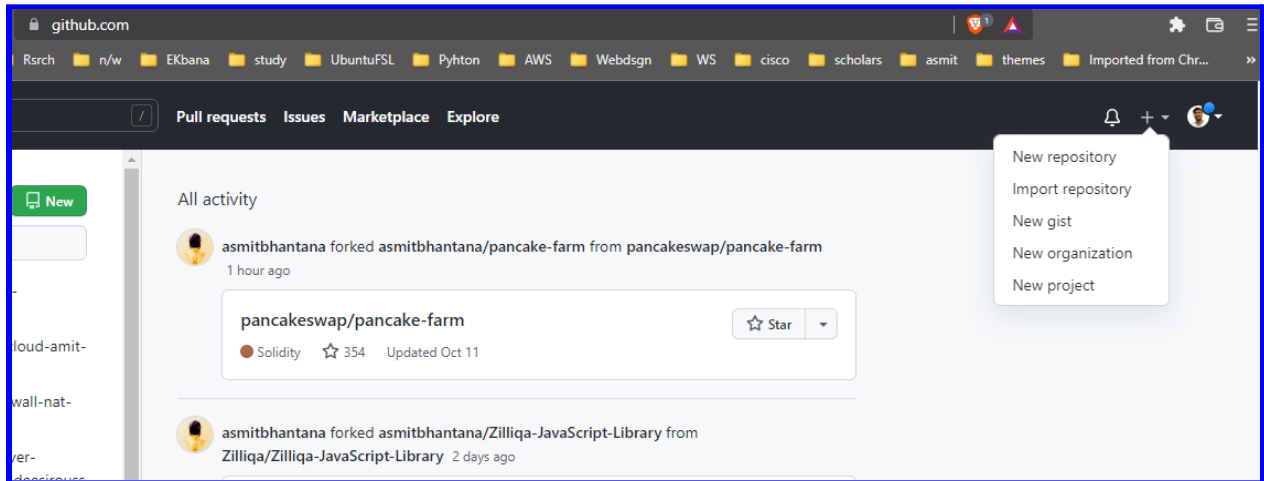


# Integrate both these scripts with one of Jenkins, Github Actions, CircleCI or TravisCI


I have used **GitHub Actions**

## Creating new repository for Lambda CI/CD



## Name - aws-cicd, Private Repo

Owner \*

 deesirouss ▾


/

Repository name \*


aws-cicd ✓

Great repository names are short and memorable. Need inspiration? How about [musical-octo-memory](#)?

Description (optional)

☐  **Public**

Anyone on the internet can see this repository. You choose who can commit.

☒  **Private**

You choose who can see and commit to this repository.

**Initialize this repository with:**

Skip this step if you're importing an existing repository.

☐ **Add a README file**

This is where you can write a long description for your project. [Learn more.](#)

