SSO SETUP PART 2

Implement Microsoft Entra ID User Provisioning in Norvia Backoffice

Objective

Before implementing SSO in Intelli Campus and Intelli Broadcast, we will update Norvia Backoffice to manage Microsoft Entra ID users.

Norvia Backoffice already allows administrators to:

- Create and manage tenants.
- Create and manage products.
- Create and manage users through UserExtended.
- Assign product-specific roles through UserExtendedRoleMapping.

STEP 1: Understand the Updated Architecture

The application will:

1. Create the user in `UserExtended`.
2. Create a Microsoft Entra account through Microsoft Graph.
3. Store the Entra Object ID in `ExternalIdentityMapping`.
4. Create the product-role assignment in `UserExtendedRoleMapping`.

If the administrator later assigns Intelli Broadcast to the same user, the application will reuse the existing Entra identity.

Note: The user's Microsoft sign-in name will use your verified Entra domain, such as `john01@yourtenant.onmicrosoft.com`. The email entered in Norvia Backoffice can remain a separate contact email.

STEP 2: Update the Azure SQL Database, Alter existing tables

Step 2.1: Open Azure SQL Query Editor

1. Open Azure Portal.
2. Navigate to **SQL databases**.
3. Select `norvia-labs-database-dev`.
4. Open **Query editor**.
5. Sign in.
6. Click **New query**.

Step 2.2: Update NorviaProducts

You proposed adding these fields:

- SSO Tenant ID
- SSO App ID
- SSO App Secret

RUN below Query

```
IF COL_LENGTH('dbo.NorviaProducts', 'SSOTenantID') IS NULL
BEGIN
  ALTER TABLE dbo.NorviaProducts
  ADD SSOTenantID NVARCHAR(100) NULL;
END;

IF COL_LENGTH('dbo.NorviaProducts', 'SSOAppID') IS NULL
BEGIN
  ALTER TABLE dbo.NorviaProducts
  ADD SSOAppID NVARCHAR(100) NULL;
END;
```

```
IF COL_LENGTH('dbo.NorviaProducts', 'SSOAppSecretReference') IS NULL
BEGIN
  ALTER TABLE dbo.NorviaProducts
  ADD SSOAppSecretReference NVARCHAR(500) NULL;
END;
```

Step 2.3: Create ExternalIdentityMapping

This table maps your existing application users to Microsoft Entra users.

If you already created it in the previous part, this query will skip creation.

Execute:

```
IF OBJECT_ID('dbo.ExternalIdentityMapping', 'U') IS NULL
BEGIN
  CREATE TABLE dbo.ExternalIdentityMapping
  (
    ID BIGINT IDENTITY(1,1) NOT NULL PRIMARY KEY,

    UserExtendedID BIGINT NOT NULL,

    IdentityProvider NVARCHAR(50) NOT NULL,

    ExternalTenantID NVARCHAR(100) NOT NULL,

    ExternalUserID NVARCHAR(100) NOT NULL,

    CreatedOn DATETIME2 NOT NULL
      DEFAULT SYSUTCDATETIME(),
```

```
    CONSTRAINT FK_ExternalIdentityMapping_UserExtended
      FOREIGN KEY (UserExtendedID)
      REFERENCES dbo.UserExtended(Id),

    CONSTRAINT UQ_ExternalIdentityMapping
      UNIQUE
      (
        IdentityProvider,
        ExternalTenantID,
        ExternalUserID
      )
  );
END;
```

Step 2.3: Update User Extended Table To accommodate the Entra Email

Add PersonalEmail Column

```
IF COL_LENGTH('dbo.UserExtended', 'PersonalEmail') IS NULL
BEGIN
  ALTER TABLE dbo.UserExtended
  ADD PersonalEmail NVARCHAR(255) NULL;
END;
```

Move Existing Email to PersonalEmail

```
UPDATE dbo.UserExtended

SET PersonalEmail = Email

WHERE PersonalEmail IS NULL

  AND Email IS NOT NULL;
```

