# The Tenant Management Application

The [TenantManagement](https://github.com/PowerBiDevCamp/TenantManagement/tree/main/TenantManagement) application is a sample .NET 5 application which demonstrates how to manage service principals within a large-scale Power BI embedding environment with 1000's of customer tenants. Let's start by explaining what is meant by a tenant.

If you have worked with Azure AD, the word **"tenant"** might make you think of an Azure AD tenant. However, the concept of a tenant is different for this sample application. In this context, each tenant represents a customer for which you are embedding Power BI reports using the app-owns-data embedding model. In order to manage a multi-tenant environment, you must create a separate tenant for each customer. Provisioning a new customer tenant for Power BI embedding typically involves writing code to create a Power BI workspace, import a PBIX file, patch datasource credentials and start a dataset refresh operation.

The problem that **TenantManagement** application addresses is a Power BI Service limitation which restricts users and service principals from being a member of more than 1000 workspaces. If you are implementing app-owns-data embedding in an application which uses a single service principal, Microsoft will only support you in creating up to 1000 workspaces.

The **TenantManagement** application demonstrates how to work around the 1000 workspace limitation by implementing a service principal pooling scheme. Here is how it works. Each service principal can support up to 1000 workspaces. Therefore, creating a service principal pool of 10 service principals makes it possible to create and manage 10,000 customer tenant workspaces in a fashion that is supported by Microsoft.

In addition to implementing a service principal pooling scheme, the **TenantManagement** application also demonstrates how to create and manage a separate service principal for each customer tenant workspace. An application design which maintains a one-to-one relationship between service principals and customer tenant workspaces is what Microsoft recommends as a best practice because it provides the greatest amount of isolation especially with respect datasource credentials.

You can follow the steps in this document to set up the **TenantManagement** application for testing. To complete these steps, you will require a Microsoft 365 tenant in which you have permissions to create and manage Azure AD applications and security groups. You will also need Power BI Service administrator permissions to configure Power BI settings to give service principals to ability to access the Power BI Service API. If you do not have a Microsoft 365 environment for testing, you can create one for free by following the steps in [Create a Development Environment for Power BI Embedding](https://github.com/PowerBiDevCamp/Camp-Sessions/raw/master/Create%20Power%20BI%20Development%20Environment.pdf).

## Setting up your development environment

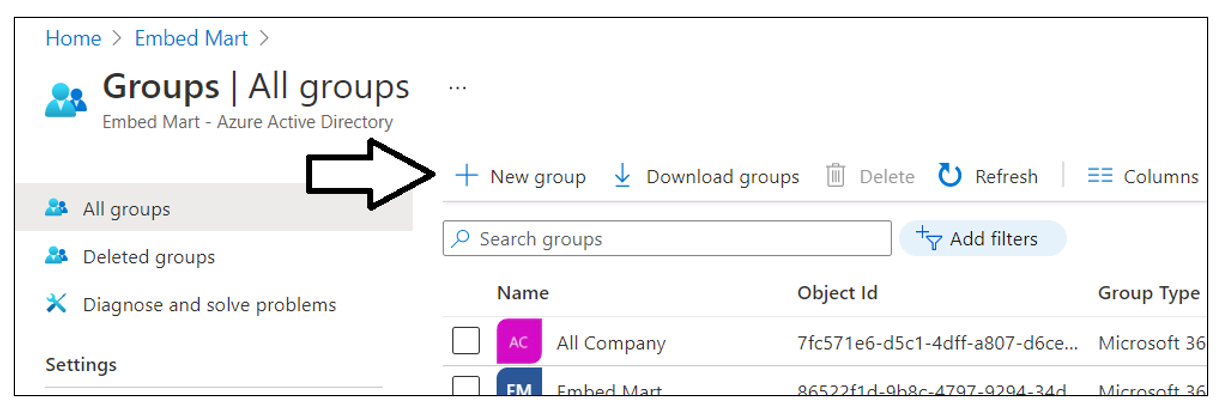
To set up the TenantManagement application doe testing, you will need to configure a Microsoft 365 envviroment with the following tasks.

1. Create a Security Group in Azure AD named Power BI Apps
2. Configure Power BI Tenant-Level Settings for Service Principal Access
3. Create the Azure AD Application for the TenantManagement Application

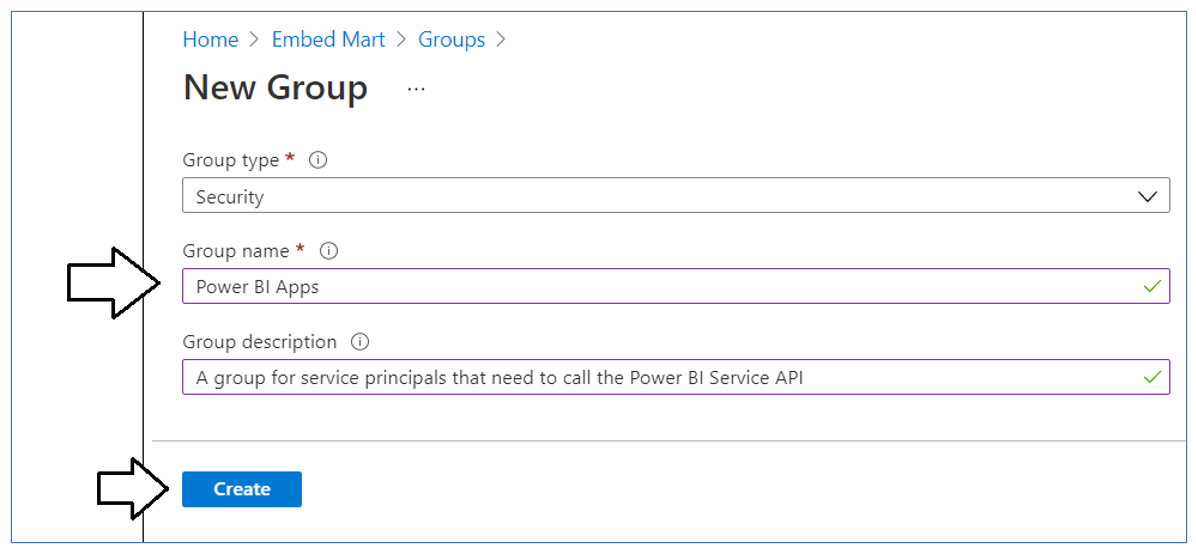
The following three sections will step through each of these setup tasks.

