

# Secrets NET8 EF8 Blazor VS DEV V1

Version 1: 3/12/2024

- NET CORE 8.02
- EF 8
- Blazor/Razor
- VS 17.9.2
- Development Environment

Most development secrets are stored Using Azure Key Vault, similar to those used in production. The local workstation Secrets Manager tool stores sensitive data during application development.

There are multiple ways to retrieve your secrets. In this example, we are going to use the “DefaultAzureCredential” method to get an instance of the credentials object.

[Safe storage of app secrets in development in ASP.NET Core | Microsoft Learn](#)

## Table of Contents

Place a Secret in an Azure Key Vault .....	2
Create Azure Key Vault .....	2
Register your application with Azure .....	12
Name .....	15
Application ID .....	16
Object ID .....	16
Get Values from App registrations .....	16
Prepare your project .....	16
Add Azure Secrets Class to VS Project.....	17
Add Azure Secrets to Programs.cs .....	17
Other Secrets in the Development Code .....	17
Initialize user-secrets in .NET CLI.....	17
Enhance local secrets from Visual Studio .....	18
Enhancements to retrieve local user-secrets.....	18
Changes to Program.cs.....	18
Changes to Secrets.cs.....	19

## Place a Secret in an Azure Key Vault


### Create Azure Key Vault

If you haven't already, create an Azure Key Vault:

Launch Microsoft Azure:

Type "key vault" into search:

Select Key vaults: 

Select <+Create>:  to display the Create a key vault form:

### Create a key vault

Select the subscription to manage deployed resources and costs. Use resource groups like folders to organize and manage all your resources.

Subscription \*

Visual Studio Professional Subscription

Resource group \*

(New) rg\_secrets

[Create new](#)

#### Instance details

Key vault name \* ⓘ

kv-secrets-vs-dev ✓

Region \*

Central US

Pricing tier \* ⓘ

Standard

#### Recovery options

Soft delete protection will automatically be enabled on this key vault. This feature allows you to recover or permanently delete a key vault and secrets for the duration of the retention period. This protection applies to the key vault and the secrets stored within the key vault.

To enforce a mandatory retention period and prevent the permanent deletion of key vaults or secrets prior to the retention period elapsing, you can turn on purge protection. When purge protection is enabled, secrets cannot be purged by users or by Microsoft.

Soft-delete ⓘ

Enabled

Days to retain deleted vaults \* ⓘ

90

Purge protection ⓘ

☒ Disable purge protection (allow key vault and objects to be purged during retention period)

☐ Enable purge protection (enforce a mandatory retention period for deleted vaults and vault objects)

Previous

Next

Review + create

Select your Subscription: (In this example I selected my "Visual Studio Professional Subscription")

Select your Resource Group (or create a new resource group)

Enter your key vault name (I use a prefix followed by a dash for all of my names wherever I can). In this instance I am using kv-secrets-vs-dev. (This must be globally unique for all users in the Region)

Select the Azure Region (I use Central US or East US 2 since both of them provide a full slate of services)

Select the Standard pricing tier since it is free.

I use the defaults for the Recovery options

Select <Next>:

## Create a key vault

BasicsAccess configurationNetworkingTagsReview + create

### Configure data plane access for this key vault

To access a key vault in data plane, all callers (users or applications) must have proper authentication and authorization

#### Permission model

Grant data plane access by using a [Azure RBAC](#) or [Key Vault access policy](#)

☒ Azure role-based access control (recommended) ⓘ

☐ Vault access policy ⓘ

#### Resource access

☐ Azure Virtual Machines for deployment ⓘ

☐ Azure Resource Manager for template deployment ⓘ

☐ Azure Disk Encryption for volume encryption ⓘ

I don't make any changes to the Access Configuration page.

Select <Next>:

## Create a key vault

Basics Access configuration **Networking** Tags Review + create

You can connect to this key vault either publicly, via public IP addresses or service endpoints, or privately, using a private endpoint.

Public network access

- ☒ Enabled  
Allow all inbound and outbound access with option to restrict select inbound access using resource access configurations for this resource
- ☐ Disabled  
Restrict inbound access while allowing outbound access
- ☐ Secure by Perimeter (Most Restricted)  
Restrict all inbound and outbound access using a network security perimeter




You can change this or configure another connectivity method after this resource has been created. [Learn more](#)



Enabling public network access will make this resource available publicly. Unless public access is required, we recommend using a more restricted access type. [Learn more](#)

### Public Access

Allow access from:

- ☒ All networks
- ☐ Selected networks
-  Traffic from all public networks can access this resource. This is not recommended for private applications or environments. [Learn more](#)

### Private endpoint

Create a private endpoint to allow a private connection to this resource. Additional private endpoint connections can be created with

[+ Create a private endpoint](#)

Name	Subscription	Resource group	Region	Subnet
------	--------------	----------------	--------	--------

Previous

Next

Review + create

I use the defaults for the Networking page since I want to be able to access the key vault from my personal workstation.

