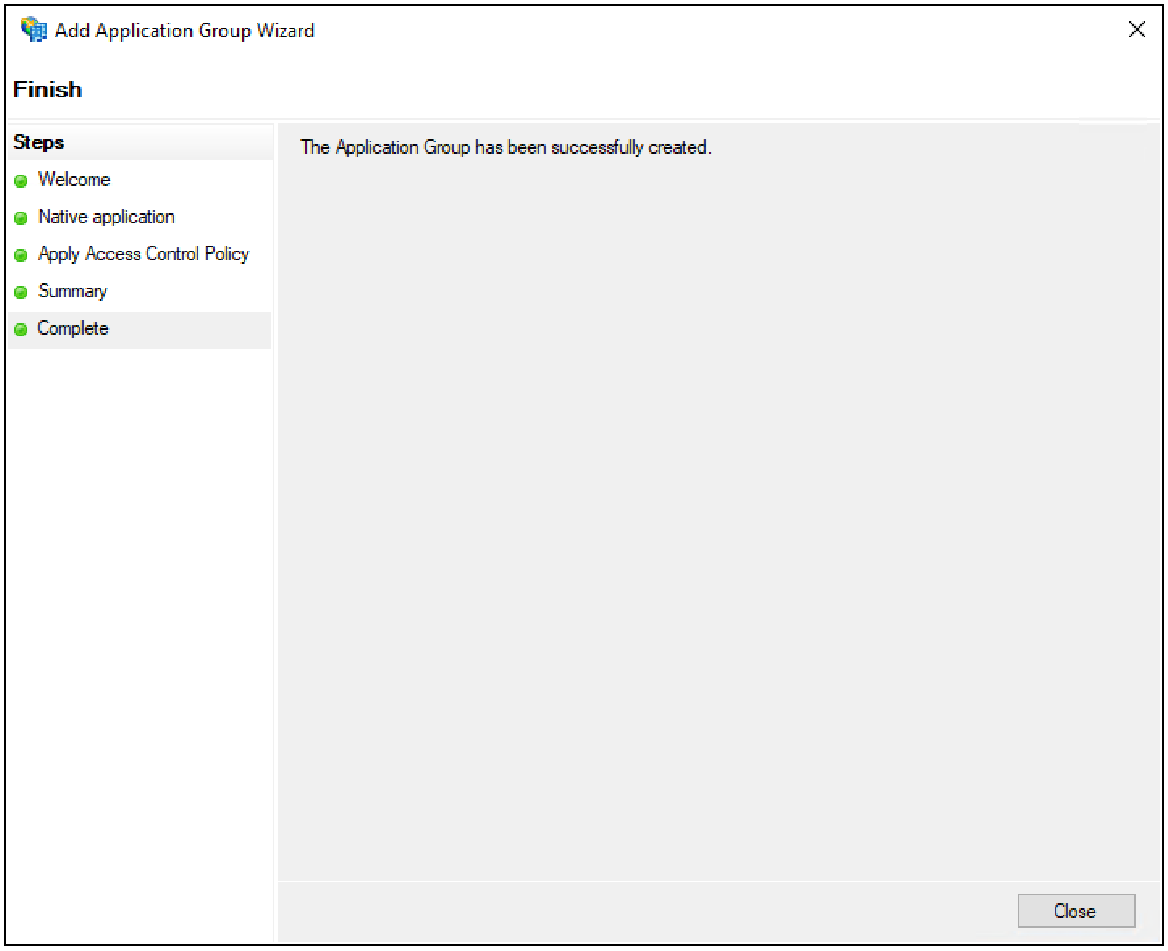
Name: Polaris - Web application

Identifiers: 0a586b1e-eeb0-4c8a-8381-50e9cafec240

Access control policy: Permit everyone

Application permissions:

10. On the **Summary** tab, review the settings and then select **Next**.


Add Application Group Wizard

Finish

Steps
- Welcome
- Native application
- Apply Access Control Policy
- Summary
- Complete

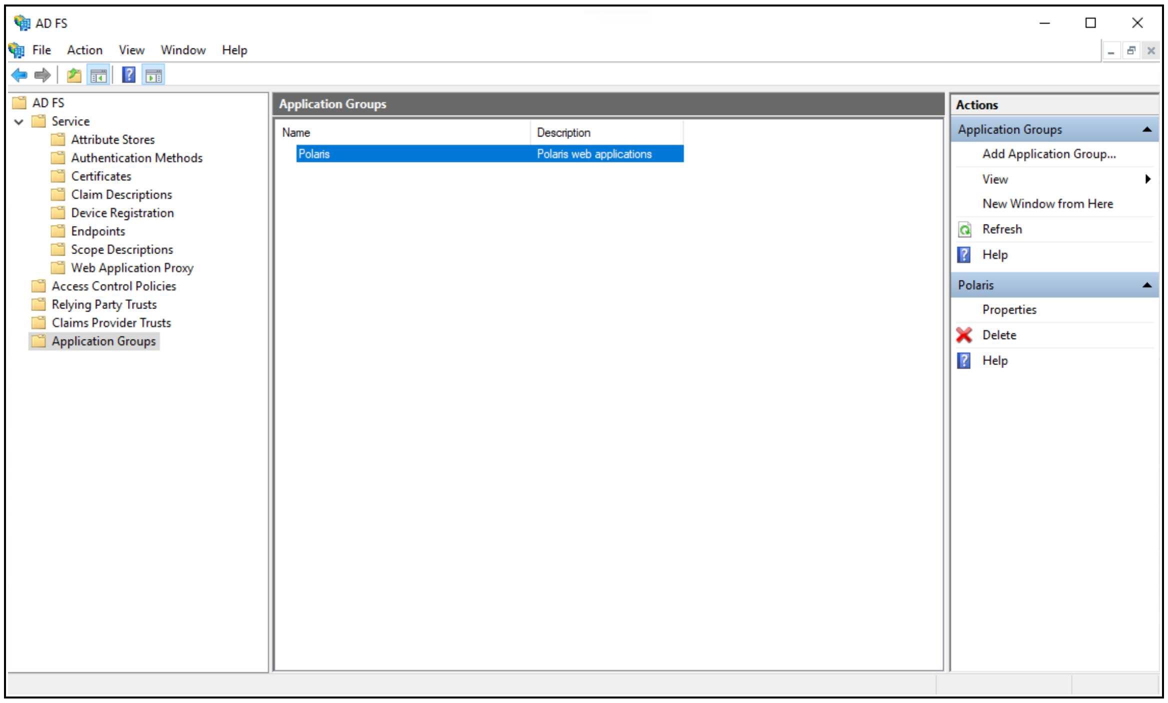
The Application Group has been successfully created.

11. On the **Complete** tab, select **Close**.

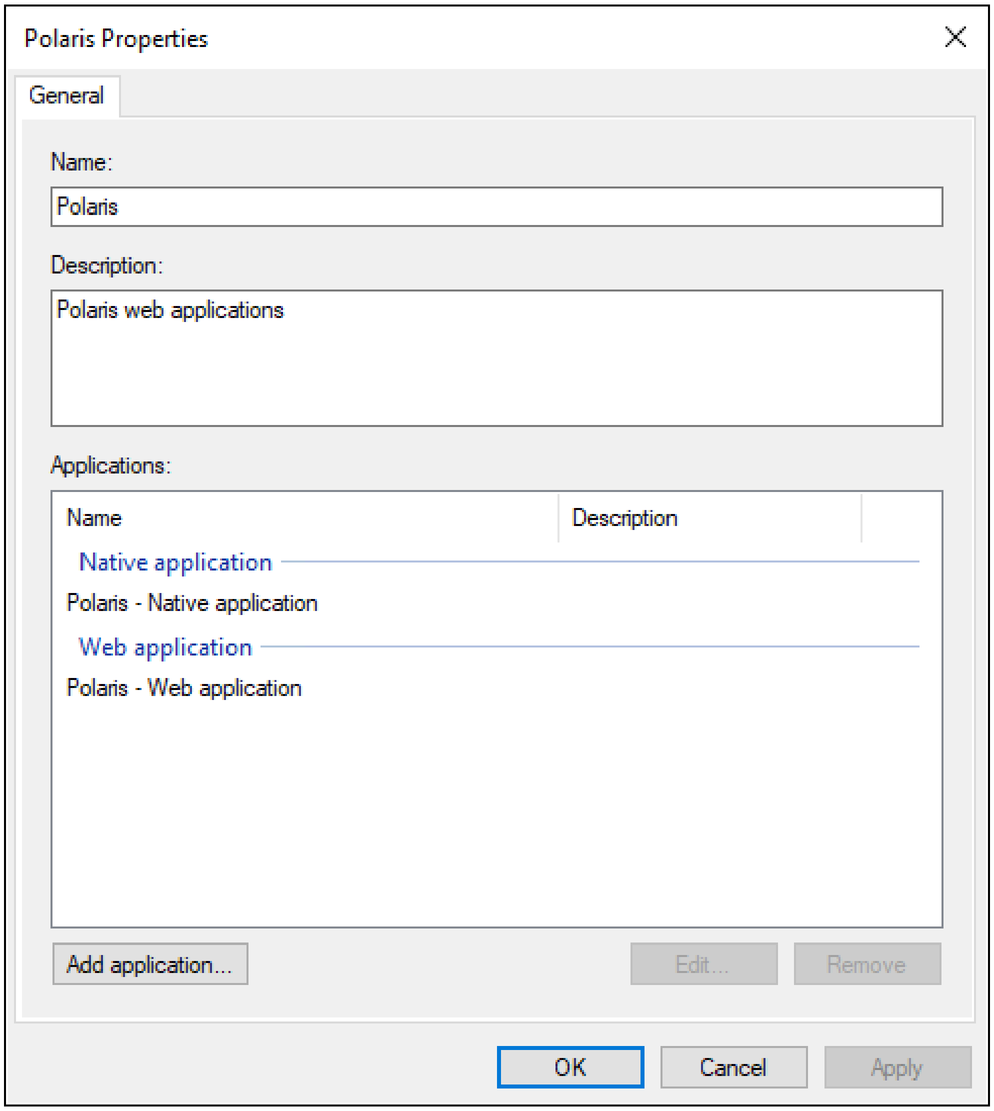
Configure the AD FS Web Application: Claims and Permitted Scopes

To configure the AD FS web application

1. Open the AD FS Management desktop application.



2. Select the **Application Groups** folder.
3. Select the **Polaris** application group, and then select **Properties**.



Name:

Polaris

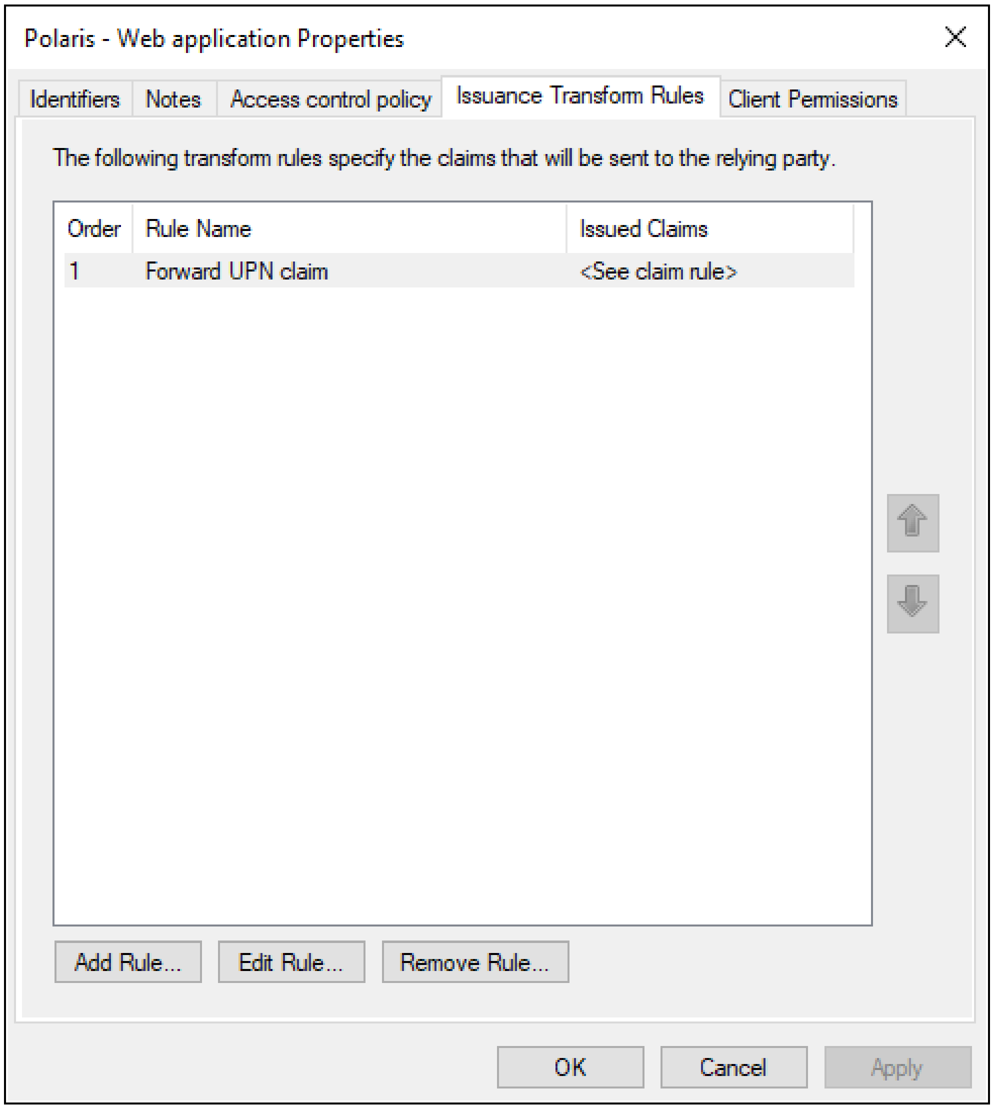
Description:

Polaris web applications

Applications:

Name	Description
Native application	
Polaris - Native application	
Web application	
Polaris - Web application	

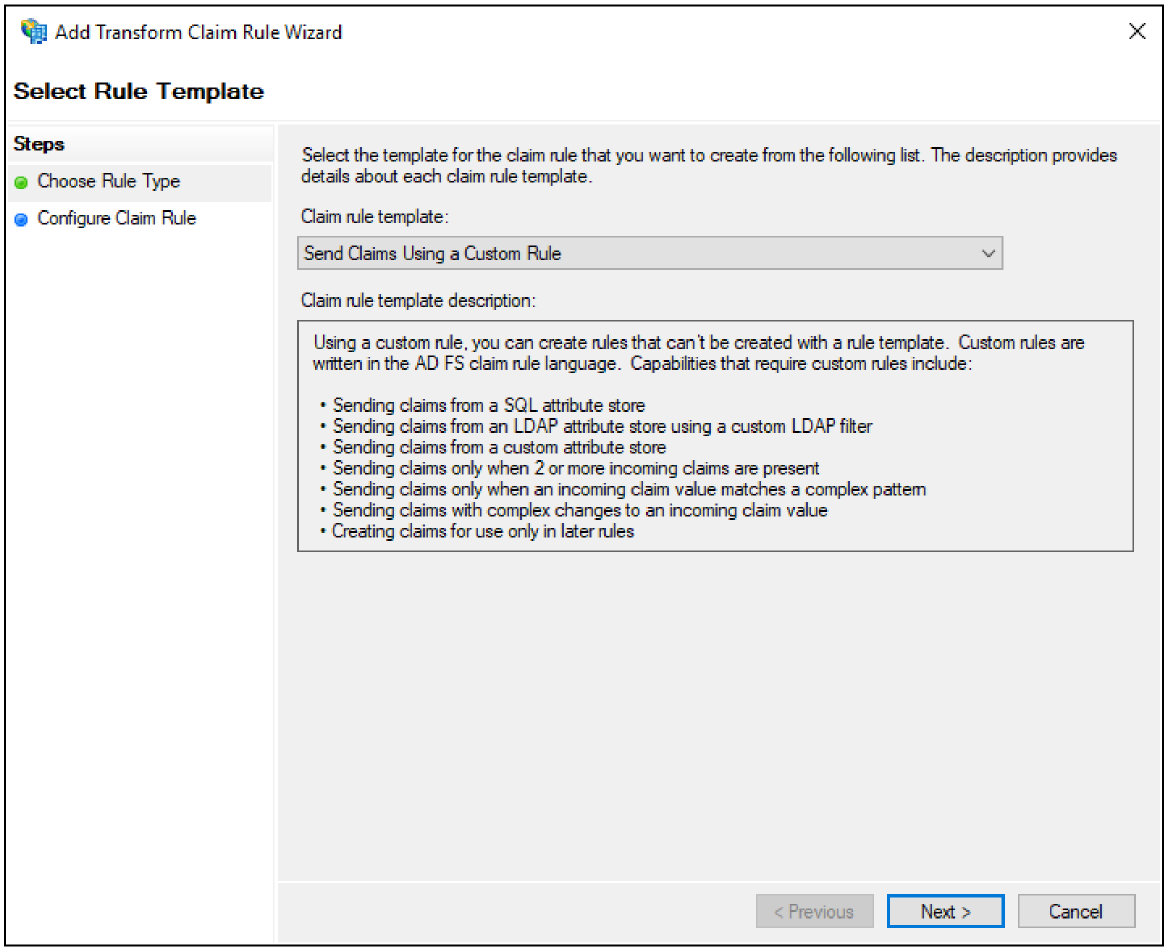
4. Select **Polaris** - **Web application**, and then select **Edit**.


Polaris - Web application Properties

The following transform rules specify the claims that will be sent to the relying party.

Order	Rule Name	Issued Claims
1	Forward UPN claim	<See claim rule>

5. Select the **Issuance Transform Rules** tab, and then select **Add Rule**.



Select the template for the claim rule that you want to create from the following list. The description provides details about each claim rule template.

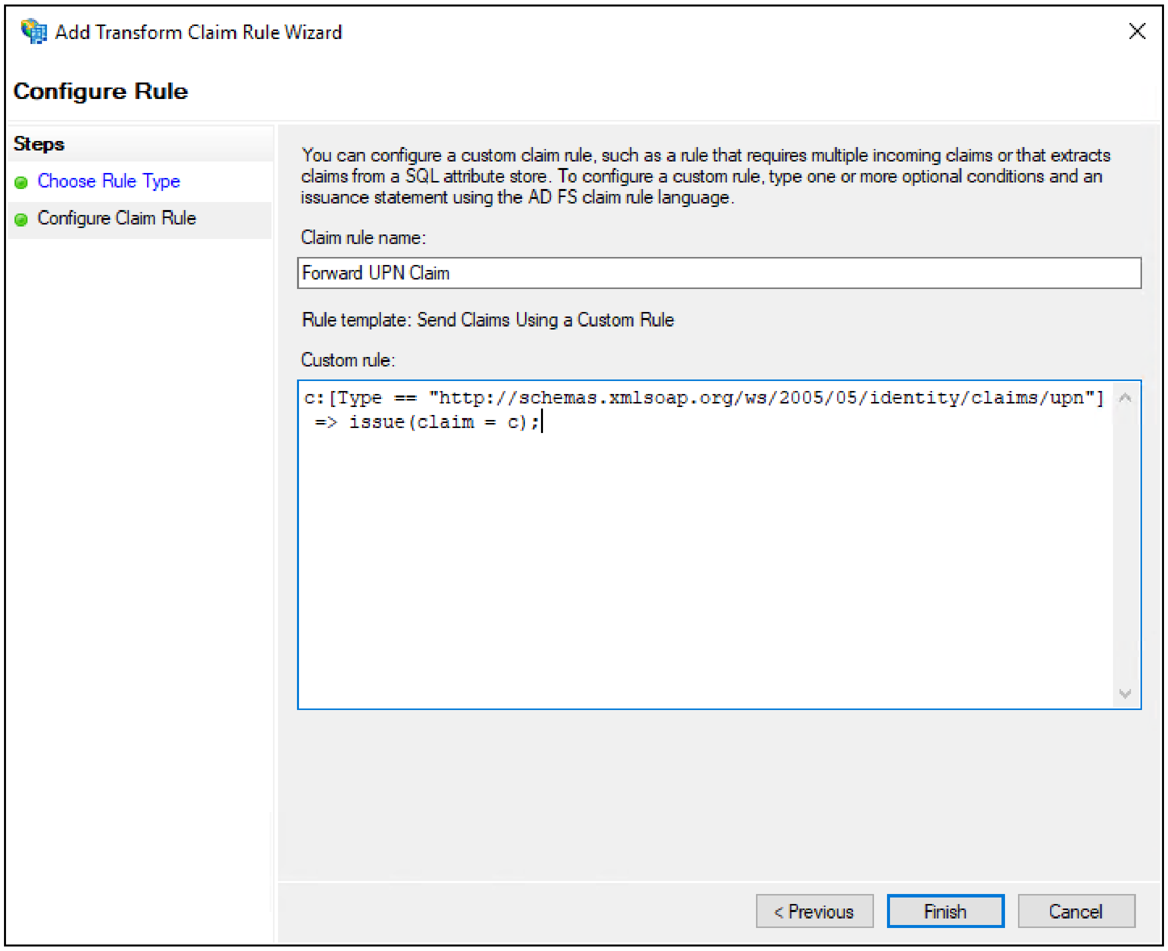
Claim rule template: Send Claims Using a Custom Rule

Claim rule template description:

Using a custom rule, you can create rules that can't be created with a rule template. Custom rules are written in the AD FS claim rule language. Capabilities that require custom rules include:

- Sending claims from a SQL attribute store
- Sending claims from an LDAP attribute store using a custom LDAP filter
- Sending claims from a custom attribute store
- Sending claims only when 2 or more incoming claims are present
- Sending claims only when an incoming claim value matches a complex pattern
- Sending claims with complex changes to an incoming claim value
- Creating claims for use only in later rules

6. On the Add Transform Claim Rule Wizard, select **Send Claims Using a Custom Rule** from the **Claim rule template** list, and then select **Next**.



You can configure a custom claim rule, such as a rule that requires multiple incoming claims or that extracts claims from a SQL attribute store. To configure a custom rule, type one or more optional conditions and an issuance statement using the AD FS claim rule language.

Claim rule name: Forward UPN Claim

Rule template: Send Claims Using a Custom Rule

Custom rule:

```
c:[Type == "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn"]
 => issue(claim = c);
```

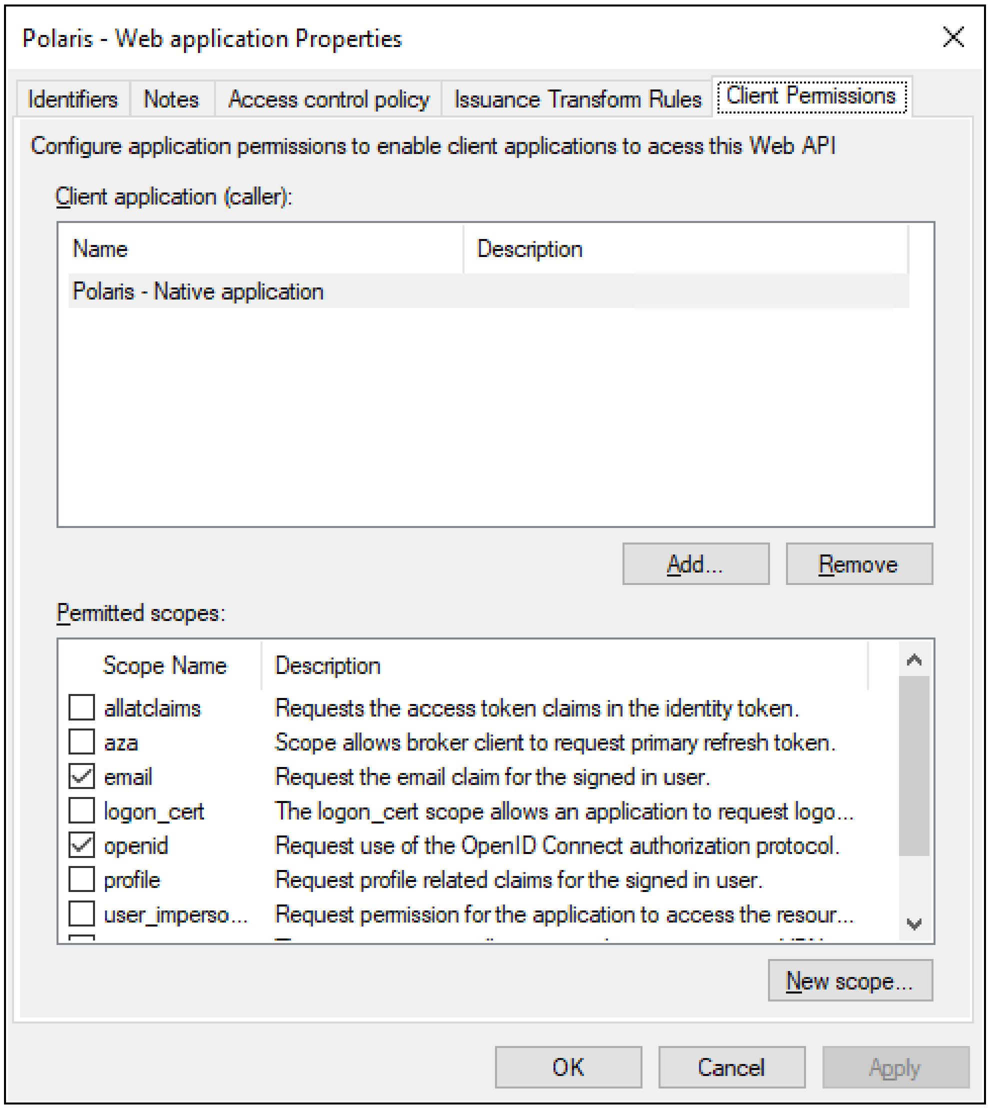
7. In the **Claim rule name** box, enter **Forward UPN Claim**.
8. In the **Custom rule** box, enter the following rule:

   ```
   c:[Type ==
   "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn"]
   => issue(claim = c);
   ```

9. Select **Finish**.
10. On the **Issuance Transform Rules** tab, select **Add Rule**.
11. On the Add Transform Claim Rule Wizard, select **Send Claims Using a Custom Rule** from the **Claim rule template** list, and then select **Next**.
12. In the **Claim rule name** box, enter **Add TenantId**.
13. In the **Custom rule** box, enter the following rule:

```
=> issue(Type =
"http://schemas.microsoft.com/identity/claims/tenantid",
Value = "polaris");
```

14. Select **Finish**.



Polaris - Web application Properties

Identifiers | Notes | Access control policy | Issuance Transform Rules | Client Permissions

Configure application permissions to enable client applications to acess this Web API

Client application (caller):

Name	Description
Polaris - Native application	

Permitted scopes:

	Scope Name	Description
☐	allatclaims	Requests the access token claims in the identity token.
☐	aza	Scope allows broker client to request primary refresh token.
✓	email	Request the email claim for the signed in user.
☐	logon_cert	The logon_cert scope allows an application to request logo...
✓	openid	Request use of the OpenID Connect authorization protocol.
☐	profile	Request profile related claims for the signed in user.
☐	user_imperso...	Request permission for the application to access the resour...

15. On the **Client Permissions** tab, verify that **email** and **openid** are selected.
16. Select **OK** to close the Web application Properties dialog.
17. Select **OK** to close the Polaris properties dialog.
18. Using the services applet, restart the Active Directory Federation Services service.

Enable CORS on AD FS To Accept Requests from Polaris APIs

To enable CORS on AD FS to accept requests from Polaris APIs

1. Refer to the information on the following page:
   - https://docs.microsoft.com/en-us/windows-server/identity/ad-fs/operations/customize-http-security-headers-ad-fs#cross-origin-resource-sharing-cors-headers
2. Use the following commands:
   - `Set-AdfsResponseHeaders -EnableCORS $true`
   - `Set-AdfsResponseHeaders -CORSTrustedOrigins` *`https://rd-polaris.polarislibrary.com,https://example2.com`*

   Note:
   Replace *`https://rd-polaris.polarislibrary.com`* and *`https://example2.com`* with your own URL or list of URLs.

Set Up Web Services and Applications

To set up each of the following web services and applications, you must configure a .json file for each of the following:

- Polaris.AdminServices (the API service)
- PolarisAdmin (the web-based Polaris System Administration application)
- Polaris.ApplicationServices (Leap's API service)
- LeapWebApp (Leap)

The four .json files are all named appsettings.user.json, but they reside in different directories:

- C:\Program Files\Polaris\7.4\Polaris.AdminServices
- C:\Program Files\Polaris\7.4\PolarisAdmin\assets
- C:\Program Files\Polaris\7.4\Polaris.ApplicationServices
- C:\Program Files\Polaris\7.4\LeapWebApp

This section contains the following topics:

- Set Up Polaris.AdminServices
- Set Up PolarisAdmin
- Set Up Polaris.ApplicationServices
- Set Up LeapWebApp

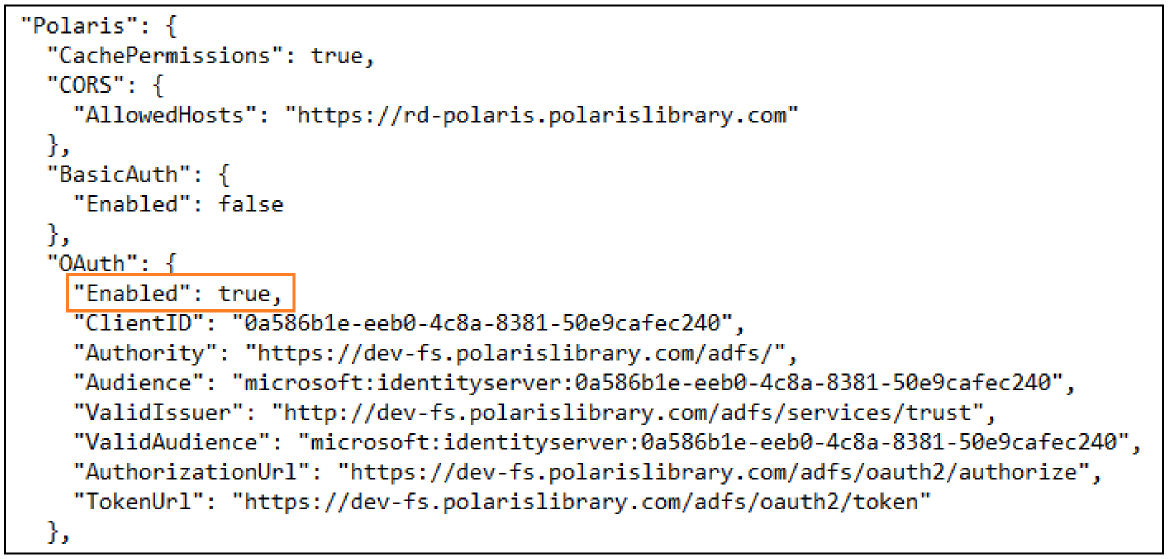
Set Up Polaris.AdminServices

To set up Polaris.AdminServices

> **Note:**
> When you edit the appsettings.user.json file, you must run the editing application (for example, Notepad) as administrator.

