SSO SETUP PART 3

Implement Microsoft Entra SSO Login in Intelli Broadcast

In Part 2, we configured **Norvia Backoffice** to create Microsoft Entra users and assign them product-specific roles.

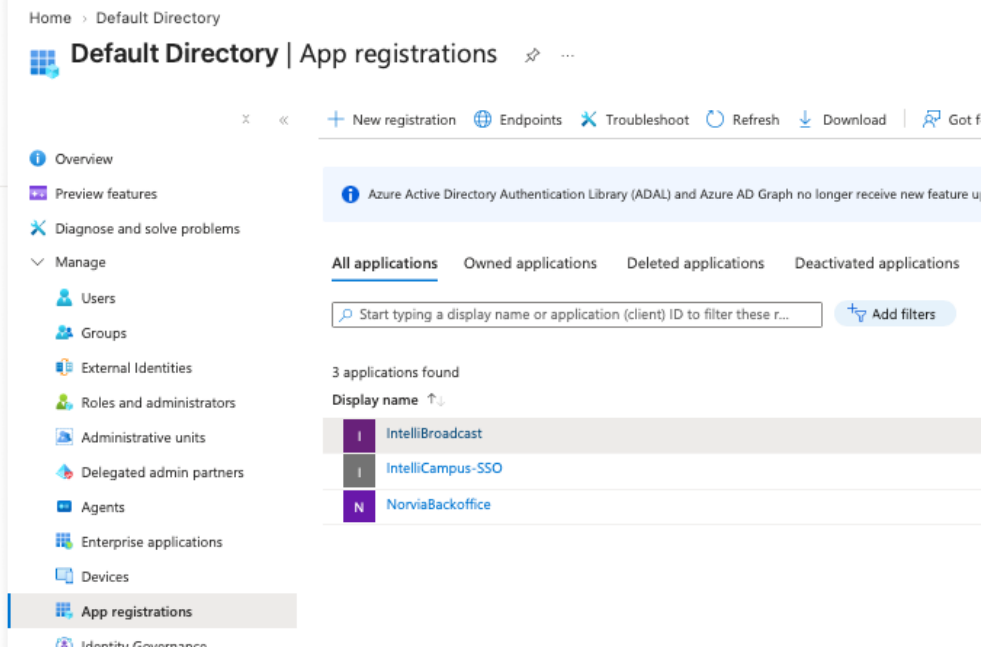
Now, in Part 3, we will update the existing **Intelli Broadcast (.NET)** application to authenticate those users through Microsoft Entra ID.

We will complete this in two stages:

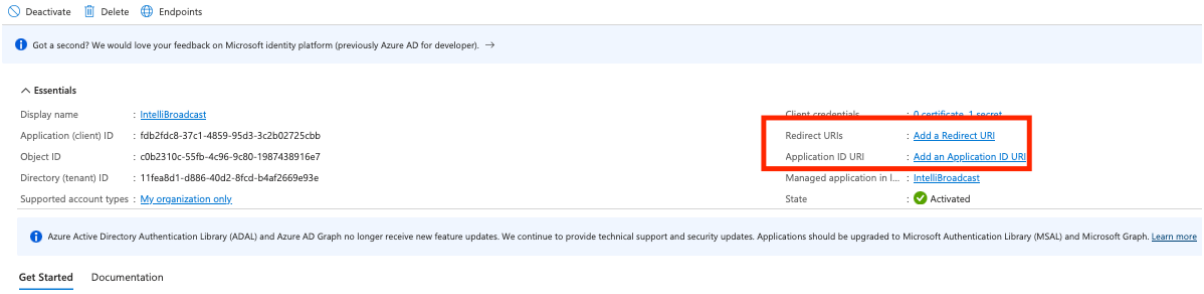
1. **Configure Intelli Broadcast in Microsoft Entra ID.**
2. **Use Codex to update the existing Intelli Broadcast login functionality.**

