Cyber Shujaa

Cloud Security Specialist

Assignment 20 - Lab 7: Key Vault (Implementing Secure Data by setting up Always Encrypted)

NAME: WELDON KIPKIRUI KENEI

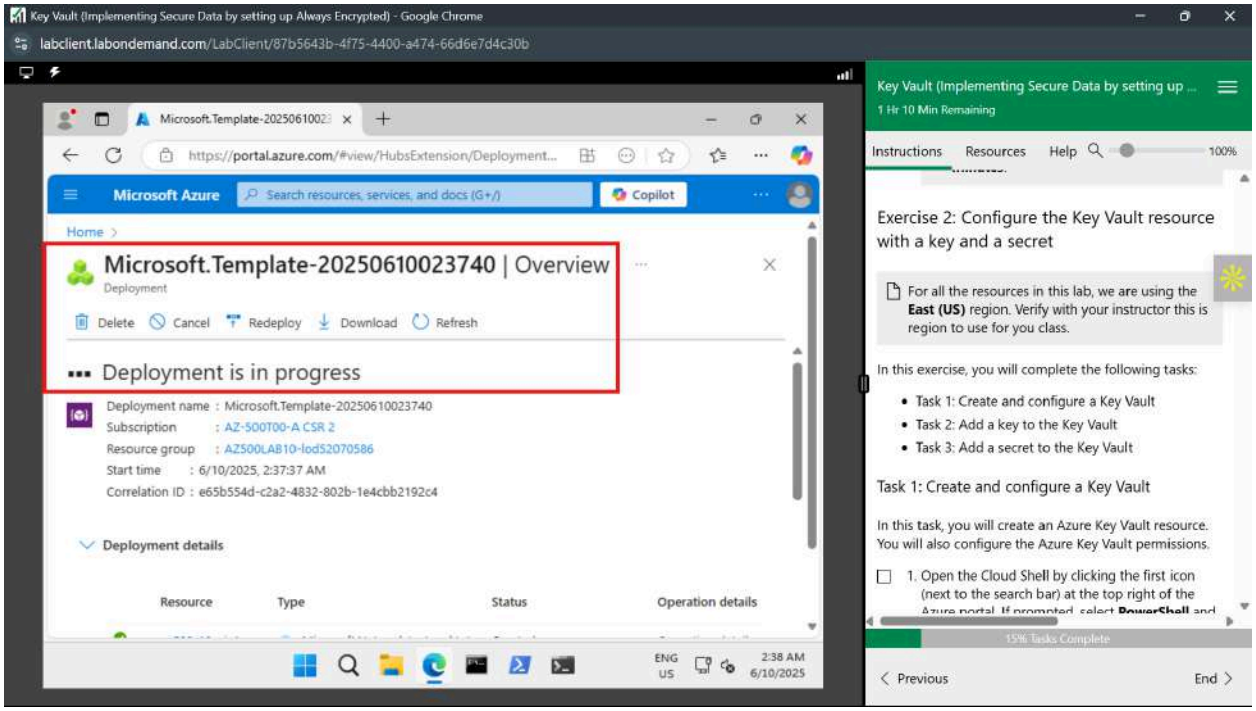
CS NO: ADC-CSS02-25027

Introduction

In this lab scenario, we will create a proof-of-concept application that uses the Azure SQL Database's support for Always Encrypted functionality. All of the secrets and keys used in this scenario will be stored in Key Vault. The application should be registered in Microsoft Entra ID to enhance its security posture.

Deploy the base infrastructure from an ARM template

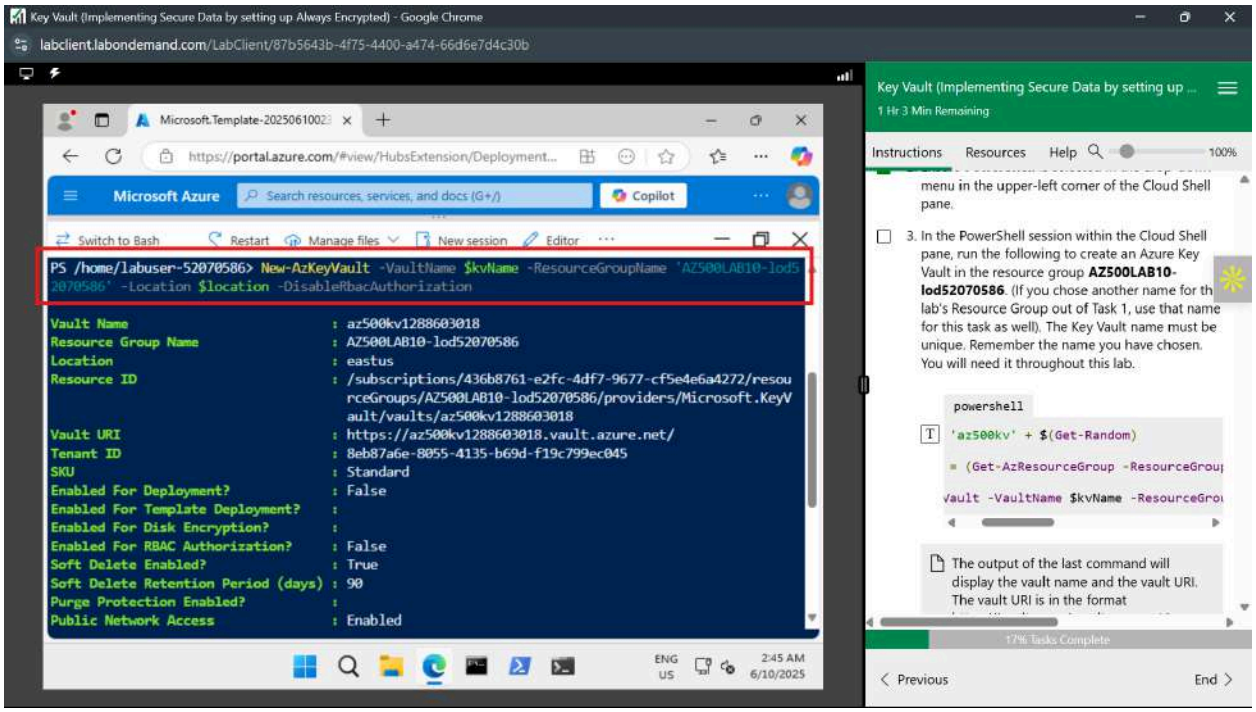
In this step, we will deploy an Azure SQL database and an Azure VM using an existing ARM template, which will automatically install Visual Studio 2019 and SQL Server Management Studio 19 as part of the deployment.



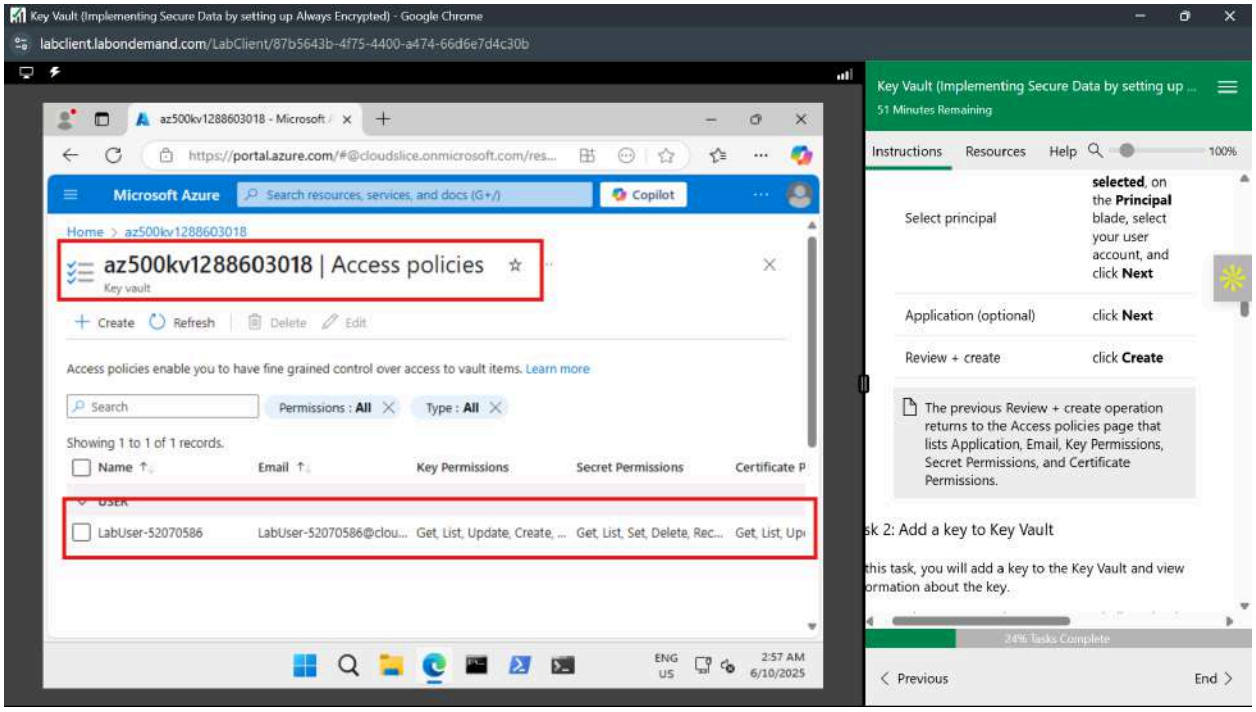
Configure the Key Vault resource with a key and a secret

