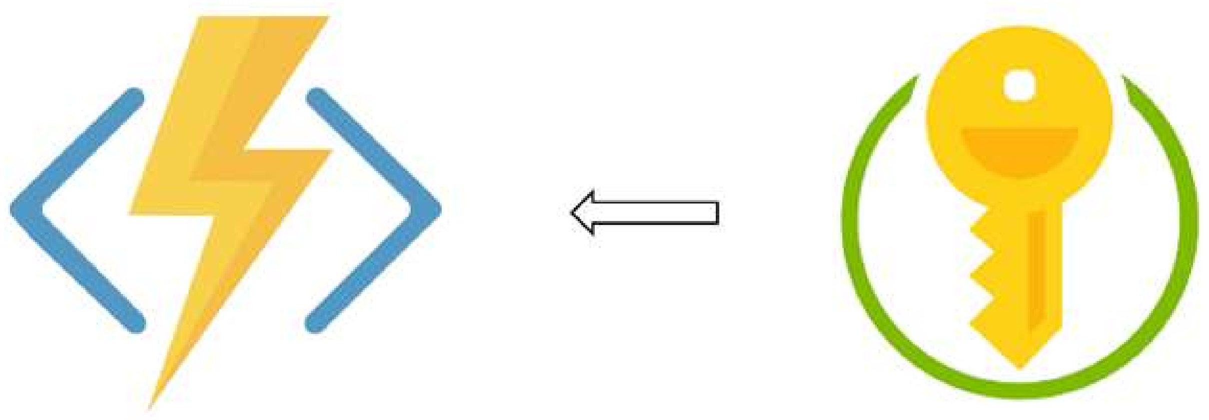
Open in app

James Leslie

Follow About

Accessing Azure Key Vault from Python Functions

James Leslie 7 hours ago · 5 min read



Why are you here?

You are busy developing a Python Azure Function and would like to access secrets from Azure Key Vault.

What have you done so far?

The steps in this article assume you have already done the following:

1. Created a local Python Azure function project in VS Code using the the Azure Functions extension

2. Deployed the Python function from your local project to a Function App in Azure

At this stage, you probably have some code that looks like the following:

```
def main(req: func.HttpRequest) -> func.HttpResponse:

    # do some stuff with hard-coded secrets here

    if success:
        return func.HttpResponse(
            'Success',
            status_code=200
        )
    else:
        return func.HttpResponse(
             'Something went wrong',
             status_code=404
        )
```

… and you are probably looking replace this with something like:

```
def main(req: func.HttpRequest) -> func.HttpResponse:

    # retrieve secrets from key vault instead

    if success:
        return func.HttpResponse(
            'Success',
            status_code=200
        )
    else:
        return func.HttpResponse(
             'Something went wrong',
             status_code=404
        )
```

If all of the above describes your current position, then read on!

1. Make changes to your local project

Let's start by making some changes to your local project in VS code.

simulates the behaviour of the deployed function.

When you first created your local project, you may have noticed a bunch of files were automatically created in your chosen project folder. Look for a file named `local.settings.json` . We will need to add some lines to this file to act as placeholder key vault secrets:

```
{
  “IsEncrypted”: false,
  “Values”: {
    “AzureWebJobsStorage”: “”,
    “FUNCTIONS_WORKER_RUNTIME”: “python”,
    “UsernameFromKeyVault”: “James”,
    “PasswordFromKeyVault”: “Pwd”
  }
}
```

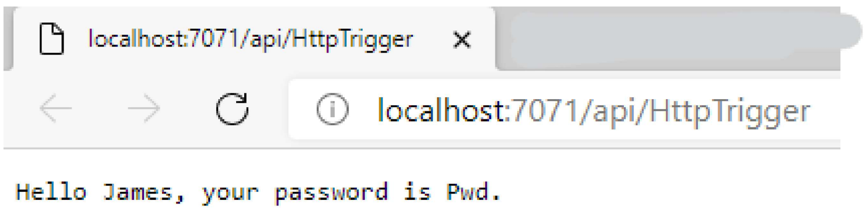
The names of the two secrets that I have added in the snippet above are `UsernameFromKeyVault` and `PasswordFromKeyVault` and their values are “James” and “Pwd” respectively.

