

# Assignment 2: Cloud Computing

## Project Description

This project focuses on gaining hands-on experience with deploying reference software (RS), a React-based video game browser application. Alongside deployment, I will implement process monitoring using Linux commands, Python scripting, and crontab to generate and track system activity. Finally, using Amazon Web Services (AWS) I will create an Ubuntu virtual machine (EC2) in AWS. I will then configure Jupyter Notebook access over the internet using Elastic IP.

---

### Project Description

### Prerequisites

### Task 1: Acquire Reference Software (RS)

### Task 2: Create Build Script

### Task 3: Monitor System Process Count

### Task 4: Set up AWS EC2 and Upload RS to it

### Task 5: Install and Run Jupyter Notebook in AWS EC2 instance

---

## Prerequisites

- AWS Account
  - Basic knowledge of Linux CLI
  - Basic Python and shell scripting skills
- 

## Task 1: Acquire Reference Software (RS)

1. Go to GitHub.
2. Search for "videogames browser app" or use this link:

<https://github.com/Alais29/react-gameapp>

3. Confirm the repository is active and cloneable.
-

## Task 2: Create Build Script

1. Create a script named `build.sh` to build the RS:

```
#!/bin/bash
# Update the package list and install npm
sudo apt update && sudo apt install npm -y

# Clone this repository "react-gameapp" from GitHub
git clone https://github.com/Alais29/react-gameapp.git

# Change to the project directory
cd react-gameapp

# Install the dependencies from package.json
npm install

# Start the application
npm start
```

2. Run the script:

```
chmod +x build.sh
./build.sh
```

## Task 3: Monitor System Process Count

Calculate the number of processes every minute by using a crontab

1. Create Infinite Loop Python Script ( `infloop.py` )

```
#!/usr/bin/env python
import time
f=open("edu.txt","w")
```

```
while(True):
    f.write("edu")
    time.sleep(0.001)

f.close()
```

2. **Run the script in background:** `nohup python infloop.py &`

```
eduardo@ubuntu:~/session1-part1$ nohup python3 infloop.py &
[1] 5411
```

3. **Kill it with:**

```
ps aux | grep infloop.py
kill -9 <process_id>
```

4. **Count Processes - Bash Script ( `count_processes.sh` )**

- Below bash script was created to upload the csv file (list of data) of the number of processes.

```
#!/bin/bash
TIMESTAMP=$(date +"%Y-%m-%d %H:%M")
OUTPUT_FILE="/home/eduardo/session1-part1/process_count.csv"
PROCESS_COUNT=$(ps aux | wc -l)

echo "$TIMESTAMP, $PROCESS_COUNT" >> $OUTPUT_FILE
```

- Make it executable:

```
chmod +x count_processes.sh
```

3. **Set Up Cron Job that runs every minute:**

- Edit cron:

```
crontab -e
```

- Add this line:

```
* * * * * /home/ubuntu/count_processes.sh
```

```
## Edit this file to introduce tasks to be run by cron.
#
# Each task to run has to be defined through a single line
# indicating with different fields when the task will be run
# and what command to run for the task
#
# To define the time you can provide concrete values for
# minute (m), hour (h), day of month (dom), month (mon),
# and day of week (dow) or use '*' in these fields (for 'any').
#
# Notice that tasks will be started based on the cron's system
# daemon's notion of time and timezones.
#
# Output of the crontab jobs (including errors) is sent through
# email to the user the crontab file belongs to (unless redirected).
#
# For example, you can run a backup of all your user accounts
# at 5 a.m every week with:
# 0 5 * * 1 tar -zcf /var/backups/home.tgz /home/
#
# For more information see the manual pages of crontab(5) and cron(8)
#
# m h  dom mon dow   command
# * * * * * /home/eduardo/checkfilesize.sh
# * * * * * /home/eduardo/count_processes.sh
```