STEP 1: Configure the Redirect URL in Microsoft Entra

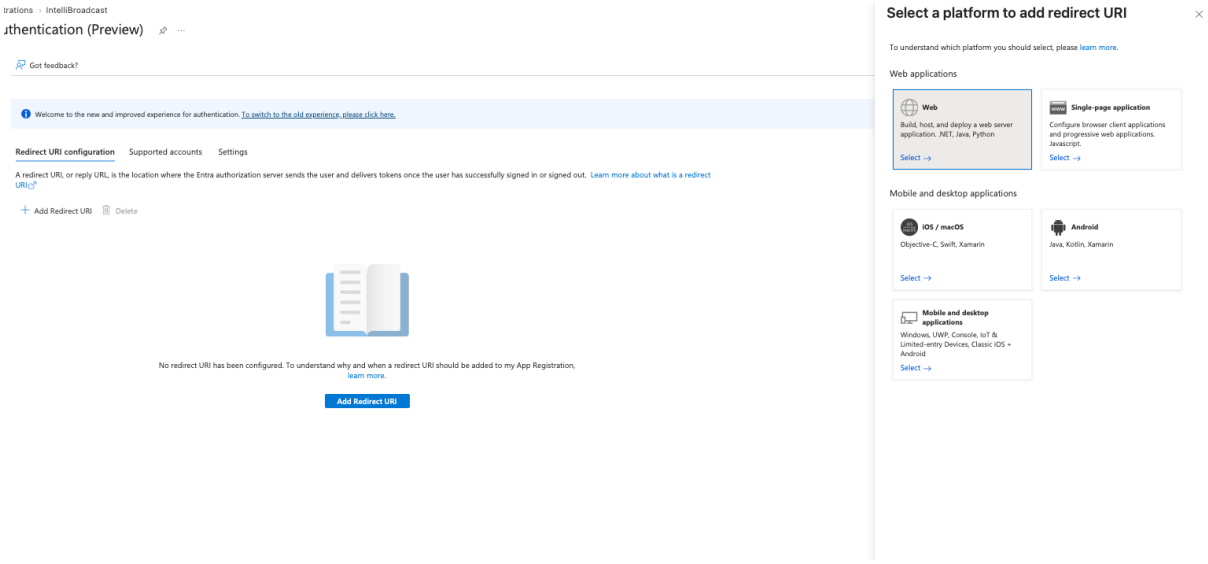
- Open portal.azure.com
- Navigate to Microsoft Entra ID
- Go to Manage >> App Registration >> All applications
- Select IntelliBroadcast (Or your application)