### Create an Azure AD security group named Power BI Apps

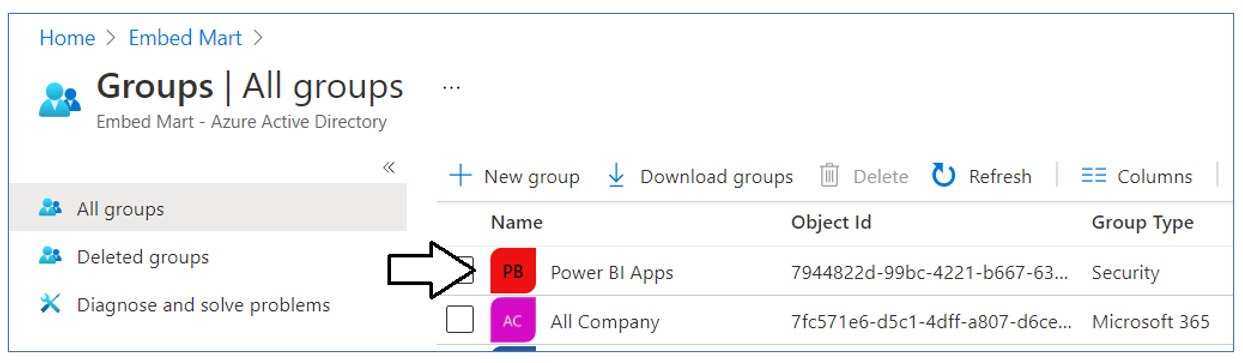
Begin by navigating to the [Groups management page](https://portal.azure.com/#blade/Microsoft_AAD_IAM/GroupsManagementMenuBlade/AllGroups) in the Azure portal. Once you get to the **Groups** page in the Azure portal, click the **New group** link.



In the **New Group** dialog, Select a **Group type** of **Security** and enter a **Group name** of **Power BI Apps**. Click the **Create** button to create the new Azure AD security group



Verify that you can see the new security group named **Power BI Apps** on the Azure portal **Groups** page.



### Configure Power BI tenant-level settings for service principal access

Next, you need you enable a tenant-level setting for Power BI named **Allow service principals to use Power BI APIs**. Navigate to the Power BI Service admin portal at <https://app.powerbi.com/admin-portal>. In the Power BI Admin portal, click the **Tenant settings** link on the left.

Graphical user interface, application

Description automatically generated

Move down in the **Developer settings** section and expand the **Allow service principals to use Power BI APIs** section.

Graphical user interface, application

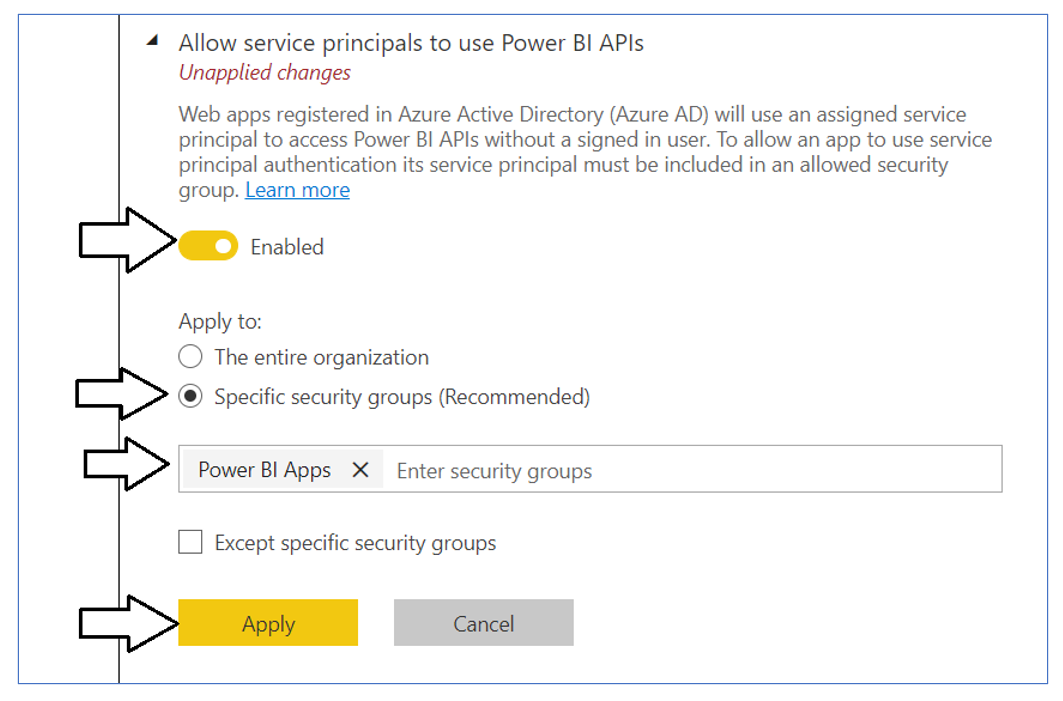
Description automatically generated

Note that the **Allow service principals to use Power BI APIs** setting is initially set to **Disabled**.

Graphical user interface, text, application, email

Description automatically generated

Change the setting to **Enabled**. After that, set the **Apply to** setting to **Specific security groups** and add the **Power BI Apps** security group as shown in the screenshot below. Click the **Apply** button to save your configuration changes.

