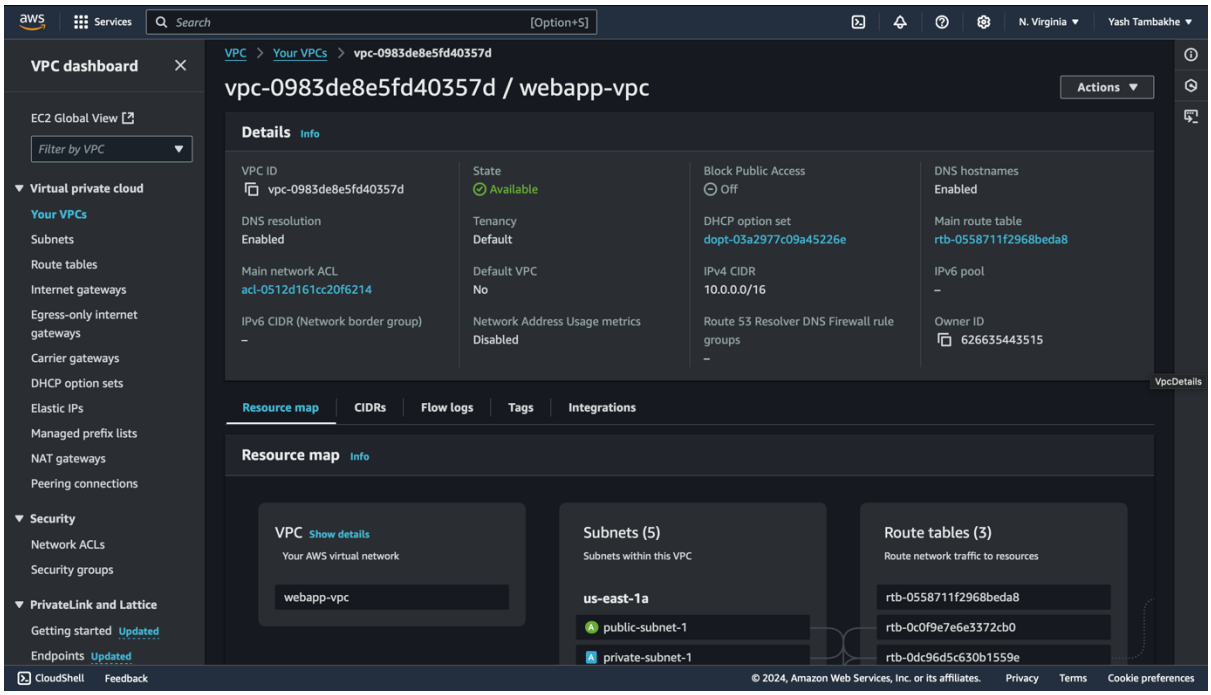
Assignment 8

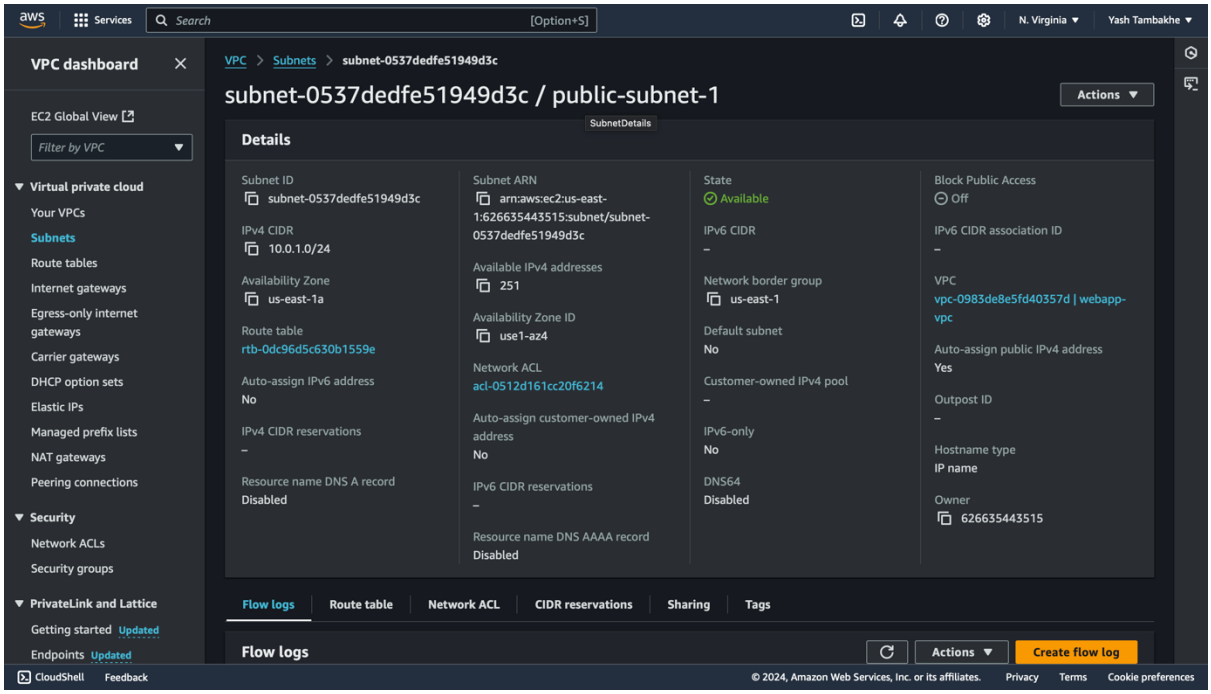
Step 1: Networking Configuration

The screenshot captures the VPC creation summary page, displaying key details such as the VPC ID, name, and CIDR block. This confirms the successful creation of the VPC with the specified configuration.

