5.Configure Route 53 with Your Custom Domain and SSL using ACM Purchase a domain (or use an existing one) and configure it in Route 53. Request an SSL certificate using ACM and associate it with the load balancer. Ensure HTTPS traffic is properly routed to the application.

Secure Application Hosting on AWS (EC2 + ALB + CloudFront + ACM + Route 53)

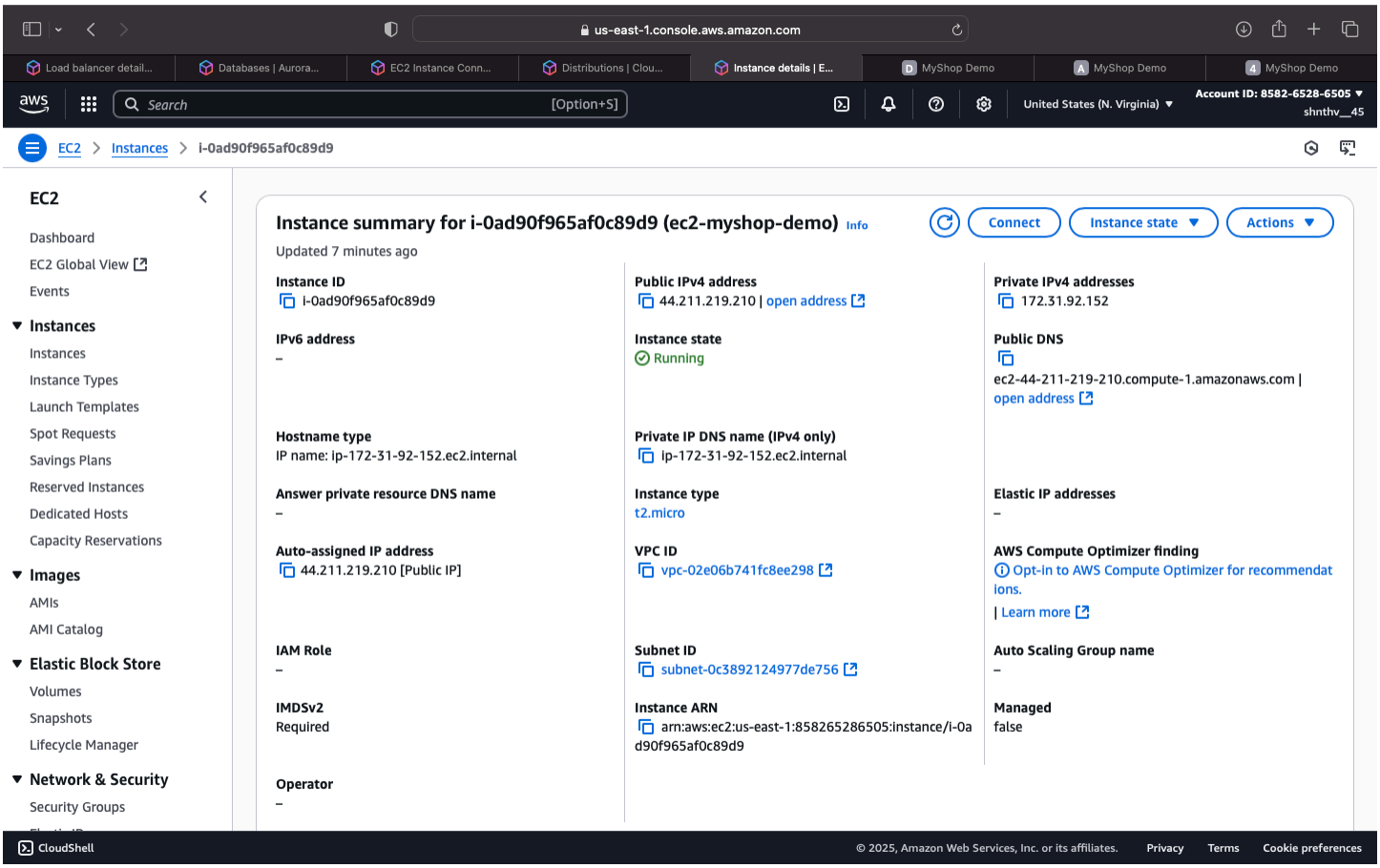
This documentation explains step by step how to:

1. Launch an **EC2 instance** to host a simple web application.
2. Configure an **Application Load Balancer (ALB)** to distribute traffic.
3. Use **CloudFront** to provide global delivery and free HTTPS.
4. Request and attach an SSL certificate with **AWS Certificate Manager (ACM)**.
5. Configure **Route 53** with a custom domain in production.
6. Attach the ACM certificate directly to the **ALB HTTPS listener** (as the requirement states).

With this demo, you'll achieve secure access (HTTPS) using AWS's free CloudFront domain. For real-world production, you would purchase a domain, configure Route 53, request ACM certificates, and associate them with the ALB.

Step 1: Launch an EC2 Instance

1. **Go to EC2 → Launch Instance.**
2. **Name: ec2-myshop-demo**
3. **AMI: Amazon Linux 2 (free tier eligible)**
4. **Instance type: t2.micro (free tier eligible)**
5. **Key pair: Select or create a key pair.**
6. **Network settings:**
   - **VPC: default VPC**
   - **Subnet: choose a public subnet**
   - **Auto-assign public IP: enabled**
   - **Security group: create sg-ec2-web with inbound rules:**
     - **HTTP (80) from 0.0.0.0/0**
     - **SSH (22) from your IP or 0.0.0.0/0 (optional, for admin)**
7. **Launch instance.**



install a web server (after SSH into EC2):

