Azure Key Vault Revision History

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Description of Modifications** | **Updated By** |
| 04/10/2018 | 1.0 | Initial Draft | Elliot Faure Garcia |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Table of Contents

[Purpose 4](#_Toc511113338)

[Creating Key Vault 5](#_Toc511113339)

[Creating Secret 6](#_Toc511113340)

[Creating Application Registration 9](#_Toc511113341)

[Associating Application Registration w/ Secret 13](#_Toc511113342)

[Application Web.config 15](#_Toc511113343)

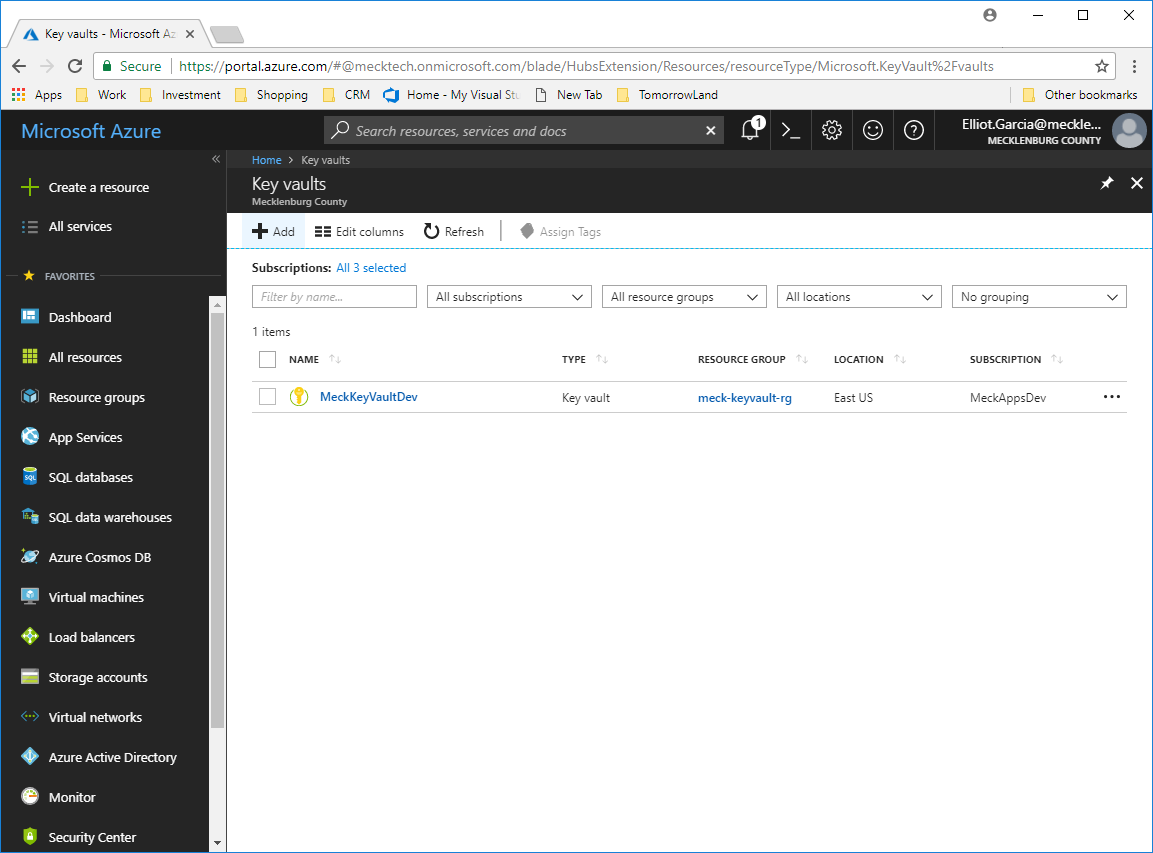
[.Net Code 16](#_Toc511113344)

Purpose

The purpose of this document is to provide a step by step tutorial on how to create an Azure Key Vault and consume it by a .Net application.

Creating Key Vault

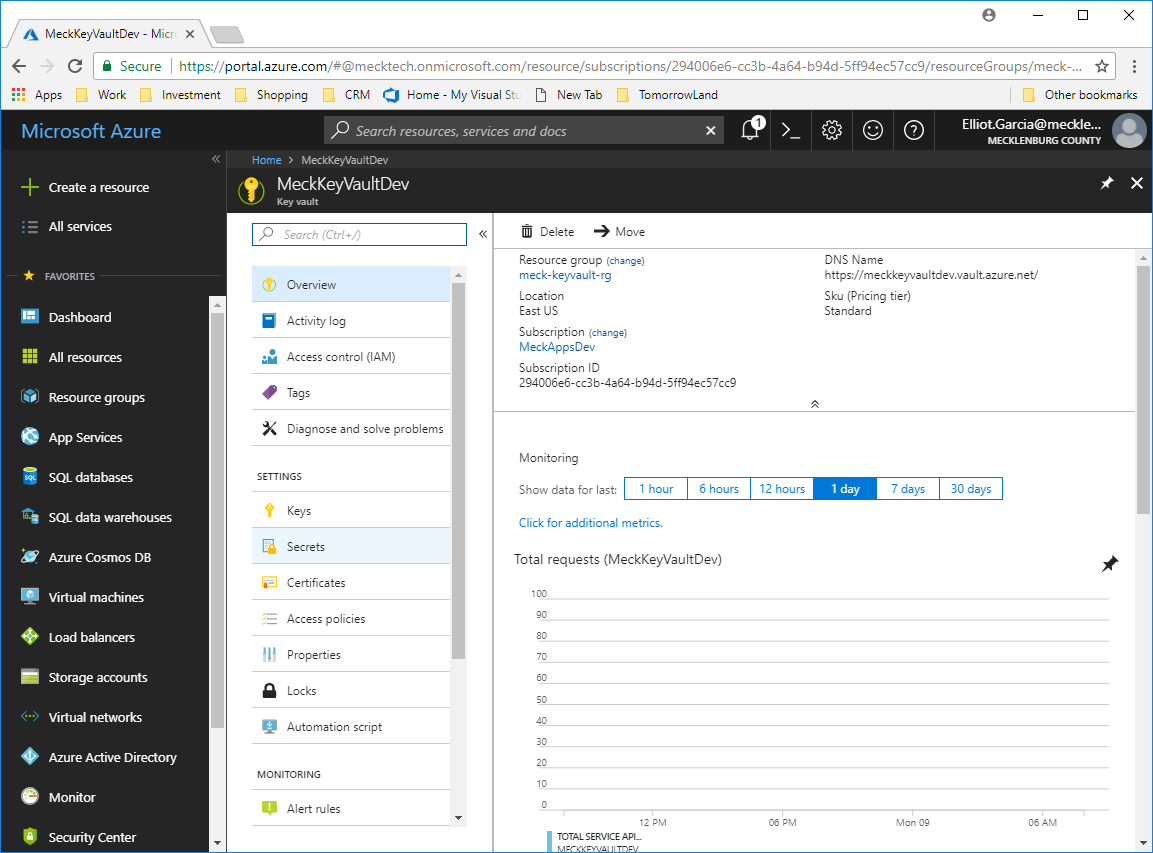
1. Log into the Azure portal and navigate to the **Key Vault** blade.
2. Click **Add**