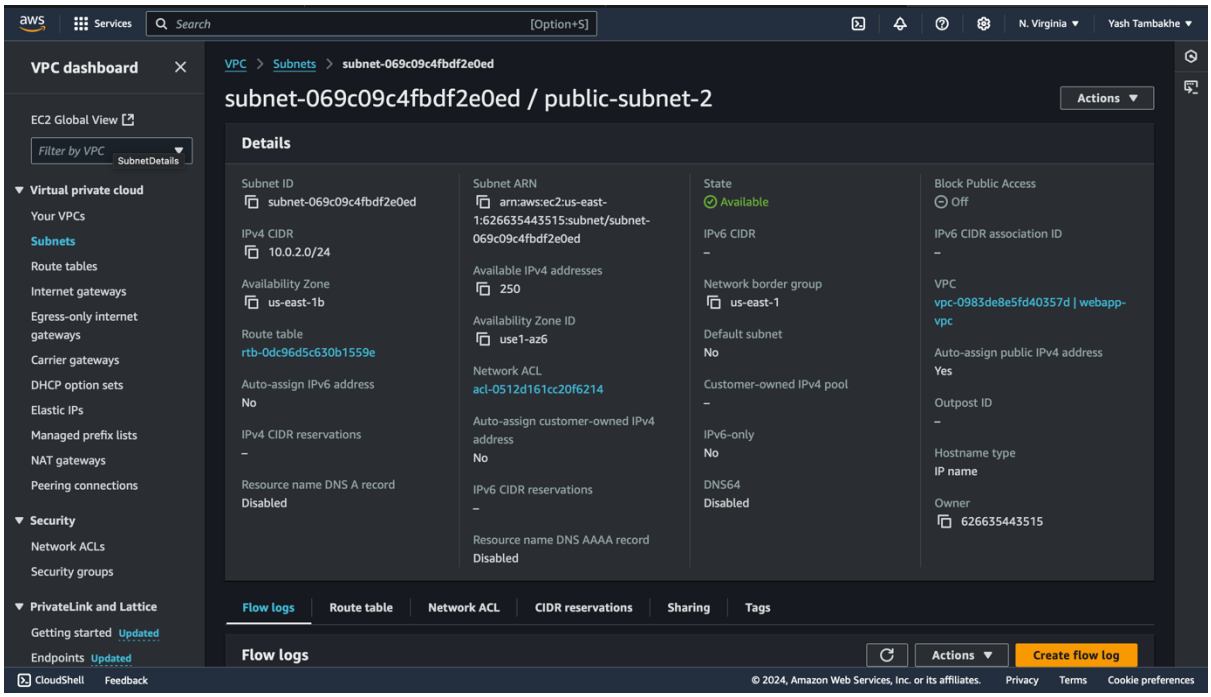


The screenshot captures the subnet configurations, showing details such as the CIDR blocks and the associated availability zones. This provides a clear overview of the network structure within the VPC.