sudo yum update -y

sudo yum install -y httpd

sudo systemctl start httpd

sudo systemctl enable httpd

echo "<h1>Hello from EC2 backend</h1>" | sudo tee /var/www/html/index.html

or

HTML

<!DOCTYPE html>

<html lang="en">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

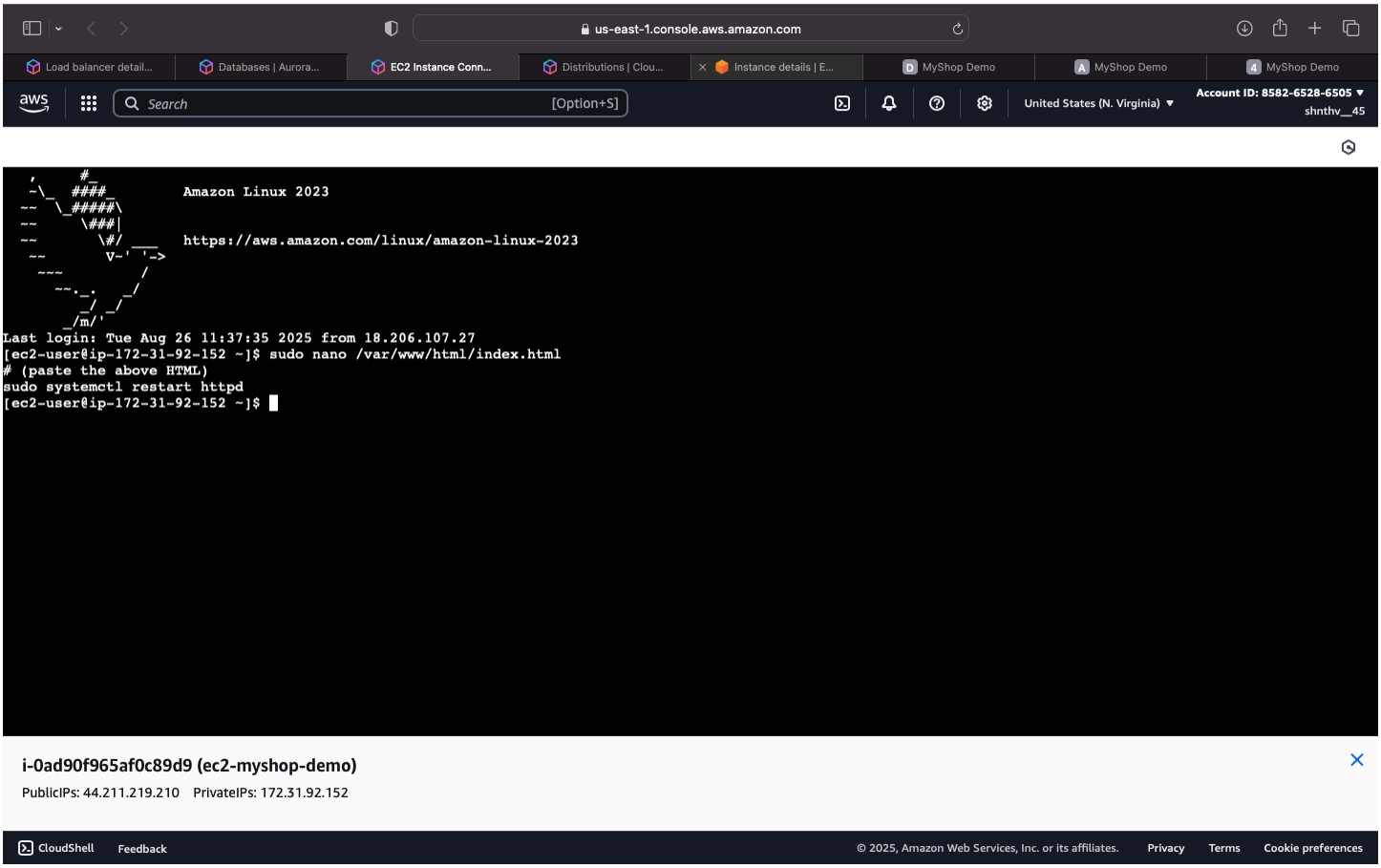
<title>MyShop Demo</title>

```
<style>
  body {
    margin: 0;
    font-family: Arial, sans-serif;
    background: linear-gradient(120deg, #4facfe, #00f2fe);
    color: #333;
    display: flex;
    flex-direction: column;
    justify-content: center;
    align-items: center;
    height: 100vh;
    text-align: center;
  }
  h1 {
    font-size: 3em;
    margin-bottom: 0.2em;
    color: #fff;
    text-shadow: 2px 2px 5px rgba(0,0,0,0.3);
  }
  p {
    font-size: 1.2em;
    color: #fefefe;
  }
  .card {
    background: rgba(255,255,255,0.15);
    padding: 20px 40px;
    border-radius: 12px;
```

```
      backdrop-filter: blur(6px);
      box-shadow: 0 8px 16px rgba(0,0,0,0.2);
    }
    footer {
      position: absolute;
      bottom: 20px;
      font-size: 0.9em;
      color: #fefefe;
    }
  </style>
</head>
<body>
  <div class="card">
    <h1>🚀 Welcome to MyShop Demo</h1>
    <p>Your EC2 instance is running successfully behind an ALB + CloudFront!</p>
  </div>
  <footer>
    Powered by <strong>AWS EC2</strong> | Secure with <strong>CloudFront & ACM</strong>
  </footer>
</body>
</html>
```

On your EC2:

```
sudo nano /var/www/html/index.html
# (paste the above HTML)
sudo systemctl restart httpd
```