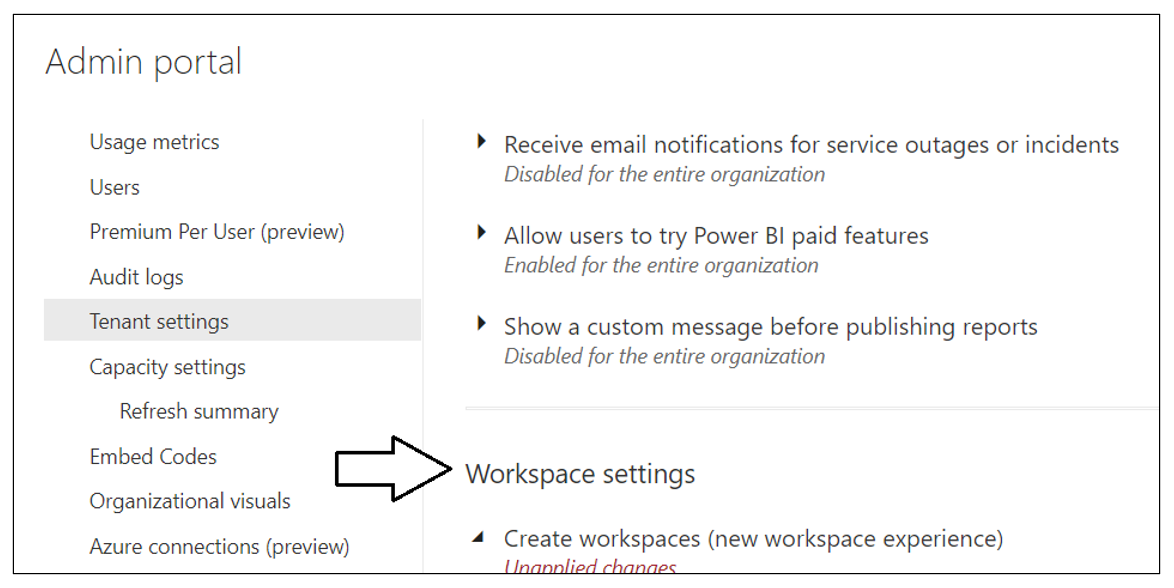


You will see a notification indicating it might take up to 15 minutes to apply these changes to the organization.

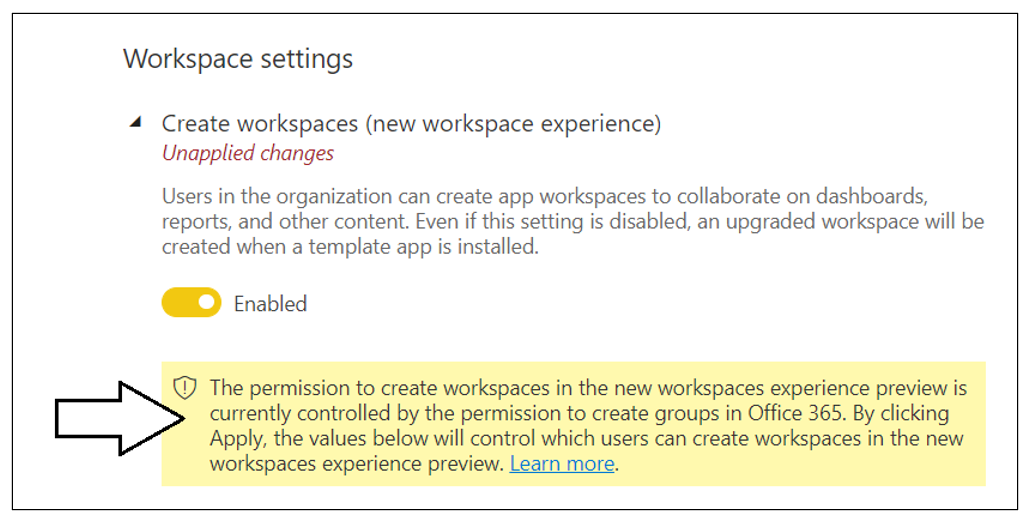
Text

Description automatically generated with medium confidence

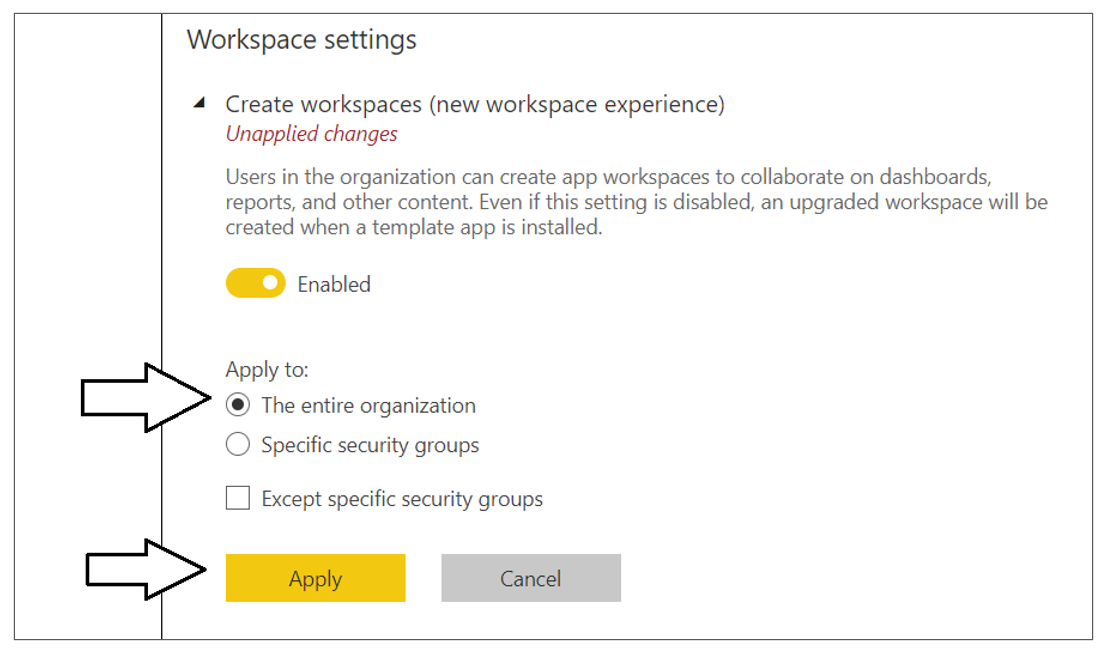
Now scroll upward in the **Tenant setting** section of the Power BI admin portal and locate the **Workspace settings** section.