☐ **Add .gitignore**

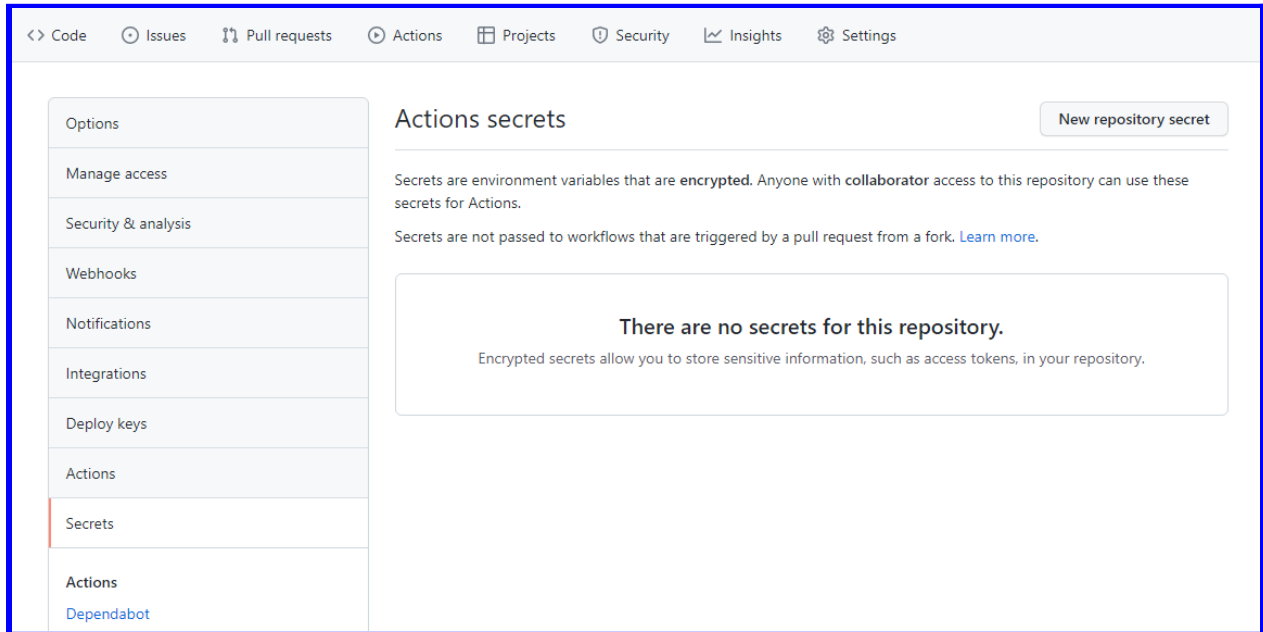
Choose which files not to track from a list of templates. [Learn more.](#)

☐ **Choose a license**

A license tells others what they can and can't do with your code. [Learn more.](#)

Create repository

## Creating Secret key for AWS Access key and Secret key



This screenshot shows the 'Actions secrets' page in a GitHub repository. The top navigation bar includes links for Code, Issues, Pull requests, Actions, Projects, Security, Insights, and Settings. On the left, a sidebar menu lists various repository settings, with 'Secrets' highlighted. The main content area is titled 'Actions secrets' and includes a 'New repository secret' button. It contains explanatory text about secrets being encrypted environment variables and a message stating 'There are no secrets for this repository.' with a brief description of what secrets are used for.