Select <Review + create>

## Create a key vault

Basics   Access configuration   Networking   Tags   Review + create

### Review + create

#### Basics

Subscription	Visual Studio Professional Subscription
Resource group	rg_secrets
Key vault name	kv-secrets-vs-dev
Region	Central US
Pricing tier	Standard
Soft-delete	Enabled
Purge protection during retention period	Disabled
Days to retain deleted vaults	90 days

#### Access configuration


Azure Virtual Machines for deployment	Disabled
Azure Resource Manager for template deployment	Disabled
Azure Disk Encryption for volume encryption	Disabled
Permission model	Azure role-based access control


#### Networking

Connectivity method	Public endpoint (all networks)
---------------------	--------------------------------

[Previous](#)[Next](#)[Create](#)

Review your entries and select <Create> to deploy the Key Vault.

 **Your deployment is complete**




Deployment name : kv-secrets-vs-dev      Start time : 3/12/2024, 2:03:53 PM  
Subscription : [Visual Studio Professional Subscription](#)      Correlation ID : 2032e185-9a83-40f2-...  
Resource group : [rg\\_secrets](#)





> Deployment details

▼ Next steps

[Go to resource](#)

Select <Go to resource>:

 **kv-secrets-vs-dev**    🔗 ☆ ...  
Key vault

 Delete     Move     Refresh     Open in mobile

Overview

Activity log

Access control (IAM)

Tags

Diagnose and solve problems

Access policies

Events

Objects

Keys

Secrets

Certificates

Settings

Essentials

Resource group [\(move\)](#)  
[rg\\_secrets](#)

Location  
Central US

Subscription [\(move\)](#)  
[Visual Studio Professional Subscription](#)

Subscription ID  
35445d0a-e078-4667-b482-75e332ca6544

Tags [\(edit\)](#)  
[Add tags](#)

Vault URI  
<https://kv-secrets-vs-dev.vault.azure.net/>

Sku (Pricing tier)  
Standard

Directory ID  
4c073dac-49fe-43be-964d-3c9245cc0276

Directory Name  
SLM Software Company

Soft-delete  
[Enabled](#)







Purge protection  
[Disabled](#)

Get started    Properties    Monitoring    Tools + SDKs    Tutorials

You will probably not have enough access rights to add a secret.

You will want to remember the “Vault URI” from this form. You will need it later.

Select <Access control (IAM)> on the left side nav bar:

 Add     Download role assignments     Edit columns     Refresh     Remove     Feedback

Check access

Role assignments    Roles    Deny assignments    Classic administrators

**My access**  
View my level of access to this resource.  
[View my access](#)

**Check access**  
Review the level of access a user, group, service principal, or managed identity has to this resource. [Learn more](#)

[Check access](#)

If you are the owner of the Azure Subscription you can add yourself.

Select <Role assignments> from the top menu:

+ Add ▾ Download role assignments Edit columns Refresh Remove Feedback

Check access **Role assignments** Roles Deny assignments Classic administrators

**Number of role assignments for this subscription** ⓘ 5 4000 **Privileged** ⓘ 0 [View assignments](#)

**All** Job function (0) Privileged (0)

Search by name or email

Type : **All** Role : **All** Scope : **All scopes**

Group by : **Role**

0 items

Name	Type	Role	Scope	Condition
No user assignments exist				

Select <Add> and <Add role assignment> from the top menu:

+ Add ▾ Download role assignments

Add role assignment

Add co-administrator

**Add role assignment** ...

**Role** Members Conditions Review + assign

A role definition is a collection of permissions. You can use the built-in roles or you can create your own custom roles. [Learn more](#)

**Job function roles** Privileged administrator roles

Grant access to Azure resources based on job function, such as the ability to create virtual machines.

Search by role name, description, or ID

Type : **All** Category : **All**

Name ↑↓	Description ↑↓	Type ↑↓	Category ↑↓	Details
Reader	View all resources, but does not allow yo...	BuiltInRole	General	<a href="#">View</a>
App Compliance Auto...	Create, read, download, modify and delet...	BuiltInRole	None	<a href="#">View</a>
App Compliance Auto...	Read, download the reports objects and r...	BuiltInRole	None	<a href="#">View</a>
Azure AI Enterprise Ne...	Can approve private endpoint connectio...	BuiltInRole	None	<a href="#">View</a>
Desktop Virtualization ...	This role is in preview and subject to cha...	BuiltInRole	None	<a href="#">View</a>

Enter "Key Vault" into the search by role name box:

[Role](#) [Members](#) [Conditions](#) [Review + assign](#)

A role definition is a collection of permissions. You can use the built-in roles or you can create your own custom roles. [Learn more](#)

[Job function roles](#) Privileged administrator roles

Grant access to Azure resources based on job function, such as the ability to create virtual machines.