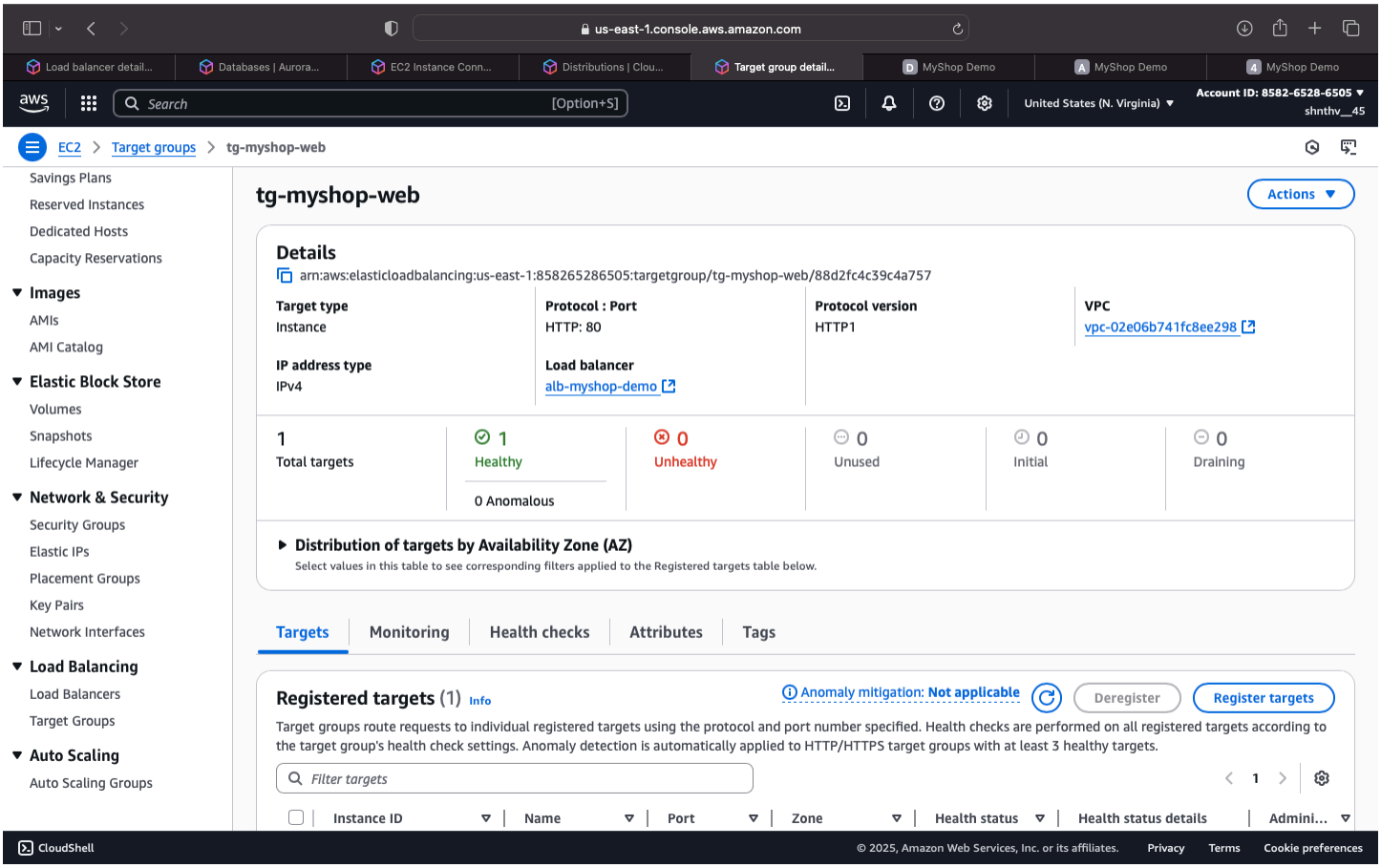


Outcome: Access http://44.211.219.210 → should display *Hello from EC2 backend*

Step 2: Create a Target Group

1. Go to **EC2 → Target Groups → Create Target Group**.
2. **Name:** tg-myshop-web
3. **Target type:** Instances
4. **Protocol / Port:** HTTP : 80
5. **VPC:** Select your VPC
6. **Health check path:** /
7. Register your EC2 instance on port 80.