< > Code Issues Pull requests Actions Projects Security Insights Settings

Options

Manage access

Security & analysis

Webhooks

Notifications

Integrations

Deploy keys

Actions

Secrets

Actions

Dependabot

### Actions secrets

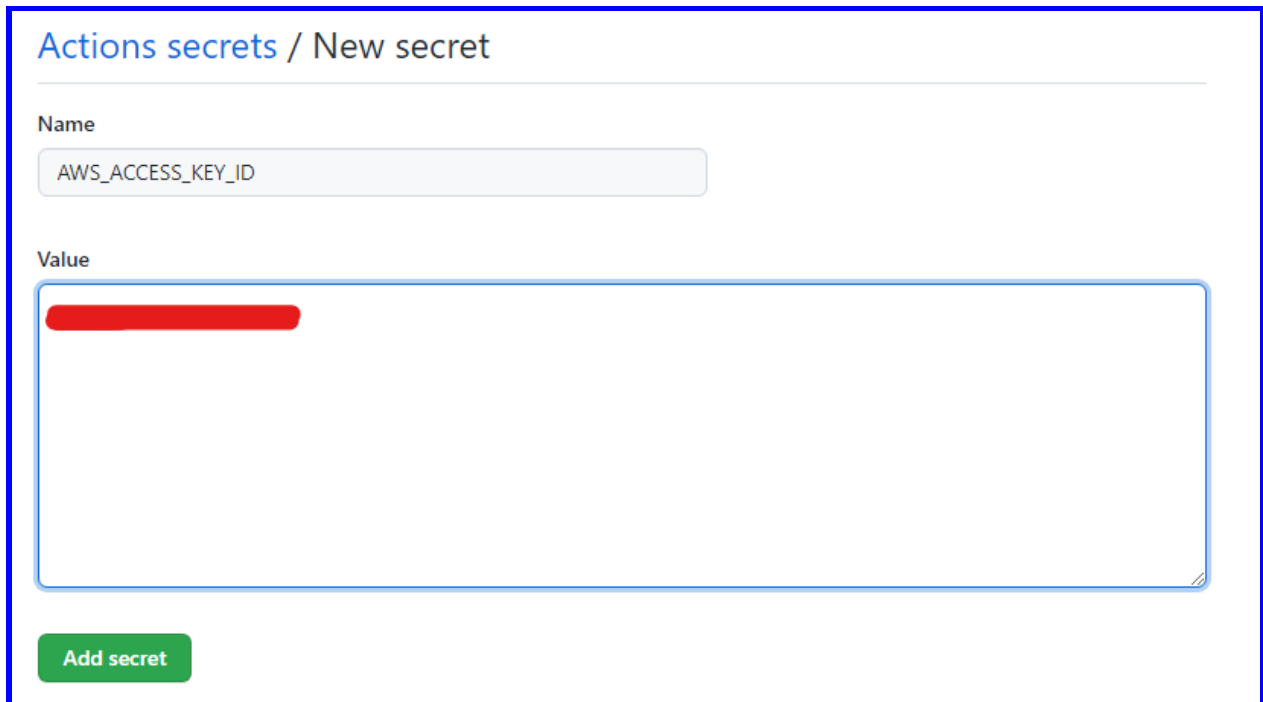
New repository secret

Secrets are environment variables that are **encrypted**. Anyone with **collaborator** access to this repository can use these secrets for Actions.

Secrets are not passed to workflows that are triggered by a pull request from a fork. [Learn more](#).

**There are no secrets for this repository.**

Encrypted secrets allow you to store sensitive information, such as access tokens, in your repository.



This screenshot shows the 'New secret' form in the 'Actions secrets' section. The title is 'Actions secrets / New secret'. There are two main input fields: 'Name' and 'Value'. The 'Name' field contains the text 'AWS\_ACCESS\_KEY\_ID'. The 'Value' field is a large text area with a red bar at the top, indicating that the content has been masked. At the bottom left of the form is a green 'Add secret' button.

### Actions secrets / New secret

Name

AWS\_ACCESS\_KEY\_ID

Value

Add secret

## Actions secrets / New secret

Name

AWS\_SECRET\_ACCESS\_KEY

Value

[Redacted value]

Add secret

## We have two Secrets

### Actions secrets

New repository secret

Secrets are environment variables that are **encrypted**. Anyone with **collaborator** access to this repository can use these secrets for Actions.

Secrets are not passed to workflows that are triggered by a pull request from a fork. [Learn more.](#)