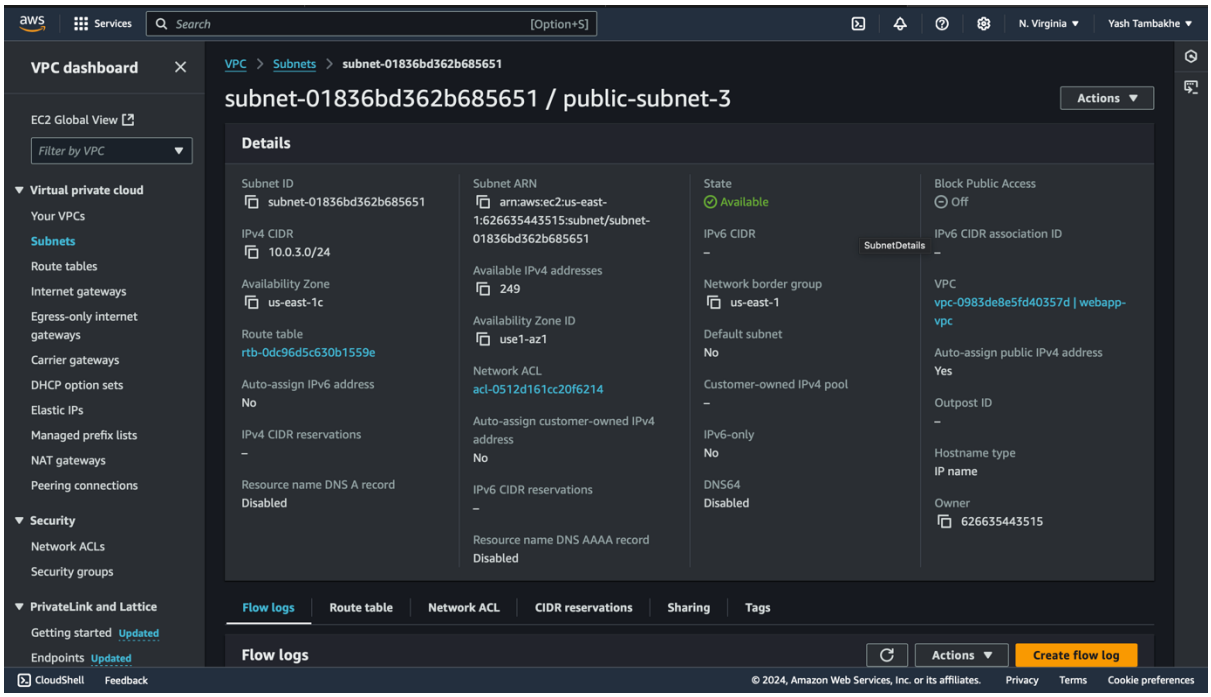
Public Subnet 1



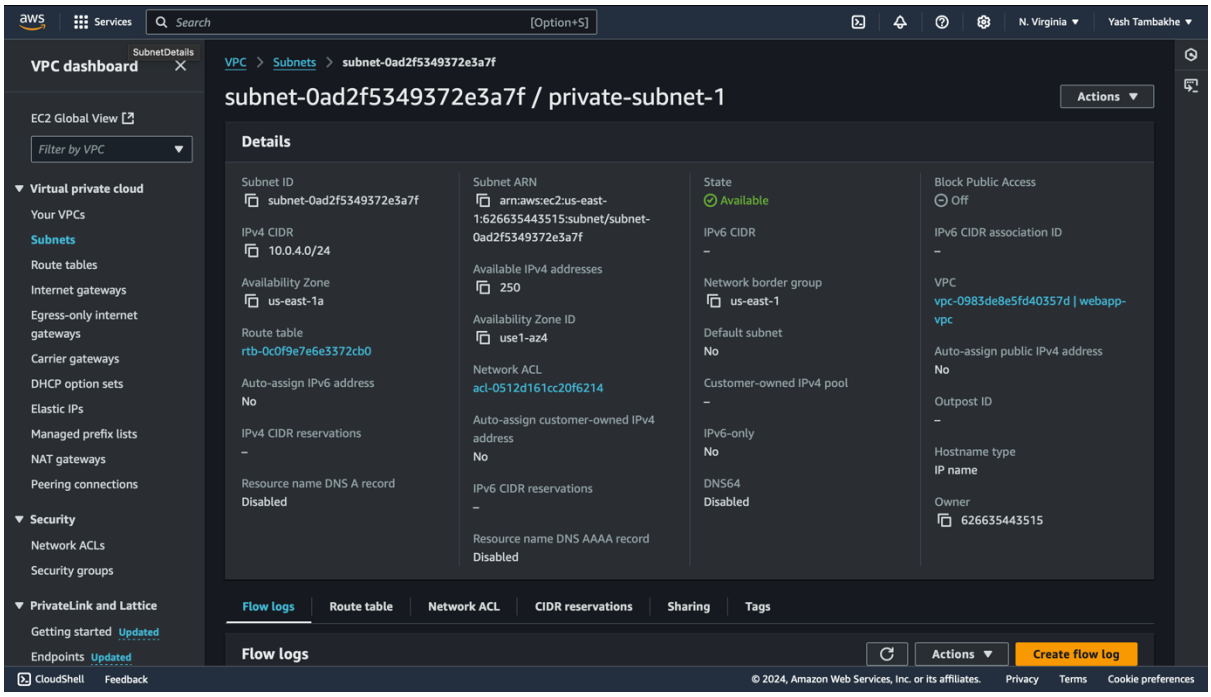
Public Subnet 2



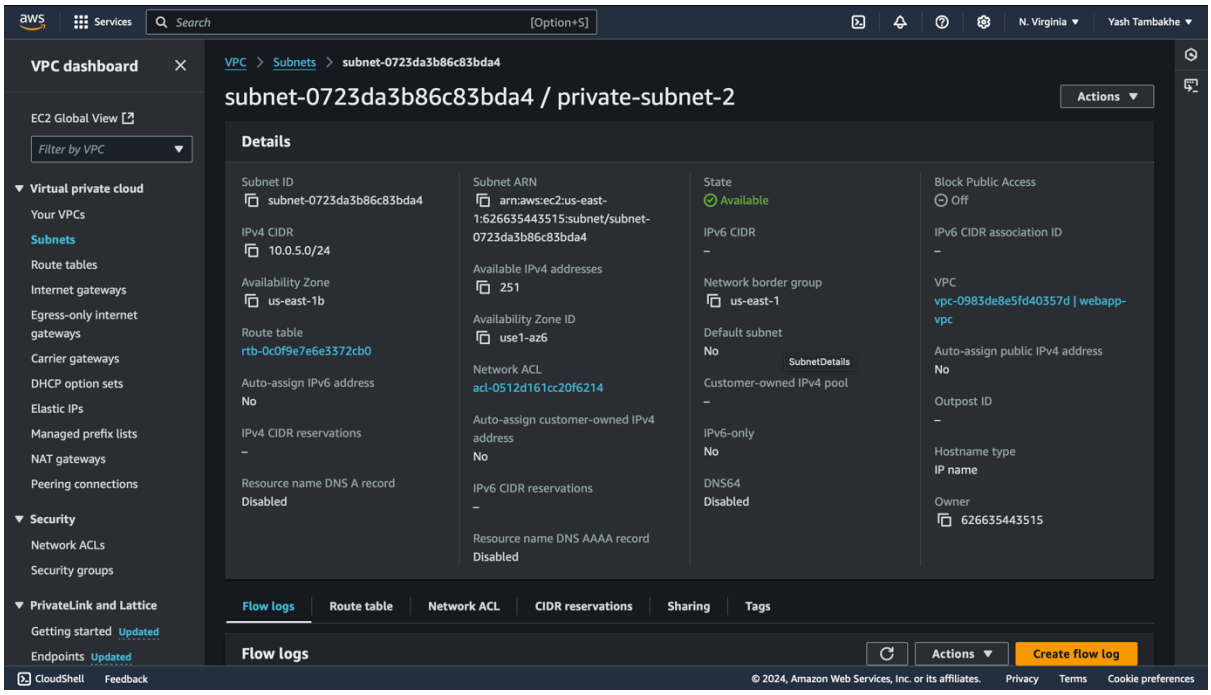
Public Subnet 3



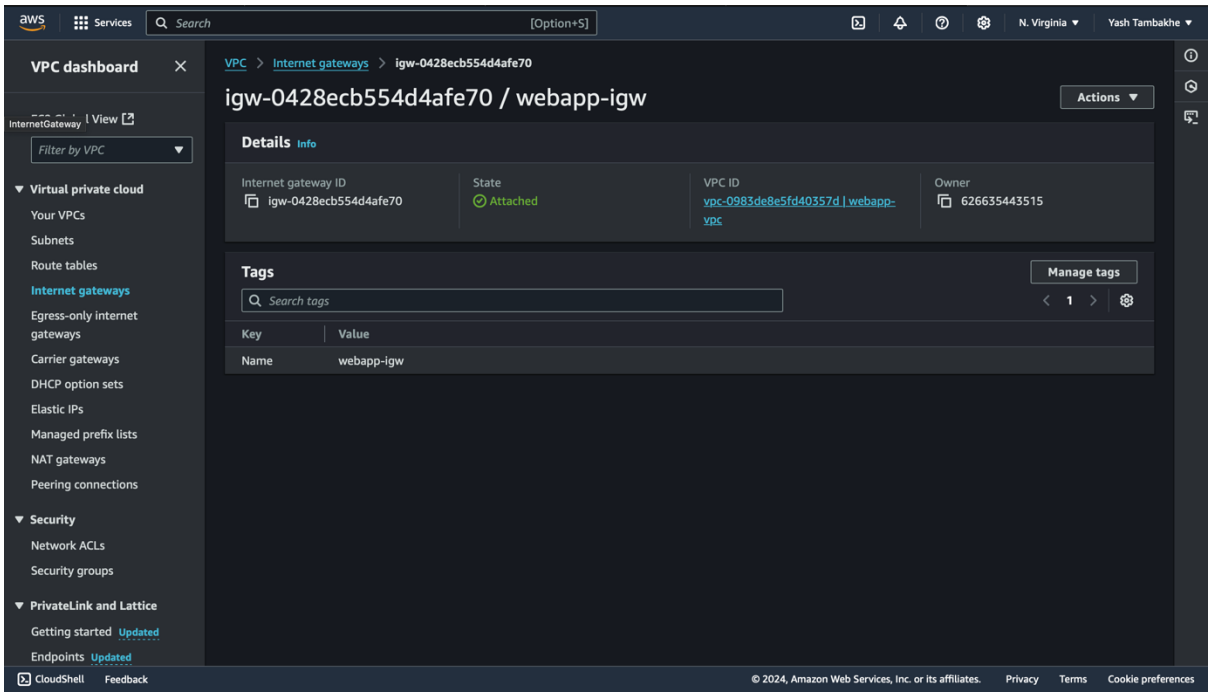
Private Subnet 1



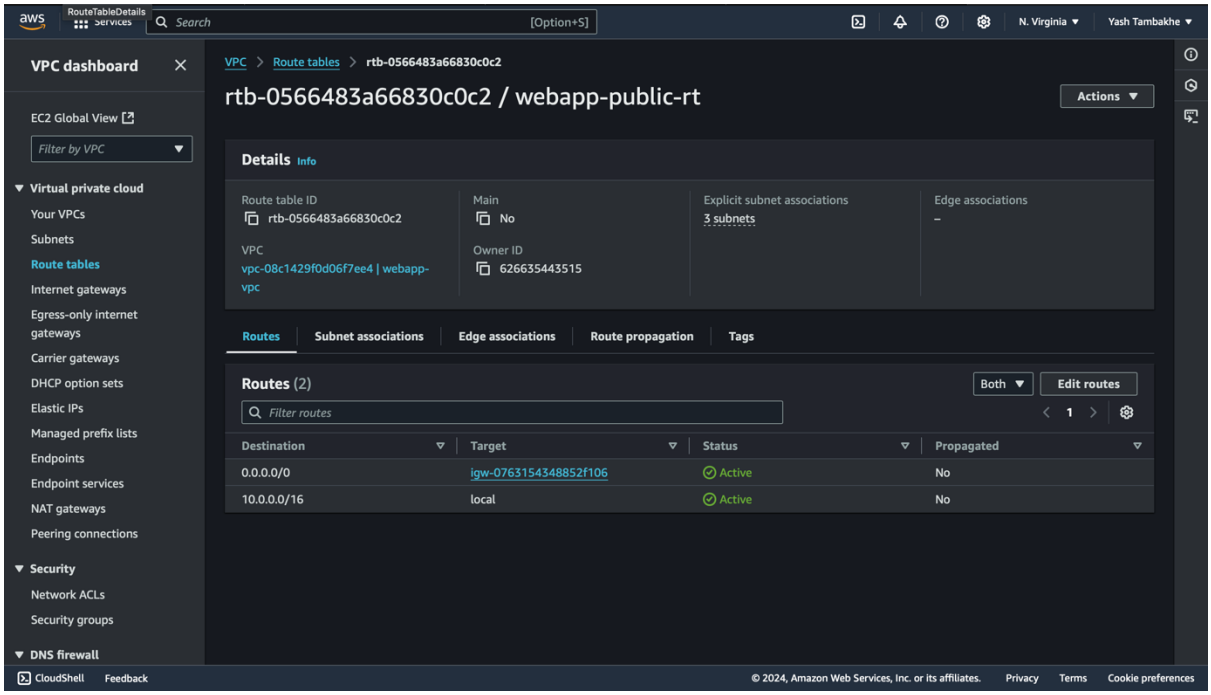
Private Subnet 2



