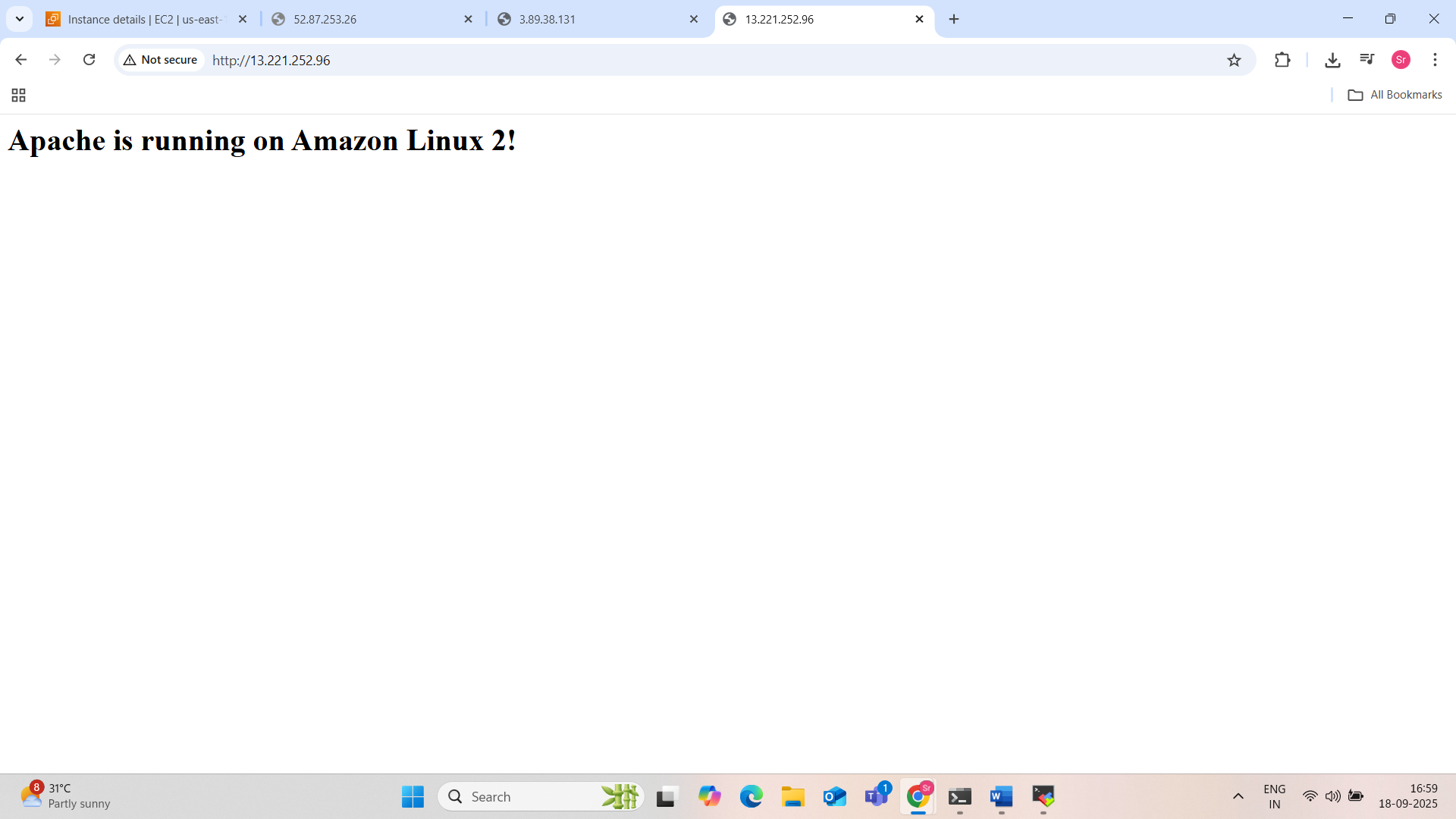
 Task on EC2

1. Launch one EC2 using Amazon Linux 2 image and add a script in user data to install Apache.



**Step 1: Log in to AWS**

* Go to the AWS Console
* Sign in with your AWS credentials.

**Step 2: Navigate to EC2**

* In the AWS Console, click **Services** > **EC2**
* Click **Instances** from the sidebar
* Click **Launch Instances**

**Step 3: Configure Basic Settings**

1. **Name**:
   * Give your instance a name like Apache-Server-AmazonLinux2
2. **Application and OS Image (AMI)**:
   * Select: **Amazon Linux 2 AMI (HVM), SSD Volume Type**
3. **Instance Type**:
   * Choose t2.micro (Free Tier eligible)
4. **Key Pair (Login)**:
   * Choose an existing key pair or create a new one
   * This is required to SSH into the instance

**Step 4: Configure Network Settings**

* **Allow SSH (port 22)** – to connect to the instance
* **Allow HTTP (port 80)** – to access Apache via browser

Create or modify the **security group** to allow:

Type: HTTP

Protocol: TCP

Port: 80

Source: 0.0.0.0/0 (Anywhere)

Make sure SSH (port 22) is also enabled for your IP or anywhere (for testing).

**Step 5: Add User Data Script (Very Important)**

Scroll down to **Advanced details**, find the **User data** field, and paste the following script:

#!/bin/bash

# Update the OS packages

yum update -y

# Install Apache web server

yum install -y httpd

# Start Apache

systemctl start httpd

# Enable Apache to start on system boot

systemctl enable httpd

# Create a simple web page

echo "<h1>Apache is running on Amazon Linux 2!</h1>" > /var/www/html/index.html

This script runs **once on first boot** and installs Apache automatically.

**Step 6: Configure Storage (Optional)**

* Default storage (8 GB) is usually fine.
* You can increase it if your app needs more space.