AWS\_ACCESS\_KEY\_ID

Updated 1 minute ago

Update

Remove



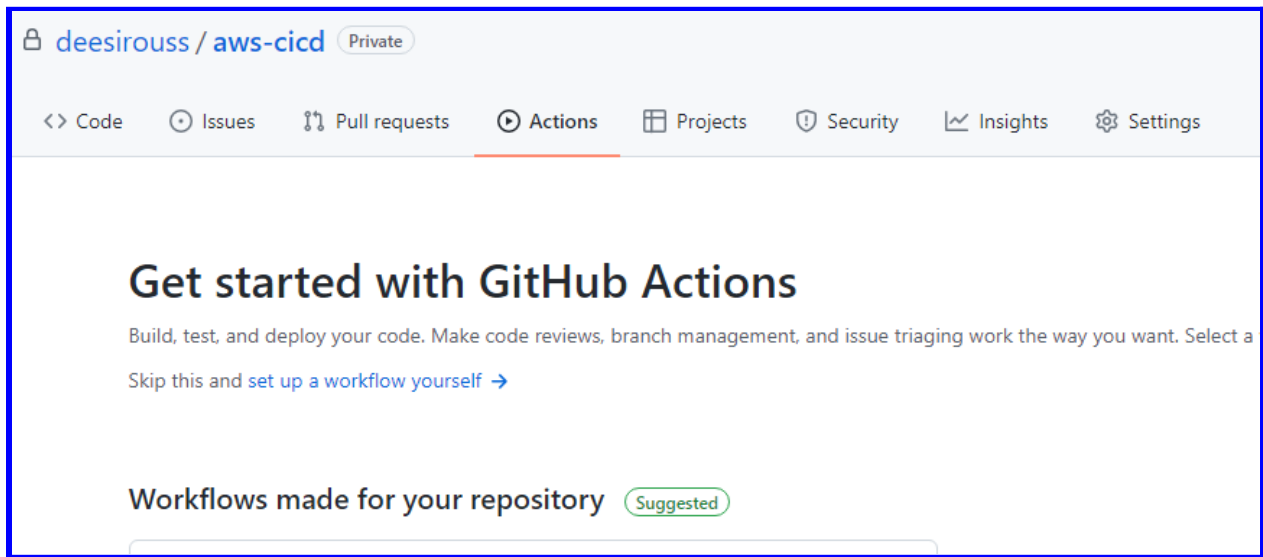
AWS\_SECRET\_ACCESS\_KEY

Updated in 15 seconds

Update

Remove

## Creating Github Actions → Actions → Set up a Workflow yourself



# This is a basic workflow to help you get started with Actions

name: Lambda CICD

# Controls when the workflow will run

on:

# Triggers the workflow on push or pull request events but only for the main branch

push:

branches: [ main ]

pull\_request:

branches: [ main ]

# Allows you to run this workflow manually from the Actions tab

workflow\_dispatch:

# A workflow run is made up of one or more jobs that can run sequentially or in parallel

jobs:

# This workflow contains a single job called "build"

build:

# The type of runner that the job will run on

runs-on: ubuntu-latest

# Steps represent a sequence of tasks that will be executed as part of the job

steps:

# Checks-out your repository under \$GITHUB\_WORKSPACE, so your job can access it

- uses: actions/checkout@v2

#AWS credentials

- name: Configure AWS credentials

```
uses: aws-actions/configure-aws-credentials@v1
with:
  aws-access-key-id: ${ secrets.AWS_ACCESS_KEY_ID }
  aws-secret-access-key: ${ secrets.AWS_SECRET_ACCESS_KEY }
  # TODO Change your AWS region here!
  aws-region: us-east-2

- name: creating Zip file of lambda-fuction.py
  run: |
    zip lambda_function.zip lambda_function.py

- name: Deploying lambda function to aws
  run: |
    aws lambda create-function --function-name bibek-demo-deployment \
    --runtime python3.9 --zip-file fileb://lambda_function.zip \
    --role arn:aws:iam::949263681218:role/service-role/bibek-api-gw-default-role-mof54wmn \
    --handler lambda_function.lambda_handler
```

If we see the build result, all jobs have been executed successfully

The screenshot displays the GitHub Actions interface for a workflow named 'Update lambda.yml' (run ID: Lambda CICD #5). The workflow status is 'Completed' with a green checkmark. The 'Summary' tab is selected, showing a list of jobs. The 'build' job is highlighted and expanded, showing its steps and duration. The 'build' job succeeded 1 minute ago in 11s. The steps listed are:

Step	Duration
Set up job	3s
Run actions/checkout@v2	0s
Configure AWS credentials	1s
creating Zip file of lambda-fuction.py	0s
Deploying lamda function to aws	6s
Post Configure AWS credentials	0s
Post Run actions/checkout@v2	1s
Complete job	0s

