## IAM Users & Groups:

Root account created by default, shouldn't be used to shared
- Users are people within your organization, can be grouped
- Groups only contain users, not other groups
- Users don't belong to a group, and user can belong to multiple groups

- Sign up for an AWS account
- Using search bar, navigate to the "**IAM**" console.
- On the left-hand side, go to **users**.

  *Note that the region section indicated that is a global service

- Click on **Create User**
- Enter is Username, set a password for the user, add/create permission group associated with the user
- *optional* add tags -AWS can be added to organize resources
- Review & create.
- Click on User List to navigate back to the IAM page: under the **Access Management** tab: you may find: **User Groups** (which you created**), Users** & **Roles**
- To log into the console with the user created: Go to **Dashboard:**
  - on the right hand under **AWS Account:** add an Account **Alias** to simplify the Root-user login URL (has to be unique)

-Log into the console using the user created: Open a different browser than your current one (to avoid being logged out of your Root Account) –

-copy your root users **Account ID/ URL** & paste into the new browser on the log in page: it will prompt = Root User/ IAM User: select IAM user & enter your new users: user name & password

successfully logged into Root user & IAM User simultaneously

## IAM MFA:

MFA = password you know + security device you own

Benefits: if a password is stolen/hacked, the account is not compromised

Set up MFA on your AWS ACCOUNT in your **Root** Account

-Start by defining a password policy: Using the left-hand tab, go to: **Account Settings**

**- Edit** Password Policy & click **Custom.** Edit the policy & hit **Save Changes**

**-** In the top right-hand corner- hit your account drop-down arrow & go to **Security Credentials**

**-**Click **Assign MFA,** name the Device, & select the type of device: **Authenticator App**

*You may find compatible application using the link below on the page

-Download app & scan the QR code to begin the setup: complete instructions on the phone, & **Add MFA** successfully added an MFA to account.

## AWS Command-line Interface Lab

-Begin by installing CLI on your hardware: https://awscli.amazonaws.com/AWSCLIV2-2.0.30.msi

-complete installation steps & open up CLI from your windows browser

-Once you have the CLI up & running: enter the following prompt to confirm installation

‘’’ aws –version’’’

-Back to the AWS console: go to IAM > users > select your user > security credentials:

-Create Access Key > select: Command-Line-Interface > next: **Create Access Key**

*Note: only time you will have access to these credentials:

-Back to CLI>

‘’’aws configure’’’

*Enter your Access Key ID & then Secret Access Key >enter region name

‘’’eu-east-1’’’ & hit enter.

-to prompt a list of your aws users:

‘’’aws iam list-users’’’

-to create a new-user

‘’’ aws iam create-user --username (enter username)

-Delete Access key for user

‘’’ aws iam delete-access-key --username (username) --access-key-id (Access Key)

Enter various prompts to familiarize yourself with Command-Line Interface Lab

## AWS Cloud Shell

IAM roles for services

- Some AWS service will need to perform actions on your behalf
- To do so, we will assign permissions to AWS services with IAM roles

Common roles:  EC2 instance roles, Lambda function roles, roles for CloudFormation

-role is a way to give AWS entities permissions to do stuff on AWS

Create roles for your users

-Inside your AWS console: search **CloudShell >** Select the region from which you want to operate

-aws iam create-role --role-name S3ReadOnlyRole --assume-role-policy-document file://trust-policy.json

## AWS & EC2

-Using the search bar in your console: go to **EC2 > Instances** (left hand side bar) >Launch Instance:

-Name your instance, select **Amazon Linux 2 AMI (**Free tier), select instance type: **t2.micro** (free tier), create: **key pair: RSA encrypted (.pem) -**hit Create key pair. The key pair will download to your compute (be sure that it is saved to your hardware disk).

-Select security group(s) – Allow SSH traffic from anywhere (0.0.0.0/0) & Allow HTTP traffic (to launch a web server)

-Configure storage- leave default settings, select volume **Delete on Termination** for volume

-Under Advanced settings: leave everything else as-is & scroll down to **User data :**

**EC2 User data**

- It is possible to bootstrap our instance using EC2 user data script
- Bootstrapping means launching commands when a machine starts
- That script is only run once at the instance first start
- EC2 user data is used to automate boot tasks such as:

· Install updates , install software, downloading common files from the internet, anything you can think of

- The EC2 user data script runs with the root user

Use the code below to copy & paste into **User data:** this script is going to executed when the instance is first started & only one in the entire lifecycle:

```
#!/bin/bash

# Use this for your user data (script
from top to bottom)

# install httpd (Linux 2 version)

yum update -y

yum install -y httpd

systemctl start httpd

systemctl enable httpd

echo "<h1>Welcome to my page
$(hostname -f)</h1>" >
/var/www/html/index.html
```

-Finally, **Launch Instance**

**-Refresh** from the instances page & wait until the Instance State shows: **Running**

**-**Once started: under **Details**: you may find **Instance ID** & your **public IPv4 address**

-copy & paste your public IPv4 address into a new tab: "http://(public ipv4 address) & hit paste -note that you MUST use the http:// protocol, or the script will not work

successfully launched your first EC2 instance

## Connect to EC2 instance via SSH:

-open up Windows PowerShell: (ensure theres no spaces in your .pem file name)

-Follow prompts: -first you must be in the directory where your .pem file exists.