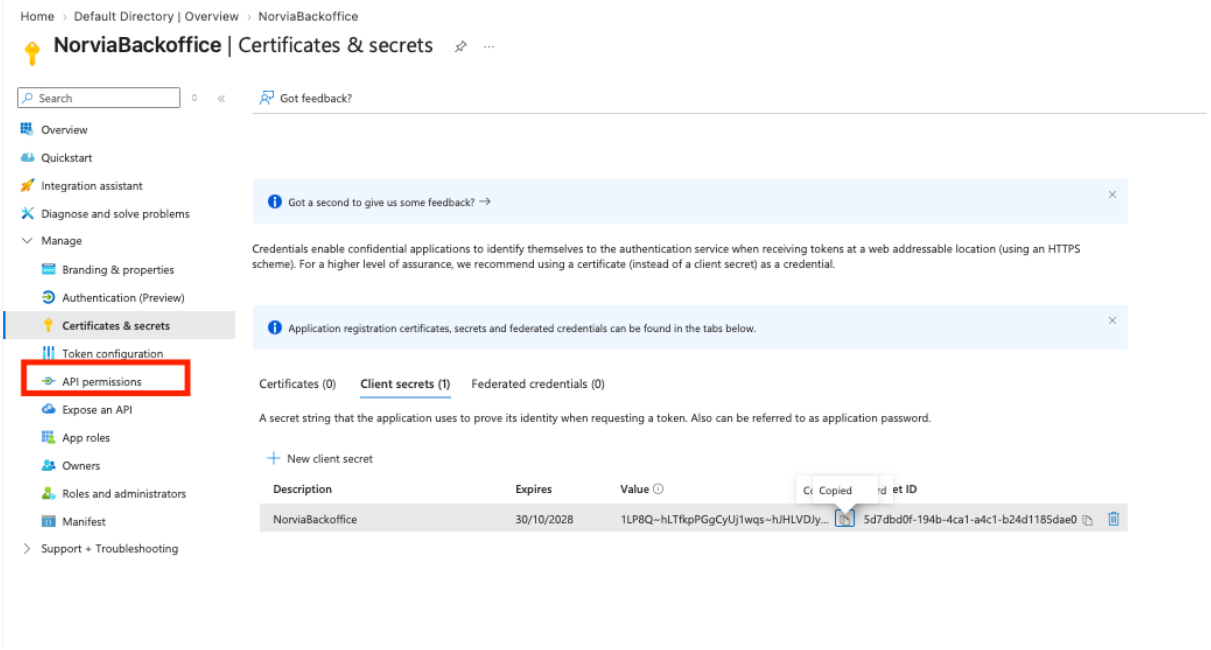
STEP 3: Configure Microsoft Graph API

We need to allow Norvia Backoffice to create and manage users in Microsoft Entra ID.

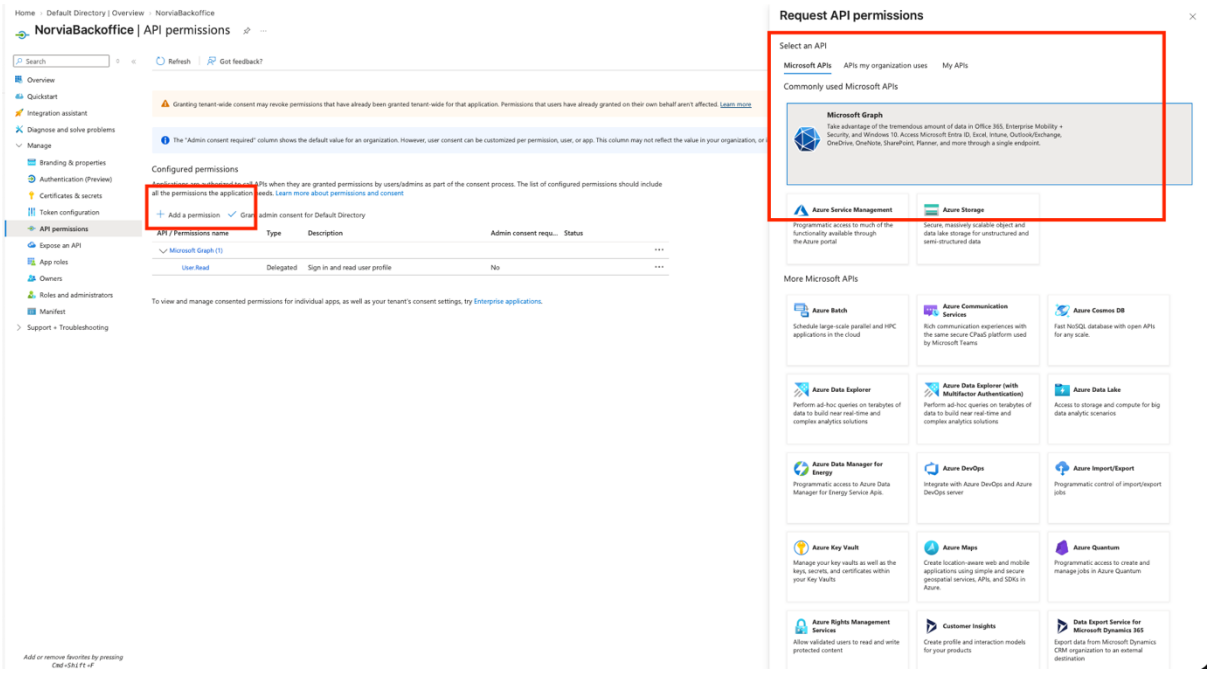
- Open Microsoft Entra ID
- Navigate to App registration
- Click New Registration
- Enter the following information
- Name: NorviaBackoffice
  Supported account Type: Single Tenant only - Default Directory
  Redirect URI: Leave Empty
- Click Register
- Copy the Application ID (Client Id)
- Copy the TenantID
- Create Client Secret for this new app
- Copy the Client secret