## Deploying Lambda function to aws Job details

```
13 ▶ Run aws lambda create-function --function-name bibek-demo-deployment \
24 {
25   "FunctionName": "bibek-demo-deployment",
26   "FunctionArn": "arn:aws:lambda:us-east-2:***:function:bibek-demo-deployment",
27   "Runtime": "python3.9",
28   "Role": "arn:aws:iam:***:role/service-role/bibek-api-gw-default-role-mof54wmn",
29   "Handler": "lambda_function.lambda_handler",
30   "CodeSize": 334,
31   "Description": "",
32   "Timeout": 3,
33   "MemorySize": 128,
34   "LastModified": "2021-12-17T06:42:51.627+0000",
35   "CodeSha256": "0qIxwI+9yvwBEZs/RSqigV7Q1uw1gN2CTa1fbbqSP4=",
36   "Version": "$LATEST",
37   "TracingConfig": {
26     "Mode": "PassThrough"
27   },
28   "RevisionId": "8dcd94dd-82f6-47d6-a5ea-d42dc95333a4",
29   "State": "Pending",
30   "StateReason": "The function is being created.",
31   "StateReasonCode": "Creating",
32   "PackageType": "Zip",
```

We can see the new lambda function has been deployed recently

Lambda > Functions

Functions (11) Last fetched 17 seconds ago ⌂ Actions ▾ Create function

🔍 Filter by tags and attributes or search by keyword < 1 2 > ⚙️

<input type="checkbox"/>	Function name ▾	Description	Package type ▾	Runtime ▾	Code size ▾	Last modified ▾
<input type="checkbox"/>	bibek-demo-deployment	-	Zip	Python 3.9	334.0 byte	41 seconds ago

Code source Info

File Edit Find View Go Tools Window Test ▾ Deploy Changes deployed

🔍 Go to Anything (Ctrl-P)

Environment

- ▾ bibek-demo-deploy ⚙️
  - 📄 lambda\_function.py

```
1 import json
2
3 def lambda_handler(event, context):
4     # TODO implement
5     return {
6         'statusCode': 200,
7         'body': json.dumps({"Hello": "Default from Bibek Mishra"})
8     }
9
```

## Slight changes in json file for message

```
import json
def lambda_handler(event, context):
    # TODO implement
    return {
        'statusCode': 200,
        'body': json.dumps({"Hello": "CI/CD from Bibek Mishra"})
    }
```

## Changing the workflow to update the created function

```
#AWS credentials
- name: Configure AWS credentials
  uses: aws-actions/configure-aws-credentials@v1
  with:
    aws-access-key-id: ${ secrets.AWS_ACCESS_KEY_ID }
    aws-secret-access-key: ${ secrets.AWS_SECRET_ACCESS_KEY }
    # TODO Change your AWS region here!
    aws-region: us-east-2

- name: creating Zip file of lambda-fuction.py
  run: |
    zip lambda_function.zip lambda_function.py

- name: Deploying lambda function to aws
  run: |
    aws lambda update-function-code --function-name bibek-demo-deployment \
    --zip-file fileb://lambda_function.zip
```

## Workflow triggered itself

The screenshot shows a GitHub Actions workflow run titled "Update lambda.yml Lambda CICD #7". The workflow is in a "Completed" state, indicated by a green checkmark. The left sidebar shows a "Summary" tab and a "Jobs" section with a single job named "build" marked as successful. The main panel displays the "build" job details, showing it "succeeded 26 seconds ago in 10s". A list of steps is shown, each with a green checkmark and a duration:


Step	Duration
Set up job	3s
Run actions/checkout@v2	1s
Configure AWS credentials	0s
creating Zip file of lambda-fuction.py	0s
Deploying lamda function to aws	6s
Post Configure AWS credentials	0s
Post Run actions/checkout@v2	0s
Complete job	0s