These values will be accessible in the Python code by using the `os.getenv()` function. e.g. if we call `os.getenv('UsernameFromKeyVault')` in our function code, this will return the value “James”.

You can now modify your function to something like the following:

```
import os

import azure.functions as func

# access sensitive credentials from Azure Key Vault
user_name = os.getenv('UsernameFromKeyVault')
password = os.getenv('PasswordFromKeyVault')


def main(req: func.HttpRequest) -> func.HttpResponse:
    logging.info('Python HTTP trigger function processed a
request.')

    return func.HttpResponse(
        # reference credentials in function call
```

If you save this file and test locally in VS Code `(F5)` , you will see the values of your secrets returned in the function response:


localhost:7071/api/HttpTrigger
localhost:7071/api/HttpTrigger

```
Hello James, your password is Pwd.
```

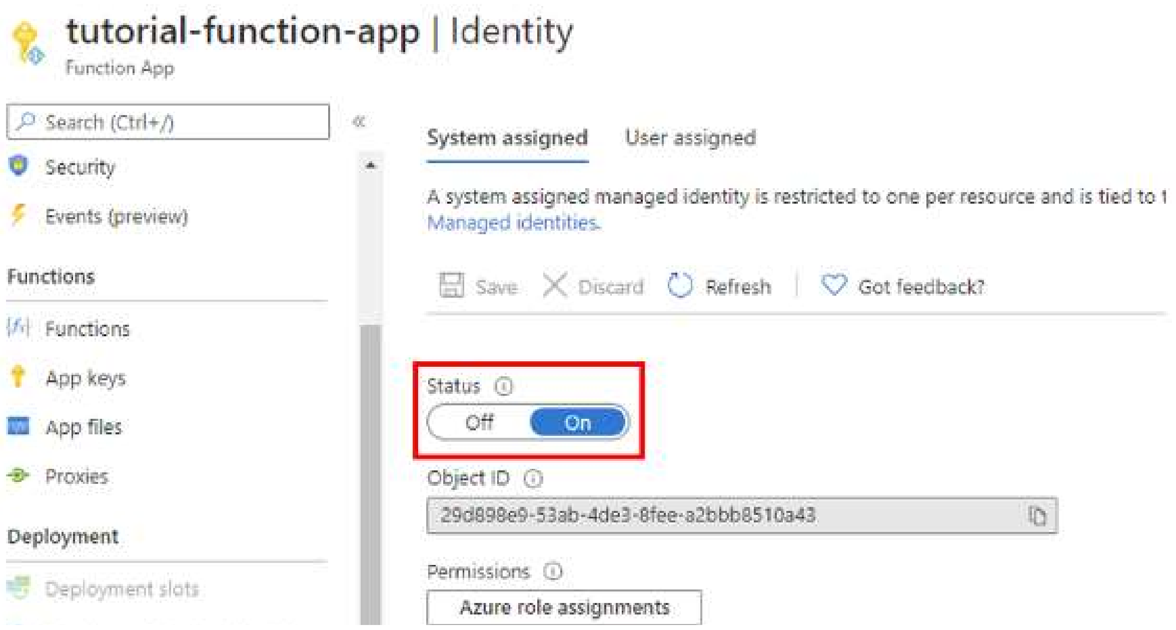
The key values have been retrieved from the local environment and used in the function return.

2. Make changes to in Azure

Next up, we will make some changes to our Azure resources so that we can access the key vault secrets in our function's environment.

