**LAB: Launch Windows EC2 Instance and Connect using RDP**

---

## Step 1: Sign in to AWS Management Console

1. Go to AWS Console.
2. Log in with your AWS credentials.

💡 *Explanation:* The AWS Management Console is the user interface to manage your AWS services, including EC2 (Elastic Compute Cloud), where you can launch and manage instances.

---

## Step 2: Navigate to EC2 Dashboard

1. In the AWS Console, search for **"EC2"** in the top search bar and select **EC2** under **Services**.
2. This will take you to the **EC2 Dashboard**, where you can manage all your EC2 instances.

💡 *Explanation:* EC2 allows you to launch virtual machines (instances) in the cloud. The EC2 Dashboard is where you'll manage these instances.

---

## Step 3: Launch Instance

1. On the EC2 Dashboard, click **"Launch Instance"** to start the process of creating a new EC2 instance.

💡 *Explanation:* This initiates the process to create and configure a new instance.

---

## Step 4: Choose an Amazon Machine Image (AMI)

1. **Select Windows AMI**: In the **Choose an Amazon Machine Image (AMI)** section, select a **Windows** AMI. For example, **Microsoft Windows Server 2019 Base**.

💡 *Explanation:* The AMI is a pre-configured template containing the operating system and software you need for your instance. Selecting a Windows AMI will ensure the instance runs Windows.

---

## Step 5: Choose an Instance Type

1. Choose an **Instance Type** (e.g., **t2.micro** for free-tier eligible or another according to your needs).
2. Click **"Next: Configure Instance Details"**.

💡 *Explanation:* The instance type defines the hardware configuration (CPU, memory, storage, etc.) for your EC2 instance. For small workloads, the **t2.micro** is a good starting point and eligible for the free tier.

---

## Step 6: Configure Instance Details

1. Set the **number of instances** and configure any other specific details (e.g., Network, Subnet).
2. Once done, click **"Next: Add Storage"**.

💡 *Explanation:* This step allows you to set up network configurations like VPC, subnets, and any other advanced settings like IAM roles or monitoring.

---

## Step 7: Add Storage

1. By default, the instance will come with a root volume. You can add additional volumes or modify the size of the root volume.
2. Once you're satisfied, click **"Next: Add Tags"**.

💡 *Explanation:* You can attach storage volumes (EBS) to your instance. You can adjust the size or add more storage as needed.

---

## Step 8: Add Tags

1. Add tags to your instance for better organization (e.g., `Key: Name, Value: MyWindowsInstance`).
2. Click **"Next: Configure Security Group"**.

💡 *Explanation:* Tags help you label and organize resources in AWS for easy identification. A Security Group acts like a virtual firewall for your instance, controlling inbound and outbound traffic.

---

## Step 9: Configure Security Group

1. Choose **Create a new security group**.
2. Add a rule to allow **RDP (Remote Desktop Protocol)**:
    - Type: **RDP**
    - Protocol: **TCP**
    - Port: **3389**
    - Source: **Anywhere** (or specify a specific IP range for security reasons).
3. Click **"Review and Launch"**.

💡 *Explanation:* The Security Group allows you to define what type of network traffic is allowed to and from the instance. RDP is necessary to connect to your Windows instance.

---

## Step 10: Review and Launch

1. Review all your configurations.
2. Click **"Launch"**.
3. A prompt will appear asking you to select an existing key pair or create a new one. Select **Create a new key pair**, name it, and download the private key file (.pem).
4. Click **"Launch Instances"**.

💡 *Explanation:* The **key pair** is used for secure access to your instance. The private key file should be saved safely because it's used for SSH access (for Linux) or RDP for Windows instances.

---

## Step 11: Connect to Your Instance

1. After the instance is running, go to the **Instances** section of the EC2 dashboard.
2. Find your instance and click on its **Instance ID**.
3. Once the instance status is **running**, click **Connect** at the top of the page.

---

## Step 12: Retrieve the Administrator Password

1. In the **Connect to your instance** window, under the **RDP Client** tab, click on **Get Password**.
2. Upload your downloaded **.pem key file** to decrypt the password for the Windows instance.

💡 *Explanation:* The password is encrypted and can be decrypted using the key pair file you downloaded earlier.

---

## Step 13: Download RDP File and Connect

1. After retrieving the password, click **Download RDP File** to download a pre-configured RDP file.
2. Open the RDP file on your computer. Enter the username **Administrator** and the decrypted password.

💡 *Explanation:* RDP allows you to connect to the Windows instance via a graphical user interface. You'll now be able to control and use your instance as if it were a local Windows machine.

Lab2: **LAB1: Launch Windows EC2 Instance and Connect using RDP**

### Step 1: Sign in to AWS Management Console

1. Go to AWS Console.
2. Log in with your AWS credentials.

💡 *Explanation:* The AWS Management Console is a web-based interface for managing your AWS services, including EC2, where you can launch and manage virtual machines.

