

SSO Implementation Guide: Azure AD (Microsoft Entra ID) + AWS Cognito + Next.js

1. Purpose

This document explains the complete Single Sign-On (SSO) implementation used in this project:

- Identity Provider (IdP): Microsoft Entra ID (Azure AD)
 - Federation Broker: AWS Cognito Hosted UI
 - Application: Next.js frontend
 - OAuth Flow: Authorization Code + PKCE
-

2. Final Architecture

1. User clicks **Sign in with Microsoft** in the app.
 2. App redirects to **AWS Cognito Hosted UI** `/oauth2/authorize`.
 3. Cognito forwards authentication to **Azure AD** (configured IdP name: `AzureAD`).
 4. Azure AD redirects back to Cognito at:
 - `https://<cognito-domain>.auth.<region>.amazoncognito.com/oauth2/idpresponse`
 5. Cognito completes federation and redirects to app callback URL:
 - `<app-callback-url>` (example: `https://<app-domain>/api/auth/callback`)
 6. App exchanges auth code at Cognito `/oauth2/token` and stores tokens.
-

3. Prerequisites

- AWS account with Cognito permissions
 - Azure/Entra tenant with App Registration permissions
 - Cognito User Pool domain configured
 - Next.js app environment variables configured
-

4. Azure AD Setup

4.1 Create App Registration

In Azure portal:

1. Go to **Microsoft Entra ID** → **App registrations** → **New registration**.
2. Name: `Philips-Sensei` (or tenant naming standard).
3. Supported account type: choose per org policy (typically single-tenant).
4. Create the application.

Home > App registrations

Task Amplify

Deactivate Delete Endpoints Preview features

Search

Overview

- Quickstart
- Integration assistant
- Diagnose and solve problems
- Manage
 - Branding & properties
 - Authentication (Preview)
 - Certificates & secrets
 - Token configuration
 - API permissions
 - Expose an API
 - App roles
 - Owners
 - Roles and administrators
 - Manifest

Essentials

Display name Task Amplify	Client credentials 0 certificate, 1 secret
Application (client) ID 24ec5af7-b4e1-4b45-962c-3b3e5cd9c379	Redirect URIs 1 web, 0 spa, 0 public client
Object ID e007087e-c7d9-4dad-82dc-b06908510c2e	Application ID URI Add an Application ID URI
Directory (tenant) ID 21f3742b-336b-4e6b-bc0b-8a0ef1a779fa	Managed application in local directory Task Amplify
Supported account types My organization only	State Activated

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

[Get Started](#) [Documentation](#)

Build your application with the Microsoft identity platform

4.2 Configure Redirect URI (Critical)

Add **Web** redirect URI:

- <https://<cognito-domain>.auth.<region>.amazoncognito.com/oauth2/idpresponse>

Why this matters:

- This URI is for **Azure** → **Cognito** federation callback.
- Do **not** use the app callback ([/api/auth/callback](#)) here.

Home > App registrations > Task Amplify

Task Amplify | Authentication (Preview)

Got feedback?

Welcome to the new and improved experience for authentication. [To switch to the old experience, please click here.](#)

Redirect URI configuration Supported accounts Settings

A redirect URI, or reply URL, is the location where the Entra authorization server sends the user and delivers tokens once the user has successfully signed in or signed out. [Learn more about what is a redirect URI](#)

+ Add Redirect URI Delete

If you want to view or change implicit grant and hybrid flows settings or front-channel logout URL, please visit the settings tab.

Start typing a reply url to filter these results Platform Type: All

Platform Type ↑↓	Redirect URI ↑↓
Web	Edit
Web	https://ap-south-1caroawzdd.auth.ap-south-1.amazoncognito.com/oauth2/idpresponse

4.3 Generate Client Secret (for OIDC)

- Go to **Certificates & secrets**.
- Create a new client secret.
- Copy and securely store the secret value.

Used where:

- Cognito OIDC IdP configuration requires Azure Client ID + Client Secret.

Home > App registrations > Task Amplify

Task Amplify | Certificates & secrets

Search

Got feedback?

Overview

Quickstart

Integration assistant

Diagnose and solve problems

Manage

Branding & properties

Authentication (Preview)

Certificates & secrets

Token configuration

API permissions

Expose an API

App roles

Owners

Roles and administrators

Manifest

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.