2.1. Enable Function App system identity

In the Azure portal, go to your Function App. Under **Settings** -> **Identity** set the status to "On":



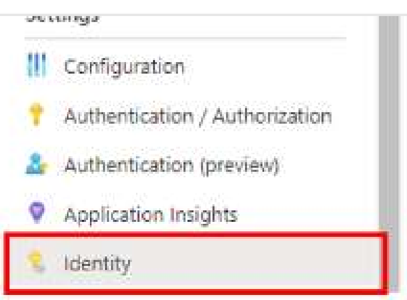
A system assigned managed identity is restricted to one per resource and is tied to t
Managed identities.

Status: Off [✓ On]

Object ID: 29d898e9-53ab-4de3-8fee-a2bbb8510a43

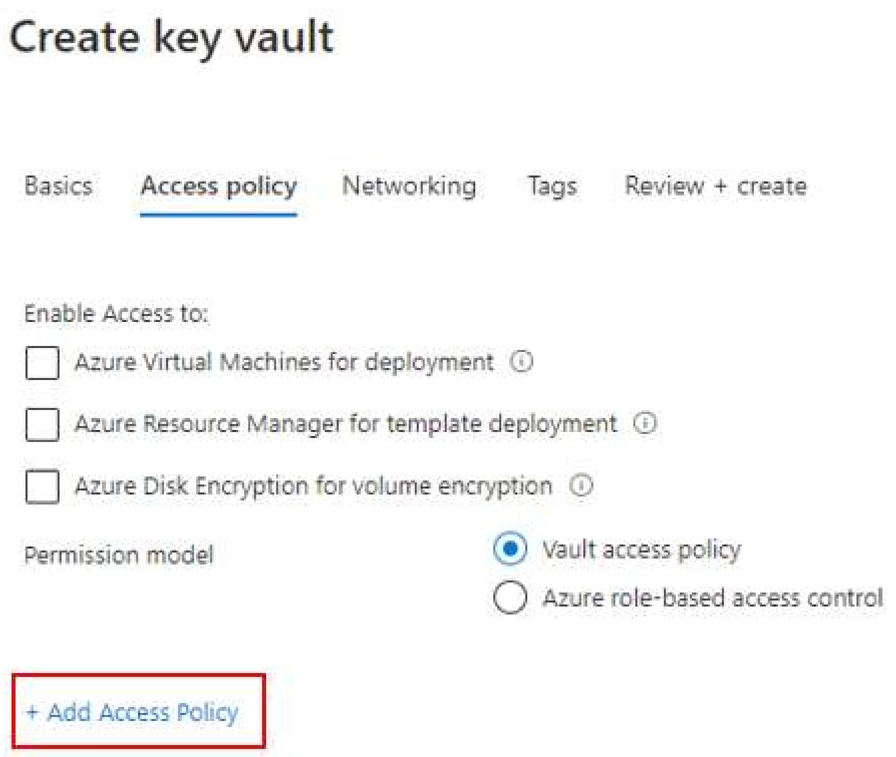
Permissions: Azure role assignments

Open in app

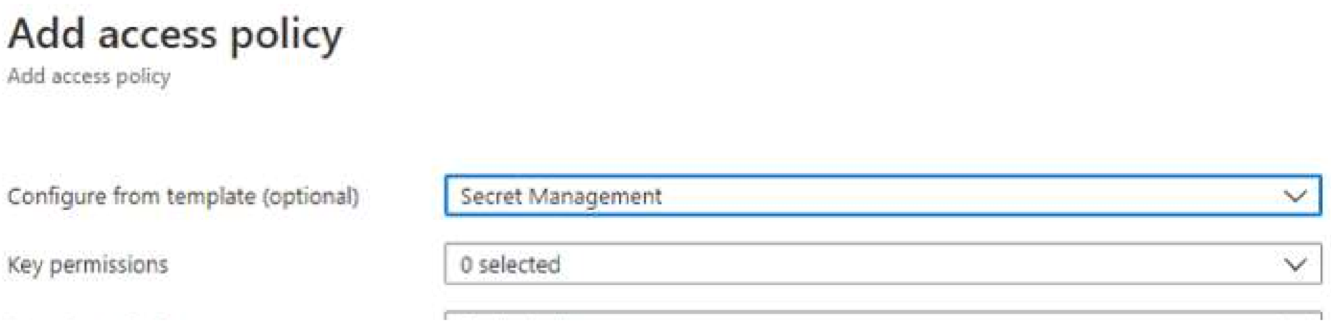


2.2. Add secrets to vault

If you haven't done so already, create a new Key Vault in your resource group. In the **Access policy** section of creating the vault, click on the text that says "+ Add Access Policy".

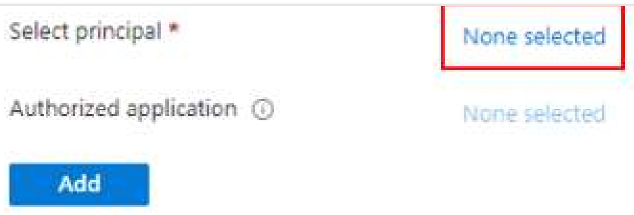