1. PS C:\Users\zaina\Documents\AWS training> "ls"

2. PS C:\Users\zaina\Documents\AWS training> "ssh -I [enter your .pem file name] ec2-user@[your public ipv4 address from your EC2 instance]"

3.It may ask you to confirm: hit YES

4. Next, update your EC2 instance:

    ```bash

    Sudo yum update -y ```

5.you may get the following response: You need to be root to perform this command.

6. to switch to root user: `sudo su`
    ```

## EC2 Instance Connect: an Alternative to SSH

-Go to instances > select instance & hit **Connect**

-You are instantly connected to your EC2 instance & may perform commands:

## EBS-Elastic Block Store

-Go to your instances > select instance > Storage tab > Under block device: Select the volume attached to the EC2 instance > Create Volume:

-select the following options: **General purpose SSD(gp2) >** Size**: 2 GiB >** Select the same availability zone as your Ec2 instance (us-east-1) > Create volume

-Once the volume is running: Attach it to your running EC2 instance:

-Go to EC2 instance & terminate your instance: go back to Elastic Block store & **note that the volume is also terminated -** lpre-set when launching our EC2 instance

## EBS Snapshots

### -Make a backup(snapshot) of your EBS volume at a point in time

-Go to Elastic block store> select volume & hit actions > **Create Snapshot >** add description of the snapshot > Create snapshot

-Go to Snapshot under EBS > select snapshot & right click (copy snapshot) > gives the ability to change the availability zone of the snapshot.

-Next, go to Snapshots > action – create volume from snapshot > allows you to save in a different availability zone> create volume. Go to volumes > both volumes are now running in different A/Z.

-Go to Snapshots > Recycle bin -protects snapshots and Amis from accidental deletion > Retention rules > Create retention rule: name , resource type: EBS snapshot, retention period (1 day), rule lock setting (unlock) > Create retention rule.

-Back to snapshots > delete snapshot > back to recycle bin – the snapshot is stored in the recycle bin for safety measure > recover the snapshot.

## AMI

## Amazon S3 lab

-Search bar >S3 > Create bucket:

-Unique bucket name, select AWS region, Disable ACL's, Block public access to bucket, disable versioning, default settings for encryption> create bucket

-Inside your computer: create a folder: S3 lab & add files such as images & documents:

-Open the bucker & upload 2 files (text, 2 image)

Next add policies to your bucket:

-Open up bucket> permissions> **allow public access** > scroll down to bucket policy **(Edit) >** policy generator: type: **S3 bucket policy**, effect: **allow**, Principal **"*"** =allow, Actions: **GetObject**, Amazon Resource name: (bucket ARM/*) >Add statement> Generate policy: copy & paste into the policy box!

```
{
  "Version": "2012-10-17",
  "Id": "Policy1709231396065",
  "Statement": [
    {
      "Sid": "Stmt1709231388704",
      "Effect": "Allow",
      "Principal": "*",
      "Action": "s3:GetObject",
      "Resource": "arn:aws:s3:::mylabbucket1203/*"
    }
  ]
}
```

publicly view the objects inside your bucket.

S3 Website:

-S3 Bucket > Properties > Static Website Hosting> enable > host static website > index document: index.html > save changes

-upload the below document onto **S3 Lab folder** on your COMPUTER using **index.html** : & then upload document into S3 bucket:

```
<html>

   <head>

      <title>My First Webpage</title>

   </head>

   <body>

      <h1> input personal
message</h1>

      <p>customize this section </p>

   </body>



   <img src="your photo.jpg"
width=500/>

</html>
```

-S3 Bucket> properties> Bucket website endpoint: click on the link to open up your website.

**S3 Versioning**:

-S3 Bucket > Properties > enable bucket versioning > save changes

-Open up your index.html & make changes to the personal messages > upload new version

-Toggle on **Show versions** – notice that there are version IDs

-refresh your static hosted website to view changes.

-S3 bucket: Click on the indem.html with version id & delete the second version to get rid of the new changes.

## AWS RDS

-Create Database > Standard Create > MYSQL > Version (8.0.23) > Template: Free Tier > DB Instance identifier: name DB > Master credentials: username & password.

-DB Instance class: db.t2.micro > storage: default settings > Connectivity: P/A: yes, >VPC: create new security group, no A/Z preference & DB port: 3306 > Create Database

-Once the DB is launched: connect RDS DB to EC2 instance:

- ``` ssh -i .\(EC2USER.pem) ec2-user@Instance public-IP address ````
- ``` yum install mysql -y````

-May get an indication that you have to be root user: ```sudo su``` & try again,

-Connect to RDS instance

- ``` mysql -h (RDS endpoint) -P (port) -u (username) -p
- Enter Password: ****
- Perform basic SQL operations: (

## Amazon Redshift And Quick Sight

-Under Amazon Redshift> create cluster: Name the cluster, Size of cluster: I'll chose: ra3.4xlarge, load sample data, configure database configurations (username, create password), default security settings & create cluster

- Create S3 bucket & upload a sample of data (Csv file)
- back to your RedShift cluster