Creating & Setup Instance in AWS

A. Initial Instance Creation

1. **Go to EC2 and select "Launch Instance".**

2. **Check the Region:**
   Before starting, make sure to select the correct region where you want to create the instance (e.g., N. Virginia `us-east-1`).
   The selected region is displayed in the top-right corner.

3. **Instance Name:**
   Provide a name for your instance (e.g., `ecommerce`).

4. **Select the OS:**
   Choose the operating system for your instance (e.g., Ubuntu).

5. **Instance Type:**
   The instance type will be automatically selected as per the free tier eligibility for that region.

6. **Create a Key Pair:**

   - Click on **Generate Key Pair**.

   - Provide a name for your key pair.

   - Set Key Pair Type to **RSA**.

   - Set Private Key Format to **.pem**.

   - **Note:** After generating, the private key file will be downloaded automatically. **Store it safely**, as you'll need it to connect via the terminal.

7. **Network Settings:**

   - You'll see the VPC (Virtual Private Cloud).

   - Edit the settings if needed and make sure it matches the same VPC as your database (if you're using AWS RDS).

- **Important:** The EC2 and database must be in the same VPC for connectivity.

8. **Storage Configuration:**

    - Allocate storage (up to 30GB is free in the Free Tier).

9. **Launch the Instance:**

    - Click on **Launch Instance**.

    - Your instance will be created.

B. Connecting to the Instance via AWS Terminal & Local Terminal

Using AWS Terminal (Browser-Based)

- Click **Connect** from the EC2 dashboard.

- It will automatically open a new tab showing your Ubuntu EC2 instance terminal.

Using Local Terminal (PC Command Line)

1. Remember the private key file you downloaded earlier (e.g., saved in `D:/Deployment`).
2. Navigate to the directory containing the key file in your terminal:

    Example:

    makefile

    CopyEdit

    ```
    D:\Deployment\Pristine Couture>
    ```

3. In the EC2 dashboard, click on **Connect**, and then select the **SSH Client** tab.

4. Copy the SSH connection command from the example provided.

5. Paste the command in your terminal (in the directory where your `.pem` file is located).

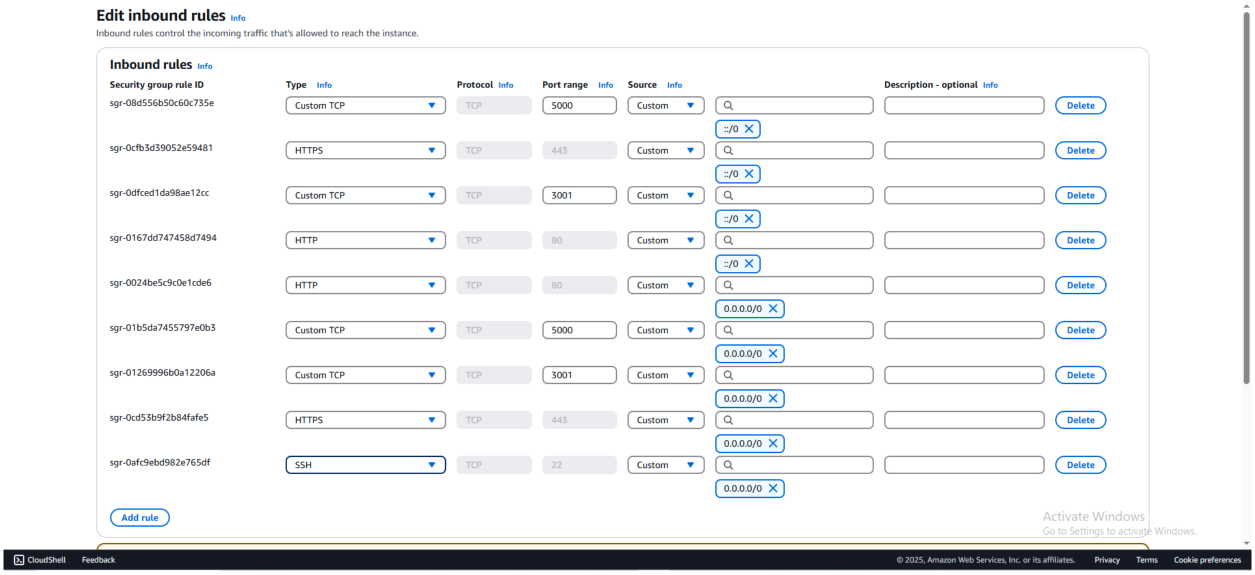
6. It may prompt a security warning. Type **yes** to continue.

7. If you get a "public key is not accessible" error, you need to change the file permissions.
    - Follow this link to resolve it: [Permission Fix Guide]
8. After setting the correct permissions, you should be successfully connected to your EC2 instance via your local terminal.

C. Setting Security Group Inbound Rules

EC2 Security Group

1. Go to the **Security Groups** section from the EC2 sidebar.
2. Find and select the **Security Group** attached to your EC2 instance.
3. Click on **Inbound Rules** > **Edit Inbound Rules**.
4. Add the required rules as per your needs (e.g., allow SSH or HTTP).



Edit inbound rules

Inbound rules control the incoming traffic that's allowed to reach the instance.

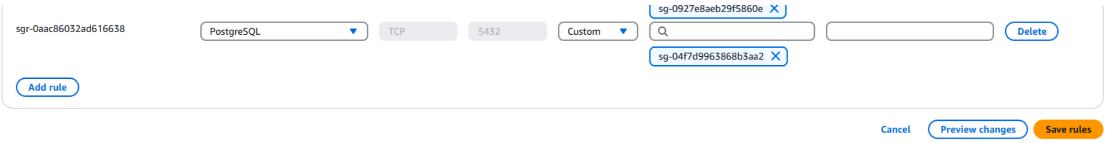
Inbound rules

Security group rule ID	Type	Protocol	Port range	Source		Description - optional
sgr-08d556b50c60c735e	Custom TCP	TCP	5000	Custom	::/0	
sgr-0cfb3d39052e59481	HTTPS	TCP	443	Custom	::/0	
sgr-0dfced1da98ae12cc	Custom TCP	TCP	3001	Custom	::/0	
sgr-0167dd747458d7494	HTTP	TCP	80	Custom	::/0	
sgr-0024be5c9c0e1cde6	HTTP	TCP	80	Custom	0.0.0.0/0	
sgr-01b5da7455797e0b3	Custom TCP	TCP	5000	Custom	0.0.0.0/0	
sgr-01269996b0a12206a	Custom TCP	TCP	3001	Custom	0.0.0.0/0	
sgr-0cd53b9f2b84fafe5	HTTPS	TCP	443	Custom	0.0.0.0/0	
sgr-0afc9ebd982e765df	SSH	TCP	22	Custom	0.0.0.0/0	

© 2025, Amazon Web Services, Inc. or its affiliates.

D. AWS RDS Security Group

1. Go to the **Security Group** linked to your RDS database.
2. Click on **Edit Inbound Rules**.
3. Since you're using PostgreSQL, add an inbound rule with the following:
    - **Type:** PostgreSQL
    - **Source:** Custom
    - **Value:** The **Security Group ID** of your EC2 instance.
4. Click **Save Rules**.



sgr-0aac86032ad616638	PostgreSQL	TCP	5432	Custom	sg-0927e8aeb29f5860e sg-04f7d9963868b3aa2	