STEP 4: Configure Microsoft Graph Permission

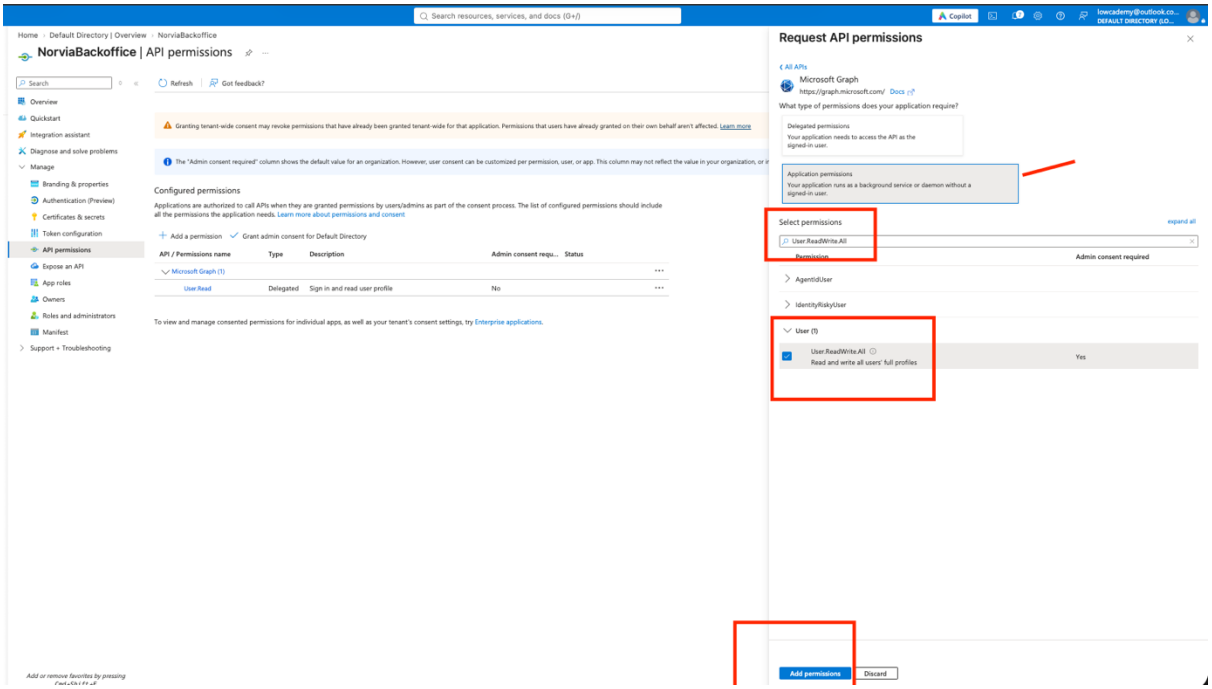
- Open API Permissions under Norviabackoffice App in MS Entra