Create key vault
Basics
Access policy
Networking
Tags
Review + create
Enable Access to:
Azure Virtual Machines for deployment
Azure Resource Manager for template deployment
Azure Disk Encryption for volume encryption
Permission model
Vault access policy
Azure role-based access control
+ Add Access Policy

Click the dropdown box to **Configure from template** and choose "Secret Management".

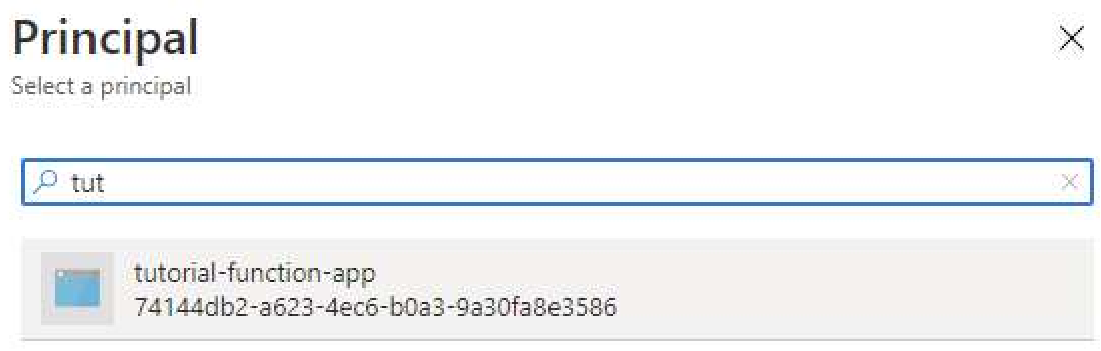


Configure from template (optional): Secret Management

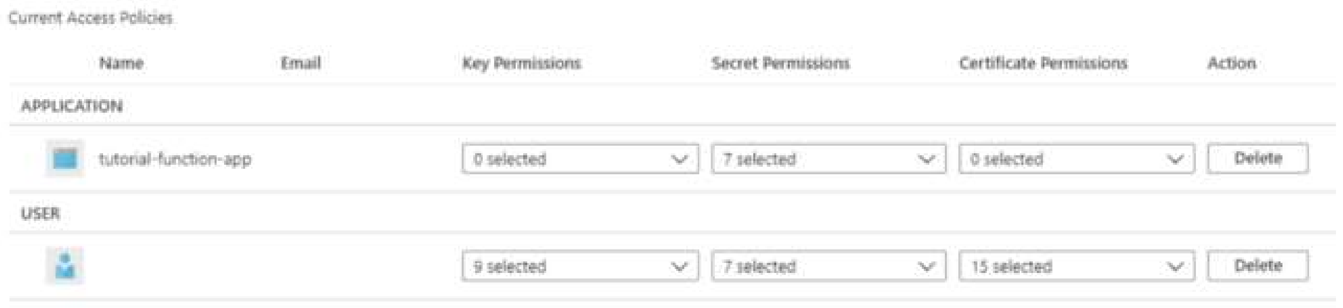
Key permissions: 0 selected



Click on the text that says "None selected" and search for the name of your function app in the panel on the right to provide it with access to the vault.