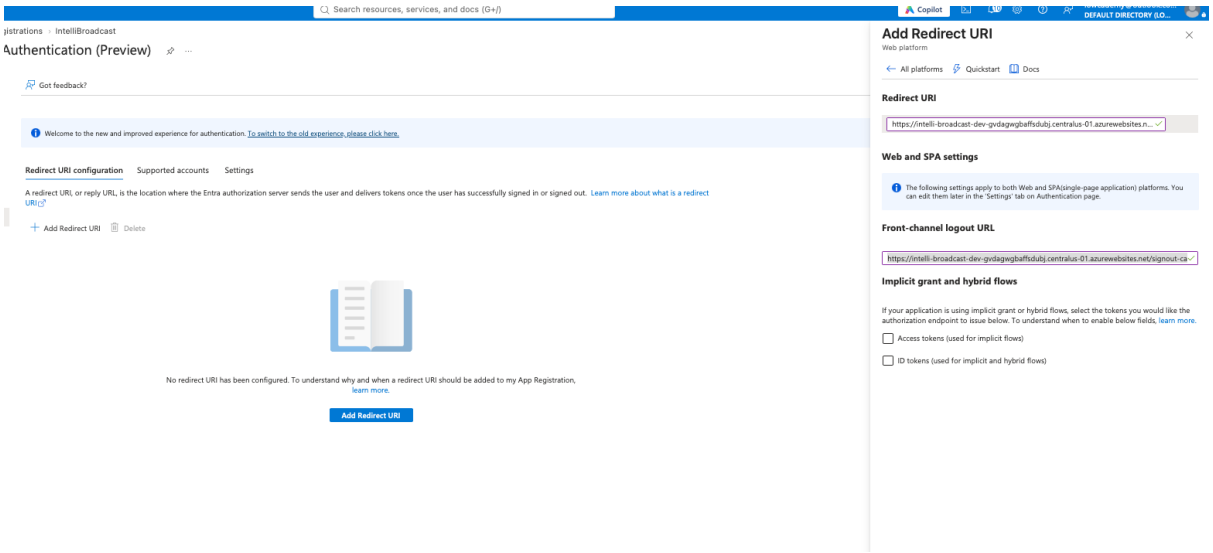


- Click on Add a redirect URI



- Click on Add button
- Select Web
- And Provide the URL





Provide Valid URLS

Required URI

<your Intelli Broadcast URL>/ `signin-oidc`

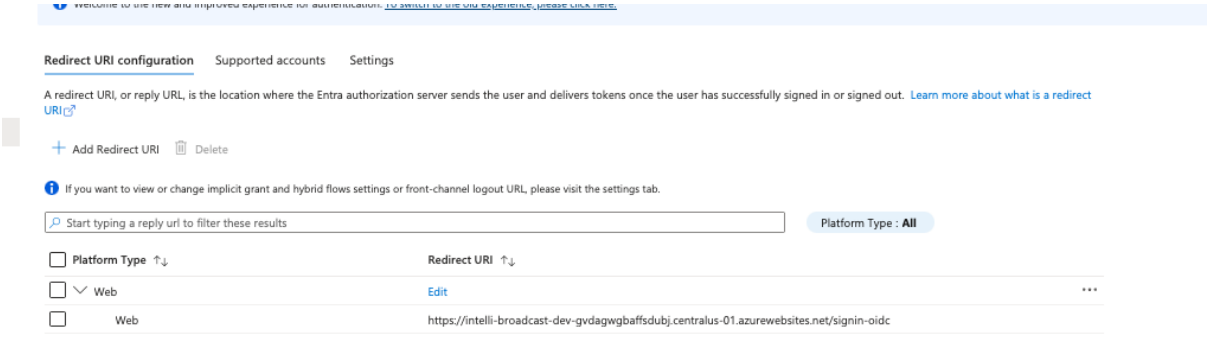
Front-channel logout URL

<your Intelli Broadcast URL>/ `signin-oidc`

For Example:
https://intelli-broadcast-dev-gvdagwgbaffsdubj.centralus-01.azurewebsites.net/signin-oidc

https://intelli-broadcast-dev-gvdagwgbaffsdubj.centralus-01.azurewebsites.net/signout-callback-oidc

- Click on Configure.
- IT should look like



STEP 2: Verify the client ID and secret

In the Entra App Check the Client ID, Client Secret, and Tenant ID and keep that saved. You will have to configure that under Norvia Backoffice against the product.

You should have details handy like

App:

IntelliBroadcast

AppID: fdb2fdc8-37c1-4859-95d3-3c2b02725cbb

TenantID: 11fea8d1-d886-40d2-8fcd-b4af2669e93e

Secret: 3pA8Q~Qnj-g4mjck-JaZSciXqr4spMvFIJoEMayn

Redirect URI: https://intelli-broadcast-dev-gvdagwgbaffsdubj.centralus-01.azurewebsites.net/signin-oidc

Logout URL: https://intelli-broadcast-dev-gvdagwgbaffsdubj.centralus-01.azurewebsites.net/signout-callback-oidc

STEP 3: Understand the new login flow.

Currently, Intelli Broadcast has its own login procedure.

We will replace its authentication entry point with Microsoft Entra SSO.

Updated flow

User opens Intelli Broadcast

|

v

User clicks Login

|

v

Redirect to Microsoft Entra

|

v

User enters:

Login Email + Password

|

v

Microsoft Entra authenticates

|

v

Redirect to Intelli Broadcast

|

v

Validate Microsoft identity

|

v

Get Entra Tenant ID + Object ID

|

v

Check ExternalIdentityMapping

|

v

Find UserExtended

|

v

Check UserExtended.IsActive

|

v

Check Tenant.IsActive

|

v

Check Intelli Broadcast Product Role

|

v

Create Application Session

|

v

Open Intelli Broadcast Dashboard

How will Intelli Broadcast identify the user?

Suppose Microsoft Entra authenticates:

`john.smith@lowcademy.onmicrosoft.com`

Microsoft returns the user's identity claims.

The application will use:

- `oid` — Microsoft Entra Object ID
- `tid` — Microsoft Entra Tenant ID

These values identify the Microsoft user.

We will use them to find the corresponding `ExternalIdentityMapping` record.

From there:

`ExternalIdentityMapping → UserExtended → UserExtendedRoleMapping`

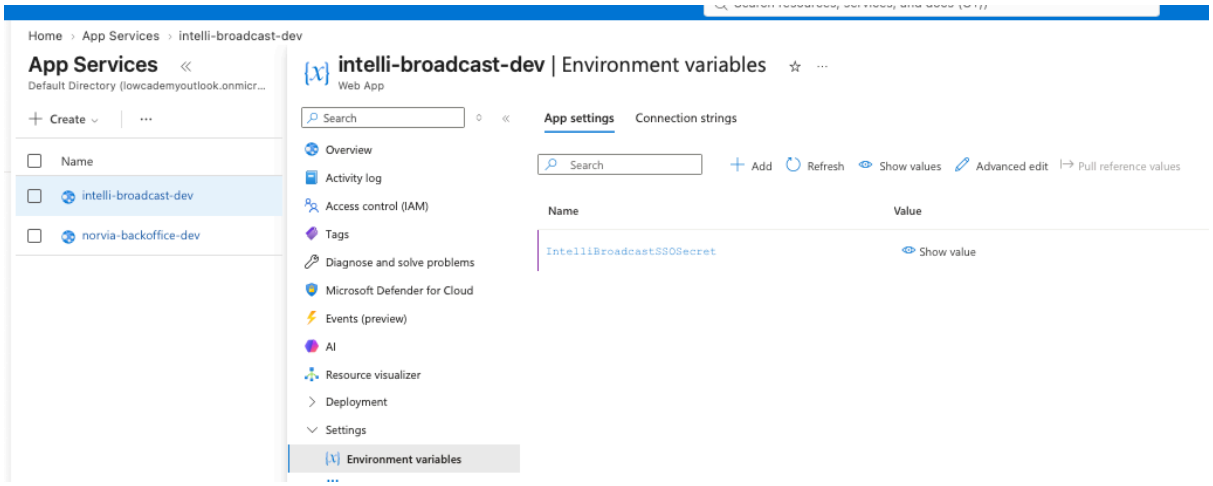
We will not use email matching as the primary identity lookup.