**Step 7: Tags (Optional)**

* You can add a tag like:
  + **Key**: Name
  + **Value**: MyApacheServer

**Step 8: Review and Launch**

* Review all settings
* Click **Launch Instance**
* Wait a few seconds until the instance is running

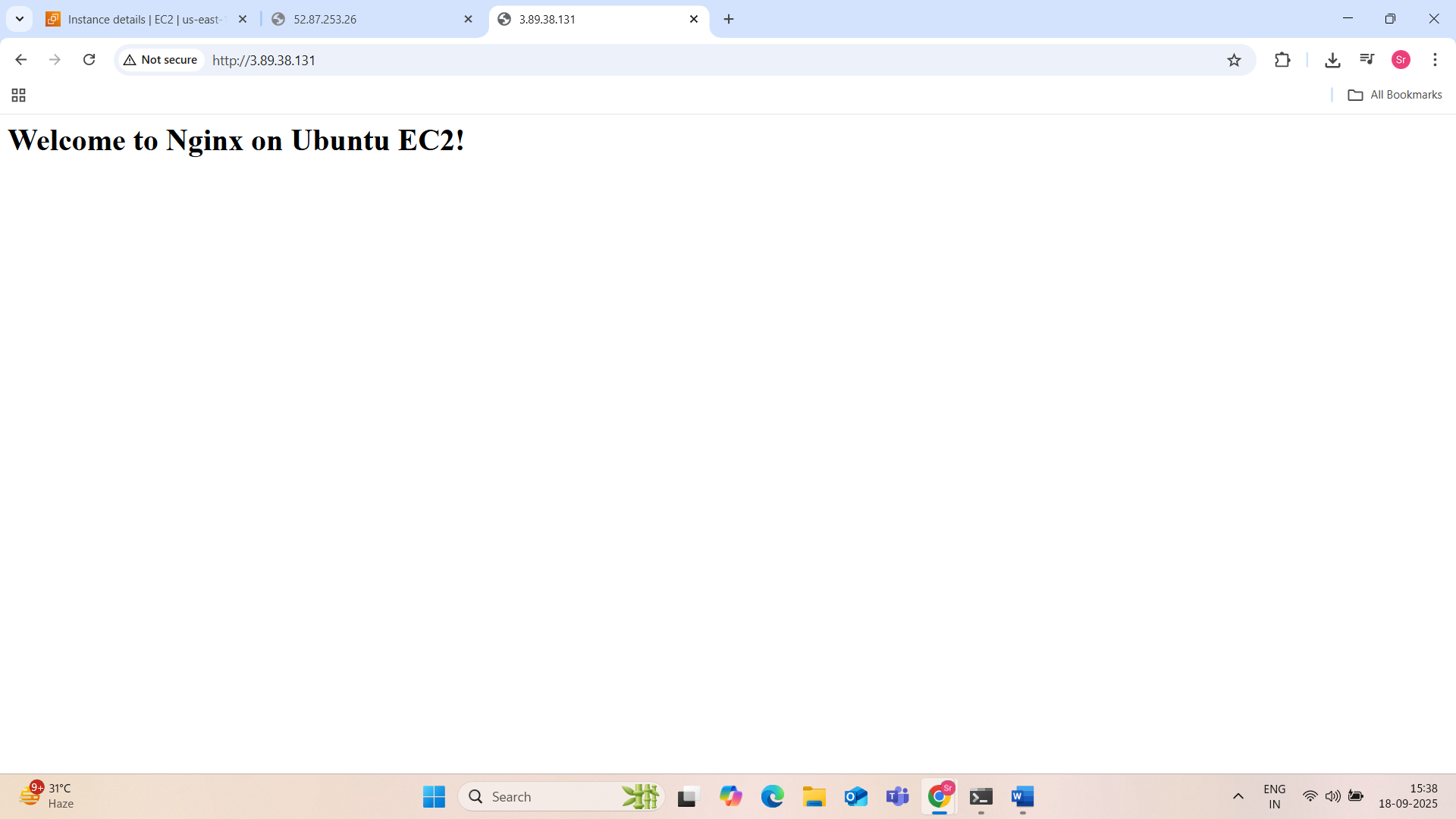
**Step 9: Test Apache Web Server**

1. Go to **EC2 > Instances**
2. Select your instance
3. Copy the **Public IPv4 address**
4. Open a browser and go to:

http://<your-ec2-public-ip>You should see:

Apache is running on Amazon Linux 2!

2.Launch one EC2 using Ubuntu image and add a script in user data to install Nginx.



**Step 1: Go to AWS EC2 Dashboard**

* Log in to the AWS Management Console.
* Navigate to **EC2 Dashboard**:  
  Services > Compute > EC2

**Step 2: Launch a New Instance**

1. Click **Launch Instance**.
2. **Name your instance** (e.g., Ubuntu-Nginx-Server).

**Step 3: Choose Ubuntu AMI**

* In the “Application and OS Images (Amazon Machine Image)” section:
  + Click **Browse more AMIs**.
  + Search for:  
    ubuntu 20.04 or ubuntu 22.04
  + Choose:  
    **Ubuntu Server 20.04 LTS (HVM), SSD Volume Type - ami-xxxxxxxxxxxxxxxxx**
  + Make sure it’s an **official image** published by Canonical.

**Step 4: Choose Instance Type**

* Choose **t2.micro** (Free Tier eligible).

**Step 5: Configure Key Pair**

* Select an existing **key pair** or create a new one.
* You’ll need this to SSH into your instance later.

**Step 6: Network Settings (Security Group)**

* **Allow SSH (port 22)** – for admin access.
* **Allow HTTP (port 80)** – so you can access Nginx from browser.

You can either:

* Create a new security group with these rules.
* Or select an existing one that already allows port 80 and 22.

**Step 7: Add User Data Script**

Scroll down to **Advanced Details**, and in the **User data** section, paste the following script:

#!/bin/bash

# Update package lists

apt-get update -y

# Install Nginx

apt-get install nginx -y

# Enable Nginx to start on boot

systemctl enable nginx

# Start Nginx service now

systemctl start nginx

# Optional: Create a simple homepage

echo "<html><h1>Welcome to Nginx on Ubuntu EC2!</h1></html>" > /var/www/html/index.html

**Explanation of the Script:**

| **Command** | **Purpose** |
| --- | --- |
| #!/bin/bash | Tells the OS to run the script using bash |
| apt-get update -y | Updates the list of available packages |
| apt-get install nginx -y | Installs the Nginx web server |
| systemctl enable nginx | Ensures Nginx runs on boot |
| systemctl start nginx | Starts the Nginx server now |
| echo ... | Creates a basic HTML homepage |

**Step 8: Launch the Instance**

* Click **Launch Instance**.
* Wait for the instance to go into the **Running** state.

