Azure App Registration for Teams and Skype Presence Synchronization

In order to be able to synchronize user presence with Microsoft Teams and Skype for Business an Application must be created in Azure.

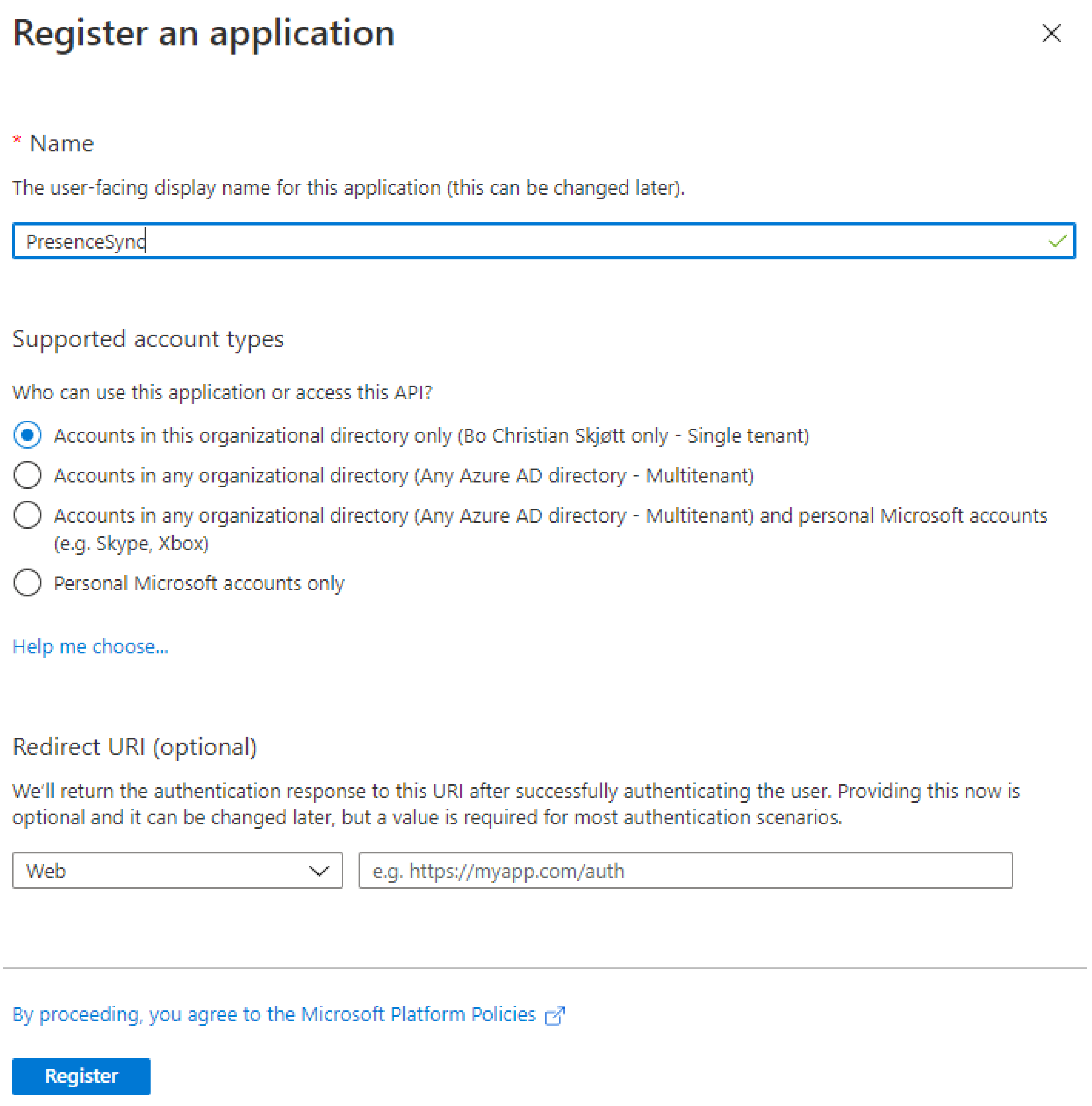
Your Azure Active Directory must be of type **Azure AD for Office 365** and must have an Office 365 Enterprise license at level E1 or higher.