- View output:

```
cat /home/ubuntu/process_count.csv
```

or open `process_count.csv`

Open	process_count.csv	Ln 1, Col 1
1	2024-09-30 19:30, 325	
2	2024-09-30 19:31, 323	
3	2024-09-30 19:32, 331	
4	2024-09-30 19:33, 331	
5	2024-09-30 19:34, 330	
6	2024-09-30 19:35, 331	
7	2024-09-30 19:36, 328	
8	2024-09-30 19:37, 319	
9	2024-09-30 19:38, 322	
10	2024-09-30 19:39, 322	
11	2024-09-30 19:40, 319	
12	2024-09-30 19:41, 321	
13	2024-09-30 19:42, 322	
14	2024-09-30 19:43, 318	
15	2024-09-30 19:44, 318	
16	2024-09-30 19:45, 318	
17	2024-09-30 19:46, 317	
18	2024-09-30 19:47, 316	
19	2024-09-30 19:48, 322	
20	2024-09-30 19:49, 322	
21	2024-09-30 19:50, 322	
22	2024-09-30 19:51, 323	
23	2024-09-30 19:52, 323	
24	2024-09-30 19:53, 323	
25	2024-09-30 19:54, 323	
26	2024-09-30 19:55, 322	
27	2024-09-30 19:56, 321	
28	2024-09-30 19:57, 321	
29	2024-09-30 19:58, 316	
30	2024-09-30 19:59, 316	
31	2024-09-30 20:00, 316	
32	2024-09-30 20:01, 317	

## Task 4: Set up AWS EC2 and Upload RS to it

### 1. Create AWS EC2 instance

- Go to AWS Console → EC2 → Launch Instance
- AMI: **Ubuntu Server 20.04 LTS**
- Instance type: **t2.micro** (Free tier)
- Create or select an existing **key pair**.
- Configure **security group**:

Inbound rules

Outbound rules

Tags

Inbound rules (6)

Manage tags

Edit inbound rules

Q Search

<

1

>

<input type="checkbox"/>	Name	Security group rule...	IP version	Type	Protocol	Port range
<input type="checkbox"/>	-	sgr-00e1961655bca97...	IPv6	SSH	TCP	22
<input type="checkbox"/>	-	sgr-07071211da6235...	IPv4	Custom TCP	TCP	8889
<input type="checkbox"/>	-	sgr-058f2de0587098a16	IPv4	SSH	TCP	22
<input type="checkbox"/>	-	sgr-06f1ab85caf92b676	IPv6	HTTPS	TCP	443
<input type="checkbox"/>	-	sgr-0396109063564fcdf	IPv4	HTTPS	TCP	443
<input type="checkbox"/>	-	sgr-0d036a696c96671...	IPv6	Custom TCP	TCP	8889

- Allow SSH (port 22)
- Allow HTTP (port 80)
- Allow custom TCP for **port 8889** (Jupyter)
- Launch instance

## 2. Connect to Instance (Replace values)

```
ssh -i ~/key_pairs/edu.pem ubuntu@<Public-IP>
```

Example:

SSH into the instance: example:

```
ssh -i key_pairs/edu.pem ubuntu@44.194.8.13
```

aws Services Search [Alt+S] N. Virginia eb.training @ 0384-6274-8802

EC2 Dashboard EC2 Global View Events Console-to-Code [Preview](#)

▼ Instances

- Instances
- Instance Types
- Launch Templates
- Spot Requests
- Savings Plans
- Reserved Instances
- Dedicated Hosts
- Capacity Reservations

▼ Images

- AMIs
- AMI Catalog

▼ Elastic Block Store

- Volumes
- Snapshots

**Instances (1/1) Info** Last updated 41 minutes ago [Connect](#) [Instance state](#) [Actions](#) [Launch instances](#)

Find Instance by attribute or tag (case-sensitive) All states

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability zone
<input checked="" type="checkbox"/>	i-08f4a037b6811c264	Running	t2.micro	2/2 checks passed	View alarms +	us-east-1a