Note that a new Power BI tenant has an older policy where only users who have the permissions to create Office 365 groups can create new Power BI workspaces. You must reconfigure this setting so that service principals in the **Power BI Apps** group will be able to create new workspaces.



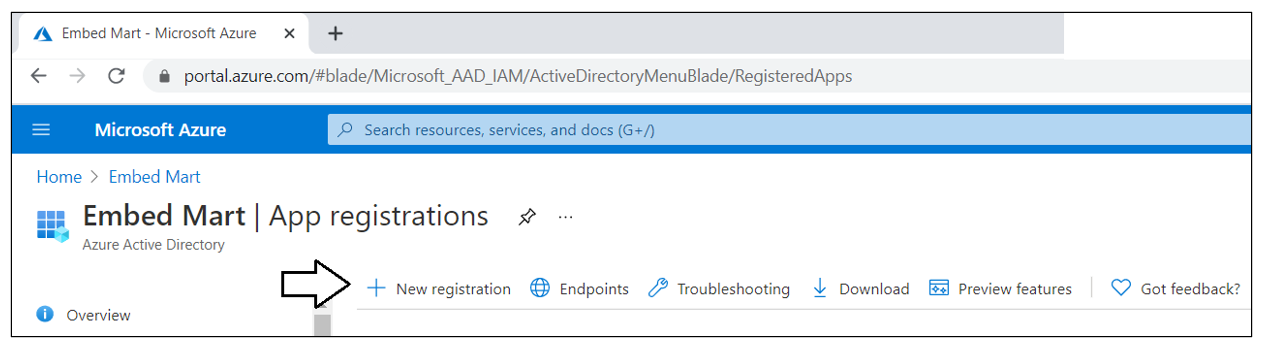
In the **Workspace settings** section, set the **Apply to** setting to **The entire organization**. Click the **Apply** button to save your configuration changes.



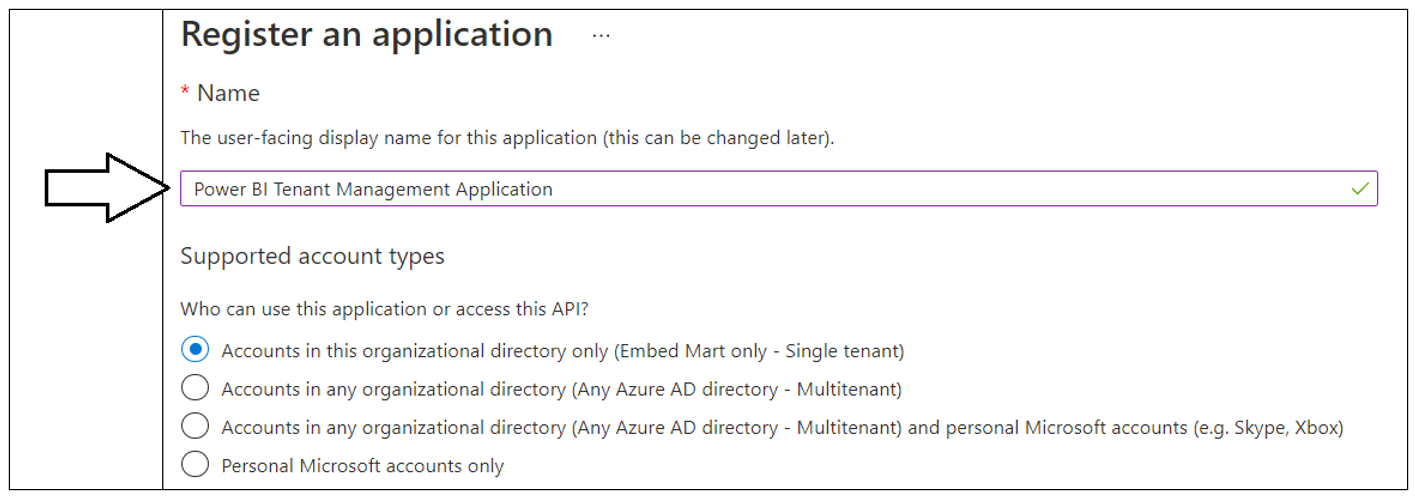
You have now completed the configuration of Power BI tenant-level settings.

### Create the Azure AD Application for the TenantManagement Application

When you login to the Azure portal to create the new Azure AD application, make sure you log in using a user account in the same tenant which contains the Power BI reports you'd like to embed. Begin by navigating to the [App registration](https://portal.azure.com/#blade/Microsoft_AAD_IAM/ActiveDirectoryMenuBlade/RegisteredApps) page in the Azure portal and click the **New registration** link.