App creation in Azure portal

Go to your Active Directory in the Azure portal and follow the steps below.

Step 1. Register an App

Click App registrations, New registration.



Register an application

* Name

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

PresenceSync

Supported account types

Who can use this application or access this API?

- [✓ Accounts in this organizational directory only (Bo Christian Skjøtt 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 | e.g. https://myapp.com/auth

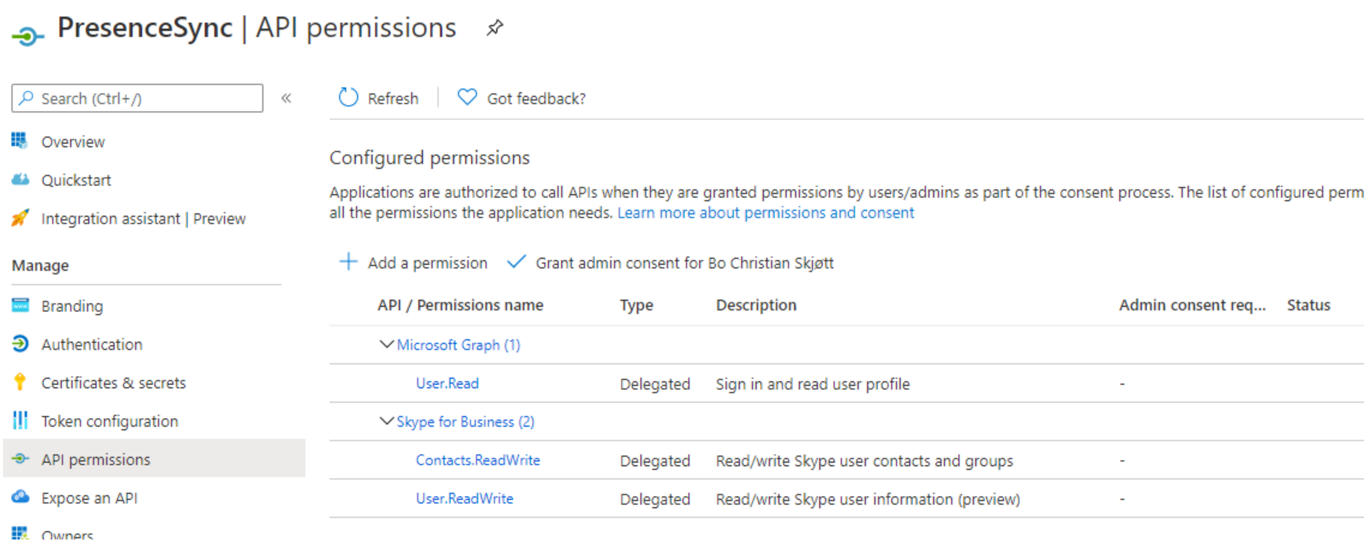
By proceeding, you agree to the Microsoft Platform Policies

Click the Register button

Step 2. Add API permissions

Select **API permissions** and add these **Skype for Business** permissions of type **Delegated permissions**:

- Contacts.ReadWrite
- User.ReadWrite



Configured permissions

Applications are authorized to call APIs when they are granted permissions by users/admins as part of the consent process. The list of configured perm all the permissions the application needs. Learn more about permissions and consent