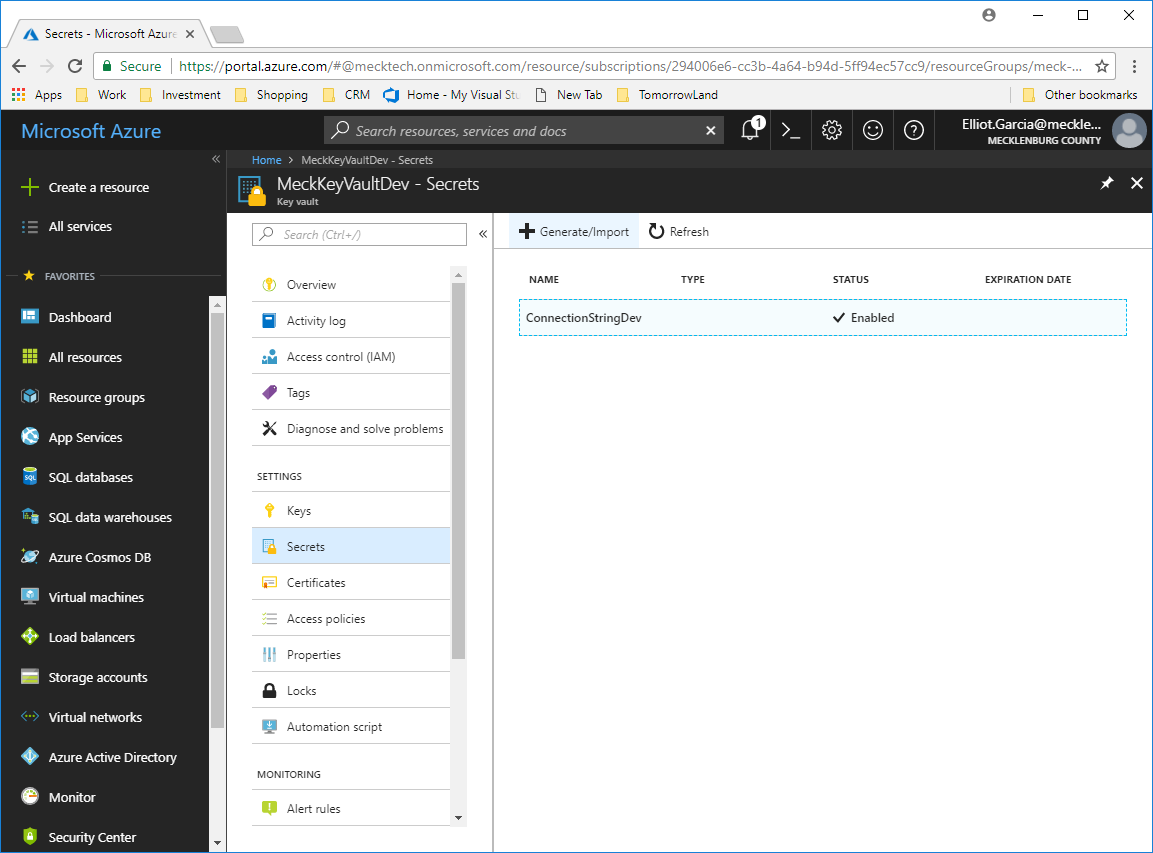
1. Enter **Name**
2. Select **Subscription**
3. Create or Select and existing **Resource Group**
4. Select **Location**
5. Leave **Pricing Tier** and **Access Policies** with default values
6. Click **Create**

Creating Secret

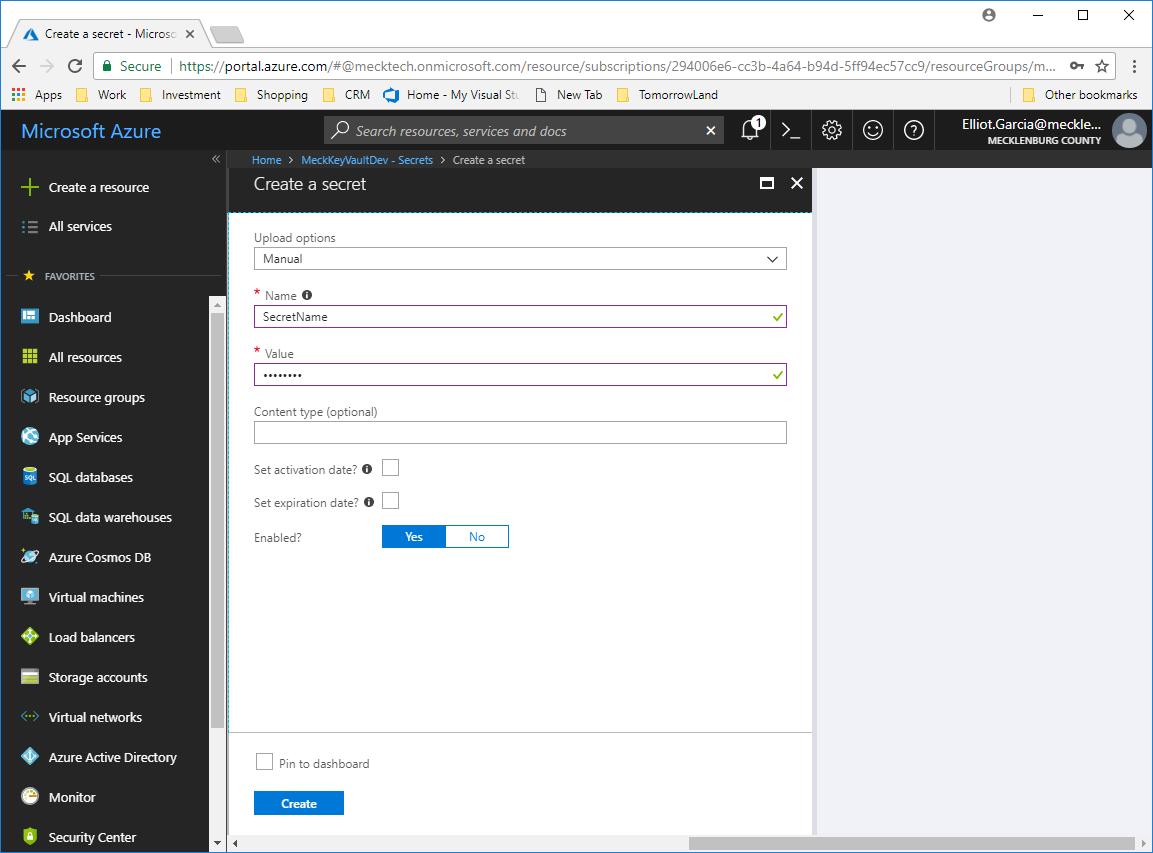
1. Navigate to the newly created Key Vault
2. Select **Secrets**