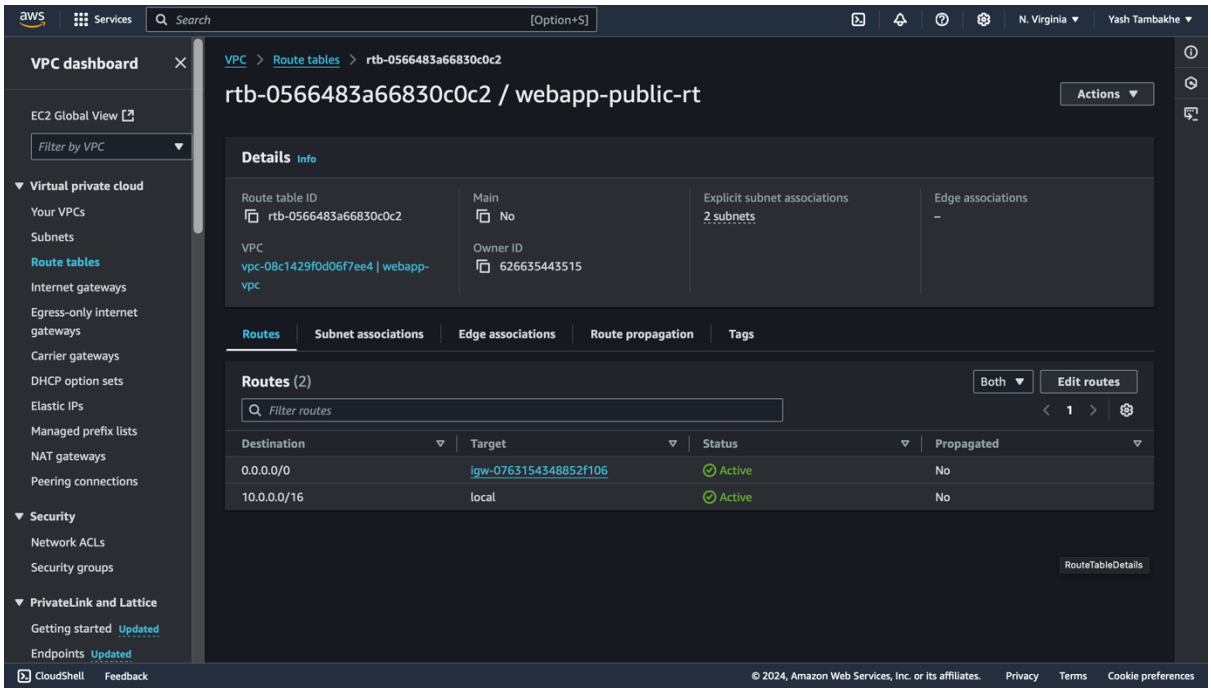
The screenshot shows the Internet Gateway (IGW) attached to the VPC. It displays details such as the IGW ID, its name, and the associated VPC ID, confirming that the IGW is successfully connected to enable internet access for resources within the VPC.



