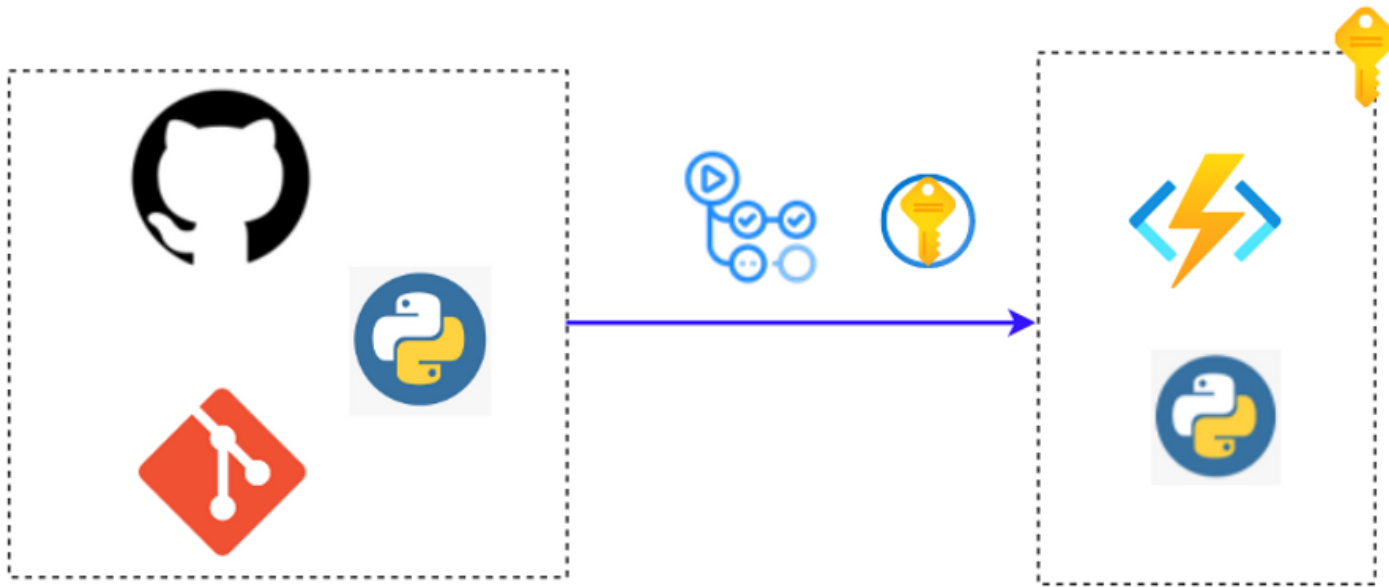


# LetsDevOps: Deploy Python Function to Azure Function App using GitHub Actions, Setup CI/CD

## Introduction

In this section we will learn how to Deploy Python Function to Azure Function App using GitHub Actions. With this we can also implement CI/CD for Azure Function App.

## Architecture



# CI/CD Workflow

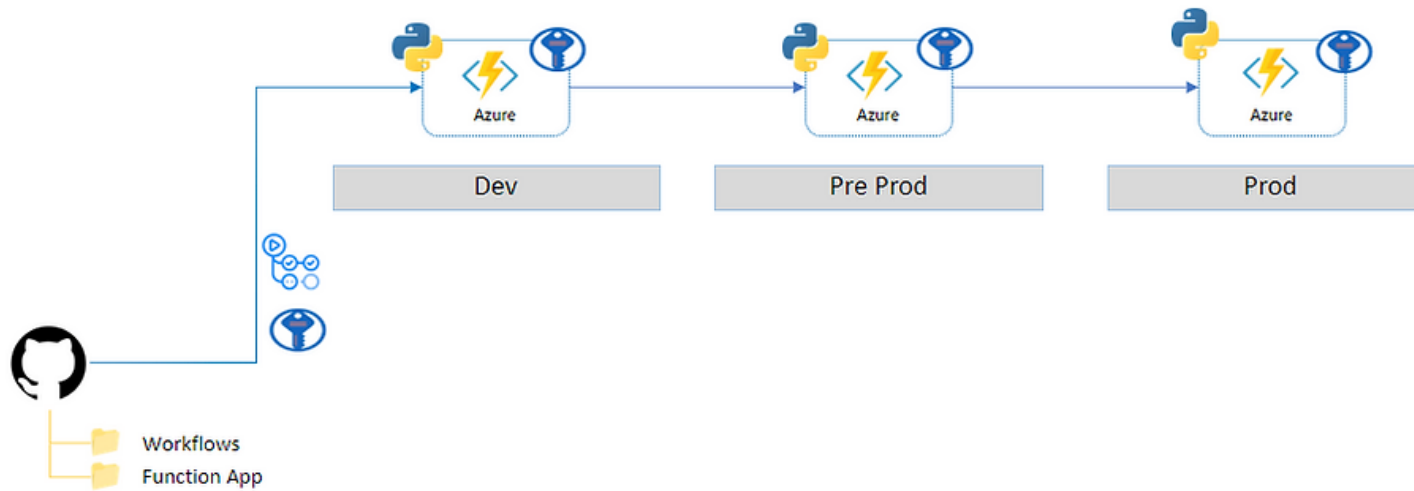
1. To implement CI/CD we are using the Source Code from GitHub Repo
2. GitHub Actions Workflow will

