Pipeops

Infrastructure as Code (IaC) with Python

Alex Idowu

CTO @ PipeOps

What is IaC?

IaC is short for Infrastructure as Code

IaC is all about handling and setting up computing infrastructure using files that machines can read, instead of relying on manual setups, and click-ops.

Features of IaC?

- **Consistency:** Ensures reliable, secure, and efficient infrastructure, leading to stable and scalable operations.
- **Version Control:** TTracks different versions of your infrastructure setup, similar to how you manage app code.
- **Idempotency:** Allows infrastructure deployments to be run multiple times without changing the result.
- **Leverage Existing Toolchains:** Python for IaC seamlessly integrates with your current tools for testing, dependency management, and version control, without needing to learn anything new.

Why Python for IaC?

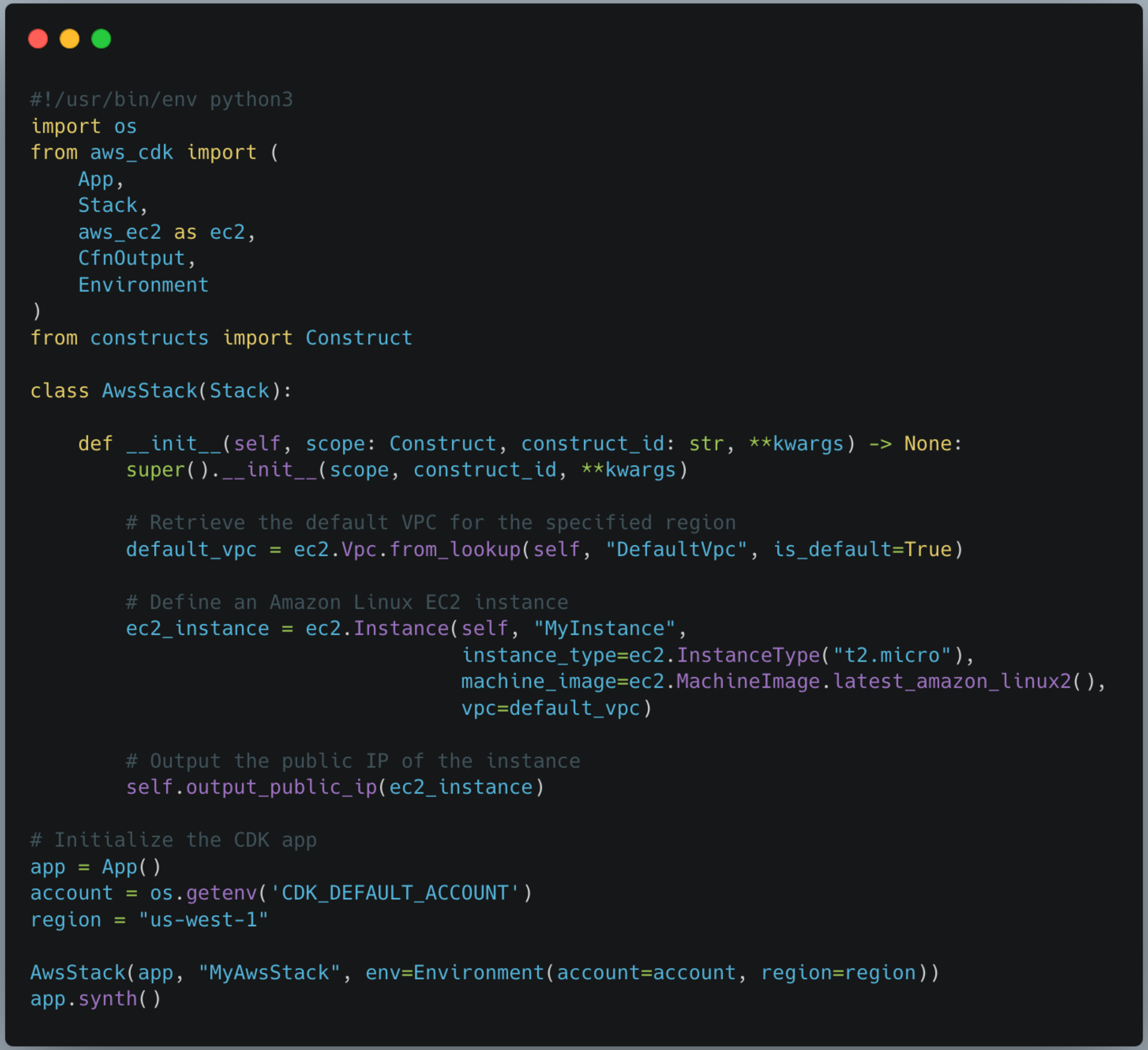
- **Rich Ecosystem:** Python has a rich ecosystem of libraries and frameworks, such as Boto3, Ansible, and the AWS CDK, which provide robust tools for managing infrastructure.
- **Strong Community:** Python has a large and active community; *i mean that's why we are here.*😂