Verify that OAuth is Enabled

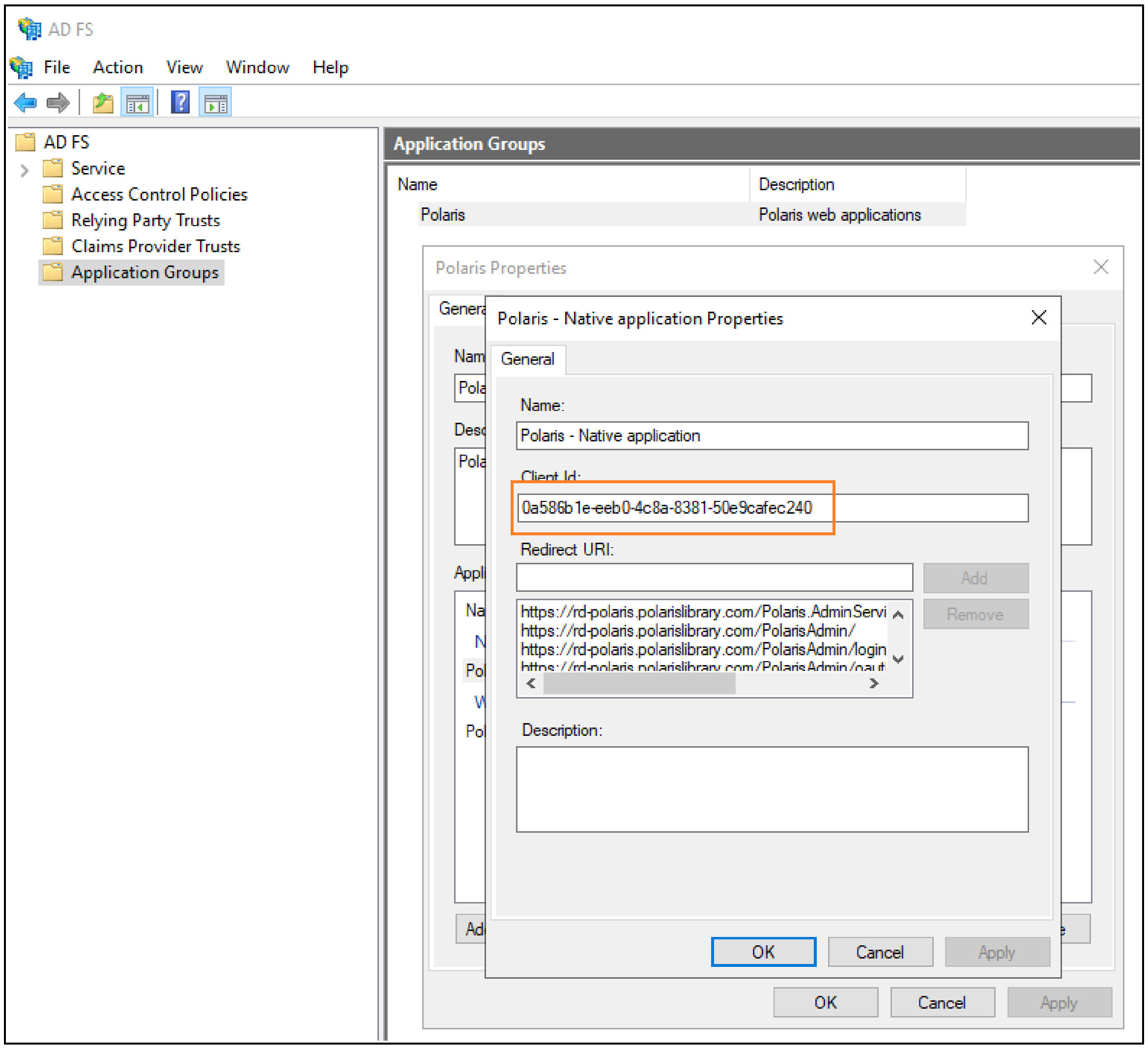
- Open C:\Program Files\Polaris\7.4\Polaris.AdminServices\appsettings.user.json and verify `Polaris.OAuth.Enabled` is set to `true`.



```
"Polaris": {
  "CachePermissions": true,
  "CORS": {
    "AllowedHosts": "https://rd-polaris.polarislibrary.com"
  },
  "BasicAuth": {
    "Enabled": false
  },
  "OAuth": {
    "Enabled": true,
    "ClientID": "0a586b1e-eeb0-4c8a-8381-50e9cafec240",
    "Authority": "https://dev-fs.polarislibrary.com/adfs/",
    "Audience": "microsoft:identityserver:0a586b1e-eeb0-4c8a-8381-50e9cafec240",
    "ValidIssuer": "http://dev-fs.polarislibrary.com/adfs/services/trust",
    "ValidAudience": "microsoft:identityserver:0a586b1e-eeb0-4c8a-8381-50e9cafec240",
    "AuthorizationUrl": "https://dev-fs.polarislibrary.com/adfs/oauth2/authorize",
    "TokenUrl": "https://dev-fs.polarislibrary.com/adfs/oauth2/token"
  },
```

Update the Client ID

1. On the AD FS server, open AD FS Management desktop application.



Name: Polaris - Native application

Client Id: 0a586b1e-eeb0-4c8a-8381-50e9cafec240

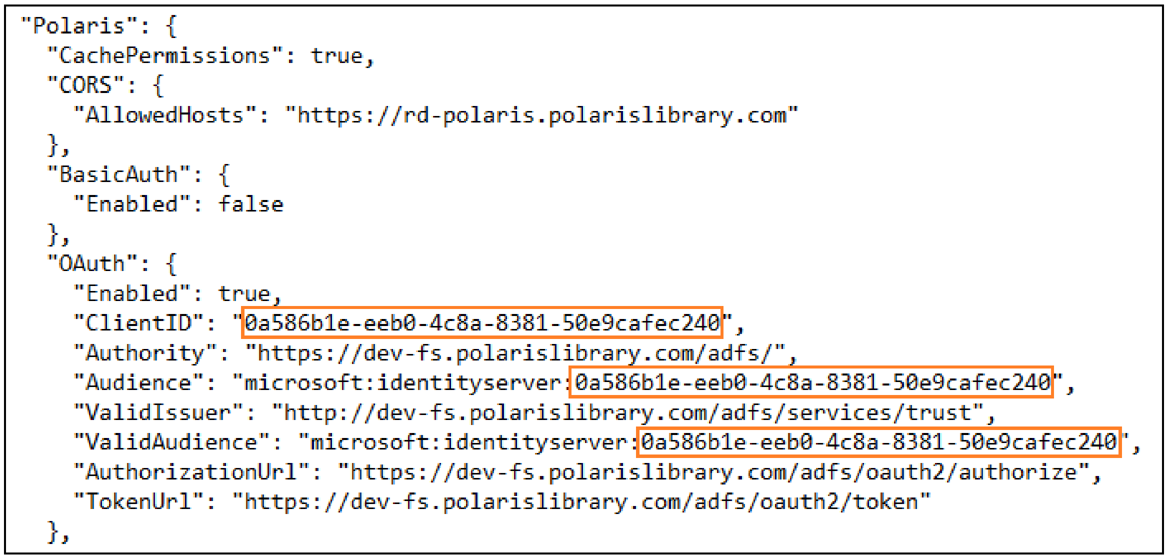
Redirect URI:

```
https://rd-polaris.polarislibrary.com/Polaris.AdminServi
https://rd-polaris.polarislibrary.com/PolarisAdmin/
https://rd-polaris.polarislibrary.com/PolarisAdmin/login
https://rd-polaris.polarislibrary.com/PolarisAdmin/oaut
```

Description:

2. Copy the client ID from the Polaris - Native application properties dialog.
3. Paste the copied client ID into the appsettings.user.json file.
4. If you started from the template, replace `[client-id-that-might-look-like-a-guid]` with the copied client ID.

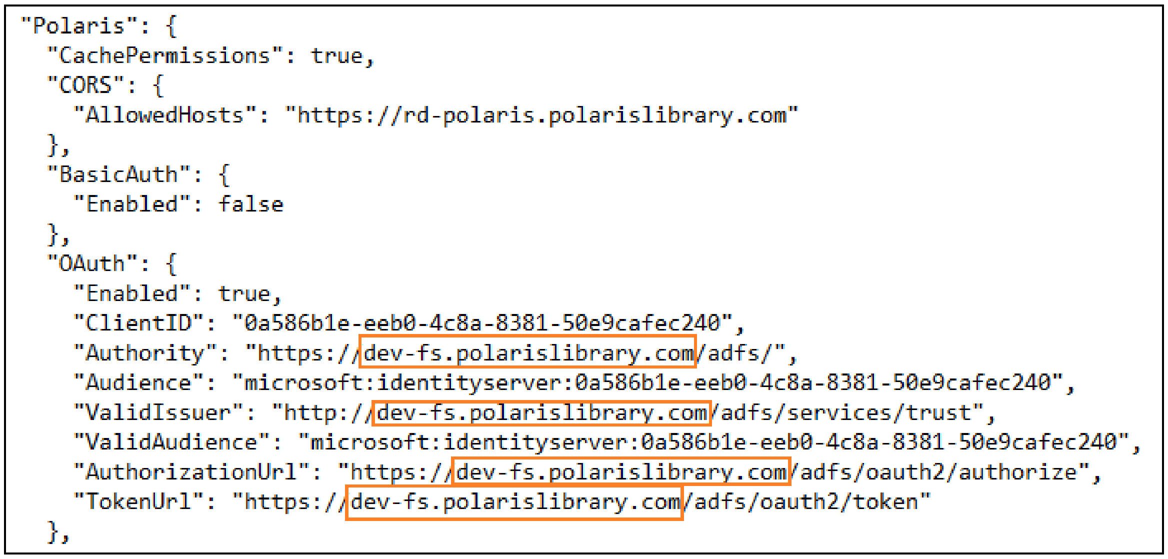
   It should look like the following image when complete (your client ID will be different):



```
"Polaris": {
  "CachePermissions": true,
  "CORS": {
    "AllowedHosts": "https://rd-polaris.polarislibrary.com"
  },
  "BasicAuth": {
    "Enabled": false
  },
  "OAuth": {
    "Enabled": true,
    "ClientID": "0a586b1e-eeb0-4c8a-8381-50e9cafec240",
    "Authority": "https://dev-fs.polarislibrary.com/adfs/",
    "Audience": "microsoft:identityserver:0a586b1e-eeb0-4c8a-8381-50e9cafec240",
    "ValidIssuer": "http://dev-fs.polarislibrary.com/adfs/services/trust",
    "ValidAudience": "microsoft:identityserver:0a586b1e-eeb0-4c8a-8381-50e9cafec240",
    "AuthorizationUrl": "https://dev-fs.polarislibrary.com/adfs/oauth2/authorize",
    "TokenUrl": "https://dev-fs.polarislibrary.com/adfs/oauth2/token"
  },
```

Update the AD FS Server Location

1. If you started from the template, replace *`[my-adfs-server-domain-name]`* with the AD FS server address.
2. It should look like the following when complete (your AD FS server address will be different):



```
"Polaris": {
  "CachePermissions": true,
  "CORS": {
    "AllowedHosts": "https://rd-polaris.polarislibrary.com"
  },
  "BasicAuth": {
    "Enabled": false
  },
  "OAuth": {
    "Enabled": true,
    "ClientID": "0a586b1e-eeb0-4c8a-8381-50e9cafec240",
    "Authority": "https://dev-fs.polarislibrary.com/adfs/",
    "Audience": "microsoft:identityserver:0a586b1e-eeb0-4c8a-8381-50e9cafec240",
    "ValidIssuer": "http://dev-fs.polarislibrary.com/adfs/services/trust",
    "ValidAudience": "microsoft:identityserver:0a586b1e-eeb0-4c8a-8381-50e9cafec240",
    "AuthorizationUrl": "https://dev-fs.polarislibrary.com/adfs/oauth2/authorize",
    "TokenUrl": "https://dev-fs.polarislibrary.com/adfs/oauth2/token"
  },
```

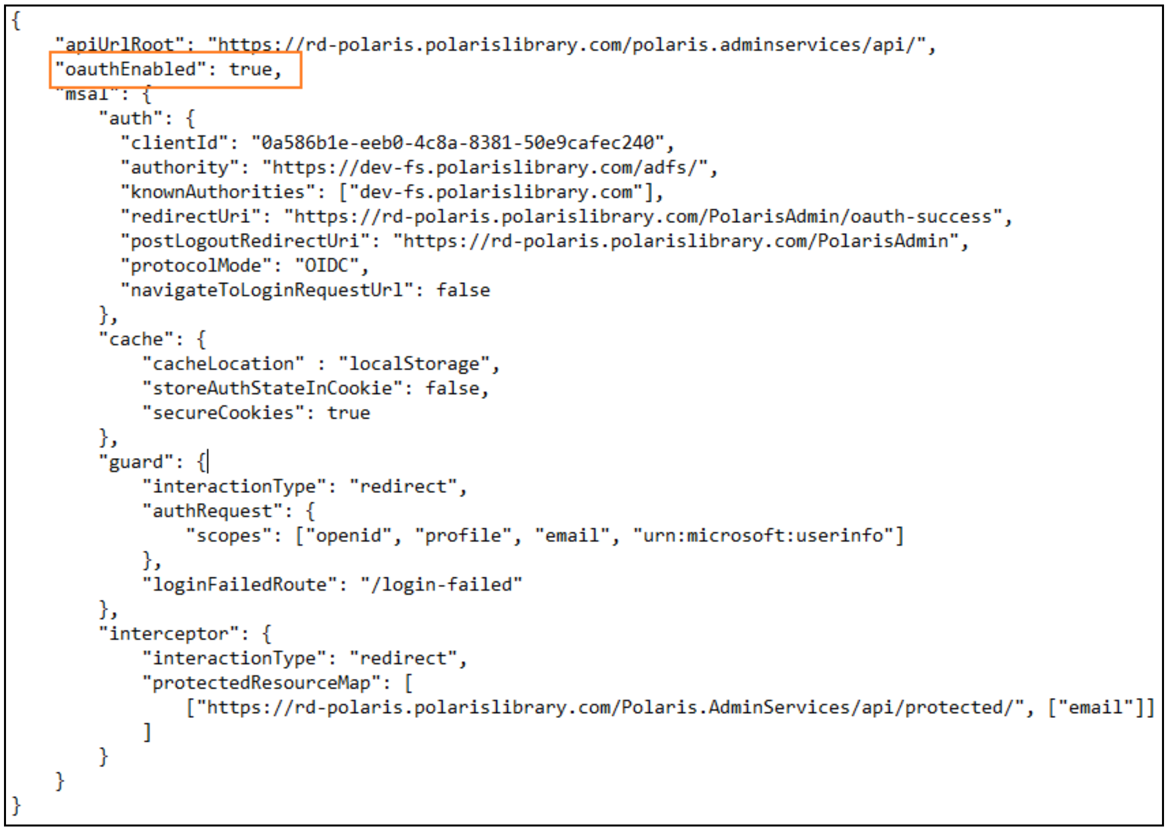
Set Up PolarisAdmin

To set up PolarisAdmin

> **Note:**
> When you edit the appsettings.user.json file, you must run the editing application (for example, Notepad) as administrator.

Verify that OAuth is Enabled

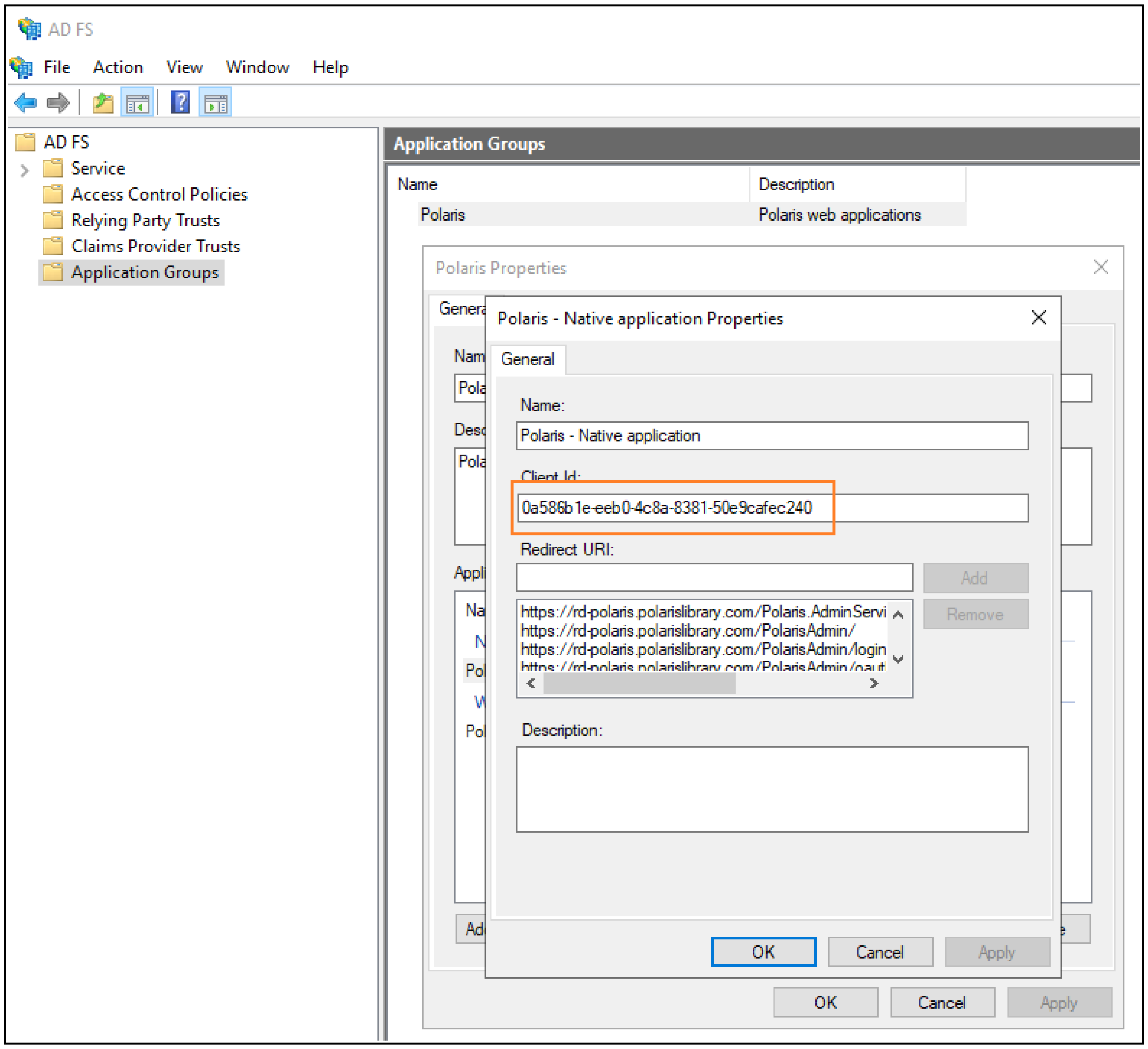
- Open C:\Program Files\Polaris\7.4\PolarisAdmin\assets\appsettings.user.json and verify that `oauthEnabled` is set to `true`.



```
{
    "apiUrlRoot": "https://rd-polaris.polarislibrary.com/polaris.adminservices/api/",
    "oauthEnabled": true,
    "msal": {
        "auth": {
          "clientId": "0a586b1e-eeb0-4c8a-8381-50e9cafec240",
          "authority": "https://dev-fs.polarislibrary.com/adfs/",
          "knownAuthorities": ["dev-fs.polarislibrary.com"],
          "redirectUri": "https://rd-polaris.polarislibrary.com/PolarisAdmin/oauth-success",
          "postLogoutRedirectUri": "https://rd-polaris.polarislibrary.com/PolarisAdmin",
          "protocolMode": "OIDC",
          "navigateToLoginRequestUrl": false
        },
        "cache": {
            "cacheLocation" : "localStorage",
            "storeAuthStateInCookie": false,
            "secureCookies": true
        },
        "guard": {
            "interactionType": "redirect",
            "authRequest": {
                "scopes": ["openid", "profile", "email", "urn:microsoft:userinfo"]
            },
            "loginFailedRoute": "/login-failed"
        },
        "interceptor": {
            "interactionType": "redirect",
            "protectedResourceMap": [
                ["https://rd-polaris.polarislibrary.com/Polaris.AdminServices/api/protected/", ["email"]]
            ]
        }
    }
}
```

Update the Client ID

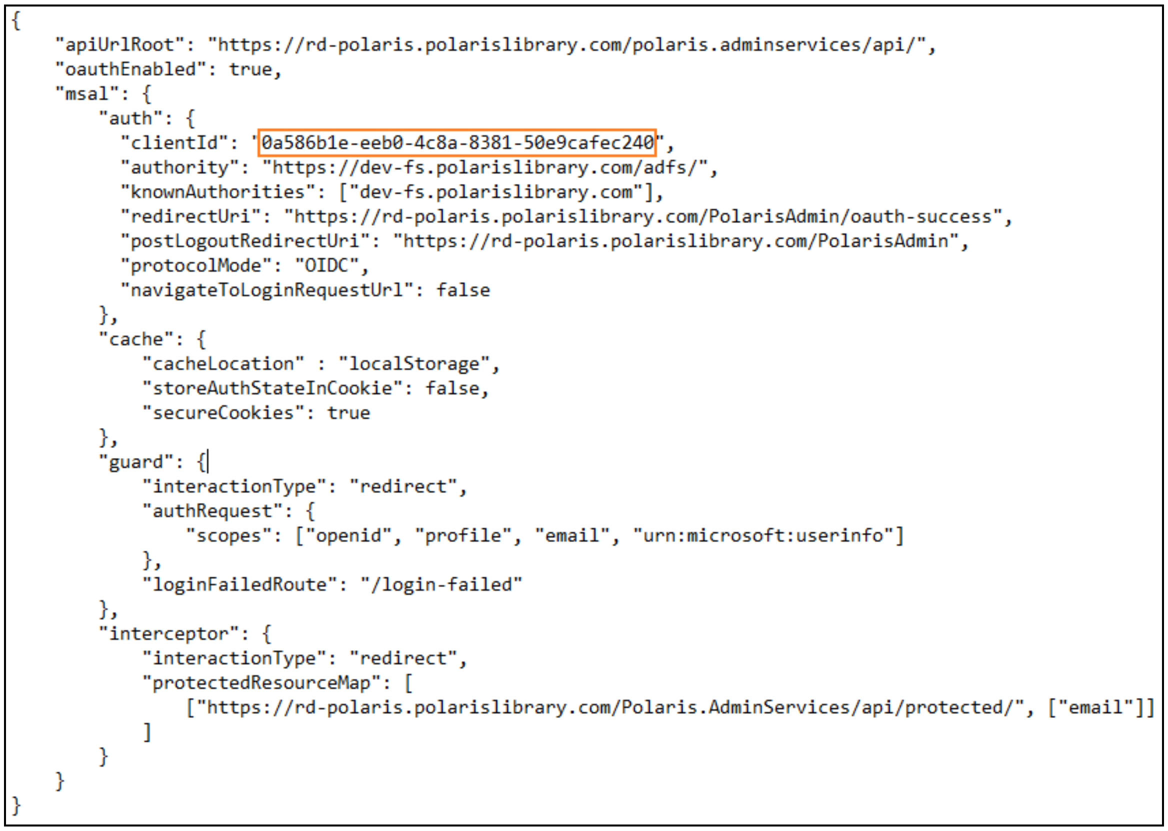
1. On the AD FS server, open AD FS Management desktop application.



Name:
Polaris - Native application
Client Id:
0a586b1e-eeb0-4c8a-8381-50e9cafec240
Redirect URI:
https://rd-polaris.polarislibrary.com/Polaris.AdminServi
https://rd-polaris.polarislibrary.com/PolarisAdmin/
https://rd-polaris.polarislibrary.com/PolarisAdmin/login
https://rd-polaris.polarislibrary.com/PolarisAdmin/oaut
Description:

2. Copy the client ID from the Polaris - Native application Properties dialog.
3. Paste the copied client ID into the appsettings.user.json file.
4. If you started from the template, replace `[CLIENTID-ASSIGNED-IN-ADFS]` with the copied client ID.

   It should look like the following when complete (your client ID will be different):

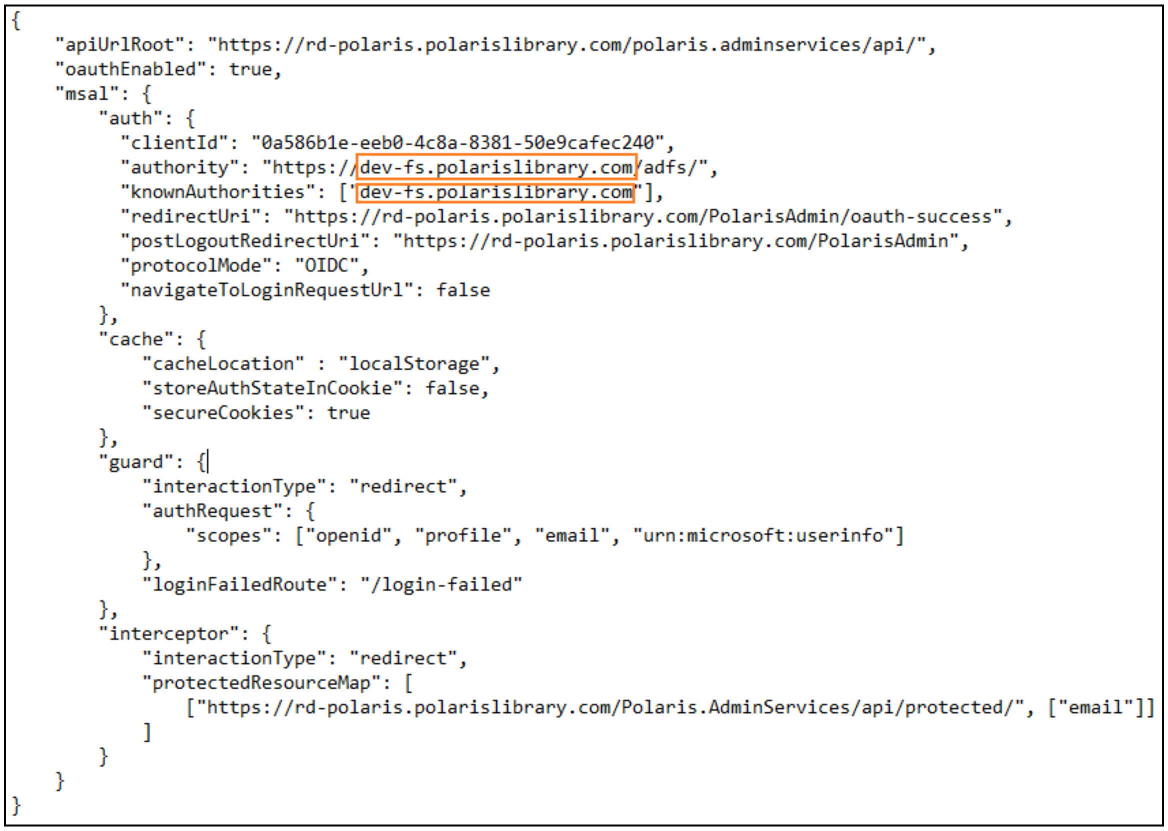


```
{
    "apiUrlRoot": "https://rd-polaris.polarislibrary.com/polaris.adminservices/api/",
    "oauthEnabled": true,
    "msal": {
        "auth": {
          "clientId": "0a586b1e-eeb0-4c8a-8381-50e9cafec240",
          "authority": "https://dev-fs.polarislibrary.com/adfs/",
          "knownAuthorities": ["dev-fs.polarislibrary.com"],
          "redirectUri": "https://rd-polaris.polarislibrary.com/PolarisAdmin/oauth-success",
          "postLogoutRedirectUri": "https://rd-polaris.polarislibrary.com/PolarisAdmin",
          "protocolMode": "OIDC",
          "navigateToLoginRequestUrl": false
        },
        "cache": {
            "cacheLocation" : "localStorage",
            "storeAuthStateInCookie": false,
            "secureCookies": true
        },
        "guard": {
            "interactionType": "redirect",
            "authRequest": {
                "scopes": ["openid", "profile", "email", "urn:microsoft:userinfo"]
            },
            "loginFailedRoute": "/login-failed"
        },
        "interceptor": {
            "interactionType": "redirect",
            "protectedResourceMap": [
                ["https://rd-polaris.polarislibrary.com/Polaris.AdminServices/api/protected/", ["email"]]
            ]
        }
    }
}
```

Update AD FS Server Location

- If you started from the template, replace *[ADFS-SERVER-ADDR]* with the AD FS server address.