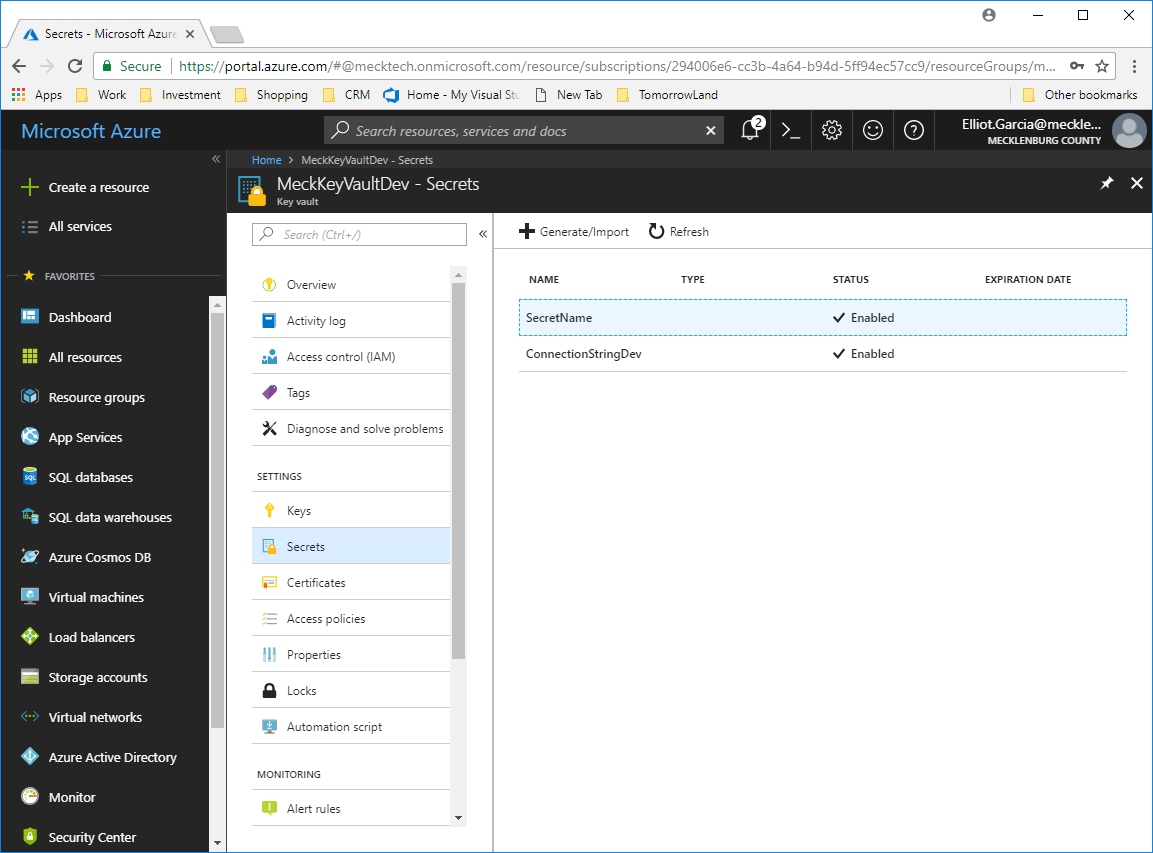
1. Click **Generate/Import**



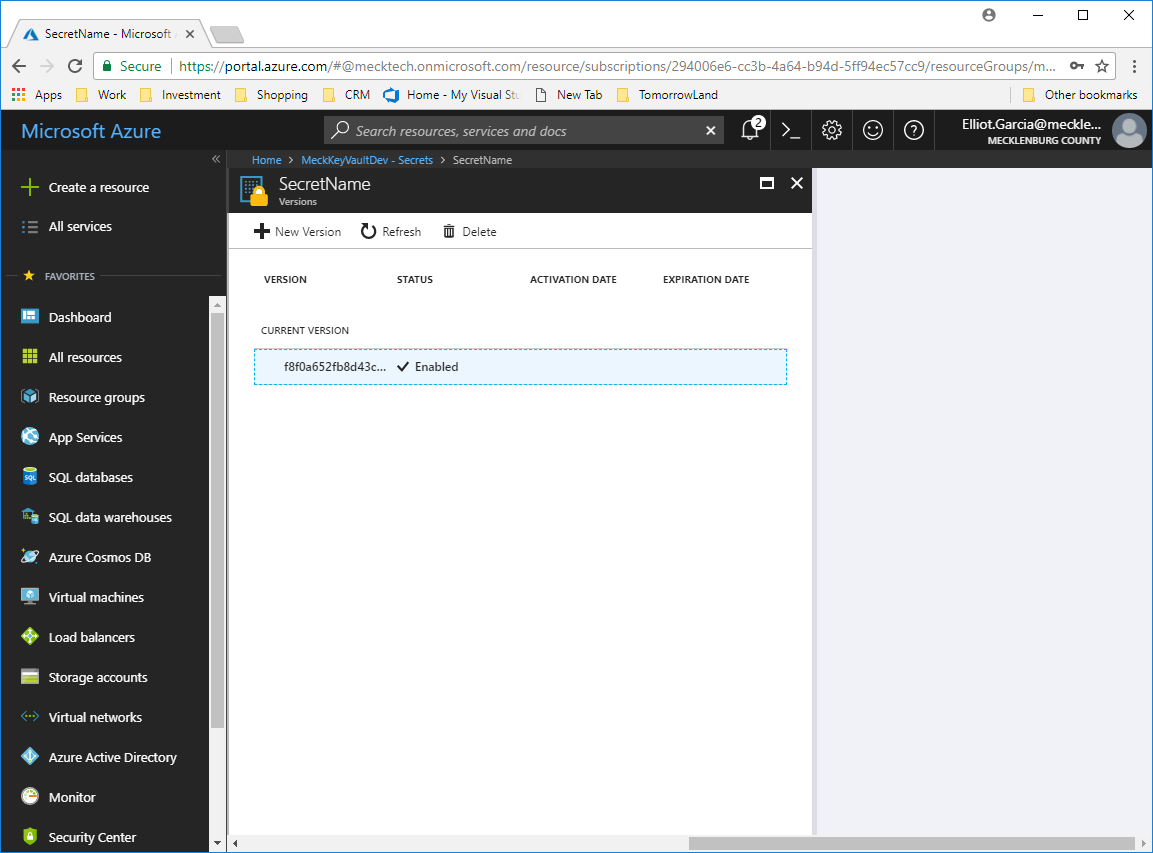
1. Select **Manual** for **Upload Options**
2. Enter a **Name**
3. Enter a **Value** (This will be the Connection String, Password etc. Consumed by the application)
4. Leave **Set Activation Date** and **Set Expiration Date** unchecked
5. Set **Enabled** to ‘Yes’



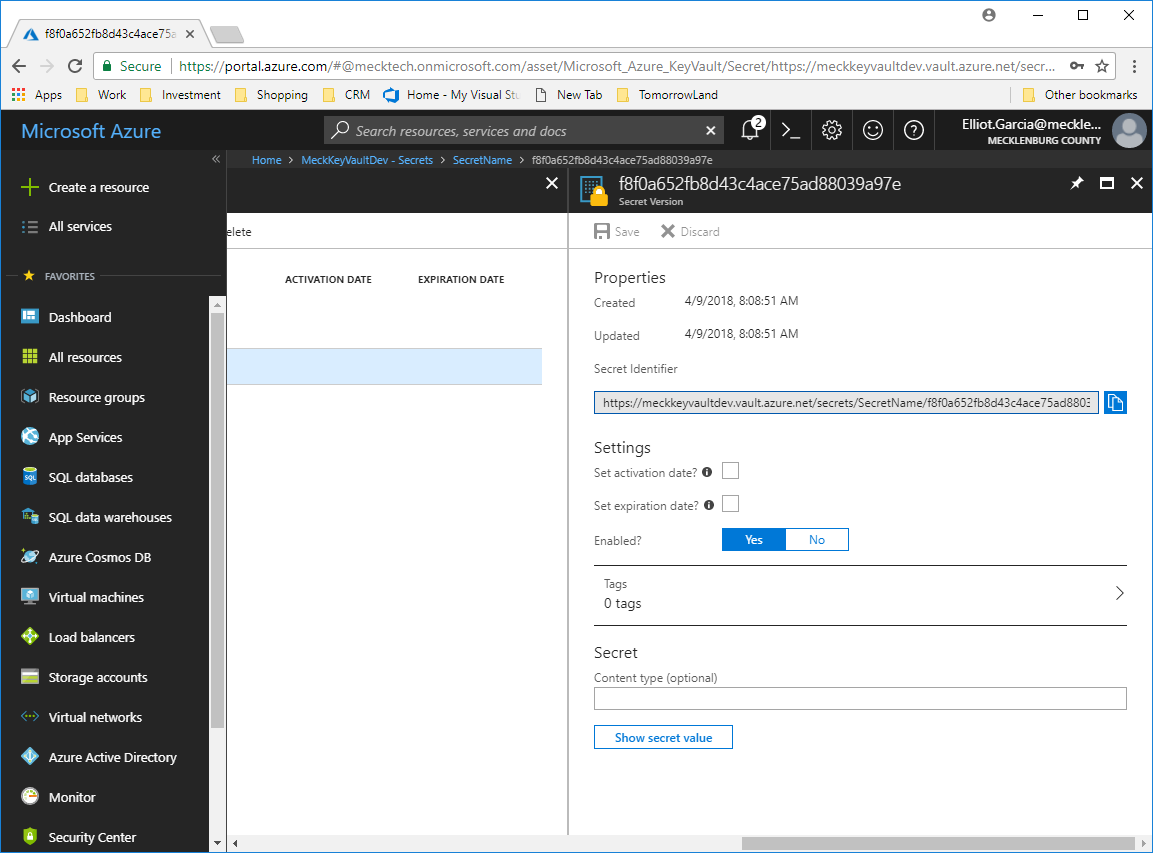
1. Select the newly created **Secret** from the list.