Popular Python Tools for IaC.

AWS CDK (Cloud Development Kit)

- **Purpose:** AWS CDK allows you to define AWS infrastructure using familiar programming languages, including Python, and deploy it using AWS CloudFormation.
- **Example Use:** Create and manage AWS resources using Python code.
- **State Management:** AWS CDK uses AWS CloudFormation to manage the state of your infrastructure.

Sample Code 1.



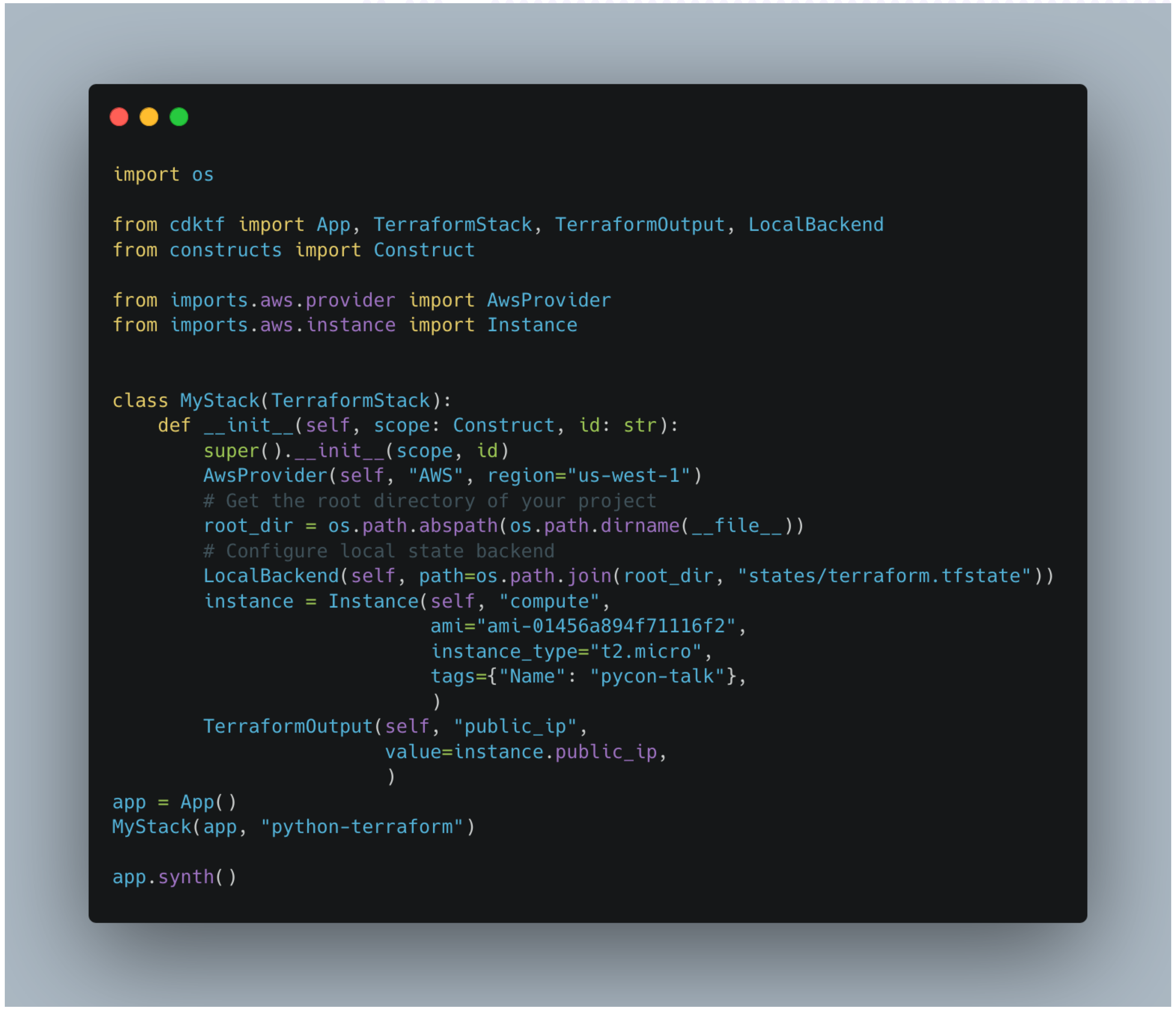
```python
#!/usr/bin/env python3
import os
from aws_cdk import (
    App,
    Stack,
    aws_ec2 as ec2,
    CfnOutput,
    Environment
)
from constructs import Construct

class AwsStack(Stack):

    def __init__(self, scope: Construct, construct_id: str, **kwargs) -> None:
        super().__init__(scope, construct_id, **kwargs)

        # Retrieve the default VPC for the specified region
        default_vpc = ec2.Vpc.from_lookup(self, "DefaultVpc", is_default=True)

        # Define an Amazon Linux EC2 instance
        ec2_instance = ec2.Instance(self, "MyInstance",
                                    instance_type=ec2.InstanceType("t2.micro"),
                                    machine_image=ec2.MachineImage.latest_amazon_linux2(),
                                    vpc=default_vpc)

        # Output the public IP of the instance
        self.output_public_ip(ec2_instance)

# Initialize the CDK app
app = App()
account = os.getenv('CDK_DEFAULT_ACCOUNT')
region = "us-west-1"

AwsStack(app, "MyAwsStack", env=Environment(account=account, region=region))
app.synth()
```