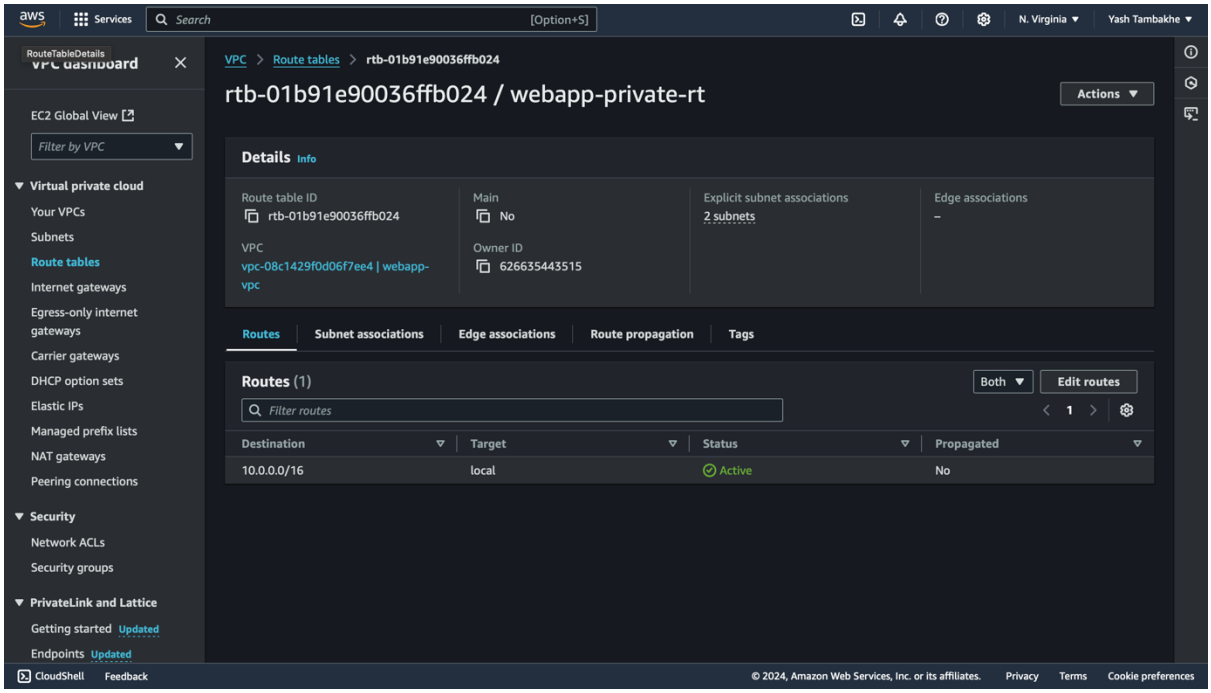
The screenshot shows the public route table associated with the VPC, highlighting the route `0.0.0.0/0` that directs traffic to the Internet Gateway (IGW). This confirms that the route table is configured to allow internet access for resources in the public subnet.