Certificates (0) Client secrets (1) Federated credentials (0)

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

New client secret

Description	Expires	Value	Secret ID
Task Amplify Secret	2/20/2028	C3O*****	f2777774-4d14-4933-8b6a-c700dd823...

4.4 API Permissions

Ensure delegated OpenID permissions are granted:

- openid
- profile
- email

Home > App registrations > Task Amplify

Task Amplify | API permissions

Search

Refresh

Got feedback?

Overview

Quickstart

Integration assistant

Diagnose and solve problems

Manage

Branding & properties

Authentication (Preview)

Certificates & secrets

Token configuration

API permissions

Expose an API

App roles

Owners

Roles and administrators

Manifest

Add a permission

Grant admin consent for Default Directory

API / Permissions name	Type	Description	Admin consent req...	Status
Microsoft Graph (1)				
User.Read	Delegated	Sign in and read user profile	No	

Other permissions granted for Default Directory

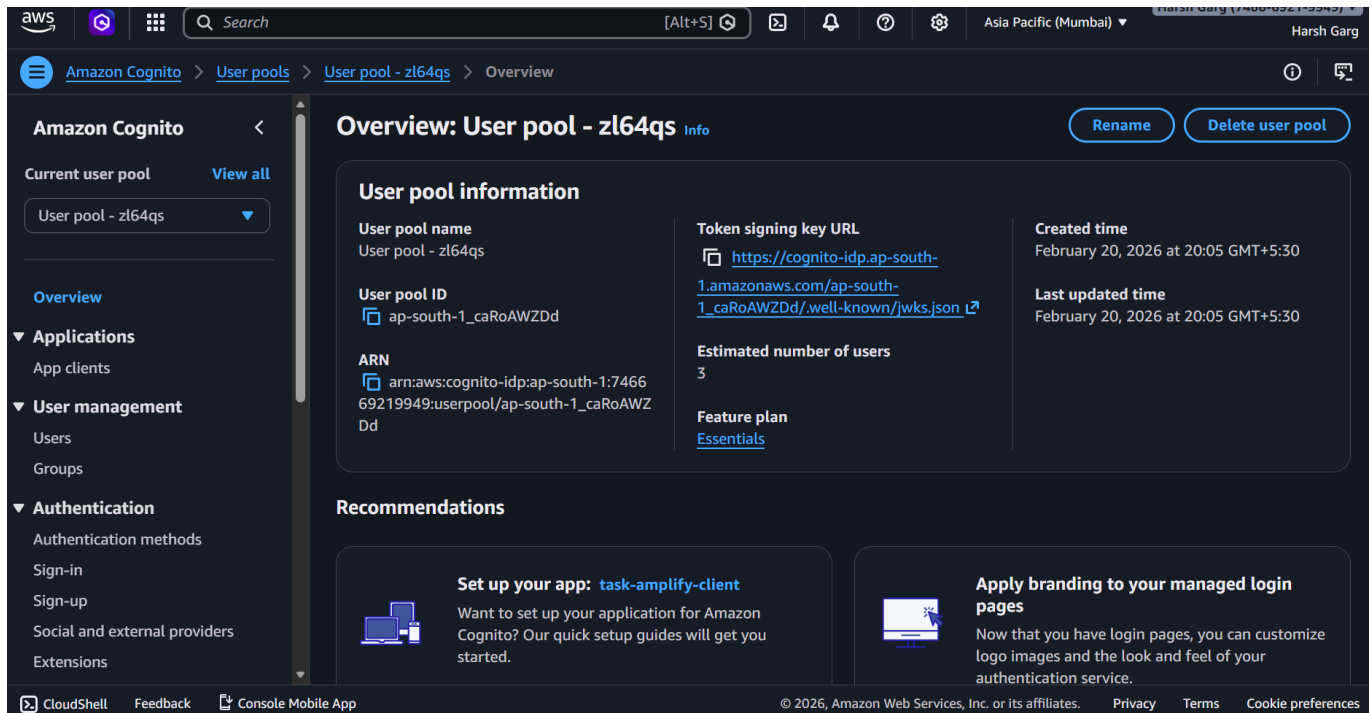
These permissions have been granted for Default Directory but aren't in the configured permissions list. If your application requires these permissions, you should consider adding them to the configured permissions list.