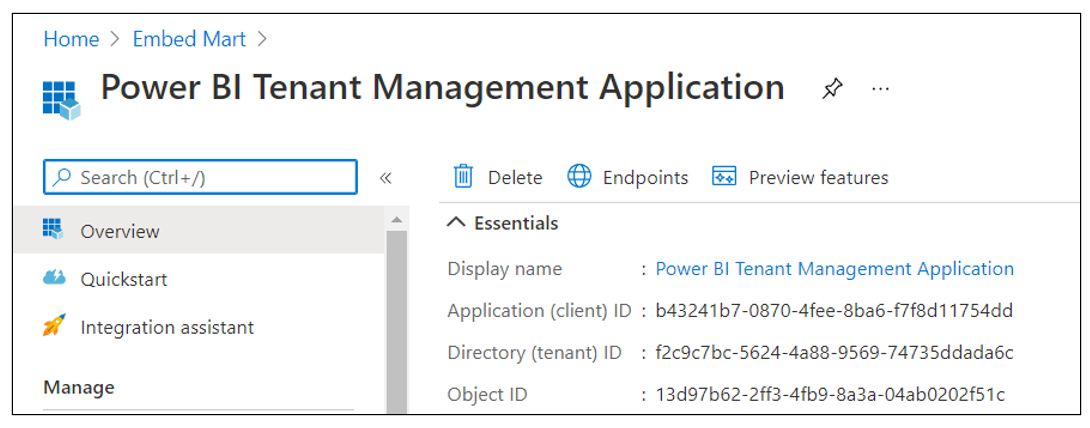
On the **Register an application** page, enter an application name such as **Power BI Tenant Management Application** and accept the default selection for **Supported account types** of **Accounts in this organizational directory only**.



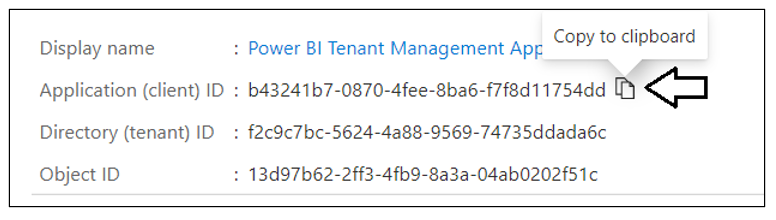
In the **Redirect URI** section leave the default selection of **Web** in the dropdown box. In the textbox to the right of the dropdown, enter a Redirect URI of **https://localhost:44300/signin-oidc**. Click the **Register** button to create the new Azure AD application.



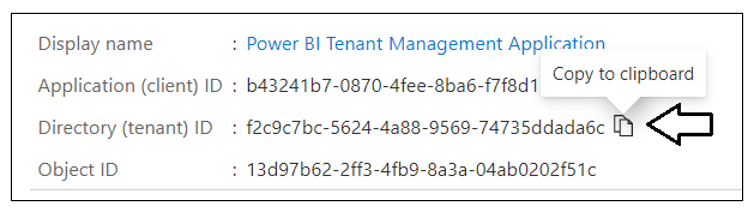
After creating a new Azure AD application in the Azure portal, you should see the Azure AD application overview page which displays the **Application ID**. Note that the ***Application ID*** is often called the ***Client ID***, so don't let this confuse you. You will need to copy this Application ID and store it so you can use it later to configure the project's support for Client Credentials Flow.



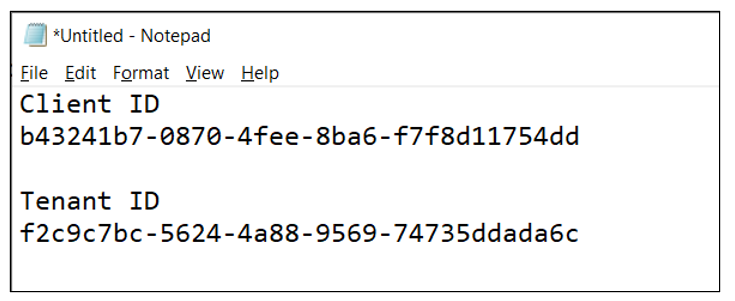
Copy the **Client ID** (aka Application ID) and paste it into a text document so you can use it later in the setup process. Note that this is the **Client ID** value that will be used by **TenantManagement** project to authenticate users.



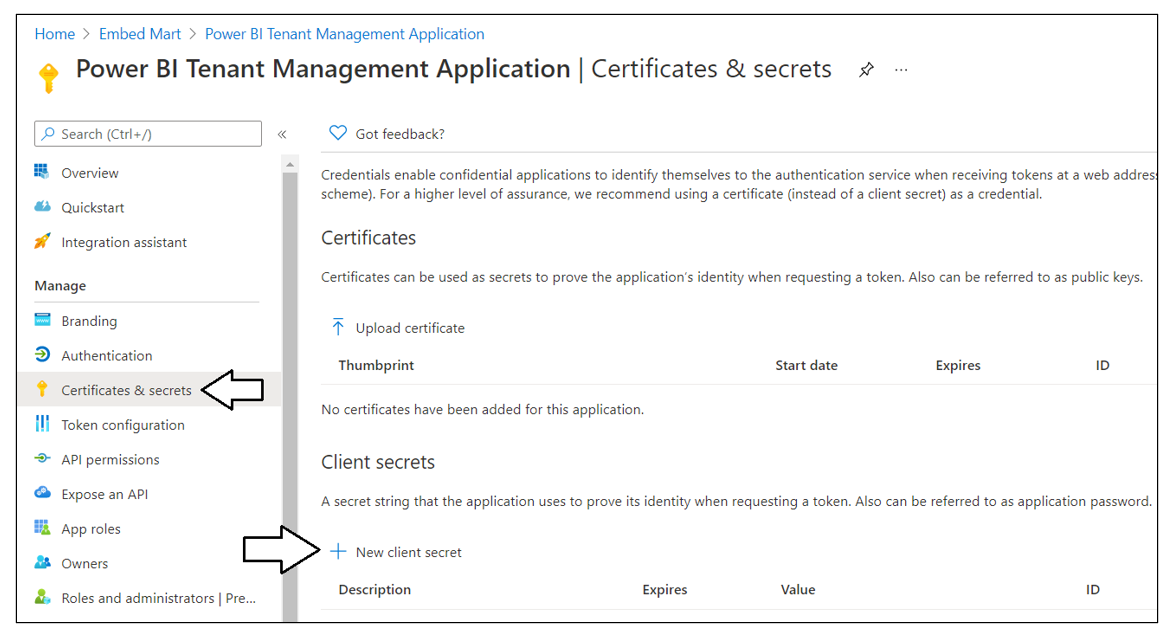
Next, repeat the same step by copying the **Tenant ID** and copying that into the text document as well.