The screenshot shows the public route table associated with the VPC, including the subnet associations. It displays the subnets linked to the route table, confirming that these subnets are configured to use the public route table for internet access through the Internet Gateway (IGW).

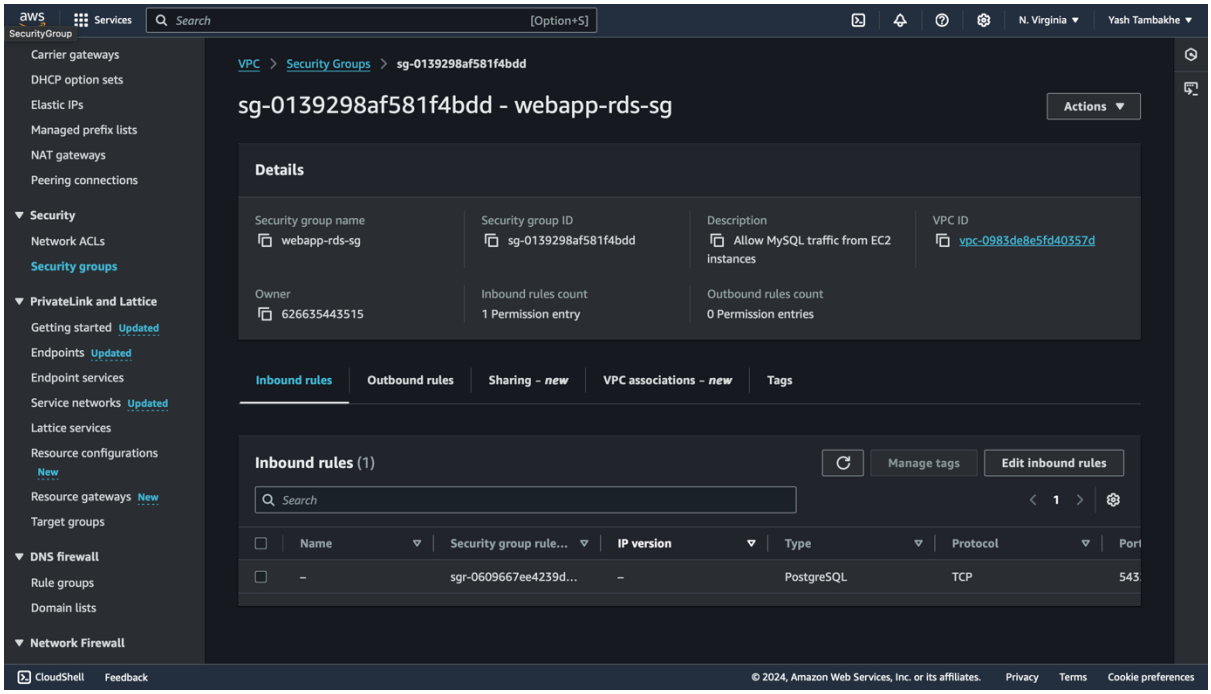


The screenshot shows the private route table associated with the VPC, including its subnet associations. It displays the subnets linked to the private route table, confirming that these subnets are configured for internal communication without direct internet access.