Click on "Add" and then confirm that your list of current access policies for the key vault looks like the list below before creating the vault.



Current Access Policies

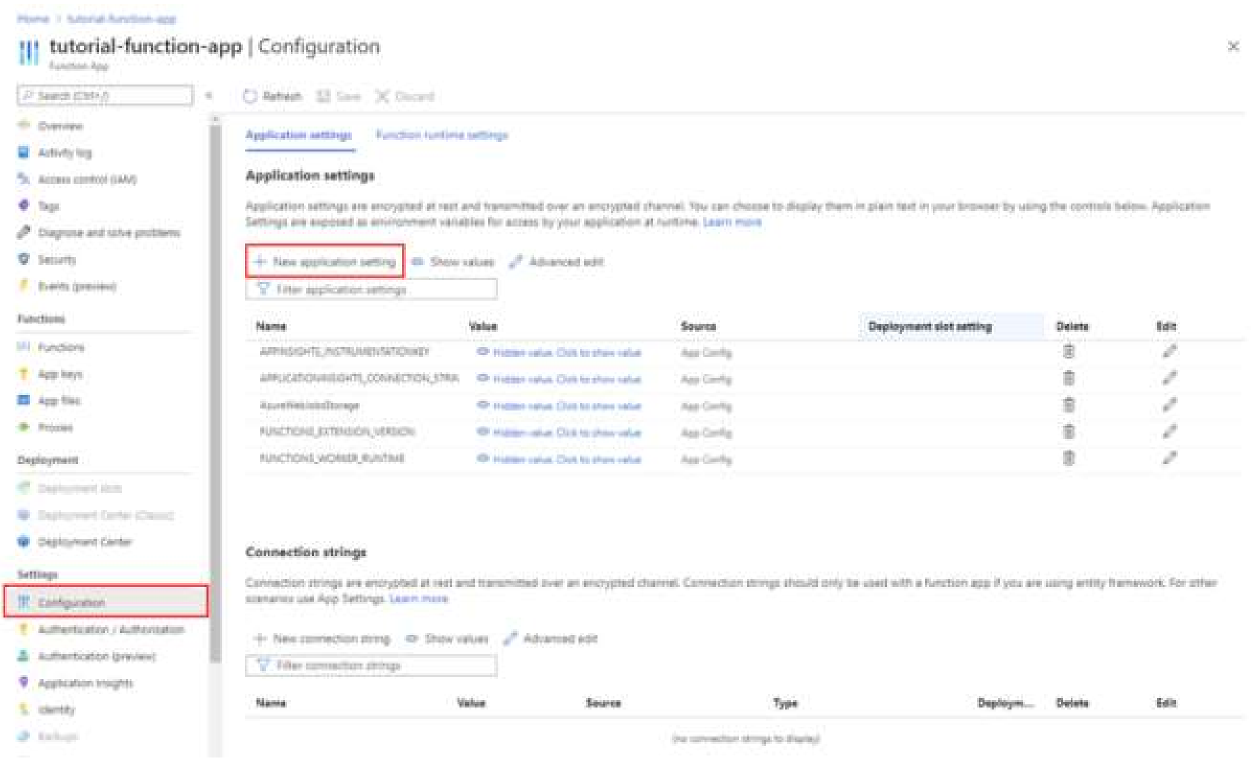
Name	Email	Key Permissions	Secret Permissions	Certificate Permissions	Action
APPLICATION					
tutorial-function-app		0 selected	7 selected	0 selected	Delete
USER					
		9 selected	7 selected	15 selected	Delete

Finally, add your desired secrets to the vault. While the *values* of these secrets do not need to be the same, their names need to be exactly the same as the ones you added to your `local.settings.json` file. In this example, we named them **UsernameFromKeyVault** and **PasswordFromKeyVault**. To illustrate the point, I have also changed the secret value of the password stored in the vault.