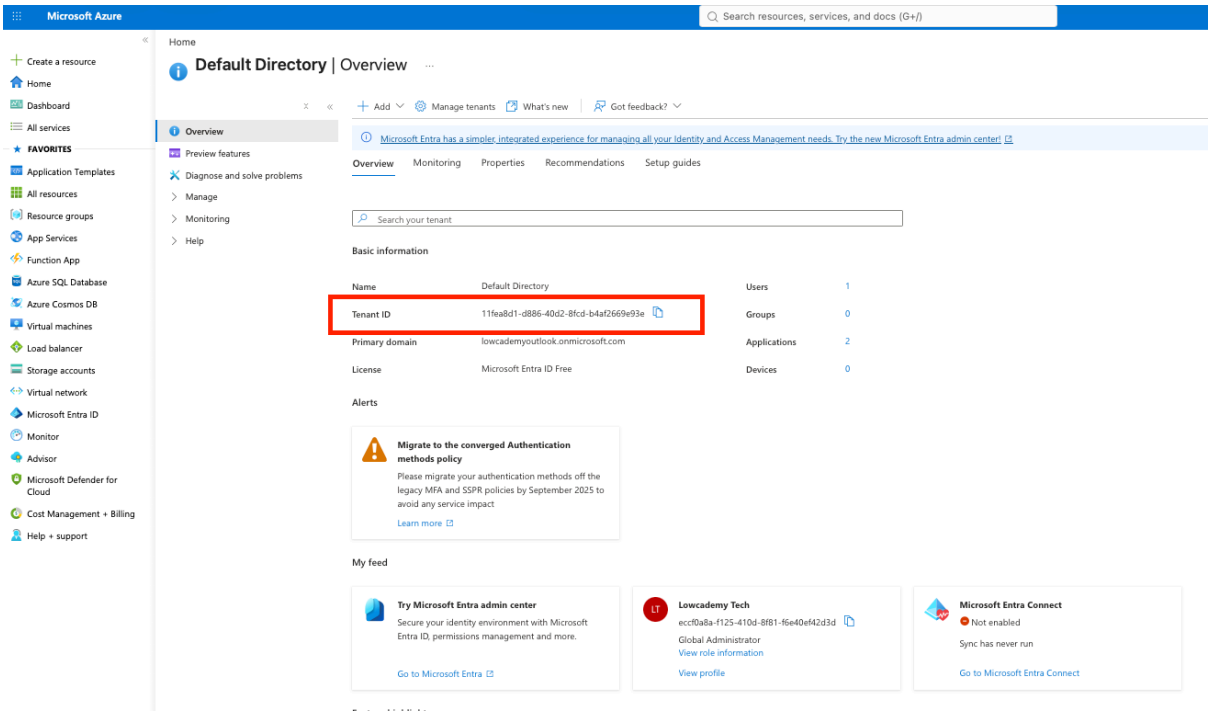
- Click Add Permission
- Select Microsoft Graph

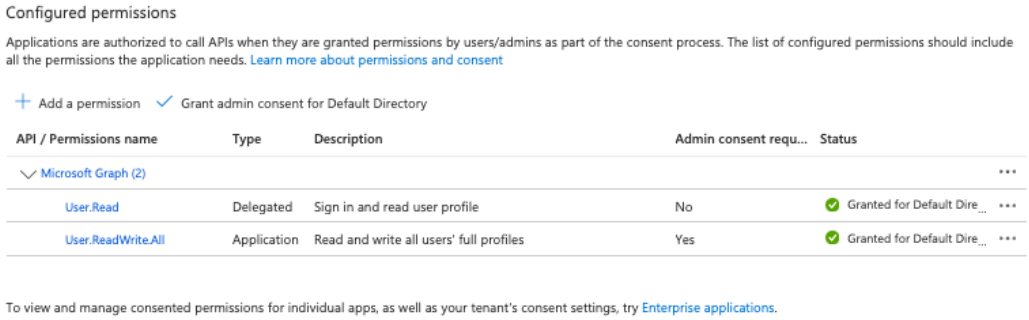


- Select Application Permissions.
- Search for User.ReadWrite.All
- Add the permissions.