Outcome: EC2 instance shows as **Healthy**.

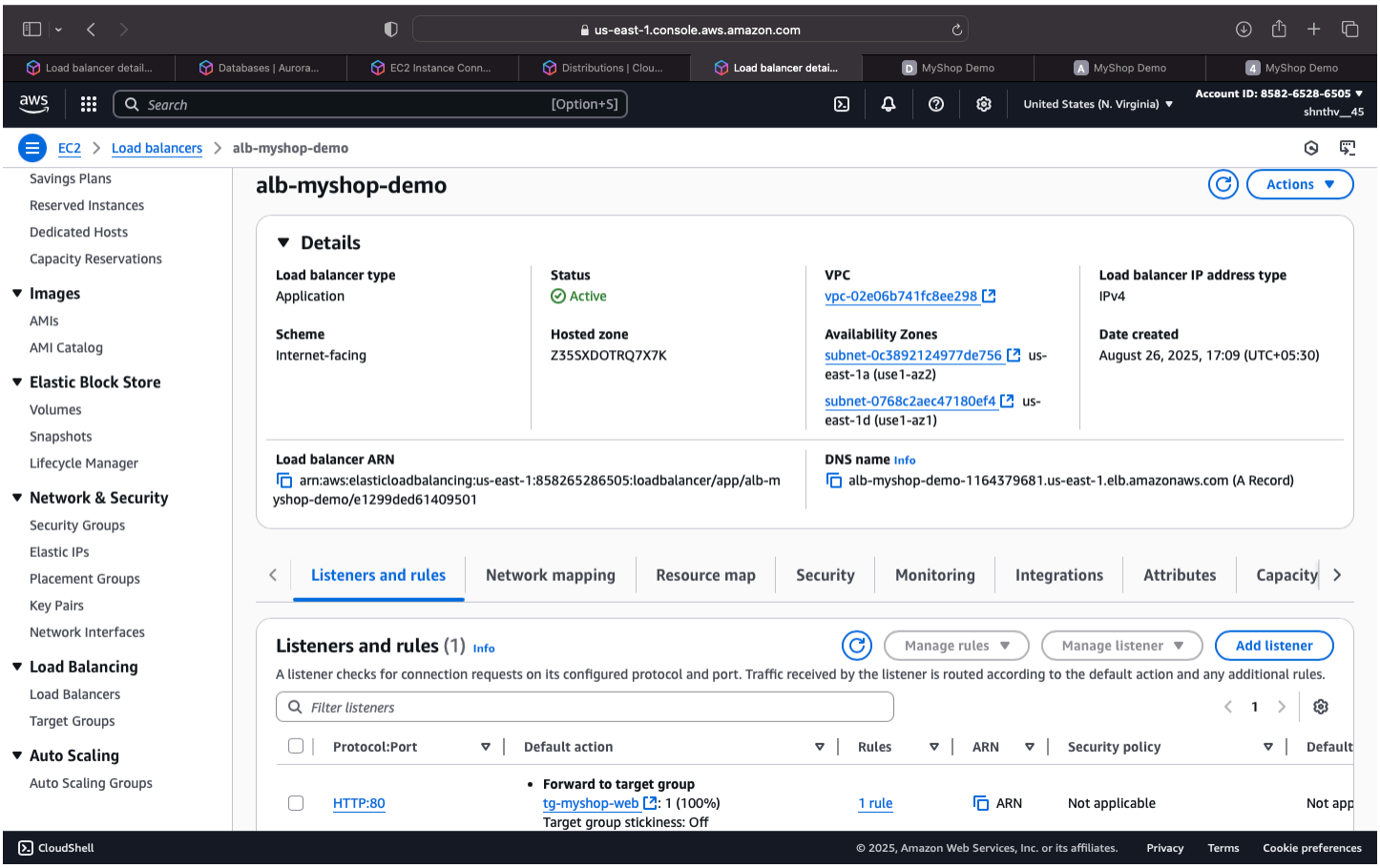


Step 3: Create an Application Load Balancer (ALB)