In this step, we will configure an Azure Key Vault and add a key and a secret to the Key Vault. Use these commands in the Cloud Shell.

```
$kvName = 'az500kv' + $(Get-Random)
$location = (Get-AzResourceGroup -ResourceGroupName 'AZ500LAB10-lod52070586').Location
New-AzKeyVault -VaultName $kvName -ResourceGroupName 'AZ500LAB10-lod52070586' -Location $location -DisableRbacAuthorization
```

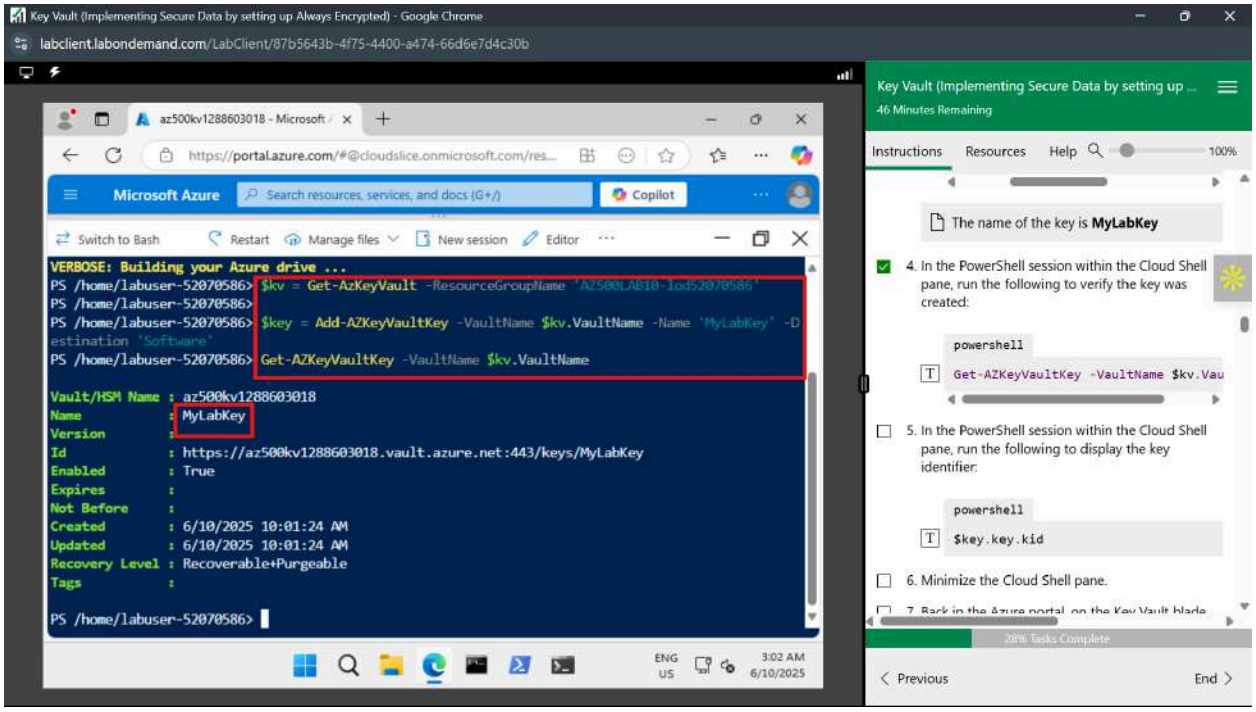


We will then create the access policies for our key vault from the Azure portal as shown below.



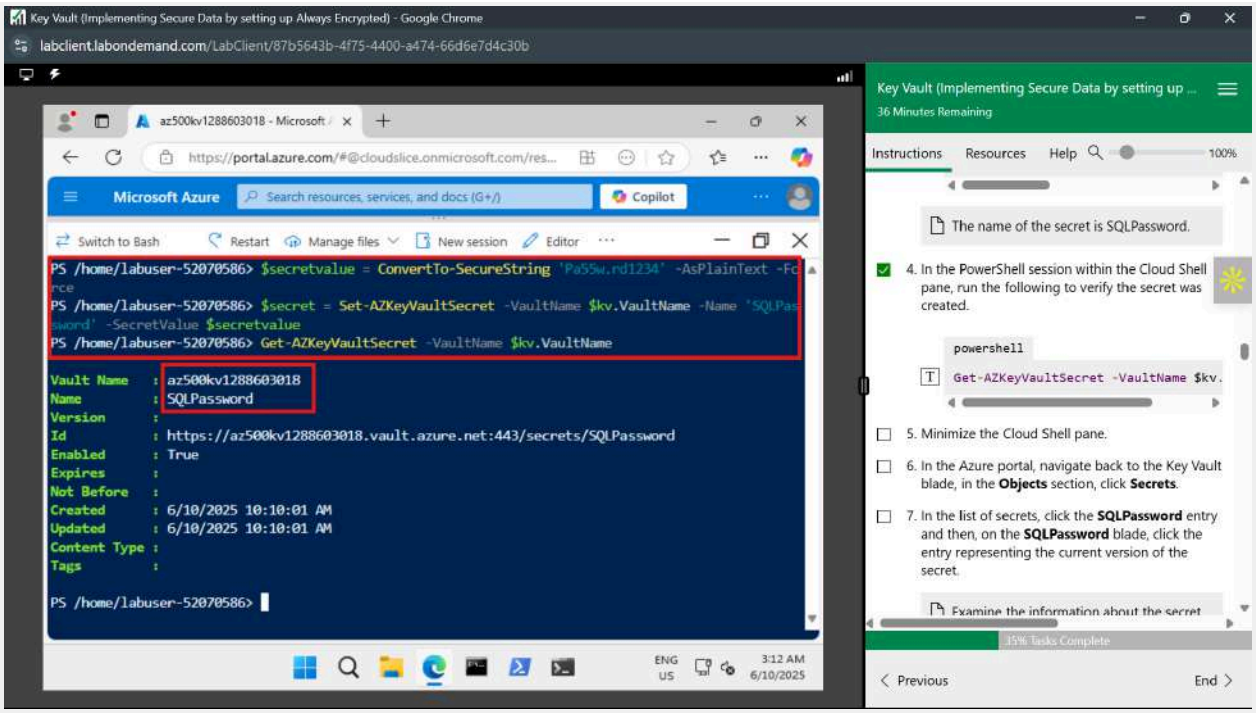
Next, we will run the following to add a software-protected key to the Key Vault:

```
$kv = Get-AzKeyVault -ResourceGroupName 'AZ500LAB10-lod52070586'
$key = Add-AZKeyVaultKey -VaultName $kv.VaultName -Name 'MyLabKey' -Destination 'Software'
```



We then add a secret to our KeyVault. Using the Cloud Shell, we run the following commands;