1. Select the version of the newly created **Secret**

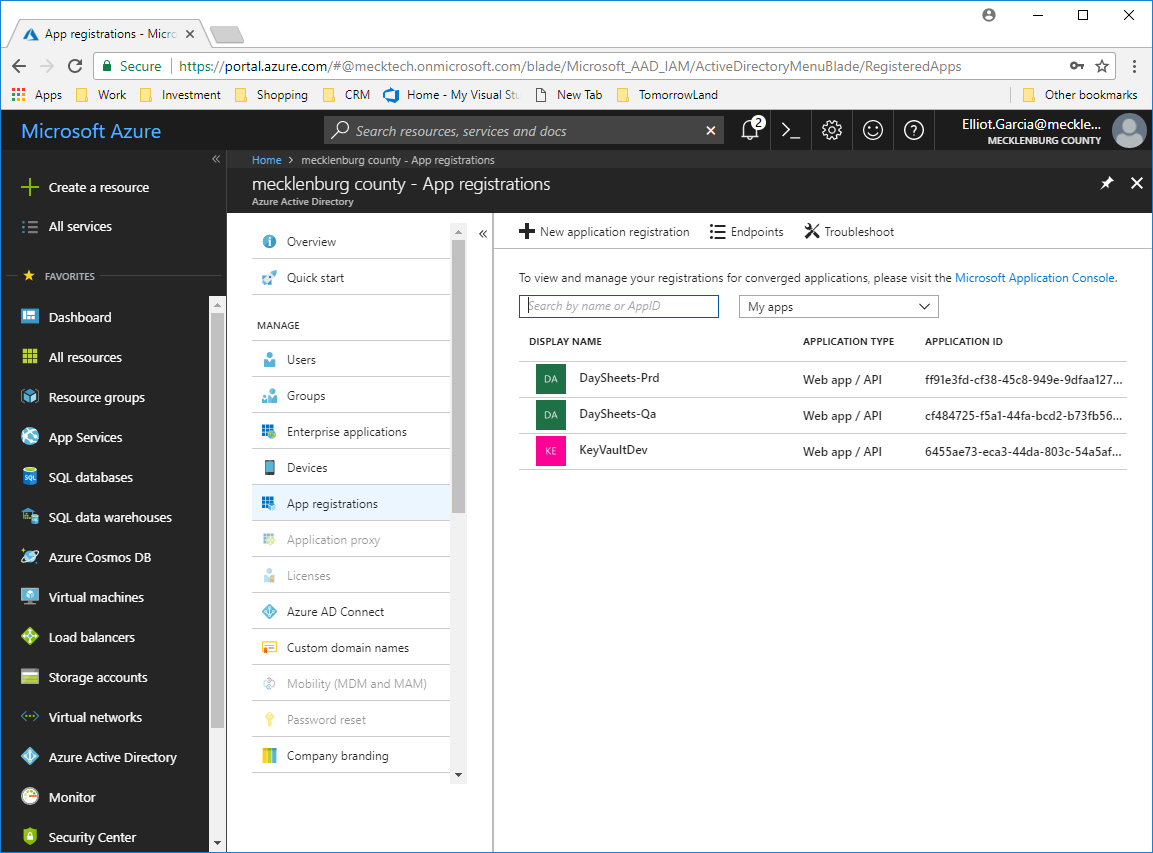


1. The **Secret Identifier** is the URI that will be used by the application to find the secret in Azure

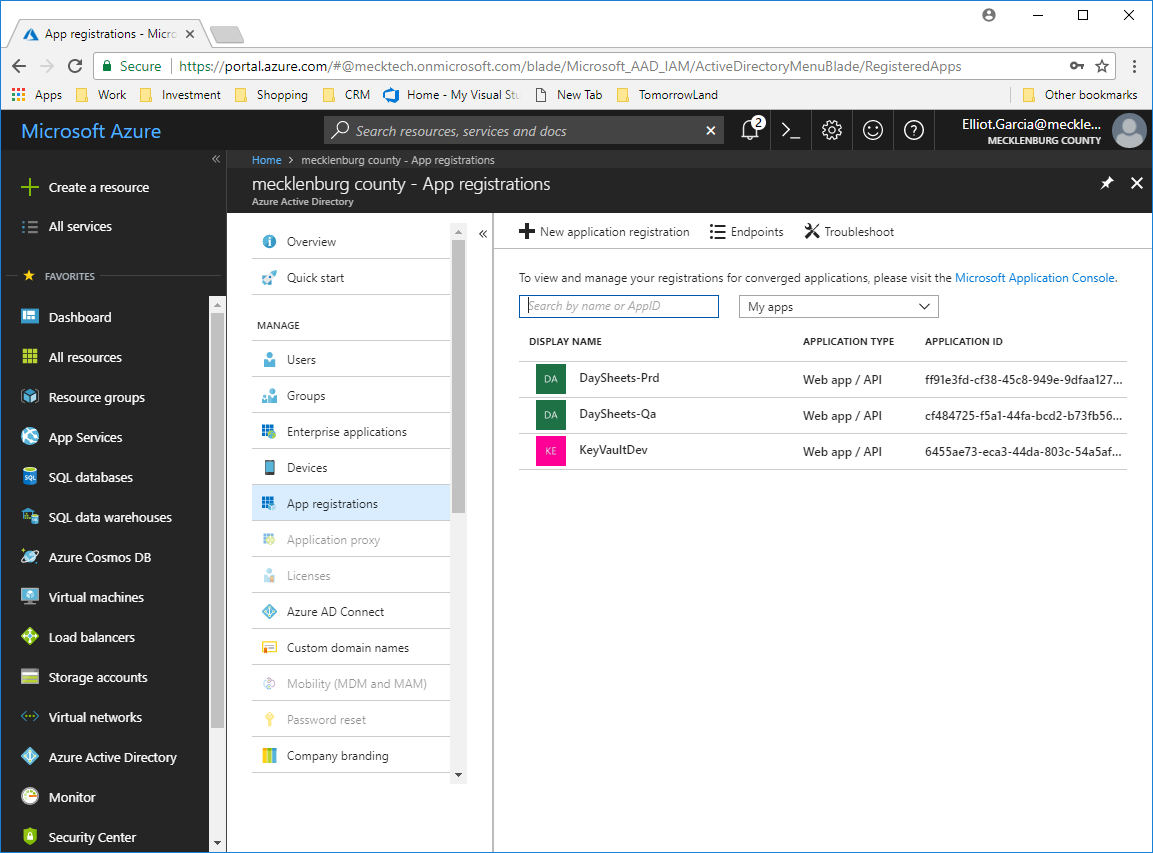


Creating Application Registration

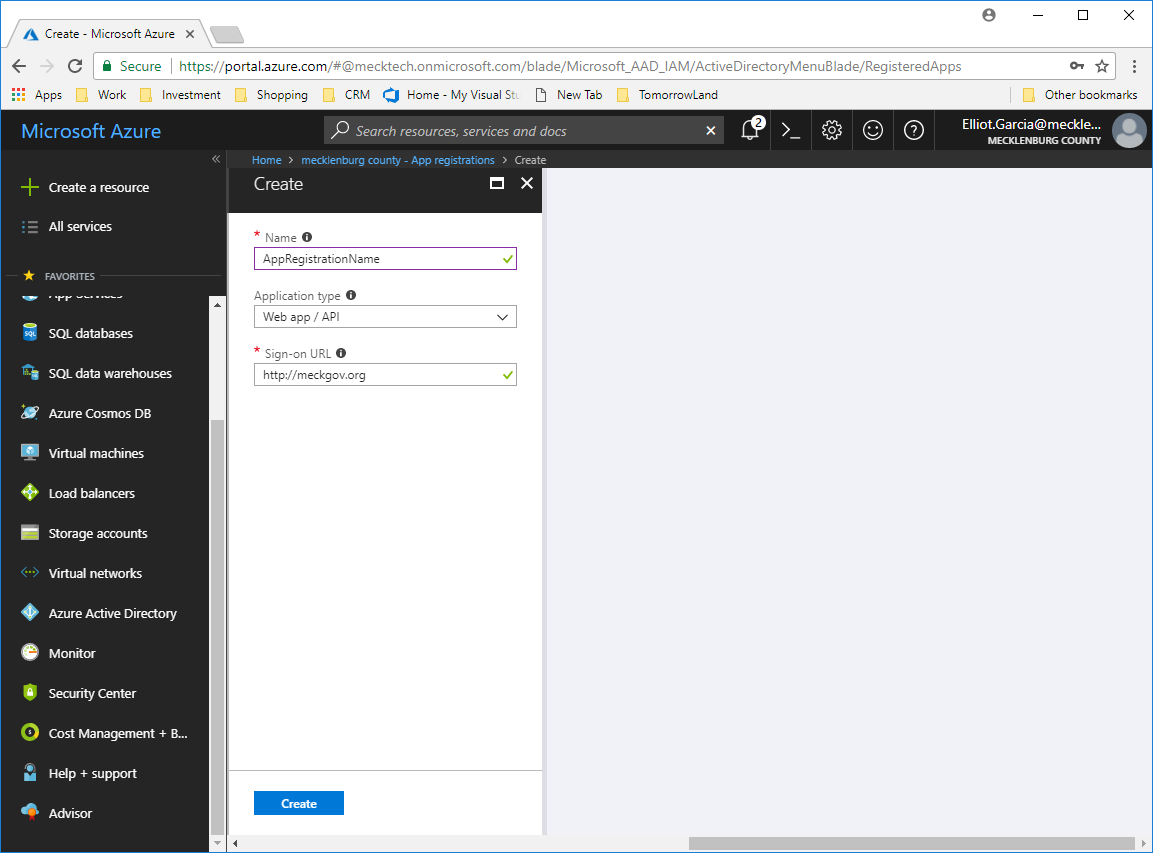
1. Log into the Azure portal and navigate to the **Azure Active Directory** blade.