1. Go to **EC2 → Load Balancers → Create Load Balancer → Application Load Balancer**.
2. **Name:** alb-myshop-demo
3. **Scheme:** Internet-facing
4. **IP address type:** IPv4
5. **Network mapping:** Select two public subnets in different AZs.
6. **Security group:** Create sg-alb-public with inbound rule:
   - HTTP (80) from 0.0.0.0/0
7. **Listeners:**
   - Add HTTP : 80 → forward to tg-myshop-web
8. Click **Create load balancer**.

Outcome: ALB DNS (example):

http://alb-myshop-demo-1164379681.us-east-1.elb.amazonaws.com



Visit http://<ALB-DNS> ➜ loads the EC2 page.

Step 4: Create a CloudFront Distribution (HTTPS for Demo)

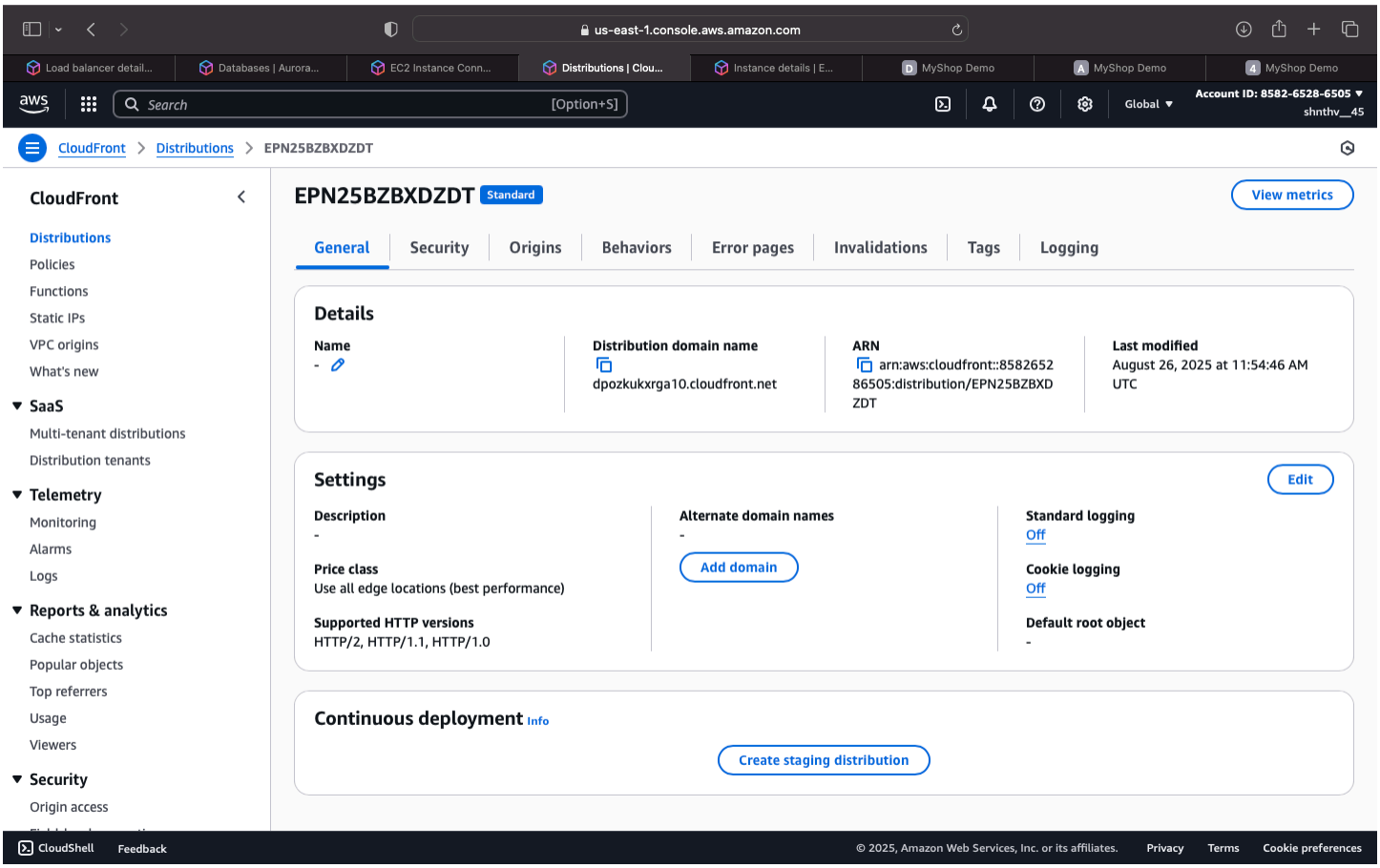
1. Go to **CloudFront ➜ Create Distribution**.
2. **Origin domain:** Enter your ALB DNS name.
3. **Origin name:** origin-myshop-alb
4. **Origin protocol policy:** HTTP only (for demo).
5. **Default cache behavior:**
   - Viewer protocol policy: Redirect HTTP to HTTPS
   - Allowed methods: GET, HEAD (or ALL if needed)
   - Cache policy: CachingDisabled (for dynamic apps)
6. **Settings:**
   - Alternate domain names: *(leave blank for demo)*
   - SSL certificate: Use default CloudFront certificate (*.cloudfront.net)