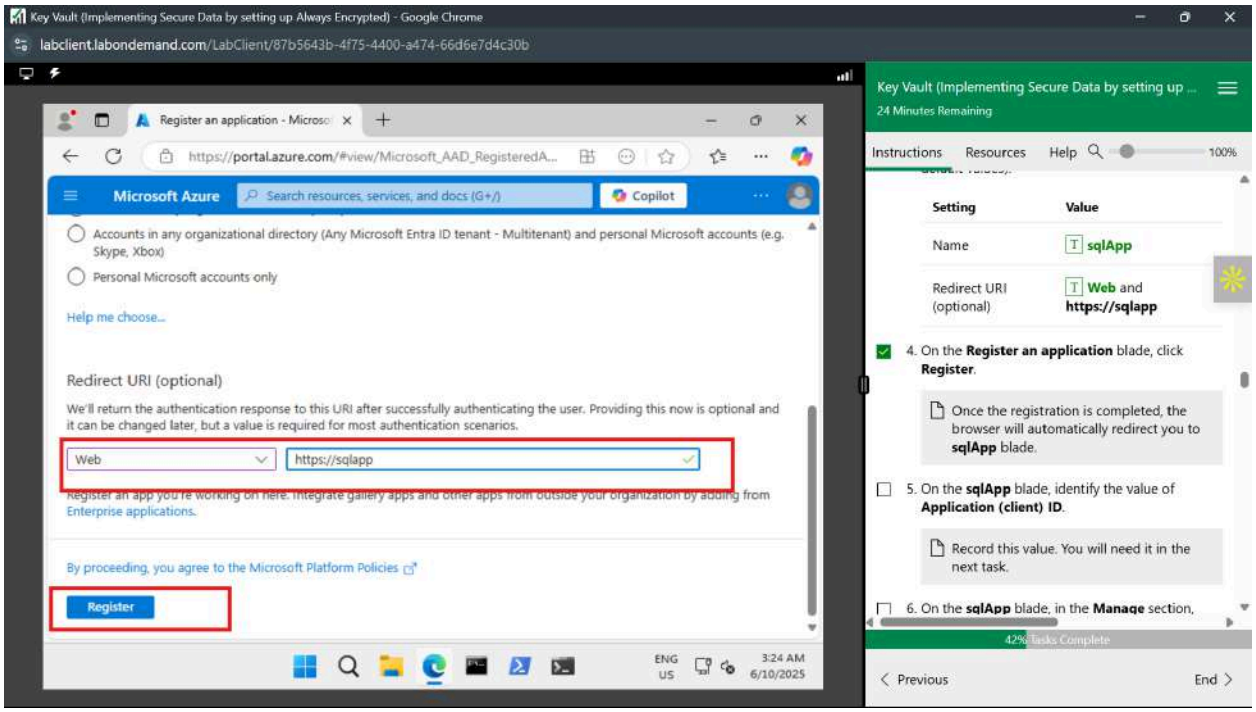
1. `$secretvalue = ConvertTo-SecureString 'Pa55w.rd1234' -AsPlainText -Force`
2. `$secretvalue = ConvertTo-SecureString 'Pa55w.rd1234' -AsPlainText -Force`



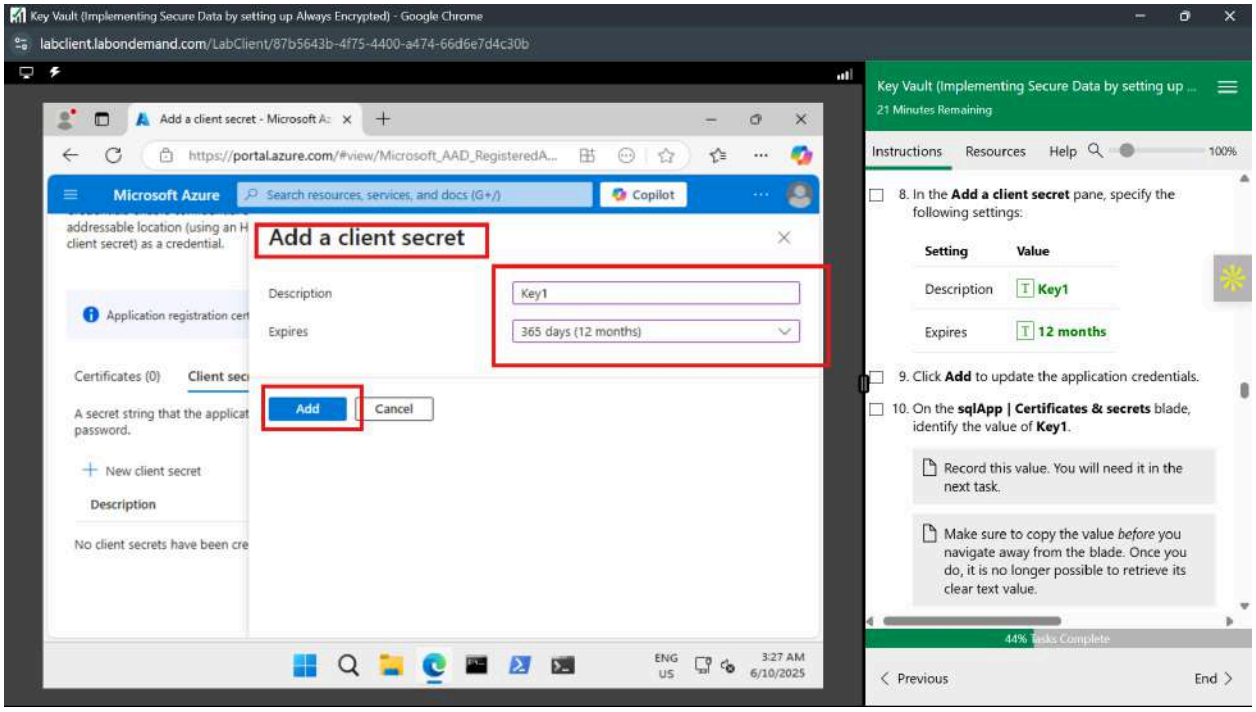
Configure an Azure SQL database and a data-driven application

In this task, we will enable a client application to access the Azure SQL Database service, create a policy allowing the application access to the Key Vault, and retrieve the SQL Azure database ADO.NET Connection String, log on to the Azure VM running Visual Studio 2019 and SQL Management Studio 19 ND Create a table in the SQL Database and select data columns for encryption.

First, we will register an application, in this case, sqlapp, as shown below, using the app registration service in the Azure portal.



We then configure the app secret key as shown below.

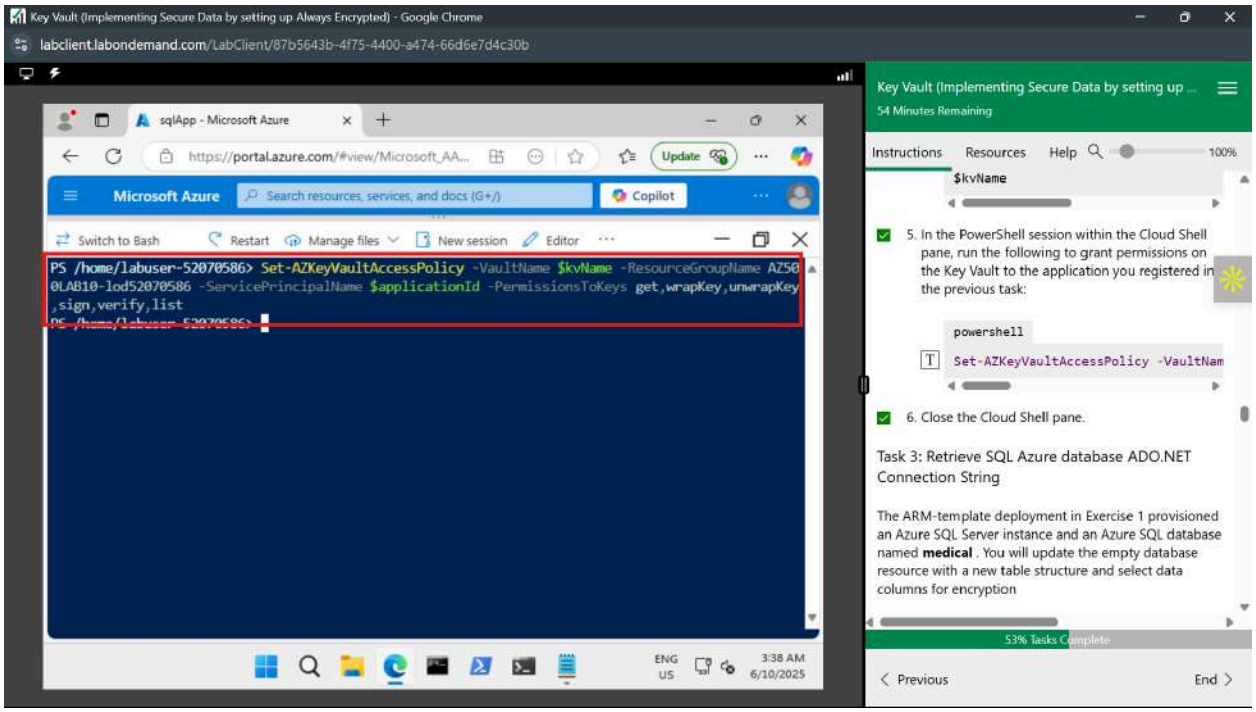


Next, we will grant the newly registered app permissions to access secrets stored in the Key Vault. We will run the following to create a variable storing the Application (client) ID we recorded in the previous tasks.

