LAB Step : Create an IAM User with Programmatic Access

1. **Log in to AWS Management Console**.
2. **Navigate to IAM**:
   - In the search bar, type "IAM" and select **IAM**.
3. **Create a New User**:
   - In the IAM Dashboard, click on **Users** on the left side.
   - Click **Add user**.
   - Provide a username for the new IAM user (e.g., `cli-user`).
   - Select **Programmatic access** under "Access type". This generates an Access Key ID and Secret Access Key for CLI use.
   - Click **Next: Permissions**.
4. **Assign Permissions**:
   - Select **Attach policies directly**.
   - Attach basic policies like `AdministratorAccess` or create a custom policy based on your needs.
   - Click **Next: Tags**.
5. **Review and Create**:
   - Review your settings and click **Create user**.
   - Note the **Access Key ID** and **Secret Access Key**. You'll use these to configure the AWS CLI.

Step 2: Install AWS CLI

1. Download and install AWS CLI:
   - For Windows: Download AWS CLI
   - For macOS: Run `brew install awscli` if you have Homebrew.
   - For Linux: Run `sudo apt install awscli` or `sudo yum install awscli` based on your distro.
2. Verify the installation by running:

```
aws --version
```

Step 3: Configure AWS CLI with Your Access Keys

1. Open your terminal/command prompt and run:

```
aws configure
```

2. Enter the following details:
   - **AWS Access Key ID**: (Enter the key from IAM creation step)
   - **AWS Secret Access Key**: (Enter the secret key)
   - **Default region name**: (e.g., `us-east-1`)
   - **Default output format**: (e.g., `json` or `text`)

Step 4: Practice Basic AWS CLI Commands

AWS CLI Commands to Practice

1. **List S3 Buckets**:

```
aws s3 ls
```

2. **Create an S3 Bucket**:

```
aws s3 mb s3://my-new-bucket
```

3. **Upload a File to S3**:

```
aws s3 cp myfile.txt s3://my-new-bucket/
```

4. **List EC2 Instances**:

```
aws ec2 describe-instances
```

5. **Create an EC2 Instance (Using a Specific AMI)**:

```
aws ec2 run-instances --image-id ami-xxxxxxxx --count 1 --instance-
type t2.micro --key-name my-key-pair
```

6. **List IAM Users**:

```
aws iam list-users
```

7. **Describe Available Regions**:

```
aws ec2 describe-regions
```

Step 5: Practice Advanced AWS CLI Commands

Advanced AWS CLI Commands to Try

1. **Stop an EC2 Instance**:

```
aws ec2 stop-instances --instance-ids i-xxxxxxxxxxxxxxxxx
```

2. **Start an EC2 Instance**:

```
aws ec2 start-instances --instance-ids i-xxxxxxxxxxxxxxxx
```

3. **Create an Auto Scaling Group**:

```
aws autoscaling create-auto-scaling-group --auto-scaling-group-name
my-auto-scaling-group --launch-configuration-name my-launch-config --
min-size 1 --max-size 5 --desired-capacity 2 --availability-zones us-
east-1a
```

4. **Configure an AWS Lambda Function**:

```
aws lambda create-function --function-name my-function --runtime
nodejs14.x --role arn:aws:iam::account-id:role/my-role --handler
index.handler --zip-file fileb://my-function.zip
```

5. **Create a CloudWatch Alarm**:

```
aws cloudwatch put-metric-alarm --alarm-name HighCPUUtilization --
metric-name CPUUtilization --namespace AWS/EC2 --statistic Average --
period 300 --threshold 80 --comparison-operator GreaterThanThreshold
--evaluation-periods 2 --alarm-actions arn:aws:sns:us-east-1:account-
id:my-topic
```

6. **Describe DynamoDB Table**:

```
aws dynamodb describe-table --table-name my-table
```

7. **CloudFormation Stack Operations**:
   - **Create a Stack**:

```
aws cloudformation create-stack --stack-name my-stack --
template-body file://template.json
```

   - **Delete a Stack**:

```
aws cloudformation delete-stack --stack-name my-stack
```

1. Advanced EC2 Commands

Start an EC2 Instance with Specific Tags

```
aws ec2 start-instances --instance-ids i-xxxxxxxx --tag-specifications
'ResourceType=instance,Tags=[{Key=Environment,Value=Production},{Key=Owner,
Value=Admin}]'
```

Stop Multiple EC2 Instances

```
aws ec2 stop-instances --instance-ids i-xxxxxxxx i-yyyyyyyy i-zzzzzzzz
```

Terminate an EC2 Instance

```
aws ec2 terminate-instances --instance-ids i-xxxxxxxx
```

Create an EC2 Key Pair

```
aws ec2 create-key-pair --key-name MyNewKeyPair --query 'KeyMaterial' --
output text > MyNewKeyPair.pem
```

Attach an EBS Volume to an Instance

```
aws ec2 attach-volume --volume-id vol-xxxxxxxx --instance-id i-xxxxxxxx --
device /dev/sdf
```

Modify an EC2 Instance Attribute (e.g., Change Instance Type)

```
aws ec2 modify-instance-attribute --instance-id i-xxxxxxxx --instance-type
'{"Value": "t2.medium"}'
```

Create an EC2 Security Group

```
aws ec2 create-security-group --group-name MySecurityGroup --description
"Security group for my app"
```