Popular Python Tools for IaC.

Terraform CDK (Cloud Development Kit)

- **Purpose:** Allows you to define cloud infrastructure using familiar programming languages, including Python.
- **Example Use:** Create Terraform configuration using Python code.

Sample Code 2.



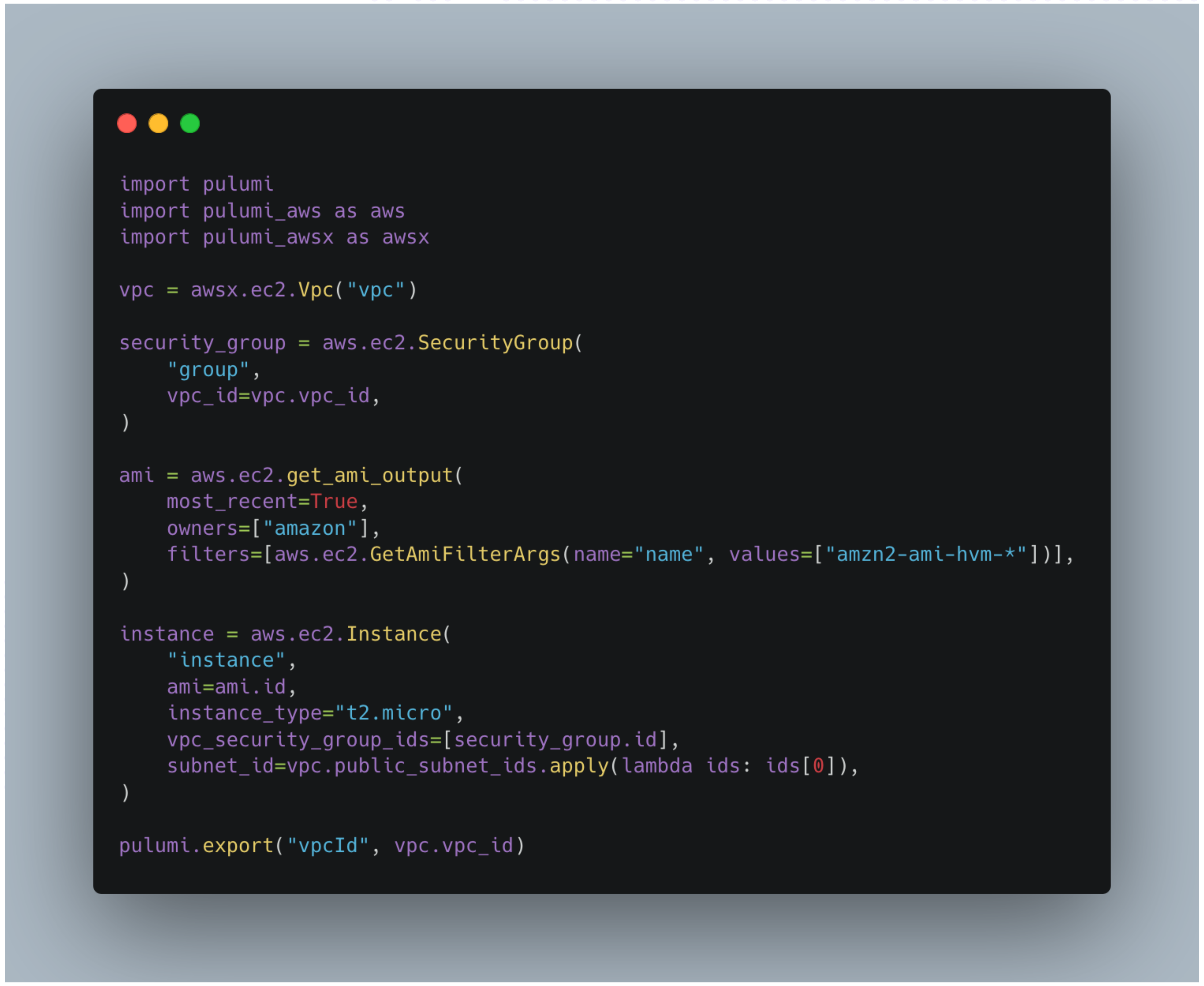
```python
import os

from cdktf import App, TerraformStack, TerraformOutput, LocalBackend
from constructs import Construct

from imports.aws.provider import AwsProvider
from imports.aws.instance import Instance


class MyStack(TerraformStack):
    def __init__(self, scope: Construct, id: str):
        super().__init__(scope, id)
        AwsProvider(self, "AWS", region="us-west-1")
        # Get the root directory of your project
        root_dir = os.path.abspath(os.path.dirname(__file__))
        # Configure local state backend
        LocalBackend(self, path=os.path.join(root_dir, "states/terraform.tfstate"))
        instance = Instance(self, "compute",
                            ami="ami-01456a894f71116f2",
                            instance_type="t2.micro",
                            tags={"Name": "pycon-talk"},
                            )
        TerraformOutput(self, "public_ip",
                        value=instance.public_ip,
                        )
app = App()
MyStack(app, "python-terraform")

app.synth()
```

Popular Python Tools for IaC.

Pulumi

- **Purpose:** Allows you to define cloud infrastructure using familiar programming languages, including Python.
- **Example Use:** Create AWS resources using Python code.