**i-08f4a037b6811c264**

Details Status and alarms Monitoring Security Networking Storage Tags

▼ Instance summary info

Instance ID i-08f4a037b6811c264	Public IPv4 address 44.194.8.13   <a href="#">open address</a>	Private IPv4 addresses 172.31.83.202
IPv6 address -	Instance state Running	Public IPv4 DNS ec2-44-194-8-13.compute-1.amazonaws.com   <a href="#">open address</a>
Hostname type IP name: ip-172-31-83-202.ec2.internal	Private IP DNS name (IPv4 only) ip-172-31-83-202.ec2.internal	Elastic IP addresses 44.194.8.13 [Public IP]
Answer private resource DNS name IPv4 (A)	Instance type t2.micro	

```
ubuntu@ip-172-31-83-202: ~$ cat /etc/os-release
PRETTY_NAME="Ubuntu 24.04 LTS"
NAME="Ubuntu"
VERSION_ID="24.04"
VERSION="24.04 LTS (Noble Numbat)"
VERSION_CODENAME=noble
ID=ubuntu
ID_LIKE=debian
HOME_URL="https://www.ubuntu.com/"
SUPPORT_URL="https://help.ubuntu.com/"
BUG_REPORT_URL="https://bugs.launchpad.net/ubuntu/"
PRIVACY_POLICY_URL="https://www.ubuntu.com/legal/terms-and-policies/privacy-policy"
UBUNTU_CODENAME=noble
LOGO=ubuntu-logo
ubuntu@ip-172-31-83-202:~$
```

### 3. Upload RS to EC2

```
scp -i /path/to/your-key.pem /local/path/to/RS-file ec2-user@your-instance-ip:/remote/path/
```

# Task 5: Install and Run Jupyter Notebook in AWS EC2 instance

Since we already have python within the server, we just need to install Jupyter in the EC2 instance

## 1. Install Jupyter in AWS instance

```
sudo apt update
sudo apt install jupyter-core -y
sudo apt install jupyter -y
```

## 2. Run Jupyter server:

```
jupyter notebook --port 8889 --allow-root --no-browser --ip=0.0.0.0
```

- Then Obtain this part:

```
ubuntu@ip-172-31-83-202:~$ jupyter notebook --port 8889 --allow-root --no-browser --ip=0.0.0.0
[I 08:04:36.558 NotebookApp] Writing notebook server cookie secret to /home/ubuntu/.local/share/jupyter/runtime/notebook_cookie_secret
[I 08:04:36.724 NotebookApp] Serving notebooks from local directory: /home/ubuntu
[I 08:04:36.724 NotebookApp] Jupyter Notebook 6.4.12 is running at:
[I 08:04:36.724 NotebookApp] http://ip-172-31-83-202:8889/?token=534798d28ea5025e1097812a843478522f983c4b4ca58183
[I 08:04:36.724 NotebookApp] or http://127.0.0.1:8889/?token=534798d28ea5025e1097812a843478522f983c4b4ca58183
[I 08:04:36.724 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
[C 08:04:36.726 NotebookApp]

To access the notebook, open this file in a browser:
file:///home/ubuntu/.local/share/jupyter/runtime/nbserver-5722-open.html
Or copy and paste one of these URLs:
http://ip-172-31-83-202:8889/?token=534798d28ea5025e1097812a843478522f983c4b4ca58183
or http://127.0.0.1:8889/?token=534798d28ea5025e1097812a843478522f983c4b4ca58183
```

- Also this part:

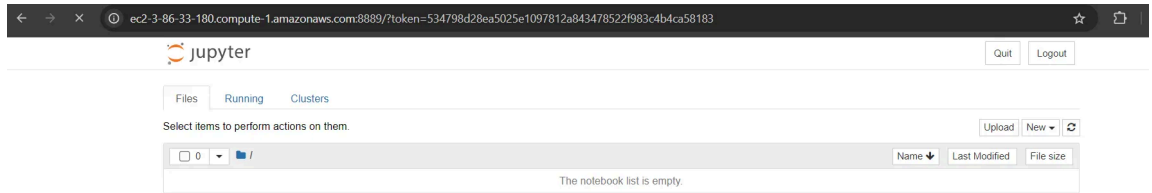
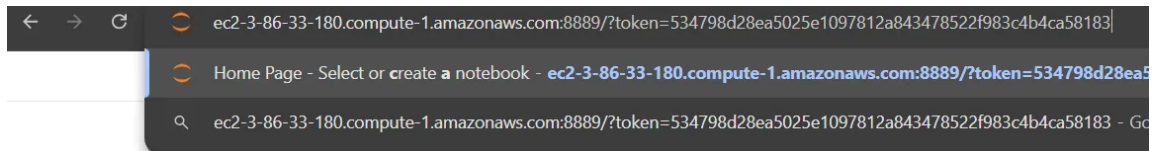
The screenshot displays the 'Instance summary' page for the EC2 instance i-08f4a037b6811c264. The instance is currently in a 'Running' state. The summary is organized into three columns of key-value pairs. The first column lists the Instance ID, IPv6 address (none), Hostname type (ip-172-31-83-202.ec2.internal), Answer private resource DNS name (IPv4 (A)), and Auto-assigned IP address (3.86.33.180 [Public IP]). The second column shows the Public IPv4 address (3.86.33.180), Instance state (Running), Private IP DNS name (ip-172-31-83-202.ec2.internal), Instance type (t2.micro), and VPC ID (vpc-0827a287ab0e9d35c). The third column includes Private IPv4 addresses (172.31.83.202), Public IPv4 DNS (ec2-3-86-33-180.compute-1.amazonaws.com), Elastic IP addresses (none), and an AWS Compute Optimizer finding (Opt-in to AWS Compute Optimizer for recommendations).