**Step 9: Test Nginx**

1. Go to **Instances** in the EC2 Dashboard.
2. Find your instance and copy its **Public IPv4 address**.
3. Open your browser and go to:

http://<your-public-ip>

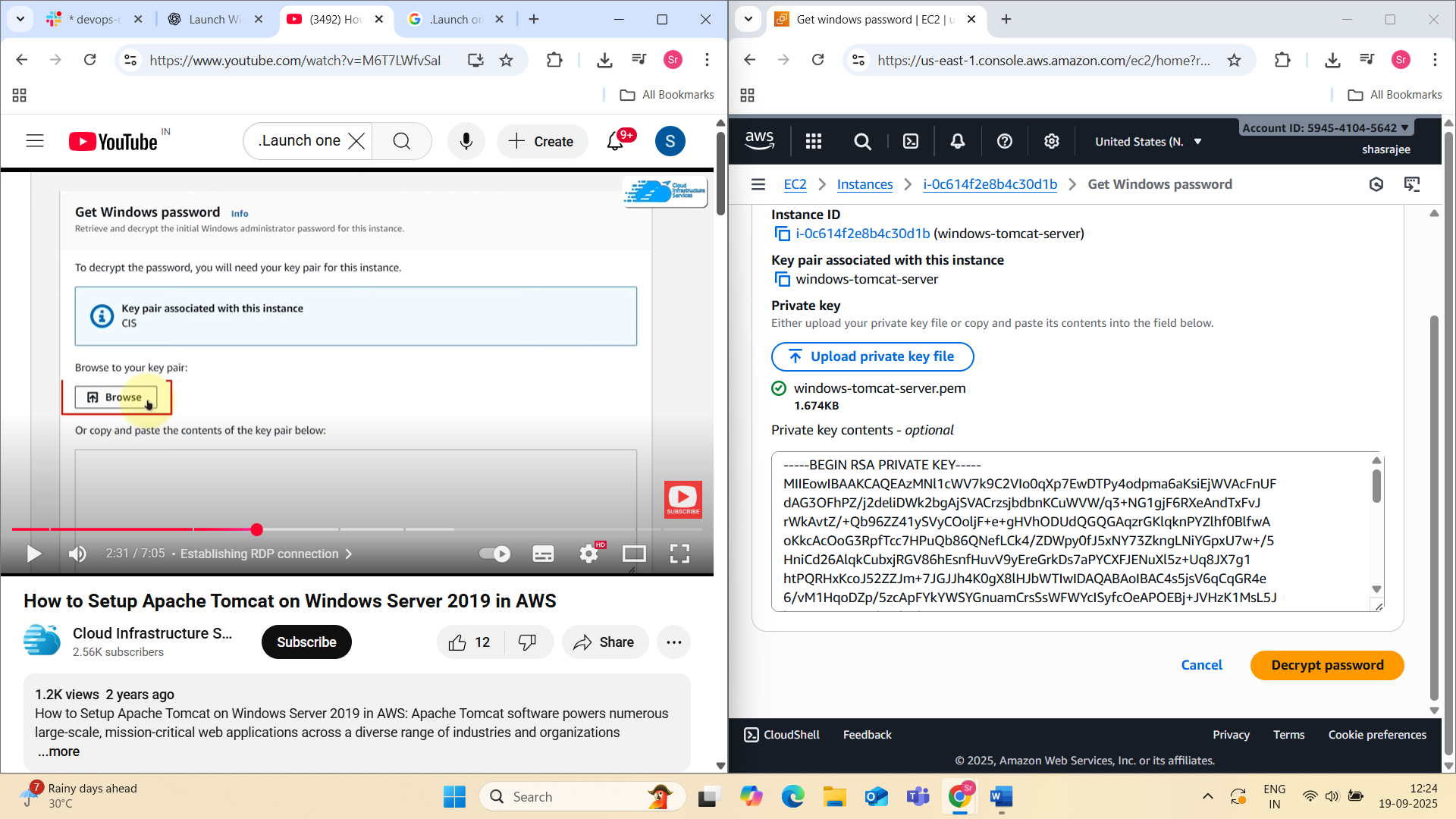
You should see:

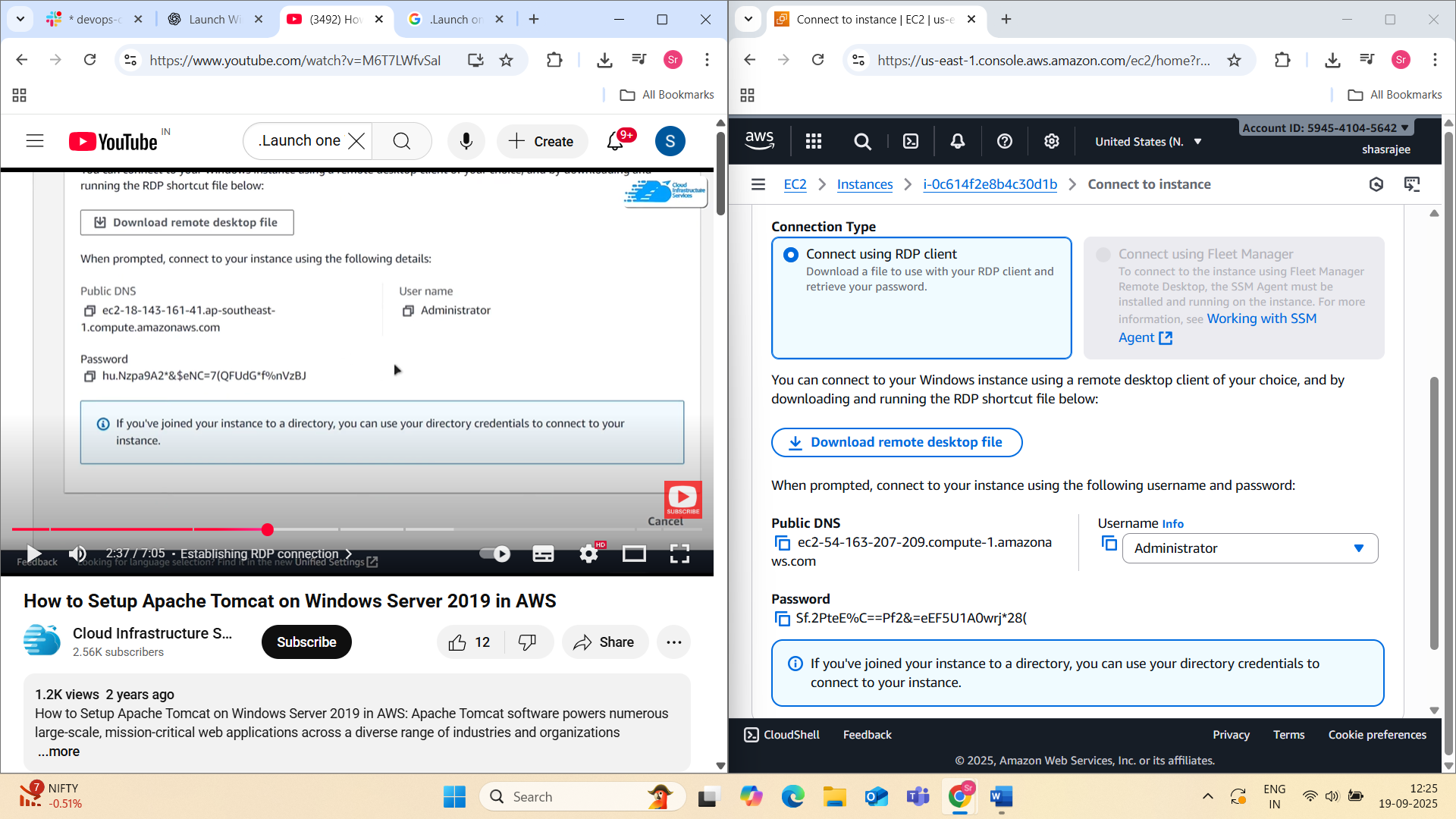
Welcome to Nginx on Ubuntu EC2!