  It should look like the following when complete (your AD FS server address will be different):

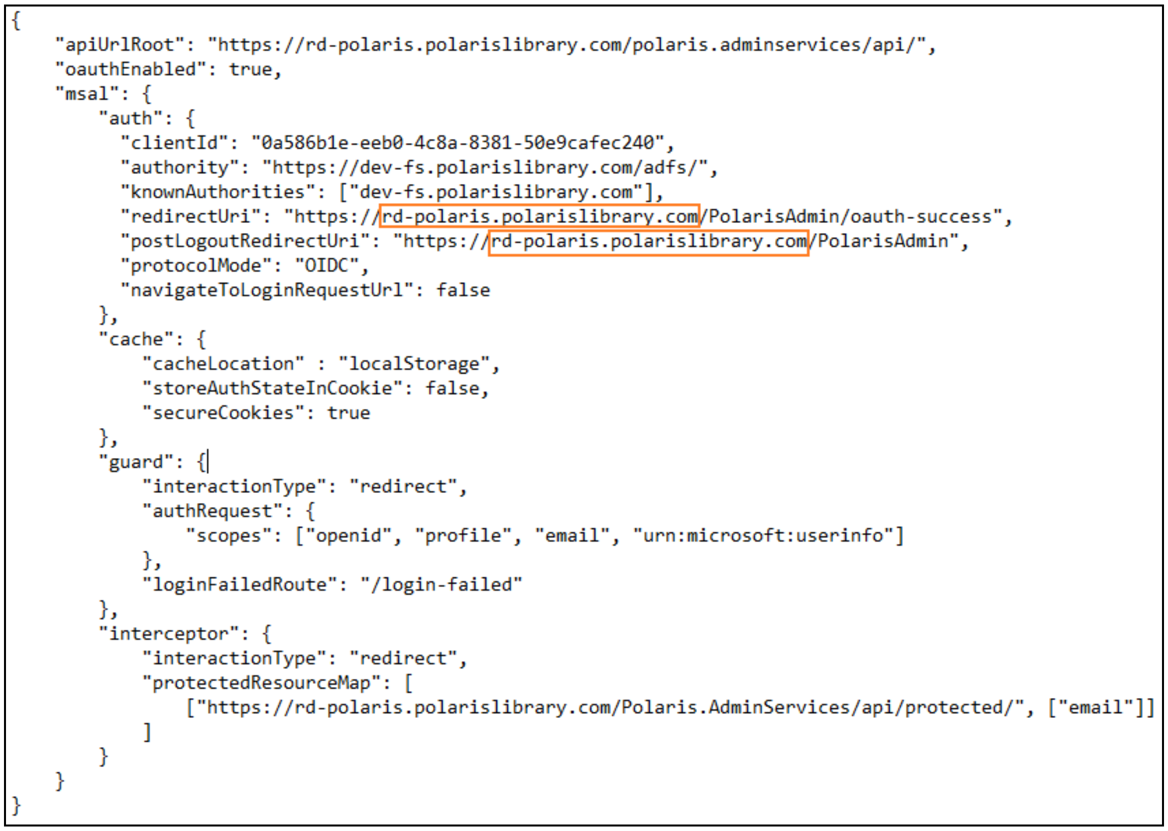


```
{
    "apiUrlRoot": "https://rd-polaris.polarislibrary.com/polaris.adminservices/api/",
    "oauthEnabled": true,
    "msal": {
        "auth": {
          "clientId": "0a586b1e-eeb0-4c8a-8381-50e9cafec240",
          "authority": "https://dev-fs.polarislibrary.com/adfs/",
          "knownAuthorities": ['dev-fs.polarislibrary.com'],
          "redirectUri": "https://rd-polaris.polarislibrary.com/PolarisAdmin/oauth-success",
          "postLogoutRedirectUri": "https://rd-polaris.polarislibrary.com/PolarisAdmin",
          "protocolMode": "OIDC",
          "navigateToLoginRequestUrl": false
        },
        "cache": {
            "cacheLocation" : "localStorage",
            "storeAuthStateInCookie": false,
            "secureCookies": true
        },
        "guard": {
            "interactionType": "redirect",
            "authRequest": {
                "scopes": ["openid", "profile", "email", "urn:microsoft:userinfo"]
            },
            "loginFailedRoute": "/login-failed"
        },
        "interceptor": {
            "interactionType": "redirect",
            "protectedResourceMap": [
                ["https://rd-polaris.polarislibrary.com/Polaris.AdminServices/api/protected/", ["email"]]
            ]
        }
    }
}
```

Update Polaris Admin Server Location

- If you started from the template, replace *[POLADMIN-SERVER-ADDR]* with the AD FS server address.

  It should look like the following image when complete (your AD FS server address will be different):

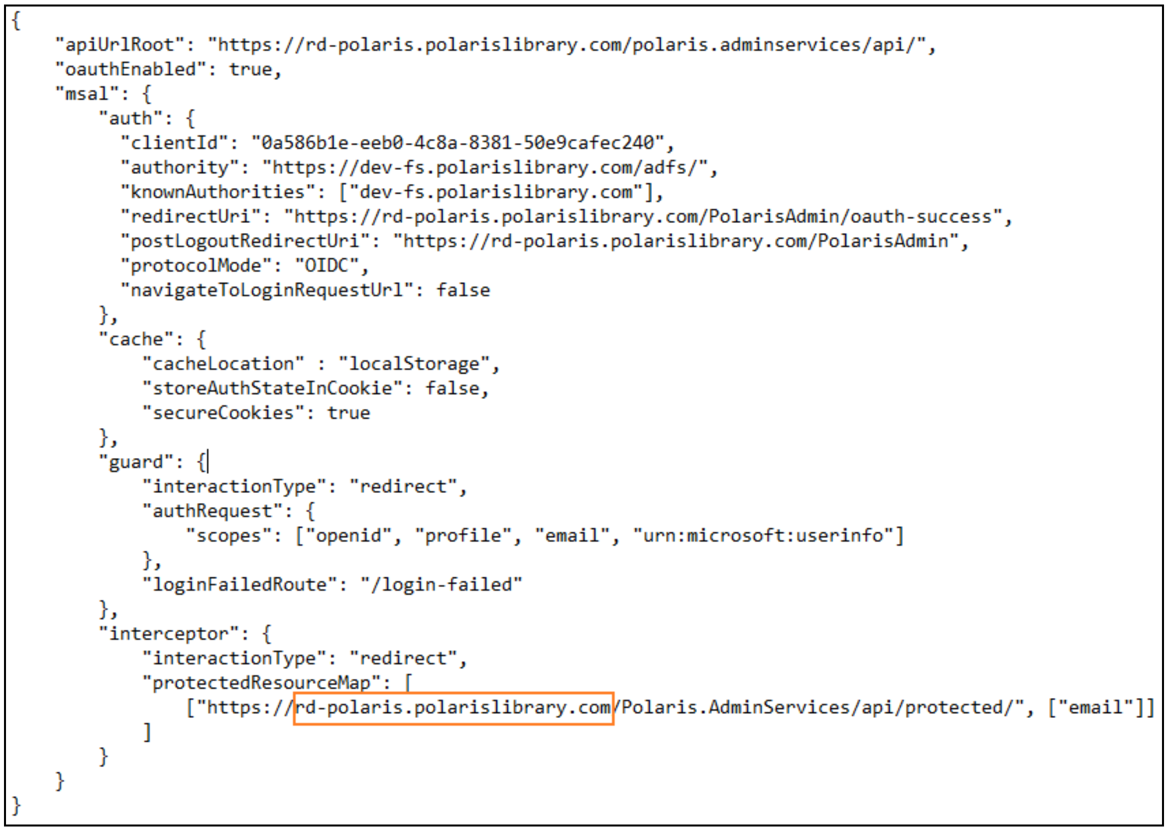


```
{
    "apiUrlRoot": "https://rd-polaris.polarislibrary.com/polaris.adminservices/api/",
    "oauthEnabled": true,
    "msal": {
        "auth": {
          "clientId": "0a586b1e-eeb0-4c8a-8381-50e9cafec240",
          "authority": "https://dev-fs.polarislibrary.com/adfs/",
          "knownAuthorities": ["dev-fs.polarislibrary.com"],
          "redirectUri": "https://rd-polaris.polarislibrary.com/PolarisAdmin/oauth-success",
          "postLogoutRedirectUri": "https://rd-polaris.polarislibrary.com/PolarisAdmin",
          "protocolMode": "OIDC",
          "navigateToLoginRequestUrl": false
        },
        "cache": {
            "cacheLocation" : "localStorage",
            "storeAuthStateInCookie": false,
            "secureCookies": true
        },
        "guard": {
            "interactionType": "redirect",
            "authRequest": {
                "scopes": ["openid", "profile", "email", "urn:microsoft:userinfo"]
            },
            "loginFailedRoute": "/login-failed"
        },
        "interceptor": {
            "interactionType": "redirect",
            "protectedResourceMap": [
                ["https://rd-polaris.polarislibrary.com/Polaris.AdminServices/api/protected/", ["email"]]
            ]
        }
    }
}
```

Update Polaris Admin Services (API) Server Location

- If you started from the template, replace *[POLADMINSVC-SERVER-ADDR]* with the AD FS server address.

  It should look like the following image when complete (your AD FS server address will be different):



```
{
    "apiUrlRoot": "https://rd-polaris.polarislibrary.com/polaris.adminservices/api/",
    "oauthEnabled": true,
    "msal": {
        "auth": {
          "clientId": "0a586b1e-eeb0-4c8a-8381-50e9cafec240",
          "authority": "https://dev-fs.polarislibrary.com/adfs/",
          "knownAuthorities": ["dev-fs.polarislibrary.com"],
          "redirectUri": "https://rd-polaris.polarislibrary.com/PolarisAdmin/oauth-success",
          "postLogoutRedirectUri": "https://rd-polaris.polarislibrary.com/PolarisAdmin",
          "protocolMode": "OIDC",
          "navigateToLoginRequestUrl": false
        },
        "cache": {
            "cacheLocation" : "localStorage",
            "storeAuthStateInCookie": false,
            "secureCookies": true
        },
        "guard": {
            "interactionType": "redirect",
            "authRequest": {
                "scopes": ["openid", "profile", "email", "urn:microsoft:userinfo"]
            },
            "loginFailedRoute": "/login-failed"
        },
        "interceptor": {
            "interactionType": "redirect",
            "protectedResourceMap": [
                ["https://rd-polaris.polarislibrary.com/Polaris.AdminServices/api/protected/", ["email"]]
            ]
        }
    }
}
```

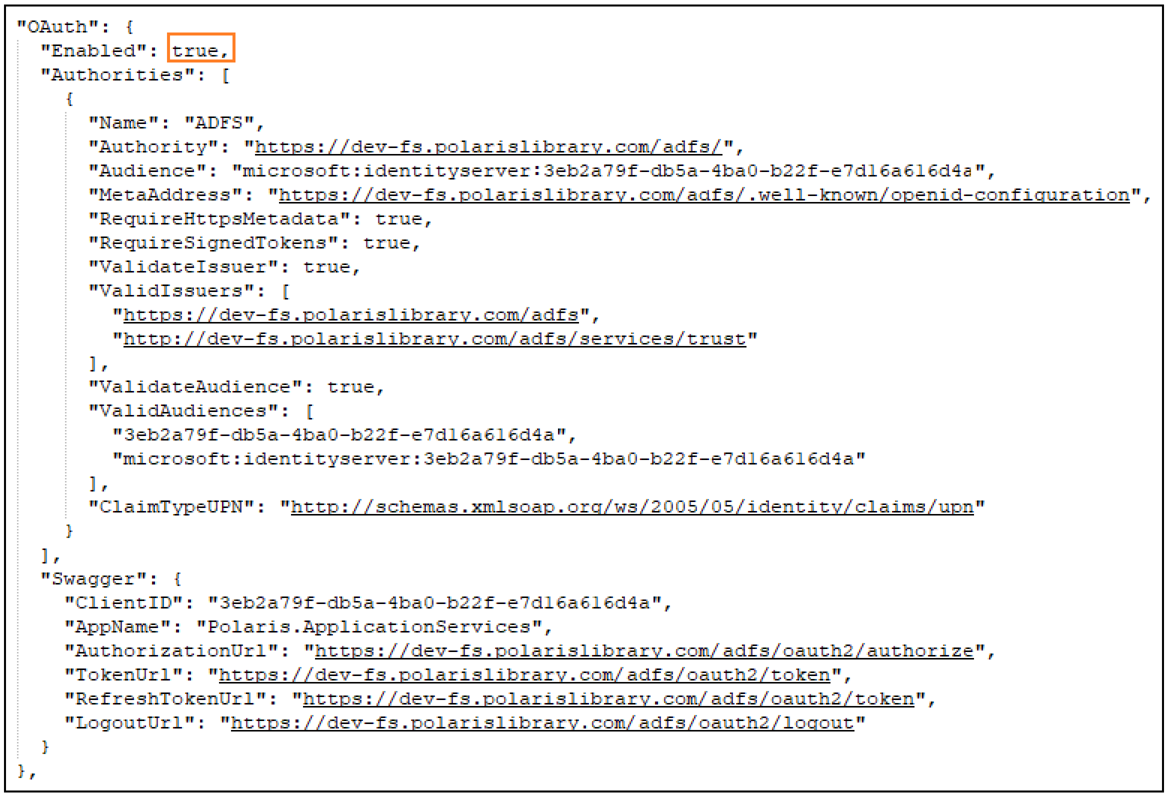
Set Up Polaris.ApplicationServices

To set up Polaris.ApplicationServices

> **Note:**
> When you edit the appsettings.user.json file, you must run the editing application (for example, Notepad) as administrator.

Verify that OAuth Is Enabled

- Open C:\Program Files\Polaris\7.4\Polaris.ApplicationServices\appsettings.user.json and verify that `OAuth.Enabled` is set to `true`.

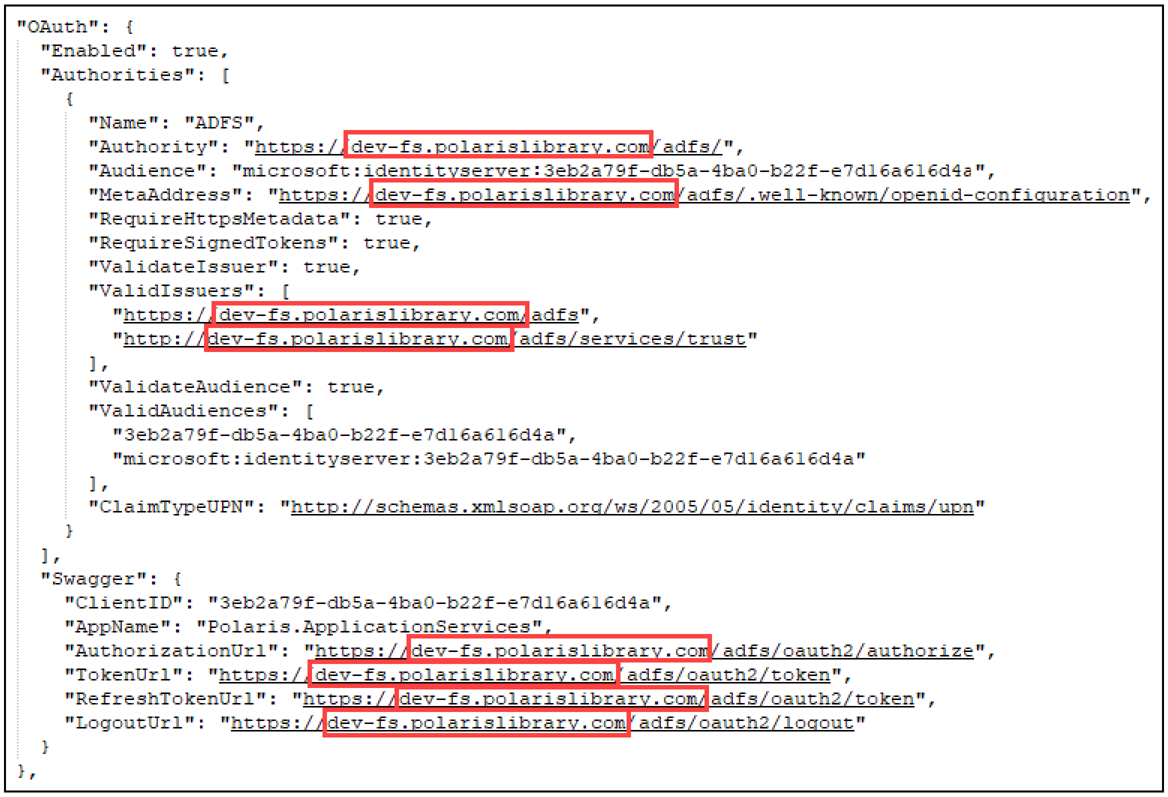


```
"OAuth": {
  "Enabled": true,
  "Authorities": [
    {
      "Name": "ADFS",
      "Authority": "https://dev-fs.polarislibrary.com/adfs/",
      "Audience": "microsoft:identityserver:3eb2a79f-db5a-4ba0-b22f-e7d16a616d4a",
      "MetaAddress": "https://dev-fs.polarislibrary.com/adfs/.well-known/openid-configuration",
      "RequireHttpsMetadata": true,
      "RequireSignedTokens": true,
      "ValidateIssuer": true,
      "ValidIssuers": [
        "https://dev-fs.polarislibrary.com/adfs",
        "http://dev-fs.polarislibrary.com/adfs/services/trust"
      ],
      "ValidateAudience": true,
      "ValidAudiences": [
        "3eb2a79f-db5a-4ba0-b22f-e7d16a616d4a",
        "microsoft:identityserver:3eb2a79f-db5a-4ba0-b22f-e7d16a616d4a"
      ],
      "ClaimTypeUPN": "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn"
    }
  ],
  "Swagger": {
    "ClientID": "3eb2a79f-db5a-4ba0-b22f-e7d16a616d4a",
    "AppName": "Polaris.ApplicationServices",
    "AuthorizationUrl": "https://dev-fs.polarislibrary.com/adfs/oauth2/authorize",
    "TokenUrl": "https://dev-fs.polarislibrary.com/adfs/oauth2/token",
    "RefreshTokenUrl": "https://dev-fs.polarislibrary.com/adfs/oauth2/token",
    "LogoutUrl": "https://dev-fs.polarislibrary.com/adfs/oauth2/logout"
  }
},
```

Update the AD FS Server Location

- If you started from the template, replace `adfs-server-address` with the AD FS server address.

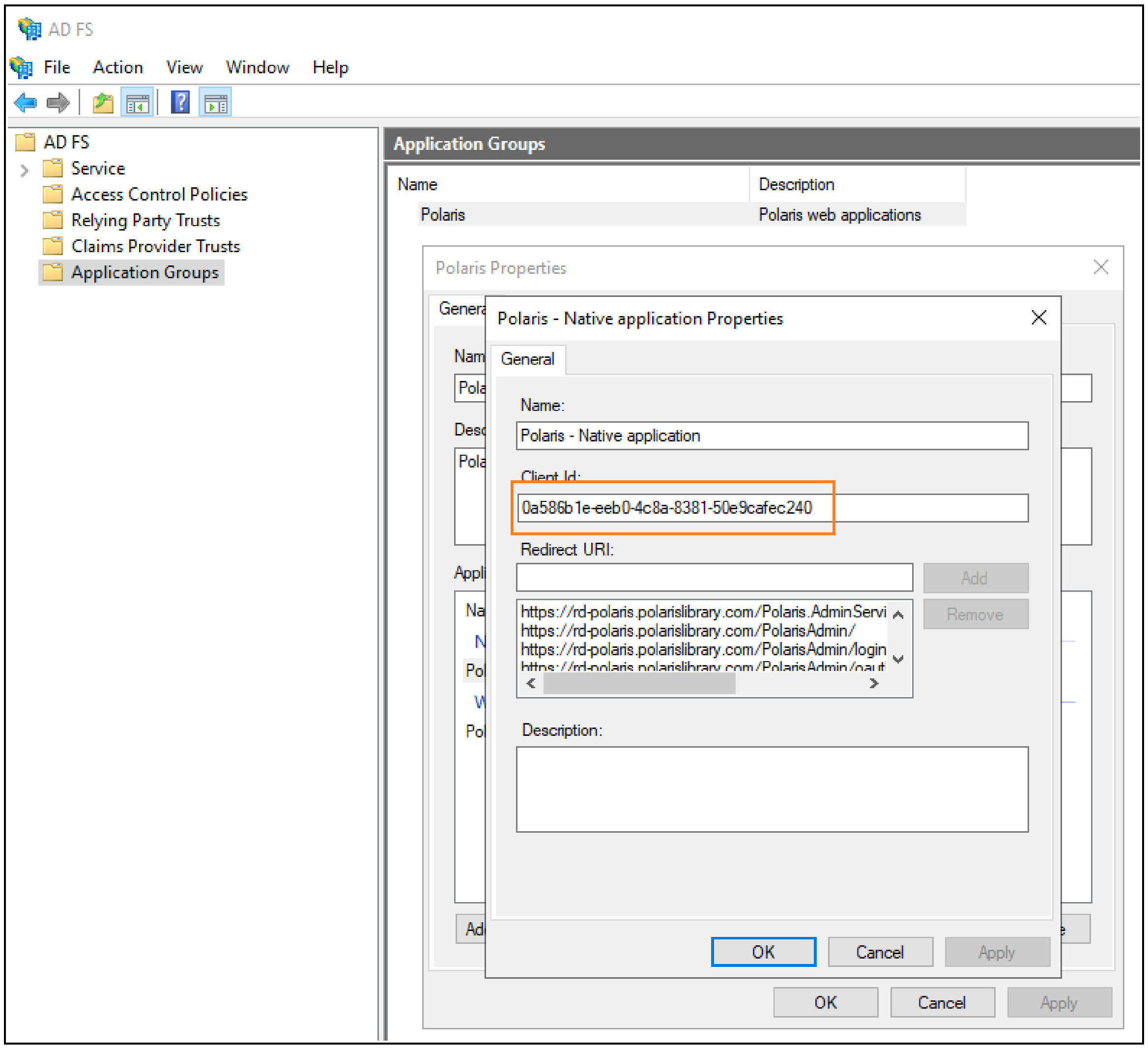
  It should look like the following when complete (your AD FS server address will be different):



```
"OAuth": {
  "Enabled": true,
  "Authorities": [
    {
      "Name": "ADFS",
      "Authority": "https://dev-fs.polarislibrary.com/adfs/",
      "Audience": "microsoft:identityserver:3eb2a79f-db5a-4ba0-b22f-e7d16a616d4a",
      "MetaAddress": "https://dev-fs.polarislibrary.com/adfs/.well-known/openid-configuration",
      "RequireHttpsMetadata": true,
      "RequireSignedTokens": true,
      "ValidateIssuer": true,
      "ValidIssuers": [
        "https://dev-fs.polarislibrary.com/adfs",
        "http://dev-fs.polarislibrary.com/adfs/services/trust"
      ],
      "ValidateAudience": true,
      "ValidAudiences": [
        "3eb2a79f-db5a-4ba0-b22f-e7d16a616d4a",
        "microsoft:identityserver:3eb2a79f-db5a-4ba0-b22f-e7d16a616d4a"
      ],
      "ClaimTypeUPN": "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn"
    }
  ],
  "Swagger": {
    "ClientID": "3eb2a79f-db5a-4ba0-b22f-e7d16a616d4a",
    "AppName": "Polaris.ApplicationServices",
    "AuthorizationUrl": "https://dev-fs.polarislibrary.com/adfs/oauth2/authorize",
    "TokenUrl": "https://dev-fs.polarislibrary.com/adfs/oauth2/token",
    "RefreshTokenUrl": "https://dev-fs.polarislibrary.com/adfs/oauth2/token",
    "LogoutUrl": "https://dev-fs.polarislibrary.com/adfs/oauth2/logout"
  }
},
```

Update the Client ID

1. On the AD FS server, open the AD FS Management desktop application.



Name	Description
Polaris	Polaris web applications

Name:
Polaris - Native application

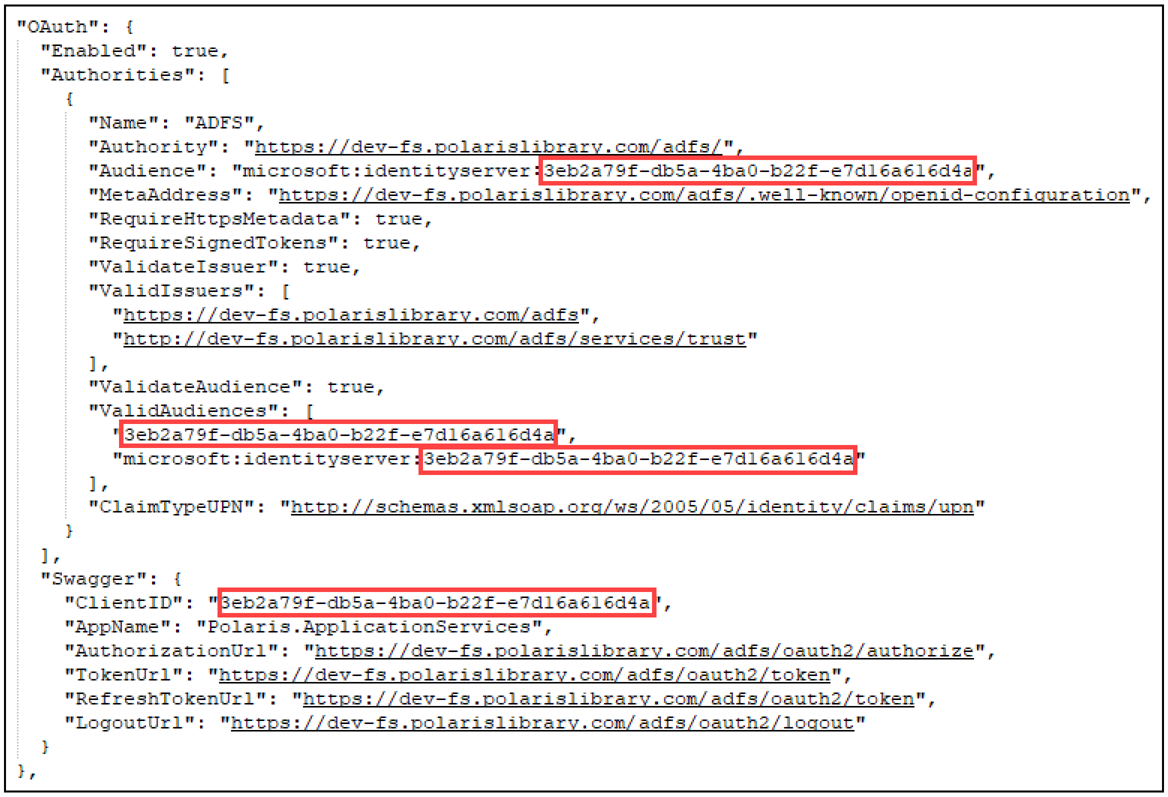
Client Id:
0a586b1e-eeb0-4c8a-8381-50e9cafec240

Redirect URI:
https://rd-polaris.polarislibrary.com/Polaris.AdminServi
https://rd-polaris.polarislibrary.com/PolarisAdmin/
https://rd-polaris.polarislibrary.com/PolarisAdmin/login
https://rd-polaris.polarislibrary.com/PolarisAdmin/oaut

Description:

2. Copy the client ID from the Polaris - Native application Properties dialog.
3. Paste the copied client ID into the appsettings.user.json file.
4. If you started from the template, replace `client-id-configured-in-adfs` with the copied client ID.

   It should look like the following when complete (your client ID will be different):



```
"OAuth": {
  "Enabled": true,
  "Authorities": [
    {
      "Name": "ADFS",
      "Authority": "https://dev-fs.polarislibrary.com/adfs/",
      "Audience": "microsoft:identityserver:3eb2a79f-db5a-4ba0-b22f-e7d16a616d4a",
      "MetaAddress": "https://dev-fs.polarislibrary.com/adfs/.well-known/openid-configuration",
      "RequireHttpsMetadata": true,
      "RequireSignedTokens": true,
      "ValidateIssuer": true,
      "ValidIssuers": [
        "https://dev-fs.polarislibrary.com/adfs",
        "http://dev-fs.polarislibrary.com/adfs/services/trust"
      ],
      "ValidateAudience": true,
      "ValidAudiences": [
        "3eb2a79f-db5a-4ba0-b22f-e7d16a616d4a",
        "microsoft:identityserver:3eb2a79f-db5a-4ba0-b22f-e7d16a616d4a"
      ],
      "ClaimTypeUPN": "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn"
    }
  ],
  "Swagger": {
    "ClientID": "3eb2a79f-db5a-4ba0-b22f-e7d16a616d4a",
    "AppName": "Polaris.ApplicationServices",
    "AuthorizationUrl": "https://dev-fs.polarislibrary.com/adfs/oauth2/authorize",
    "TokenUrl": "https://dev-fs.polarislibrary.com/adfs/oauth2/token",
    "RefreshTokenUrl": "https://dev-fs.polarislibrary.com/adfs/oauth2/token",
    "LogoutUrl": "https://dev-fs.polarislibrary.com/adfs/oauth2/logout"
  }
},
```

Set Up LeapWebApp

To set up LeapWebApp

> **Note:**
> When you edit the appsettings.user.json file, you must run the editing application (for example, Notepad) as administrator.