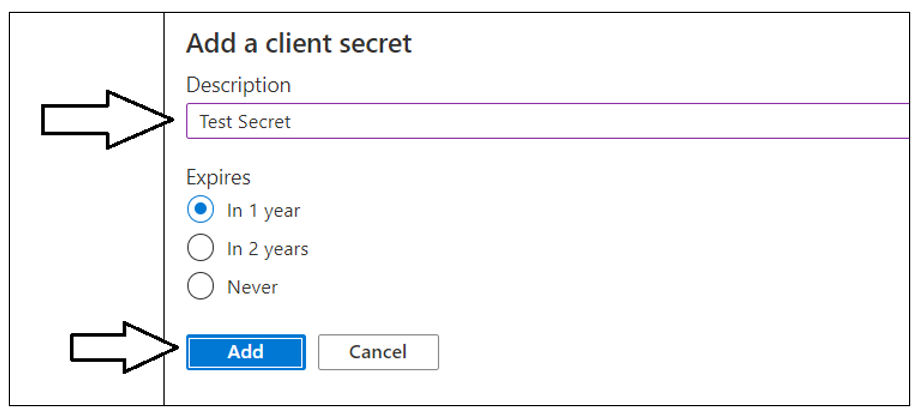
Your text document should now contain the **Client ID** and **Tenant ID** as shown in the following screenshot.



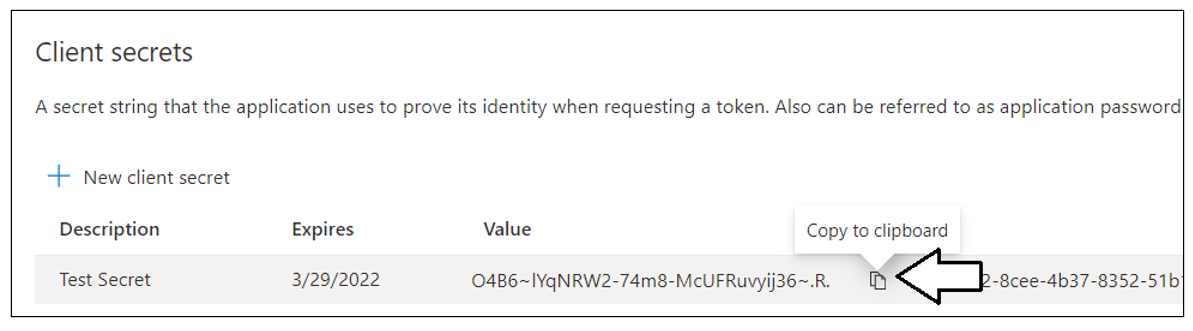
Next, you need to create a Client Secret for the application. Click on the **Certificates & secrets** link in the left navigation to move to the **Certificates & secrets** page. On the **Certificates & secrets** page, click the **New client secret** button as shown in the following screenshot.



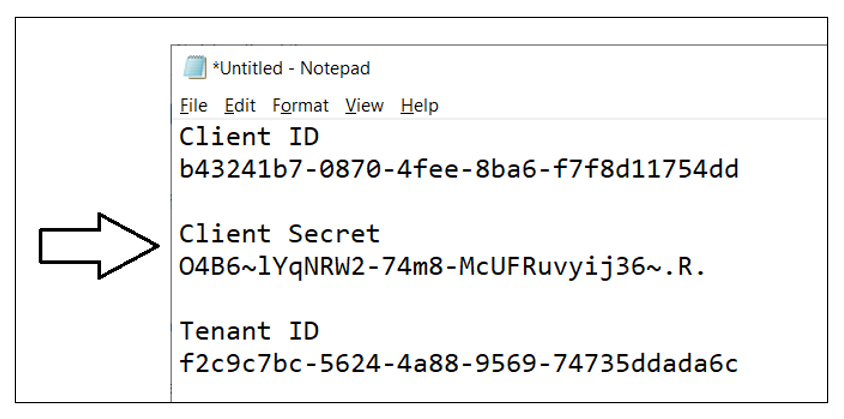
In the **Add a client secret** dialog, add a text description such as **Test Secret** and then click the **Add** button to create the new Client Secret.



Once you have created the Client Secret, you should be able to see its **Value** in the **Client secrets** section. Click on the **Copy to clipboard** button to copy the Client Secret into the clipboard.



Paste the **Client Secret** into the same text document with the **Client ID** and **Tenant ID**.



## Testing the Tenant Management project with Visual Studio 2019

Here are the steps. You must install Visual Studio 2019or Visual Studio Code to run this sample, You can use either of these although this document will cover the details of getting the project running with Visual Studio 2019. You will have to figure itu out on your own if you are using Visual Studio Code.

### Download the Source Code

The source code for the **TenantManagement** project is maintained in a GitHib repository at the following URL.

* <https://github.com/PowerBiDevCamp/TenantManagement>

You can download the project source files as a ZIP archive using [this link](https://github.com/PowerBiDevCamp/TenantManagement/archive/refs/heads/main.zip). If you are familiar with the **git** utility, you can clone the project source files to your local computer using the following **git** command.

git clone <https://github.com/PowerBiDevCamp/TenantManagement.git>

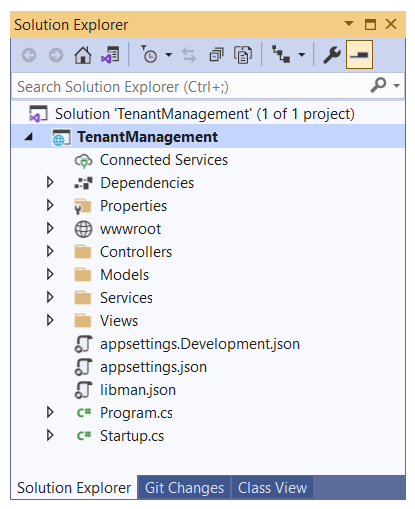
Once you have downloaded the source files for the **TenantManagement** repository to your local computer, you will see there is a top-level folder named **TenantManagement** which contains several files including a Visual Studio solution file named **TenantManagement.sln**.