- Click on Grant Admin consent



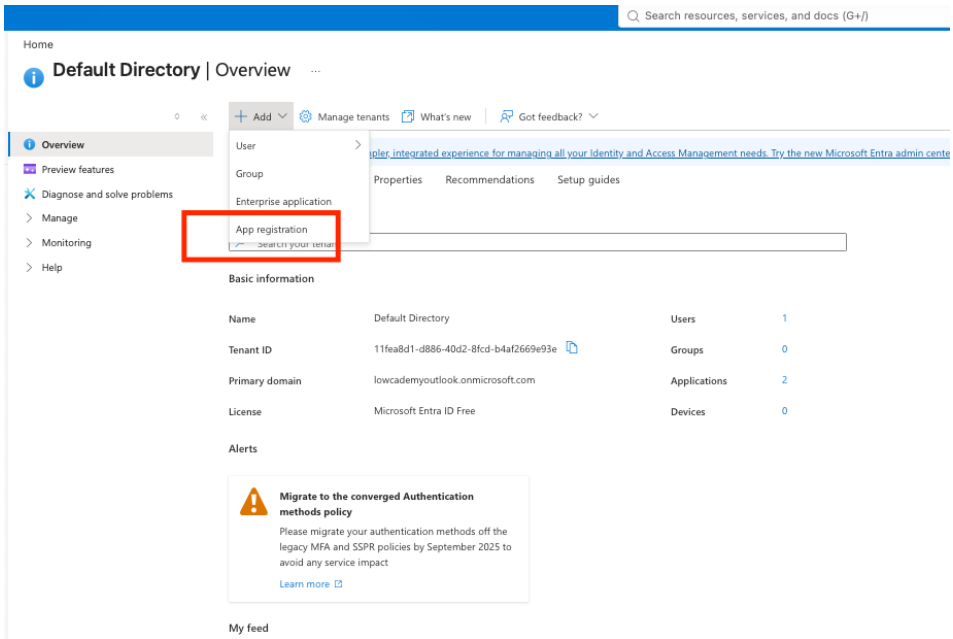


This permission allows the application to create and manage users in the directory, subject to Microsoft's permission and role restrictions.

It is a powerful permission, so only trusted Norvia Backoffice administrators should be allowed to initiate provisioning.

STEP 5: Configure Microsoft Graph Credentials in Norvia Backoffice.

- Open Azure Portal
- Nvaigate to App Services - Norvia Backoffice - Settings - Environment Variables