Verify that OAuth Is Enabled

- Open C:\Program Files\Polaris\7.4\LeapWebApp\appsettings.user.json and verify that `OAuthEnabled` is set to `true`.

```
"OAuthEnabled": true,
"OAuth": {
  "Authority": "https://dev-fs.polarislibrary.com/adfs/",
  "ClientSecret": null,
  "MetadataAddress": "https://dev-fs.polarislibrary.com/adfs/.well-known/openid-configuration",
  "KnownAuthorities": [ "dev-fs.polarislibrary.com" ],
  "CallbackPath": "/signin-oidc",
  "SignedOutCallbackPath": "/signout-callback-oidc",
  "SignedOutRedirectUri": "/login",
  "RemoteAuthenticationTimeout": 15,
  "RemoteFailureRedirectUri": "/leapwebapp/logout",
  "ResponseMode": "form_post",
  "ResponseType": "code id_token token",
  "UsePkce": true
  "UsePkce": true
},
```

Update the AD FS Server Location

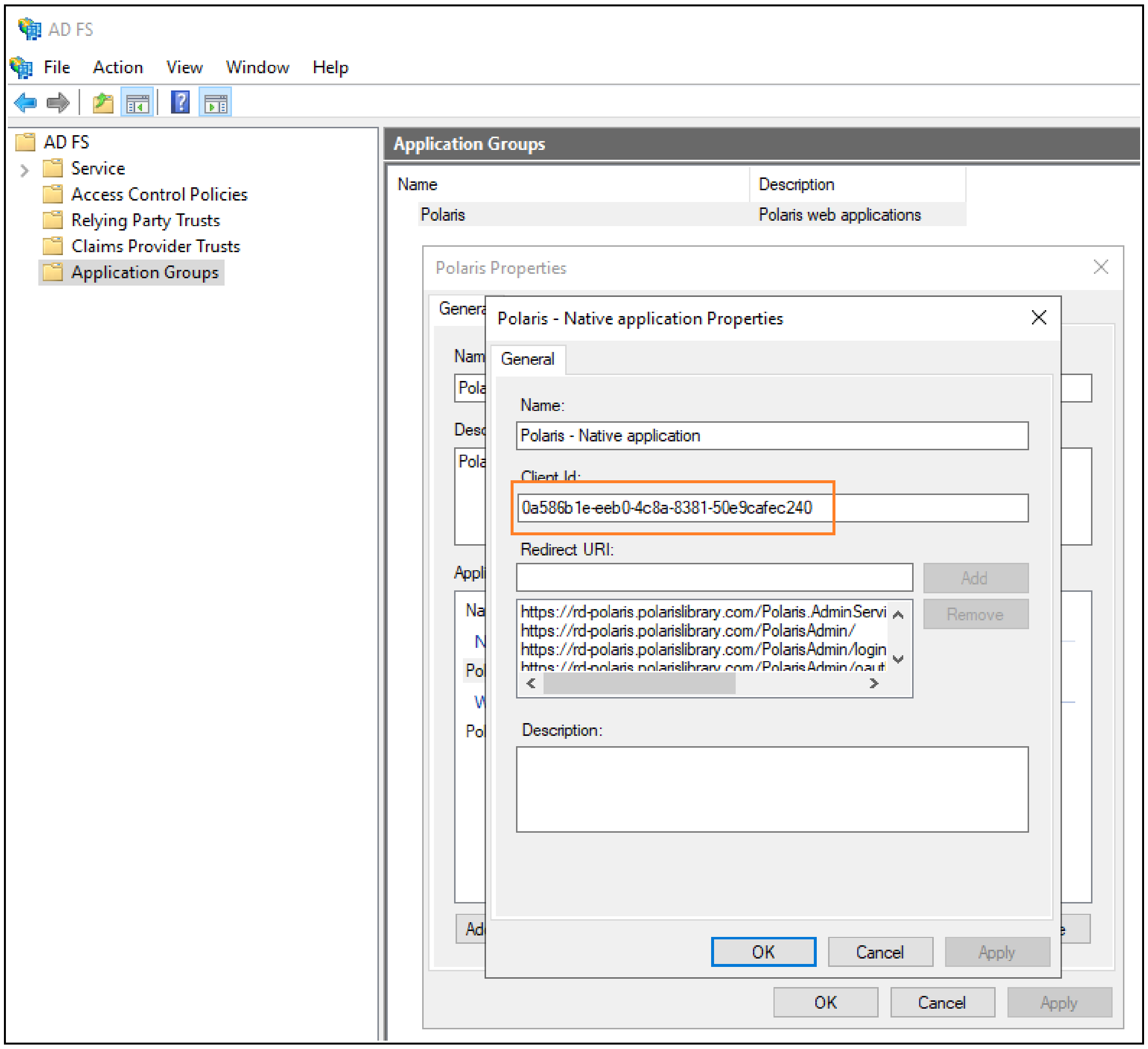
- If you started from the template, replace *`[adfs-server-address]`* with the AD FS server address.

  It should look like the following when complete (your AD FS server address will be different):

```
"OAuthEnabled": true,
"OAuth": {
  "Authority": "https://dev-fs.polarislibrary.com/adfs/",
  "ClientId": "0a586b1e-eeb0-4c8a-8381-50e9cafec240",
  "ClientSecret": null,
  "MetadataAddress": "https://dev-fs.polarislibrary.com/adfs/.well-known/openid-configuration",
  "KnownAuthorities": [ "dev-fs.polarislibrary.com" ],
  "CallbackPath": "/signin-oidc",
  "SignedOutCallbackPath": "/signout-callback-oidc",
  "SignedOutRedirectUri": "/login",
  "RemoteAuthenticationTimeout": 15,
  "RemoteFailureRedirectUri": "/leapwebapp/logout",
  "ResponseMode": "form_post",
  "ResponseType": "code id_token token",
  "UsePkce": true
},
```

Update the Client ID

1. On the AD FS server, open the AD FS Management desktop application.



Name	Description
Polaris	Polaris web applications

Name: Polaris - Native application

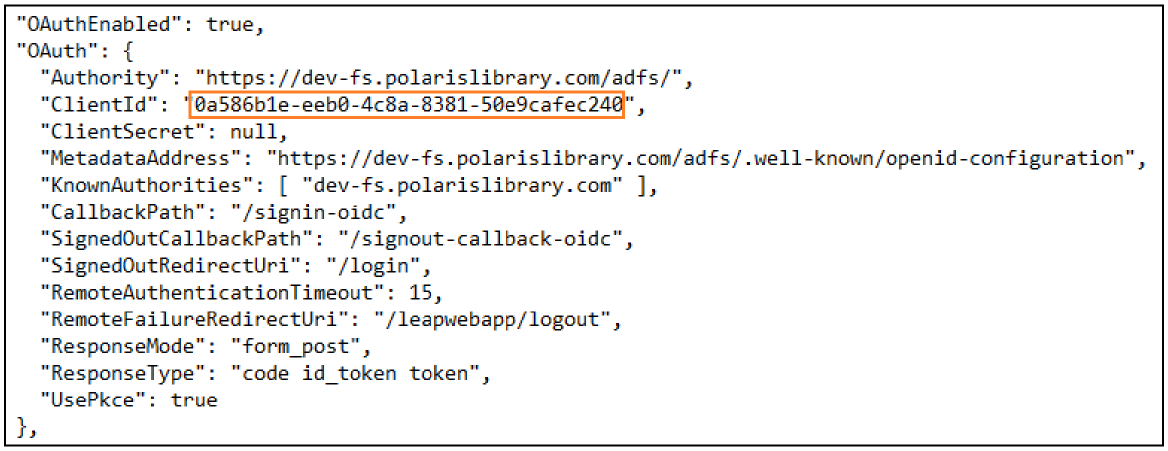
Client Id: 0a586b1e-eeb0-4c8a-8381-50e9cafec240

Redirect URI:
https://rd-polaris.polarislibrary.com/Polaris.AdminServi
https://rd-polaris.polarislibrary.com/PolarisAdmin/
https://rd-polaris.polarislibrary.com/PolarisAdmin/login
https://rd-polaris.polarislibrary.com/PolarisAdmin/oaut

Description:

2. Copy the client ID from the Polaris - Native application Properties dialog.
3. Paste the copied client ID into the appsettings.user.json file.
4. If you started from the template, replace `client-id-configured-in-adfs` with the copied client ID.

   It should look like the following when complete (your client ID will be different):



```
"OAuthEnabled": true,
"OAuth": {
  "Authority": "https://dev-fs.polarislibrary.com/adfs/",
  "ClientId": "0a586b1e-eeb0-4c8a-8381-50e9cafec240",
  "ClientSecret": null,
  "MetadataAddress": "https://dev-fs.polarislibrary.com/adfs/.well-known/openid-configuration",
  "KnownAuthorities": [ "dev-fs.polarislibrary.com" ],
  "CallbackPath": "/signin-oidc",
  "SignedOutCallbackPath": "/signout-callback-oidc",
  "SignedOutRedirectUri": "/login",
  "RemoteAuthenticationTimeout": 15,
  "RemoteFailureRedirectUri": "/leapwebapp/logout",
  "ResponseMode": "form_post",
  "ResponseType": "code id_token token",
  "UsePkce": true
},
```

(Optional) Disable Permission Overrides in Leap

By default, permission overrides are enabled in Leap.

To disable permission overrides

- Add the following value to the root level object in the .json file:

  `"client_permissionoverride_enabled": "false",`

> **Note:**
> If you disable permission overrides in Leap, we recommend that you also update the language string that controls the title of the Permission Override dialog. For more information, search for the SW_CI_ PERMISSION_OVRRD_TTL language string in the Polaris Web Admin Tool (Language Editor) Guide.

Enable Session Storage for LeapWebApp

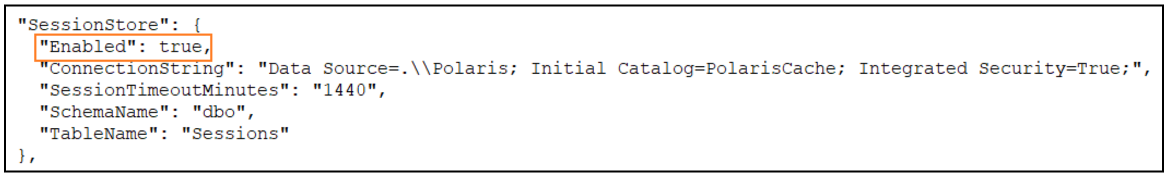
Enable session storage for the best user experience when using OAuth and OIDC.

Microsoft SQL Server Express 2019 (or a newer version) must be installed to use session storage. You install SQL Server Express separately. It is not part of the Leap installation.

To enable session storage

- Open C:\Program Files\Polaris\7.4\LeapWebApp\appsettings.user.json and set `SessionStore.Enabled` to `true`.

Note:
You must run the editing application (for example, Notepad) as administrator.



```
"SessionStore": {
  "Enabled": true,
  "ConnectionString": "Data Source=.\\Polaris; Initial Catalog=PolarisCache; Integrated Security=True;",
  "SessionTimeoutMinutes": "1440",
  "SchemaName": "dbo",
  "TableName": "Sessions"
},
```

Customize the AD FS Pages

Use the following resources to customize AD FS pages:

- https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/dn280950(v=ws.11)
  - Get-AdfsGlobalWebContent
  - Set-AdfsGlobalWebContent

    Examples:
    Customize the examples below to suit your library's needs.

    ```
    PS C:\Windows\system32> Set-AdfsGlobalWebContent -
    SignOutPageDescriptionText "You have successfully signed
    out.<br>If you have been directed here immediately after
    signing in, your session may have timed out."

    PS C:\Windows\system32> Set-AdfsWebTheme -TargetName
    default -Logo @{path="c:\ADFS Custom\leap_logo.png"}

    PS C:\Windows\system32> Set-AdfsGlobalWebContent -
    CompanyName "Polaris R&D"
    ```
- Advanced customization:
  - https://docs.microsoft.com/en-us/previous-versions/windows/it-pro/windows-server-2012-r2-and-2012/dn636121(v=ws.11)

Change the Access Token Lifetime

The default token lifetime for both access and ID tokens is 60 minutes. Execute the following command to increase the expiration time to 24 hours:

```
Set-AdfsWebApiApplication -TokenLifetime 1440 -TargetIdentifier
"0a586b1e-eeb0-4c8a-8381-50e9cafec240"
```

Note:
Replace *`TargetIdentifier`* with the Polaris Application Group native application client ID.

Bind a New SSL Certificate

If your web server certificate expires, use the instructions below to bind a new SSL certificate.

To bind a new SSL certificate

1. Install the certificate using Certificates Management.
2. Set the service communications certificate using the AD FS Management Console:
    a. Expand the Services folder.
    b. Select a new certificate.
    c. Restart the AD FS service.
3. Attach the certificate to AD FS using PowerShell:
    a. Get the certificate's thumbprint by viewing the certificate.

    ```
    c:\> Set-AdfsSslCertificate –Thumbprint
    e8fd5016542796214e94f72d76095f9fc587c731
    ```

    b. Restart the AD FS service.

Troubleshoot

Force a logout

- https://*AD FS server address*/adfs/oauth2/logout

Note:
Replace *AD FS server address* with your library's AD FS server address.

AD FS in one-way trust

Problem: Only local accounts are authenticating

Solution: Make sure the account running the AD FS service is a parent domain account and not a local account.

Receiving "User is not a valid Polaris user." error

- Check the setting Polaris.OAuth.ValidIssuer in the Polaris.AdminServices appsettings.user.json file.

  Example value: http://*AD FS server address*/adfs/services/trust

  Note:
  Replace *AD FS server address* with your library's AD FS server address.

- Verify a domain is attached to AD user accounts so the UPN claim can be added to the ID token's claims.

  The UPN claim should look like user@mydomain.com.

Troubleshoot Redirect URIs

Redirect URIs are case-sensitive.

Configuring Azure AD

Important:

The mechanism used to connect an Azure AD user to a Polaris user is the user principal name (UPN) in the format of an email address. For example, user@mydomain.com.

During the account verification process, we use the `openid` and `profile` scopes, which triggers Azure AD to return the `upn` claim or the `preferred_username` claim (or both). These must be returned in the name@domain format. The Polaris.ApplicationServices (API) can then use that information to map the Azure AD user to a Polaris user. If the `preferred_username` is a generic name, phone number, or other value, you can choose to apply the `email` scope to return the email.

See Configure LeapWebApp for Use with Azure AD for more information.

To configure Polaris OAuth support with Azure AD, perform the following tasks:

1. Register LeapWebApp with Azure AD.
2. Create client credentials.
3. Add authentication redirect URIs.
4. Expose the Polaris.ApplicationServices API.
5. Configure an ID token.
6. Set up users and groups.
7. Control access to LeapWebApp using Azure AD.
8. Set up web services and applications.

After you complete these tasks, Add a URL rewrite rule for LeapWebApp.

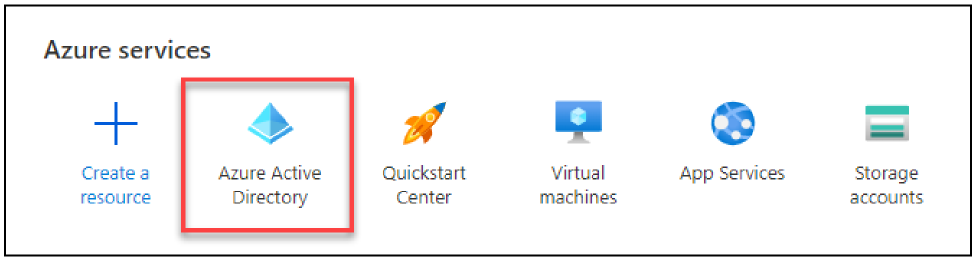
Register LeapWebApp with Azure AD

To register LeapWebApp with Azure AD

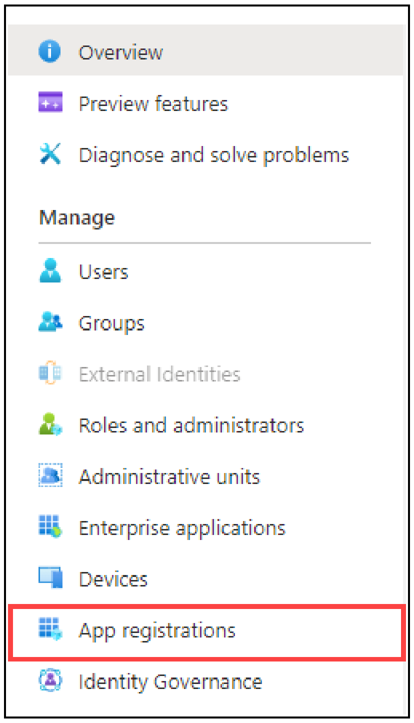
1. Sign in to the Azure portal:

   https://portal.azure.com/

2. In the **Azure services** list, select **Azure Active Directory**.



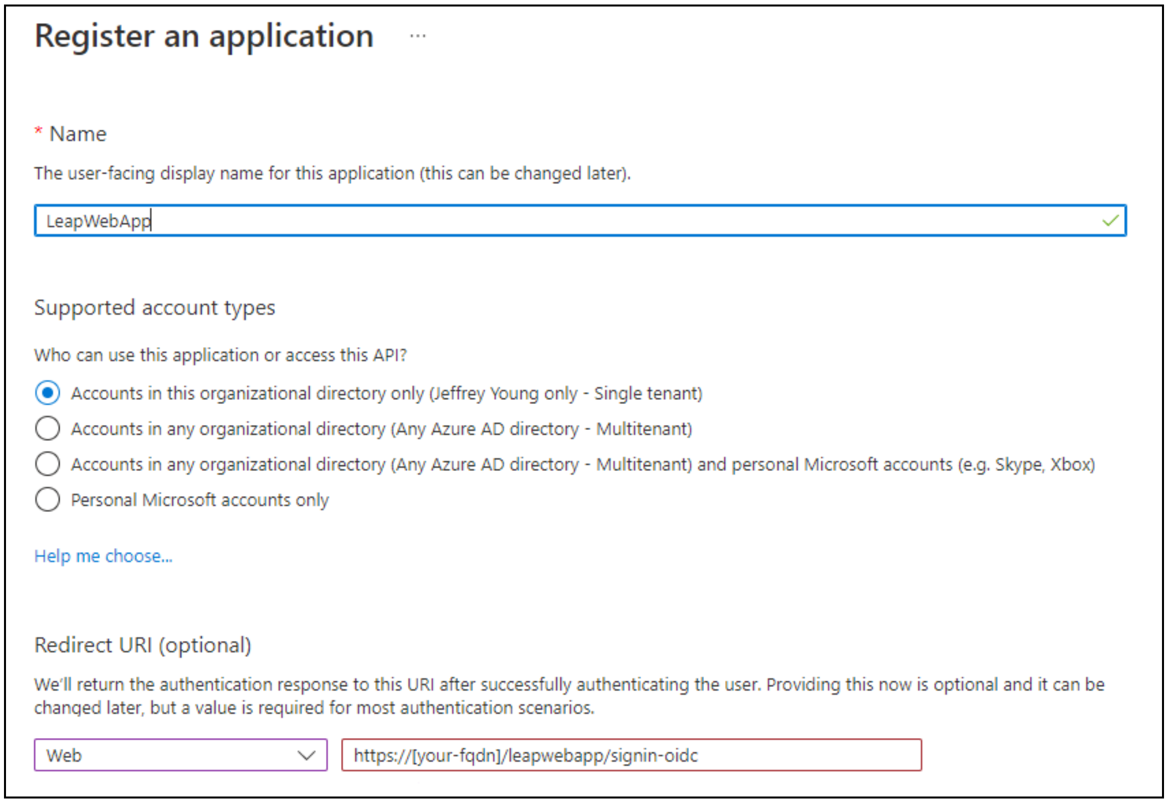
3. In the list of options at the left side of the screen, select **App registrations**.



The App registrations page appears.

4. Select **New registration**.

The Register an application dialog appears.



Register an application

* Name

The user-facing display name for this application (this can be changed later).

LeapWebApp

Supported account types

Who can use this application or access this API?

[● Accounts in this organizational directory only (Jeffrey Young only - Single tenant)]

○ Accounts in any organizational directory (Any Azure AD directory - Multitenant)

○ Accounts in any organizational directory (Any Azure AD directory - Multitenant) and personal Microsoft accounts (e.g. Skype, Xbox)

○ Personal Microsoft accounts only

Help me choose...

Redirect URI (optional)

We'll return the authentication response to this URI after successfully authenticating the user. Providing this now is optional and it can be changed later, but a value is required for most authentication scenarios.

Web | https://[your-fqdn]/leapwebapp/signin-oidc

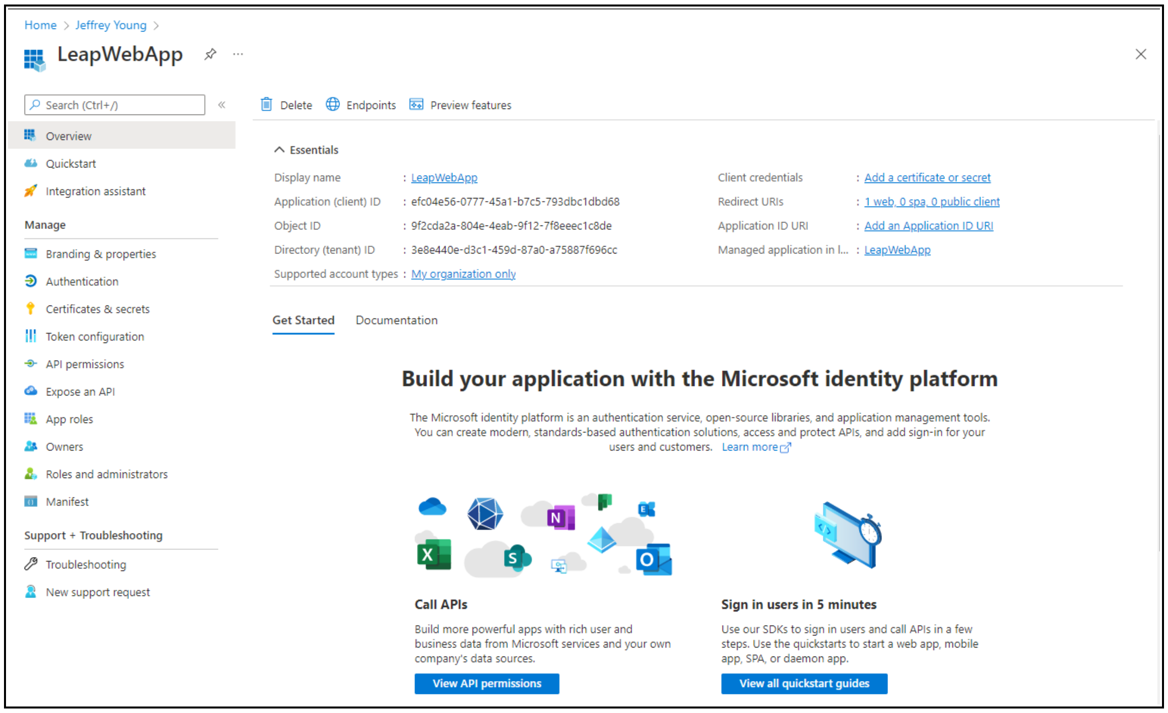
5. Enter "LeapWebApp" in the **Name** field.
6. Select an option from the **Supported account types** list.
7. Add a redirect URI:
   a. Select the **Web** URI type.
   b. Enter an address that uses the following format:

      https://*[FQDN]*/leapwebapp/signin-oidc

      Notes:

      - Replace *[FQDN]* with the fully-qualified domain name of your LeapWebApp server.
      - Example: https://leap.mylibrary.org/leapwebapp/signin-oidc
8. Select **Register**.

   The page for your new LeapWebApp application appears.


Home > Jeffrey Young >

LeapWebApp

Delete | Endpoints | Preview features

Essentials

Display name	: LeapWebApp	Client credentials	: Add a certificate or secret
Application (client) ID	: efc04e56-0777-45a1-b7c5-793dbc1dbd68	Redirect URIs	: 1 web, 0 spa, 0 public client
Object ID	: 9f2cda2a-804e-4eab-9f12-7f8eeec1c8de	Application ID URI	: Add an Application ID URI
Directory (tenant) ID	: 3e8e440e-d3c1-459d-87a0-a75887f696cc	Managed application in l...	: LeapWebApp
Supported account types	: My organization only		

Build your application with the Microsoft identity platform

The Microsoft identity platform is an authentication service, open-source libraries, and application management tools. You can create modern, standards-based authentication solutions, access and protect APIs, and add sign-in for your users and customers. Learn more

Call APIs

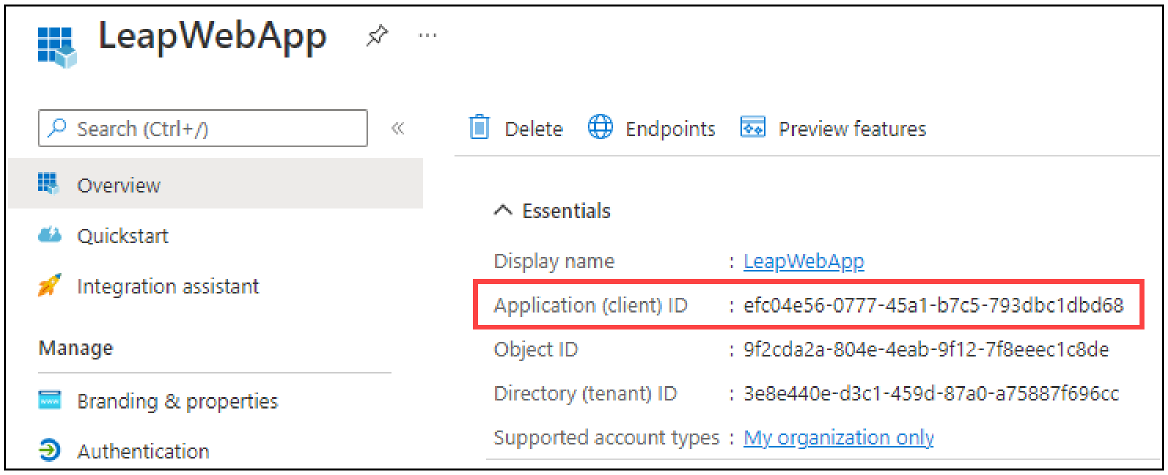
Build more powerful apps with rich user and business data from Microsoft services and your own company's data sources.

Sign in users in 5 minutes

Use our SDKs to sign in users and call APIs in a few steps. Use the quickstarts to start a web app, mobile app, SPA, or daemon app.

9. Copy the application (client) ID and paste it into Notepad (or a similar text editor) and save the file. You must have this value to complete several procedures later in the Azure AD configuration process.

   In the example below, the application (client) ID is "efc04e56-0777-45a1-b7c5-793dbc1dbbd68".


LeapWebApp

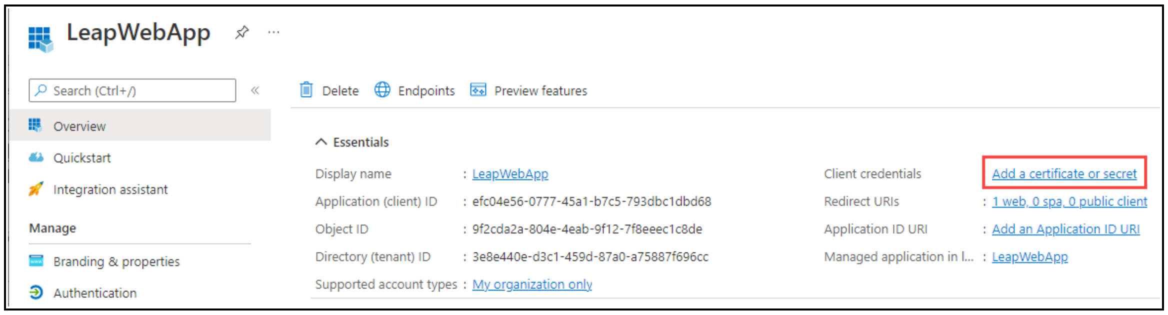
Essentials

Display name	: LeapWebApp
Application (client) ID	: efc04e56-0777-45a1-b7c5-793dbc1dbd68
Object ID	: 9f2cda2a-804e-4eab-9f12-7f8eeec1c8de
Directory (tenant) ID	: 3e8e440e-d3c1-459d-87a0-a75887f696cc
Supported account types	: My organization only

Create Client Credentials

To create client credentials for LeapWebApp

1. On the LeapWebApp page, select **Add a certificate or secret**.



LeapWebApp

Search (Ctrl+/) | Delete | Endpoints | Preview features

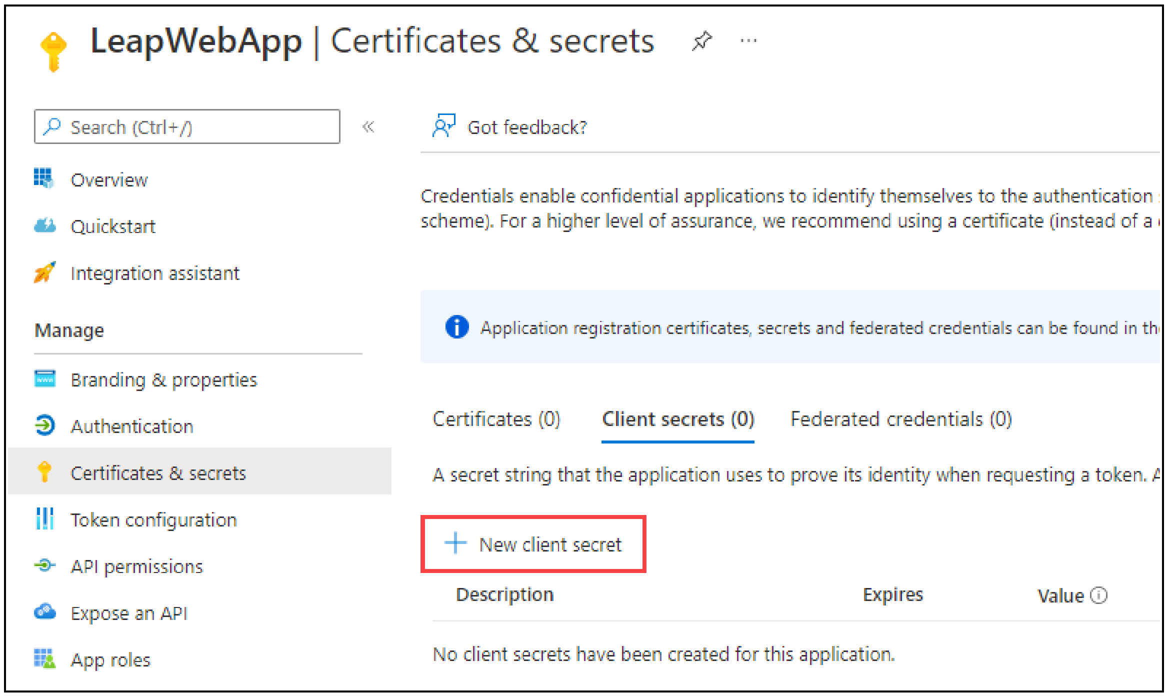
Overview
Quickstart
Integration assistant
Manage
Branding & properties
Authentication

Essentials

Display name	: LeapWebApp	Client credentials	: Add a certificate or secret
Application (client) ID	: efc04e56-0777-45a1-b7c5-793dbc1dbd68	Redirect URIs	: 1 web, 0 spa, 0 public client
Object ID	: 9f2cda2a-804e-4eab-9f12-7f8eeec1c8de	Application ID URI	: Add an Application ID URI
Directory (tenant) ID	: 3e8e440e-d3c1-459d-87a0-a75887f696cc	Managed application in l...	: LeapWebApp
Supported account types	: My organization only		

The Certificates & secrets page appears.