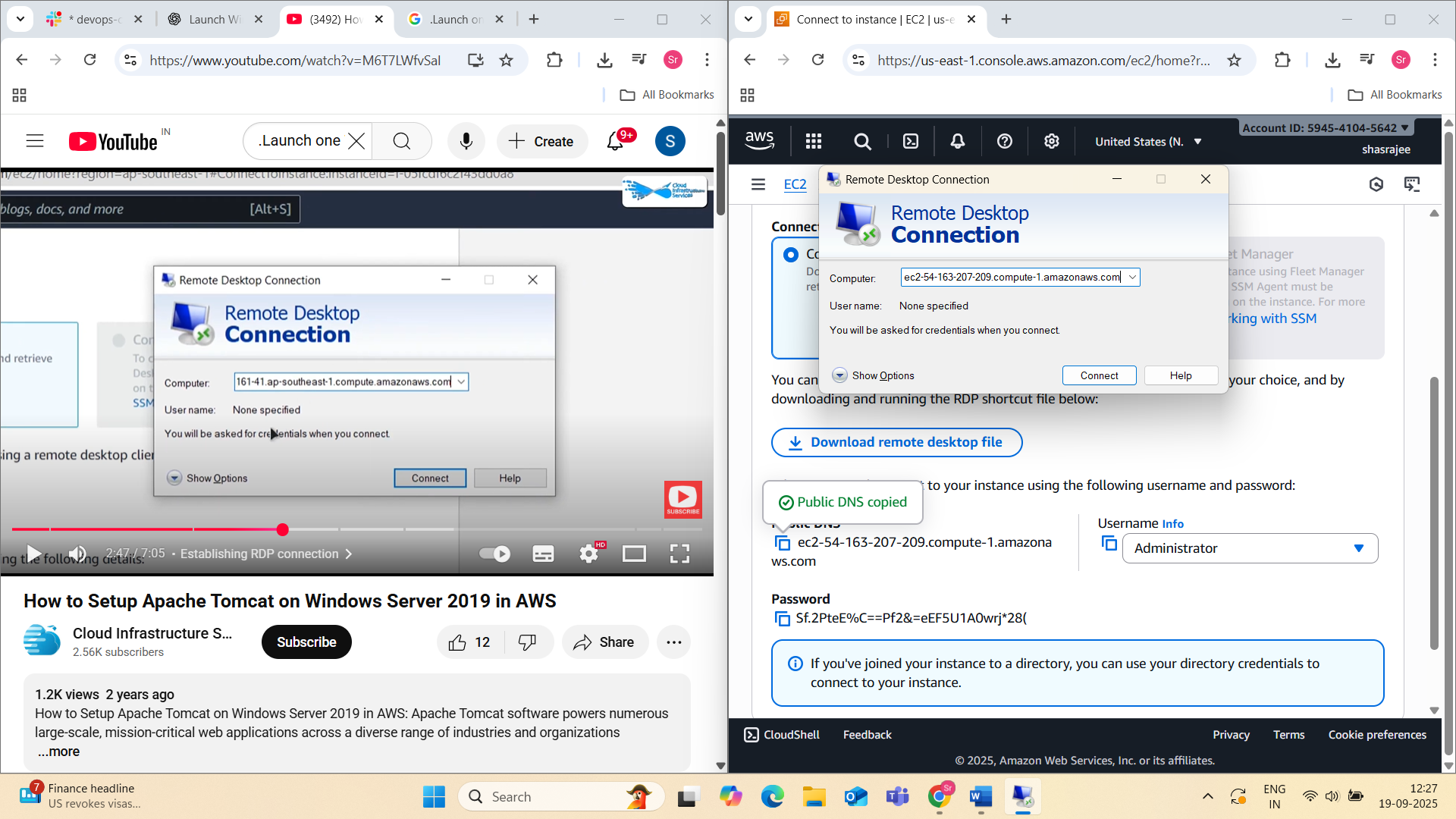
3.Launch one Windows server and install Tomcat on Windows.