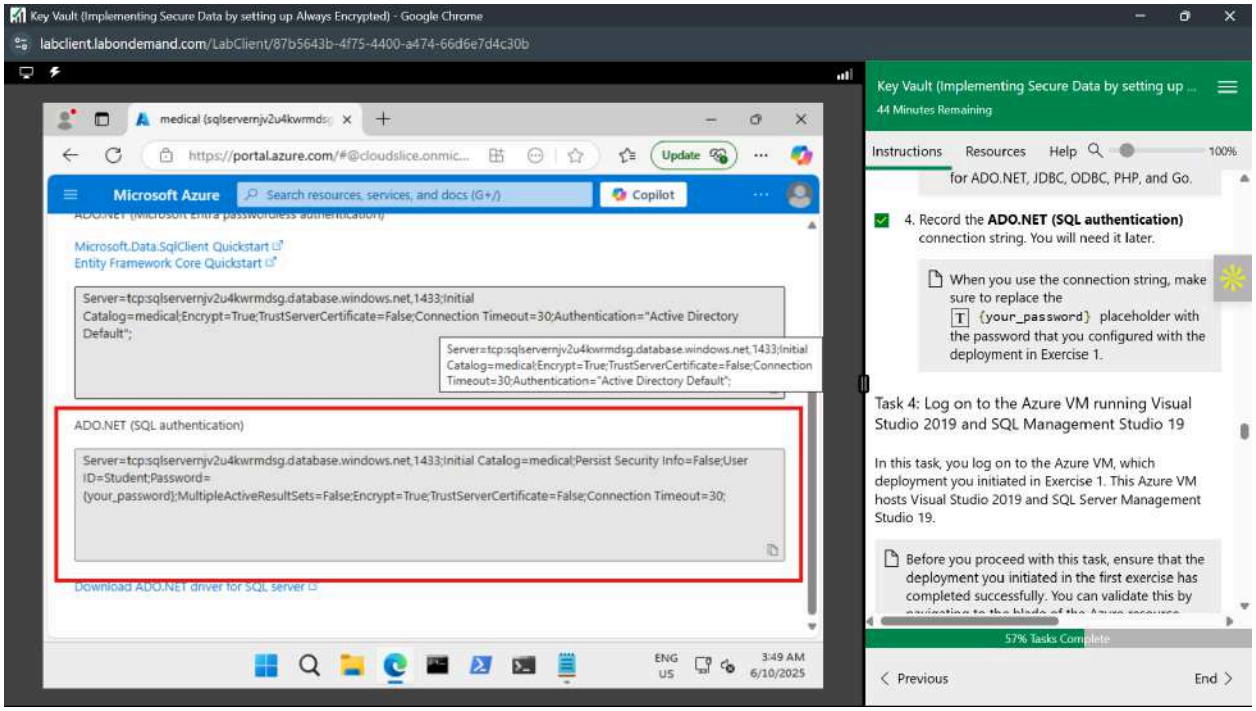
```
$applicationId = '<Azure_AD_Application_ID>'
```

We will run the following to grant permissions on the Key Vault to the application we registered in the previous task:

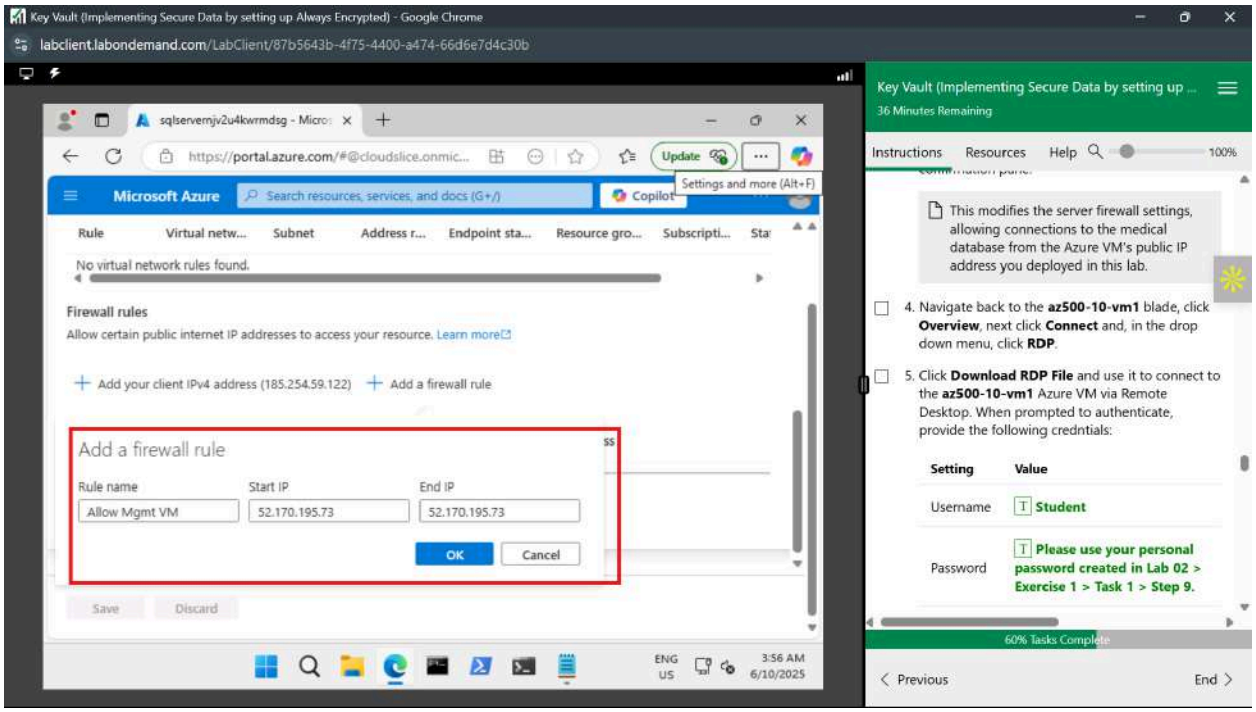
```
Set-AZKeyVaultAccessPolicy -VaultName $kvName -ResourceGroupName AZ500LAB10-lod52070586 -ServicePrincipalName $applicationId -PermissionsToKeys get,wrapKey,unwrapKey,sign,verify,list
```