2. Select the **Client secrets** tab.
3. Select **New client secret**.



LeapWebApp | Certificates & secrets

Search (Ctrl+/)

Overview
Quickstart
Integration assistant
Manage
Branding & properties
Authentication
Certificates & secrets
Token configuration
API permissions
Expose an API
App roles

Got feedback?

Credentials enable confidential applications to identify themselves to the authentication
scheme). For a higher level of assurance, we recommend using a certificate (instead of a

Application registration certificates, secrets and federated credentials can be found in th

Certificates (0) | Client secrets (0) | Federated credentials (0)

A secret string that the application uses to prove its identity when requesting a token. A

New client secret

Description	Expires	Value
No client secrets have been created for this application.		

The Add a client secret dialog appears.

4. Enter a description in the **Description** field.
5. Select an option from the **Expires** list to specify when the client credentials expire.
6. Select **Add**.

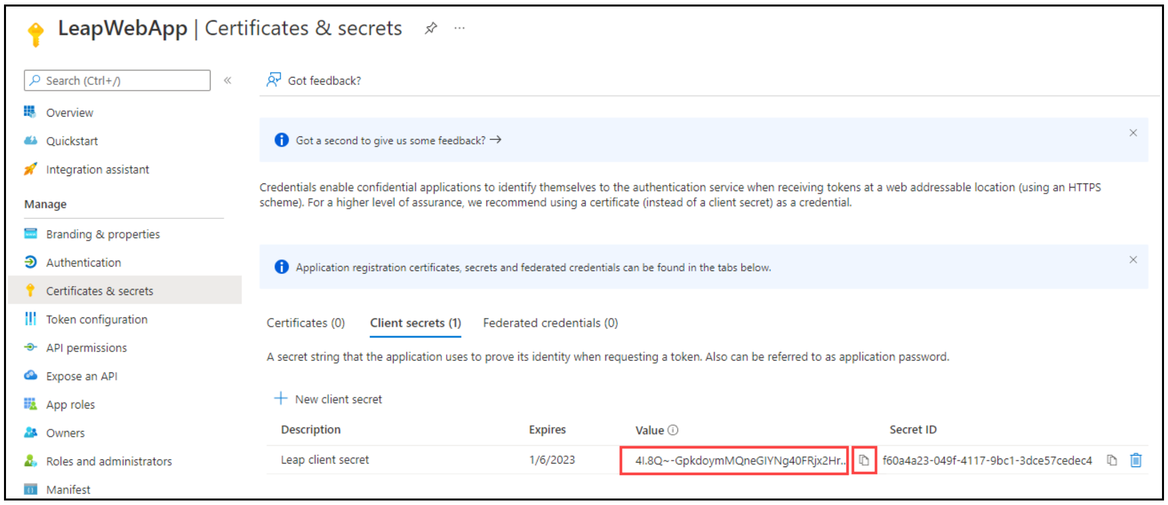
   The Add a client secret dialog closes. The client secret for LeapWebApp appears on the **Client secrets** tab of the Certificates & secrets page.

7. Copy the text in the **Value** column, then paste it into Notepad (or a similar text editor) and save the file. You must have this value to complete the Configure LeapWebApp for Use with Azure AD procedure.

   Important:

   - You must save this value now. Only a portion of the value appears when you return to the page later.
   - Use the copy icon to be sure that you are copying the entire value.

   In the example below, the value is "4I.8Q~-GpkdoymMQneGIYNg40FRjx2Hr1wWLDcbr".



Credentials enable confidential applications to identify themselves to the authentication service when receiving tokens at a web addressable location (using an HTTPS scheme). For a higher level of assurance, we recommend using a certificate (instead of a client secret) as a credential.

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

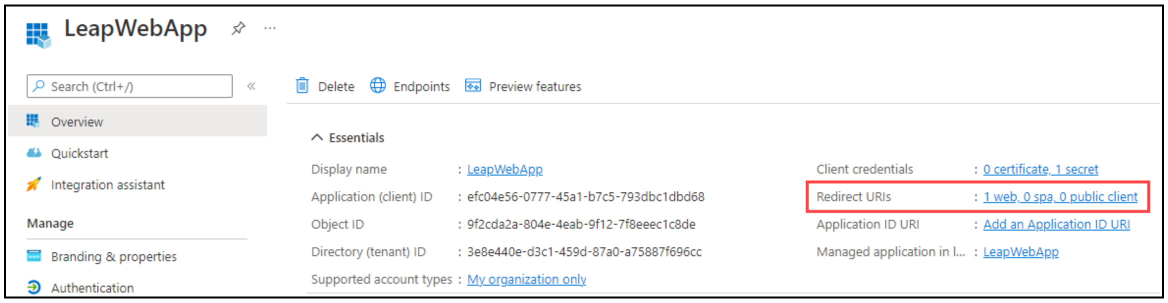
A secret string that the application uses to prove its identity when requesting a token. Also can be referred to as application password.

Description	Expires	Value	Secret ID
Leap client secret	1/6/2023	4I.8Q~-GpkdoymMQneGIYNg40FRjx2Hr...	f60a4a23-049f-4117-9bc1-3dce57cedec4

Add Authentication Redirect URIs

To add authentication redirect URIs

1. On the LeapWebApp page, select the link beside **Redirect URIs**.



Display name	: LeapWebApp
Application (client) ID	: efc04e56-0777-45a1-b7c5-793dbc1dbd68
Object ID	: 9f2cda2a-804e-4eab-9f12-7f8eeec1c8de
Directory (tenant) ID	: 3e8e440e-d3c1-459d-87a0-a75887f696cc
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 l...	: LeapWebApp

   The Authentication page appears.

2. Select **Add a platform**.

   The Configure platforms dialog appears.

3. Select the **Single-page application** tile.

   The Configure single-page application dialog appears.

4. In the **Redirect URIs** field, enter an address that uses the following format:

   https://*[FQDN]*/leapwebapp/signin-override-oidc

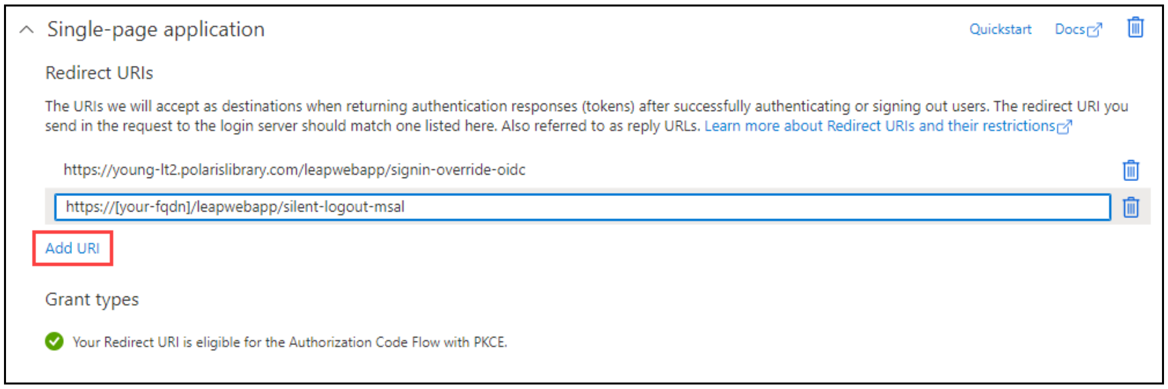
   Notes:

   - Replace *[FQDN]* with the fully-qualified domain name of your LeapWebApp server.
   - Example: https://leap.mylibrary.org/leapwebapp/signin-override-oidc

5. Select **Configure**.

   The new redirect URI appears on the Authentication page in the **Single-page application** list.

6. Select **Add URI**.



Redirect URIs

The URIs we will accept as destinations when returning authentication responses (tokens) after successfully authenticating or signing out users. The redirect URI you send in the request to the login server should match one listed here. Also referred to as reply URLs. Learn more about Redirect URIs and their restrictions

https://young-lt2.polarislibrary.com/leapwebapp/signin-override-oidc

https://[your-fqdn]/leapwebapp/silent-logout-msal

Add URI

Grant types

Your Redirect URI is eligible for the Authorization Code Flow with PKCE.

7. Enter an address that uses the following format:

   https://*[FQDN]*/leapwebapp/silent-logout-msal

   Notes:

   - Replace *[FQDN]* with the fully-qualified domain name of your LeapWebApp server.
   - Example: https://leap.mylibrary.org/leapwebapp/silent-logout-msal

8. Select **Add URI** again.
9. Enter an address that uses the following format:

   https://*[FQDN]*/Polaris.ApplicationServices/swagger/oauth2-redirect.html

   Notes:

   - Replace *[FQDN]* with the fully-qualified domain name of your LeapWebApp server.
   - Example: https://leap.mylibrary.org/Polaris.ApplicationServices/swagger/oauth2-redirect.html

10. Select **Add URI** again.
11. Enter an address that uses the following format:

    https://*[FQDN]*/PolarisAdmin/oauth-success

    Notes:

- Replace *[FQDN]* with the fully-qualified domain name of your LeapWebApp server.
- Example: https://leap.mylibrary.org/PolarisAdmin/oauth-success

12. Select **Add URI** again.
13. Enter an address that uses the following format:

    https://*[FQDN]*/Polaris.AdminServices/swagger/oauth2-redirect.html

    Notes:

    - Replace *[FQDN]* with the fully-qualified domain name of your LeapWebApp server.
    - Example: https://leap.mylibrary.org/Polaris.AdminServices/swagger/oauth2-redirect.html

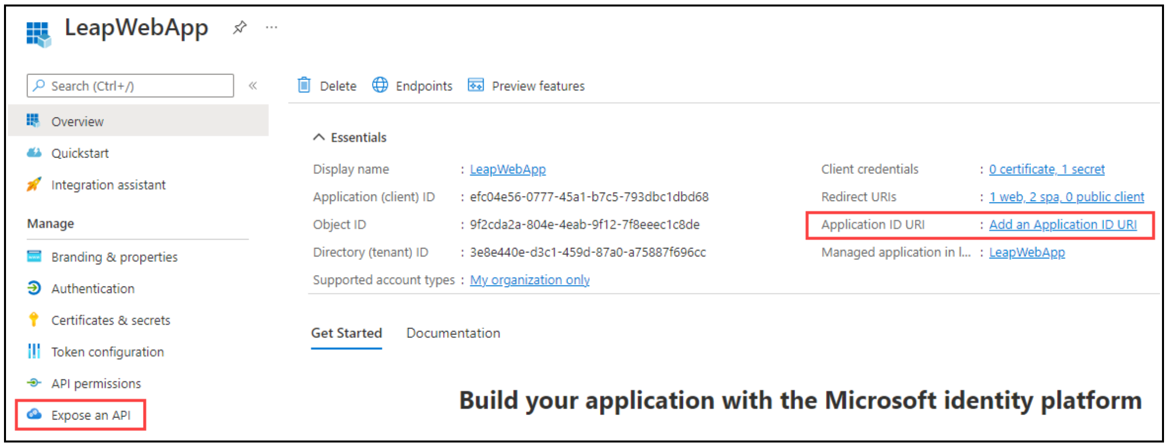
14. Select **Save** to save the five redirect URIs.

Expose the Polaris.ApplicationServices API

To expose the Polaris.ApplicationServices API

1. On the LeapWebApp page, select **Add an Application ID URI**.

   Note:
   You can also select **Expose an API** in the list of options at the left side of the screen.



LeapWebApp

Essentials

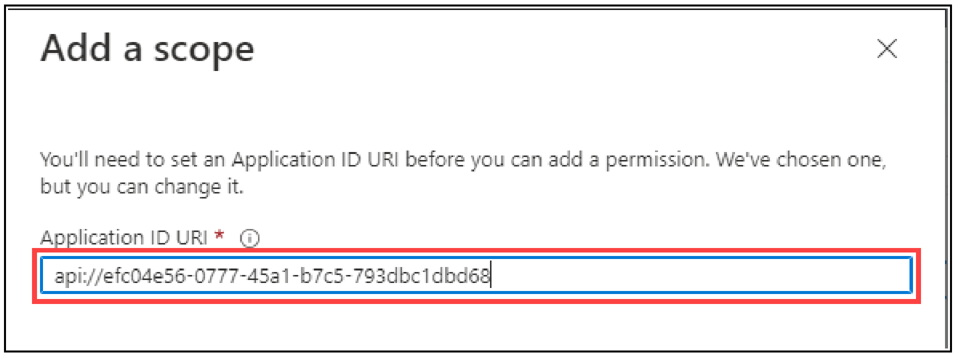
Display name : LeapWebApp
Application (client) ID : efc04e56-0777-45a1-b7c5-793dbc1dbd68
Object ID : 9f2cda2a-804e-4eab-9f12-7f8eeec1c8de
Directory (tenant) ID : 3e8e440e-d3c1-459d-87a0-a75887f696cc
Supported account types : My organization only
Client credentials : 0 certificate, 1 secret
Redirect URIs : 1 web, 2 spa, 0 public client
Application ID URI : Add an Application ID URI
Managed application in l... : LeapWebApp

Build your application with the Microsoft identity platform

   The Expose an API page appears.

2. Select **Add a scope**.

   The Add a scope dialog appears. The **Application ID URI** field contains an automatically-generated URI.



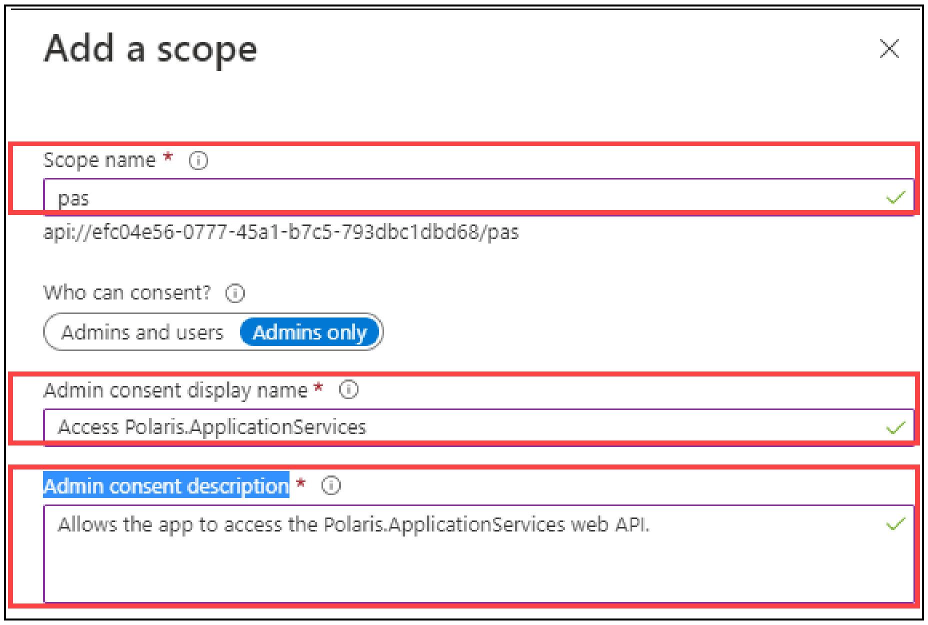
Add a scope

You'll need to set an Application ID URI before you can add a permission. We've chosen one, but you can change it.

Application ID URI * : api://efc04e56-0777-45a1-b7c5-793dbc1dbd68

3. Select **Save and continue**.

The Add a scope dialog refreshes.



Scope name *
pas
api://efc04e56-0777-45a1-b7c5-793dbc1dbd68/pas

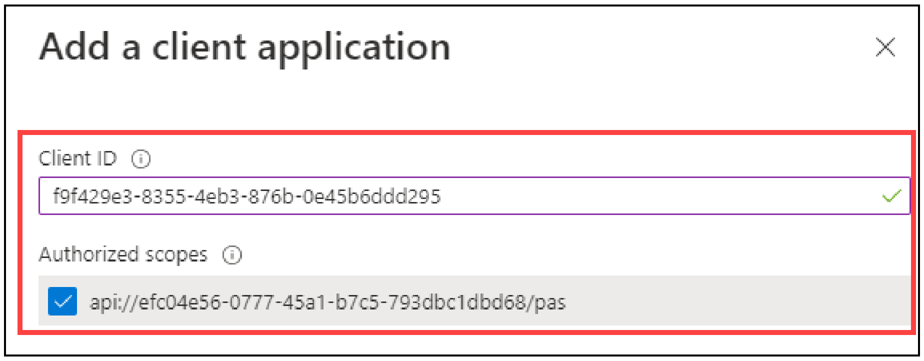
Admin consent display name *
Access Polaris.ApplicationServices

Admin consent description *
Allows the app to access the Polaris.ApplicationServices web API.

4. Enter "pas" in the **Scope name** field.
5. Enter "Access Polaris.ApplicationServices" in the **Admin consent display name** field.
6. Enter "Allows the app to access the Polaris.ApplicationServices web API." in the **Admin consent description** field.
7. Select **Add scope**.

   The Azure portal saves the scope and closes the Add a scope dialog.
8. On the Expose an API page, select **Add a client application**.

   The Add a client application dialog appears.



f9f429e3-8355-4eb3-876b-0e45b6ddd295

api://efc04e56-0777-45a1-b7c5-793dbc1dbd68/pas

9. In the **Client ID** field, enter the Application (client) ID that you copied and saved during the Register LeapWebApp with Azure AD procedure.
10. Select the **Authorized scopes** checkbox.
11. Select **Add application**.

    The Azure portal saves your changes and closes the Add a client application dialog.
12. On the Expose an API page, copy the new scope, then paste it into Notepad (or a similar text editor) and save the file. Your value will be similar to this one:

    api://efc04e56-0777-45a1-b7c5-793dbc1dbd68/pas

    You must have this value to complete two procedures later in the Azure AD configuration process.

Configure an ID Token

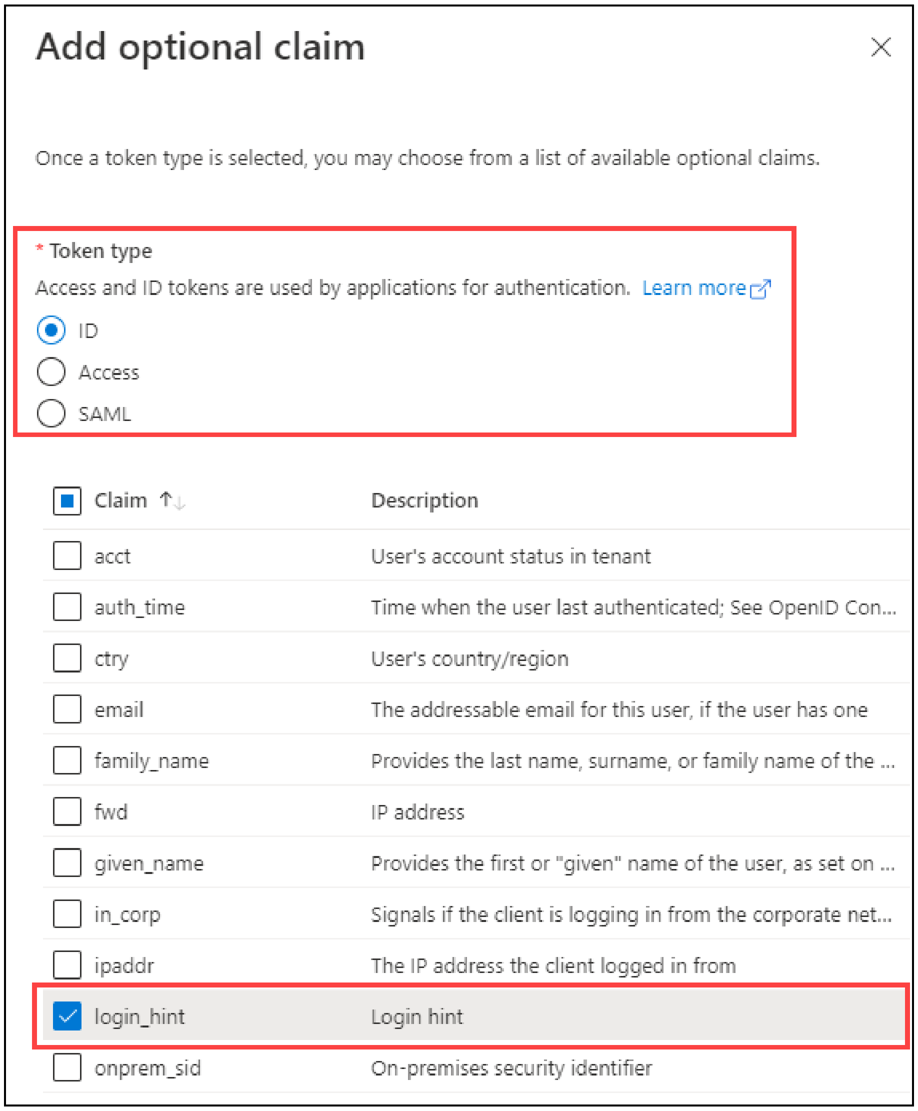
To allow Leap to sign out of specific accounts, you must add an ID token that contains the login_hint claim.

To configure an ID token

1. On the LeapWebApp Overview page, select **Token configuration** from the list of options at the left side of the screen.

   The Token configuration page appears.
2. Select **Add optional claim**.

   The Add optional claim dialog appears.



Add optional claim

Once a token type is selected, you may choose from a list of available optional claims.

* Token type

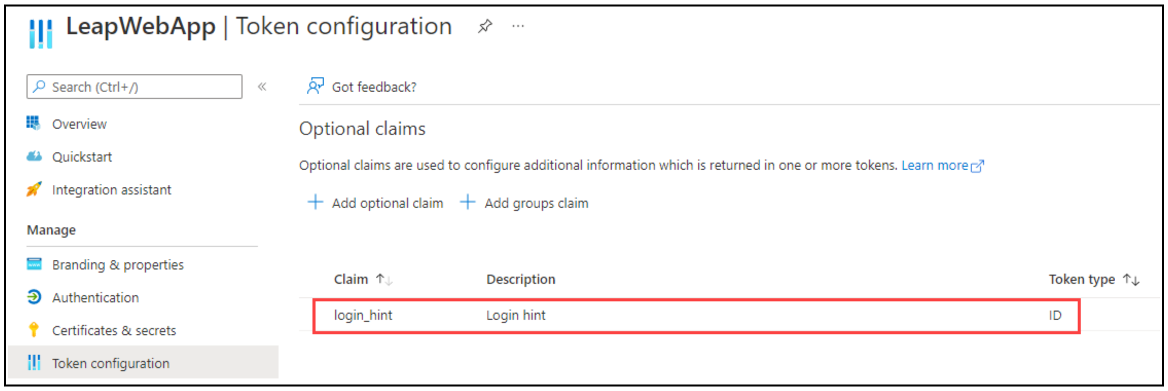
Access and ID tokens are used by applications for authentication. Learn more

[✓ ID] Access SAML

Claim	Description
acct	User's account status in tenant
auth_time	Time when the user last authenticated; See OpenID Con...
ctry	User's country/region
email	The addressable email for this user, if the user has one
family_name	Provides the last name, surname, or family name of the ...
fwd	IP address
given_name	Provides the first or "given" name of the user, as set on ...
in_corp	Signals if the client is logging in from the corporate net...
ipaddr	The IP address the client logged in from
✓ login_hint	Login hint
onprem_sid	On-premises security identifier

3. Set the **Token type** setting to the **ID** option.
4. Select the **login_hint** checkbox.
5. Select **Add**.

   The Azure portal saves the token and closes the Add optional claim dialog.
6. Verify that the new login_hint claim appears on the Token configuration page.



Optional claims

Optional claims are used to configure additional information which is returned in one or more tokens. Learn more

Claim	Description	Token type
login_hint	Login hint	ID

Set Up Users and Groups

To set up users and groups

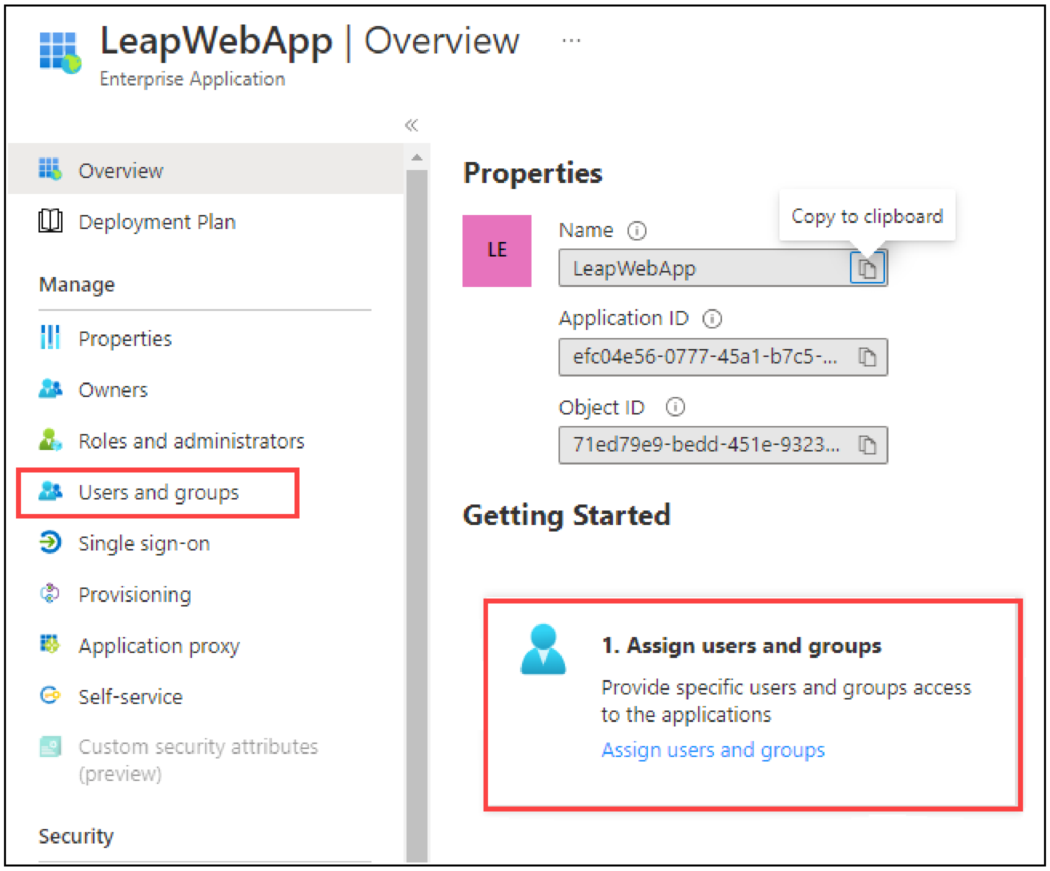
1. On the Azure AD Overview page, select the **Enterprise applications** option from the list at the left side of the screen.

   Note:
   You can also select **Enterprise applications** from the list of services on the Azure portal home page.

   The All applications page appears.

2. Select the **LeapWebApp** link.

   The LeapWebApp Overview page appears.



3. Select the **Users and groups** option from the list at the left side of the screen. You can also select the **Assign users and groups** tile.

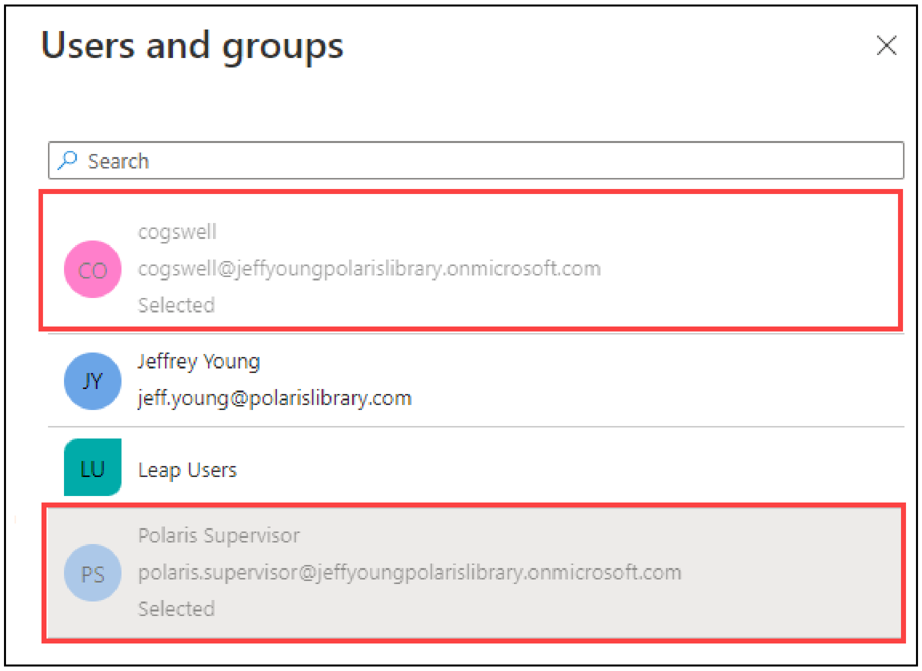
   The Users and groups page appears.

4. Select **Add user/group**.

   The Add Assignment page appears.

5. Select the **None Selected** link.

   The Users and groups dialog appears.