---

### Step 2: Navigate to EC2 Dashboard

1. In the AWS Console, search for **"EC2"** in the top search bar and select **EC2** under **Services**.
2. This will take you to the **EC2 Dashboard**.

💡 *Explanation:* The EC2 Dashboard allows you to manage your instances and other EC2 resources.

---

### Step 3: Launch Instance

1. On the EC2 Dashboard, click **"Launch Instance"** to start the process of creating a new EC2 instance.

💡 *Explanation:* Clicking this button starts the process of creating and configuring a new EC2 instance.

---

## Step 4: Choose an Amazon Machine Image (AMI)

1. **Select Ubuntu AMI**: In the **Choose an Amazon Machine Image (AMI)** section, select an **Ubuntu** AMI (e.g., **Ubuntu Server 20.04 LTS**).
2. Click **"Next: Choose an Instance Type"**.

💡 *Explanation:* An AMI is a pre-configured operating system template. Ubuntu AMIs provide a ready-to-use Linux operating system for your EC2 instance.

---

## Step 5: Choose an Instance Type

1. Choose an **Instance Type** (e.g., **t2.micro** for free-tier eligible or another instance type depending on your needs).
2. Click **"Next: Configure Instance Details"**.

💡 *Explanation:* The instance type determines the hardware configuration of your instance (e.g., CPU, memory, storage).

---

## Step 6: Configure Instance Details

1. Configure any specific details such as the number of instances, network, subnet, and IAM roles (if required).
2. Once done, click **"Next: Add Storage"**.

💡 *Explanation:* In this step, you define networking and other configuration settings. You can also create multiple instances if needed.

---

## Step 7: Add Storage

1. By default, a root volume will be created for the instance. You can modify the size or add additional volumes as necessary.
2. Click **"Next: Add Tags"**.

💡 *Explanation:* Storage (EBS volumes) is used to store data persistently. You can configure additional volumes or adjust the size of the root volume.

---

## Step 8: Add Tags

1. Add tags for organizing your resources (e.g., `Key: Name, Value: MyUbuntuInstance`).
2. Click **"Next: Configure Security Group"**.

💡 *Explanation:* Tags help you label and track resources in AWS. A Security Group acts as a virtual firewall controlling access to your instance.

---

## Step 9: Configure Security Group

1. Choose **Create a new security group**.
2. Add the following inbound rules:
   - Type: **SSH**
   - Protocol: **TCP**
   - Port: **22**
   - Source: **Anywhere** (or specify a specific IP range for security purposes).
3. Click **"Review and Launch"**.

💡 *Explanation:* The Security Group defines which network traffic is allowed to reach your instance. SSH is the protocol used for secure login to your Linux instance.

---

## Step 10: Review and Launch

1. Review all your configuration settings.
2. Click **"Launch"**.
3. A prompt will ask you to select an existing key pair or create a new one. Choose **Create a new key pair**, name it, and download the private key file (**.pem**).
4. Click **"Launch Instances"**.

💡 *Explanation:* The **key pair** allows secure access to your instance. You'll use the private key file for SSH login.

---

## Step 11: Connect to Your Instance

1. Go to the **Instances** section of the EC2 Dashboard.
2. Locate your instance and wait until it shows the **running** status.
3. Click on the **Instance ID** of the instance you just created.

## Step 12: Retrieve the Public IP Address

1. In the **Description** section of your instance, find the **IPv4 Public IP**. This is the IP address you'll use to connect to your instance.

💡 *Explanation:* The public IP address allows you to connect to your instance from anywhere via the internet.

---

## Step 13: Connect Using SSH

1. Open a terminal on your local machine.
2. Use the following command to connect to your instance:

```
ssh -i /path/to/your-key.pem ubuntu@<your-public-ip>
```

   - Replace `/path/to/your-key.pem` with the path to your private key file.
   - Replace `<your-public-ip>` with the public IP address of your EC2 instance.
3. If prompted about the authenticity of the host, type **yes** to continue.

💡 *Explanation:* SSH is the protocol used to securely connect to your Linux instance. The `ubuntu` user is the default administrative user on Ubuntu-based instances.

---

## Step 14: Access Your Instance

- Once connected, you'll have terminal access to your Ubuntu instance, and you can start managing the system, install packages, and run commands.

Lab 3:Installing webserver on Windows and Linus Instances:

3a) Installing Apache Web Server on Ubuntu (Linux) EC2 Instance

### *Step 1: Connect to Your EC2 Instance*

1. **SSH into your Ubuntu instance**:

```
ssh -i /path/to/your-key.pem ubuntu@<your-public-ip>
```

💡 *Explanation:* The first step is connecting to your EC2 instance using SSH. You use the private key (`.pem`) and public IP of your instance to authenticate and access it.