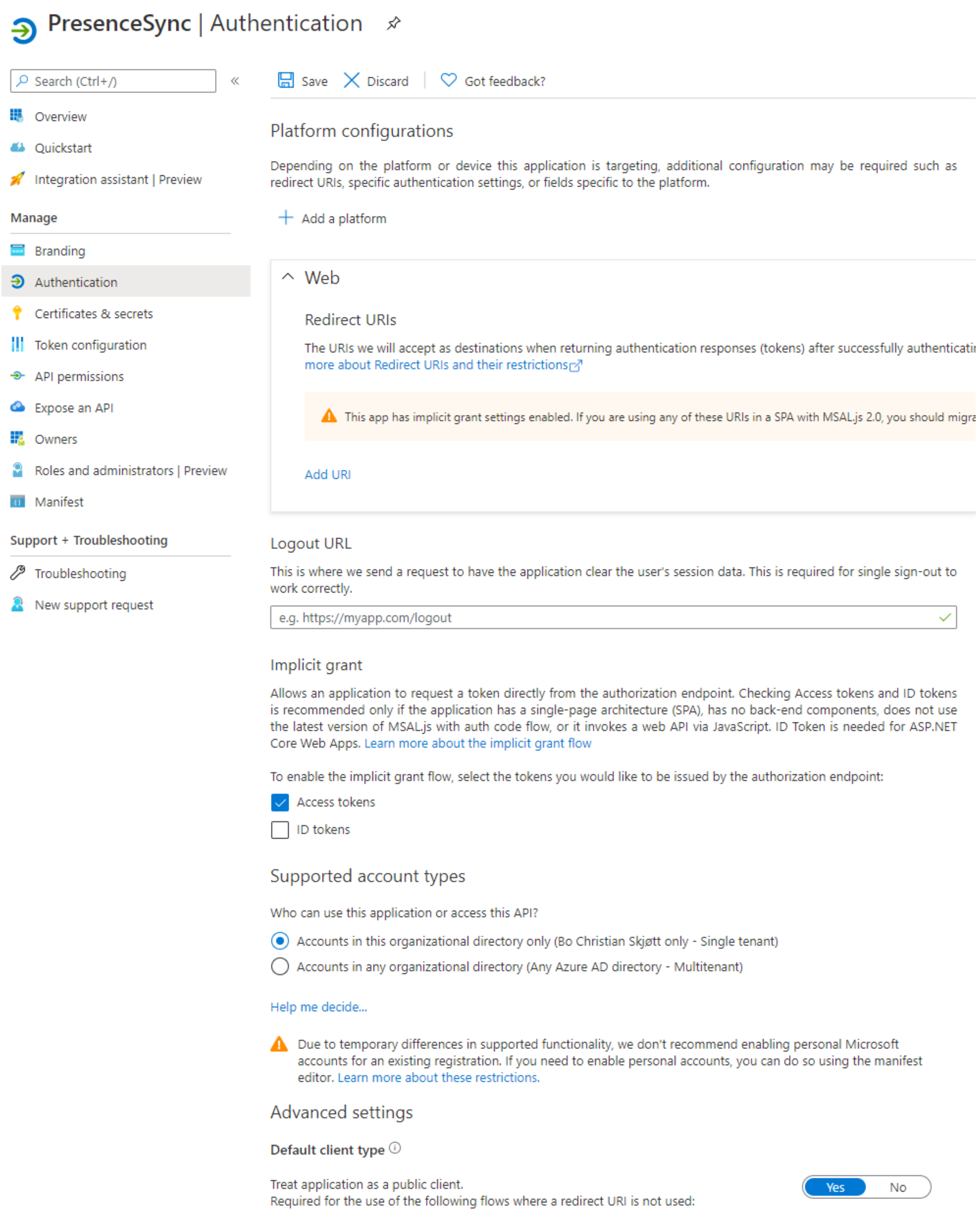
API / Permissions name	Type	Description	Admin consent req...	Status
Microsoft Graph (1)				
User.Read	Delegated	Sign in and read user profile	-	
Skype for Business (2)				
Contacts.ReadWrite	Delegated	Read/write Skype user contacts and groups	-	
User.ReadWrite	Delegated	Read/write Skype user information (preview)	-	

Click on the **"Grant admin consent for ..."** button.

Step 3. Configure Authentication

Select **Authentication** in the menu.

Make selections as shown in this screenshot



PresenceSync | Authentication

Platform configurations

Depending on the platform or device this application is targeting, additional configuration may be required such as redirect URIs, specific authentication settings, or fields specific to the platform.