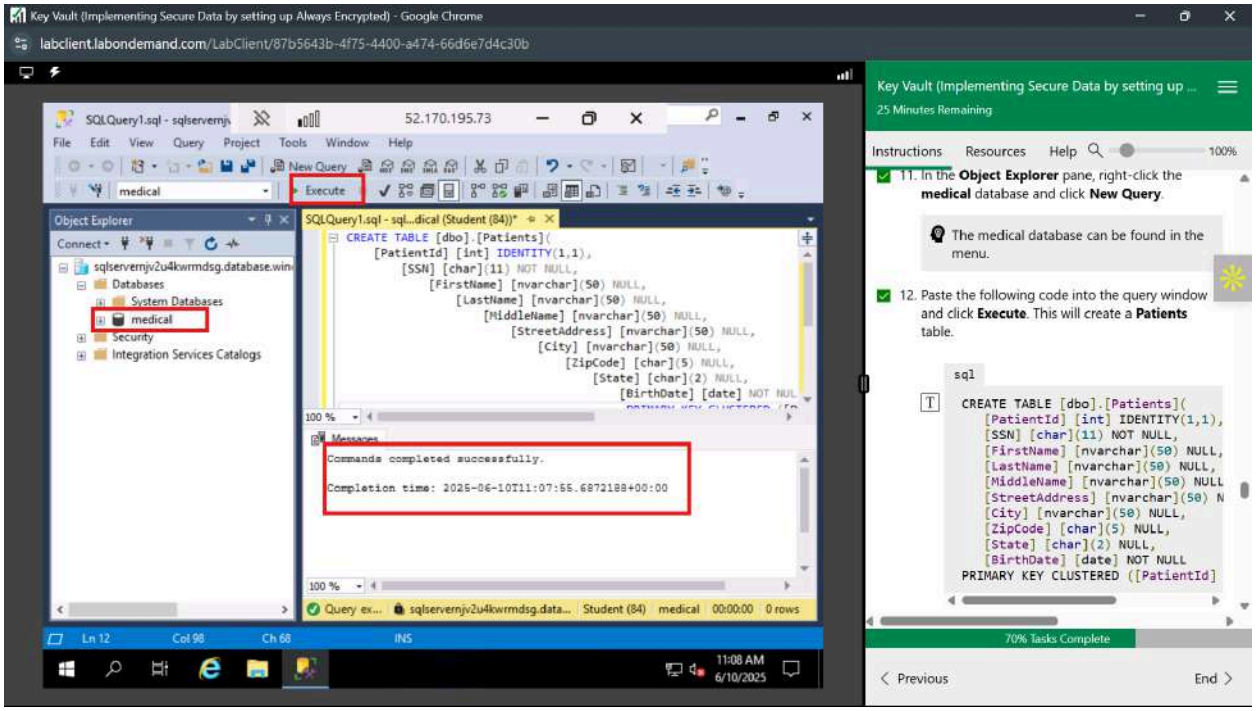
Since we have an empty database provisioned by the ARM template we deployed earlier, we can update it with a new table structure and select data columns for encryption. First, we will retrieve the ADO.NET string for SQL authentication as shown;



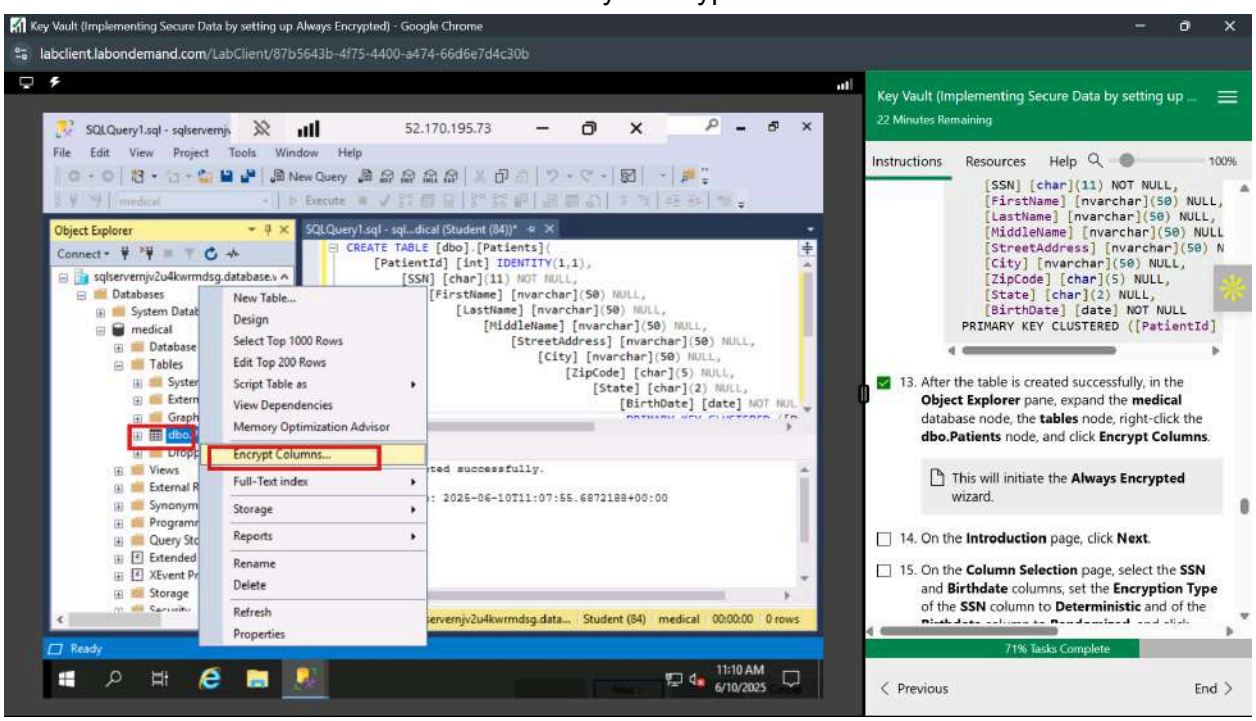
Now we can configure our SQL database firewall rules to allow our virtual machine that was deployed earlier to publicly access it using the following firewall rules.

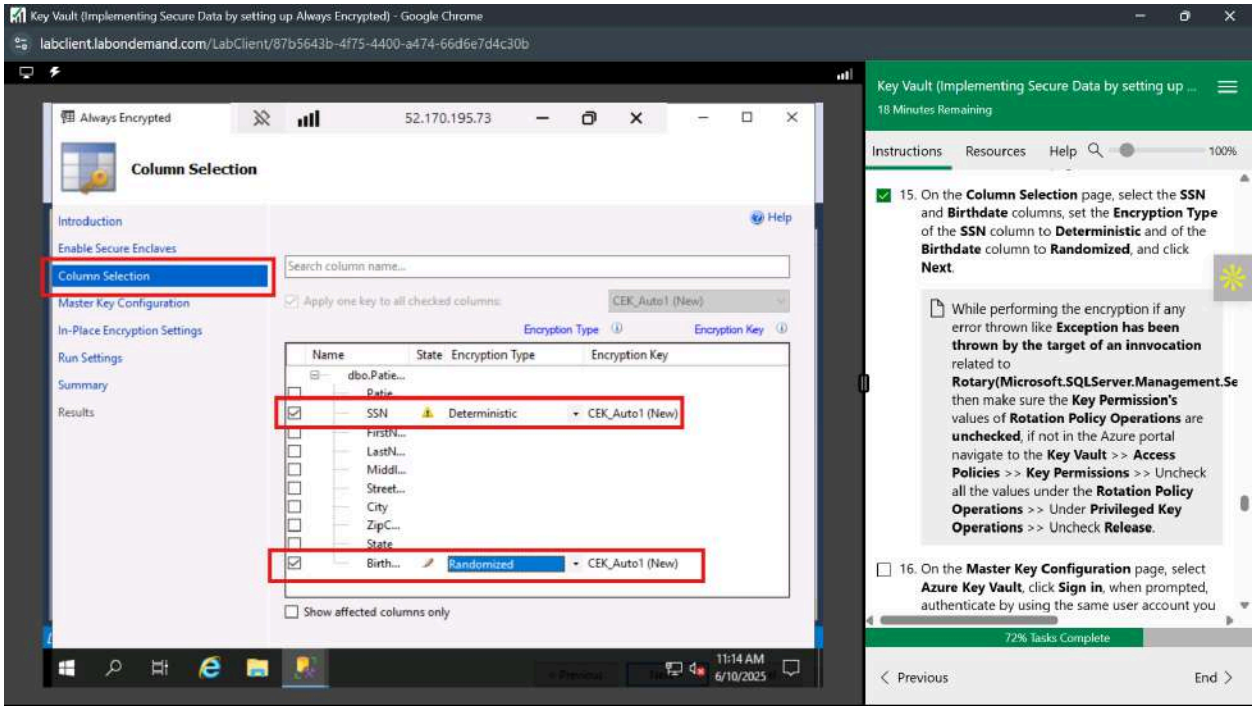


From our VM, we can connect to the database using the SQL Server Management Studio app installed. We then interact with our database with queries to update the table structures as shown;