Click **Create distribution**.

Outcome: CloudFront domain (example):

dpozkukxrga10.cloudfront.net

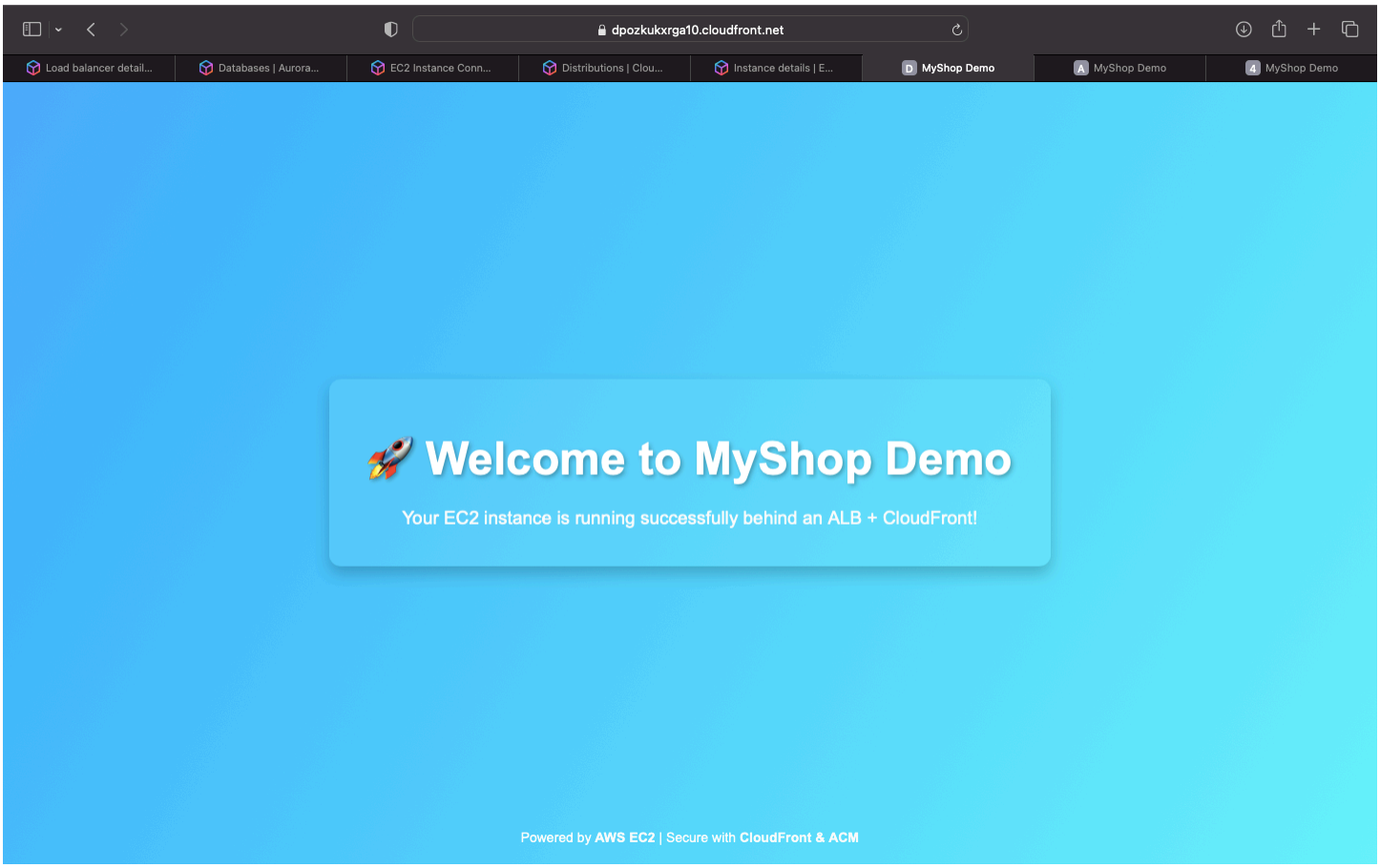
Visit https://dpozkukxrga10.cloudfront.net ➜ app loads securely over HTTPS.