× Type : All Category : All

Name ↑↓	Description ↑↓	Type ↑↓	Category ↑↓	Details
Key Vault Administrator	Perform all data plane operations on a ke...	BuiltInRole	Security	<a href="#">View</a>
Key Vault Certificate U...	Read certificate contents. Only works for ...	BuiltInRole	None	<a href="#">View</a>
Key Vault Certificates ...	Perform any action on the certificates of ...	BuiltInRole	Security	<a href="#">View</a>
Key Vault Contributor	Lets you manage key vaults, but not acce...	BuiltInRole	Security	<a href="#">View</a>
Key Vault Crypto Officer	Perform any action on the keys of a key v...	BuiltInRole	Security	<a href="#">View</a>
Key Vault Crypto Servi...	Read metadata of keys and perform wrap...	BuiltInRole	Security	<a href="#">View</a>
Key Vault Crypto Servi...	Release keys. Only works for key vaults th...	BuiltInRole	None	<a href="#">View</a>
Key Vault Crypto User	Perform cryptographic operations using ...	BuiltInRole	Security	<a href="#">View</a>
Key Vault Data Access ...	Manage access to Azure Key Vault by ad...	BuiltInRole	None	<a href="#">View</a>
Key Vault Reader	Read metadata of key vaults and its certif...	BuiltInRole	Security	<a href="#">View</a>
Key Vault Secrets Officer	Perform any action on the secrets of a ke...	BuiltInRole	Security	<a href="#">View</a>
Key Vault Secrets User	Read secret contents. Only works for key ...	BuiltInRole	Security	<a href="#">View</a>

[Review + assign](#) [Previous](#) [Next](#)

Select “Key Vault Administrator”:

Select <Members> from the top menu:



## Add role assignment ...

Role **Members** Conditions Review + assign

**Selected role**  
Key Vault Administrator

**Assign access to**

☒ User, group, or service principal  
☐ Managed identity

**Members**  
[+ Select members](#)

Name	Object ID	Type
No members selected		

**Description**

Optional

Select <+ Select members>

The Select members dialog displays and you can select yourself and others:


## Select members

Select ⓘ

sam

No users, groups, or service principals found.

Selected members:

 Sam Matzen  
smatzen@slm.com

Remove

Select <Select> to add the members to the Key Vault Administrators group.

Role

**Members**


Conditions

Review + assign

**Selected role**  
Key Vault Administrator

**Assign access to**  
☒ User, group, or service principal  
☐ Managed identity

**Members**  
[+ Select members](#)

Name	Object ID	Type	
Sam Matzen	b0c00e2e-34e0-4d49-bd2a-039d8...	User	

You aren't done yet:

Select <Next> or <Review + assign> to get the Review + assign form:

[Role](#)
[Members](#)
[Conditions](#)
[Review + assign](#)

**Role**  
Key Vault Administrator

**Scope**  
/subscriptions/35445d0a-e078-4667-b482-75e332ca6544/resourceGroups/rg-ltslm/providers/Microsoft.KeyVault/vaults/kv-xxxxxxx

**Members**

Name	Object ID	Type
Sam Matzen	b0c00e2e-34e0-4d49-bd2a-039d849e45a5	User

**Description**  
No description

Select <Review and assign> to assign the selected users to the role.

Now, go back and do the same thing for Key Vault Reader so you can have access to the secrets from your workstation.

When completed, your Role Assignments form should look something like this:

[All](#)
[Job function \(2\)](#)
[Privileged \(0\)](#)

Type : **All**
Role : **All**
Scope : **All scopes**
Group by : **Role**

2 items (2 Users)

Name	Type	Role	Scope	Condition
<div> <div>Key Vault Administrator (1)</div> <div> <input type="checkbox"/> <div> Sam Matzen  smatzen@slm.com </div> </div> </div>	User	Key Vault Administrator ⓘ	This resource	None
<div> <div>Key Vault Reader (1)</div> <div> <input type="checkbox"/> <div> Sam Matzen  smatzen@slm.com </div> </div> </div>	User	Key Vault Reader ⓘ	This resource	None

Now you should be able to add the SyncFusion secret:

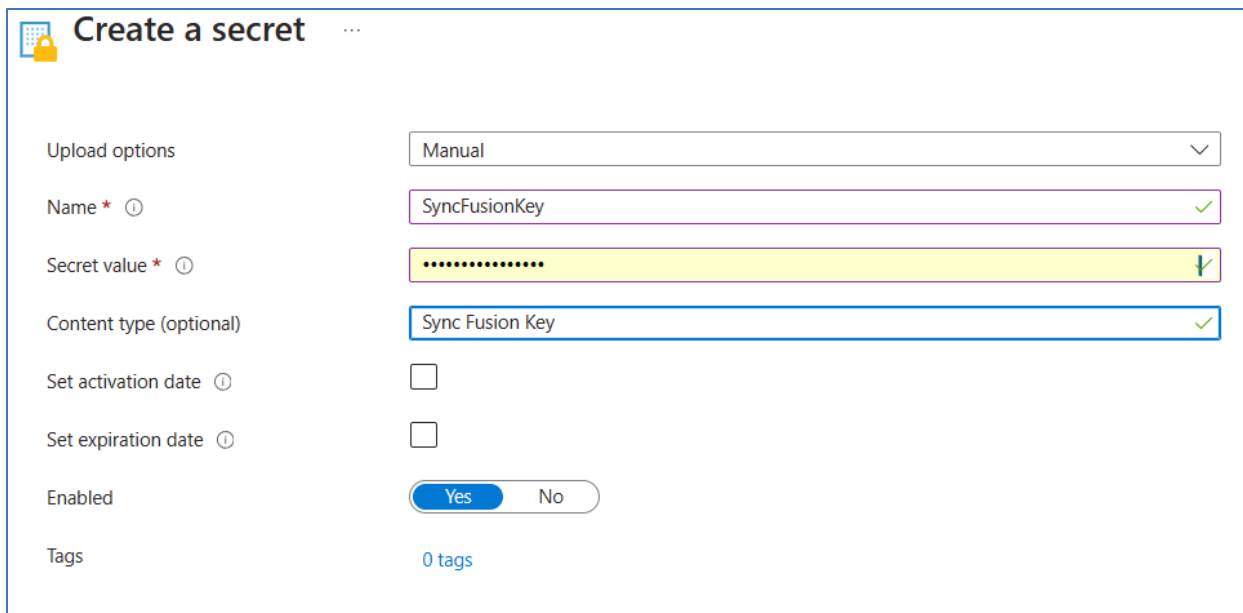
Select <Secrets> from the menu on the left:

[+ Generate/Import](#)
[Refresh](#)
[Restore Backup](#)
[View sample code](#)
[Manage deleted secrets](#)

Name	Type	Status	Expiration date
There are no secrets available.			

Select <+ Generate/Import> :

[+ Generate/Import](#)



**Create a secret** ...

Upload options: Manual

Name \* ⓘ: SyncFusionKey ✓

Secret value \* ⓘ: ..... ✓

Content type (optional): Sync Fusion Key ✓

Set activation date ⓘ: ☐

Set expiration date ⓘ: ☐

Enabled: Yes No

Tags: 0 tags

Select Upload Options of “Manual”

Enter a Name for the key. For this example, I have used “SyncFusionKey”

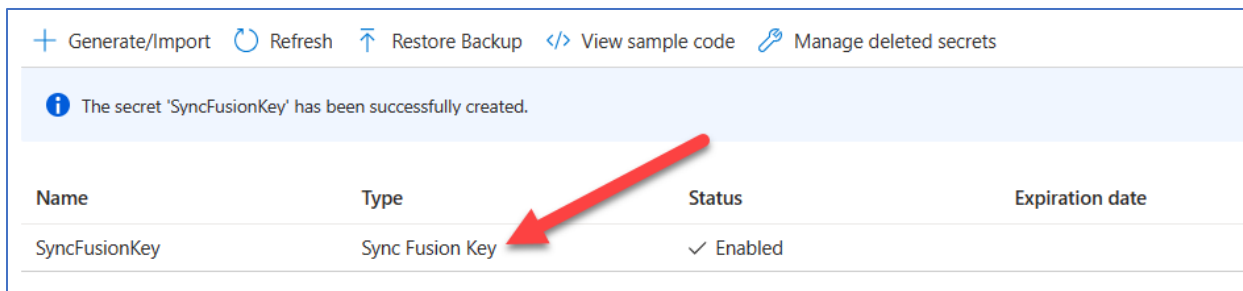
Paste your Sync Fusion Activation Key into the Secret Value

Enter a description of the key in Content Type. This shows up on other screens.

Clear both the Set Activation Date and Set Expiration Date checkboxes.

Enabled is the default.

Select <Create> to create the secret and you should see something like:




+ Generate/Import   Refresh   Restore Backup   View sample code   Manage deleted secrets

**i** The secret 'SyncFusionKey' has been successfully created.

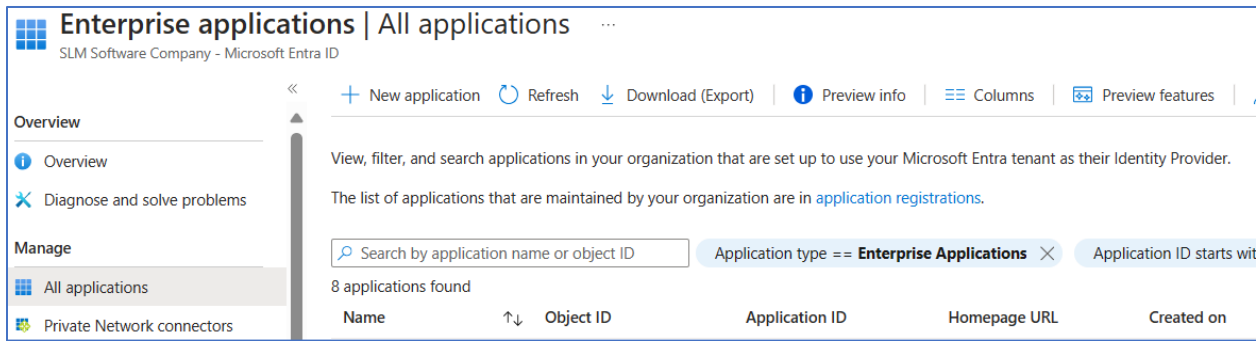
Name	Type	Status	Expiration date
SyncFusionKey	Sync Fusion Key	✓ Enabled	