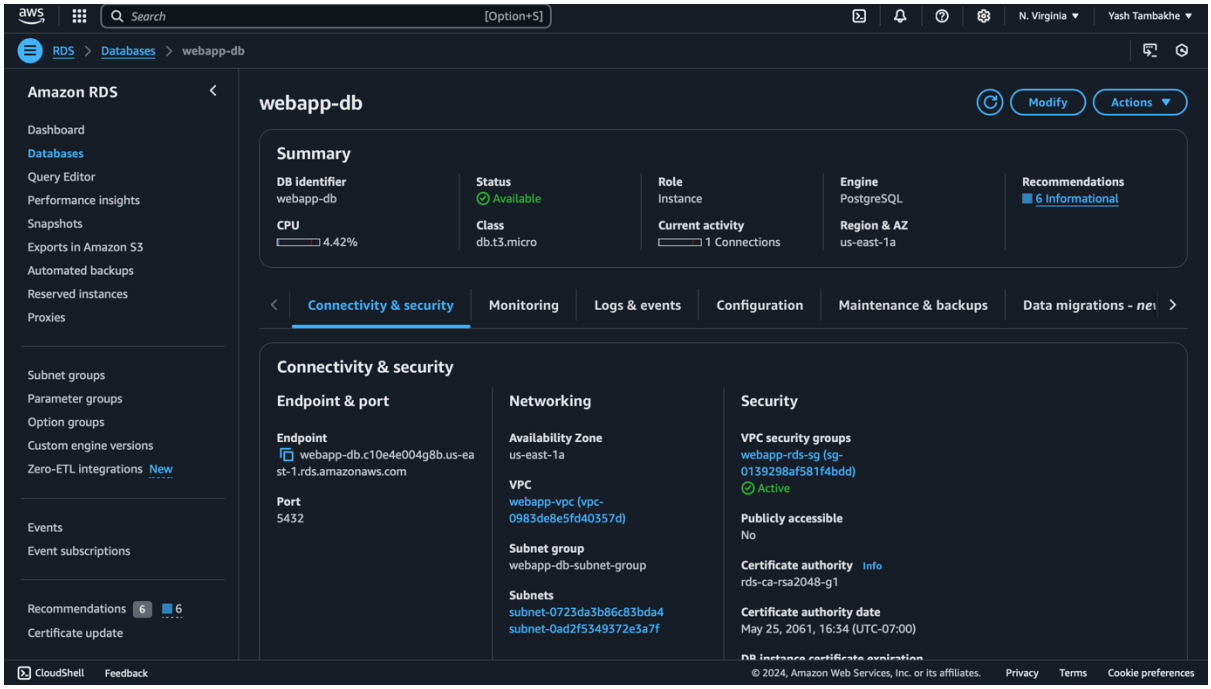
Step 2: Security Groups

The screenshot displays the RDS security group rules, showing inbound configurations to control database access and outgoing traffic.



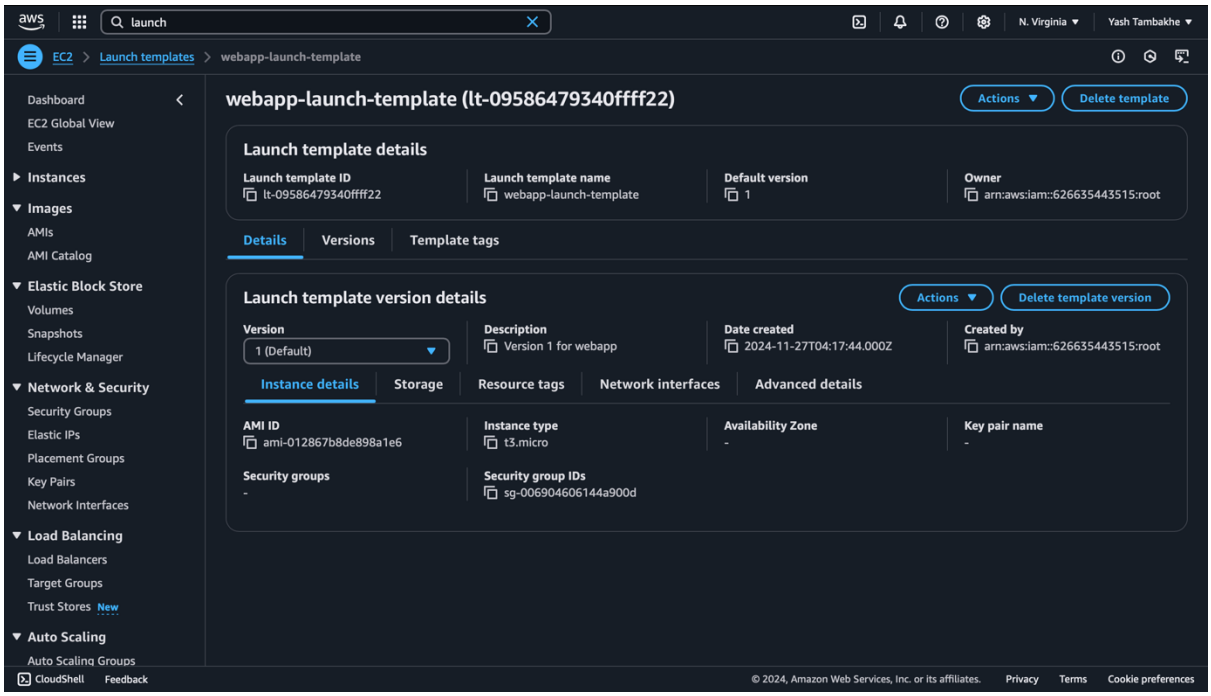
Step 3: Provision RDS Instance for the Web Application

The screenshot shows the RDS instance configuration summary, including details such as the instance class, database engine, and network settings. This confirms the setup of the RDS instance with the specified resources and connectivity configurations