2. Select **App Registration**



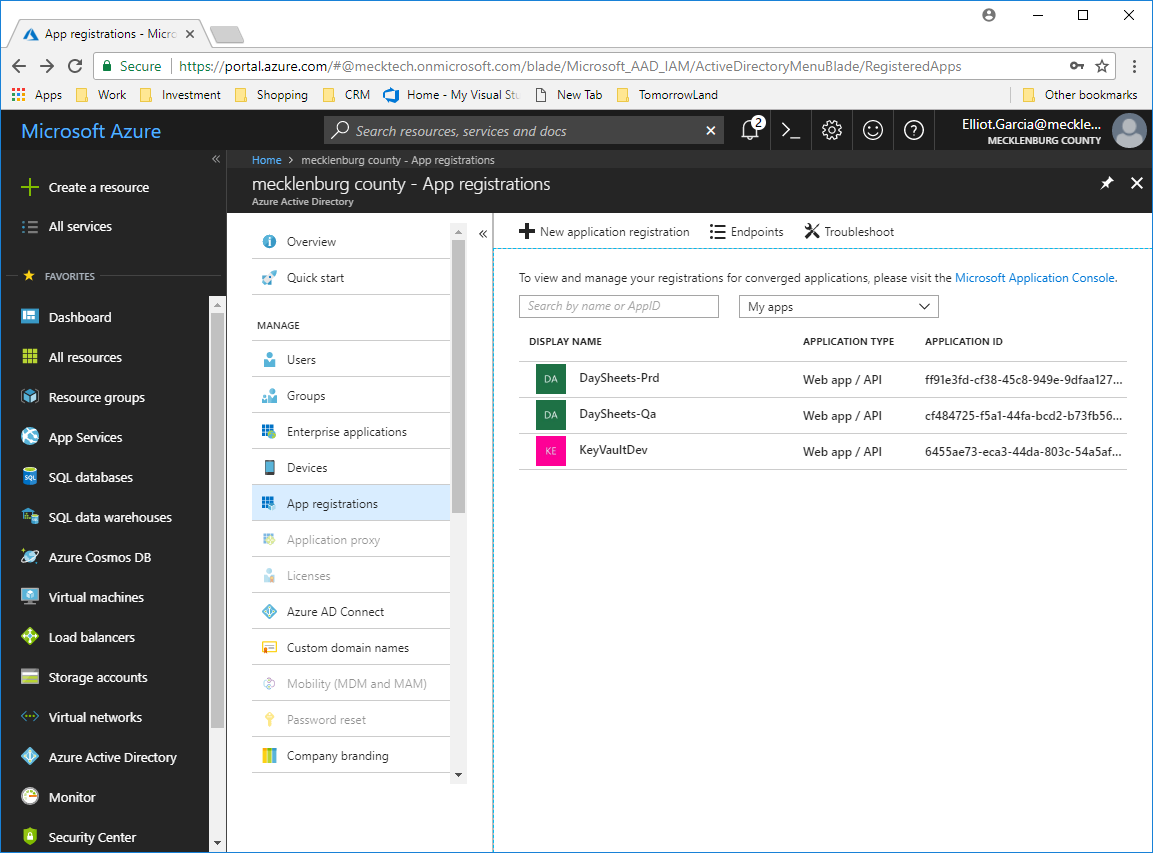
1. Click **New Application Registration**



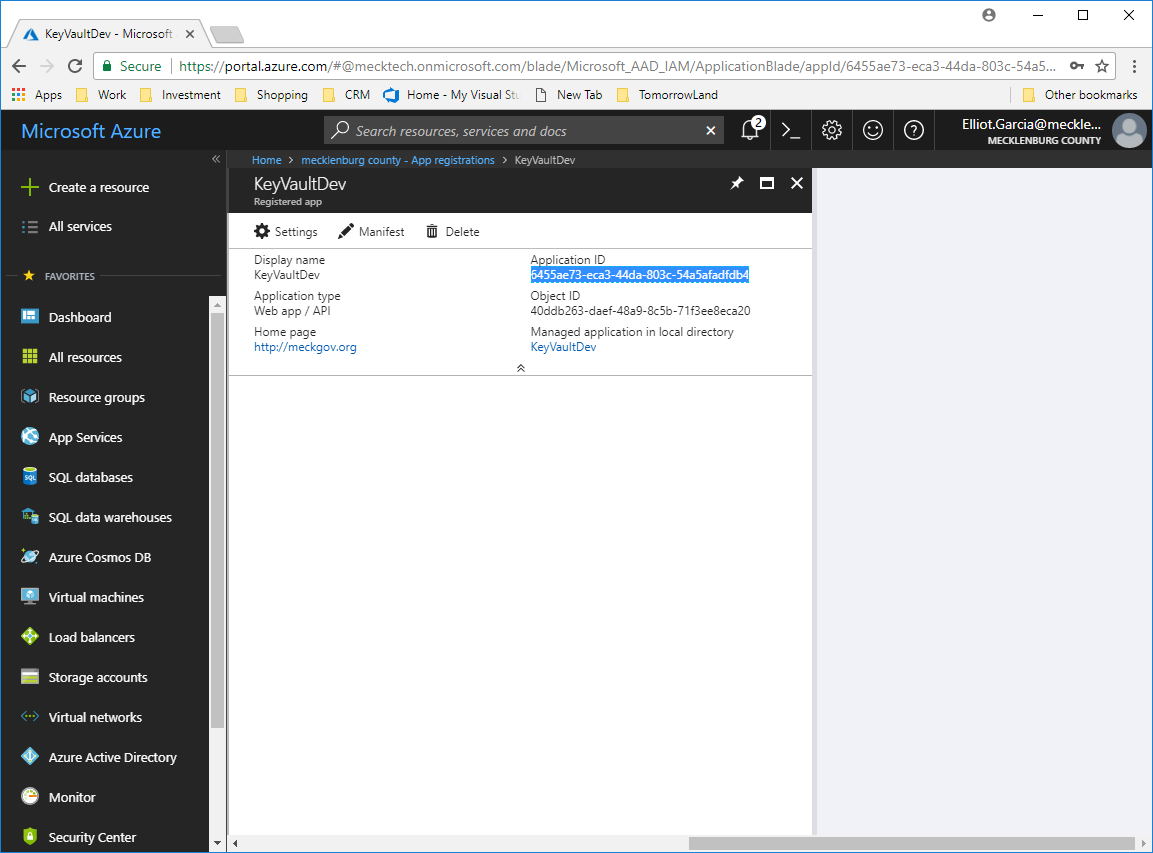
1. Enter a **Name**
2. Select **Application Type** of **Web App / API**
3. Enter a **Sign-on URL** (This can be anything as your application will not be using a sign-on page)



