AWS Cloud Infrastructure Deployment and Automation

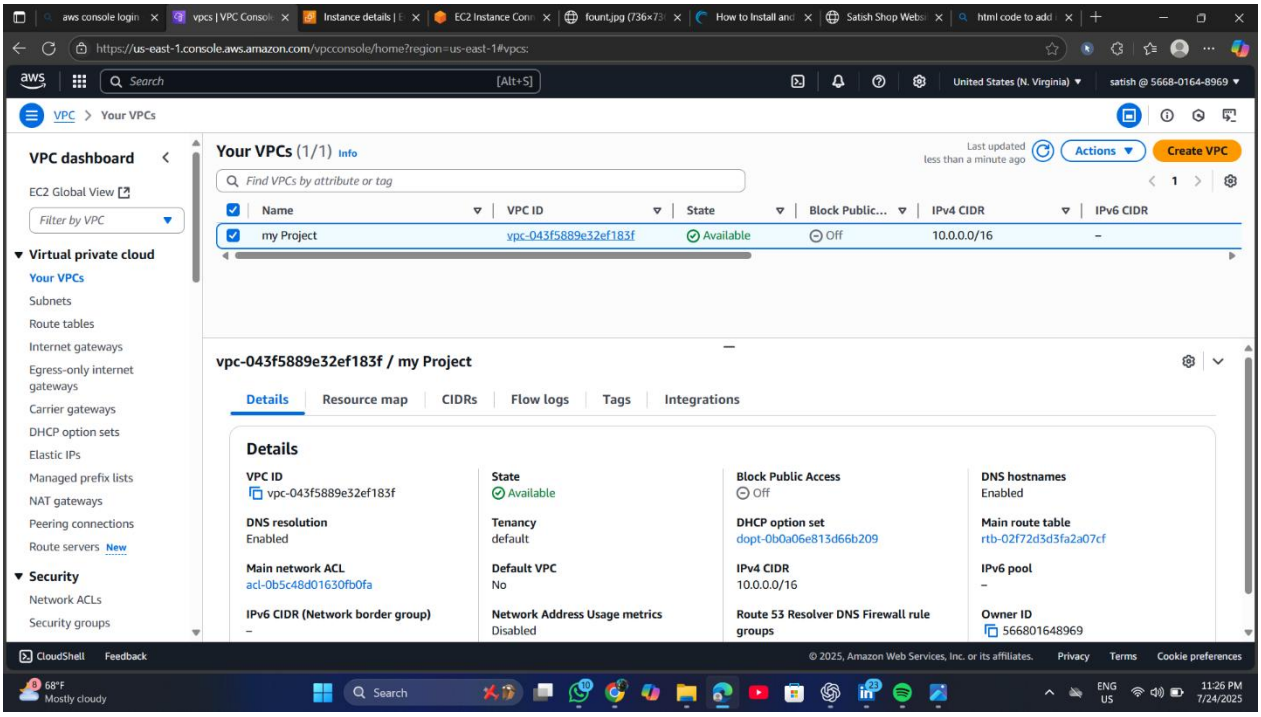
1. Create a VPC

What:
A Virtual Private Cloud (VPC) is a logically isolated section of AWS where you launch AWS resources.

How:

- In AWS Console, go to **VPC**, click **Create VPC**.
- Choose default or custom settings (CIDR block, subnets, etc).

Why:
This isolates your application's network for security and control.



2. Create S3 and Upload File

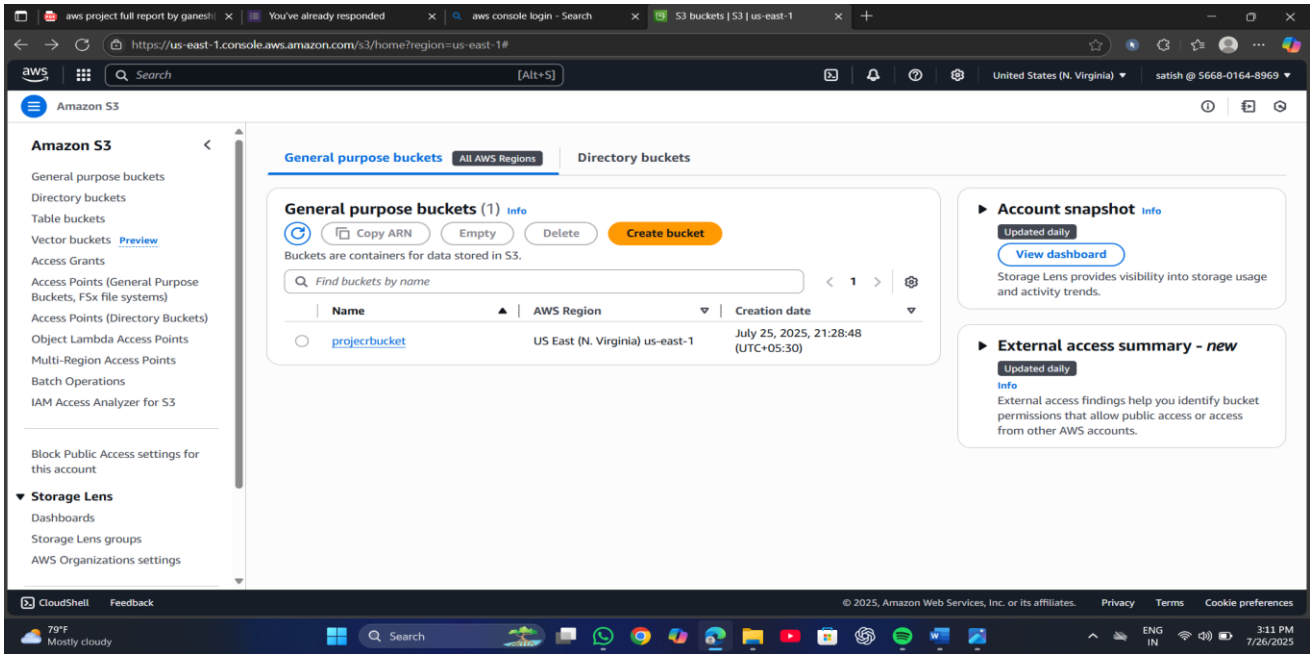
What:
S3 is Amazon's object storage.

How:

- In AWS Console, go to **S3**, **Create Bucket**.
- Upload files (e.g., static assets or media for your app).

Why:
S3 is used for storing files, static data, backups, etc.

AWS Cloud Infrastructure Deployment and Automation



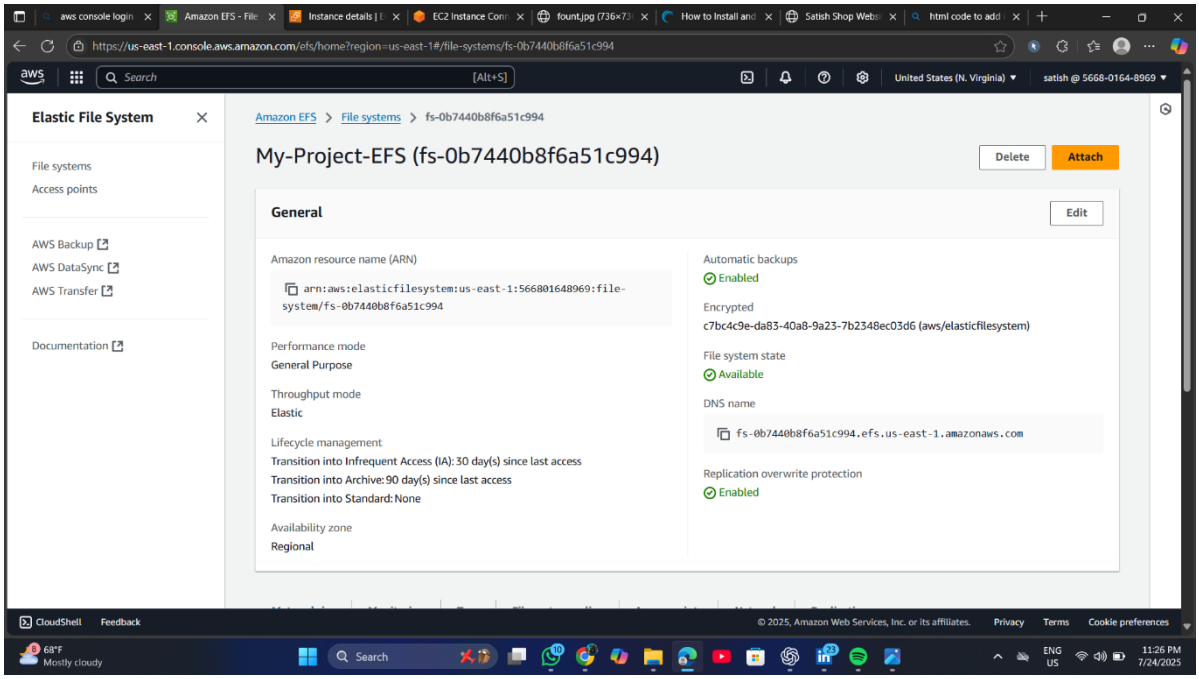
3. Create EFS

What:
EFS (Elastic File System) is scalable file storage for use with EC2.