+ Add a platform

Web

Redirect URIs

The URIs we will accept as destinations when returning authentication responses (tokens) after successfully authenticati
more about Redirect URIs and their restrictions

This app has implicit grant settings enabled. If you are using any of these URIs in a SPA with MSAL.js 2.0, you should migra

Add URI

Logout URL

This is where we send a request to have the application clear the user's session data. This is required for single sign-out to work correctly.

e.g. https://myapp.com/logout

Implicit grant

Allows an application to request a token directly from the authorization endpoint. Checking Access tokens and ID tokens is recommended only if the application has a single-page architecture (SPA), has no back-end components, does not use the latest version of MSAL.js with auth code flow, or it invokes a web API via JavaScript. ID Token is needed for ASP.NET Core Web Apps. Learn more about the implicit grant flow

To enable the implicit grant flow, select the tokens you would like to be issued by the authorization endpoint:

- [✓] Access tokens
- [ ] ID tokens

Supported account types

Who can use this application or access this API?

- ● Accounts in this organizational directory only (Bo Christian Skjøtt only - Single tenant)
- ○ Accounts in any organizational directory (Any Azure AD directory - Multitenant)

Help me decide...

Due to temporary differences in supported functionality, we don't recommend enabling personal Microsoft accounts for an existing registration. If you need to enable personal accounts, you can do so using the manifest editor. Learn more about these restrictions.

Advanced settings

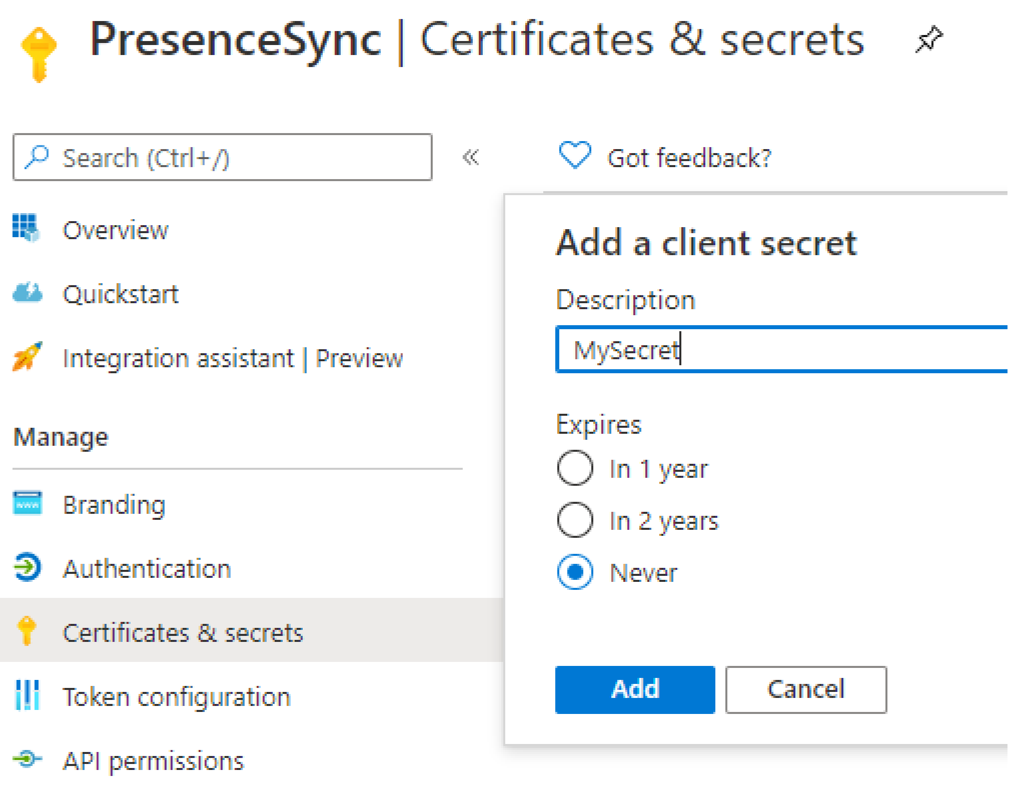
Default client type

Treat application as a public client.
Required for the use of the following flows where a redirect URI is not used:

[✓ Yes] No

Step 4. Add a Client Secret

Select **Certificates and secrets** in the menu. Click **New client secret**.