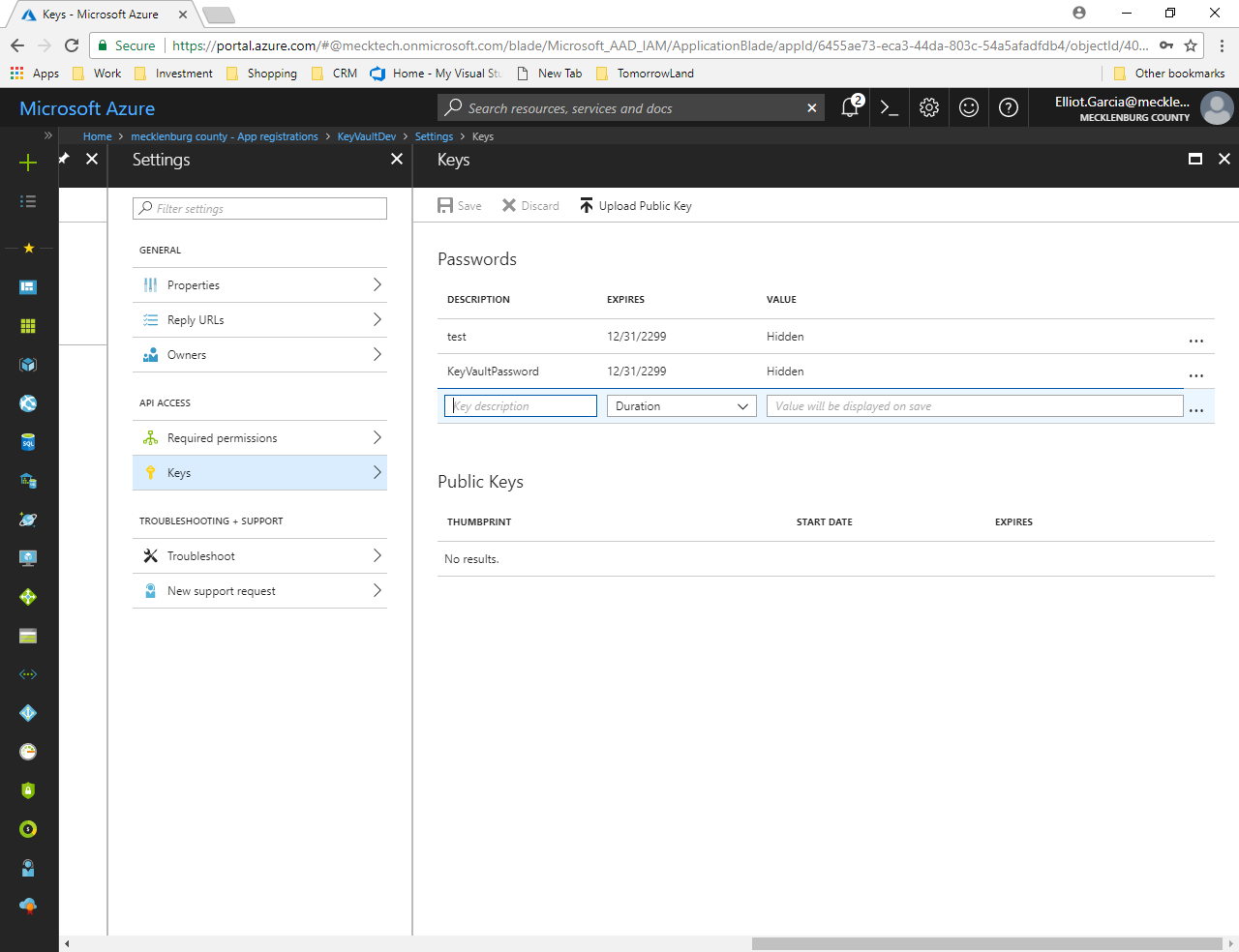
1. Click **Create**
2. Select the newly created **Application Registration**