That is important because email addresses can change, while the Microsoft Entra Object ID identifies the account.

STEP 4: Configure Intelli Broadcast App Service

- Go to App Services
- Open Intelli Broadcast
- Open Settings
- Open Environment Variable
- And Set

`IntelliBroadcastSSOSecret = Actual Microsoft Entra Client secret`



Note: the Same Secret should be store while creating/update product in Norvia Backoffice against the Intelli Broadcast.

STEP 5: Make code changes in Intelli Broadcast

==========PROMPT BEIGN==========

Task: Replace Existing Login with Microsoft Entra SSO

Intelli Broadcast is an existing ASP.NET Core multi-tenant application. Microsoft Entra ID and its App Registration are already configured.

Replace only the existing login and logout functionality with Microsoft Entra SSO. Everything else must remain unchanged.

1. Application Settings

Add these settings to `appsettings.json`:

```
{
  "ProductHash": "",
  "AppBaseUrl": "",
  "SSO": {
    "CallbackPath": "/signin-oidc",
    "SignedOutCallbackPath": "/signout-callback-oidc"
  }
}
```

I will configure the actual ProductHash and AppBaseUrl values.

2. Load SSO Configuration

During application startup:

1. Read ProductHash from application settings.
2. Query the existing Azure SQL `NorviaProducts` table using ProductHash.
3. Retrieve `ID`, `SSOTenantID`, `SSOAppID`, and `SSOAppSecretReference`.
4. Resolve the client secret from the application setting named in `SSOAppSecretReference`.
5. Configure Microsoft Entra OpenID Connect authentication using these values.

Use authorization code flow and ASP.NET Core cookie authentication.

3. Configure SSO URLs

Use the following settings:

- Application Base URL: `AppBaseUrl`

- Login Callback Path: `SSO:CallbackPath`
- Logout Callback Path: `SSO:SignedOutCallbackPath`

The complete callback URLs are:

- Login: `{AppBaseUrl}/signin-oidc`
- Logout: `{AppBaseUrl}/signout-callback-oidc`

Configure these paths in the OpenID Connect middleware.

The middleware must handle these endpoints automatically. Do not create separate controllers or pages for them.

Ensure the generated redirect URI uses the configured HTTPS application host and matches the URI registered in Microsoft Entra.

Use the standard Microsoft OpenID Connect logout flow.

4. Replace Existing Login

When a user opens Intelli Broadcast:

1. Check whether a valid application session exists.
2. If valid, continue to the existing application.
3. Otherwise, automatically redirect to Microsoft's login page.
4. After successful Microsoft authentication, receive the callback at `/signin-oidc`.
5. Identify the user using validated Microsoft Entra `tid` and `oid` claims.
6. Validate the user using:
    - `ExternalIdentityMapping`
    - `UserExtended`
    - `Tenant`
    - `UserExtendedRoleMapping`
    - `ProductUserRole`
7. Confirm that the user and tenant are active and that the user has a valid Intelli Broadcast product role.
8. If authorized, create the existing application session and open the originally requested page.
9. Otherwise, deny access without issuing an authenticated application session.

Protect the Index page and existing application pages so unauthenticated requests trigger Microsoft Entra login automatically.

Do not create a custom login page.

5. Logout

Update the existing logout functionality to:

1. Clear the local authentication session.

2. Initiate Microsoft Entra sign-out.
3. Handle the logout callback through `/signout-callback-oidc`.
4. Redirect the user to the existing public or login destination without automatically starting another login challenge.

6. Restrictions

- Do not rebuild the application.
- Do not redesign existing screens.
- Do not modify business functionality.
- Do not change database schema or create migrations.
- Do not change existing tenant isolation or role permissions.
- Reuse the existing architecture and session management.
- Do not implement Microsoft Graph provisioning.
- Do not hardcode SSO credentials.

Only modify authentication configuration, login, session integration, and logout.

Build the application and fix compilation errors.

Provide the modified files, required packages, and configuration values I must populate.

========== Prompt END ==========

TEST and Deploy the App