Add a client secret

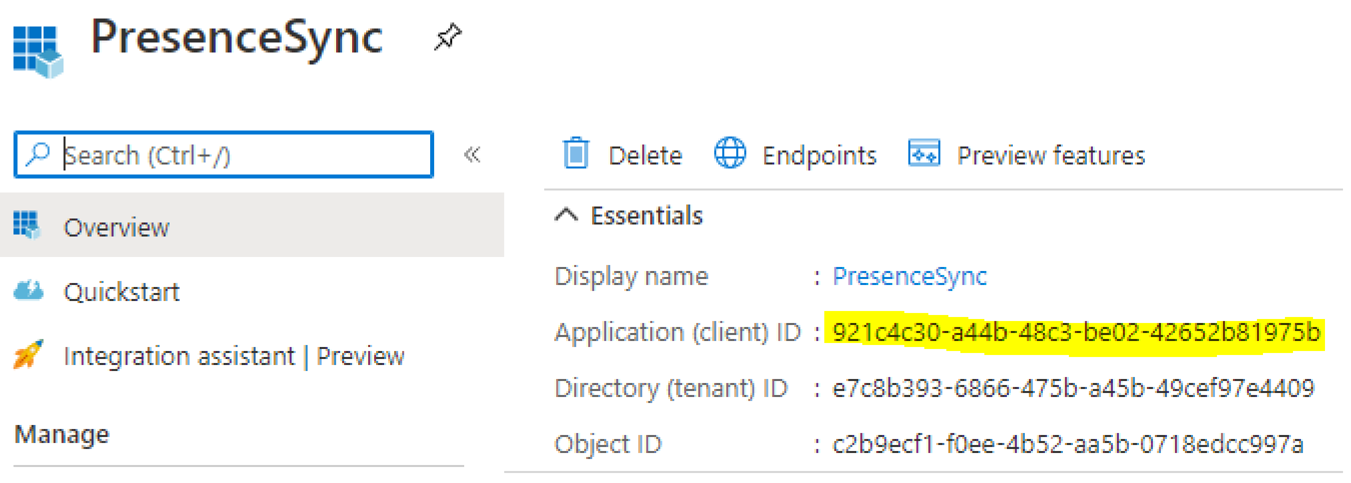
Description: MySecret

Expires: In 1 year / In 2 years / [✓ Never]

Take a copy of the generated secret. It is only shown during creation. You need to enter the secret in the Novus Configuration UI.

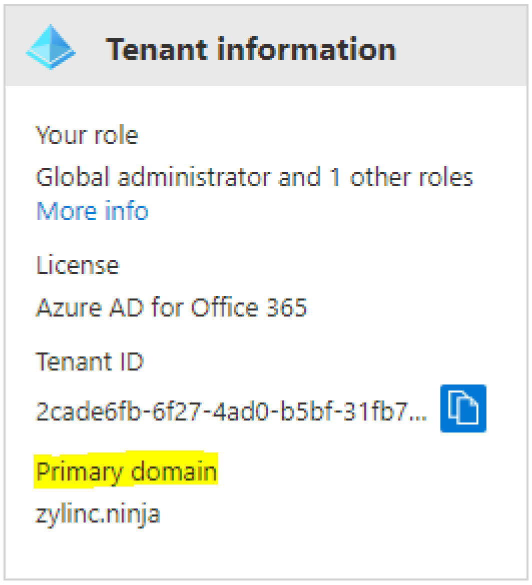
Step 5. Get the Client ID and Tenant ID for the application

Go to the Application Overview page and copy the Client ID. You need to enter this ID in the Novus Configuration UI.