Users and groups

Search

cogswell
cogswell@jeffyoungpolarislibrary.onmicrosoft.com
Selected

Jeffrey Young
jeff.young@polarislibrary.com

Leap Users

Polaris Supervisor
polaris.supervisor@jeffyoungpolarislibrary.onmicrosoft.com
Selected

6. Select the users and groups that you want to allow access to LeapWebApp.
7. Click **Select**.

   The Users and groups dialog closes.
8. On the Add Assignment page, select **Assign**.

   The Azure portal saves the user and group assignments.

Control Access to LeapWebApp Using Azure AD

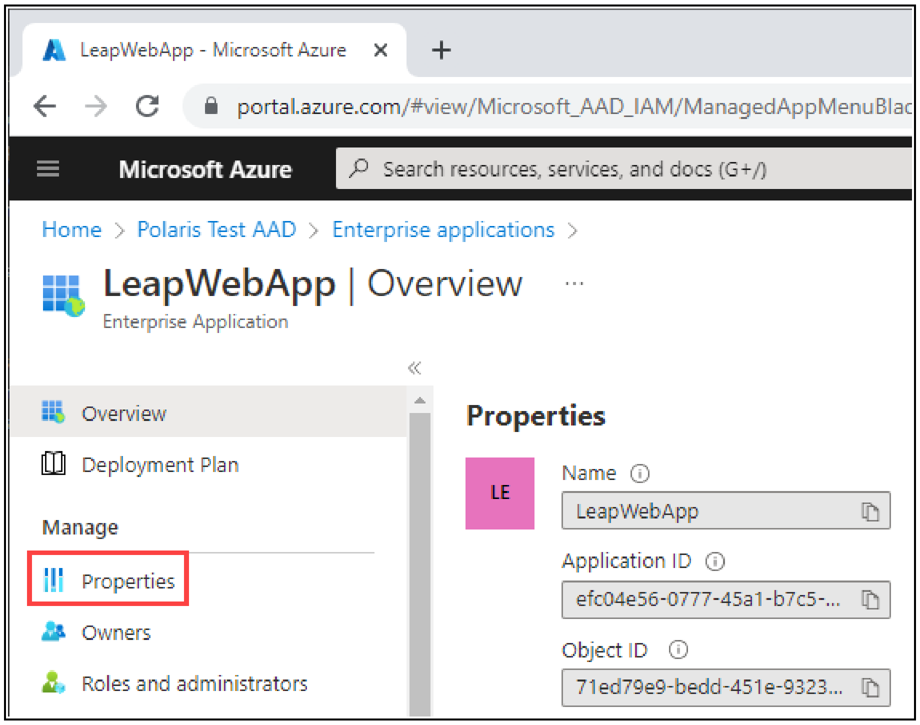
To control access to LeapWebApp using Azure AD

1. On the Azure AD Overview page, select the **Enterprise applications** option from the list at the left side of the screen.

   Note:
   You can also select **Enterprise applications** from the list of services on the Azure portal home page.

   The All applications page appears.
2. Select the **LeapWebApp** link.

The LeapWebApp Overview page appears.


LeapWebApp - Microsoft Azure
portal.azure.com/#view/Microsoft_AAD_IAM/ManagedAppMenuBlac
Microsoft Azure
Search resources, services, and docs (G+/)
Home > Polaris Test AAD > Enterprise applications >
LeapWebApp | Overview
Enterprise Application
Overview
Deployment Plan
Manage
Properties
Owners
Roles and administrators
Properties
LE

Name: LeapWebApp

Application ID: efc04e56-0777-45a1-b7c5-...

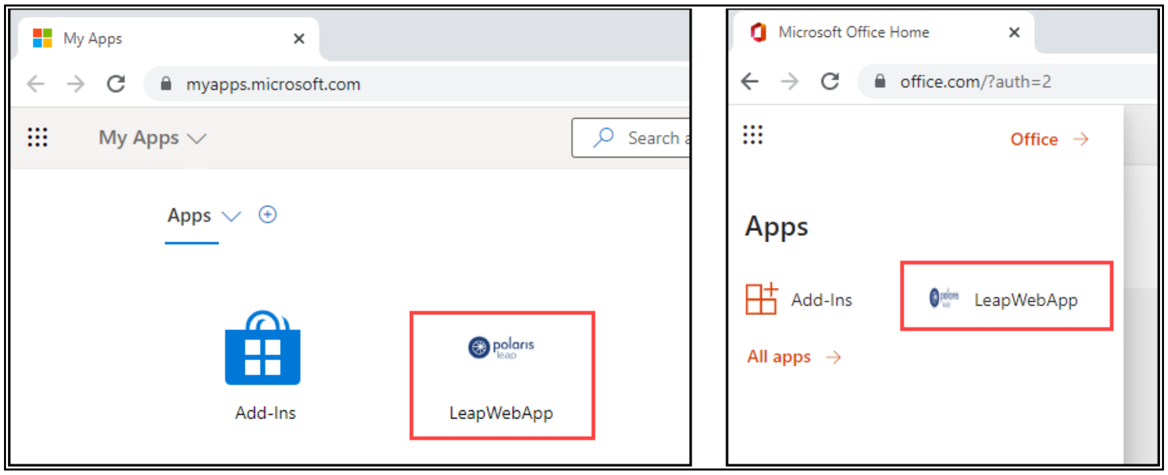
Object ID: 71ed79e9-bedd-451e-9323...

3. Select the **Properties** option from the list at the left side of the screen.

   The Properties page appears.

4. Set the **Assignment requirement?** setting to **Yes**. This allows access to be controlled by the users and groups assigned to the LeapWebApp enterprise application. (When it is set to **No**, all users can sign in.)

5. Set the **Visible to users?** setting to **Yes**. This makes the LeapWebApp application visible to users in their Microsoft My Apps portal and on their Office 365 page.



Important:
When a user accesses LeapWebApp from the Microsoft My Apps portal or their Office 365 page, they might have to click the Polaris Leap Sign In button. This is because cookies are a part of the Leap authentication process.

6. Select **Save**.

   The Azure portal saves your changes.

Set Up Web Services and Applications

To set up each of the following web services and applications, you must configure a .json file for each of the following:

- LeapWebApp (Leap)
- Polaris.ApplicationServices (Leap's API service)
- PolarisAdmin (the web-based Polaris System Administration application)
- Polaris.AdminServices (the API service)

The four .json files are named appsettings.user.json, but they reside in different directories:

- C:\Program Files\Polaris\7.4\LeapWebApp
- C:\Program Files\Polaris\7.4\Polaris.ApplicationServices

- C:\Program Files\Polaris\7.4\PolarisAdmin\assets
- C:\Program Files\Polaris\7.4\Polaris.AdminServices

This section contains the following topics:

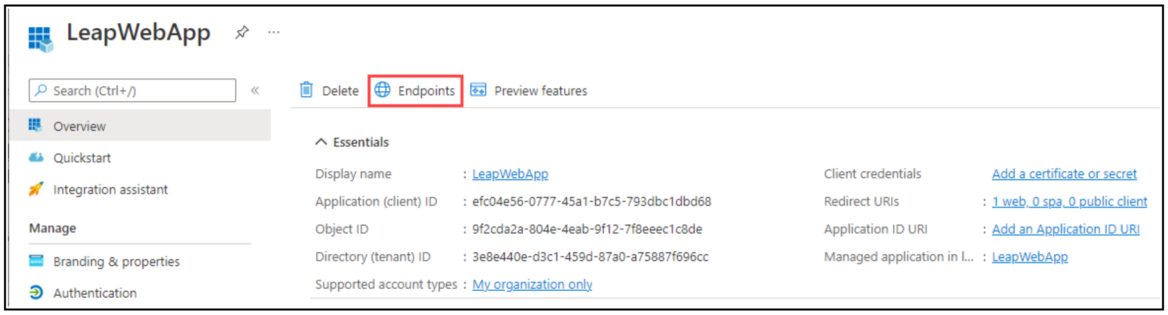
Configure LeapWebApp for Use with Azure AD

To configure LeapWebApp, you will update the C:\Program Files\Polaris\7.4\LeapWebApp\appsettings.user.json file using the following information:

- Endpoint URIs copied from the Azure portal
- Values that you copied and saved during earlier steps in the Azure AD configuration process

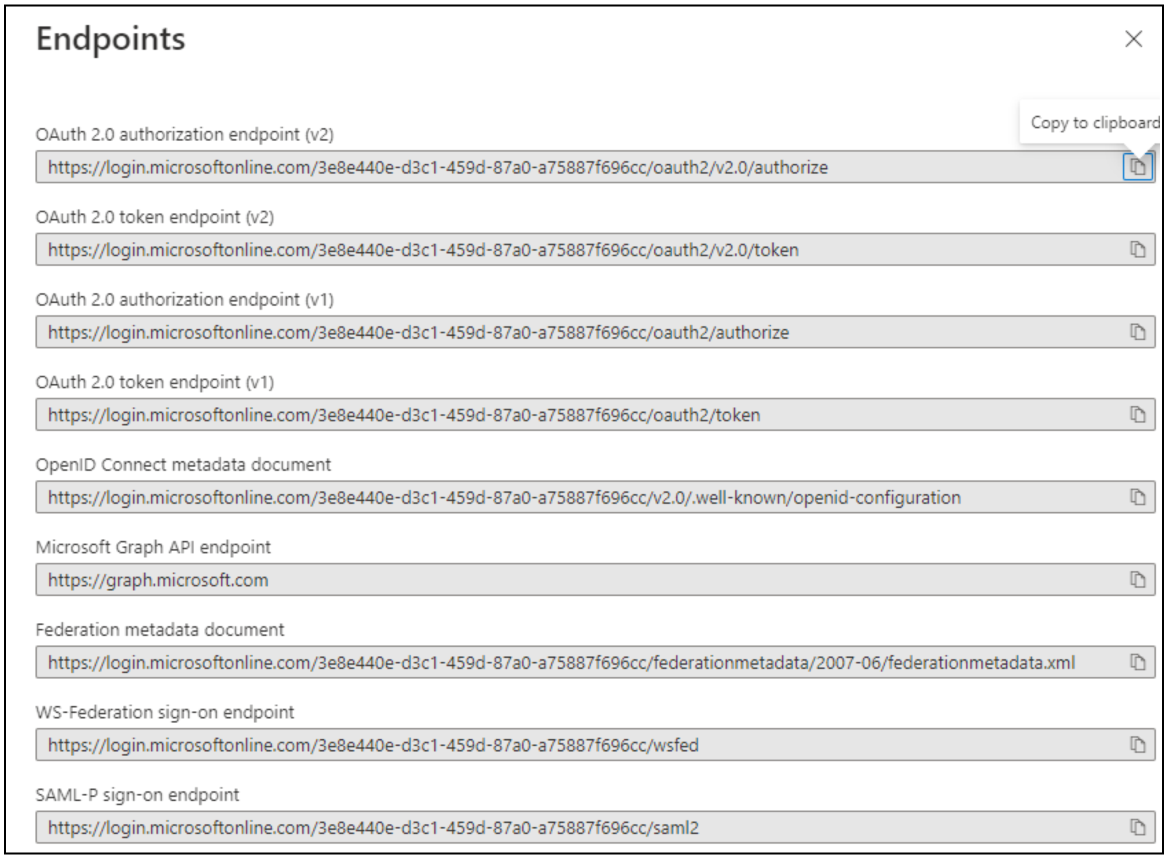
To configure LeapWebApp

1. In the Azure portal, select **App registrations** from the list of options at the left side of the screen.
2. Select **LeapWebApp**.
3. On the LeapWebApp page, select **Endpoints**.



Display name : LeapWebApp
Application (client) ID : efc04e56-0777-45a1-b7c5-793dbc1dbd68
Object ID : 9f2cda2a-804e-4eab-9f12-7f8eeec1c8de
Directory (tenant) ID : 3e8e440e-d3c1-459d-87a0-a75887f696cc
Supported account types : My organization only
Client credentials Add a certificate or secret
Redirect URIs : 1 web, 0 spa, 0 public client
Application ID URI : Add an Application ID URI
Managed application in l... : LeapWebApp

The Endpoints dialog appears. Leave this browser tab open so that you can copy endpoint URIs from it and paste them into the LeapWebApp appsettings.user.json file.



OAuth 2.0 authorization endpoint (v2)
https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/authorize

OAuth 2.0 token endpoint (v2)
https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/token

OAuth 2.0 authorization endpoint (v1)
https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/authorize

OAuth 2.0 token endpoint (v1)
https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/token

OpenID Connect metadata document
https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/v2.0/.well-known/openid-configuration

Microsoft Graph API endpoint
https://graph.microsoft.com

Federation metadata document
https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/federationmetadata/2007-06/federationmetadata.xml

WS-Federation sign-on endpoint
https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/wsfed

SAML-P sign-on endpoint
https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/saml2

4. Open the C:\Program Files\Polaris\7.4\LeapWebApp\appsettings.user.json file.

   Note:
   You must run the editing application (for example, Notepad) as administrator.

5. On the Endpoints dialog, copy the root value and tenant ID from the **OAuth 2.0 authorization endpoint (v2)** box and paste it into the `Authority` property in the .json file. Your value will be similar to this one:

   ```
   https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-
   a75887f696cc/
   ```

   Note:
   The value must include the trailing slash character.

6. In the .json file, replace the `ClientId` property value with the application (client) ID you copied during the Register LeapWebApp with Azure AD step. Your value will be similar to this one:

   ```
   efc04e56-0777-45a1-b7c5-793dbc1dbd68
   ```

7. In the .json file, replace the `ClientSecret` property value with the client secret you copied and saved during the Create Client Credentials step. Your value will be similar to this one:

   ```
   4I.8Q~-GpkdoymMQneGIYNg40FRjx2Hr1wWLDcbr
   ```

8. On the Endpoints dialog, copy the value from the **OpenID Connect metadata document** box and paste it into the `MetadataAddress` property in the .json file. Your value will be similar to this one:

   ```
   https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-
   a75887f696cc/v2.0/.well-known/openid-configuration
   ```

9. On the Endpoints dialog, copy the FQDN of the Microsoft server from the **OpenID Connect metadata document** box and paste it into the `KnownAuthorities` property in the .json file. The value will be identical to this one:

   ```
   login.microsoftonline.com
   ```

10. In the .json file, update the `Scopes` property to add the scope you copied and saved during the Expose the Polaris.ApplicationServices API step. Your value will be similar to this one:

    ```
    api://efc04e56-0777-45a1-b7c5-793dbc1dbd68/pas
    ```

    Note:
    If you want Azure AD to return the user's email address, add `"email"` to the `Scopes` property. You might choose to do this if the `preferred_username` is a generic name, phone number, or other value.

11. In the .json file, set the `AlternateUpnClaimType` property to `"preferred_username"`.

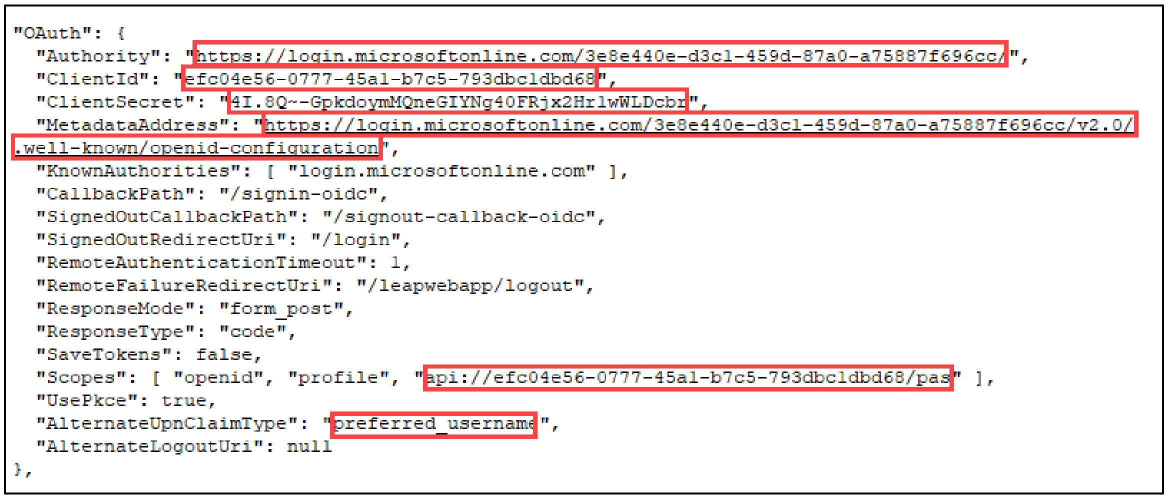
    Note:
    You can also set this property to `"email"`, if you choose.

12. (Optional) Disable permission overrides in Leap. By default, permission overrides are enabled in Leap. If you want to disable overrides:
    - Add the following value to the root level object in the .json file:

      ```
      "client_permissionoverride_enabled": "false",
      ```

      Note:
      If you disable permission overrides in Leap, we recommend that you also update the language string that controls the title of the Permission Override dialog. For more information, search for the SW_CI_PERMISSION_OVRRD_TTL language string in the Polaris Web Admin Tool (Language Editor) Guide.

13. Save the .json file. Your updated file should look similar to the example below.



```
"OAuth": {
  "Authority": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/",
  "ClientId": "efc04e56-0777-45a1-b7c5-793dbc1dbd68",
  "ClientSecret": "4I.8Q~-GpkdoymMQneGIYNg40FRjx2Hr1wWLDcbr",
  "MetadataAddress": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/v2.0/
.well-known/openid-configuration",
  "KnownAuthorities": [ "login.microsoftonline.com" ],
  "CallbackPath": "/signin-oidc",
  "SignedOutCallbackPath": "/signout-callback-oidc",
  "SignedOutRedirectUri": "/login",
  "RemoteAuthenticationTimeout": 1,
  "RemoteFailureRedirectUri": "/leapwebapp/logout",
  "ResponseMode": "form_post",
  "ResponseType": "code",
  "SaveTokens": false,
  "Scopes": [ "openid", "profile", "api://efc04e56-0777-45a1-b7c5-793dbc1dbd68/pas" ],
  "UsePkce": true,
  "AlternateUpnClaimType": "preferred_username",
  "AlternateLogoutUri": null
},
```

14. Leave the browser tab displaying the Endpoints dialog open, and continue to the Configure Polaris.ApplicationServices for Use with Azure AD procedure.

Configure Polaris.ApplicationServices for Use with Azure AD

To configure Polaris.ApplicationServices, you will update the C:\Program Files\Polaris\7.4\Polaris.ApplicationServices\appsettings.user.json file using the following information:

- Endpoint URIs copied from the Azure portal
- Values that you copied and saved during earlier steps in the Azure AD configuration process

To configure Polaris.ApplicationServices

1. Open the C:\Program Files\Polaris\7.4\Polaris.ApplicationServices\appsettings.user.json file.

   > **Note:**
   > You must run the editing application (for example, Notepad) as administrator.

2. In the `Authorities` array, set the `Name` property to `"AzureAD"`.
3. In the .json file, update the `Authority` property:
   a. On the Endpoints dialog of the Azure AD portal, copy the value from the **OAuth 2.0 authorization endpoint (v2)** box but omit the trailing *authorize*.
   b. Paste this value into the `Authority` property in the .json file. Your value will be similar to this one:

      ```
      https://login.microsoftonline.com/3e8e440e-d3c1-459d-
      87a0-a75887f696cc/oauth2/v2.0/
      ```

4. In the .json file, update the `Audience` property:
   a. Locate the application (client) ID you copied and saved during the Register LeapWebApp with Azure AD step.
   b. Use it to construct a string with the following format:

      api://*[application (client) ID]*

   c. Paste this value into the `Audience` property. Your value will be similar to this one:

      ```
      api://efc04e56-0777-45a1-b7c5-793dbc1dbd68
      ```

5. On the Endpoints dialog, copy the value from the **OpenID Connect metadata document** box and paste it into the `MetaAddress` property in the .json file. Your value will be similar to this one:

   ```
   https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-
   a75887f696cc/oauth2/v2.0/well-known/openid-configuration
   ```

6. In the .json file, add a value to the `ValidIssuers` property:

   a. On the Endpoints dialog, copy the tenant ID from the **OAuth 2.0 authorization endpoint (v2)** box.
   b. Use it to construct a URI with the following format:

      https://sts.windows.net/*[tenant ID]*
   c. Paste this value into the `ValidIssuers` property. Your value will be similar to this one:

      ```
      https://sts.windows.net/3e8e440e-d3c1-459d-87a0-
      a75887f696cc/
      ```
7. In the .json file, add a second value to the `ValidIssuers` property:
   a. On the Endpoints dialog, copy the value from the **OAuth 2.0 authorization endpoint (v2)** box but omit the trailing *authorize*.
   b. Paste this value into the `ValidIssuers` property in the .json file. Your value will be similar to this one:

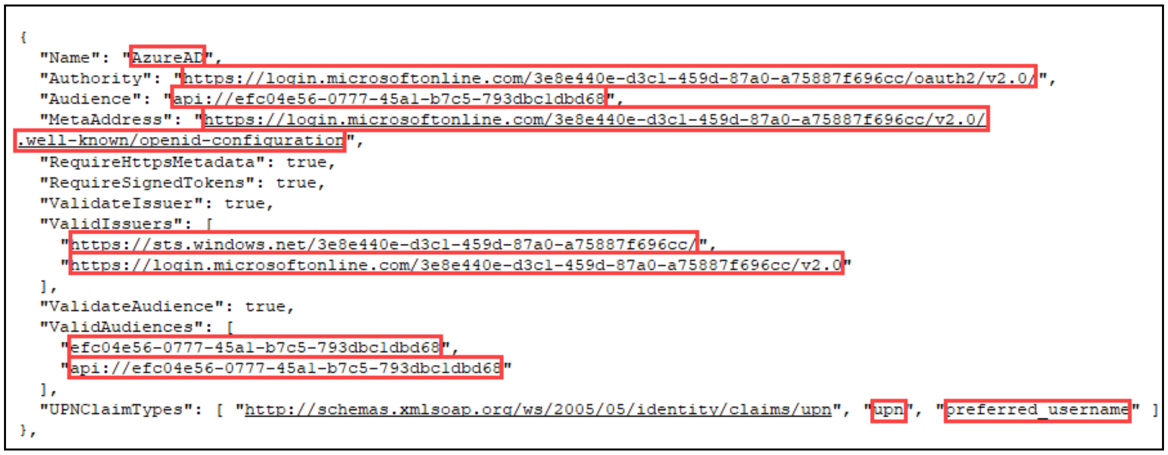
      ```
      https://login.microsoftonline.com/3e8e440e-d3c1-459d-
      87a0-a75887f696cc/oauth2/v2.0
      ```
8. In the .json file, add a value to the `ValidAudiences` property:
   a. Locate the application (client) ID you copied and saved during the Register LeapWebApp with Azure AD step.
   b. Paste this value into the `ValidAudiences` property. Your value will be similar to this one:

      ```
      efc04e56-0777-45a1-b7c5-793dbc1dbd68
      ```
9. In the .json file, add a second value to the `ValidAudiences` property:
   a. Locate the application (client) ID you copied and saved during the Register LeapWebApp with Azure AD step.
   b. Use it to construct a string with the following format:

      api://*[application (client) ID]*
   c. Paste this value into the `ValidAudiences` property. Your value will be similar to this one:

      ```
      api://efc04e56-0777-45a1-b7c5-793dbc1dbd68
      ```

10. In the .json file, update the `UPNClaimTypes` property to add `"upn"` and `"preferred_username"` if those values are not already present. Your updated values in the `Authorities` array should look similar to the example below.



```
{
  "Name": "AzureAD",
  "Authority": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/",
  "Audience": "api://efc04e56-0777-45a1-b7c5-793dbc1dbd68",
  "MetaAddress": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/v2.0/
.well-known/openid-configuration",
  "RequireHttpsMetadata": true,
  "RequireSignedTokens": true,
  "ValidateIssuer": true,
  "ValidIssuers": [
    "https://sts.windows.net/3e8e440e-d3c1-459d-87a0-a75887f696cc/",
    "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/v2.0"
  ],
  "ValidateAudience": true,
  "ValidAudiences": [
    "efc04e56-0777-45a1-b7c5-793dbc1dbd68",
    "api://efc04e56-0777-45a1-b7c5-793dbc1dbd68"
  ],
  "UPNClaimTypes": [ "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn", "upn", "preferred_username" ]
},
```

11. In the `Swagger` property in the .json file, update the `ClientID` property:

    a. Locate the application (client) ID you copied and saved during the Register LeapWebApp with Azure AD step.

    b. Paste this value into the `ClientID` property. Your value will be similar to this one:

    `efc04e56-0777-45a1-b7c5-793dbc1dbd68`

12. On the Endpoints dialog, copy the value from the **OAuth 2.0 authorization endpoint (v2)** box and paste it into the `AuthorizationUrl` property in the .json file. Your value will be similar to this one:

    `https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/authorize`

13. In the .json file, update the `TokenUrl` property:

    a. On the Endpoints dialog, copy the value from the **OAuth 2.0 token endpoint (v2)** box but replace *authorize* with *token*.

    b. Paste the value into the `TokenUrl` property in the .json file. Your value will be similar to this one:

```
https://login.microsoftonline.com/3e8e440e-d3c1-459d-
87a0-a75887f696cc/oauth2/v2.0/token
```

14. In the .json file, update the `RefreshTokenUrl` property:
    a. On the Endpoints dialog, copy the value from the **OAuth 2.0 token endpoint (v2)** box but replace *authorize* with *token*.
    b. Paste the value into the `RefreshTokenUrl` property in the .json file. Your value will be similar to this one:

    ```
    https://login.microsoftonline.com/3e8e440e-d3c1-459d-
    87a0-a75887f696cc/oauth2/v2.0/token
    ```

15. In the .json file, update the `LogoutUrl` property:
    a. On the Endpoints dialog, copy the value from the **OAuth 2.0 authorization endpoint (v2)** box but replace *authorize* with *logout*.
    b. Paste the value into the `LogoutUrl` property in the .json file. Your value will be similar to this one:

    ```
    https://login.microsoftonline.com/3e8e440e-d3c1-459d-
    87a0-a75887f696cc/oauth2/v2.0/logout
    ```

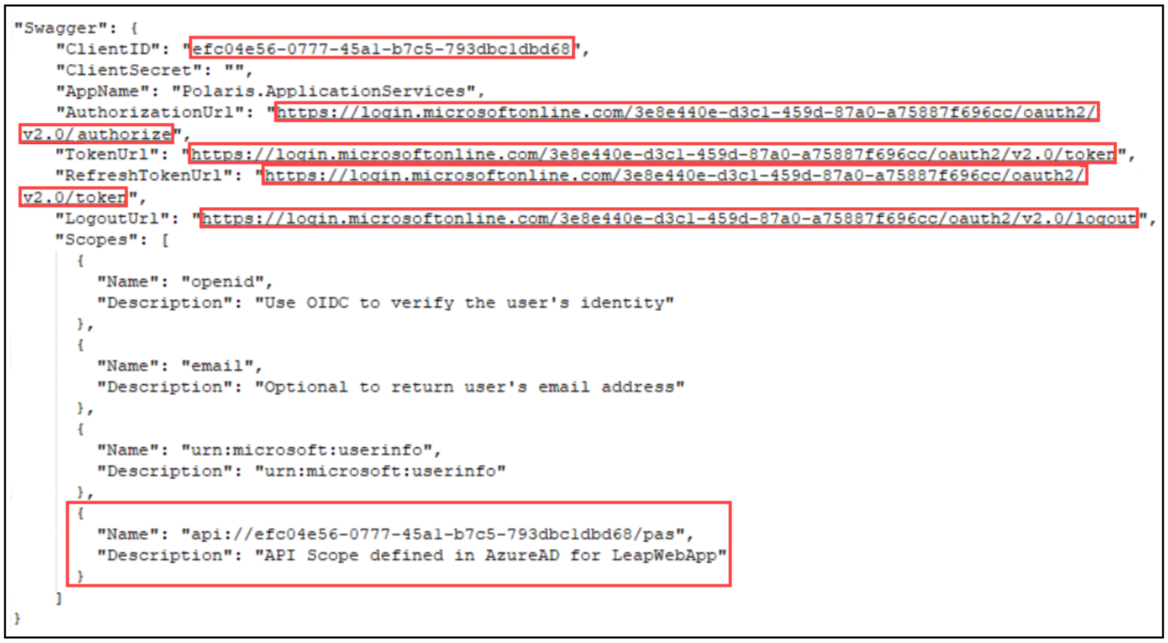
16. In the .json file, add a new scope `Name` and `Description`:
    a. Locate the scope you copied and saved during the Expose the Polaris.ApplicationServices API step.
    b. Copy the scope and use it to construct a new `Name` property. Your value will be similar to this one:

    ```
    "Name": "api://efc04e56-0777-45a1-b7c5-793dbc1dbd68/pas"
    ```

    c. Paste the `Name` property into the .json file.
    d. Add a `Description` property that matches the example below:

    ```
    "Description": "API Scope defined in AzureAD for
    LeapWebApp"
    ```

17. Save the .json file. Your updated values in the `Swagger` property should look similar to the example below.



```
"Swagger": {
    "ClientID": "efc04e56-0777-45a1-b7c5-793dbc1dbd68",
    "ClientSecret": "",
    "AppName": "Polaris.ApplicationServices",
    "AuthorizationUrl": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/
v2.0/authorize",
    "TokenUrl": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/token",
    "RefreshTokenUrl": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/
v2.0/token",
    "LogoutUrl": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/logout",
    "Scopes": [
      {
        "Name": "openid",
        "Description": "Use OIDC to verify the user's identity"
      },
      {
        "Name": "email",
        "Description": "Optional to return user's email address"
      },
      {
        "Name": "urn:microsoft:userinfo",
        "Description": "urn:microsoft:userinfo"
      },
      {
        "Name": "api://efc04e56-0777-45a1-b7c5-793dbc1dbd68/pas",
        "Description": "API Scope defined in AzureAD for LeapWebApp"
      }
    ]
}
```

Note:
Changes to the appsettings.user.json files do note take effect until the IIS application pools are restarted or IIS is reset.