How:

- Go to EFS, **Create file system**.
- Set up mount targets, security groups.

Why:
EFS can be mounted on multiple EC2s at once, useful for sharing files or app data.



AWS Cloud Infrastructure Deployment and Automation

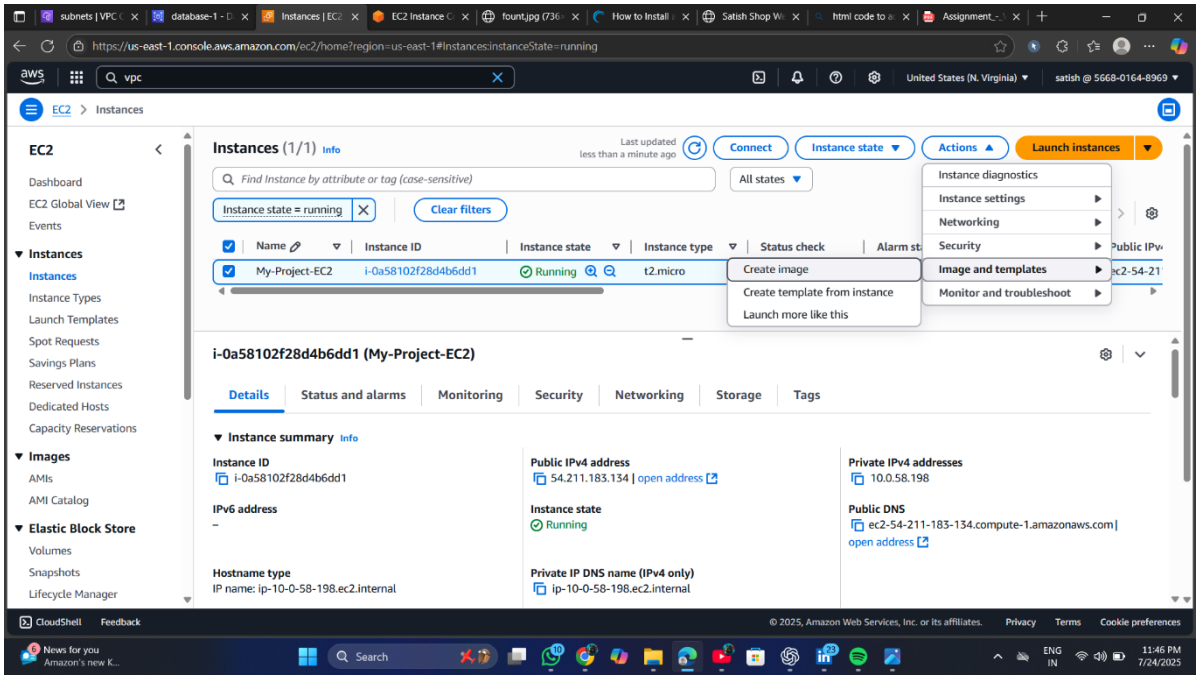
4. Launch EC2, Install Nginx, Configure index.html

What:
EC2 are the virtual servers. Nginx is a web server.

How:

- Launch an EC2 instance (Amazon Linux/Ubuntu).
- SSH into instance.
- Install nginx: #bash sudo apt update sudo apt upgrade sudo apt install nginx sudo systemctl status nginx
- Create and edit /var/www/html/index.html.

Why:
This is your web server that will host your app site.



5. Login to Your Ubuntu Instance

What:
SSH into your EC2 machine to perform administrative tasks.

How: bash ssh -i your-key.pem

ubuntu@<public-ip>

Use your PEM key and the public IP from the AWS console.

6. Update & Upgrade OS (on EC2)

What:

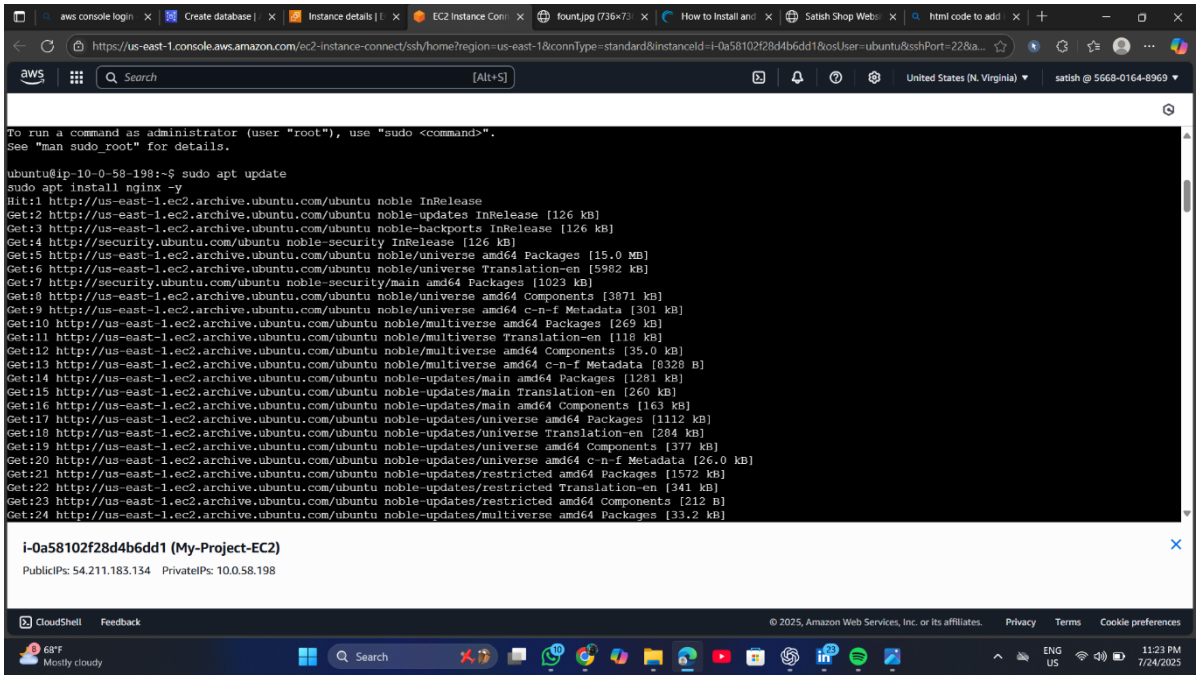
AWS Cloud Infrastructure Deployment and Automation

Keeps your system secure and up to date.