Display name	: PresenceSync
Application (client) ID	: 921c4c30-a44b-48c3-be02-42652b81975b
Directory (tenant) ID	: e7c8b393-6866-475b-a45b-49cef97e4409
Object ID	: c2b9ecf1-f0ee-4b52-aa5b-0718edcc997a

Go to the Active Directory overview page and copy the **Primary domain** value.



Tenant information

Your role
Global administrator and 1 other roles
More info

License
Azure AD for Office 365

Tenant ID
2cade6fb-6f27-4ad0-b5bf-31fb7...

Primary domain
zylinc.ninja

This is the value you need to enter as **Tenant ID** in the Novus Configuration UI.

Step 6. Create a user account for synchronization

The UCWA API used for presence synchronization requires a user account that has an Office 365 license. This account is called the sync user and the username (email) and password of this user must be configured in the Novus Configuration UI.

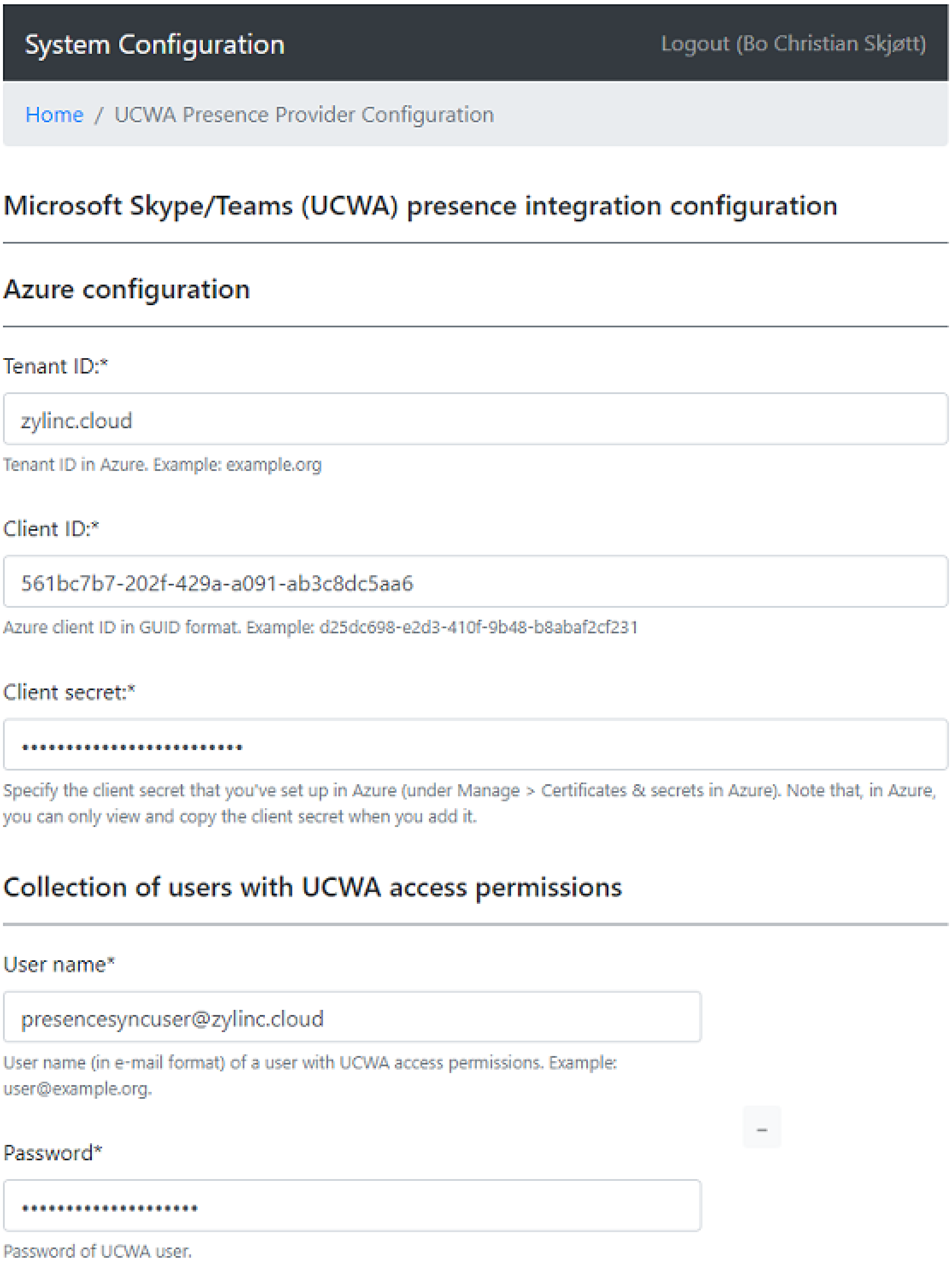
The sync user must have multi-factor authentication disabled.