Configure PolarisAdmin for Use with Azure AD

To configure PolarisAdmin, you will update the C:\Program Files\Polaris\7.4\PolarisAdmin\assets\appsettings.user.json file. You will use several values copied from the Azure AD portal.

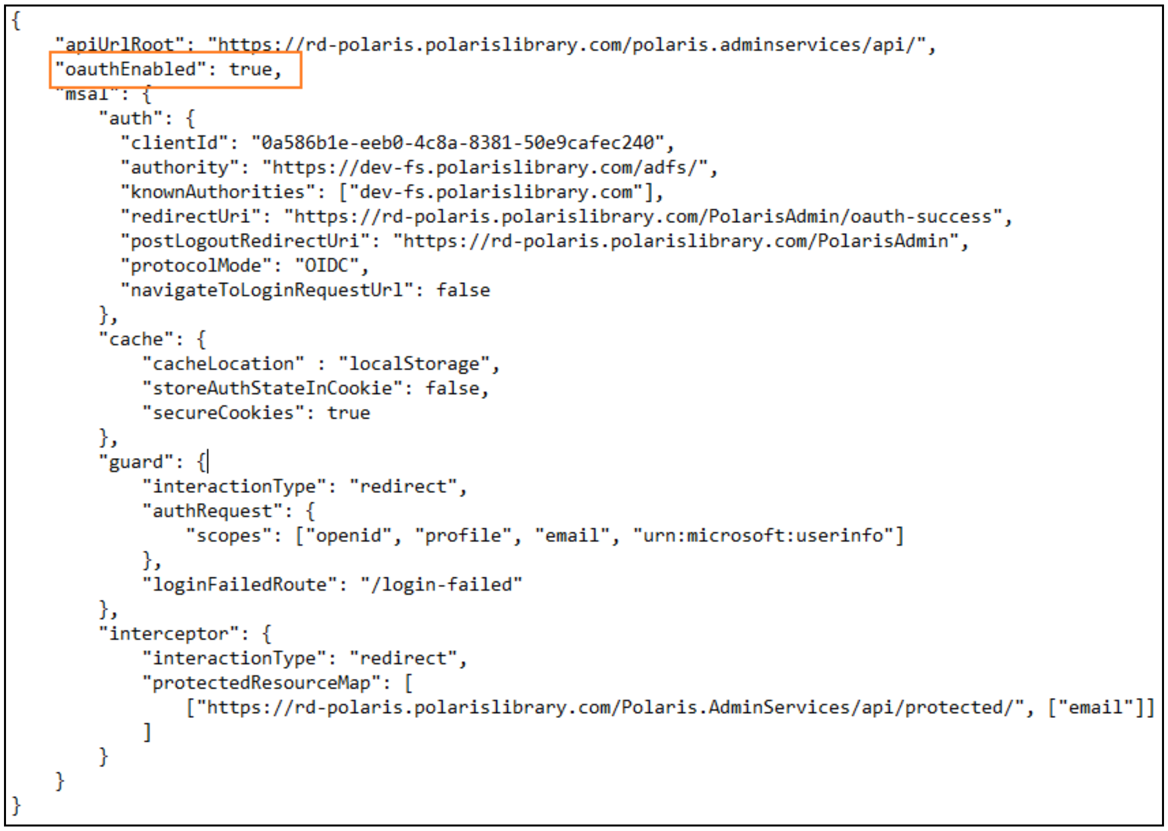
Important:
By default, the appsettings.user.json template file contains configuration settings that apply to AD FS. Polaris 7.4 includes a RELEASE-NOTES.md file that contains the template settings that apply to Azure AD.

To configure PolarisAdmin

1. Open the C:\Program Files\Polaris\7.4\PolarisAdmin\assets\appsettings.user.json file.

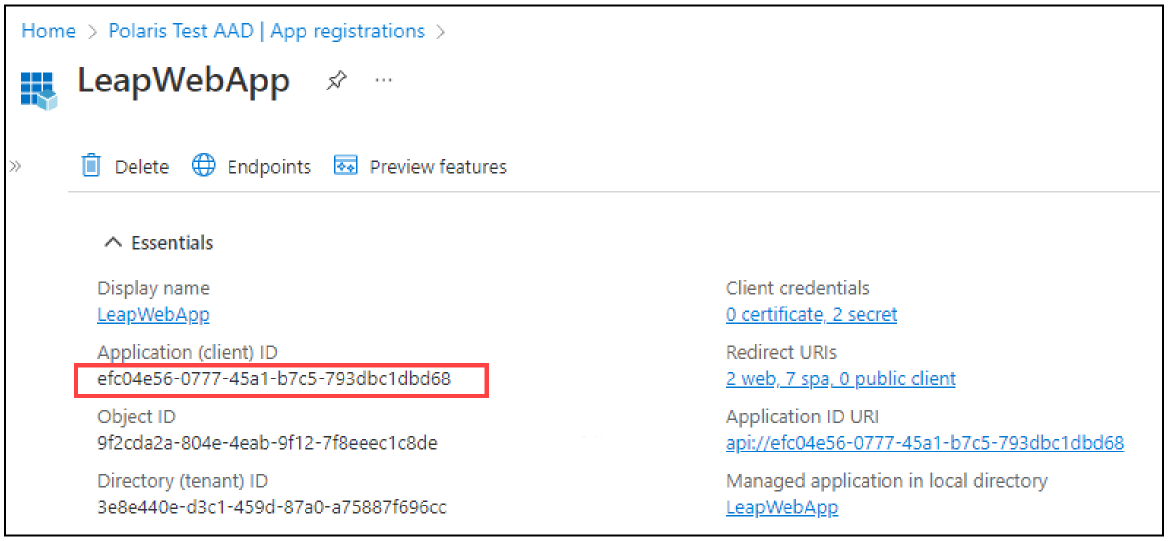
> **Note:**
> You must run the editing application (for example, Notepad) as administrator.

2. Verify that `oauthEnabled` is set to `true`.



```
{
    "apiUrlRoot": "https://rd-polaris.polarislibrary.com/polaris.adminservices/api/",
    "oauthEnabled": true,
    "msal": {
        "auth": {
          "clientId": "0a586b1e-eeb0-4c8a-8381-50e9cafec240",
          "authority": "https://dev-fs.polarislibrary.com/adfs/",
          "knownAuthorities": ["dev-fs.polarislibrary.com"],
          "redirectUri": "https://rd-polaris.polarislibrary.com/PolarisAdmin/oauth-success",
          "postLogoutRedirectUri": "https://rd-polaris.polarislibrary.com/PolarisAdmin",
          "protocolMode": "OIDC",
          "navigateToLoginRequestUrl": false
        },
        "cache": {
            "cacheLocation" : "localStorage",
            "storeAuthStateInCookie": false,
            "secureCookies": true
        },
        "guard": {
            "interactionType": "redirect",
            "authRequest": {
                "scopes": ["openid", "profile", "email", "urn:microsoft:userinfo"]
            },
            "loginFailedRoute": "/login-failed"
        },
        "interceptor": {
            "interactionType": "redirect",
            "protectedResourceMap": [
                ["https://rd-polaris.polarislibrary.com/Polaris.AdminServices/api/protected/", ["email"]]
            ]
        }
    }
}
```

3. In the Azure portal, copy the Application (client) ID.



Display name
LeapWebApp

Application (client) ID
efc04e56-0777-45a1-b7c5-793dbc1dbd68

Object ID
9f2cda2a-804e-4eab-9f12-7f8eeec1c8de

Directory (tenant) ID
3e8e440e-d3c1-459d-87a0-a75887f696cc

Client credentials
0 certificate, 2 secret

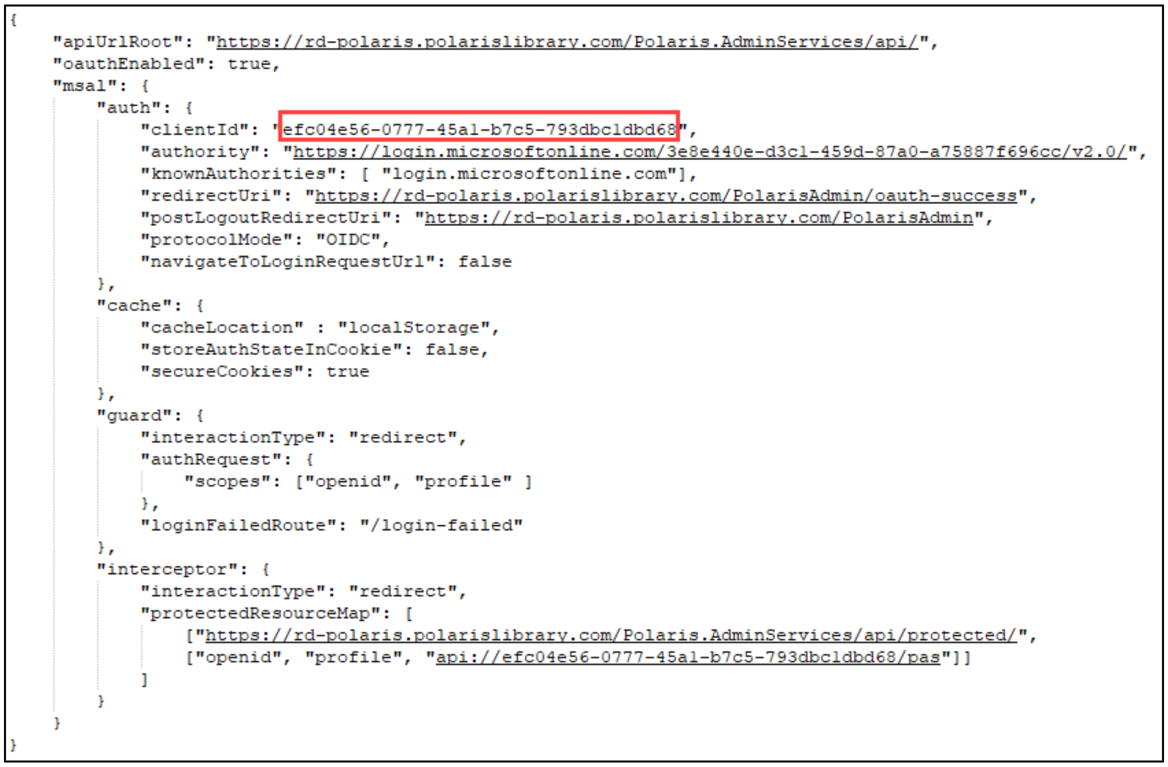
Redirect URIs
2 web, 7 spa, 0 public client

Application ID URI
api://efc04e56-0777-45a1-b7c5-793dbc1dbd68

Managed application in local directory
LeapWebApp

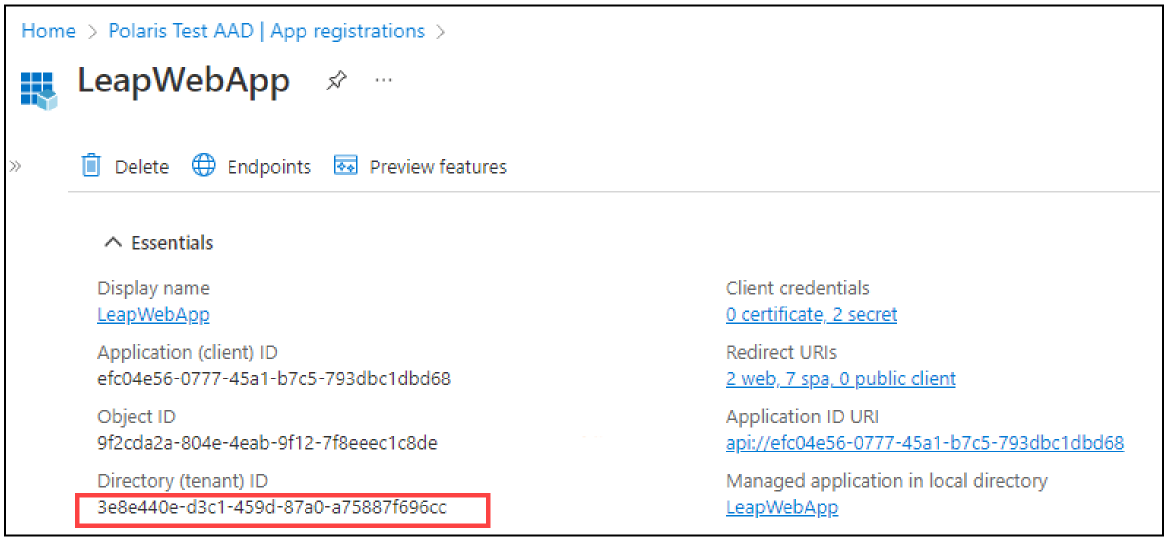
4. Paste the copied client ID into the appsettings.user.json file. If you started from the template settings provided in the RELEASE-NOTES.md file, replace *[CLIENTID-ASSIGNED-IN-AZUREAD]* with the copied client ID.

   When complete, your file should look like the following example (your client ID will be different):



```
{
    "apiUrlRoot": "https://rd-polaris.polarislibrary.com/Polaris.AdminServices/api/",
    "oauthEnabled": true,
    "msal": {
        "auth": {
            "clientId": "efc04e56-0777-45a1-b7c5-793dbc1dbd68",
            "authority": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/v2.0/",
            "knownAuthorities": [ "login.microsoftonline.com"],
            "redirectUri": "https://rd-polaris.polarislibrary.com/PolarisAdmin/oauth-success",
            "postLogoutRedirectUri": "https://rd-polaris.polarislibrary.com/PolarisAdmin",
            "protocolMode": "OIDC",
            "navigateToLoginRequestUrl": false
        },
        "cache": {
            "cacheLocation" : "localStorage",
            "storeAuthStateInCookie": false,
            "secureCookies": true
        },
        "guard": {
            "interactionType": "redirect",
            "authRequest": {
                "scopes": ["openid", "profile" ]
            },
            "loginFailedRoute": "/login-failed"
        },
        "interceptor": {
            "interactionType": "redirect",
            "protectedResourceMap": [
                ["https://rd-polaris.polarislibrary.com/Polaris.AdminServices/api/protected/",
                ["openid", "profile", "api://efc04e56-0777-45a1-b7c5-793dbc1dbd68/pas"]]
            ]
        }
    }
}
```

5. In the .json file, update the `authority` to use `login.microsoftonline.com`.
6. Update the `knownAuthorities` to use `login.microsoftonline.com`.
7. In the Azure portal, copy the Directory (tenant) ID.



Home > Polaris Test AAD | App registrations >

LeapWebApp

Delete Endpoints Preview features

Essentials

Display name
LeapWebApp

Application (client) ID
efc04e56-0777-45a1-b7c5-793dbc1dbd68

Object ID
9f2cda2a-804e-4eab-9f12-7f8eeec1c8de

Directory (tenant) ID
3e8e440e-d3c1-459d-87a0-a75887f696cc

Client credentials
0 certificate, 2 secret

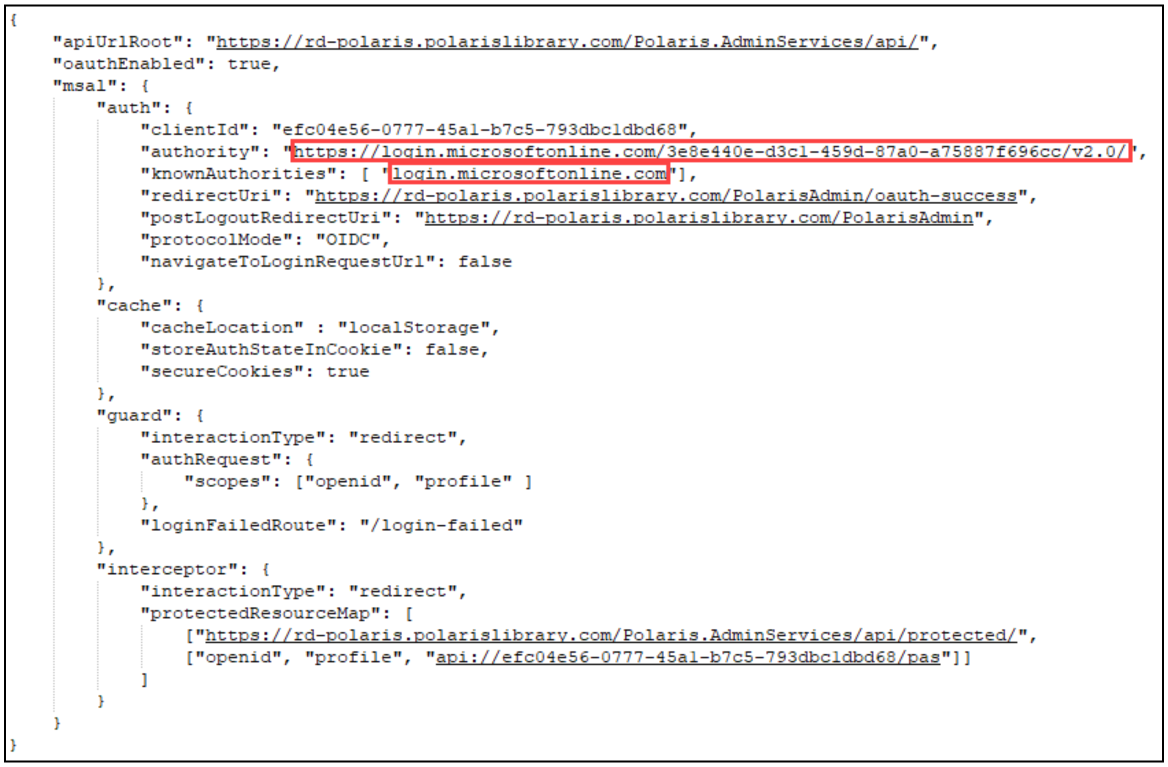
Redirect URIs
2 web, 7 spa, 0 public client

Application ID URI
api://efc04e56-0777-45a1-b7c5-793dbc1dbd68

Managed application in local directory
LeapWebApp

8. In the .json file, paste the copied tenant ID into the `authority`. If you started from the template settings provided in the RELEASE-NOTES.md file, replace *[TENANTID-ASSIGNED-IN-AZUREAD]* with the copied tenant ID.

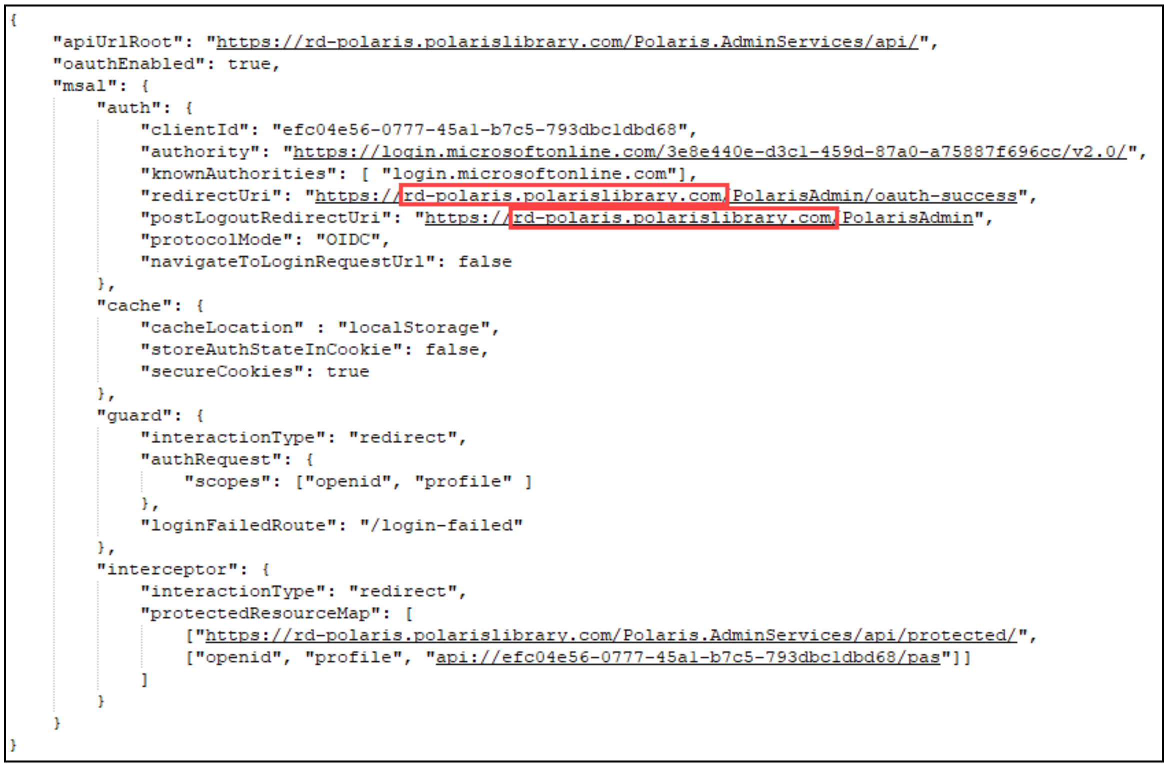
   When complete, your file should look like the following example (your tenant ID will be different):



```
{
    "apiUrlRoot": "https://rd-polaris.polarislibrary.com/Polaris.AdminServices/api/",
    "oauthEnabled": true,
    "msal": {
        "auth": {
            "clientId": "efc04e56-0777-45a1-b7c5-793dbc1dbd68",
            "authority": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/v2.0/",
            "knownAuthorities": [ "login.microsoftonline.com"],
            "redirectUri": "https://rd-polaris.polarislibrary.com/PolarisAdmin/oauth-success",
            "postLogoutRedirectUri": "https://rd-polaris.polarislibrary.com/PolarisAdmin",
            "protocolMode": "OIDC",
            "navigateToLoginRequestUrl": false
        },
        "cache": {
            "cacheLocation" : "localStorage",
            "storeAuthStateInCookie": false,
            "secureCookies": true
        },
        "guard": {
            "interactionType": "redirect",
            "authRequest": {
                "scopes": ["openid", "profile" ]
            },
            "loginFailedRoute": "/login-failed"
        },
        "interceptor": {
            "interactionType": "redirect",
            "protectedResourceMap": [
                ["https://rd-polaris.polarislibrary.com/Polaris.AdminServices/api/protected/",
                ["openid", "profile", "api://efc04e56-0777-45a1-b7c5-793dbc1dbd68/pas"]]
            ]
        }
    }
}
```

9. In the .json file, update the server location in the `redirectUri` and the `postLogoutRedirectUri`. If you started from the template settings provided in the RELEASE-NOTES.md file, replace *[POLADMIN-SERVER-ADDR]* with the FQDN of the server that hosts the Polaris System Administration (web-based) user interface.

   When complete, your file should look like the following example (your server address will be different):

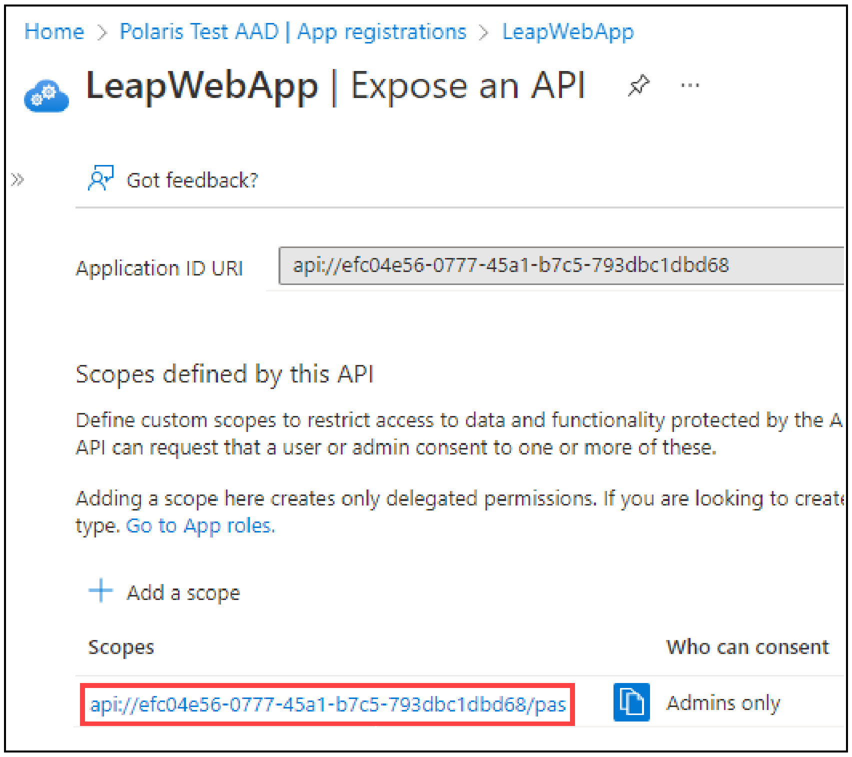


```
{
    "apiUrlRoot": "https://rd-polaris.polarislibrary.com/Polaris.AdminServices/api/",
    "oauthEnabled": true,
    "msal": {
        "auth": {
            "clientId": "efc04e56-0777-45a1-b7c5-793dbc1dbd68",
            "authority": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/v2.0/",
            "knownAuthorities": [ "login.microsoftonline.com"],
            "redirectUri": "https://rd-polaris.polarislibrary.com/PolarisAdmin/oauth-success",
            "postLogoutRedirectUri": "https://rd-polaris.polarislibrary.com/PolarisAdmin",
            "protocolMode": "OIDC",
            "navigateToLoginRequestUrl": false
        },
        "cache": {
            "cacheLocation" : "localStorage",
            "storeAuthStateInCookie": false,
            "secureCookies": true
        },
        "guard": {
            "interactionType": "redirect",
            "authRequest": {
                "scopes": ["openid", "profile" ]
            },
            "loginFailedRoute": "/login-failed"
        },
        "interceptor": {
            "interactionType": "redirect",
            "protectedResourceMap": [
                ["https://rd-polaris.polarislibrary.com/Polaris.AdminServices/api/protected/",
                ["openid", "profile", "api://efc04e56-0777-45a1-b7c5-793dbc1dbd68/pas"]]
            ]
        }
    }
}
```

10. In the .json file, update the server location in the `protectedResourceMap`. If you started from the template settings provided in the RELEASE-NOTES.md file, replace *[POLADMINSVC-SERVER-ADDR]* with the FQDN of the server that hosts the API service for Polaris System Administration (web-based).

    When complete, your file should look like the example shown in step 13 (your server address will be different).

11. In the Azure portal, LeapWebApp section of the App registrations dashboard and copy the API scope.



Home > Polaris Test AAD | App registrations > LeapWebApp

LeapWebApp | Expose an API

Got feedback?

Application ID URI api://efc04e56-0777-45a1-b7c5-793dbc1dbd68

Scopes defined by this API

Define custom scopes to restrict access to data and functionality protected by the A
API can request that a user or admin consent to one or more of these.

Adding a scope here creates only delegated permissions. If you are looking to create
type. Go to App roles.

Add a scope

Scopes	Who can consent
api://efc04e56-0777-45a1-b7c5-793dbc1dbd68/pas	Admins only

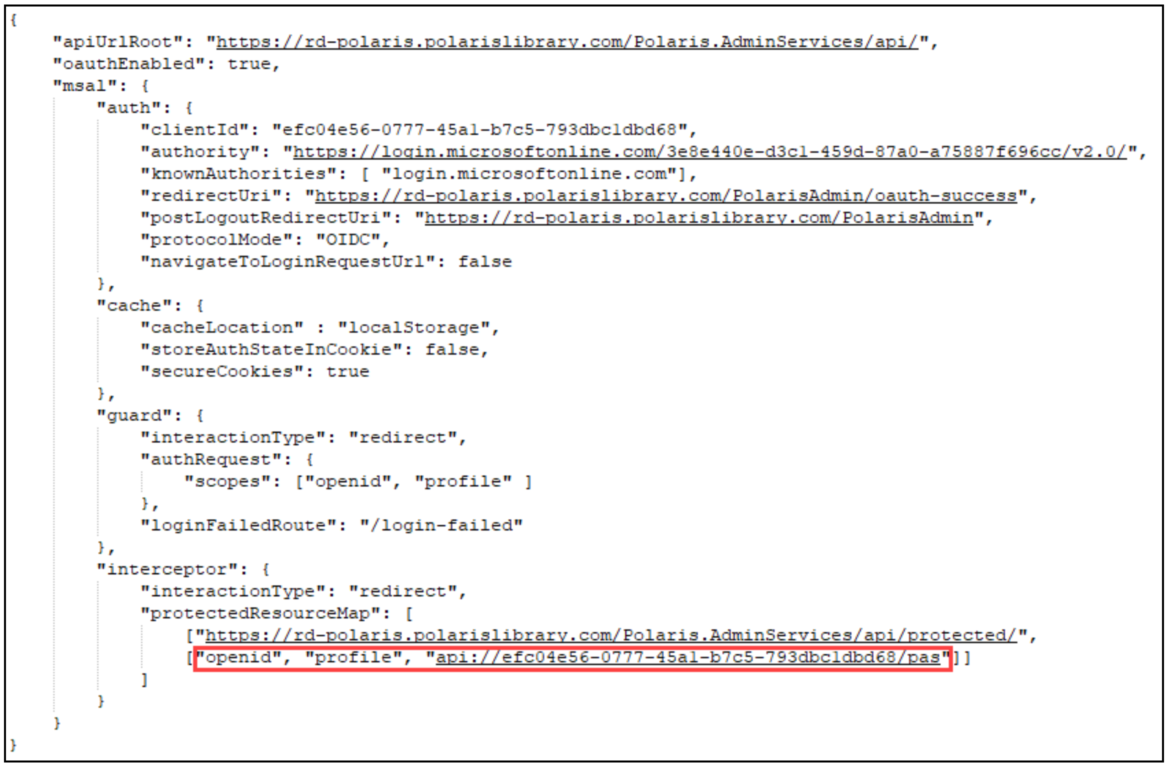
12. In the .json file, paste the copied scope into the `protectedResourceMap` array.

    When complete, your file should look like the example shown in step 13.

13. In the `protectedResourceMap` array, verify that:

    - The `openid` and `profile` scopes are listed.
    - The `email` scope is not listed.