## Build

- Checkout the code
- Create Function App Package
- Include Dependent Module in Package [Module can be added in requirements.txt]

## Deploy ( Dev --> Pre Prod --> Prod )

- Download the Package with Dependent Module
- Deploy using the Function-Deploy Action



## How to Setup CI/CD

In this section we will learn how to setup CI/CD with GitHub and GitHub Action using YAML workflow.

If you are new to GitHub Action you can follow below Tutorial.

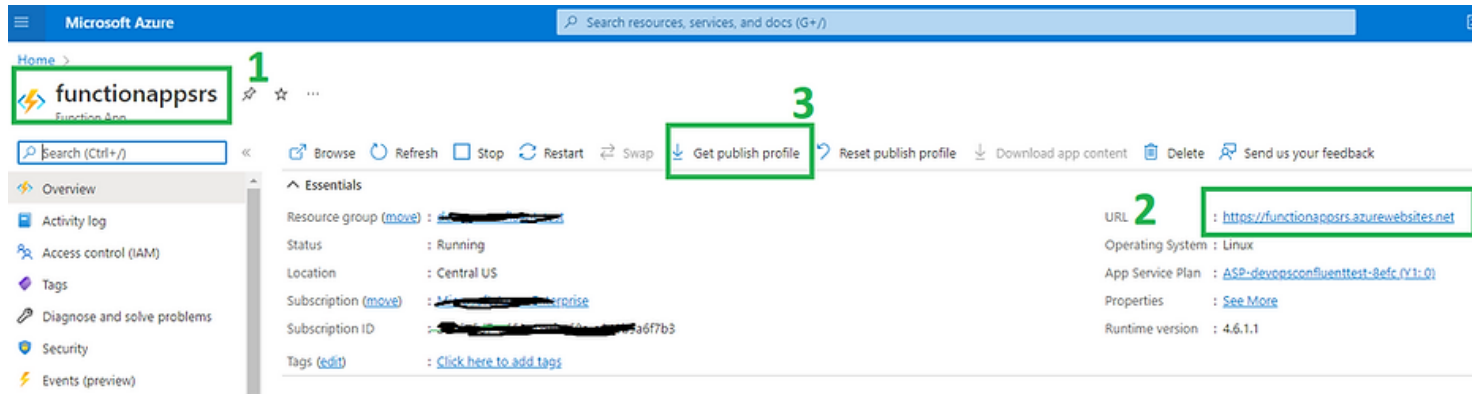
**GitHub Action Tutorial.**

## LetsDevOps: GitHub Actions Tutorial for Beginner's

### **Prerequisites**

1. Make Sure you have the Function App created with Python Runtime.

2. Get the Publish Profile Setting from the Function App Resource.
3. Import the Azure Function App Code to GitHub Repo
4. Get Azure Function App Details Like Azure Function App Name, Function App URL, Get Publish Profile credential.



Note: Since we are deploying the Python App Azure Function must be created with Runtime--> Python

## GitHub Repo Setup

As part of the prerequisite step you should have the Function App Source Code imported to GitHub Repo.

Here is the Sample Repo, you can refer.

<https://github.com/sumitraj0103/AzureFunctionCICD>

GitHub interface showing the repository **AzureFunctionCICD** (Public) by user **sumitraj0103**. The repository is located at **main** / **AzureFunctionCICD** / **FunctionApp** / **DemoSample**.