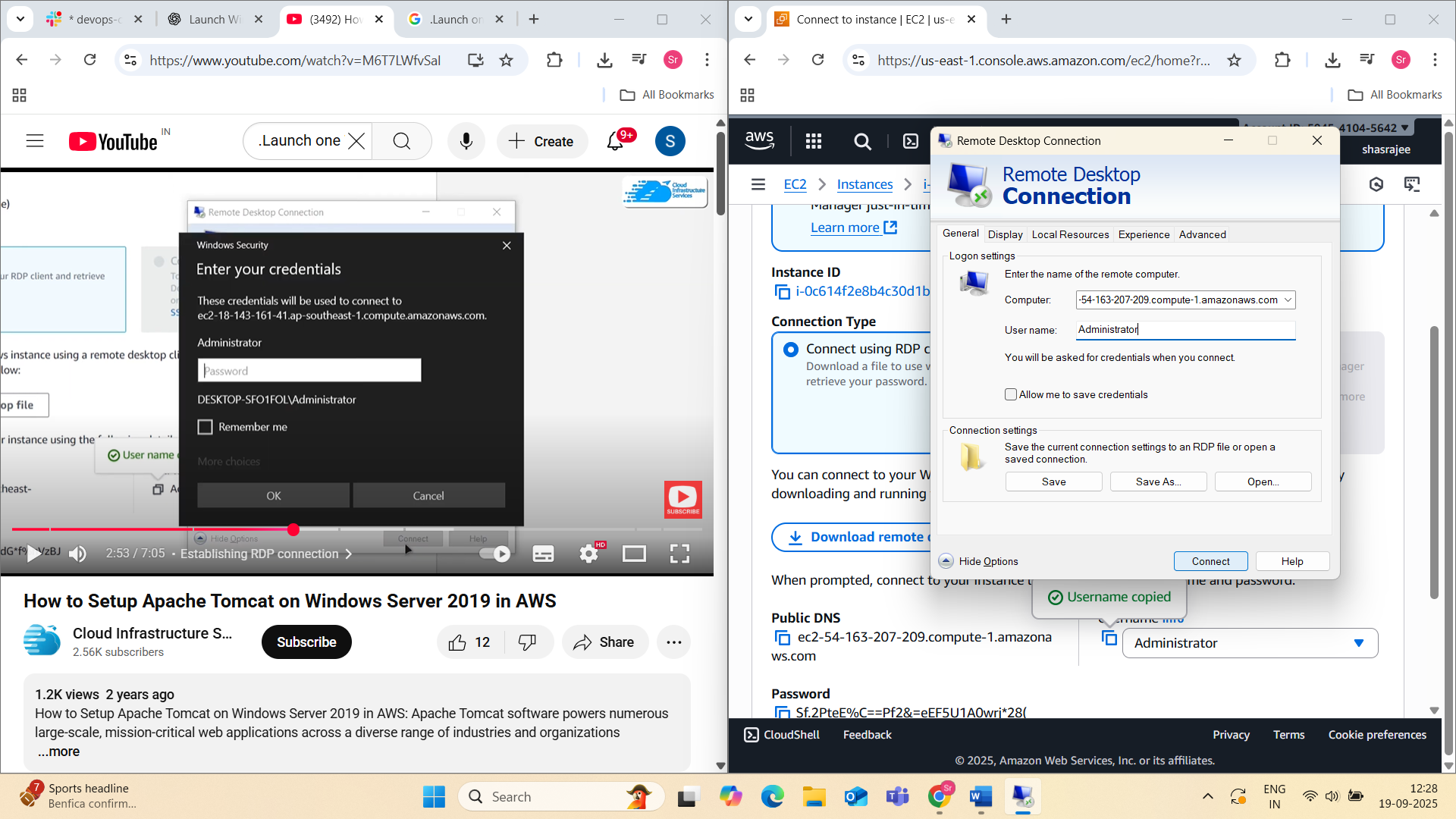
Part 1: Launch a Windows Server (on AWS EC2)

Part 2: Install Apache Tomcat on Windows Server









Part 1: Launch a Windows Server (on AWS EC2)

**Step 1: Launch EC2 Windows Server**

1. Go to **EC2 Dashboard**
2. Click **“Launch Instance”**
3. **Name the instance:** e.g., Windows-Tomcat-Server
4. **Choose AMI:** Select **Microsoft Windows Server 2019 Base** (or newer)
5. **Instance type:** Choose t2.medium (minimum 2GB RAM recommended for Tomcat)
6. **Key pair:** Select or create a new key pair to access the server (RDP)
7. **Security group settings:**
   * Allow **RDP (port 3389)** from your IP (for remote desktop)
   * Allow **HTTP (port 80)** and **Tomcat (port 8080)** from anywhere or your IP
8. **Launch instance**

**Step 3: Connect to the Windows Server**

1. Wait until the instance state is **"Running"**
2. Click the instance → Click **"Connect"** → Choose **RDP client**
3. Download the **Remote Desktop File**
4. Get the **Administrator password** (requires key pair)
5. Open .rdp file, log in with username: Administrator and the decrypted password

**Part 2: Install Apache Tomcat on Windows Server**

**Step 1: Install Java (JDK)**

Tomcat requires Java to run.

1. Open a browser on the Windows Server
2. Go to: <https://www.oracle.com/java/technologies/javase-jdk11-downloads.html>
3. Download the **Windows x64 Installer**
4. Run the installer and follow the prompts
5. After install, set environment variables:
   * Search “Environment Variables” in Windows search
   * Under **System variables**, click **New**:
     + JAVA\_HOME → C:\Program Files\Java\jdk-<version>
   * Edit Path → Add: %JAVA\_HOME%\bin
6. Verify Java install:
   * Open Command Prompt
   * Run: java -version

**Step 2: Download and Install Apache Tomcat**

1. Go to: https://tomcat.apache.org
2. Click on **Tomcat 10** or **Tomcat 9** (LTS)
3. Download the **32-bit/64-bit Windows Service Installer** (.exe)
4. Run the installer:
   * Choose **components** (select “Service” and “Host Manager”)
   * Set **Tomcat admin username/password** (used for the web UI)
   * Configure ports (default is 8080)
   * Select the **Java path** (from JAVA\_HOME)
   * Choose the installation folder (e.g., C:\Tomcat)

**Step 3: Start Tomcat Service**

1. Tomcat installs as a **Windows service**
2. Go to **Services** (search in Start menu)
3. Find **Apache Tomcat x.x**
4. Start the service
5. Open a browser on the server and go to:
6. http://localhost:8080

You should see the **Tomcat Welcome Page**

**Step 4: Allow Port 8080 in Windows Firewall**

If you want to access Tomcat from your local machine (not just localhost):

1. Open **Windows Defender Firewall**
2. Go to **Advanced Settings** → Inbound Rules
3. Click **New Rule**:
   * Rule Type: Port
   * TCP → Specific port: 8080
   * Allow the connection → Apply to all profiles
   * Name: Allow Tomcat Port 8080

**Step 5: Access Tomcat from Your Local Machine**

* Use your instance's **public IP** or public DNS:
* http://<EC2\_PUBLIC\_IP>:8080

If all firewall rules and security groups are set correctly, the Tomcat homepage will load.

4Take a snapshot of the instance created in Task 1.

**What is a Snapshot?**

A **snapshot** is a backup of an **EBS volume**. You can use it to:

* Restore the instance if something goes wrong.
* Create a new AMI (Amazon Machine Image).
* Launch a new EC2 instance with the same data/config.

**Create a Snapshot of an EC2 Instance**

**Step 1: Identify the EBS Volume**

Each EC2 instance has one or more **EBS volumes** attached. You need the **volume ID**.