How: bash sudo

apt update sudo

apt upgrade



7. Install and Check Nginx

How: #bash

sudo apt install nginx sudo systemctl status

nginx This installs and checks if nginx is

running. sudo unlink /etc/nginx/sites-

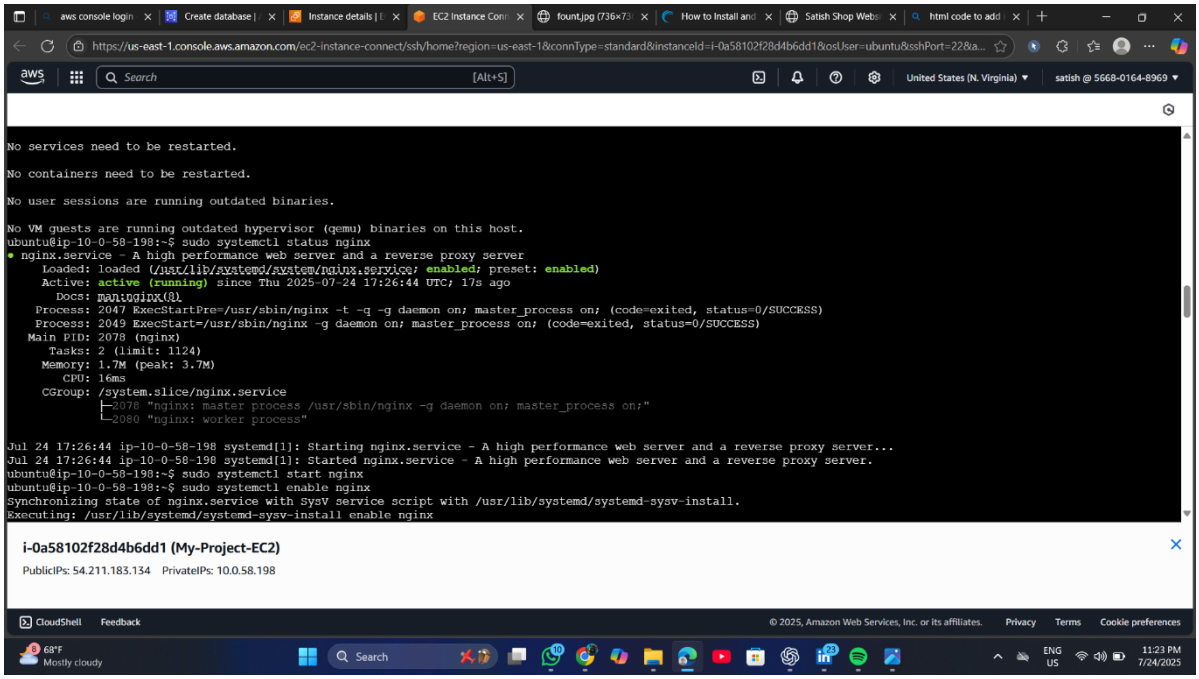
enabled/default sudo apt update

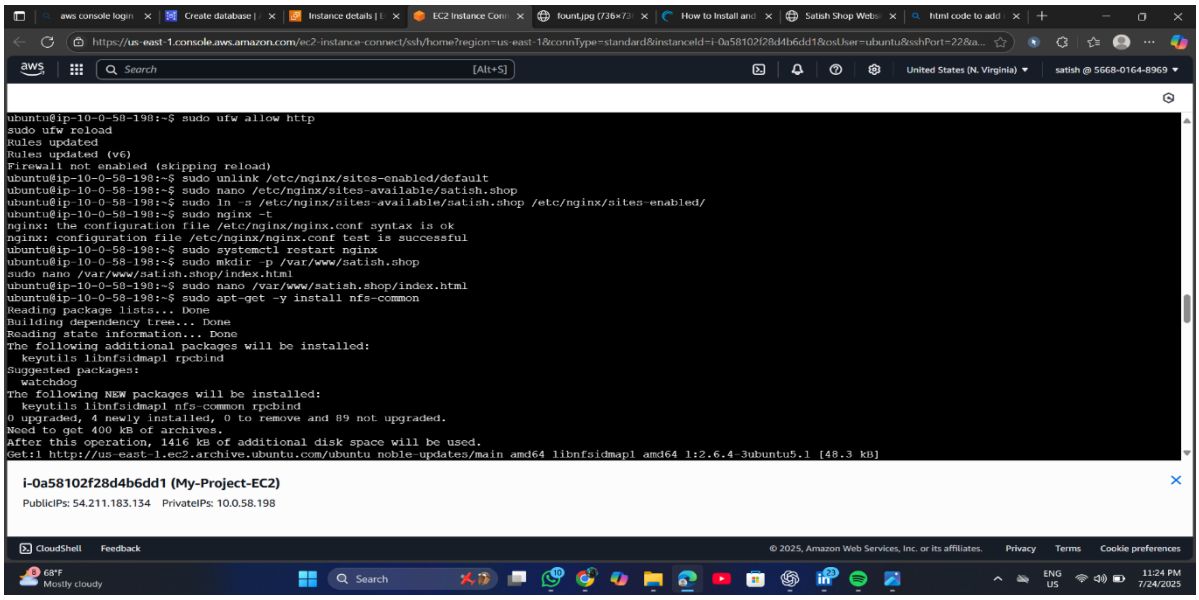
sudo systemctl status nginx

sudo mkdir -p /var/www/satish.shop/html sudo ln -s /etc/nginx/sites-

available/satish.shop /etc/nginx/sites-enabled/

AWS Cloud Infrastructure Deployment and Automation





8. Continue Nginx Setup (via Blog)

Follow the instructions from the linked Linode blog (from the "Use NGINX #" step onwards) for further nginx configuration.

AWS Cloud Infrastructure Deployment and Automation

bash

sudo apt-get -y install nfs-common

2. **Mount EFS volume** (replace fs-xxxx with your EFS file system id):

bash

3. Create or move your index.html as needed.

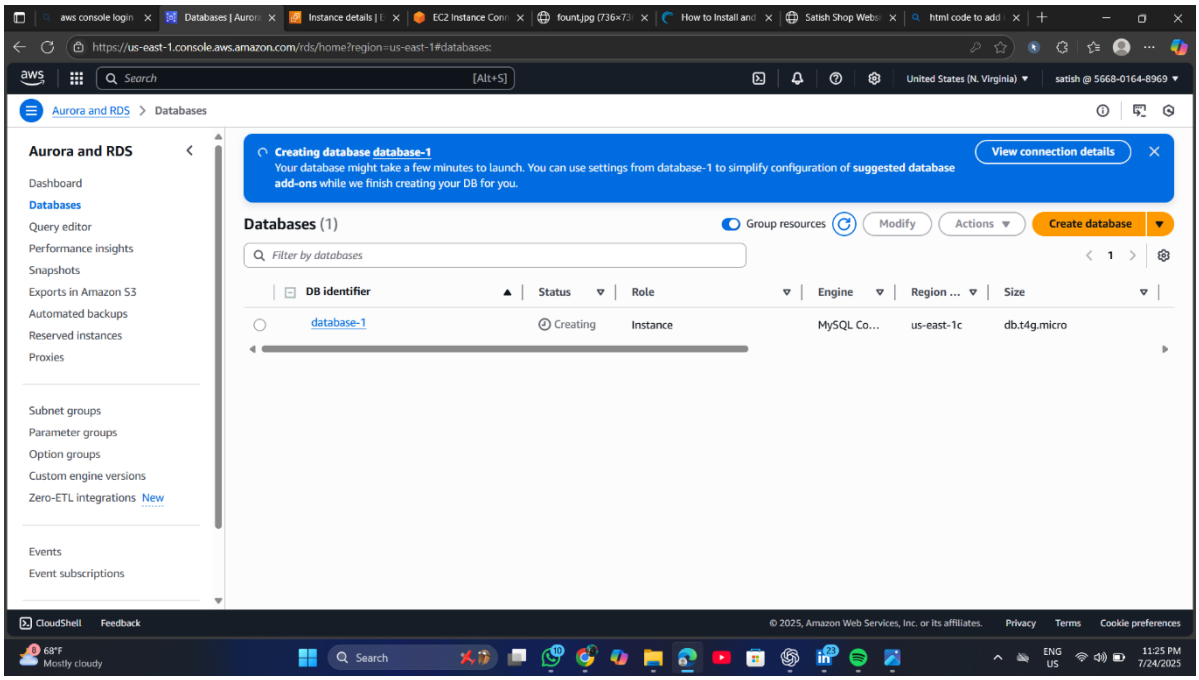
10. Create RDS Database

What:

RDS is AWS's managed relational database.