---

### *Step 2: Update System Packages*

1. Run the following commands to update the package index:

```
sudo apt update
sudo apt upgrade -y
```

💡 *Explanation:* It's always a good practice to update your system to the latest versions of packages to avoid security vulnerabilities and to ensure that you have the latest features.

---

### *Step 3: Install Apache Web Server*

1. Run the following command to install Apache:

```
sudo apt install apache2 -y
```

💡 *Explanation:* Apache is one of the most widely used open-source web servers. The `-y` flag automatically confirms the installation of dependencies.

---

### *Step 4: Enable and Start Apache Service*

1. After the installation is complete, start the Apache service:

```
sudo systemctl start apache2
```

2. To ensure Apache starts automatically on system boot, run:

```
sudo systemctl enable apache2
```

💡 *Explanation:* The `systemctl` command manages services on Linux. The `start` command launches the Apache service, and the `enable` command ensures it starts every time the system reboots.

---

### *Step 5: Configure Firewall (If Necessary)*

1. If your instance uses **UFW** (Uncomplicated Firewall), you need to allow HTTP traffic (port 80):

```
sudo ufw allow 'Apache'
sudo ufw enable
```

💡 *Explanation:* This step ensures that incoming HTTP requests can reach the Apache server on port 80, which is the default port for web traffic.

---

### *Step 6: Verify Apache Installation*

1. Open a web browser and type the **public IP address** of your instance:

```
http://<your-public-ip>
```

2. You should see the **Apache2 Ubuntu Default Page**, confirming that Apache has been installed and is working.

💡 *Explanation:* By navigating to your EC2 instance's public IP, you can verify that the Apache web server is successfully serving pages.

---

### *Step 7: Create a Simple Web Page (Optional)*

1. To modify the default web page, create an HTML file:

```
echo "<h1>Welcome to My Apache Web Server</h1>" | sudo tee
/var/www/html/index.html
```

💡 *Explanation:* This command replaces the default page with a simple custom web page. The file is stored in the `/var/www/html` directory, which is the default location for web content on Apache.

# Alternatives:

Linux machine:user Data

#!/bin/bash

# Loop until internet is available with a delay to handle slow network

while ! ping -c 1 -W 5 8.8.8.8 &>/dev/null; do

    echo "Waiting for internet connection..."

    sleep 5 # Adding delay to avoid rapid retries

done

# Update and install httpd

sudo yum update -y

sudo yum install -y httpd

# Start and enable httpd service

sudo systemctl start httpd

sudo systemctl enable httpd

# Set permissions for /var/www/html

sudo chmod 777 -R /var/www/html

# Create index.html with server hostname

echo "<h1> Welcome to Rajco Session $(hostname -f) </h1>" > /var/www/html/index.html

:

Note:UserData at the  time of launching  EC2 instance(Ubuntu) use apt instead of yum

---

## 3b ) Installing IIS Web Server on Windows EC2 Instance

### *Step 1: Connect to Your EC2 Instance*

1. Use **RDP (Remote Desktop Protocol)** to connect to your Windows instance. You can retrieve the administrator password from the EC2 console and use an RDP client.

💡 *Explanation:* To access your Windows instance, use RDP with the decrypted password. RDP gives you a graphical user interface to interact with the instance.

---

### *Step 2: Open Server Manager*

1. After logging into Windows, **open Server Manager** from the Start menu (it might open automatically).

💡 *Explanation:* Server Manager is a tool used to manage and configure server roles and features in Windows Server editions.

---

### *Step 3: Add the Web Server (IIS) Role*

1. In **Server Manager**, click on **Manage** at the top right and select **Add Roles and Features**.
2. Click **Next** until you reach the **Select features** screen.

💡 *Explanation:* The **Add Roles and Features Wizard** allows you to install different server roles and features on your Windows instance.

---

### *Step 4: Select the IIS Role*

1. In the **Select roles** screen, check **Web Server (IIS)**.
2. A pop-up will ask to add required features; click **Add Features**.
3. Continue through the wizard by clicking **Next** and then **Install**.

💡 *Explanation:* **IIS (Internet Information Services)** is a feature that provides a web server for hosting web pages on Windows Server. This step installs IIS and its necessary components.

---

### *Step 5: Verify IIS Installation*

1. After installation is complete, click **Close**.
2. Open a browser on your Windows instance and type **localhost** or **127.0.0.1** in the address bar.
3. You should see the **IIS welcome page** confirming that IIS is installed and working.

💡 *Explanation:* By navigating to `localhost`, you can confirm that IIS is correctly serving the default web page.