## Register your application with Azure

You need to register your application with Azure as an “Enterprise application”:

In Azure, enter “Enterprise Applications” in search and select  Enterprise applications:

You should see a list of your currently registered applications:



Enterprise applications | All applications

SLM Software Company - Microsoft Entra ID

Overview

View, filter, and search applications in your organization that are set up to use your Microsoft Entra tenant as their Identity Provider.

The list of applications that are maintained by your organization are in [application registrations](#).

Search by application name or object ID

Application type == Enterprise Applications

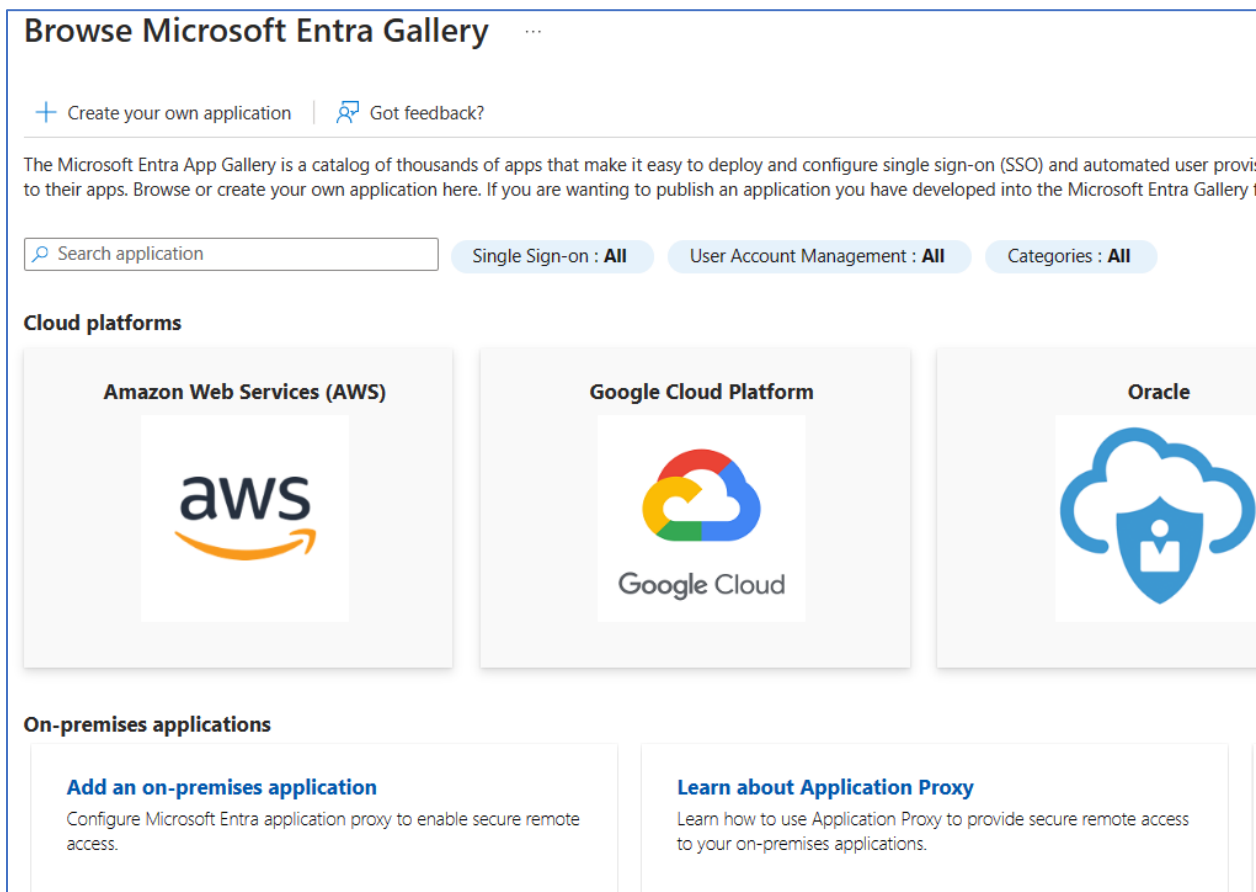
Application ID starts with

8 applications found

Name	Object ID	Application ID	Homepage URL	Created on
------	-----------	----------------	--------------	------------

Select <+ New application> [+ New application](#):

You should see a confusing page something like the following that has little to do with registering our new application:



Browse Microsoft Entra Gallery

+ Create your own application | Got feedback?

The Microsoft Entra App Gallery is a catalog of thousands of apps that make it easy to deploy and configure single sign-on (SSO) and automated user provisioning to their apps. Browse or create your own application here. If you are wanting to publish an application you have developed into the Microsoft Entra Gallery...