How:

- Go to **RDS** > **Create database**
- Choose DB engine (MySQL, Postgres, etc.)
- Set username, password, instance type.
- Create data base



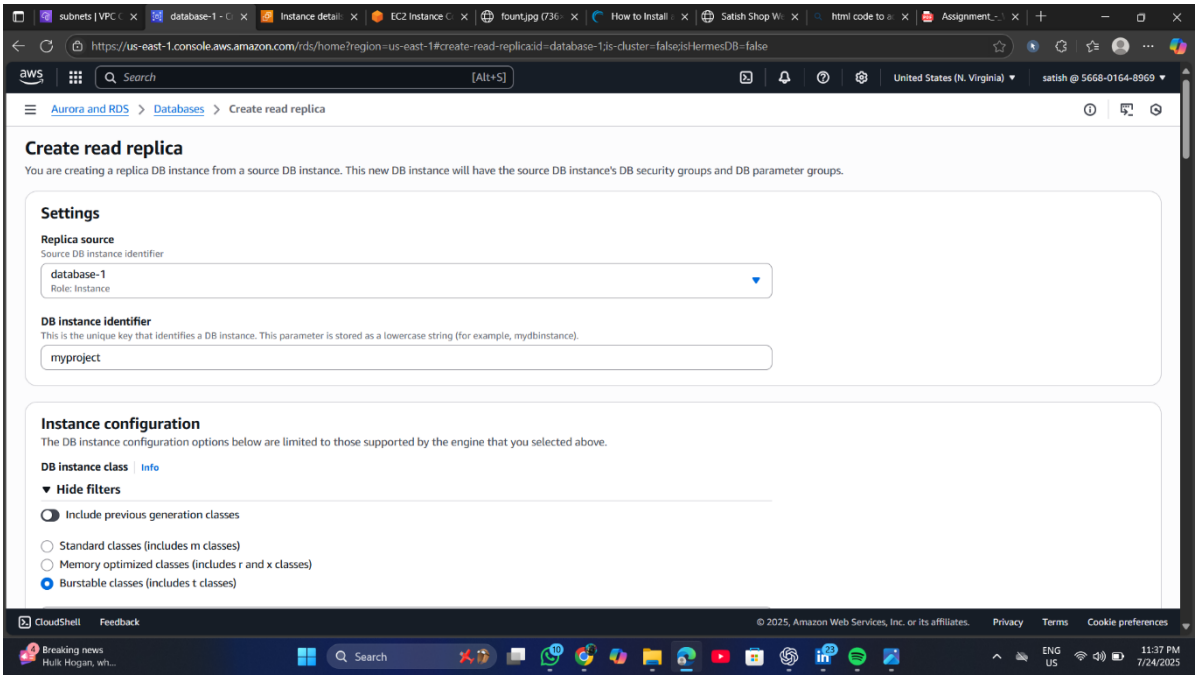
AWS Cloud Infrastructure Deployment and Automation

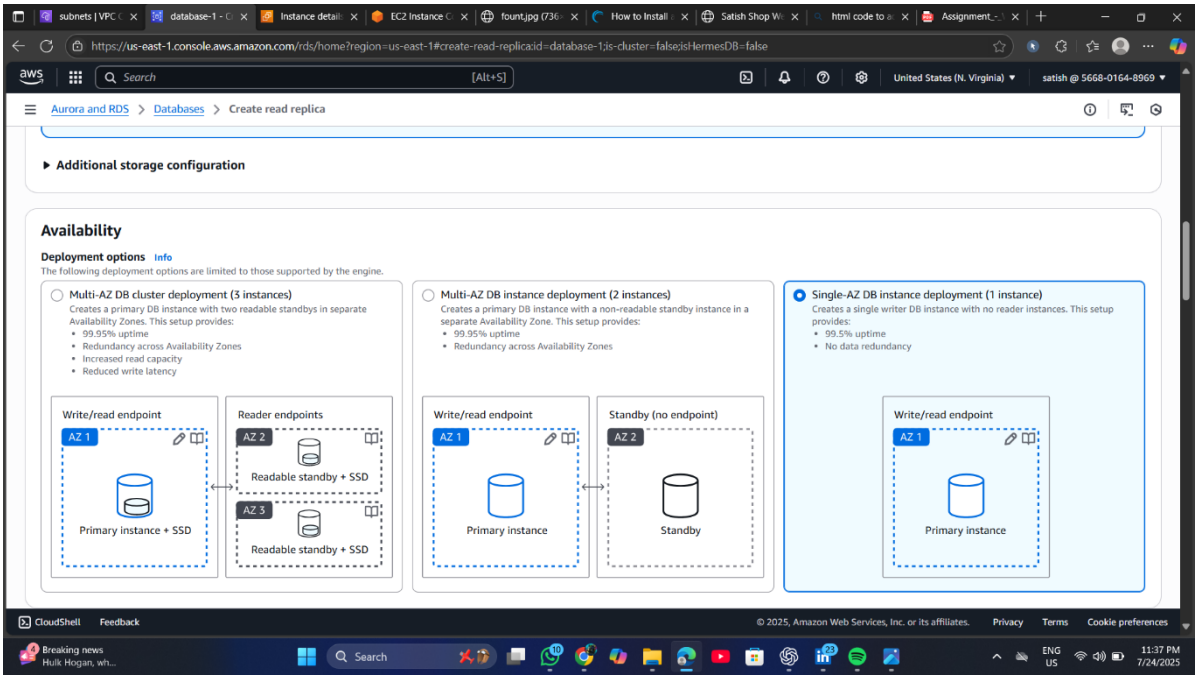
11. Create a Read Replica (Optional)

What/Why:
Improves scalability by offloading reads from the primary database.

How:

- In your RDS instance, select **Create read replica**.