**From AWS Console:**

1. Go to **EC2 Dashboard**.
2. Click on **Instances** and select your instance.
3. Scroll down to **Storage** tab.
4. Click the **Volume ID** (e.g., vol-0abcd1234ef567890) – it takes you to the EBS volume.

**Step 2: Create Snapshot (Via Consol**

1. In the **EBS Volume** screen, select the volume.
2. Click **Actions** → **Create Snapshot**.
3. Fill in:
   * **Name**: (e.g., MyApp-Backup)
   * **Description**: Something meaningful like “Snapshot before update”
4. Click **Create Snapshot**.

📌 The snapshot will now be created asynchronously. It may take a few minutes depending on volume size.

**Step 3: Monitor Snapshot**

1. Go to **Snapshots** under **Elastic Block Store** in EC2 Dashboard.
2. Find your snapshot by **Snapshot ID** or **Name**.
3. Wait until the **Status** becomes completed.

**Step 4: (Optional) Use Snapshot to Restore or Create AMI**

* You can:
  + **Create Volume** from the snapshot.
  + **Register AMI** from the snapshot to launch a new EC2 instance.

**CLI Method: Create Snapshot Using AWS CLI**

First, get the volume ID of your instance:

aws ec2 describe-instances \

--instance-id i-0abc123456789def0 \

--query "Reservations[].Instances[].BlockDeviceMappings[].Ebs.VolumeId" \

--output text

Then create the snapshot:

aws ec2 create-snapshot \

--volume-id vol-0abc123456789def0 \

--description "Snapshot before upgrade"

You’ll get a SnapshotId in the response.

5.Assign passwordless authentication for the EC2 created in Task 2.

6.Launch any EC2 using the spot purchasing option

Start Launching an Instance:

### Configure Instance Details (Enable Spot Instance)

This step is important to specify the Spot Instance option.

* Under **Purchase option**, check the box **Request Spot Instances**
* Spot options will appear:
  + **Maximum price**: Leave blank to use the current spot price or enter a max price you’re willing to pay per hour (optional)
  + **Request type**:
    - **One-time**: The instance will run until terminated or interrupted.
    - **Persistent**: AWS will try to maintain the spot request until you cancel it.
  + **Interrupt behavior**:
    - Choose what happens if the instance is interrupted (default is stop, or you can terminate)
* Fill other instance details as needed (like Network, IAM role, shutdown behavior)

### Configure Security Group

* You can create a new security group or use an existing one
* Add rules to allow necessary traffic, e.g.:
  + SSH (port 22) for Linux or RDP (port 3389) for Windows from your IP
  + HTTP (port 80) if running a web server
* Click **Review and Launch**

### Choose or Create Key Pair

* Select an existing key pair or create a new key pair for SSH/RDP access
* Download the key pair .pem file if creating new (keep it safe)
* Check the acknowledgment box
* Click **Launch Instances**

**Spot Request and Instance Launch**

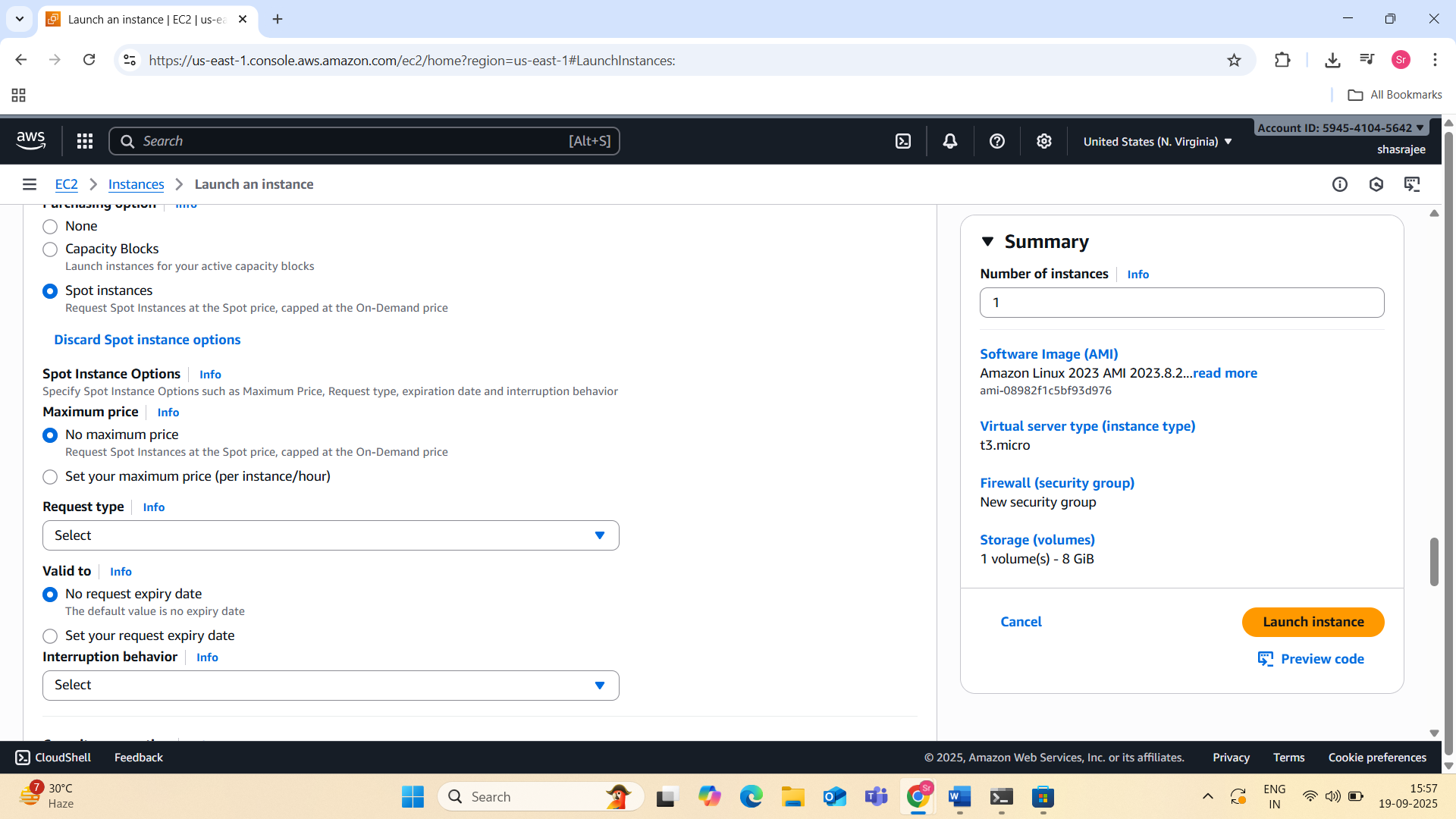
* After launching, AWS creates a **Spot Instance Request**
* Once capacity is available and your max price matches or exceeds the spot price, your instance will launch
* You can see the instance and its status in the **Instances** page