1. The **Application ID** is going to the ID used in your application to authenticate against Azure

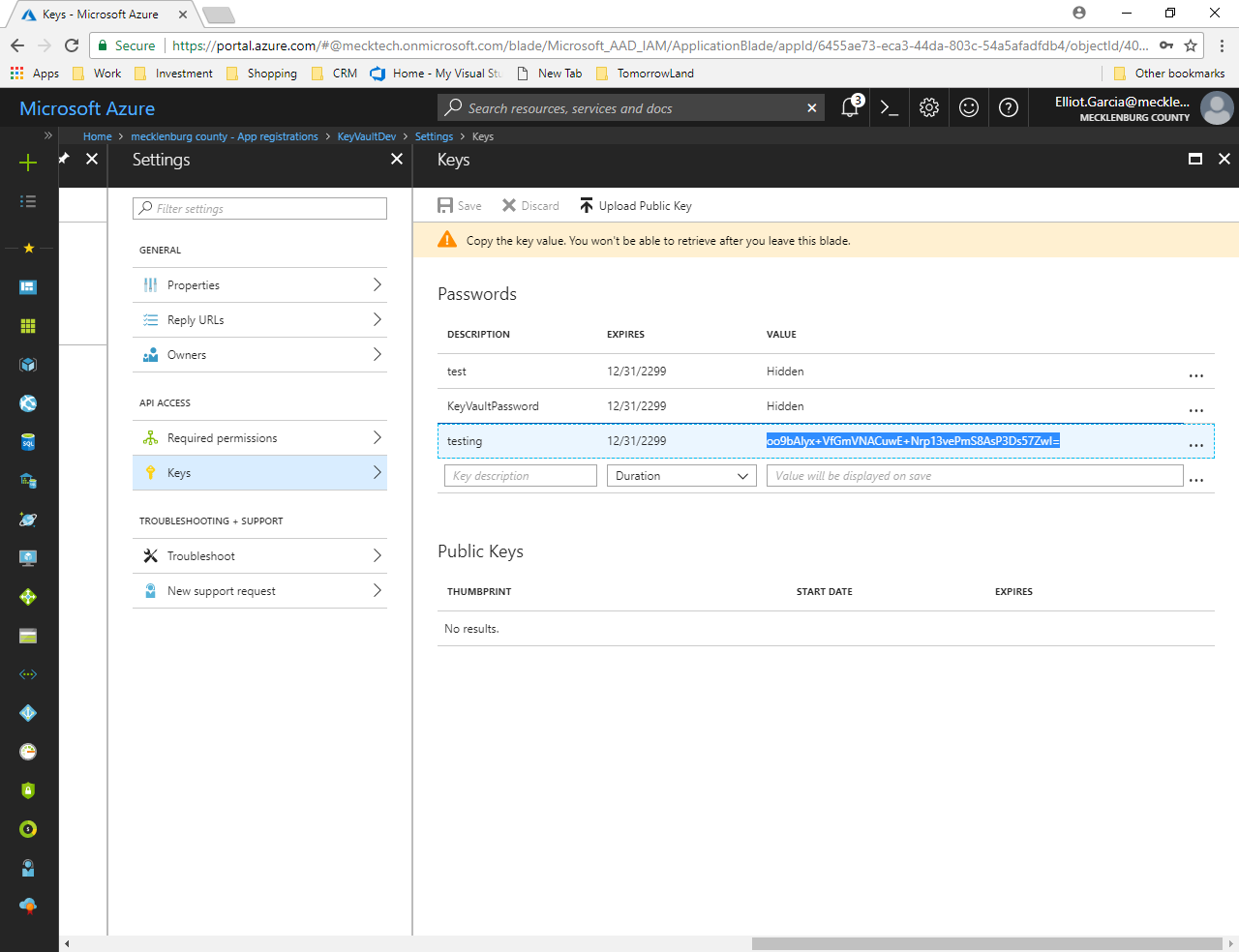


1. Click **Settings**
2. Click **Keys**



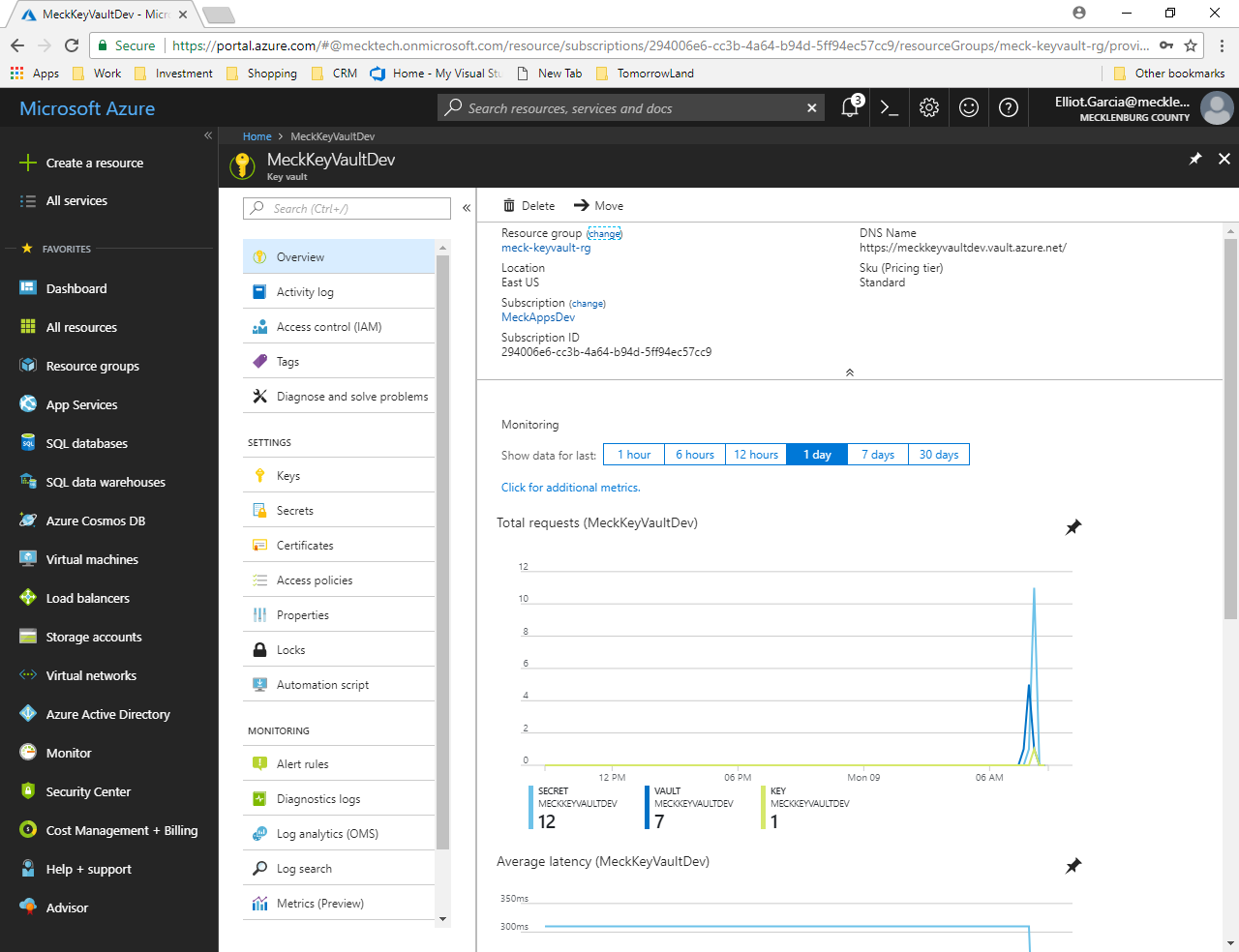
1. Enter **Description**
2. Select **Never Expires** in the **Expires** dropdown
3. Click **Save**

|  |
| --- |
| **Note** |
| A password will be generated. You will have to copy paste this password in a safe location as you will not be able to retrieve this password again once you leave the page. This the password that your application will use to authenticate against Azure to retrieve the secret. |