Authorize an Inbound Rule in a Security Group

```
aws ec2 authorize-security-group-ingress --group-id sg-xxxxxxxx --protocol
tcp --port 22 --cidr 0.0.0.0/0
```

Create a Custom Amazon Machine Image (AMI)

```
aws ec2 create-image --instance-id i-xxxxxxxx --name "MyCustomAMI" --no-reboot
```

2. Auto Scaling Commands

Create an Auto Scaling Group

```
aws autoscaling create-auto-scaling-group --auto-scaling-group-name my-auto-scaling-group --launch-configuration-name my-launch-config --min-size 1 --max-size 5 --desired-capacity 3 --availability-zones us-west-2a --vpc-zone-identifier subnet-xxxxxxxx
```

Update the Desired Capacity of an Auto Scaling Group

```
aws autoscaling update-auto-scaling-group --auto-scaling-group-name my-auto-scaling-group --desired-capacity 4
```

Create a Launch Configuration for Auto Scaling

```
aws autoscaling create-launch-configuration --launch-configuration-name my-launch-config --image-id ami-xxxxxxxx --instance-type t2.micro --security-groups sg-xxxxxxxx
```

Set up a Scaling Policy (Scale-In/Scale-Out)

```
aws autoscaling put-scaling-policy --auto-scaling-group-name my-auto-scaling-group --policy-name ScaleOutPolicy --scaling-adjustment 2 --adjustment-type ChangeInCapacity
```

Describe Auto Scaling Groups

```
aws autoscaling describe-auto-scaling-groups --auto-scaling-group-name my-auto-scaling-group
```

Delete an Auto Scaling Group

```
aws autoscaling delete-auto-scaling-group --auto-scaling-group-name my-auto-scaling-group --force-delete
```

3. Load Balancer (ELB) Commands

Create an Application Load Balancer (ALB)

```
aws elbv2 create-load-balancer --name my-load-balancer --subnets subnet-
xxxxxxxx subnet-yyyyyyyy --security-groups sg-xxxxxxxx --scheme internet-
facing --load-balancer-type application
```

Register EC2 Instances with a Target Group

```
aws elbv2 register-targets --target-group-arn
arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/my-
targets/50dc6c495c0c9188 --targets Id=i-xxxxxxxx Id=i-yyyyyyyy
```

Create a Listener for an ALB

```
aws elbv2 create-listener --load-balancer-arn
arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-
load-balancer/50dc6c495c0c9188 --protocol HTTP --port 80 --default-actions
Type=fixed-
response,FixedResponseConfig={StatusCode=200,ContentType=text/plain,Message
Body="OK"}
```

Describe Load Balancer

```
aws elbv2 describe-load-balancers --load-balancer-arns
arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-
load-balancer/50dc6c495c0c9188
```

Delete a Load Balancer

```
aws elbv2 delete-load-balancer --load-balancer-arn
arn:aws:elasticloadbalancing:us-west-2:123456789012:loadbalancer/app/my-
load-balancer/50dc6c495c0c9188
```

Delete Target Group

```
aws elbv2 delete-target-group --target-group-arn
arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/my-
targets/50dc6c495c0c9188
```

4. VPC Commands

Create a New VPC

```
aws ec2 create-vpc --cidr-block 10.0.0.0/16
```

Create a Subnet within the VPC

```
aws ec2 create-subnet --vpc-id vpc-xxxxxxxx --cidr-block 10.0.1.0/24 --
availability-zone us-west-2a
```

Create an Internet Gateway and Attach to the VPC

```
aws ec2 create-internet-gateway
aws ec2 attach-internet-gateway --vpc-id vpc-xxxxxxxx --internet-gateway-id
igw-xxxxxxxx
```

Create a Route Table

```
aws ec2 create-route-table --vpc-id vpc-xxxxxxxx
```

Associate Route Table with a Subnet

```
aws ec2 associate-route-table --route-table-id rtb-xxxxxxxx --subnet-id
subnet-xxxxxxxx
```

Modify a VPC Security Group (Add Inbound Rule)

```
aws ec2 authorize-security-group-ingress --group-id sg-xxxxxxxx --protocol
tcp --port 22 --cidr 0.0.0.0/0
```

Create a Network ACL (NACL)

```
aws ec2 create-network-acl --vpc-id vpc-xxxxxxxx
```

Add a Rule to a Network ACL

```
aws ec2 create-network-acl-entry --network-acl-id acl-xxxxxxxx --rule-
number 100 --protocol tcp --port-range From=80,To=80 --cidr-block 0.0.0.0/0
--egress --rule-action allow
```

Delete a Subnet

```
aws ec2 delete-subnet --subnet-id subnet-xxxxxxxx
```

Delete a VPC

```
aws ec2 delete-vpc --vpc-id vpc-xxxxxxxx
```

5. VPC Peering Commands

Create a VPC Peering Connection

```
aws ec2 create-vpc-peering-connection --vpc-id vpc-xxxxxxxx --peer-vpc-id
vpc-yyyyyyyy
```

Accept a VPC Peering Request

```
aws ec2 accept-vpc-peering-connection --vpc-peering-connection-id pcx-
xxxxxxxx
```

Delete a VPC Peering Connection

```
aws ec2 delete-vpc-peering-connection --vpc-peering-connection-id pcx-
xxxxxxxx
```