Search application

Single Sign-on : All | User Account Management : All | Categories : All

Cloud platforms

Amazon Web Services (AWS)

Google Cloud Platform

Oracle

On-premises applications

Add an on-premises application


Configure Microsoft Entra application proxy to enable secure remote access.

Learn about Application Proxy

Learn how to use Application Proxy to provide secure remote access to your on-premises applications.


Select <+ Create your own application> [+ Create your own application](#) and the Create your own application dialog displays:

## Create your own application ✕

 Got feedback?

If you are developing your own application, using Application Proxy, or want to integrate an application that is not in the gallery, you can create your own application here.

What's the name of your app?

Secrets\_NET8\_EF8\_Blazor\_VS\_DEV\_V1 

What are you looking to do with your application?

☐ Configure Application Proxy for secure remote access to an on-premises application

☒ Register an application to integrate with Microsoft Entra ID (App you're developing)

☐ Integrate any other application you don't find in the gallery (Non-gallery)

Enter the name of your application in the name text.

Select the "Register an application" radio button.

Select <Create>

The Register an application form displays:

## Register an application ...

**\* Name**

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

Secrets\_NET8\_EF8\_Blazor\_VS\_DEV\_V1

**Supported account types**

Who can use this application or access this API?

☒ Accounts in this organizational directory only (SLM Software Company only - Single tenant)

☐ Accounts in any organizational directory (Any Microsoft Entra ID tenant - Multitenant)


☐ Accounts in any organizational directory (Any Microsoft Entra ID tenant - 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.

Select a platform 

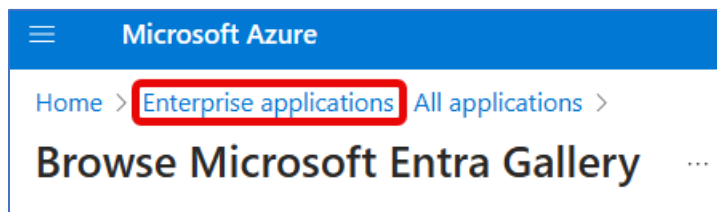
e.g. https://example.com/auth

For this exercise, accept the defaults unless you know you want to make additional selections.

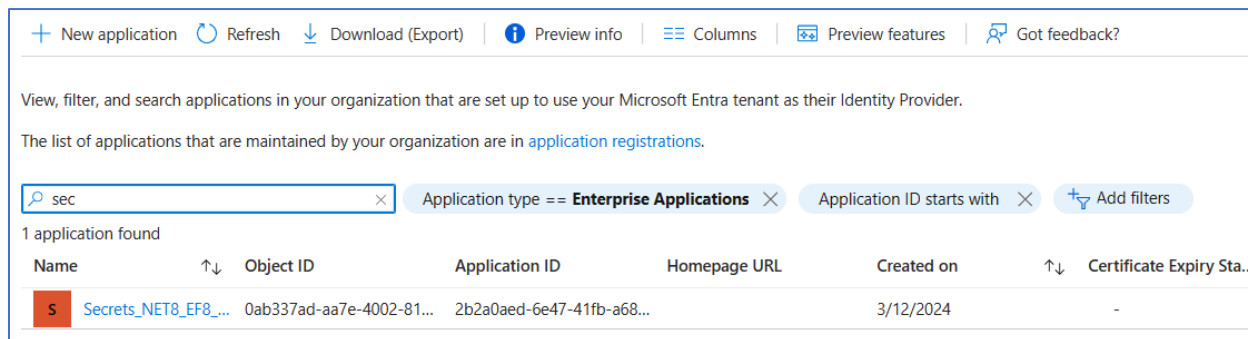
Select <Register>

The Browse Microsoft Entra Gallery form displays where you can register another application, or:

Select “Enterprise applications” from the breadcrumb bar:



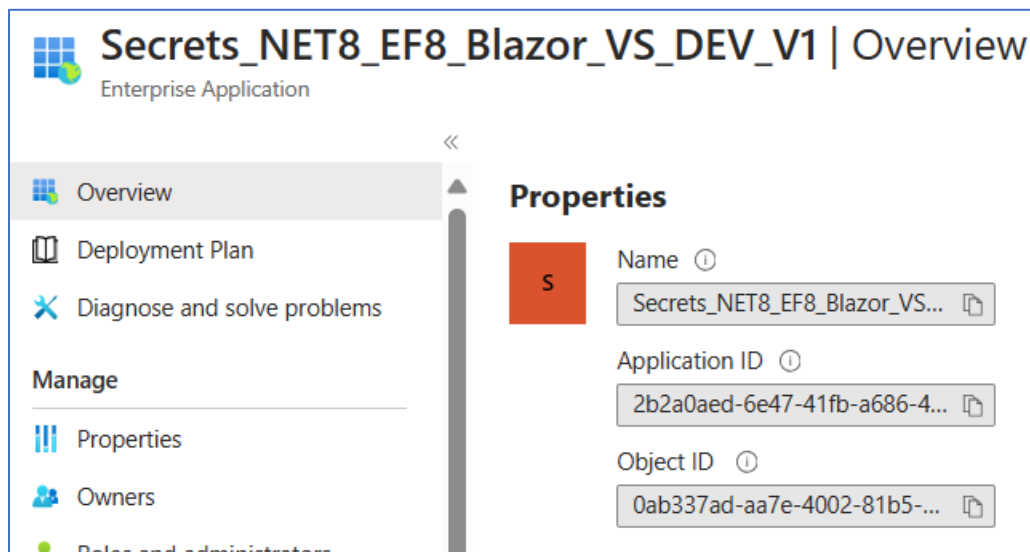
The enterprise applications form displays:



It takes a couple of minutes before your new application shows up in the Enterprise Applications form, so you may need to refresh a few times until it appears.

At a minimum you will need the Application ID to access the secrets from your application.

Select your application from the list to display the application values:



There are many options and additional data associated with an application registration, but we are only interested in the Name, Application ID, and Object ID:

### Name

The application name is “Secrets\_NET8\_EF8\_Blazor\_VS\_DEV\_V1”

## Application ID

The Application ID is “2b2a0aed-6e47-41fb-a686-4f16fea50504”

## Object ID

The Object ID is “0ab337ad-aa7e-4002-81b5-7e4b5f7f0703”

## Get Values from App registrations

Load the application on the App Registrations screen to get the client ID and tenant ID

Select <Home> from the bread-crum bar.

Type “App Registrations” in the search box.

Select <App registrations>

Select <All applications>

The screenshot shows the 'App registrations' page in the Azure portal. At the top, there are navigation links: '+ New registration', 'Endpoints', 'Troubleshooting', 'Refresh', 'Download', 'Preview features', and 'Got feedback?'. Below this is an information banner about the ADAL and Graph deprecation. The main section has tabs for 'All applications', 'Owned applications', and 'Deleted applications'. A search bar contains the text 'sec' and an 'Add filters' button. Below the search bar, it says '1 applications found'. The application list shows one application: 'Secrets\_NET8\_EF8\_Blazor\_VS\_DEV\_V1' with a small icon to its left.

Select your new application to view the “client ID” and “tenant ID”

The screenshot shows the details page for the application 'Secrets\_NET8\_EF8\_Blazor\_VS\_DEV\_V1'. The left sidebar has a search bar and navigation links: 'Overview', 'Quickstart', 'Integration assistant', 'Manage', 'Branding & properties', 'Authentication', and 'Certificates & secrets'. The main content area has a 'Got a second?' message and a table of 'Essentials'. The table lists the following details:

Display name	: Secrets_NET8_EF8_Blazor_VS_DEV_V1	Client credentials	: <a href="#">Add a certificate or secret</a>
Application (client) ID	: 2b2a0aed-6e47-41fb-a686-4f16fea50504	Redirect URIs	: <a href="#">Add a Redirect URI</a>
Object ID	: 9ec0285e-bba8-406a-b3a2-25e4b9375be7	Application ID URI	: <a href="#">Add an Application ID URI</a>
Directory (tenant) ID	: 4c073dac-49fe-43be-964d-3c9245cc0276	Managed application in L...	: <a href="#">Secrets_NET8_EF8_Blazor_VS_DEV_V1</a>
Supported account types	: <a href="#">My organization only</a>		

## Prepare your project

Using PowerShell run the following on the directory containing your project:

```
dotnet add package Azure.Identity
dotnet add package Azure.Security.KeyVault.Secrets
```

These statements loaded the necessary NuGet packages.



## Add Azure Secrets Class to VS Project

In the root of the project, add a new class named Secrets.cs with the following code:

```
using Azure.Identity;
using Azure.Security.KeyVault.Secrets;

namespace KeyVault01.Secrets
{
    public class Secrets()
    {
        public string GetSecret(string SecretName)
        {
            string userAssignedClientId = "2b2a0aed-6e47-41fb-a686-4f16fea50504";
            var credential = new DefaultAzureCredential(new DefaultAzureCredentialOptions
            {
                // set the user assigned identity in Azure
                ManagedIdentityClientId = userAssignedClientId
                // exclude Managed Identity
                , ExcludeManagedIdentityCredential = true
            });

            var client = new SecretClient(
                new Uri("https://kv-secrets-vs-dev.vault.azure.net/"), credential);

            var secret = client.GetSecret(SecretName);

            return secret.Value.Value.ToString();
        }
    }
}
```

Replace the value for userAssignedClientId with your application ID.

Replace the Uri with the Uri for your Key Vault.

## Add Azure Secrets to Programs.cs

Add the following code to the Programs.cs file to retrieve a secret:

```
// retrieve SyncFusion Key from Key Vault
Secrets secrets = new Secrets();
string SyncFusionKey = secrets.GetSecret("SyncFusionKey");
Console.WriteLine(SyncFusionKey);
```

Replace the secret name as needed.