Graphical user interface, application

Description automatically generated

### Open the Project in Visual Studio 2019

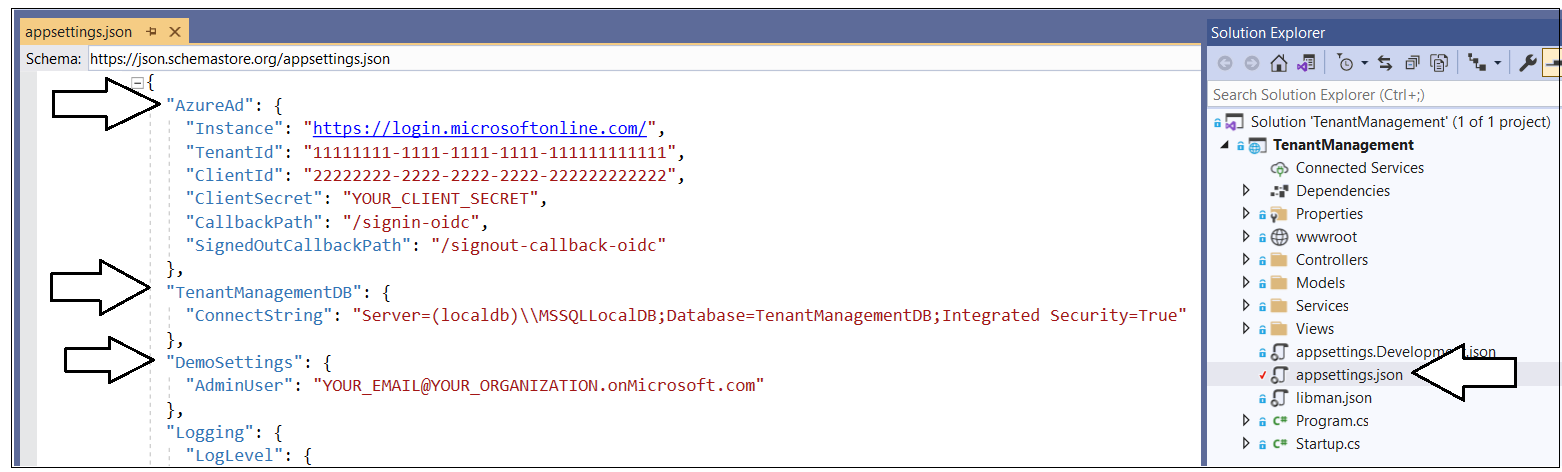
Xxxxx



zzz

### Update application settings in the appsettings.json file

Ssss



Xxx



This is the database connection string.

"Server=(localdb)\\MSSQLLocalDB;Database=TenantManagementDB;Integrated Security=True"