---

### *Step 6: Configure the Firewall (If Necessary)*

1. Open **Windows Firewall** and ensure that **HTTP (Port 80)** is allowed through the firewall for inbound connections.

💡 *Explanation:* For your IIS server to be accessible externally, you need to ensure that HTTP traffic is allowed through the Windows Firewall.

---

### *Step 7: Create a Simple Web Page (Optional)*

1. Navigate to the default IIS directory, usually located at:

   ```
   C:¥inetpub¥wwwroot
   ```

2. Open **index.html** (or create a new HTML file) and add your custom content:

   ```
   <h1>Welcome to My IIS Web Server</h1>
   ```

💡 *Explanation:* This file will be served when users access the IIS web server. You can modify this file or add other files to customize your site.

*Step 8: Access IIS Web Server Externally*

1. Open a browser and enter the **public IP address** of your Windows EC2 instance:

   `http://<your-public-ip>`

2. You should now see the custom web page you created in IIS.

💡 *Explanation:* This step ensures that your IIS server is accessible from the internet by using the public IP of your EC2 instance.

## **Lab :** Creating an Amazon Machine Image (AMI) from an EC2 instance that has a web server installed and using it as a custom AMI to launch a new EC2 instance

### Step 1: Prepare the EC2 Instance

1. **Launch an EC2 Instance**
   - Go to the AWS Management Console → **EC2 Dashboard** → **Launch Instance**.
   - Choose an appropriate **Amazon Machine Image (AMI)** (e.g., Amazon Linux, Ubuntu).
   - Select an **Instance Type** (e.g., t2.micro for free tier).
   - Configure **Networking, Security Groups, and Storage** as required.
2. **Install a Web Server**
   - Connect to the instance via **SSH**:

     ```
     sh

     ssh -i my-key.pem ec2-user@your-instance-ip
     ```

   - Install and configure the web server (e.g., Apache on Amazon Linux):

     ```
     #!/bin/bash

     # Loop until internet is available with a delay to handle slow network
     while ! ping -c 1 -W 5 8.8.8.8 &>/dev/null; do
         echo "Waiting for internet connection..."
         sleep 5 # Adding delay to avoid rapid retries
     done

     # Update and install httpd
     sudo yum update -y
     ```

```
sudo yum install -y httpd
# Start and enable httpd service
sudo systemctl start httpd
sudo systemctl enable httpd
# Set permissions for /var/www/html
sudo chmod 777 -R /var/www/html
# Create index.html with server hostname
echo "<h1> Welcome to Rajco Session $(hostname -f) </h1>" >
/var/www/html/index.html
```

3. **Verify the Web Server**
    - Open a web browser and navigate to `http://<Public-IP>` to confirm the web server is running.

---

## Step 2: Create an AMI from the EC2 Instance

1. **Go to EC2 Dashboard → Instances.**
2. **Select the Instance** running the web server.
3. Click on **Actions → Image and templates → Create Image**.
4. Provide the following details:
    - **Image Name**: e.g., `MyWebServerAMI`
    - **Image Description**: e.g., `Custom AMI with pre-installed web server`
    - **No reboot** (optional): Check if you don't want to reboot the instance during AMI creation.
5. Click **Create Image**.
6. The AMI will be created and available under **EC2 Dashboard → AMIs**.

---

## Step 3: Launch a New EC2 Instance from the Custom AMI

1. **Go to EC2 Dashboard → AMIs.**
2. Select the **custom AMI** (`MyWebServerAMI`).
3. Click **Launch Instance from Image**.
4. Configure instance details:
    - Choose **Instance Type**.
    - Select **VPC and Subnet**.
    - Configure **Security Group** (Allow HTTP/HTTPS for web access).
    - Select an **existing key pair** or create a new one.
5. Click **Launch**.

---

## Step 4: Verify the New Instance

1. Go to **EC2 Dashboard → Instances**, find the new instance.
2. Copy the **Public IP Address**.
3. Open a browser and visit `http://<New-Public-IP>` to check if the web server is running.

## LAB: Create a Launch Template for EC2 Instances with custom AMI (Apache Installed)

A **Launch Template** defines the EC2 configuration for the Auto Scaling Group.

1. Navigate to **EC2 Dashboard → Launch Templates → Create Launch Template**.
2. **Set Basic Details:**
   - **Name:** `web-app-launch-template`
3. **AMI (Amazon Machine Image):**
   - Select Custom AMI(apache Installed)
4. **Instance Type:** `t2.micro` (or select based on needs).
5. **Key Pair:** Select an existing or create a new one.
6. **Network Settings:**
   - **Select the security group:** `web-app-sg`
7. **User Data (Bootstrapping Script):**
   - Install Apache Web Server automatically:

```
#!/bin/
yum update -y
systemctl start httpd
systemctl enable httpd
```

8. **Storage:** 8GB (default).
9. **IAM Role:** Attach an IAM role if needed.
10. **Save and Create Launch Template**.