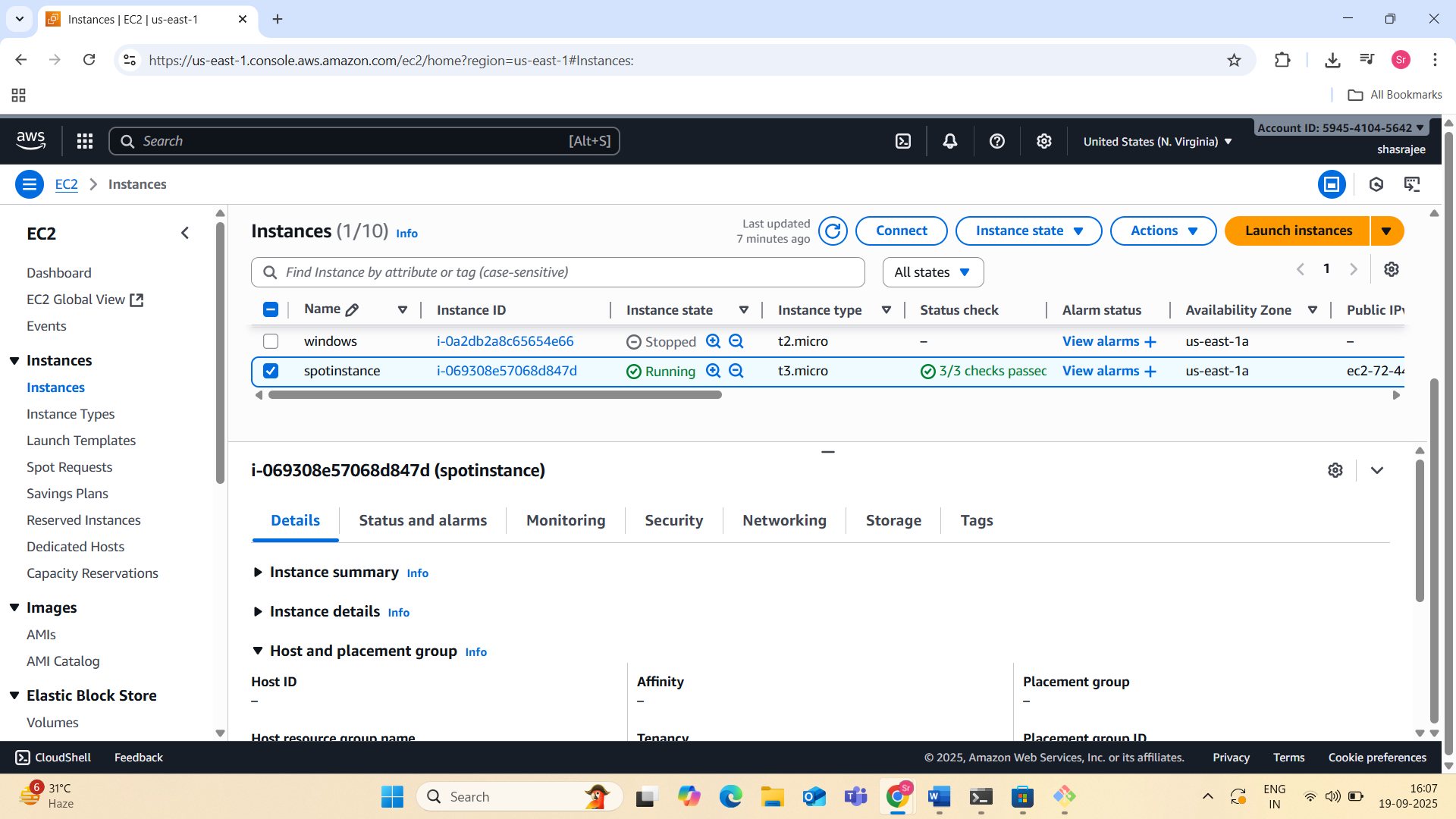
**Connect to Your Spot Instance**

* Once the instance state is **running**, select it and click **Connect**
* Use the instructions based on your OS:
  + SSH for Linux instances
  + RDP for Windows instances