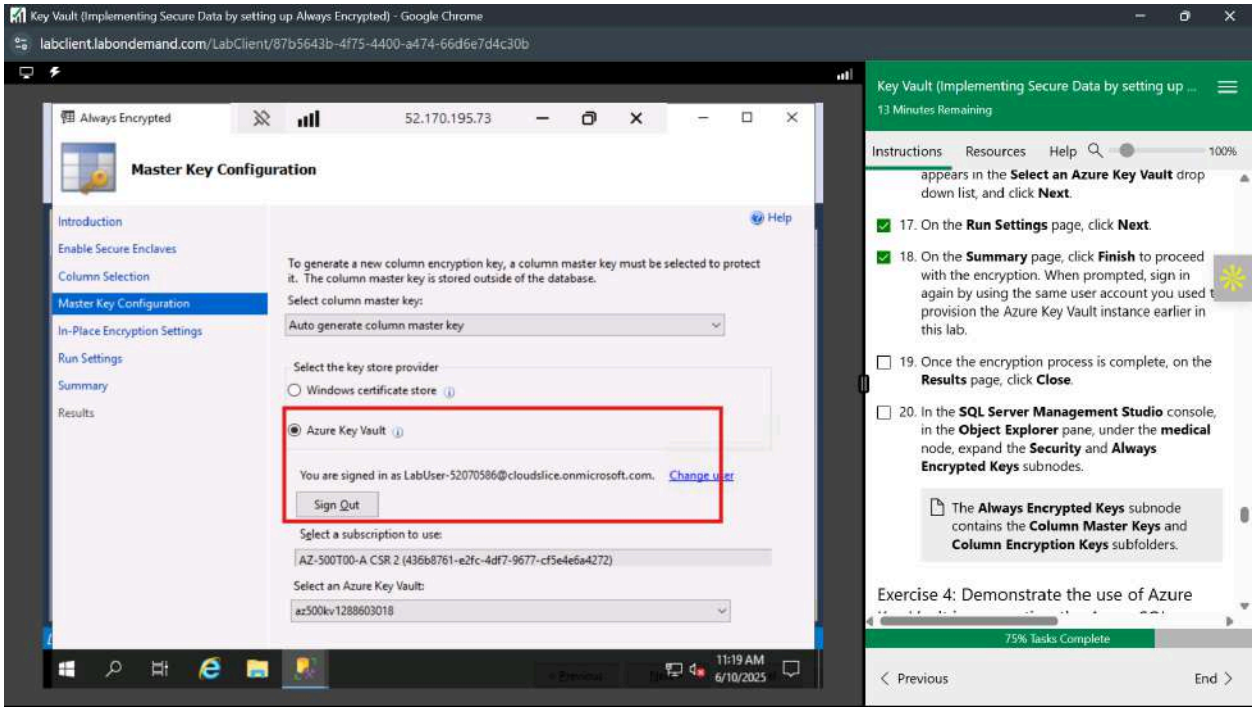


We will select a table and columns in it to always encrypt.

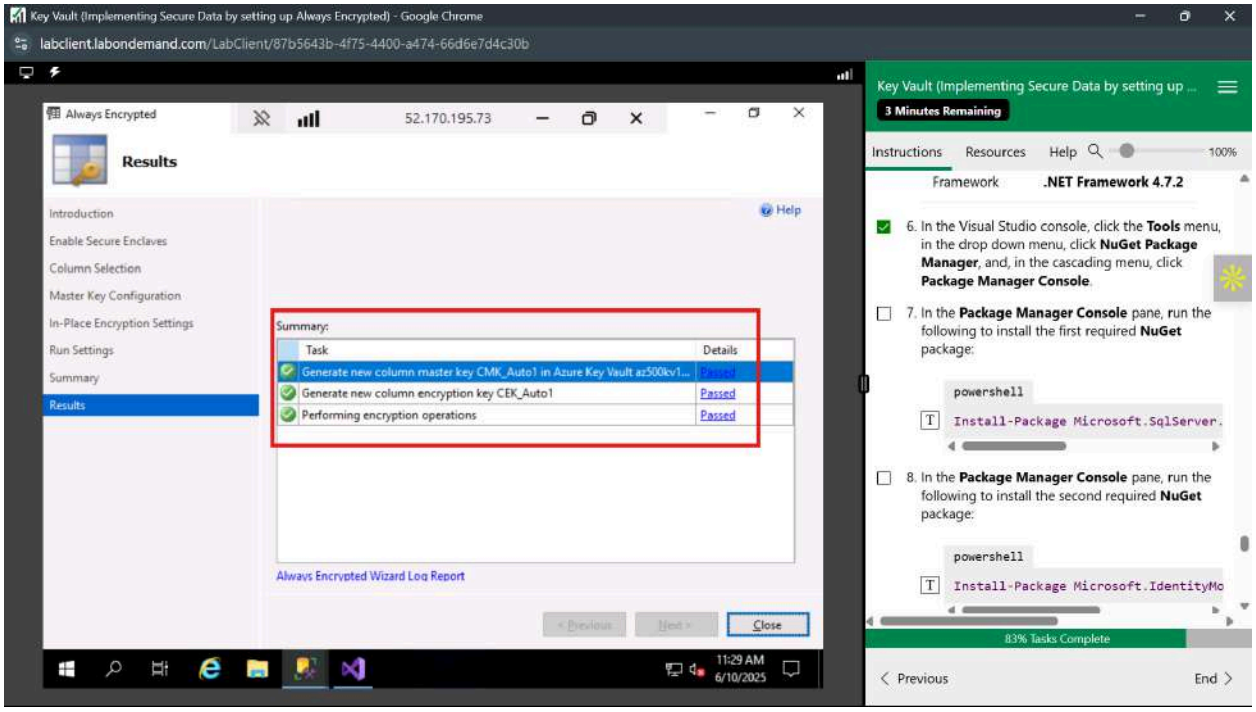


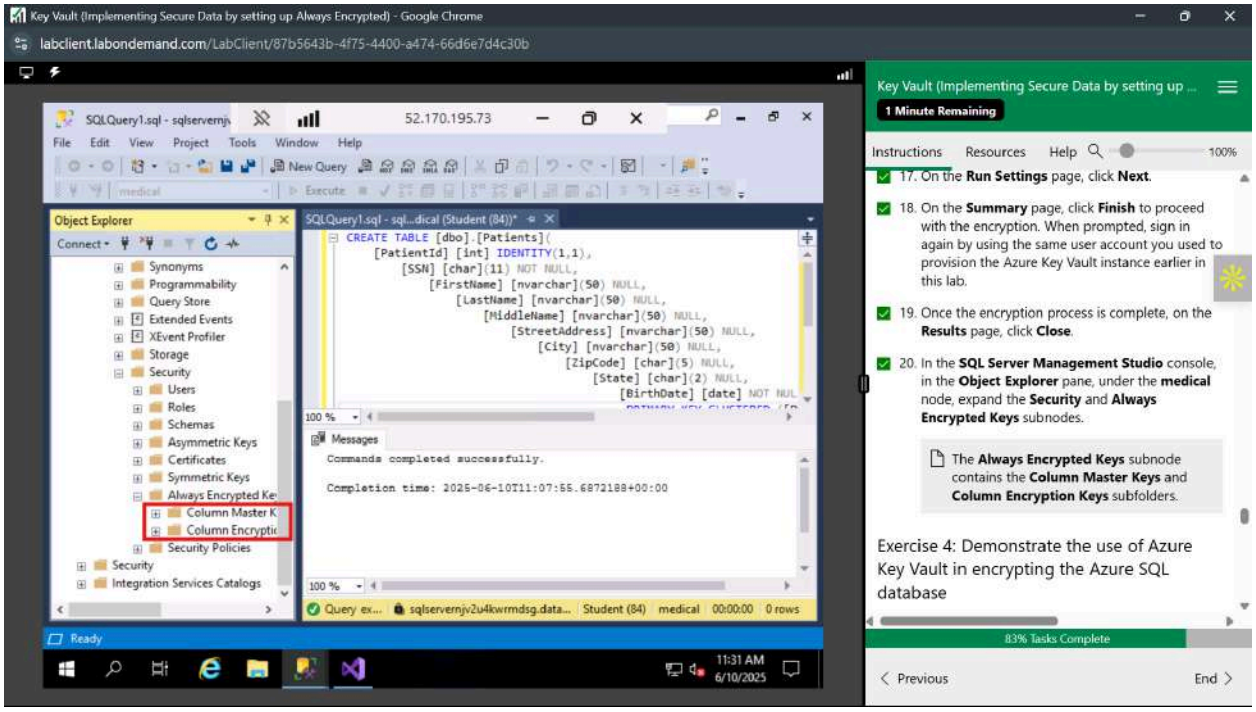


Next, we configure the master encryption key that is stored outside our database and in the key vault as shown below;



