# Exercise 4

As you learned in the lecture, you should not use your AWS account root user credentials to access AWS. Instead, create an AWS IAM user and assign permissions only necessary for the work done by the user. In this exercise, you will create an AWS IAM user, attach a customer managed AWS IAM policy to the user and set up access keys for the AWS IAM user.

An AWS IAM user is an entity that you create in AWS to represent the person or service that uses it to interact with AWS. You attach permission policies to the IAM user that determine what the user can and cannot do in AWS.
Access keys are a combination of an access key ID and a secret access key that are assigned to a user. These can be used to make programmatic calls to AWS when using the API in program code or at a

command prompt when using the AWS CLI.
**For all subsequent exercises, make sure to log in with the AWS IAM user credentials you create in this exercise, rather than the root user credentials.**

You will also create an Amazon EC2 instance, SSH into the instance, and configure AWS CLI to explore the AWS CLI commands. Then you will install Boto 3 on the instance and try out some Python scripting on the terminal. Boto 3 is the AWS SDK for Python, making it easier to integrate your Python application, library, or script with AWS services.

To begin, follow the instructions below.

## 1. Create an AWS IAM policy.

In this section, you will create an AWS IAM customer-managed policy. Customer-managed policies provide more precise control over your policies than AWS managed policies. This policy will have permissions specific to the AWS resources needed for the application you will build in this course.

- In the AWS Management Console, click **Services**, then click **IAM** to open the **IAM dashboard**.
- In the left navigation menu, click **Policies**.
- Click **Create policy**.
- Click the **JSON** tab.
- In the editor textbox, completely replace the sample policy with the following.

```
{
    "Version": "2012-10-17",
```

```
    "Statement": [
        {
            "Sid": "Sid1",
            "Effect": "Allow",
            "Action": [
                "iam:*",
                "rds:*",
                "sns:*",
                "cloudformation:*",
                "rekognition:*",
                "ec2:*",
                "cognito-idp:*",
                "sqs:*",
                "xray:*",
                "s3:*",
                "elasticloadbalancing:*",
                "cloud9:*",
                "lambda:*",
                "tag:GetResources",
                "logs:*",
                "kms:ListKeyPolicies",
                "kms:GenerateRandom",
                "kms:ListRetirableGrants",
                "kms:GetKeyPolicy",
                "kms:ListResourceTags",
                "kms:ReEncryptFrom",
                "kms:ListGrants",
                "kms:GetParametersForImport",
                "kms:ListKeys",
                "kms:GetKeyRotationStatus",
                "kms:ListAliases",
                "kms:ReEncryptTo",
                "kms:DescribeKey"
            ],
```

```
            "Resource": "*"
        }
    ]
}
```

- Click **Review Policy**.
- For **Name**, type **edXProjectPolicy**
- Click **Create policy**.
  You have successfully created an AWS IAM policy with full access to AWS IAM, Amazon EC2, Amazon S3, Amazon RDS, Amazon SNS, Amazon SQS, Amazon Rekognition, AWS Lambda, Amazon Cognito, AWS Cloud9, AWS X-Ray, and AWS CloudFormation. When you create IAM policies, follow the standard security advice of granting **least privilege** - that is, granting only the permissions required to perform a task. Determine what users need to do and then craft policies for them that let the users perform only those tasks.

## 2. Create an AWS IAM user, attach a policy to the user, and generate access keys.

In this section, you will create an AWS IAM user and attach the policy you just created to the user. You will then generate the access keys for the user. Those access keys will be used to make programmatic calls to AWS services via AWS CLI or APIs. If you are familiar with AWS IAM users, you may want to attempt to complete this section before reading the step-by-step instructions.

AWS IAM user name: edXProjectUser
Access type: Programmatic access and AWS Management Console access
Policy: edXProjectPolicy
**Important:** Download the **.csv file** with the access keys after creating

the user. Also, make sure to click the **Send email** link to get the email instructions for signing in to the AWS Management Console as edXProjectUser.

**Reminder!** *Be sure to protect your AWS account access keys like you would your credit card numbers or any other sensitive secret.*

At the end of this exercise, you will not be using the access keys again. It is a security best practice to remove IAM user credentials that are not needed. After this exercise, make sure to remove the access keys only (not the AWS Console password) for the IAM user - edXProjectUser. See more IAM Best Practices.

▶ Expand for step-by-step instructions.

## 3. Create an Amazon EC2 instance and configure AWS CLI with the access keys of the AWS IAM user edXProjectUser.

- Sign-in to your AWS account as the **edXProjectUser** AWS IAM user.
- Create an Amazon EC2 instance using the properties below. If you are familiar with Amazon EC2, you may want to attempt to complete this portion before reading the step-by-step instructions.

  Region: Oregon (us-west-2)
  Amazon Machine Image (AMI): Amazon Linux AMI *(Do not use the Amazon Linux 2 AMI)*
  Instance Type: t2.micro
  Network VPC: edx-build-aws-vpc
  Subnet: edx-subnet-public-a
  Tag: Ex4WebServer
  Security group name: Use the security group created in the third exercise, exercise3-sg.
  Key Pair: Use the key pair created in the third exercise.

▶ Expand for step-by-step instructions

- Connect to the instance using SSH. You may refer to the instructions in the **third exercise** for connecting to the instance.
- Open the **credentials.csv** file that you downloaded earlier. Find the entry for **edXProjectUser**, and note the values for **Access Key Id** and **Secret Access Key**.
- On the instance terminal, type the below command.

```
aws configure
```

- Follow the prompts on the screen and paste in the values for **Access Key Id** and **Secret Access Key.**
- For **Region**, type **us-west-2**.
- For **Default output format**, press ENTER.
  You have now configured the AWS CLI so that any CLI calls will operate with the credentials of the AWS IAM user edXProjectUser.
- Now query the information about the Amazon EC2 instances in your account. Type the command below.

```
aws ec2 describe-instances
```

  You should see a JSON output with all the information of the Amazon EC2 instances in your account. This means that you were able to successfully execute the AWS CLI command with the permissions attached to the edXProjectUser.

## 4. Install Boto 3 on the instance and explore Boto 3 APIs.

- First install Python 3 and the Boto 3 SDK. On the Amazon EC2 instance terminal, type the commands below.

```
sudo yum -y install python36
sudo pip-3.6 install boto3
```

- To start using Boto 3, type **python3** on the instance terminal and press ENTER. You should now be able to execute Python commands from your instance terminal.
- Import boto3 and create a client for the corresponding AWS service you wish to use. In this case, you can explore the EC2 APIs for Boto 3 by creating the EC2 client. Type the following.

  ```
  import boto3
  client = boto3.client('ec2')
  client.describe_instances()
  ```

  You should see a JSON output similar to the one given by the AWS CLI command.
- Now, type the command below.

  ```
  client.describe_key_pairs()
  ```

  You should see a JSON output with the information about the key pairs in your account.
- Press **Ctrl-D** to exit the python interpreter.

## Optional Challenge

In this exercise, you configured an access key (access key ID and secret access key) on your EC2 instance. Later in the course, we will introduce IAM *roles*. You may want to read ahead a little, and look at IAM Roles for Amazon EC2.

Can you see a way to complete this exercise using an IAM role on the instance, rather than the access keys you just used?

## 5. Terminate the Amazon EC2 instance.

In this section, you will terminate the Amazon EC2 instance by selecting the instance in the Amazon EC2 dashboard and clicking **Actions -> Instance State -> Terminate** .

▶ Expand for step-by-step instructions.

Learn About Verified Certificates