API / Permissions name	Type	Description	Admin consent req...	Status
Microsoft Graph (3)				
email	Delegated	View users' email address	No	Granted for Default Dire...
openid	Delegated	Sign users in	No	Granted for Default Dire...
profile	Delegated	View users' basic profile	No	Granted for Default Dire...

5. AWS Cognito Setup

5.1 User Pool + Domain

1. Create/use existing User Pool.
2. Configure Hosted UI domain.
3. Domain used in this project pattern:
 - <https://<cognito-domain>.auth.<region>.amazoncognito.com>



5.2 Add Azure AD as Identity Provider

In User Pool federation settings:

1. Add external IdP (OIDC).
2. Provider name: **AzureAD**.
3. Setup method: **Auto fill through issuer URL**.
4. Enter Azure OIDC Issuer URL in this format:
 - <https://login.microsoftonline.com/<tenant-id>/v2.0>
5. Cognito auto-populates OIDC endpoints (**authorize**, **token**, **userInfo**, **jwks_uri**) from Azure discovery metadata.
6. Enter Azure Client ID and Client Secret.
7. Enter openid profile email as your Authorized Scopes.
8. Save provider.

Edit identity provider: AzureAD

OIDC is a supported social OAuth 2.0 identity provider for Amazon Cognito user pools.

OpenID Connect (OIDC)
Configures an OIDC identity provider for your user pool.

Provider name info
Enter a friendly name for your OpenID Connect identity provider.
AzureAD

Client ID info
Enter the client ID provided by OpenID Connect identity provider.
24c5af7-b4e1-4b45-962c-3b3e5c9c379

Client secret info
Enter the client secret provided by OpenID Connect identity provider.
[Masked]

Authorized scopes info
Choose the OAuth 2.0 scopes that you will request from OpenID Connect. Scopes are groups of OpenID Connect user attributes to exchange with your app. By default, the scopes required for federation with this user pool have been selected for you. You may add more scopes if needed.
openid profile email

Separate scopes with spaces.

Identifiers - optional info
Enter identifiers for this provider. Identifiers can be used to redirect users to the correct SFP in multitenant apps.
[Empty field]

Separate each identifier by a comma.

Attribute request method info
The request method is configured in the provider application.
☒ GET
☐ POST

Setup method info
☒ Auto fill through issuer URL. Provides an issuer URL, and Cognito will populate values for the authentication endpoint, token endpoint, userinfo endpoint, and jwks_url.
☐ Manual input. Manually input values for the authentication endpoint, token endpoint, userinfo endpoint, and jwks_url.

Issuer URL info
Enter the issuer URL you received from the OIDC provider.
https://login.microsoftonline.com/21f3742b-356b-4e6b-bc0b-8a0e1a779fa/v2.0

[Cancel](#) [Save changes](#)

How to get `<tenant-id>`:

- Azure Portal → App Registrations → Your App → Overview → Tenant ID

Important clarification:

- The Issuer URL is **not generated by Cognito**.
- You provide the tenant-specific issuer URL from Azure.
- Cognito only auto-fills endpoint fields after you provide the Issuer URL.

Notes:

- If configured as SAML instead, secret is not used.
- This implementation assumes OIDC because the app uses OIDC scopes and `identity_provider=AzureAD`.

The top screenshot shows the 'Social and custom providers' page in the Amazon Cognito console. It displays a list of federated identity providers for a specific user pool. The 'AzureAD' provider is listed with an 'OIDC' type, created 7 days ago, and last updated 4 days ago. A warning box indicates that self-registration is enabled, which allows anyone on the internet to sign up for a local user account. A button 'Edit self-registration' is available.

The bottom screenshot shows the 'Identity provider: AzureAD' configuration page. It provides detailed information about the provider, including its type (OIDC), name (AzureAD), Client ID, Client secret, and various endpoints like Token endpoint, Jwks_uri endpoint, Issuer, and Authorization endpoint. There are also buttons for 'View signing certificate', 'Edit', and 'Delete'.