Instance ID	Public IPv4 address	Private IPv4 addresses
i-08f4a037b6811c264	3.86.33.180   open address	172.31.83.202
IPv6 address	Instance state	Public IPv4 DNS
-	Running	ec2-3-86-33-180.compute-1.amazonaws.com   open address
Hostname type	Private IP DNS name (IPv4 only)	Elastic IP addresses
IP name: ip-172-31-83-202.ec2.internal	ip-172-31-83-202.ec2.internal	-
Answer private resource DNS name	Instance type	AWS Compute Optimizer finding
IPv4 (A)	t2.micro	Opt-in to AWS Compute Optimizer for recommendations.
Auto-assigned IP address	VPC ID	
3.86.33.180 [Public IP]	vpc-0827a287ab0e9d35c	

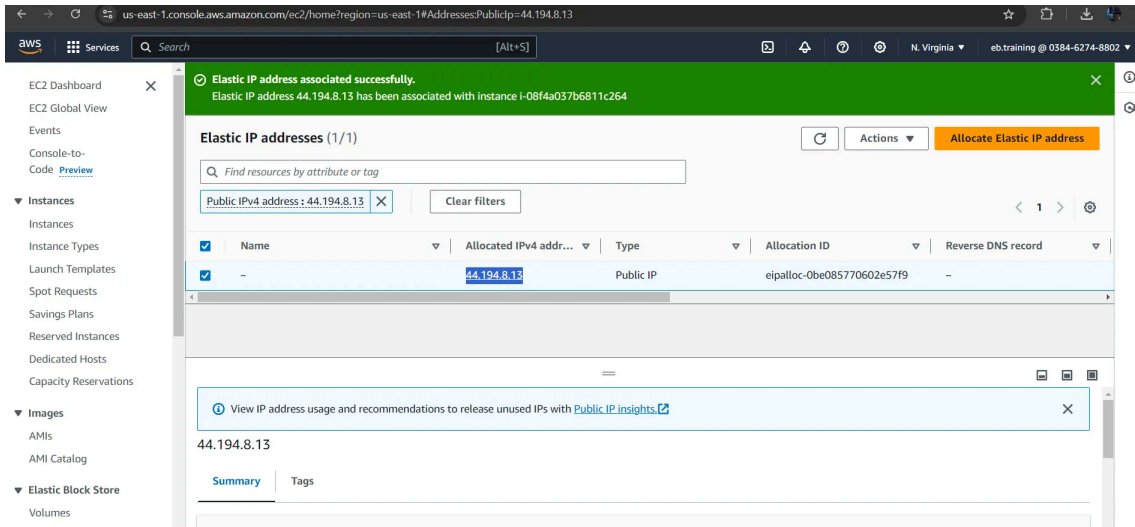


- then add both of this information into the web browser to confirm access:

<Public IPv4 DNS>:8889/?token=<token>



- Create and Associate an Elastic IP
  - In AWS → EC2 → **Elastic IPs**
  - Allocate new address
  - Click **Associate Elastic IP**
    - Select your running instance
  - Obtain the Elastic IP



- Update URL: `<Elastic IP>:8889/?token=<token>`