    When complete, your file should look like the following example:



```
{
    "apiUrlRoot": "https://rd-polaris.polarislibrary.com/Polaris.AdminServices/api/",
    "oauthEnabled": true,
    "msal": {
        "auth": {
            "clientId": "efc04e56-0777-45a1-b7c5-793dbc1dbd68",
            "authority": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/v2.0/",
            "knownAuthorities": [ "login.microsoftonline.com"],
            "redirectUri": "https://rd-polaris.polarislibrary.com/PolarisAdmin/oauth-success",
            "postLogoutRedirectUri": "https://rd-polaris.polarislibrary.com/PolarisAdmin",
            "protocolMode": "OIDC",
            "navigateToLoginRequestUrl": false
        },
        "cache": {
            "cacheLocation" : "localStorage",
            "storeAuthStateInCookie": false,
            "secureCookies": true
        },
        "guard": {
            "interactionType": "redirect",
            "authRequest": {
                "scopes": ["openid", "profile" ]
            },
            "loginFailedRoute": "/login-failed"
        },
        "interceptor": {
            "interactionType": "redirect",
            "protectedResourceMap": [
                ["https://rd-polaris.polarislibrary.com/Polaris.AdminServices/api/protected/",
                ["openid", "profile", "api://efc04e56-0777-45a1-b7c5-793dbc1dbd68/pas"]]
            ]
        }
    }
}
```

14. Save the .json file.

Configure Polaris.AdminServices for Use with Azure AD

To configure Polaris.AdminServices, you will update the C:\Program Files\Polaris\7.4\Polaris.ApplicationServices\appsettings.user.json file. You will use several values copied from the Azure AD portal.

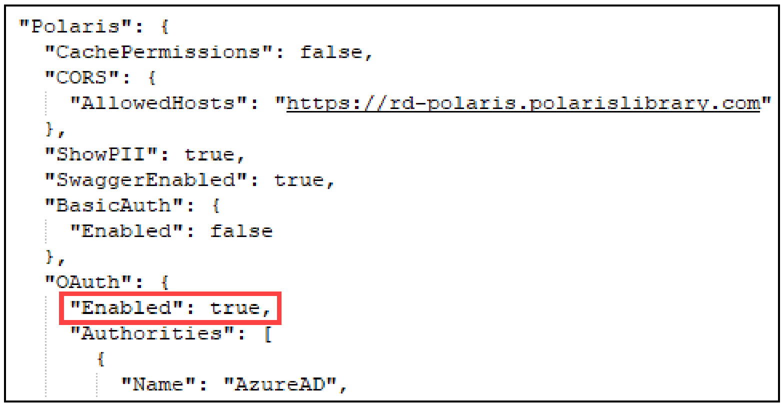
> **Important:**
> By default, the appsettings.user.json template file contains configuration settings that apply to AD FS. Polaris 7.4 includes a RELEASE-NOTES.md file that contains the template settings that apply to Azure AD.

To configure Polaris.AdminServices

1. Open the C:\Program Files\Polaris\7.4\Polaris.AdminServices\appsettings.user.json file.

Note:
You must run the editing application (for example, Notepad) as administrator.

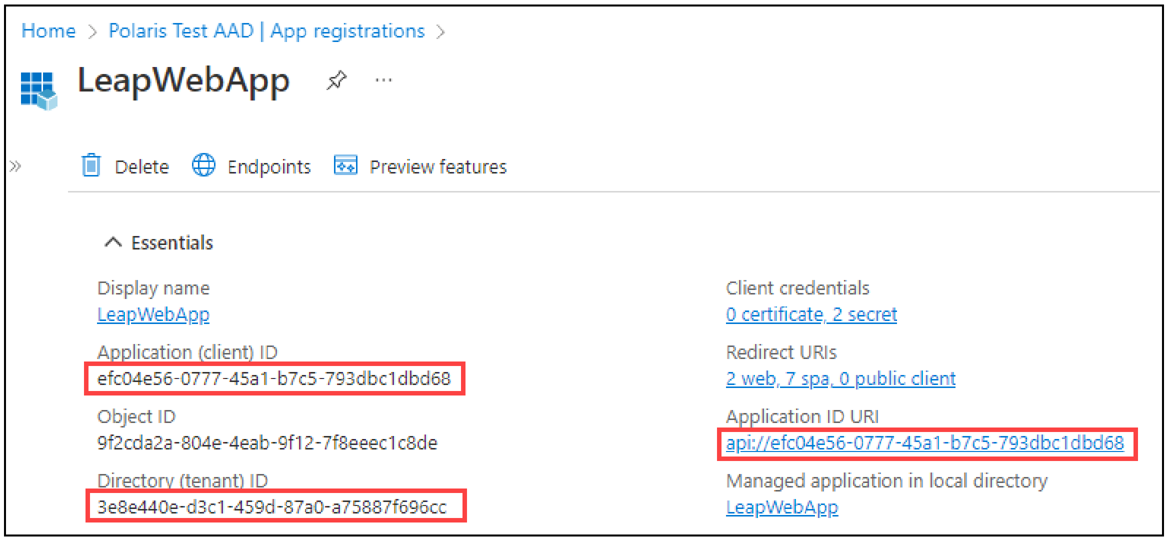
2. Verify that `Enabled` is set to `true` under the `OAuth` object.



```
"Polaris": {
  "CachePermissions": false,
  "CORS": {
    "AllowedHosts": "https://rd-polaris.polarislibrary.com"
  },
  "ShowPII": true,
  "SwaggerEnabled": true,
  "BasicAuth": {
    "Enabled": false
  },
  "OAuth": {
    "Enabled": true,
    "Authorities": [
      {
        "Name": "AzureAD",
```

3. In the Azure portal, go to the App registrations dashboard and retrieve the following values:
   - Application (client) ID
   - Directory (tenant) ID
   - Application ID URI

   You will use these values in the steps below.



Display name
LeapWebApp

Application (client) ID
efc04e56-0777-45a1-b7c5-793dbc1dbd68

Object ID
9f2cda2a-804e-4eab-9f12-7f8eeec1c8de

Directory (tenant) ID
3e8e440e-d3c1-459d-87a0-a75887f696cc

Client credentials
0 certificate, 2 secret

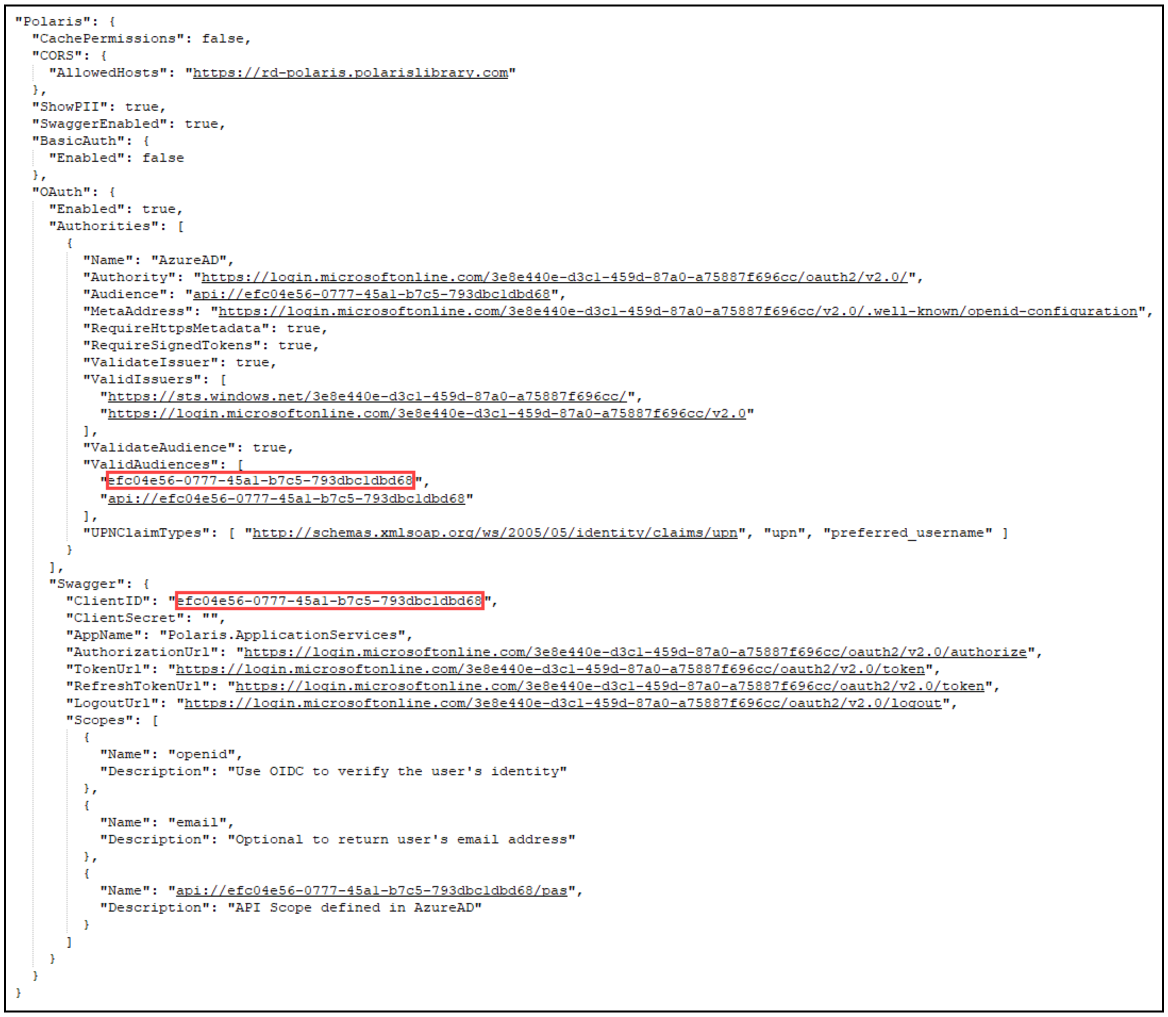
Redirect URIs
2 web, 7 spa, 0 public client

Application ID URI
api://efc04e56-0777-45a1-b7c5-793dbc1dbd68

Managed application in local directory
LeapWebApp

4. Paste the application (client) ID into the appsettings.user.json file. If you started from the template settings provided in the RELEASE-NOTES.md file, replace *[CLIENTID-ASSIGNED-IN-AZUREAD]* with the client ID from the Azure portal.

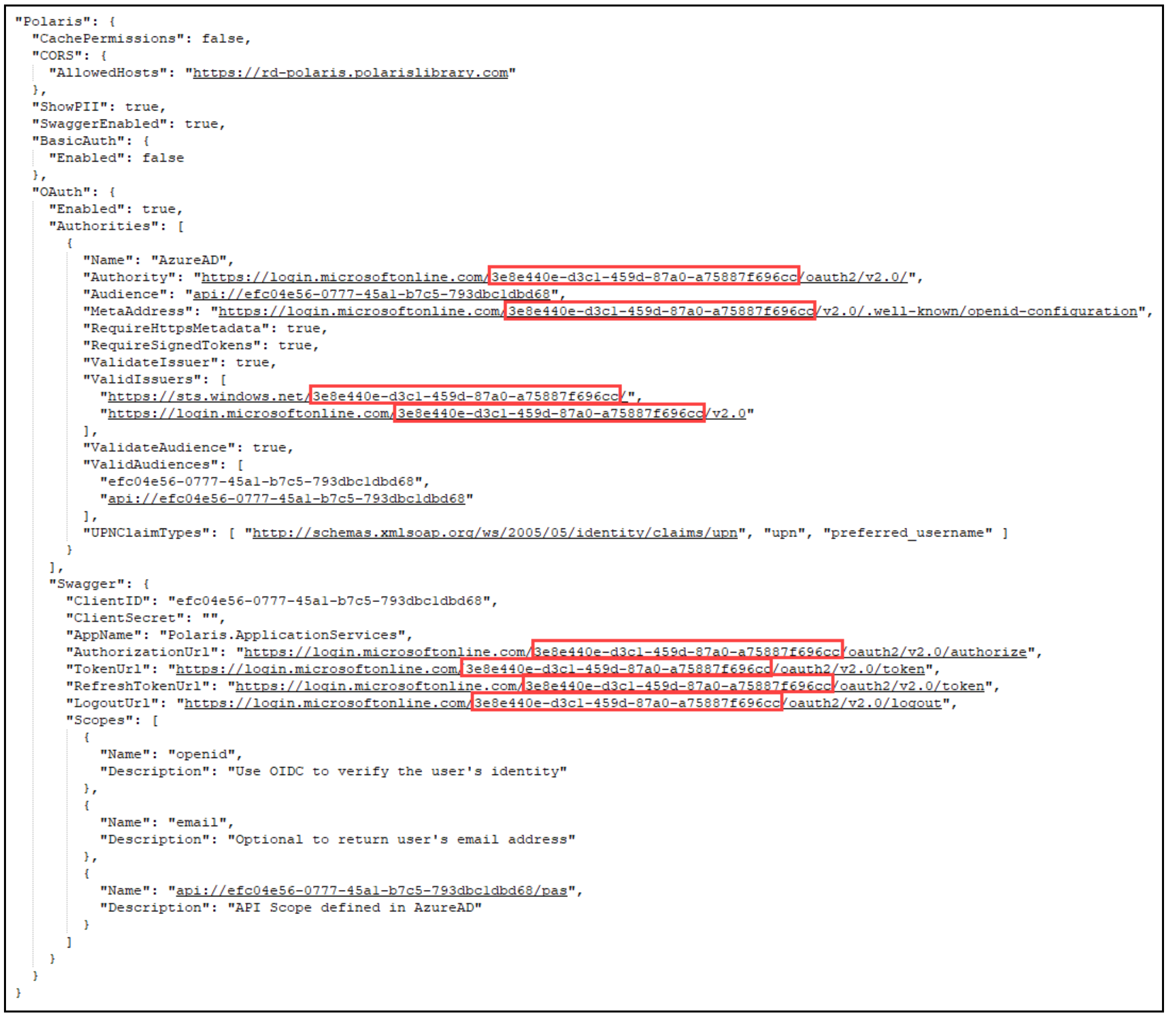
   When complete, your file should look like the following example (your client ID will be different):



```
"Polaris": {
  "CachePermissions": false,
  "CORS": {
    "AllowedHosts": "https://rd-polaris.polarislibrary.com"
  },
  "ShowPII": true,
  "SwaggerEnabled": true,
  "BasicAuth": {
    "Enabled": false
  },
  "OAuth": {
    "Enabled": true,
    "Authorities": [
      {
        "Name": "AzureAD",
        "Authority": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/",
        "Audience": "api://efc04e56-0777-45a1-b7c5-793dbc1dbd68",
        "MetaAddress": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/v2.0/.well-known/openid-configuration",
        "RequireHttpsMetadata": true,
        "RequireSignedTokens": true,
        "ValidateIssuer": true,
        "ValidIssuers": [
          "https://sts.windows.net/3e8e440e-d3c1-459d-87a0-a75887f696cc/",
          "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/v2.0"
        ],
        "ValidateAudience": true,
        "ValidAudiences": [
          "efc04e56-0777-45a1-b7c5-793dbc1dbd68",
          "api://efc04e56-0777-45a1-b7c5-793dbc1dbd68"
        ],
        "UPNClaimTypes": [ "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn", "upn", "preferred_username" ]
      }
    ],
    "Swagger": {
      "ClientID": "efc04e56-0777-45a1-b7c5-793dbc1dbd68",
      "ClientSecret": "",
      "AppName": "Polaris.ApplicationServices",
      "AuthorizationUrl": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/authorize",
      "TokenUrl": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/token",
      "RefreshTokenUrl": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/token",
      "LogoutUrl": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/logout",
      "Scopes": [
        {
          "Name": "openid",
          "Description": "Use OIDC to verify the user's identity"
        },
        {
          "Name": "email",
          "Description": "Optional to return user's email address"
        },
        {
          "Name": "api://efc04e56-0777-45a1-b7c5-793dbc1dbd68/pas",
          "Description": "API Scope defined in AzureAD"
        }
      ]
    }
  }
}
```

5. Paste the directory (tenant) ID into the appsettings.user.json file. If you started from the template settings provided in the RELEASE-NOTES.md file, replace *[TENANTID-ASSIGNED-IN-AZUREAD]* with the tenant ID from the Azure portal.

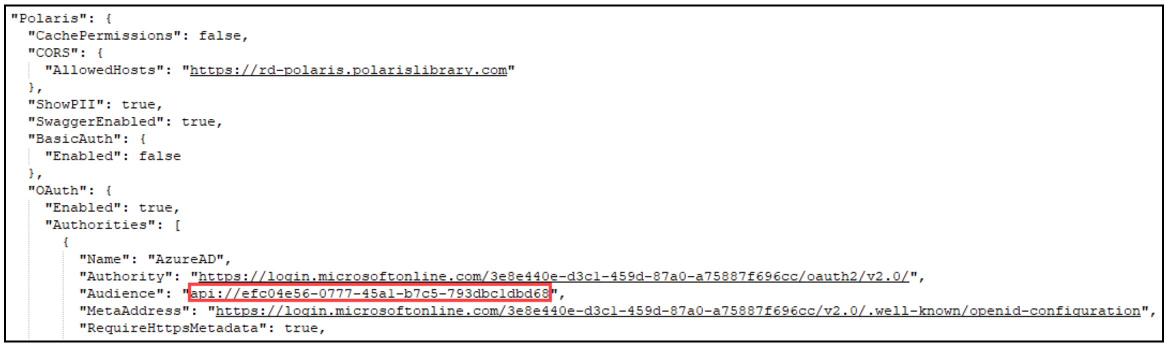
   When complete, your file should look like the following example (your tenant ID will be different):



```
"Polaris": {
  "CachePermissions": false,
  "CORS": {
    "AllowedHosts": "https://rd-polaris.polarislibrary.com"
  },
  "ShowPII": true,
  "SwaggerEnabled": true,
  "BasicAuth": {
    "Enabled": false
  },
  "OAuth": {
    "Enabled": true,
    "Authorities": [
      {
        "Name": "AzureAD",
        "Authority": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/",
        "Audience": "api://efc04e56-0777-45a1-b7c5-793dbc1dbd68",
        "MetaAddress": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/v2.0/.well-known/openid-configuration",
        "RequireHttpsMetadata": true,
        "RequireSignedTokens": true,
        "ValidateIssuer": true,
        "ValidIssuers": [
          "https://sts.windows.net/3e8e440e-d3c1-459d-87a0-a75887f696cc/",
          "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/v2.0"
        ],
        "ValidateAudience": true,
        "ValidAudiences": [
          "efc04e56-0777-45a1-b7c5-793dbc1dbd68",
          "api://efc04e56-0777-45a1-b7c5-793dbc1dbd68"
        ],
        "UPNClaimTypes": [ "http://schemas.xmlsoap.org/ws/2005/05/identity/claims/upn", "upn", "preferred_username" ]
      }
    ],
    "Swagger": {
      "ClientID": "efc04e56-0777-45a1-b7c5-793dbc1dbd68",
      "ClientSecret": "",
      "AppName": "Polaris.ApplicationServices",
      "AuthorizationUrl": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/authorize",
      "TokenUrl": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/token",
      "RefreshTokenUrl": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/token",
      "LogoutUrl": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/logout",
      "Scopes": [
        {
          "Name": "openid",
          "Description": "Use OIDC to verify the user's identity"
        },
        {
          "Name": "email",
          "Description": "Optional to return user's email address"
        },
        {
          "Name": "api://efc04e56-0777-45a1-b7c5-793dbc1dbd68/pas",
          "Description": "API Scope defined in AzureAD"
        }
      ]
    }
  }
}
```

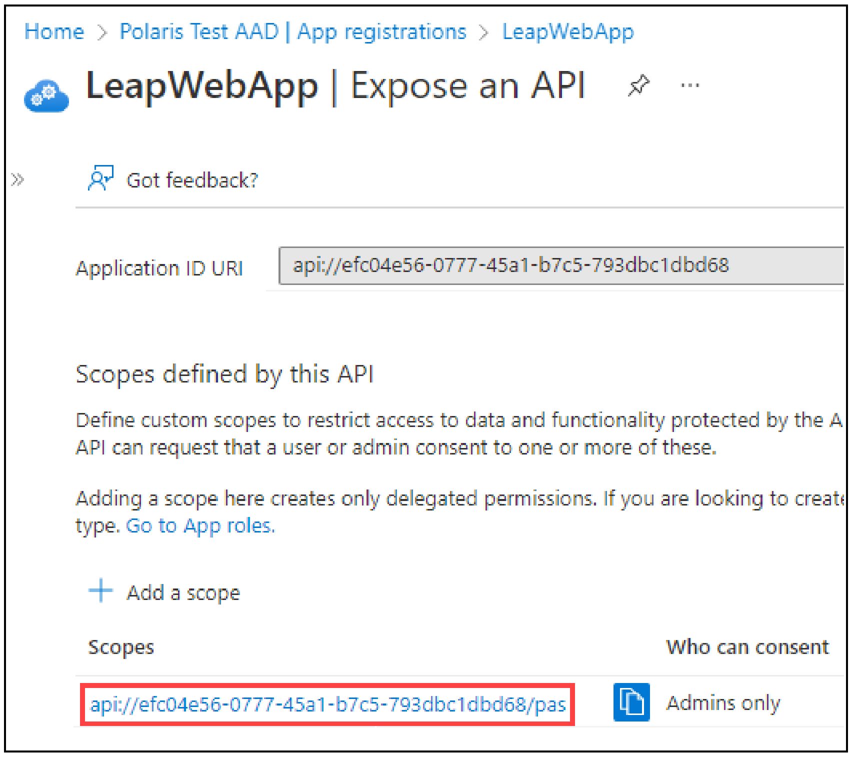
6. Paste the application ID URI into the appsettings.user.json file. If you started from the template settings provided in the RELEASE-NOTES.md file, replace *[APPID-URI-ASSIGNED-IN-AZUREAD]* with the application ID URI from the Azure portal.

   When complete, your file should look like the following example (your application ID URI will be different):



```
"Polaris": {
  "CachePermissions": false,
  "CORS": {
    "AllowedHosts": "https://rd-polaris.polarislibrary.com"
  },
  "ShowPII": true,
  "SwaggerEnabled": true,
  "BasicAuth": {
    "Enabled": false
  },
  "OAuth": {
    "Enabled": true,
    "Authorities": [
      {
        "Name": "AzureAD",
        "Authority": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/",
        "Audience": "api://efc04e56-0777-45a1-b7c5-793dbc1dbd68",
        "MetaAddress": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/v2.0/.well-known/openid-configuration",
        "RequireHttpsMetadata": true,
```

7. In the Azure portal, go to the LeapWebApp section of the App registrations dashboard and copy the API scope.


Home > Polaris Test AAD | App registrations > LeapWebApp
LeapWebApp | Expose an API
Got feedback?
Add a scope

Application ID URI: api://efc04e56-0777-45a1-b7c5-793dbc1dbd68

Scopes defined by this API

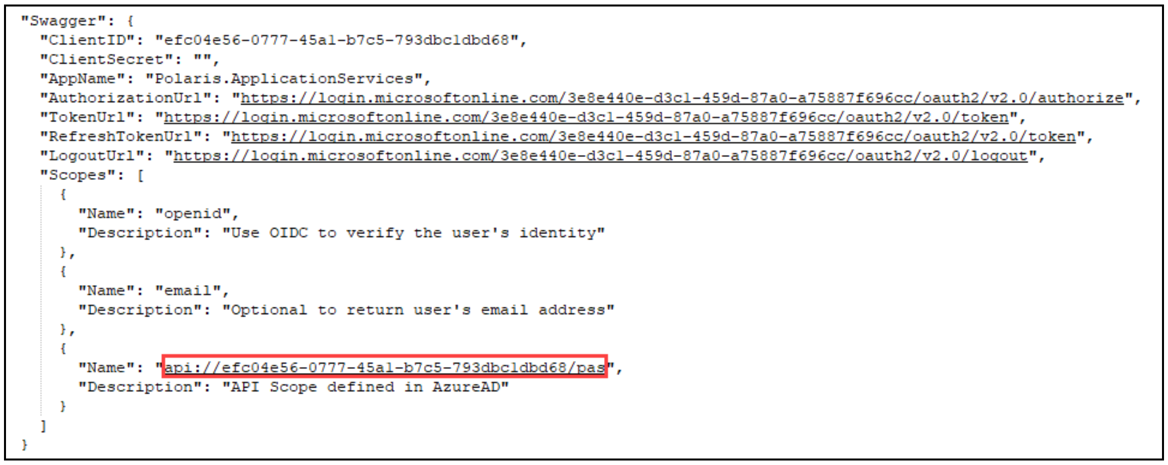
Define custom scopes to restrict access to data and functionality protected by the A
API can request that a user or admin consent to one or more of these.

Adding a scope here creates only delegated permissions. If you are looking to creat
type. Go to App roles.

Scopes	Who can consent
api://efc04e56-0777-45a1-b7c5-793dbc1dbd68/pas	Admins only

8. Paste the API scope into the appsettings.user.json file.

   When complete, your file should look like the following example (your scope value will be different):



```
"Swagger": {
  "ClientID": "efc04e56-0777-45a1-b7c5-793dbc1dbd68",
  "ClientSecret": "",
  "AppName": "Polaris.ApplicationServices",
  "AuthorizationUrl": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/authorize",
  "TokenUrl": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/token",
  "RefreshTokenUrl": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/token",
  "LogoutUrl": "https://login.microsoftonline.com/3e8e440e-d3c1-459d-87a0-a75887f696cc/oauth2/v2.0/logout",
  "Scopes": [
    {
      "Name": "openid",
      "Description": "Use OIDC to verify the user's identity"
    },
    {
      "Name": "email",
      "Description": "Optional to return user's email address"
    },
    {
      "Name": "api://efc04e56-0777-45a1-b7c5-793dbc1dbd68/pas",
      "Description": "API Scope defined in AzureAD"
    }
  ]
}
```

9. Save the .json file.

Add a URL Rewrite Rule for LeapWebApp

Adding a URL rewrite rule redirects incoming URLs to the correct address for the LeapWebApp. This must be done manually, since the library may already use other URL rewrite rules.

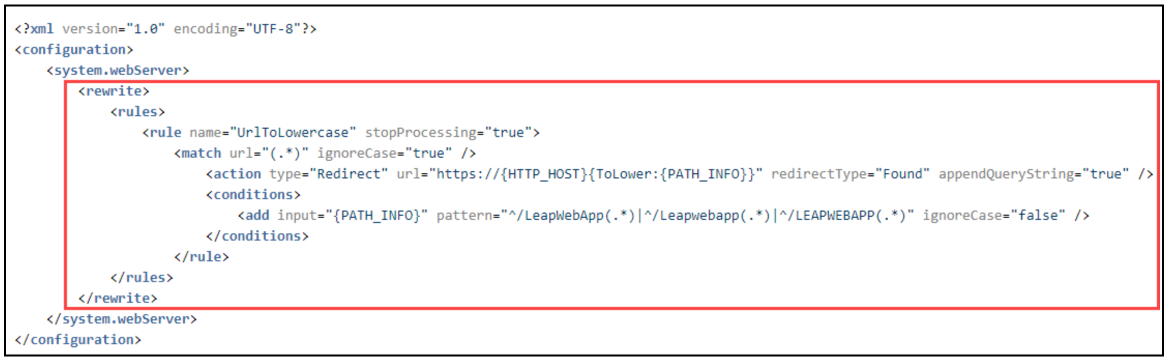
To add a URL rewrite rule, you must have the Microsoft IIS URL Rewrite 2.1 extension. For more information, see https://www.iis.net/downloads/microsoft/url-rewrite.

To add a URL rewrite rule

1. Open the root IIS web.config file, found in the following location:

   C:\inetpub\wwwroot\web.config

2. Add a rewrite rule to the **system.webServer** node.



```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <system.webServer>
        <rewrite>
            <rules>
                <rule name="UrlToLowercase" stopProcessing="true">
                    <match url="(.*)" ignoreCase="true" />
                        <action type="Redirect" url="https://{HTTP_HOST}{ToLower:{PATH_INFO}}" redirectType="Found" appendQueryString="true" />
                        <conditions>
                            <add input="{PATH_INFO}" pattern="^/LeapWebApp(.*)|^/Leapwebapp(.*)|^/LEAPWEBAPP(.*)" ignoreCase="false" />
                        </conditions>
                    </rule>
            </rules>
        </rewrite>
    </system.webServer>
</configuration>
```

> **Note:**
> For sample rewrite rule text that you can copy and paste, see Sample Rewrite Rule Text.

   In the example above, if the incoming URL includes a path that contains any of the following, the rewrite rule redirects to /leapwebapp:

   - /LeapWebApp
   - /Leapwebapp
   - /LEAPWEBAPP

3. Save the web.config file.

> **Note:**
> When registering redirect URIs for LeapWebApp in AD FS, the URIs should

be lowercase. For example:

- https://rd-polaris.polarislibrary.com/leapwebapp/signin-oidc
- https://rd-polaris.polarislibrary.com/leapwebapp/signin-override-oidc
- https://rd-polaris.polarislibrary.com/leapwebapp/signout-callback-oidc

Sample Rewrite Rule Text

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
   <system.webServer>
      <rewrite>
         <rules>
            <rule name="UrlToLowercase" stopProcessing="true">
               <match url="(.*)" ignoreCase="true" />
                  <action type="Redirect" url="https://{HTTP_HOST}
                  {ToLower:{PATH_INFO}}" redirectType="Found"
                  appendQueryString="true" />
                  <conditions>
                     <add input="{PATH_INFO}" pattern="^/LeapWebApp
                     (.*)|^/Leapwebapp(.*)|^/LEAPWEBAPP(.*)"
                     ignoreCase="false" />
                  </conditions>
            </rule>
         </rules>
      </rewrite>
   </system.webServer>
</configuration>
```

Additional URL Rewrite Resources

See Microsoft's URL Rewrite Module Configuration Reference for additional information:

- https://docs.microsoft.com/en-us/iis/extensions/url-rewrite-module/url-rewrite-module-configuration-reference