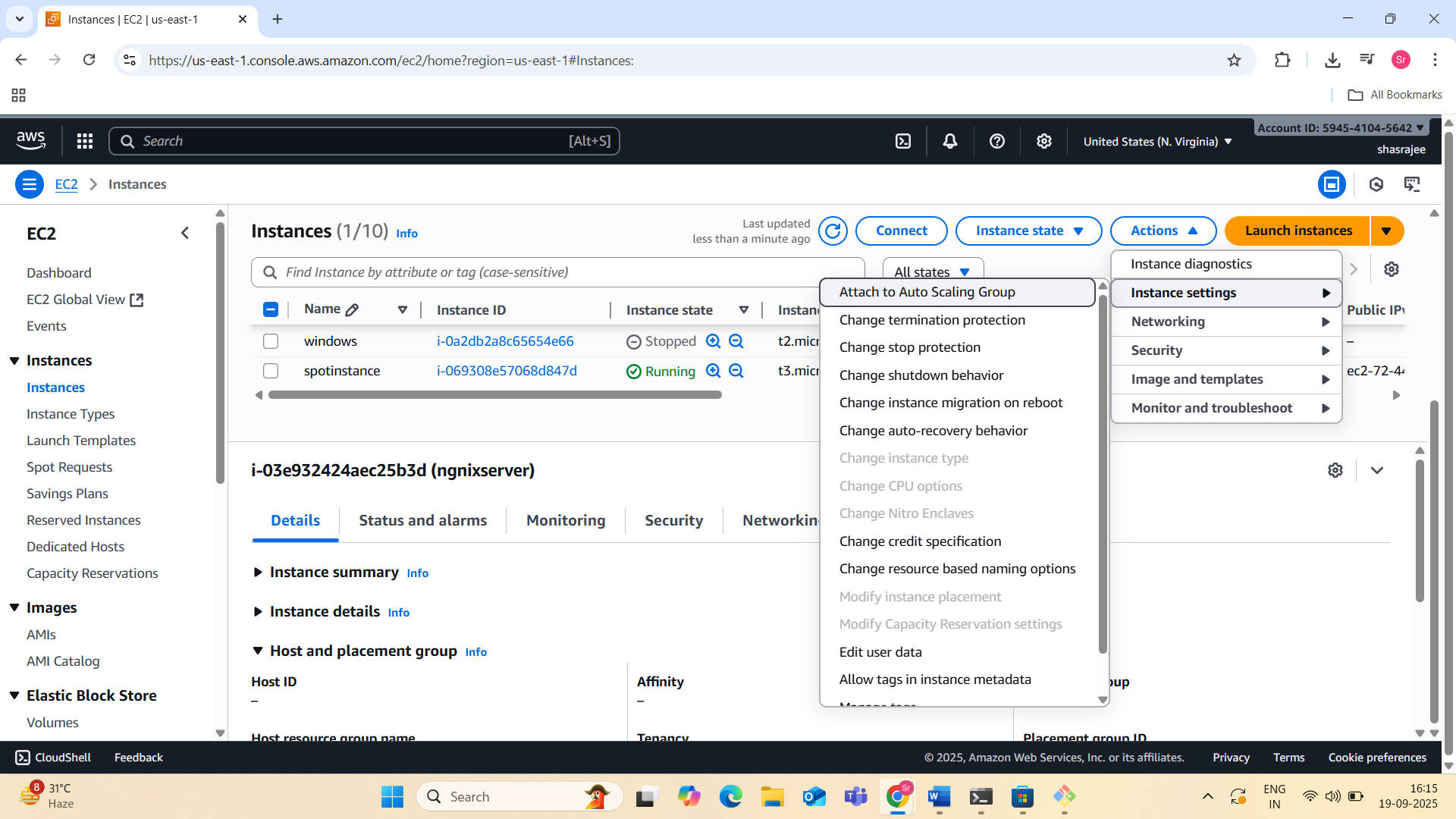
**Monitor Spot Instance Lifecycle**

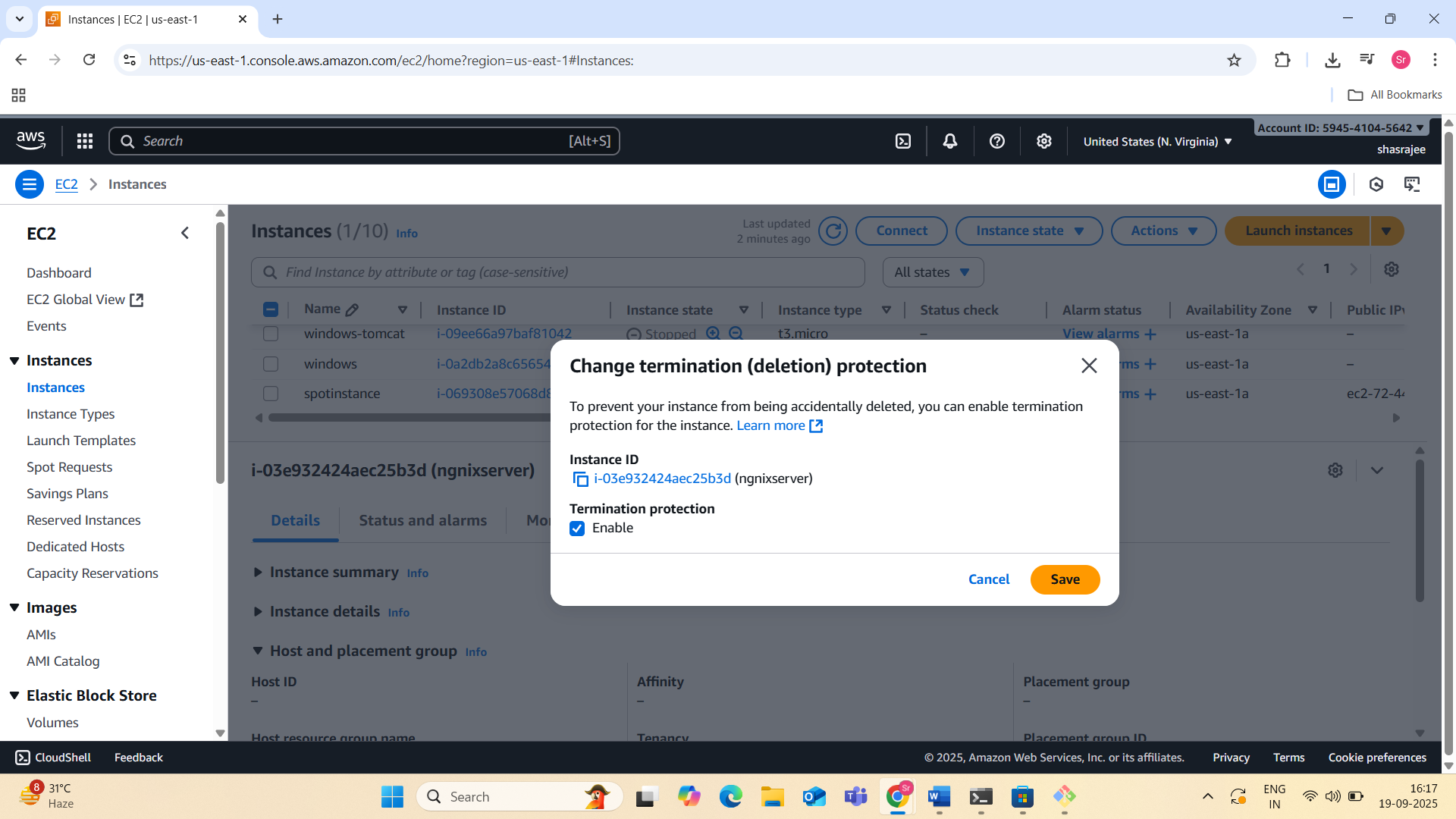
* Spot instances can be **interrupted** with a 2-minute warning if capacity is reclaimed by AWS
* AWS will send an interruption notice via instance metadata or CloudWatch Events (optional setup)
* You can prepare your application to handle interruptions gracefully (saving state, failover)





7.Enable termination policy on the EC2 created in Task 2.





8.Launch one EC2 using AWS CLI.