## Update function completed

### build

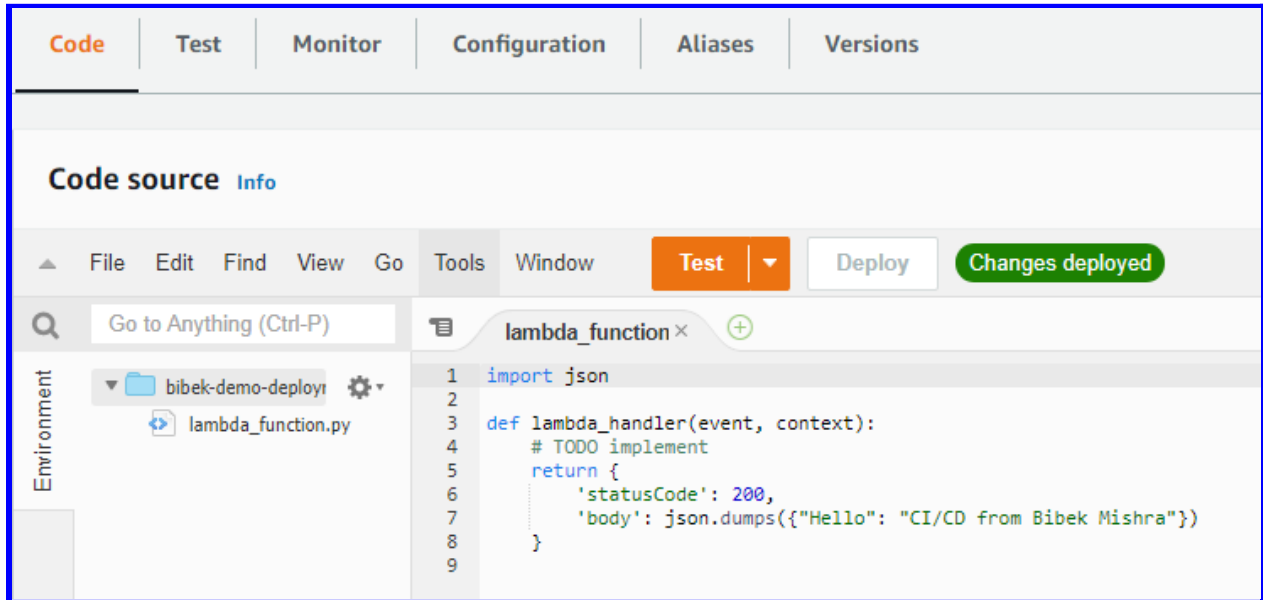
succeeded 26 seconds ago in 10s

>  creating Zip file of lambda-fuction.py

▼  Deploying lamda function to aws

```
1  ▶ Run aws lambda update-function-code --function-name bibek-demo-deployment \  
10 {  
11     "FunctionName": "bibek-demo-deployment",  
12     "FunctionArn": "arn:aws:lambda:us-east-2::*:function:bibek-demo-deployment",  
13     "Runtime": "python3.9",  
14     "Role": "arn:aws:iam::*:role/service-role/bibek-api-gw-default-role-mof54wmn",  
15     "Handler": "lambda_function.lambda_handler",  
16     "CodeSize": 334,  
17     "Description": "",  
18     "Timeout": 3,  
19     "MemorySize": 128,  
20     "LastModified": "2021-12-17T06:51:47.000+0000",  
21     "CodeSha256": "Oy72nJGrU91D2Ve56ATV10eveP6pqeM7ZA6rEJ4D6Z8=",  
22     "Version": "$LATEST",  
23     "TracingConfig": {  
24         "Mode": "PassThrough"  
25     },  
26     "RevisionId": "ab0d8b78-1afd-47ab-98fd-f8b72fc244fa",  
27     "State": "Active",  
28     "LastUpdateStatus": "InProgress",  
29     "LastUpdateStatusReason": "The function is being created.",  
30     "LastUpdateStatusReasonCode": "Creating",
```

If we see the code of lambda function it has been changed successfully



In this way we can build a pipeline through Github Actions which can update the lambda function whenever any changes are pushed to the main branch