2.3. Add vault secrets to app configuration

configuration settings.

Click on the button to create a **new application setting**:



Application settings

Application settings are encrypted at rest and transmitted over an encrypted channel. You can choose to display them in plain text in your browser by using the controls below. Application Settings are exposed as environment variables for access by your application at runtime. Learn more

+ New application setting | Show values | Advanced edit

Filter application settings

Name	Value	Source	Deployment slot setting	Delete	Edit
APPINSIGHTS_INSTRUMENTATIONKEY	Hidden value. Click to show value	App Config			
APPLICATIONINSIGHTS_CONNECTION_STRI	Hidden value. Click to show value	App Config			
AzureWebJobsStorage	Hidden value. Click to show value	App Config			
FUNCTIONS_EXTENSION_VERSION	Hidden value. Click to show value	App Config			
FUNCTIONS_WORKER_RUNTIME	Hidden value. Click to show value	App Config			

Connection strings

Connection strings are encrypted at rest and transmitted over an encrypted channel. Connection strings should only be used with a function app if you are using entity framework. For other scenarios use App Settings. Learn more

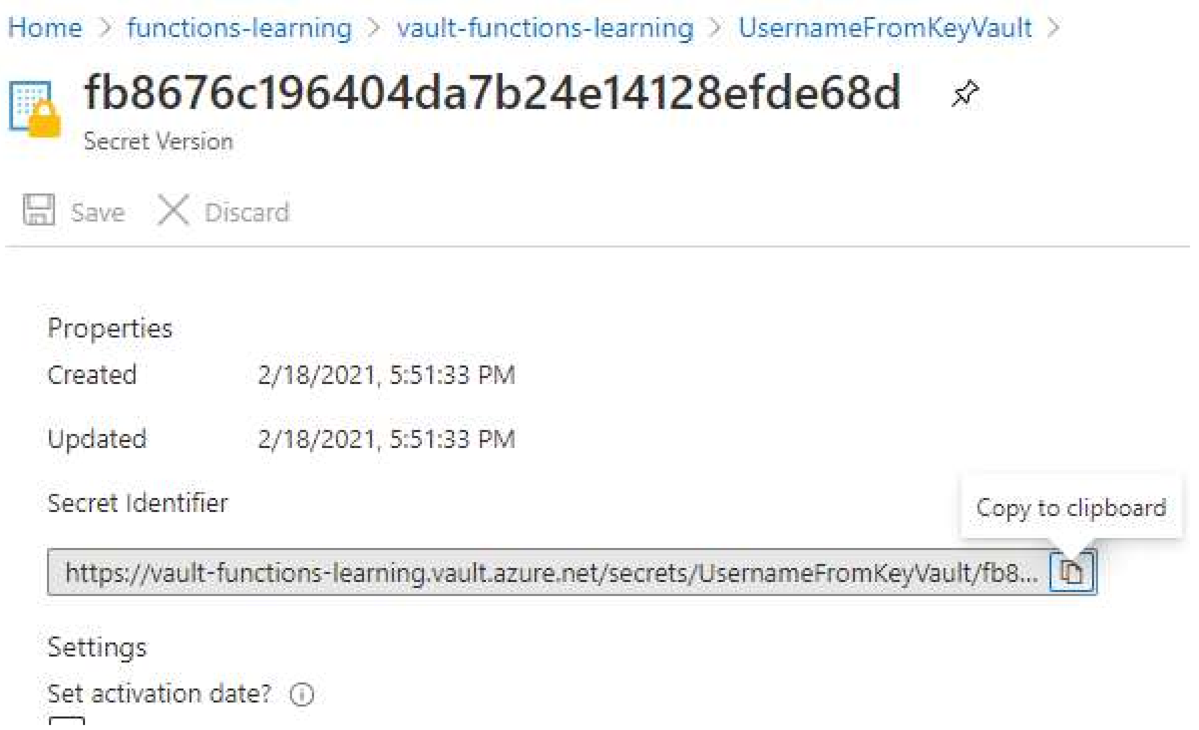
+ New connection string | Show values | Advanced edit

