Lab 2: Deploy EC2 + IAM role + S3 with parameters from SSM

1. Pre-Lab Setup (Manually in AWS CLI)

Let's store some parameters in SSM that we'll read from CDK:

```
aws ssm put-parameter --name "/lab2/instanceType" --value "t3.micro" --type String
aws ssm put-parameter --name "/lab2/keyName" --value "my-keypair" --type String
aws ssm put-parameter --name "/lab2/bucketName" --value "lab2-cdk-bucket-12345" --type String

```

✅ Replace `my-keypair` with your existing EC2 key pair name.

✅ Bucket name must be **globally unique**.

---

2. Project Initialization

```
mkdir lab2-ec2-iam-s3 && cd lab2-ec2-iam-s3
cdk init app --language python

python3 -m venv .venv
source .venv/bin/activate
pip install -r requirements.txt
pip install aws-cdk-lib constructs

```

---

3. Stack Code

Edit `lab2_ec2_iam_s3/lab2_ec2_iam_s3_stack.py`:

```
from aws_cdk import (
    Stack,
    aws_ec2 as ec2,
    aws_iam as iam,
    aws_s3 as s3,
    aws_ssm as ssm,
    RemovalPolicy
)
from constructs import Construct

class Lab2Ec2IamS3Stack(Stack):

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

        # ---- Fetch parameters from SSM ----
        instance_type = ssm.StringParameter.from_string_parameter_name(
            self, "InstanceTypeParam",
            string_parameter_name="/lab2/instanceType"
        ).string_value

        key_name = ssm.StringParameter.from_string_parameter_name(
            self, "KeyNameParam",
            string_parameter_name="/lab2/keyName"
        ).string_value

```

```
        bucket_name = ssm.StringParameter.from_string_parameter_name(
            self, "BucketNameParam",
            string_parameter_name="/lab2/bucketName"
        ).string_value

        # ---- IAM Role for EC2 ----
        role = iam.Role(self, "EC2Role",
            assumed_by=iam.ServicePrincipal("ec2.amazonaws.com"),
            managed_policies=[
iam.ManagedPolicy.from_aws_managed_policy_name("AmazonS3FullAccess")
            ]
        )

        # ---- Security Group ----
        vpc = ec2.Vpc(self, "VPC", max_azs=2)
        sg = ec2.SecurityGroup(self, "EC2SG",
            vpc=vpc,
            description="Allow SSH",
            allow_all_outbound=True
        )
        sg.add_ingress_rule(ec2.Peer.any_ipv4(), ec2.Port.tcp(22),
"Allow SSH")

        # ---- EC2 Instance ----
        ec2_instance = ec2.Instance(self, "MyInstance",
            vpc=vpc,
            instance_type=ec2.InstanceType(instance_type),
            machine_image=ec2.MachineImage.latest_amazon_linux2(),
            key_name=key_name,
            role=role,
            security_group=sg
        )

        # ---- S3 Bucket ----
        bucket = s3.Bucket(self, "MyBucket",
            bucket_name=bucket_name,
            versioned=True,
            removal_policy=RemovalPolicy.DESTROY,
            auto_delete_objects=True
        )

```

---

4. Synthesize & Deploy

```
cdk synth
cdk deploy

```

CDK will:

- Fetch values from **SSM**.
- Create a **VPC** with 2 AZs.
- Launch **EC2 instance** with IAM role (S3 access).
- Create an **S3 bucket** with name from SSM.

---

5. Validate

- In AWS Console → EC2 → check your new instance.
- Connect via SSH:

```
ssh -i my-keypair.pem ec2-user@<EC2_PUBLIC_IP>

```

- List S3 buckets from instance (thanks to IAM role):

```
aws s3 ls

```

---

6. Cleanup

```
cdk destroy

```

✅ Summary

In this lab, you:

- Stored **parameters in SSM** (`instanceType` , `keyName` , `bucketName`).
- Pulled them dynamically in **CDK v2 (Python)**.
- Deployed an **EC2 instance with IAM role** and an **S3 bucket**.
- Validated by accessing S3 from the instance.