AWS Cloud Infrastructure Deployment and Automation

12. Connect EC2 to RDS How:

1. Install MySQL client:

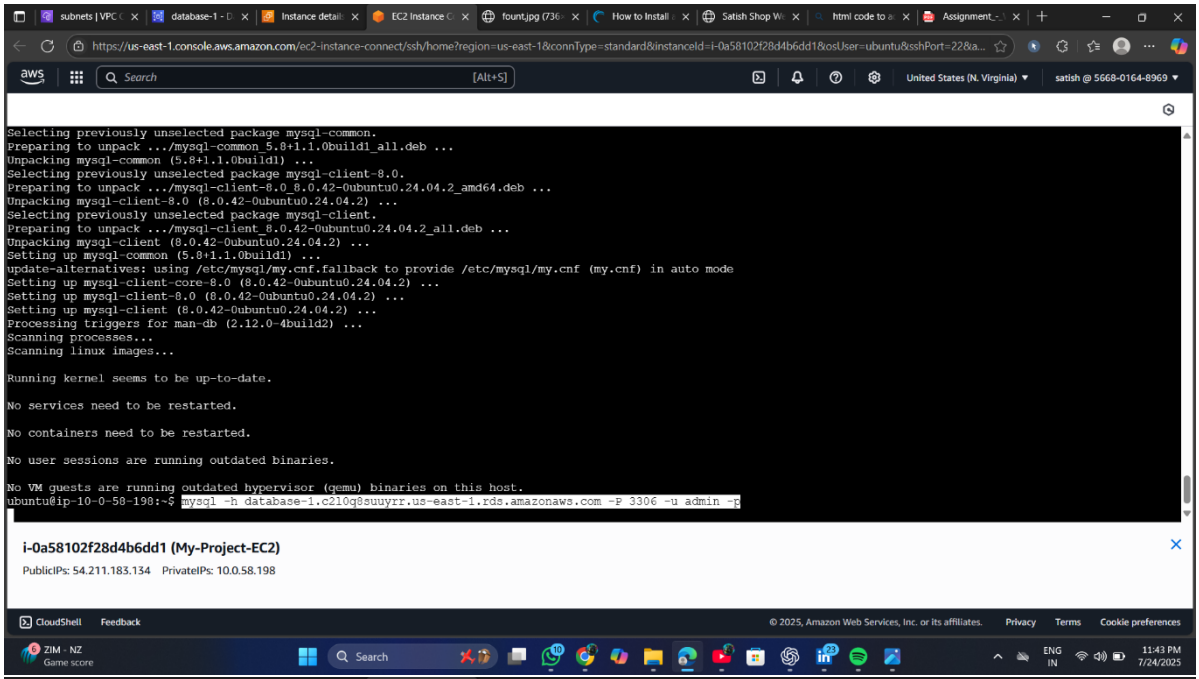
   bash sudo apt-get install mysql-

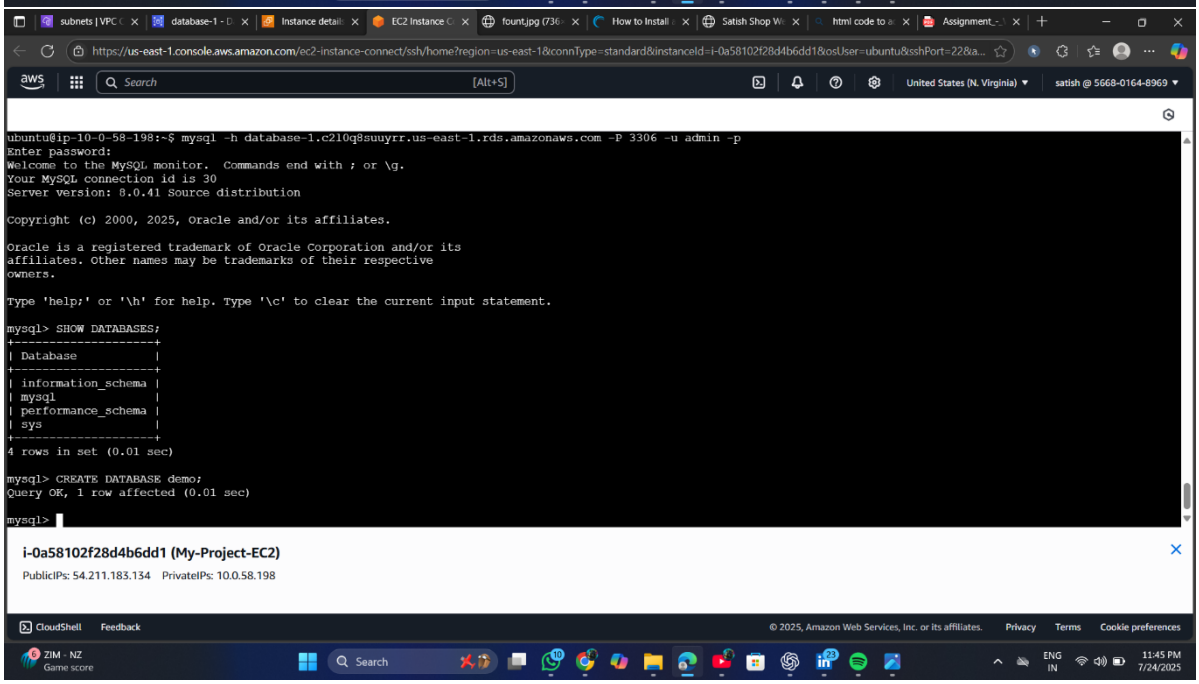
   client

2. Connect to the DB:

   bash

   mysql -h database-1.c210q8suuyrr.us-east-1.rds.amazonaws.com -P 3306 -u admin -p





AWS Cloud Infrastructure Deployment and Automation

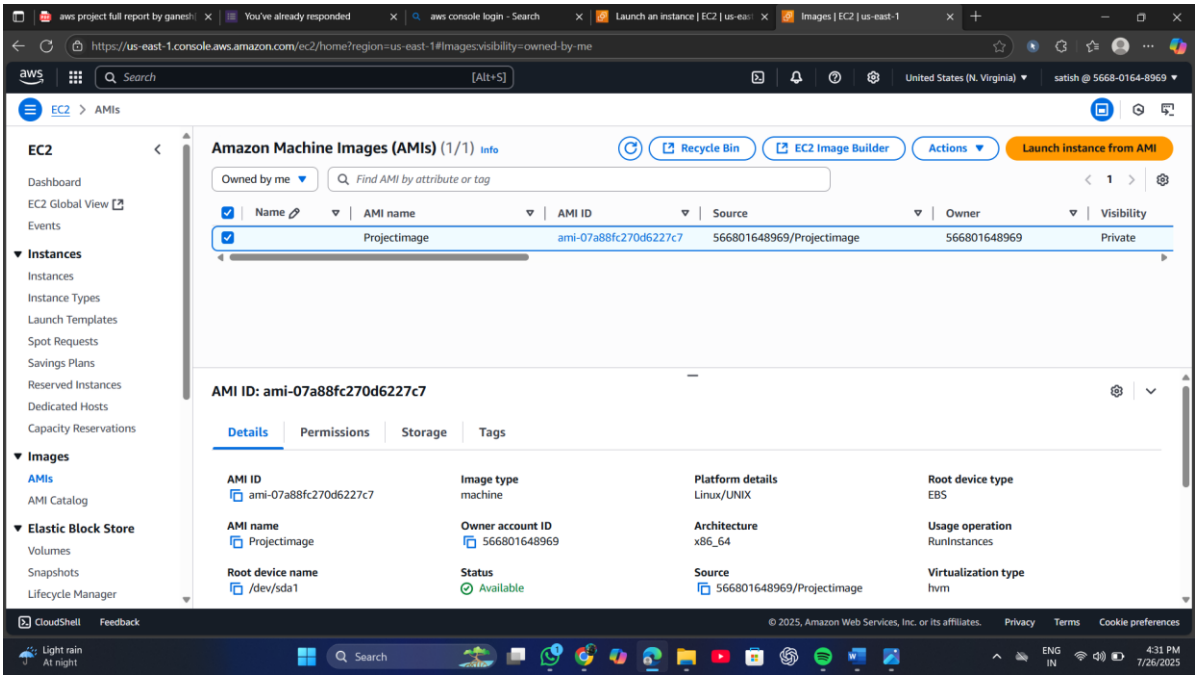
13. Create AMI

What:
AMI (Amazon Machine Image): A snapshot/template of your EC2 instance.

Why:
Allows quick scaling or disaster recovery.

How:

- Select EC2 instance, -> Actions -> Image and Templates -> Create Image.



14. Configure ALB (Application Load Balancer) and ASG (Auto Scaling Group)

What:

- **ALB:** Distributes incoming traffic to multiple EC2 instances.
- **ASG:** Automatically adjusts the number of EC2 instances based on demand.

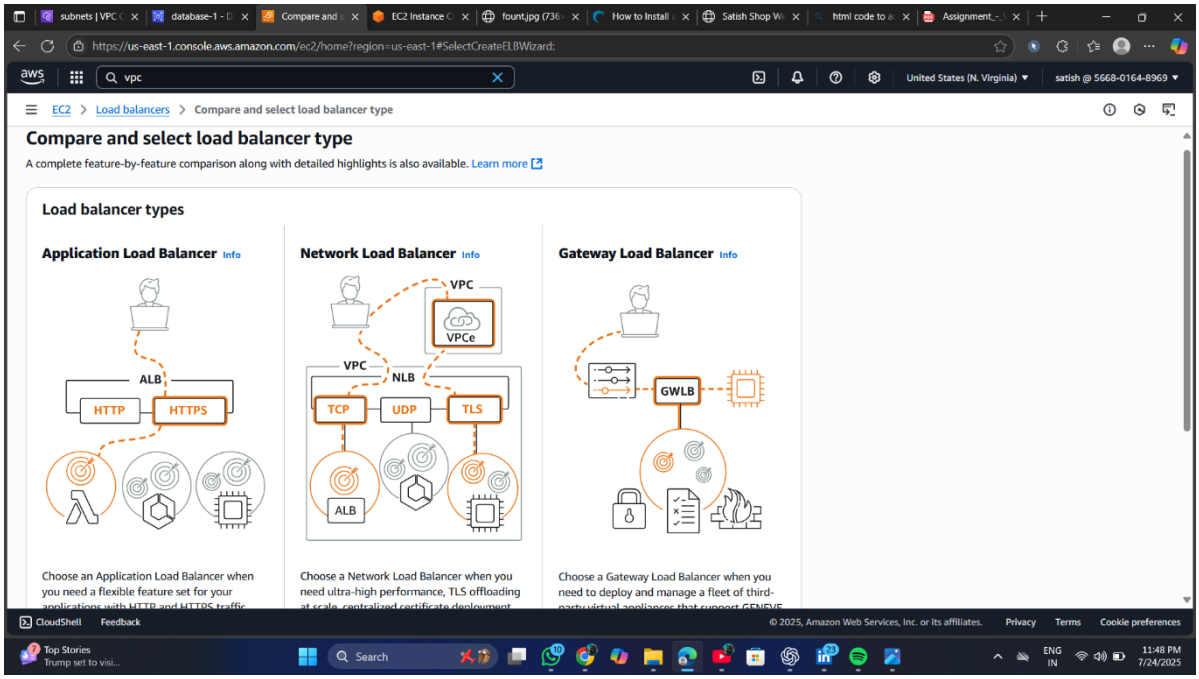
How:

1. **Create ALB:**
    - Go to **EC2 > Load Balancers**.
    - Click **Create Load Balancer** > **Application Load Balancer**.
    - Set up listeners (usually HTTP & HTTPS), choose subnets, and security groups.
    - Register your EC2 instances (or the target group that will auto-register them via ASG).
2. **Create Launch Template/Configuration:**

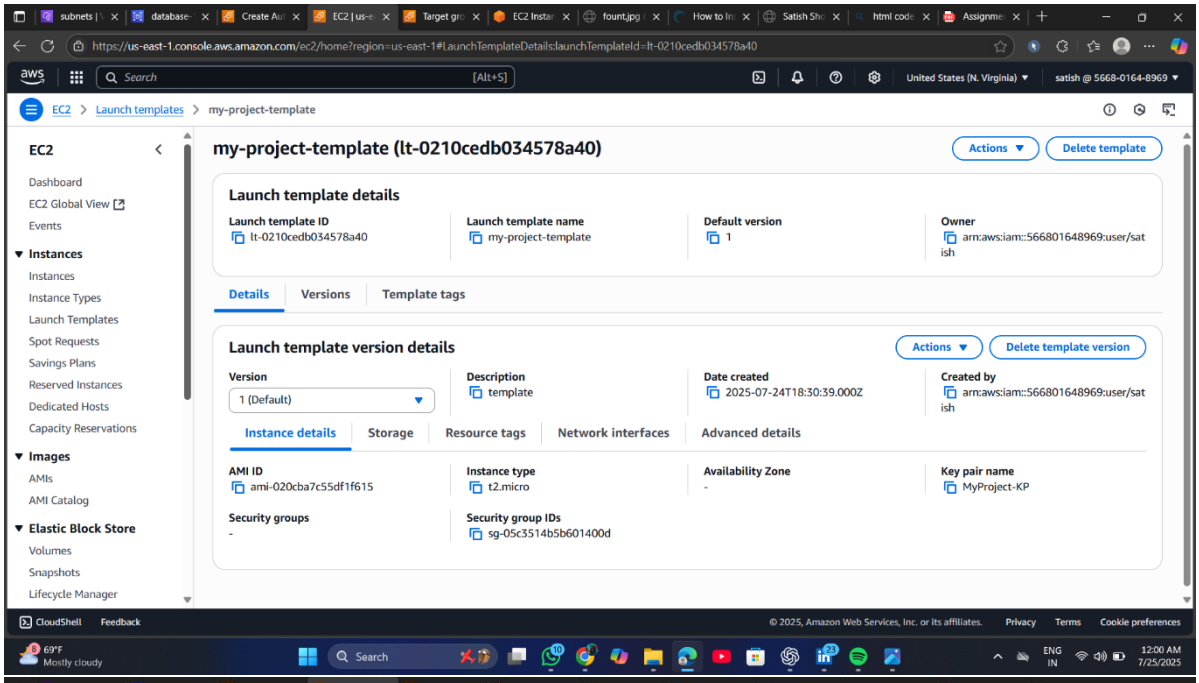
AWS Cloud Infrastructure Deployment and Automation

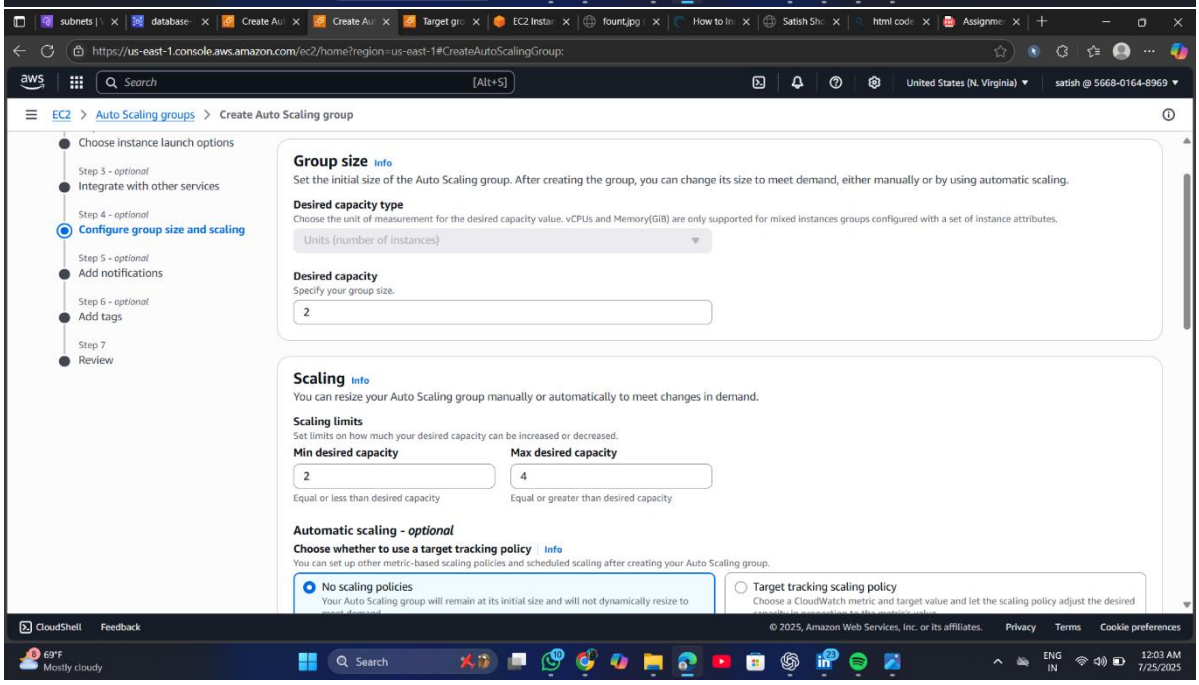
- Go to **EC2 > Launch Templates** and click **Create launch template**.
- Reference your previously created AMI.

3. **Create Auto Scaling Group:**
   - In **EC2 > Auto Scaling Groups**, click **Create Auto Scaling group**.
   - Pick your launch template (from the previous step).
   - Select VPC, subnets, and attach your Target Group (from the ALB).
   - Define min, max, and desired number of EC2 instances.
   - Set scaling policies (optional, for automatic scaling based on CPU/memory, etc.).