Filter connection strings

Name	Value	Source	Type	Deploym...	Delete	Edit

(no connection strings to display)

In your other tab, open the information of the username secret and click on the enabled version to see its details. There is a field here called **Secret Identifier**, copy its value and head back to the other tab.



Home > functions-learning > vault-functions-learning > UsernameFromKeyVault >
Save
Discard
Copy to clipboard

fb8676c196404da7b24e14128efde68d

Secret Version

Properties

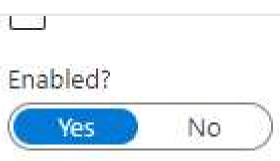
Created	2/18/2021, 5:51:33 PM
Updated	2/18/2021, 5:51:33 PM

Secret Identifier

https://vault-functions-learning.vault.azure.net/secrets/UsernameFromKeyVault/fb8...

Settings

Set activation date?

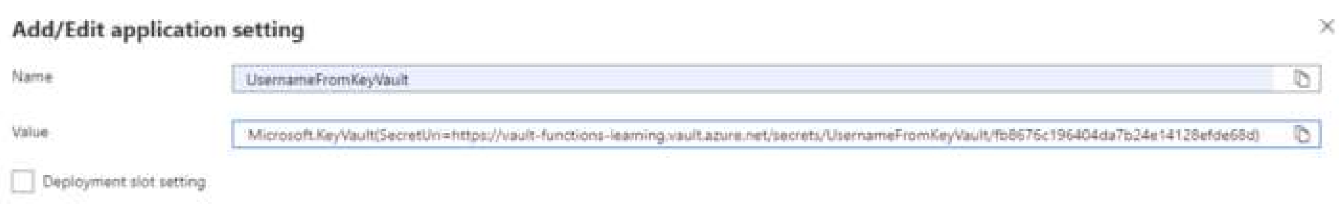


Enabled?

[✓ Yes] No

Now, paste this value into the Value field of the new application setting and then add some additional text in the following format:

```
@Microsoft.KeyVault(SecretUri=<copied-value-from-clipboard>)
```