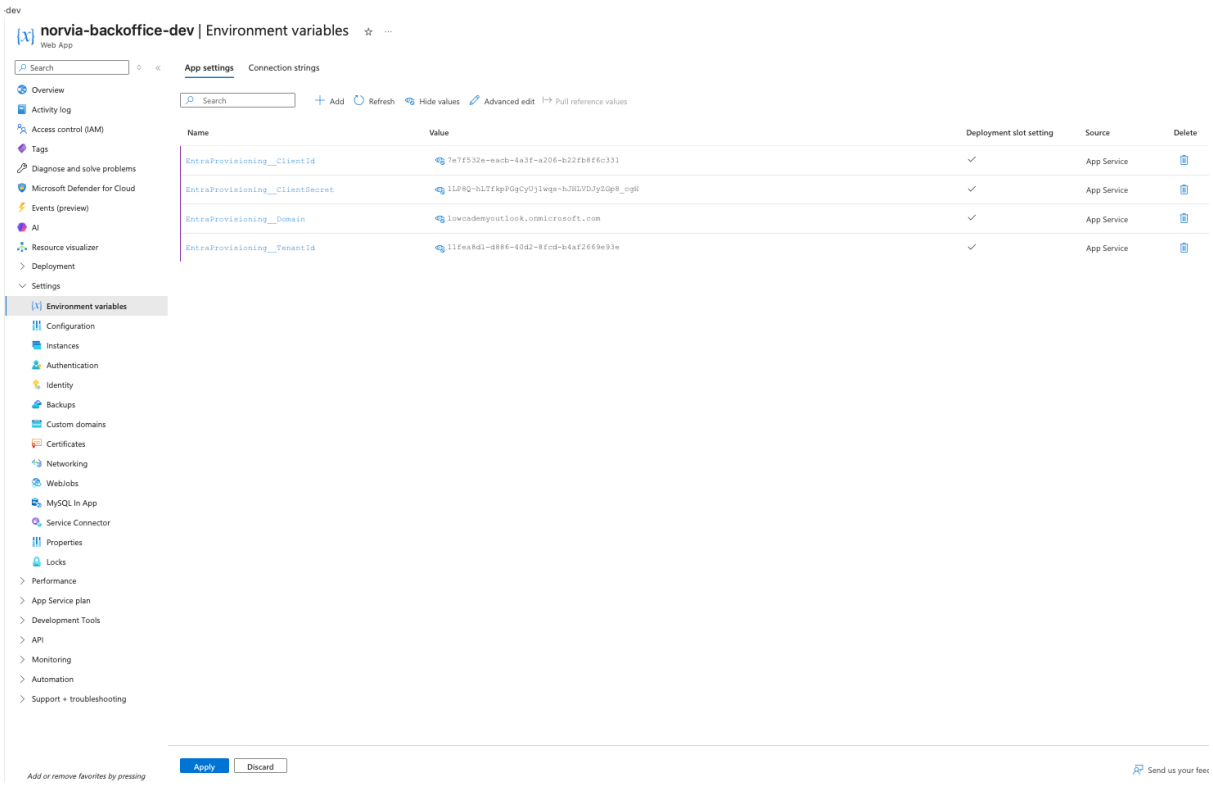
- Add the Following Configuration

EntraProvisioning__TenantId : Your Tenant ID

EntraProvisioning__ClientId: Norvia Backoffice Client ID/App ID

EntraProvisioning__ClientSecret: Norvia Backoffice Secret

EntraProvisioning__Domain: yourtenant.onmicrosoft.com (Primary Domain of Entra Account)



Click on Apply

STEP 6: Make code changes in NorviaBackoffice

Use Below Prompt to Update the Code

========== Prompt Begin ==========

TASK: Update Existing Norvia Backoffice Application for Microsoft Entra ID

Project Context

Norvia Backoffice is an existing, fully developed ASP.NET Core application.

It already supports:

- Tenant Management
- Product Management
- Tenant Administrator Creation
- Tenant Administrator Updates
- Product and Role Assignments

Two applications use the same Azure SQL database:

1. Intelli Campus — OutSystems ODC
2. Intelli Broadcast — ASP.NET Core

We want to integrate Microsoft Entra ID into the **existing Product Management and Tenant Administrator Create/Update flows**.

IMPORTANT INSTRUCTIONS

- Do not rebuild the application.
- Do not create new modules or management screens.
- Do not change Tenant Management.
- Do not modify unrelated functionality.
- Do not change the existing application architecture.
- Reuse existing controllers, services, repositories, and UI components.
- Keep the implementation simple.
- This is a development/training application.

Implement changes only in the specified functionality.

STEP 1: Configure Microsoft Graph Integration

First, analyze the existing application structure.

Identify how services, database operations, and configuration are currently implemented.

Add Microsoft Graph integration using the existing architecture.

Use Microsoft Graph SDK or HTTP client.

Application Configuration

Read the following settings:

```
{
  "EntraProvisioning": {
    "TenantId": "",
    "ClientId": "",
    "ClientSecret": "",
    "Domain": ""
  }
}
```

These settings are already configured in Azure App Service environment variables.

The `Domain` value contains the verified Microsoft Entra primary domain.

Example:

```
lowcademy.onmicrosoft.com
```

Use this domain to generate Microsoft login usernames.

Microsoft Graph Operations

Implement reusable methods for:

1. Creating a Microsoft Entra user.
2. Checking whether a Microsoft username already exists.
3. Retrieving a Microsoft Entra user.
4. Updating a Microsoft Entra user's name.
5. Resetting a Microsoft Entra user's password.

Use server-side Microsoft Graph authentication.

Do not hardcode credentials or expose them to the browser.

STEP 2: Update Existing Product Management Flow

Locate the existing:

- Create Product functionality.
- Edit Product functionality.

Do not redesign these screens.

The `NorviaProducts` table already contains:

- `SSOTenantID`
- `SSOAppID`
- `SSOAppSecretReference`

Required Changes

Add three fields to the existing Product Create/Edit form:

1. SSO Tenant ID

Maps to:

`NorviaProducts.SSOTenantID`

2. SSO Client ID

Maps to:

`NorviaProducts.SSOAppID`

3. SSO Client Secret Reference

Maps to:

`NorviaProducts.SSOAppSecretReference`

Save these values using the existing product-management logic.

Load them when editing an existing product.

Keep these fields optional.

Do not modify existing ProductHash functionality.

The secret reference should identify an application setting containing the actual secret, not the secret value itself.