Step 7. Make sync user owner of the application

1. Go to the Application you created above.
2. Click on **Owners**.
3. Click **Add owners** and add the sync user as owner.

Step 8. Configure Teams/Skype Presence in Novus Configuration UI

In the Novus Configuration UI go to **UCWA Presence Provider Configuration** and enter the values you have copied from the previous steps.



System Configuration

Logout (Bo Christian Skjøtt)

Home / UCWA Presence Provider Configuration

Microsoft Skype/Teams (UCWA) presence integration configuration

Azure configuration

Tenant ID:*

zylinc.cloud

Tenant ID in Azure. Example: example.org

Client ID:*

561bc7b7-202f-429a-a091-ab3c8dc5aa6

Azure client ID in GUID format. Example: d25dc698-e2d3-410f-9b48-b8abaf2cf231

Client secret:*

••••••••••••••••••••••••

Specify the client secret that you've set up in Azure (under Manage > Certificates & secrets in Azure). Note that, in Azure, you can only view and copy the client secret when you add it.

Collection of users with UCWA access permissions

User name*

presencesyncuser@zylinc.cloud

User name (in e-mail format) of a user with UCWA access permissions. Example: user@example.org.

Password*

••••••••••••••••••••

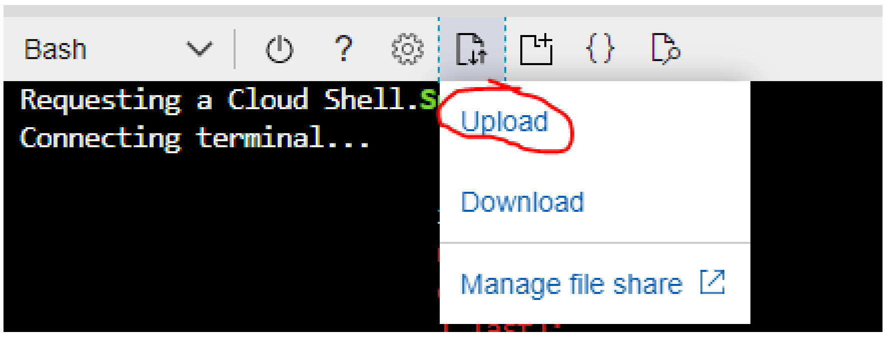
Password of UCWA user.

App creation using Azure CLI commands

The steps above can also be done with the Azure CLI commands below.

Copy the **requiredResourceAccess.json** file to the Azure storage (clouddrive) used by Azure CLI.

If you are using the **Cloud Shell** in the Azure Portal then you can click on the Upload File button in its menubar as shown below



```
Requesting a Cloud Shell.S
Connecting terminal...
```

Create an application with this command

```
az ad app create \
--display-name PresenceSync \
--password VerySecretWord#1234 \
--end-date 2100-12-31 \
--required-resource-accesses requiredResourceAccess.json
```

Replace the password with your choice.

Grant admin consent for the requested API permissions with this command

```
az ad app permission admin-consent --id 00000000-0000-0000-0000-000000000000
```

where 00000000-0000-0000-0000-000000000000 must be replaced with the actual ID of the application created above.

(Note: the **az ad app permission admin-consent** fails with an exception)