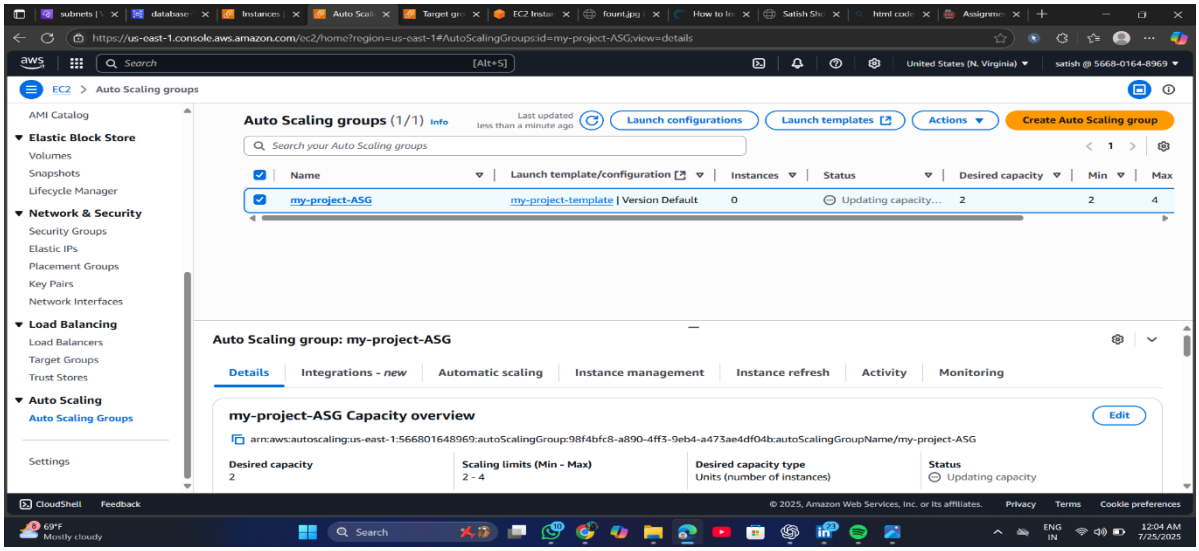


AWS Cloud Infrastructure Deployment and Automation





AWS Cloud Infrastructure Deployment and Automation



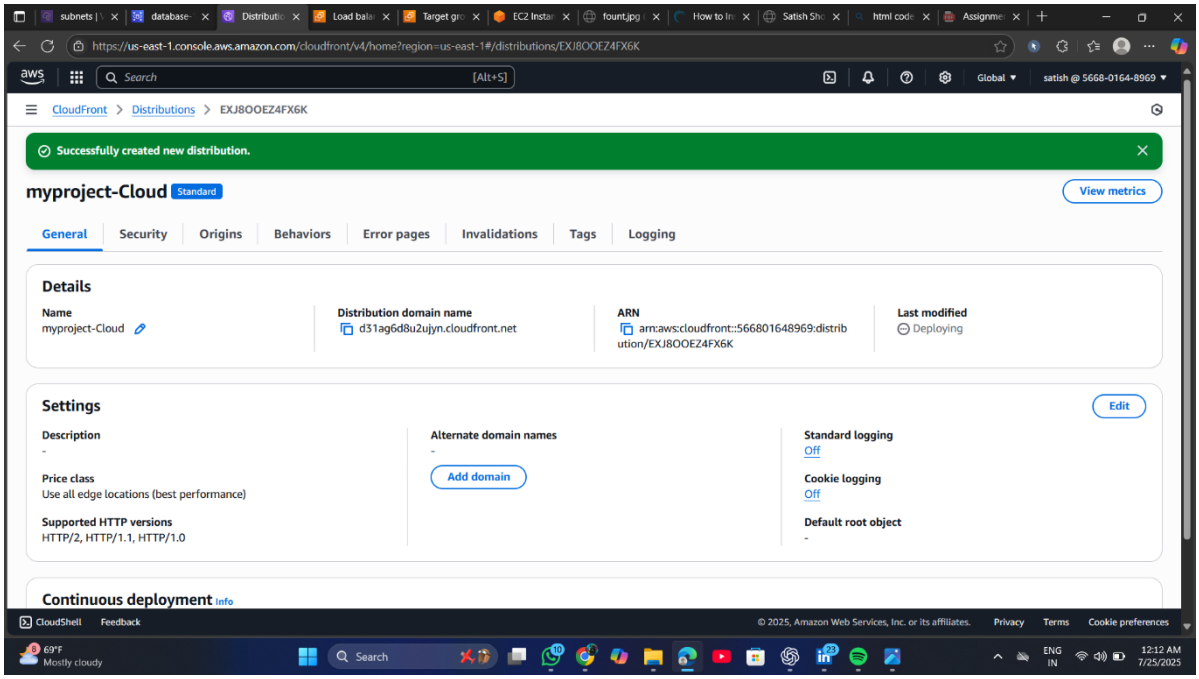
15. Create and Configure CloudFront with ALB

What:

CloudFront is AWS's global Content Delivery Network (CDN); it caches and distributes content closer to users worldwide.

How:

1. Go to **CloudFront > Distributions > Create Distribution**.
2. For origin, enter your ALB's DNS name (found in EC2 > Load Balancers).
3. Configure whether to use HTTP or HTTPS (set up certificates as needed).
4. Customize cache behavior, price class, and any desired settings.
5. Click **Create Distribution**. CloudFront assigns a Distribution Domain Name (e.g., d1xyz.cloudfront.net).



AWS Cloud Infrastructure Deployment and Automation

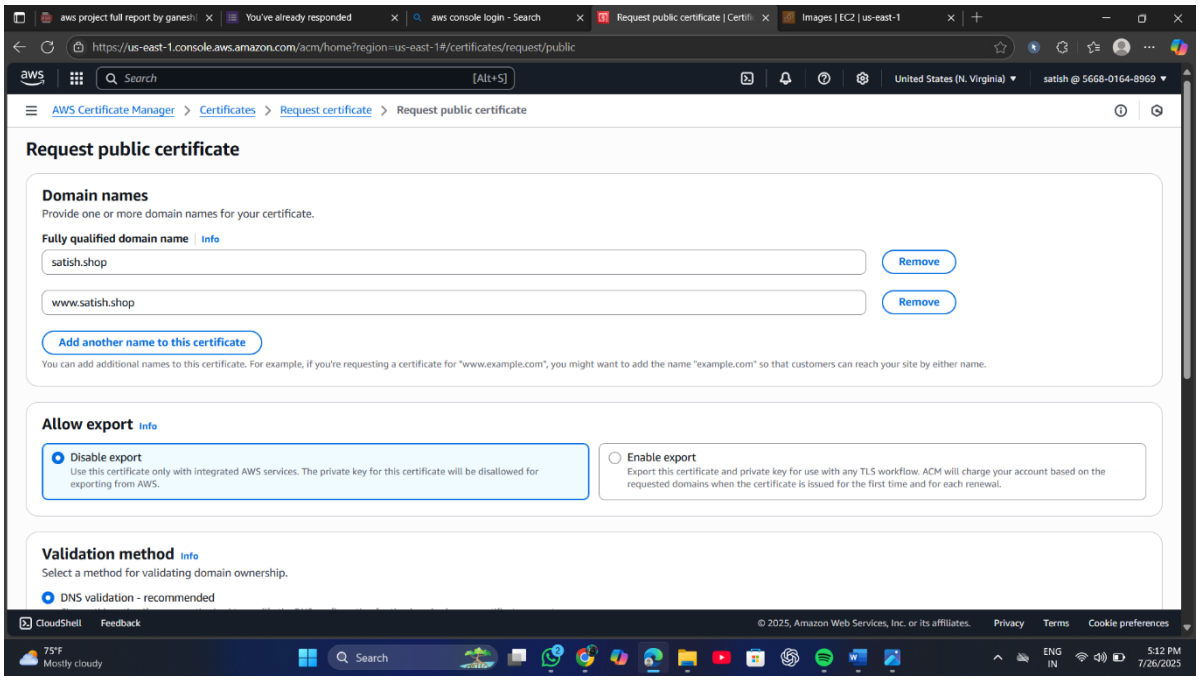
16. Create CNAME, Update Route 53 Records from ACM

What:

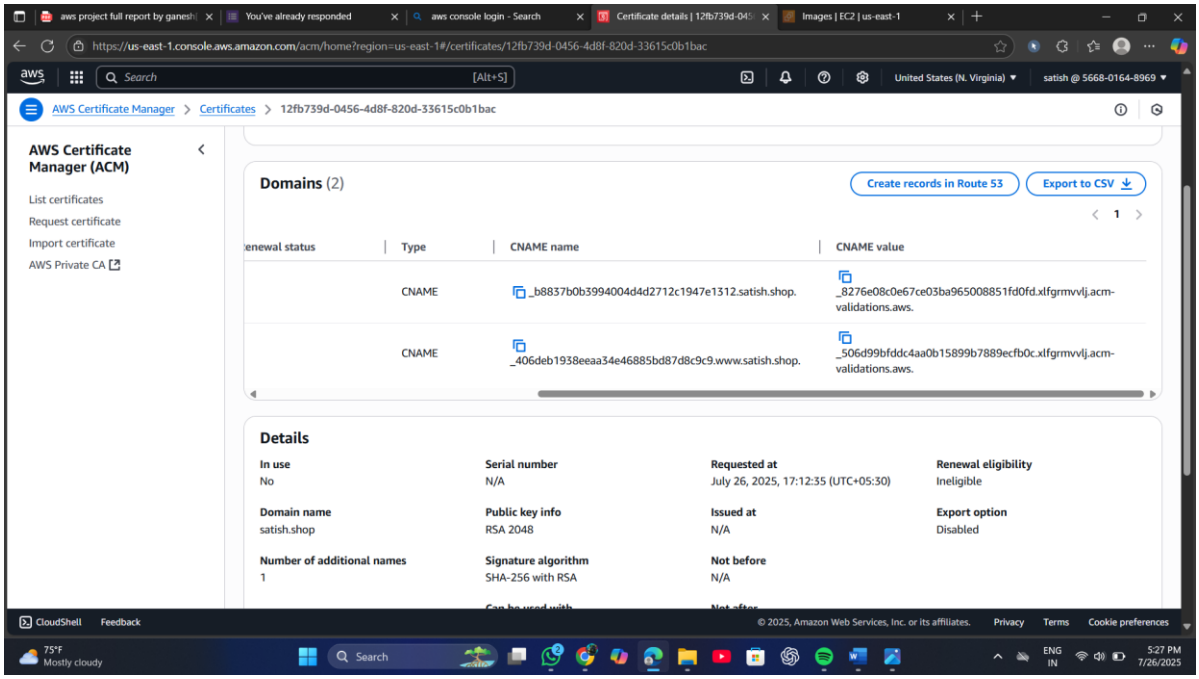
- **ACM (AWS Certificate Manager):** Issues SSL/TLS certificates for HTTPS.
- **Route 53:** AWS DNS service.