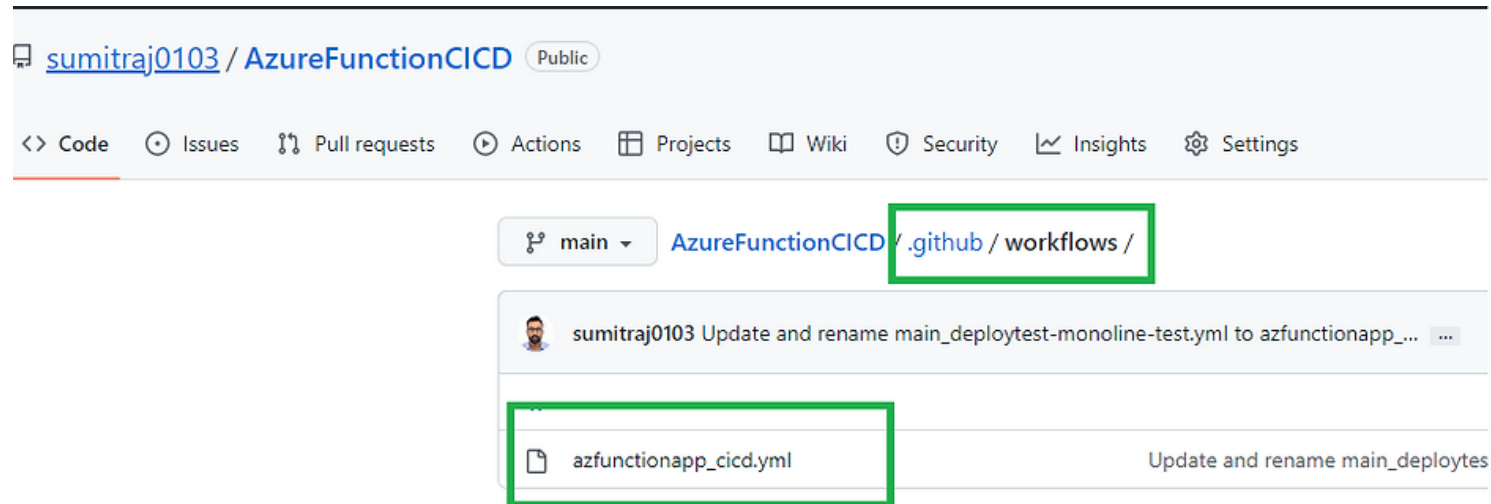
The repository contains the following files and folders:

File/Folder	Added repo
HttpTrigger1	Added repo
.funcignore	Added repo
.gitignore	Added repo
host.json	Added repo
requirements.txt	Added repo

## CI/CD Workflow Setup

In GitHub Action, workflow is termed like Pipeline. Hence we will setup the Workflow which will deploy the Function App.

Create Azure Function App workflow in the Repo Folder and save the File as:  
`.github/workflows/azfunctionapp_cicd.yml`



This sample file is available at.

[https://github.com/sumitraj0103/AzureFunctionCICD/edit/main/.github/workflows/azfunctionapp\\_cicd.yml](https://github.com/sumitraj0103/AzureFunctionCICD/edit/main/.github/workflows/azfunctionapp_cicd.yml)

```
# Docs for the Azure Web Apps Deploy action:
https://github.com/azure/functions-action
# More GitHub Actions for Azure: https://github.com/Azure/actions
# More info on Python, GitHub Actions, and Azure Functions:
https://aka.ms/python-webapps-actions
```

```
name: Function-CICD-Monoline
```

```
on:
```

```
  workflow_dispatch:
```

```
    inputs:
```

```
      # This is the path of your Azure Function in Git.
```

```
      GIT_FunctionApp_Name:
```

```
        description: 'Provide the Function App Name'
```

```
        required: true
```

```
        default: 'functionappsrs'
```

```
      GIT_FunctionApp_URL:
```

```
        description: 'Provide the Function App URL'
```

```
        required: true
```

```
        default: 'https://functionappsrs.azurewebsites.net'
```

```
      # This is the path of your Function app in Git.
```

```
      GIT_FunctionApp_PATH:
```

```
        description: 'Relative path for Function'
```

```
        required: true
```

```
        default: '/FunctionApp/DemoSample/'
```



```

jobs:
  build:
    runs-on: ubuntu-latest
    env:
      AZURE_FUNCTIONAPP_PACKAGE_PATH: ${GITHUB_WORKSPACE}/${GITHUB_EVENT_INPUTS_GIT_FUNCTIONAPP_PATH} # set this to the path to
your web app project, defaults to the repository root
      PYTHON_VERSION: '3.9' # set this to the python version to use
(supports 3.6, 3.7, 3.8)
    steps:
      - name: Checkout repository
        uses: actions/checkout@v2

      - name: Setup Python version
        uses: actions/setup-python@v1
        with:
          python-version: ${ENV_PYTHON_VERSION}

      #Install All the Required Dependent Modules
      - name: Install dependencies
        run: |
          pushd $GITHUB_WORKSPACE/${GITHUB_EVENT_INPUTS_GIT_FUNCTIONAPP_PATH}
          pip install -r requirements.txt --
target=".python_packages/lib/site-packages"
          popd

      - name: Upload artifact for deployment job
        uses: actions/upload-artifact@v2