Step 5: Verify End-to-End Flow

- Test EC2 directly ➜ http://44.211.219.210 ➜ works.
- Test ALB ➜http://alb-myshop-demo-1164379681.us-east-1.elb.amazonaws.com ➜ works.
- Test CloudFront ➜ https://dpozkukxrga10.cloudfront.net ➜ works with lock icon.
- Test HTTP➜HTTPS redirect on CloudFront ➜ automatically redirects.

Outcome: Application is secure and globally available.



Step 6: Production Setup with Route 53 + Custom Domain + ACM

If you purchase a domain (e.g., myshop.com):

6.1 Route 53 Setup

1. Create a **Route 53 Hosted Zone** for myshop.com.
2. Update your registrar's nameservers to Route 53.

6.2 Request ACM Certificates

1. Go to **ACM (us-east-1 for CloudFront)** → Request a public certificate for www.myshop.com.
2. Validate via DNS (CNAME record in Route 53).
3. Go to **ACM (same region as ALB, e.g., ap-south-1)** → Request a public certificate for app.myshop.com.
4. Validate via DNS.

6.3 Attach Certificate to ALB

1. In **EC2 → Load Balancers → Listeners**, edit or add **HTTPS (443)** listener.
2. Choose the ACM certificate for app.myshop.com (regional cert).
3. Default action: Forward to tg-myshop-web.

Outcome: The ALB can now terminate HTTPS traffic with your ACM certificate.

6.4 Update CloudFront

1. Add www.myshop.com as an **Alternate Domain Name (CNAME)**.
2. Attach the ACM certificate (us-east-1) for www.myshop.com.
3. Origin protocol policy → HTTPS only (CloudFront → ALB).

6.5 Route 53 DNS Records

1. Add **A/AAAA Alias Record**: www.myshop.com → CloudFront distribution.
2. (Optional) Add apex myshop.com → CloudFront distribution.

Outcome: Users access your app via https://www.myshop.com (CloudFront) → https://app.myshop.com (ALB) → EC2 backend.