How:

1. In **ACM** (Certificate Manager), request a new certificate for your domain.
2. ACM will provide a CNAME record for DNS validation.
3. Go to **Route 53 > Hosted zones**, and for your domain, **create a new CNAME record** with the name and value provided by ACM (for validation).
4. Wait for certificate validation (ACM status will update to "Issued").
5. Later, **in Route 53**, add/modify DNS records to point your domain/subdomain to your ALB or CloudFront:
    - **A/AAAA Alias** record for root/apex domains.
    - **CNAME** for subdomains.



AWS Cloud Infrastructure Deployment and Automation



17. Buy Domain, Use Route 53

How:

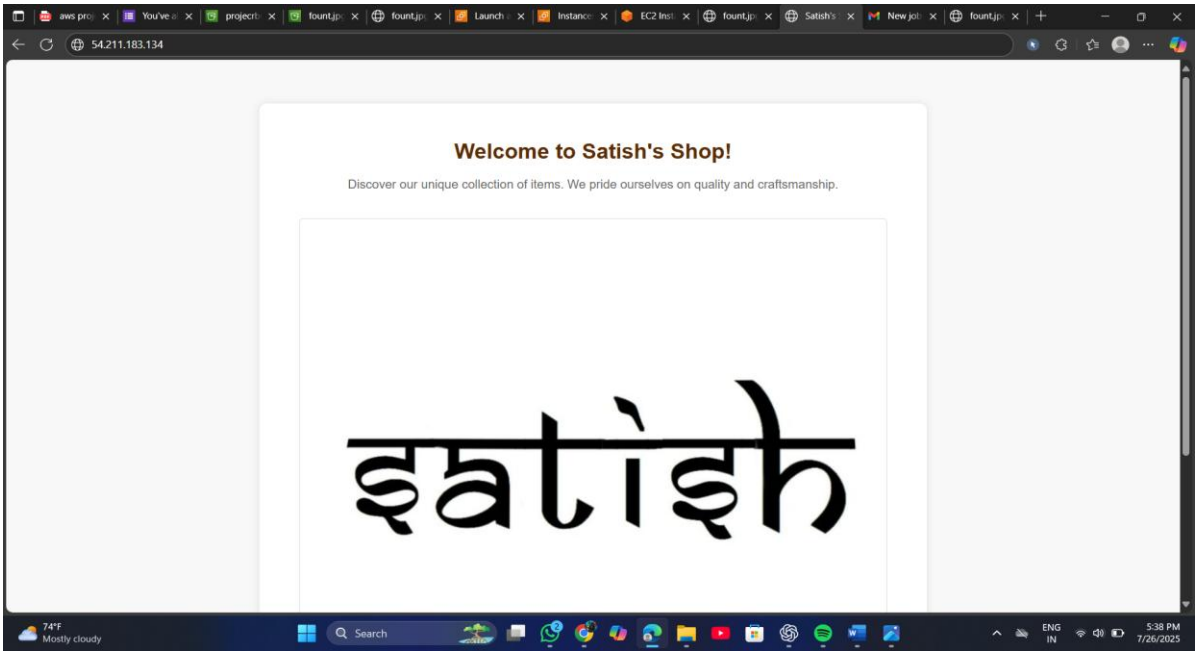
1. In **Route 53 > Registered Domains**, click **Register Domain** and search for your desired name.
2. Complete the registration and payment steps.
3. After registration, AWS sets up a Hosted Zone for your domain.
4. If using another registrar, update your domain's nameservers to those provided by AWS Route 53.

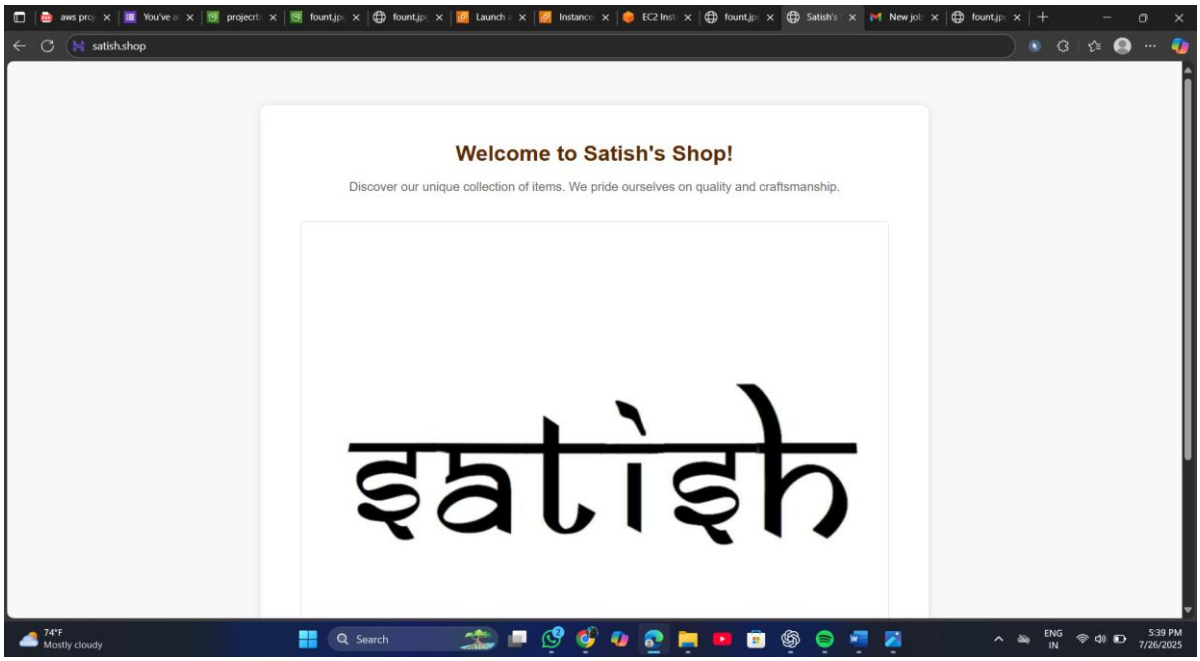
18. Configure Route 53 with CloudFront

How:

1. In **Route 53 > Hosted zones**, select your domain.
2. Create a new record:
    - **Type:** A or AAAA (Alias).
    - **Name:** (e.g., www or leave blank for the root domain).
    - **Alias:** Yes.
    - **Alias Target:** Select your CloudFront distribution's domain name (shows in dropdown if in the same account).
3. Save the record.

AWS Cloud Infrastructure Deployment and Automation





Now, internet users who visit your domain will:

- Be routed via Route 53 (DNS).
- Go through CloudFront (CDN/caching).
- Reach your Application Load Balancer (ALB).
- Get served by available EC2 instances (managed by Auto Scaling Group—ASG).
- Benefit from secure HTTPS (provided by ACM certificate).

Satish Kumar Singilese

Mail: satishkumarcse2003@gmail.com

AWS Cloud Infrastructure Deployment and Automation