```

```
    with:
      name: python-app
      path: |
        ${github.workspace}}/${github.event.inputs.GIT_FunctionApp_PATH }}
        !venv/

Deploy:
  runs-on: ubuntu-latest
  needs: build
  environment:
    name: 'Production'
    url: ${github.event.inputs.GIT_FunctionApp_URL }}

  steps:
    - name: Download artifact from build job
      uses: actions/download-artifact@v2
      with:
        name: python-app
        path: ${github.workspace}}/${github.event.inputs.GIT_FunctionApp_PATH }}

    - name: 'Deploy to Azure Functions'
      uses: Azure/functions-action@v1
      id: deploy-to-function
      with:
        app-name: ${github.event.inputs.GIT_FunctionApp_Name }}
        package: ${github.workspace}}/${github.event.inputs.GIT_FunctionApp_Path }}
```

```
github.event.inputs.GIT_FunctionApp_PATH }}  
  publish-profile: ${{ secrets.FUNCTIONAPP_PUBLISH }}
```

## Update the YAML Workflow

Once Workflow file is imported to your GitHub, Some Details Needs to be Updated as per Function App which was acquired at Prerequisite Step.

1. Function App Name, Function App URL and Function App Git Location.

AzureFunctionCICD / .github / workflows / azfunctionapp\_cicd.yml in main

&lt;&gt; Edit file

Preview changes

```
6
7 on:
8
9   workflow_dispatch:
10     inputs:
11       # This is the path of your Azure Function in Git.
12       GIT_FunctionApp_Name:
13         description: 'Provide the Function App Name'
14         required: true
15         default: 'functionappsrs'
16
17       GIT_FunctionApp_URL:
18         description: 'Provide the Function App URL'
19         required: true
20         default: 'https://functionapps.azurewebsites.net'
21
22       # This is the path of your Function app in Git.
23       GIT_FunctionApp_PATH:
24         description: 'Relative path for Function'
25         required: true
26         default: '/FunctionApp/DemoSample/'
27
```

2. Make Sure the Downloaded Publish Profile Content is updated under the secret. You can also manage this using Key Vault.

The screenshot displays the GitHub repository settings page for the repository 'sumitraj0103 / AzureFunctionCICD'. The 'Settings' tab is selected in the top navigation bar, indicated by a green box and the number '1'. In the left sidebar, the 'Secrets' option is highlighted with a green box and the number '2'. The main content area shows the 'Actions secrets' section. Under 'Environment secrets', it states 'There are no secrets for this repository's environments.' Below this, the 'Repository secrets' section is highlighted with a green box and the number '3', showing a single secret named 'FUNCTIONAPP\_PUBLISH' which was updated 42 minutes ago. Buttons for 'Update' and 'Remove' are visible next to the secret name.

# How to Run CI/CD

Now all the Configuration is completed we are ready to deploy.

Go to GitHub Account --> Actions --> Run the Workflow

The screenshot displays the GitHub Actions interface for the repository 'sumitraj0103 / AzureFunctionCICD'. The 'Actions' tab is selected, and the workflow 'Function-CICD-Monoline' is highlighted. The workflow is triggered by 'workflow\_dispatch'. The 'Run workflow' button is visible, and the 'Run workflow' dropdown menu is open, showing the 'Run workflow' button again. The interface includes a search bar for workflow runs and a table of workflow runs.

1. The 'Actions' tab is selected in the repository navigation bar.

2. The 'Function-CICD-Monoline' workflow is selected from the list of workflows.

3. The 'Run workflow' button is clicked, opening the 'Run workflow' dropdown menu.

4. The 'Run workflow' button is clicked in the dropdown menu.

Use workflow from

Branch: main

Provide the Function App Name \*

functionappsr

Provide the Function App URL \*

https://functionappsr.azurewebsites.net

Relative path for Function \*

/FunctionApp/DemoSample/

Run workflow

After the Successful Run you can see the Output like.

## Function-CICD-Monoline Function-CICD-Monoline #1

Summary

Jobs

build

Deploy

### build

succeeded 1 hour ago in 17s

- > ✓ Set up job
- > ✓ Checkout repository
- > ✓ Setup Python version
- > ✓ Install dependencies
- > ✓ Upload artifact for deployment job
- > ✓ Post Checkout repository
- > ✓ Complete job



Check whether the Function App Deployed or Not.

[Home](#) > [functionappsrs](#)

# functionappsrs | Functions ...

Function App

[+ Create](#)[Refresh](#)[Delete](#)[Overview](#)[Activity log](#)[Access control \(IAM\)](#)[Tags](#)[Diagnose and solve problems](#)[Security](#)[Events \(preview\)](#)[Functions](#)[Functions](#)[App keys](#)[App files](#)[Proxies](#)

This function has been edited through an external editor. Portal editing is disabled.



Name ↑↓

Trigger ↑↓



HttpTrigger1

HTTP

# Demo

Setup CI/CD for Azure Function App, Python Function using GitHub Actions