Do not modify other product-management functionality.

STEP 3: Update Existing Tenant Administrator Creation Flow

Locate the existing **Create Tenant Administrator** functionality.

The existing form contains:

- Tenant
- First Name
- Surname
- Email
- Phone Number
- Address
- Product Access
- Password
- Retype Password
- Active Administrator

We need to modify this existing form.

3.1 Update Email Fields

The `UserExtended` table already contains:

- `Email`
- `PersonalEmail`

Personal Email

Rename the existing Email field to:

Personal Email

Bind it to:

```
UserExtended.PersonalEmail
```

The administrator will enter this value.

Example:

`john@gmail.com`

Dummy personal email addresses must also be accepted.

Login Email

Add a new read-only field:

Login Email

Bind it to:

`UserExtended.Email`

This field will contain the Microsoft Entra login username.

Example:

`john.smith@lowcademy.onmicrosoft.com`

The administrator must not be able to edit this field manually.

3.2 Automatically Generate Login Email

When the administrator enters First Name and Surname:

Generate the login email automatically.

Use:

`EntraProvisioning:Domain`

Example

First Name: John

Surname: Smith

Generated Login Email:

`john.smith@lowcademy.onmicrosoft.com`

Duplicate Checking

Check whether the generated username already exists in:

1. `UserExtended.Email`
2. Microsoft Entra ID

If the username exists, append an incremental number.

Example:

- john.smith@lowcademy.onmicrosoft.com
- john.smith2@lowcademy.onmicrosoft.com
- john.smith3@lowcademy.onmicrosoft.com

Continue until an available username is found.

Display the generated email in the read-only Login Email field.

Recheck availability when the administrator submits the form.

3.3 Update Existing User Creation Logic

When the administrator clicks **Create Administrator**, execute the following process.

Step 1: Validate Existing Form

Use the existing validation logic.

Validate the password against Microsoft Entra requirements.

Step 2: Generate Login Email

Generate a unique Microsoft login email.

Step 3: Create Microsoft Entra User

Call Microsoft Graph API.

Create the user using:

- First Name
- Surname
- Display Name
- Generated Login Email
- Administrator-entered Password
- Active Status

Use the same password entered by the administrator as the initial Microsoft Entra password.

Set:

```
forceChangePasswordNextSignIn = false
```

where supported by Microsoft Entra policies.

Step 4: Save UserExtended

Continue using the existing UserExtended creation logic.

Store:

```
UserExtended.Email = Generated Login Email
```

```
UserExtended.PersonalEmail = Administrator-entered Personal Email
```

Preserve all other existing fields and behavior.

If the application uses local password authentication, preserve its existing secure password-hashing behavior. Do not store plaintext passwords.

Step 5: Create Identity Mapping

After Microsoft Entra user creation, retrieve its Object ID.

Create a record in the existing `ExternalIdentityMapping` table.

Save:

- UserExtendedID
- IdentityProvider = MicrosoftEntra
- ExternalTenantID = Microsoft Entra Tenant ID
- ExternalUserID = Microsoft Entra Object ID
- CreatedOn = Current UTC timestamp

Step 6: Assign Product Roles

Continue using the existing `UserExtendedRoleMapping` functionality.

Do not change the existing role-assignment logic.

Step 7: Display Result

Display the existing success message.

Also show the generated Microsoft Login Email.

Do not display the password after creation.

STEP 4: Update Existing Tenant Administrator Edit Flow

Locate the existing **Edit Tenant Administrator** functionality.

Do not redesign the screen.

4.1 Update Email Fields

Display:

Personal Email

Editable.

Bound to:

`UserExtended.PersonalEmail`

Login Email

Read-only.

Bound to:

`UserExtended.Email`

Do not regenerate Login Email when editing an existing user.

4.2 Update Microsoft Entra User Information

When the administrator updates the user's First Name or Surname:

1. Update UserExtended using existing logic.
2. Retrieve the Microsoft Entra Object ID from ExternalIdentityMapping.
3. Update the Microsoft Entra user's name using Microsoft Graph.

Do not change the Microsoft login username.

When Personal Email changes:

Update `UserExtended.PersonalEmail` only.

Do not modify the Microsoft login username.

4.3 Update Password

Reuse the existing password fields in the Edit Tenant Administrator form.