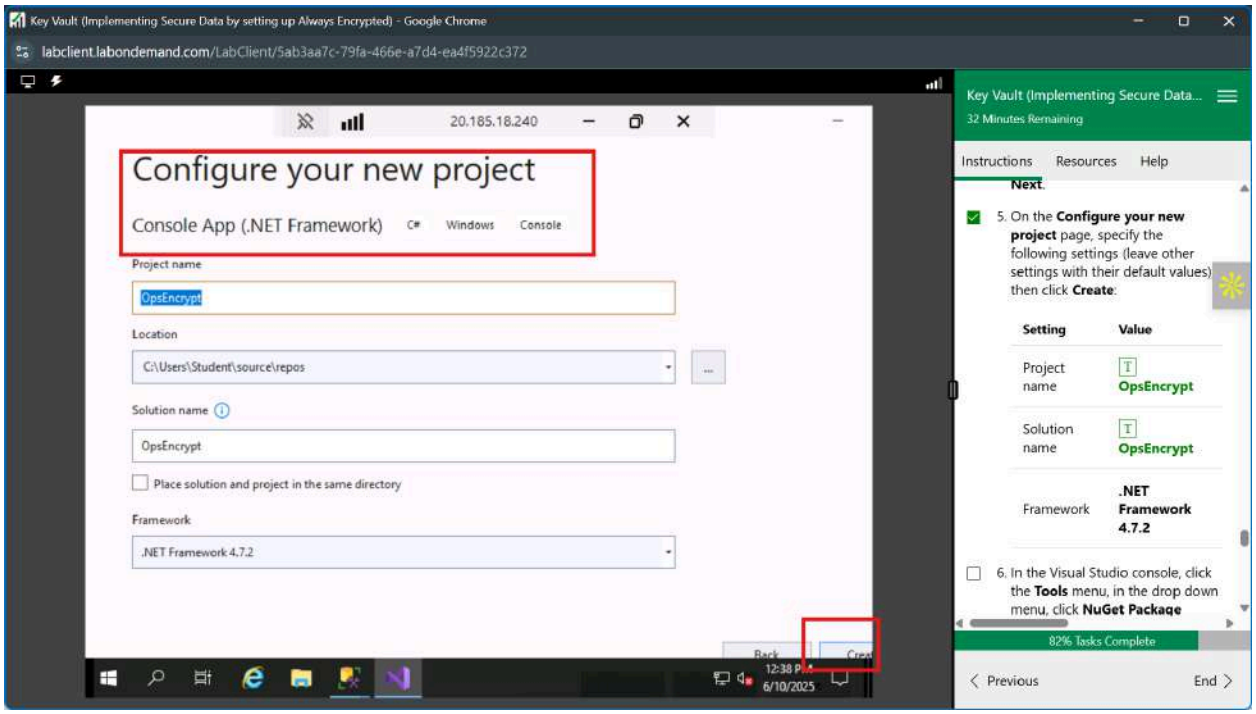
From the result we can see that every section and encryption attempt passed.





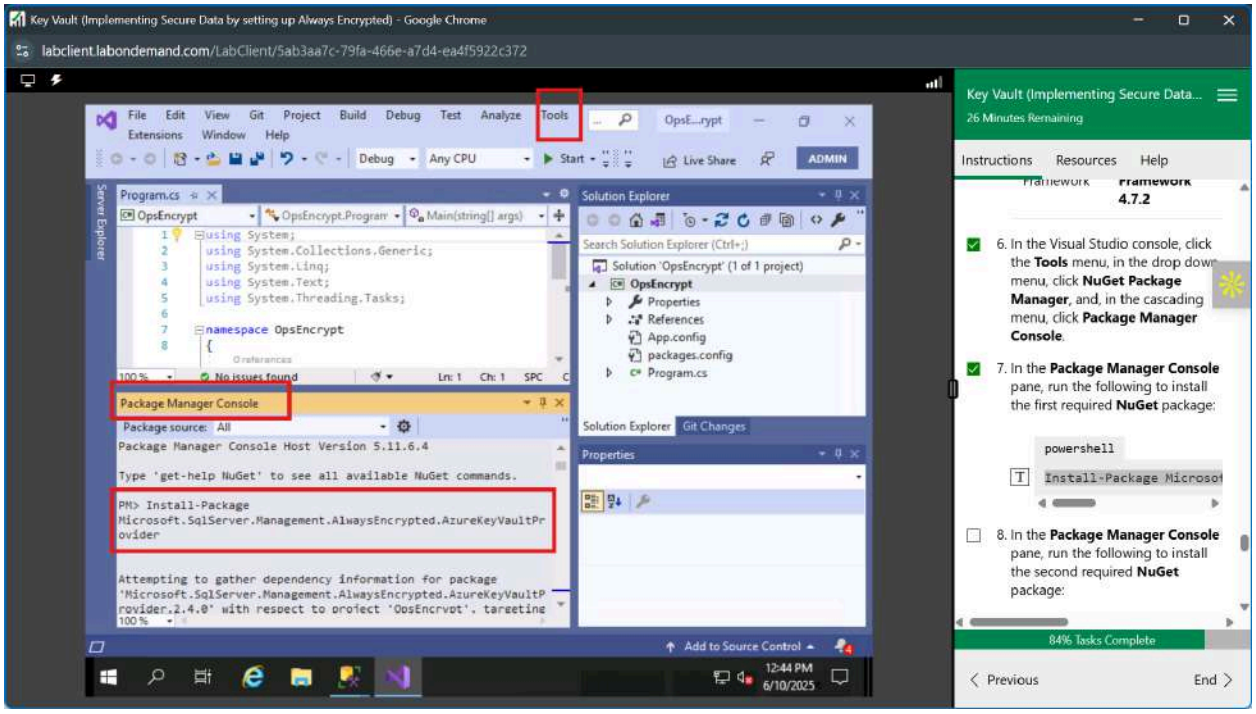
Demonstrate the use of Azure Key Vault in encrypting the Azure SQL database

In this task, we will run a data-driven application to demonstrate the use of Azure Key Vault in encrypting the Azure SQL database. We will create a console application using Visual Studio to load data into the encrypted columns and then access that data securely using a connection string that accesses the key in the Key Vault.