5.3 Configure App Client

In Cognito App Client settings:

1. Enable **Authorization code grant**.
2. Callback URL:
 - `<app-callback-url>` (example: `https://<app-domain>/api/auth/callback`)
3. Sign-out URL:
 - `<app-logout-url>` (example: `https://<app-domain>/login`)
4. Allowed scopes:
 - `openid`
 - `profile`
 - `email`
5. Enable Identity providers: **AzureAD** (and Cognito native if needed by policy).

Managed login pages
Configure the managed login pages for this app client.

Allowed callback URLs info
Enter at least one callback URL to redirect the user back to after authentication. This is typically the URL for the app receiving the authorization code issued by Cognito. You may use HTTPS URLs, as well as custom URL schemes.
URL
 Remove
Length of callback URL must be between 1 and 1024 characters. Valid characters are letters, marks, numbers, symbols, and punctuations. Amazon Cognito requires HTTPS over HTTP except for http://localhost for testing purposes only. App callback URLs such as myapp://example are also supported. Must not contain a fragment.
Add another URL
You can add 99 more URLs.

Default redirect URL
The default redirect URL in app clients with one assigned IDP replaces redirect_uri in authentication requests. Must be in the Allowed callback URLs list.
 Remove
Length of sign-out URL must be between 1 and 1024 characters. Valid characters are letters, marks, numbers, symbols, and punctuations. Amazon Cognito requires HTTPS over HTTP except for http://localhost for testing purposes only. App sign-out URLs such as myapp://example are also supported. Must not contain a fragment.
Add another URL
You can add 99 more URLs.

Identity providers info
Select the identity providers that will be available to this app client.

Select identity providers +

AzureAD x Cognito user pool x
Users can sign in to Cognito using an email, phone number, or username.

OAuth 2.0 grant types info
Choose at least one OAuth 2.0 grant type to configure how Cognito will deliver tokens to this app. We have populated suggested options based on the app type you selected.

Select OAuth 2.0 grant types ▼

Authorization code grant x
Provides an authorization code as the response.

OpenID Connect scopes info
Choose at least one OpenID Connect (OIDC) scope to specify the attributes this app client can retrieve for access tokens. We have populated suggested options based on the application type and required attributes you selected.

Select OIDC scopes ▼

Email x OpenID x Profile x
Requires OpenID to be selected.

Custom scopes info
Select custom scopes that you will authorize for this app. Custom scopes are configured with resource servers.

Select custom scopes ▼ +

Cancel Save changes

6. Application Configuration

In `.env.local` (frontend):

```
NEXT_PUBLIC_COGNITO_REGION=<aws-region>
NEXT_PUBLIC_COGNITO_USER_POOL_ID=<user-pool-id>
NEXT_PUBLIC_COGNITO_CLIENT_ID=<app-client-id>
NEXT_PUBLIC_COGNITO_DOMAIN=<cognito-domain-prefix>
NEXT_PUBLIC_COGNITO_REDIRECT_URI=<app-callback-url>
NEXT_PUBLIC_COGNITO_LOGOUT_URI=<app-logout-url>
NEXT_PUBLIC_COGNITO_SCOPES=openid profile email
NEXT_PUBLIC_IDENTITY_PROVIDER=AzureAD
```

Behavior implemented:

- Login button triggers Hosted UI URL generation.
- URL includes:
 - `response_type=code`
 - `identity_provider=AzureAD`
 - `code_challenge_method=S256` (PKCE)
- Callback page exchanges code at Cognito `/oauth2/token`.
- Tokens are stored in browser storage for session continuity.

7. End-to-End Validation Steps

1. Open app login page.
2. Click **Sign in with Microsoft**.
3. Confirm browser goes to Cognito Hosted UI authorize endpoint.
4. Confirm redirect to Microsoft login.

5. Sign in with test account.
6. Confirm redirect path sequence:
 - Azure → `/oauth2/idpresponse` (Cognito)
 - Cognito → `/api/auth/callback` (app)
7. Confirm app lands on `/home` with authenticated session.

Evidence to capture:

- Browser address bar at each redirect stage
 - Cognito Hosted UI network calls
 - Successful `/oauth2/token` response
-