## Other Secrets in the Development Code

Now we have the core secrets like Activation Keys and Connection Strings in the Azure Key Vault, we need to remove the Application ID and Secret Names from the code. We do this by using the Secret Manager.

### Initialize user-secrets in .NET CLI

```
cd {your path to repos}\repos\Secrets_NET8_EF8_Blazor_VS_DEV_V1
dotnet user-secrets init -project Secrets_NET8_EF8_Blazor_VS_DEV_V1
```

This command adds a UserSecretsID to the project file:

```
<PropertyGroup>
  <TargetFramework>net8.0</TargetFramework>
  <Nullable>enable</Nullable>
  <ImplicitUsings>enable</ImplicitUsings>
  <UserSecretsId>532b3f6b-92a1-48b7-8c80-9dda51ec15</UserSecretsId>
</PropertyGroup>
```

In this example we have two user-secrets we want to put in the local secrets store so they are not checked-in to GitHub:

"Application:Id" "2b2a0aed-6e47-41fb-a686-4f16fea50504"

"Application:KeyVaultURI" "https://kv-secrets-vs-dev.vault.azure.net/"

"SecretName:SyncFusionKey" "SyncFusionKey"

Save the initial values with the these .NET CLI commands:

```
cd {your path to repos}\repos\Secrets_NET8_EF8_Blazor_VS_DEV_V1

dotnet user-secrets set "Application:Id" "2b2a0aed-6e47-41fb-a686-4f16fea50504" --project
Secrets_NET8_EF8_Blazor_VS_DEV_V1

dotnet user-secrets set "Application:KeyVaultURI" "https://kv-secrets-vs-dev.vault.azure.net/" --project
Secrets_NET8_EF8_Blazor_VS_DEV_V1

dotnet user-secrets set "SecretName:SyncFusionKey" "SyncFusionKey" --project Secrets_NET8_EF8_Blazor_VS_DEV_V1

dotnet user-secrets list --project Secrets_NET8_EF8_Blazor_VS_DEV_V1
```

The output from the "list" command is:

```
SecretName:SyncFusionKey = SyncFusionKey
Application:KeyVaultURI = https://kv-secrets-vs-dev.vault.azure.net/
Application:Id = 2b2a0aed-6e47-41fb-a686-4f16fea50504
```

## Enhance local secrets from Visual Studio

With the secrets loaded into the local secrets store, you can edit the secrets store from Visual Studio by right-clicking the project name and selecting <Manage User Secrets>. (I have seen how the initial configuration can be performed from within Visual Studio, but have never tried it.)

```
{
  "Application:KeyVaultURI": "https://kv-secrets-vs-dev.vault.azure.net/",
  "Application:Id": "2b2a0aed-6e47-41fb-a686-4f16fea50504",
  "SecretName:SyncFusionKey": "SyncFusionKey"
}
```

Edit the secrets as needed and select <Save> to persist the changes.

## Enhancements to retrieve local user-secrets

Now we need a way to retrieve the secrets from the local secret store.

In Program.cs:

```
var movieApiKey = builder.Configuration["Movies:ServiceApiKey"];
```

In a .razor page:

```
private readonly IConfiguration _config;

public IndexModel(IConfiguration config)
{
    _config = config;
}

public void OnGet()
{
    var moviesApiKey = _config["Movies:ServiceApiKey"];
```

## Changes to Program.cs

```
// retrieve SyncFusion Key from Key Vault
```

```

Secrets secrets = new Secrets();
string SyncFusionKey = secrets.GetSecret(builder.Configuration,
    builder.Configuration["SecretName:SyncFusionKey"] ?? "").ToString();
Console.WriteLine(SyncFusionKey);

```

We need a reference to the Configuration object passed to the GetSecrets method, and retrieve a secret directly from the Configuration object.

### Changes to Secrets.cs

```

using Azure.Identity;
using Azure.Security.KeyVault.Secrets;

namespace Secrets_NET8_EF8_Blazor_VS_DEV_V1
{
    public class Secrets()
    {
        public string GetSecret(IConfiguration config, string SecretName)
        {
            string userAssignedClientId = config["Application:Id"] ?? "";
            var credential = new DefaultAzureCredential(new DefaultAzureCredentialOptions
            {
                // set the user assigned identity in Azure
                ManagedIdentityClientId = userAssignedClientId,
                // exclude Managed Identity
                ExcludeManagedIdentityCredential = true
            });

            var client = new SecretClient(
                new Uri(config["Application:KeyVaultURI"] ?? ""), credential);

            var secret = client.GetSecret(SecretName);

            return secret.Value.Value.ToString();
        }
    }
}

```

First: We add a parameter for the Configuration (config) object reference.

Second: We retrieve the "Application:Id" from the configuration object.

Third: We retrieve the "Application:KeyVaultURI" from the configuration object.

Our code has no direct references to anything in the Azure Key Vault. The secret names are stored in the local user-secret store, so they are not checked into NuGet.