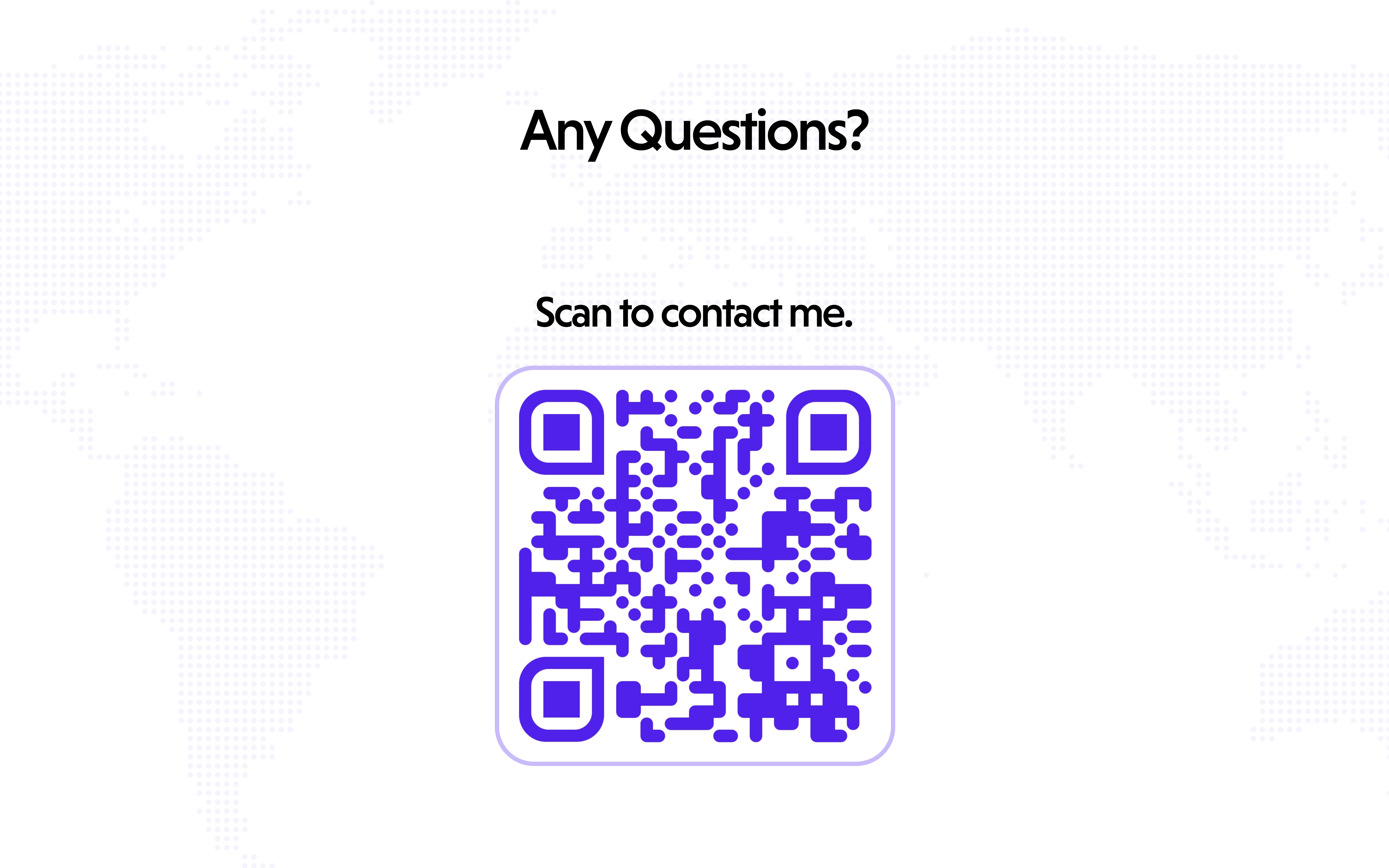
Sample Code 3.



```python
import pulumi
import pulumi_aws as aws
import pulumi_awsx as awsx

vpc = awsx.ec2.Vpc("vpc")

security_group = aws.ec2.SecurityGroup(
    "group",
    vpc_id=vpc.vpc_id,
)

ami = aws.ec2.get_ami_output(
    most_recent=True,
    owners=["amazon"],
    filters=[aws.ec2.GetAmiFilterArgs(name="name", values=["amzn2-ami-hvm-*"])],
)

instance = aws.ec2.Instance(
    "instance",
    ami=ami.id,
    instance_type="t2.micro",
    vpc_security_group_ids=[security_group.id],
    subnet_id=vpc.public_subnet_ids.apply(lambda ids: ids[0]),
)

pulumi.export("vpcId", vpc.vpc_id)
```

Enough talking

Now let's get our hands dirty.



Any Questions?

Scan to contact me.