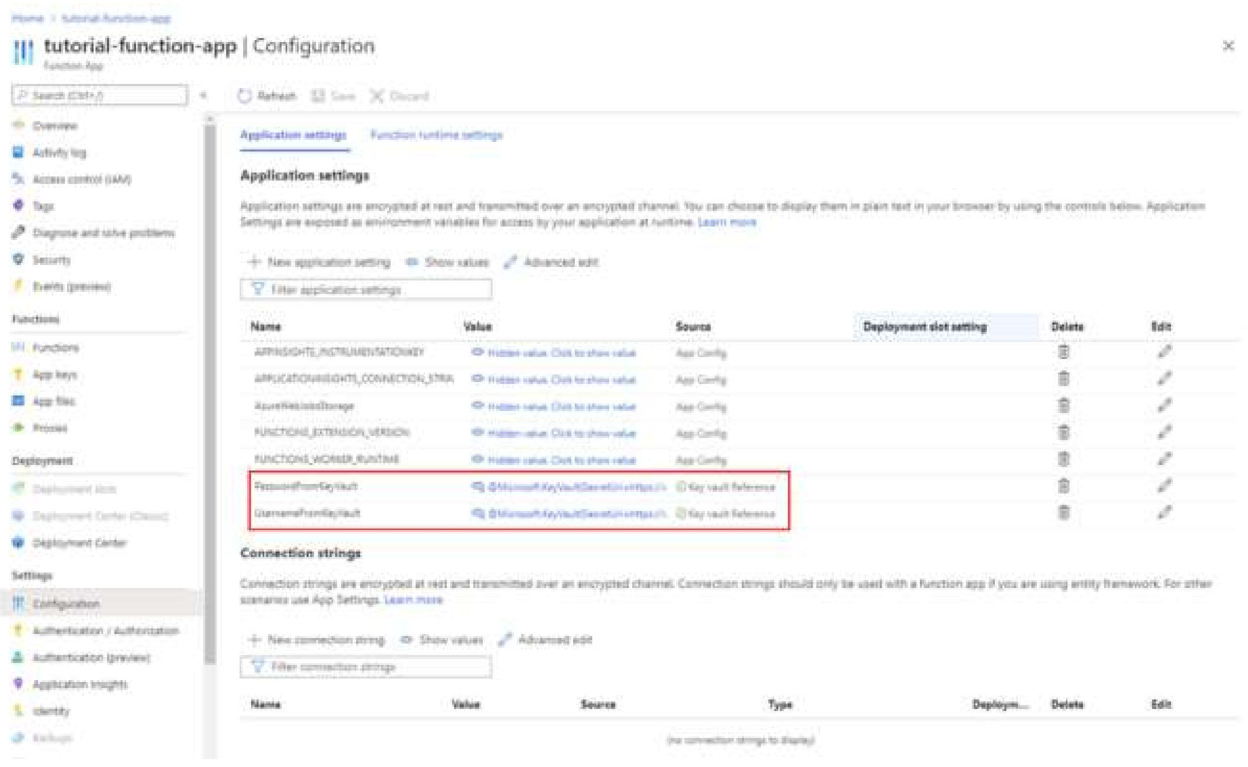


Add/Edit application setting

Name	UsernameFromKeyVault
Value	Microsoft.KeyVault(SecretUri=https://vault-functions-learning.vault.azure.net/secrets/UsernameFromKeyVault/fb8676c196404da7b24e14128efde68d)

Deployment slot setting

Do this for both the UsernameFromKeyVault and PasswordFromKeyVault secrets and you should see both appear with green checkmarks in your list of application settings.



tutorial-function-app | Configuration

Function App

Application settings

Application settings are encrypted at rest and transmitted over an encrypted channel. You can choose to display them in plain text in your browser by using the controls below. Application Settings are exposed as environment variables for access by your application at runtime. Learn more

Name	Value	Source	Deployment slot setting	Delete	Edit
APPINSIGHTS_INSTRUMENTATIONKEY	Hidden value. Click to show value	App Config			
APPLICATIONINSIGHTS_CONNECTION_STRIN	Hidden value. Click to show value	App Config			
AzureWebJobsStorage	Hidden value. Click to show value	App Config			
FUNCTIONS_EXTENSION_VERSION	Hidden value. Click to show value	App Config			
FUNCTIONS_WORKER_RUNTIME	Hidden value. Click to show value	App Config			
PasswordFromKeyVault	@Microsoft.KeyVault(SecretUri=https://...	✓ Key vault Reference			
UsernameFromKeyVault	@Microsoft.KeyVault(SecretUri=https://...	✓ Key vault Reference			

Connection strings

Connection strings are encrypted at rest and transmitted over an encrypted channel. Connection strings should only be used with a function app if you are using entity framework. For other scenarios use App Settings. Learn more

Name	Value	Source	Type	Deploym...	Delete	Edit

(no connection strings to display)

At this stage, you are all set to deploy the function and execute it using its URL. Once you have copied the function URL into your browser, you should see the same message as before, but with whatever secret values you used in your key vault.

```
Hello James, your password is password.
```

Azure Python Azure Functions