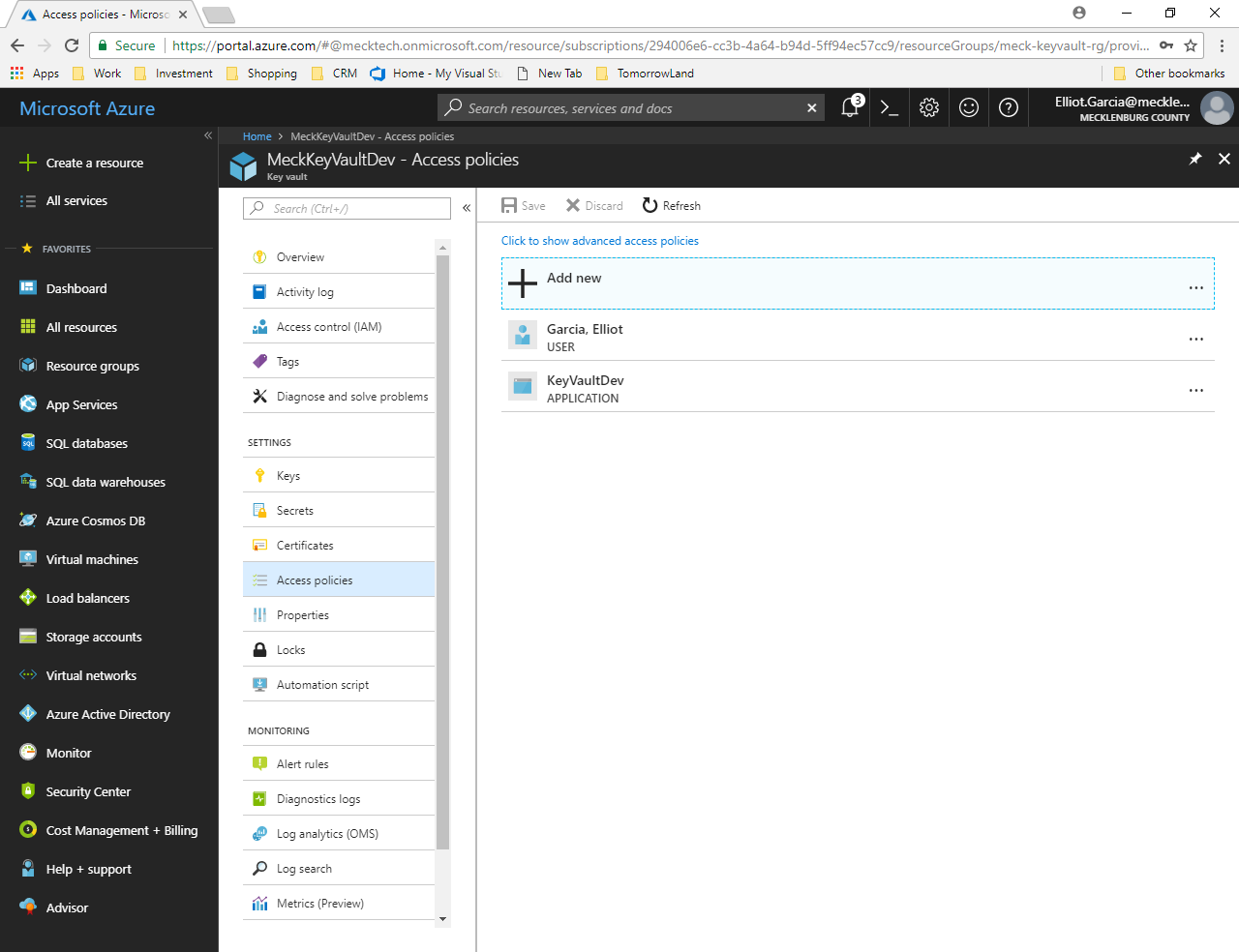


Associating Application Registration w/ Secret

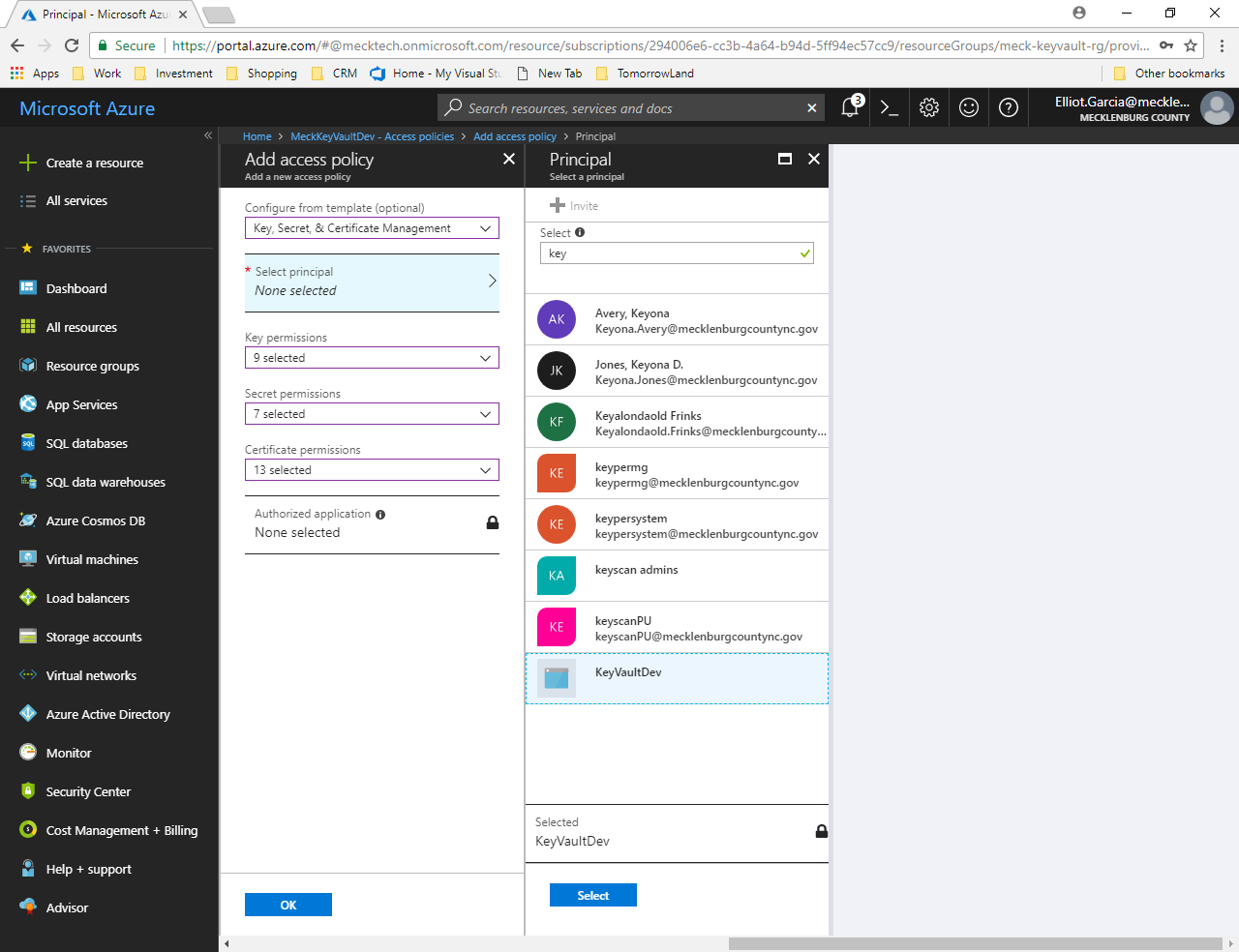
1. Navigate to the **Key Vault**
2. Click **Access Policy**



1. Click **Add New**



1. Select **Key, Secret & Certificate Management** in **Configure from Template**
2. Click **Select Principal**
3. Search for **Application Registration** created in the previous step



1. Click **Select**
2. Configure the **Permissions** for **Keys, Secrets and Certificates**
3. Click **OK**

Application Web.config

<appSettings>

**<!-- Azure Information: Azure Active Directory / App Registration / Application ID -->**

<add key="KeyVaultApplicationId" value="6455ae73-eca3-44da-803c-54a5afadfdb4" />

**<!-- Azure Information: Azure Active Directory / App Registration / Keys / ClientSecret -->**

<add key="KeyVaultApplicationPassword" value="VE29OnAVLe52InTiIj2F445TtZ0o3TEC9zUH0xl21b4=" />

**<!-- Azure Information: KeyVault Needed / Secrets / Keys / Secret Needed / Version Needed / Secret Identifier -->**

<add key="KeyVaultSecretIdentifier" value="https://meckkeyvaultdev.vault.azure.net/secrets/ConnectionStringDev/830380c7a0d247d7bf033947558bd587" />

</appSettings>

MeckKeyVault Application Password EwDEz87mF7TjvRNFDuVpCYktdCpsn2h1W4ulwgr2S48=

.Net Code

using Microsoft.Azure.KeyVault;

using Microsoft.IdentityModel.Clients.ActiveDirectory;

using System;

using System.Threading.Tasks;

using System.Web.Configuration;

namespace Meck.Azure.KeyVault

{

public class Utility

{

public static string EncryptedSecret { get; set; }

private static async Task<string> GetToken(string authority, string resource, string scope)

{

var authContext = new AuthenticationContext(authority);

ClientCredential clientCred = new ClientCredential(WebConfigurationManager.AppSettings["KeyVaultApplicationId"],

WebConfigurationManager.AppSettings["KeyVaultApplicationPassword"]);

AuthenticationResult result = await authContext.AcquireTokenAsync(resource, clientCred);

if (result == null)

throw new InvalidOperationException("Failed to obtain the JWT token");

return result.AccessToken;

}

public static async Task<string> GetSecret()

{

var keyVaultClient = new KeyVaultClient(new KeyVaultClient.AuthenticationCallback(GetToken));

var secret = await keyVaultClient.GetSecretAsync(WebConfigurationManager.AppSettings["KeyVaultSecretIdentifier"]);

EncryptedSecret = secret.Value;

return EncryptedSecret;

}

}

}