If the administrator enters a new password:

1. Validate the password.
2. Retrieve the Microsoft Entra Object ID.
3. Reset the Microsoft Entra password using Microsoft Graph.
4. Set `forceChangePasswordNextSignIn = false` where supported.
5. Continue the existing user-update process.

If password fields are empty, do not reset the password.

Do not store Microsoft passwords in plaintext.

Preserve existing local password-hashing behavior only if local authentication is still required.

4.4 Update Product Roles

Continue using the existing `UserExtendedRoleMapping` logic.

If another product is assigned to the user:

- Reuse the existing Microsoft Entra account.
- Do not create another Microsoft Entra user.
- Update only the product-role mappings.

If product access is removed:

- Update the existing role mappings.
- Do not delete the Microsoft Entra account.

4.5 Update Active Status

Continue updating:

`UserExtended.IsActive`

Do not automatically disable the Microsoft Entra account.

Application access will be controlled by UserExtended.IsActive and the existing product-role mappings.

STEP 5: Handle Existing Users During Updates

Some users may already exist in UserExtended without Microsoft Entra identity mappings.

When an administrator edits such a user:

1. Check ExternalIdentityMapping.
2. If the Microsoft identity mapping exists, use it.
3. If no mapping exists, allow the administrator to provision the user in Microsoft Entra as part of the existing Edit flow.
4. Generate a unique Microsoft login email.
5. Use the password explicitly entered by the administrator.
6. Create the Microsoft Entra account.
7. Update UserExtended.Email with the generated login email.
8. Preserve UserExtended.PersonalEmail.
9. Create ExternalIdentityMapping.
10. Preserve existing product-role assignments.

Do not automatically provision existing users merely because an unrelated field was edited.

Do not create duplicate Microsoft Entra accounts.

STEP 6: Basic Error Handling

Handle the following errors:

- Microsoft Graph configuration missing.
- Microsoft Graph authentication failure.
- Duplicate Microsoft login email.
- Microsoft Entra password validation failure.
- Microsoft Entra user creation failure.
- Microsoft Entra user update failure.
- Microsoft Entra password-reset failure.
- SQL database operation failure.

Display clear error messages.

Do not expose passwords, client secrets, or access tokens.

If Microsoft Entra creation succeeds but SQL saving fails, report the partial failure and retain the Entra Object ID in a diagnostic log for manual recovery.

Keep error handling simple.

STEP 7: Testing

Test only the modified functionality.

Product Management

1. Create a product with SSO configuration.
2. Edit the product's SSO configuration.
3. Verify values are saved in NorviaProducts.

Tenant Administrator Creation

4. Enter First Name and Surname.
5. Verify Login Email is generated automatically.
6. Verify duplicate username handling.
7. Create a Tenant Administrator.
8. Verify the Microsoft Entra user is created.
9. Verify the administrator-entered password works, subject to Microsoft policies.
10. Verify UserExtended.Email contains the generated login email.
11. Verify UserExtended.PersonalEmail contains the personal email.
12. Verify ExternalIdentityMapping is created.
13. Verify existing product-role assignments work.

Tenant Administrator Update

14. Update First Name or Surname.
15. Verify Microsoft Entra display name is updated.
16. Update Personal Email.
17. Verify Login Email remains unchanged.
18. Update the password.
19. Verify the Microsoft Entra password is updated.
20. Assign another product to the same user.
21. Verify no duplicate Microsoft account is created.
22. Verify existing product-role functionality remains unchanged.

Existing Users

23. Edit an existing user without an Entra mapping.

24. Explicitly provision the user in Microsoft Entra.
25. Verify the mapping is created.
26. Verify existing product roles remain unchanged.

FINAL INSTRUCTIONS

This application is already developed.

Only update these existing flows:

1. Product Create/Edit.
2. Tenant Administrator Create.
3. Tenant Administrator Edit.

Add Microsoft Graph integration only to support these flows.

Do not modify Tenant Management.

Do not create new management screens.

Do not modify unrelated application functionality.

Do not create or alter database tables.

Do not generate migrations.

Do not introduce unnecessary infrastructure.

Reuse the existing application architecture and UI.

Keep the changes minimal and focused.

Build the application and resolve compilation errors.

Provide a short summary of modified files and instructions for testing the three updated flows.

========== Prompt End ==========

STEP 6: Perform Testing and Deploy the App