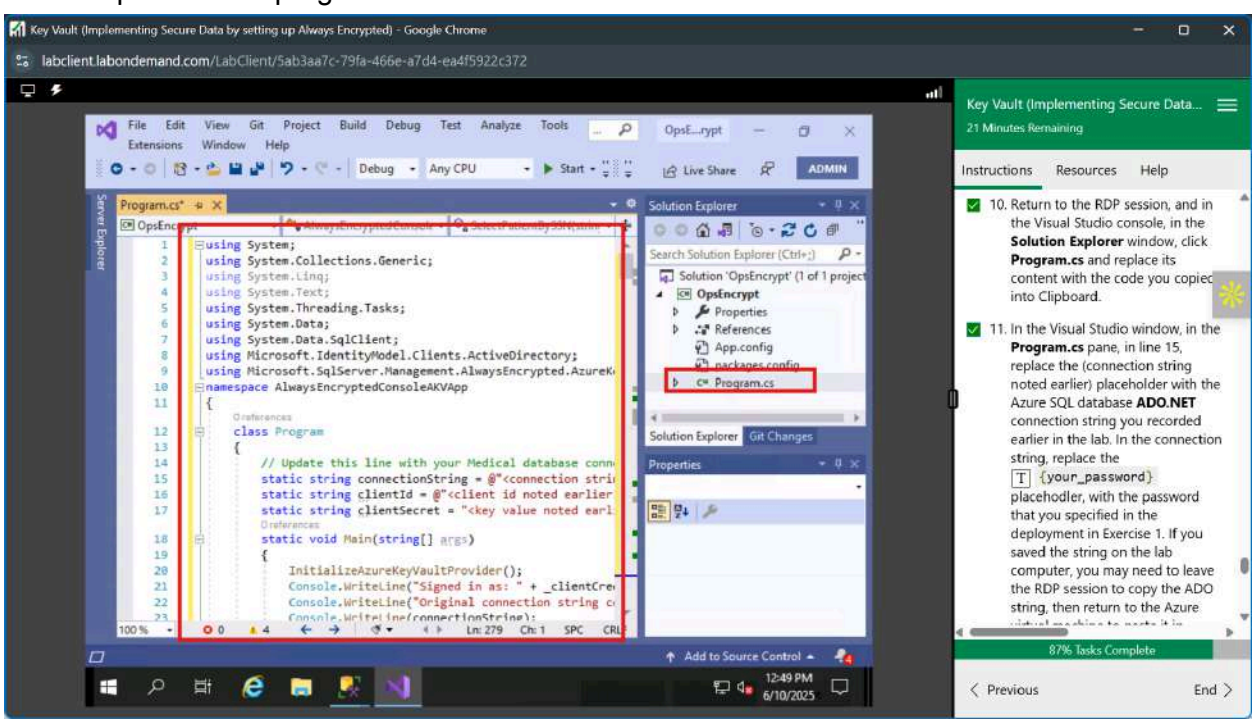


We then navigate to the tools menu and install necessary packages using the following command;

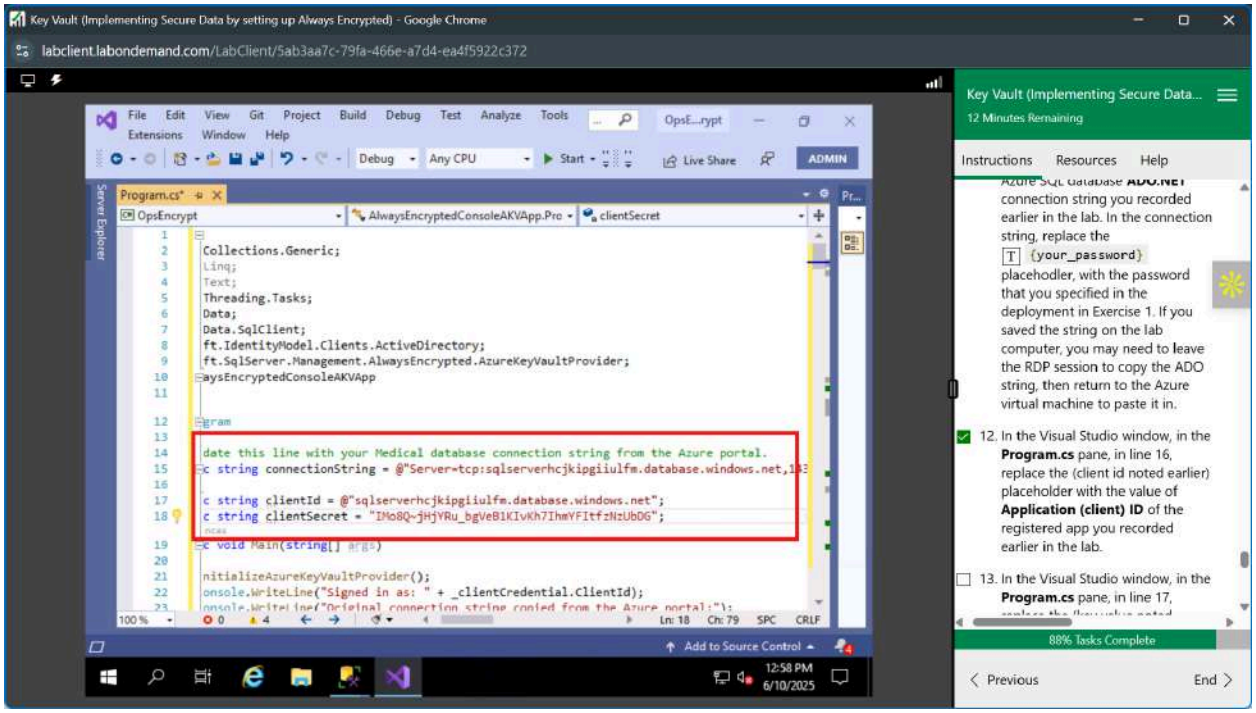
```
Install-Package
Microsoft.SqlServer.Management.AlwaysEncrypted.AzureKeyVaultProvider
```



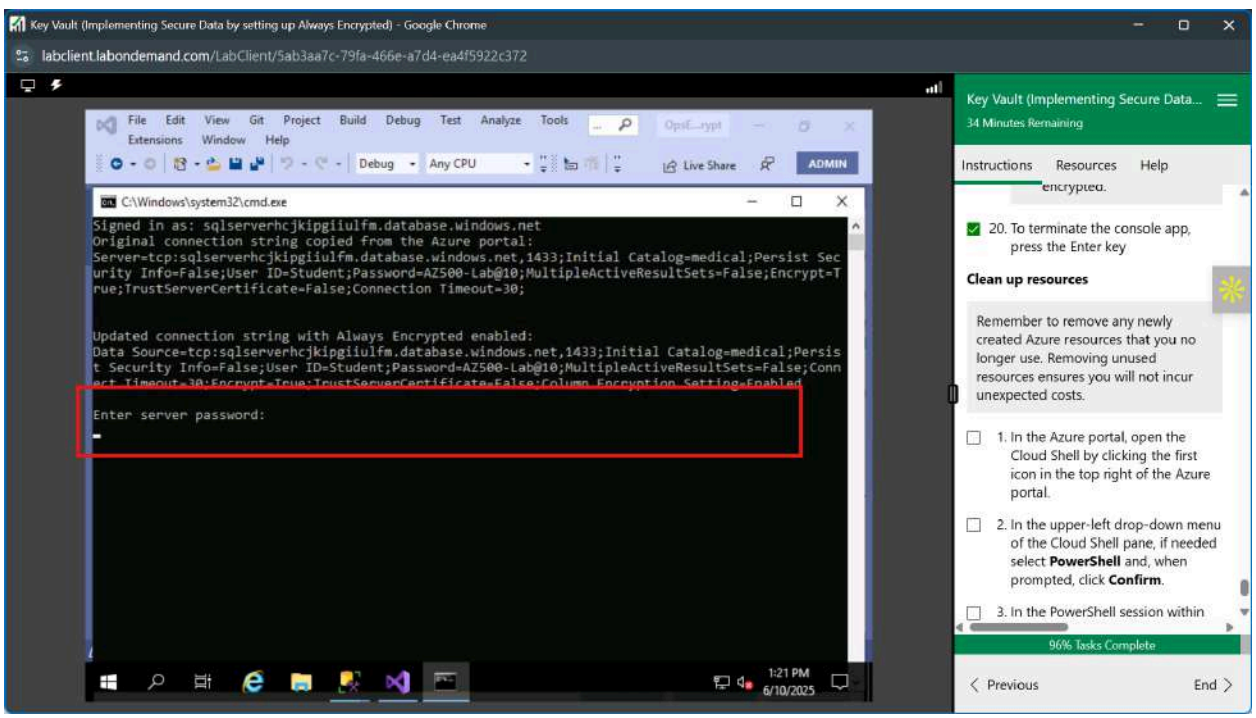
We then paste the C program code in our lab files into the new Visual Studio window.



From the code, we can replace the ADO.NET connection string in line 15, the client ID of our SQLApp in line 16, and the secret key generated in Key1 in line 17 as shown below;



We then run the code as shown below



Conclusion

To conclude, we have demonstrated how we can provision cloud infrastructure using an ARM Template. We have also managed to deploy and configure Azure KeyVault using the Azure CloudShell. We have also implemented always always-encrypted feature for sensitive columns in our database.