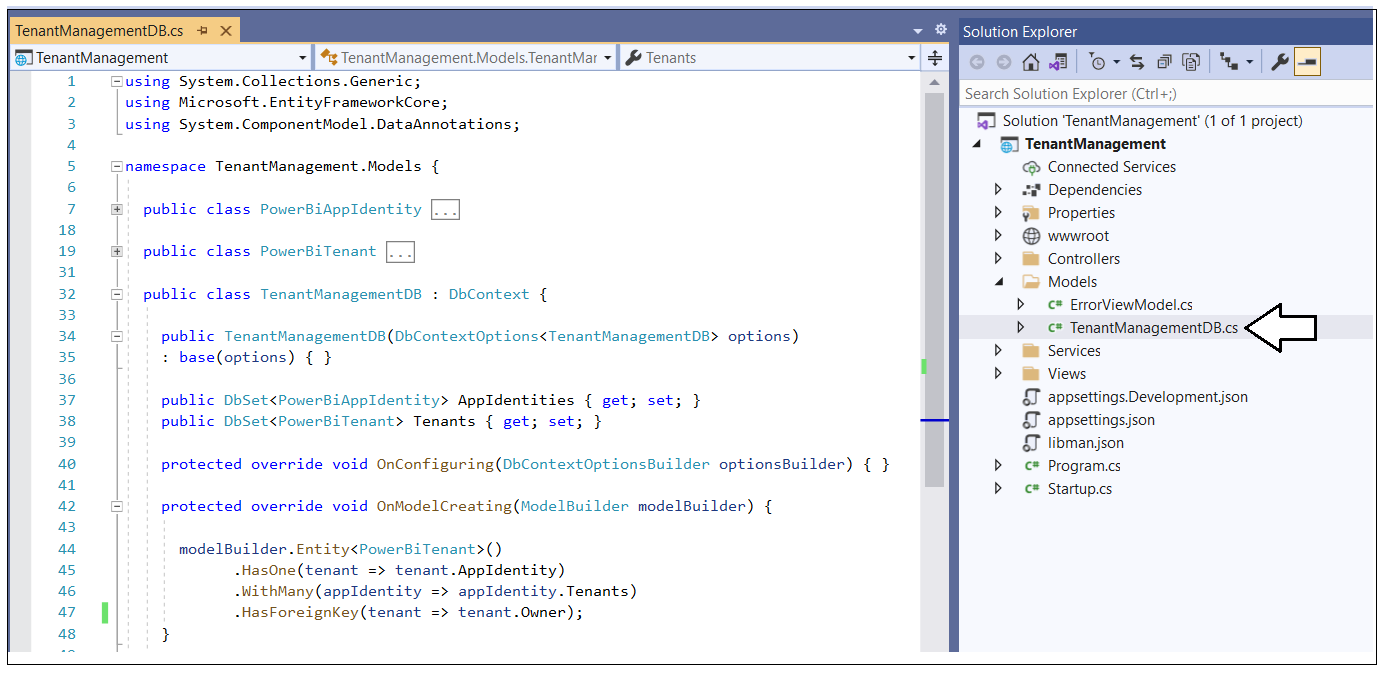
X



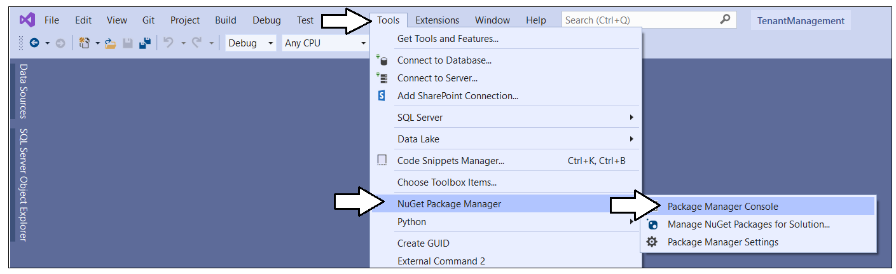
x

### Create the TenantManagementDB database

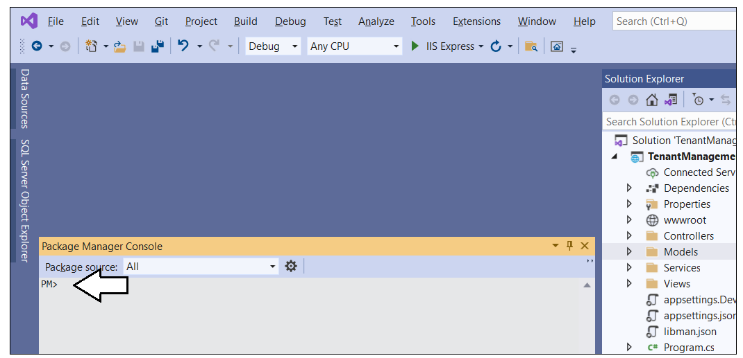
Xxxx



X



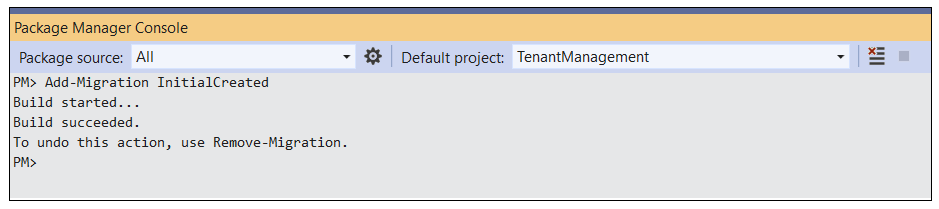
Z



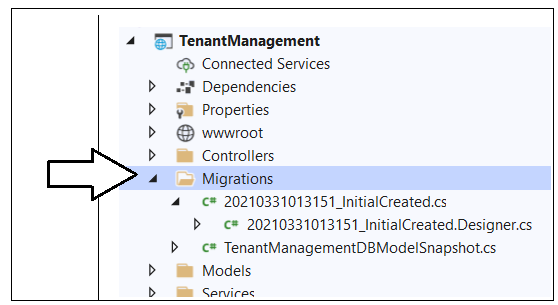
Zz

Add-Migration InitialCreate

xxx



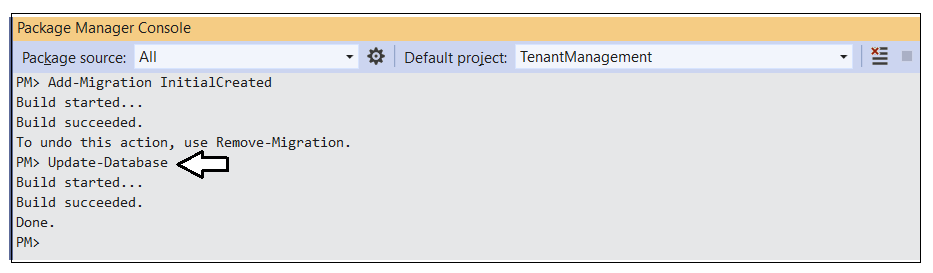
Ssss



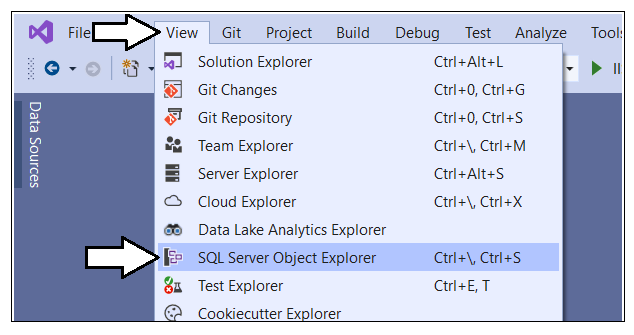
Ss

Update-Database

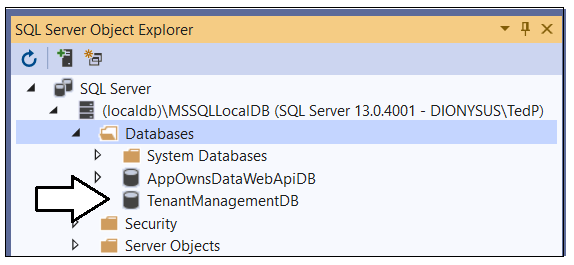
Xx



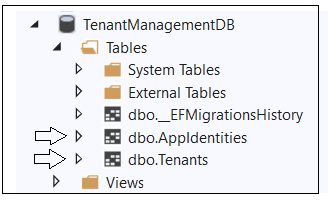
Xxx



Xxx



Xx



xxx

## Test the Tenant Management Application

Xxxx