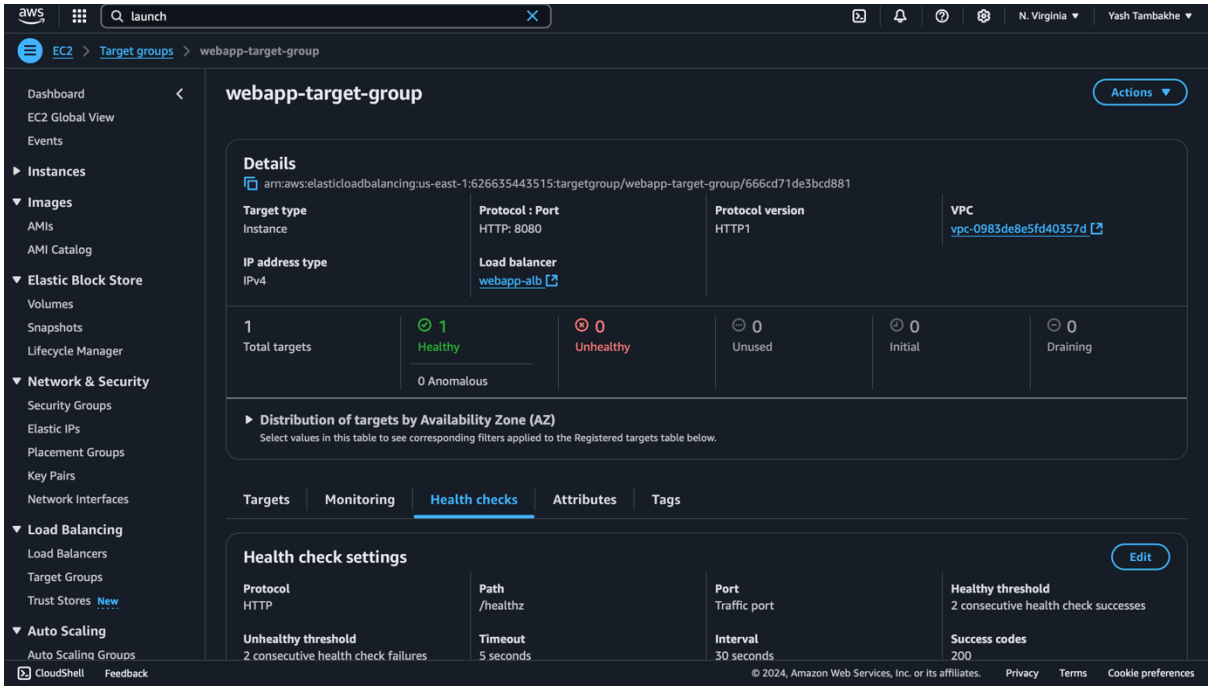
Step 4: EC2 Instances Running the Web Application

The screenshot displays the Launch Template configuration page, showing the selected instance type and the user data script. This confirms the configuration of the template for launching EC2 instances with predefined settings and initialization scripts.

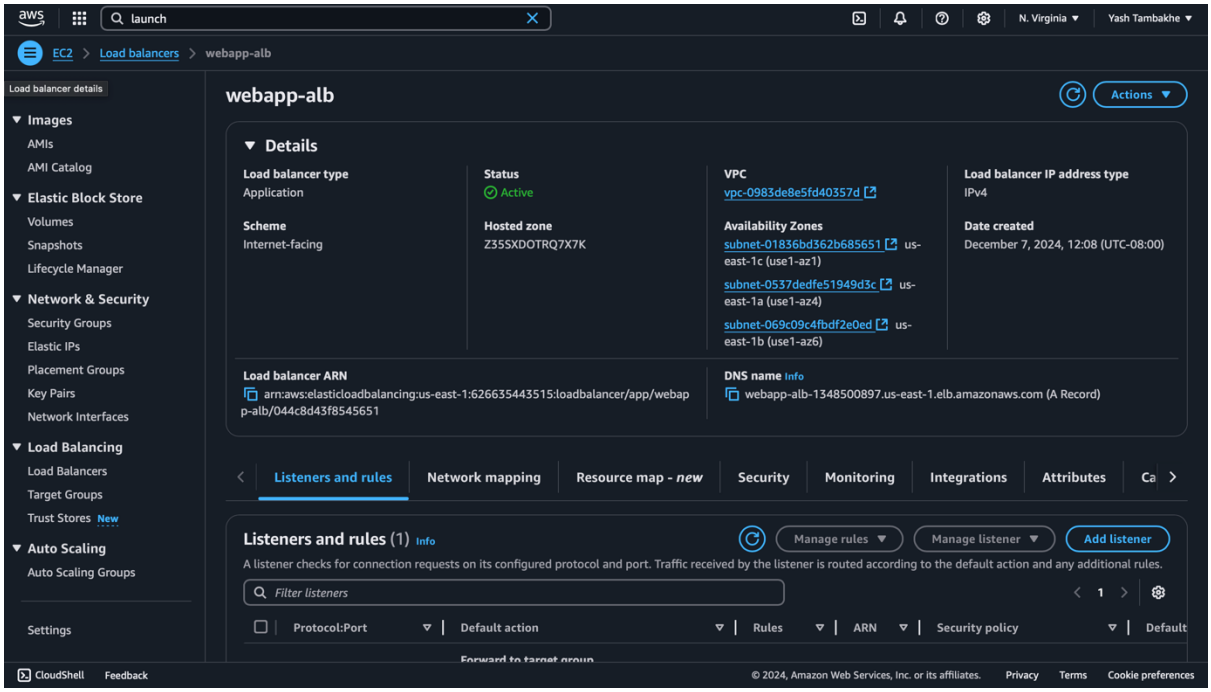


Step 5: Application Load Balancer (ALB) Configuration

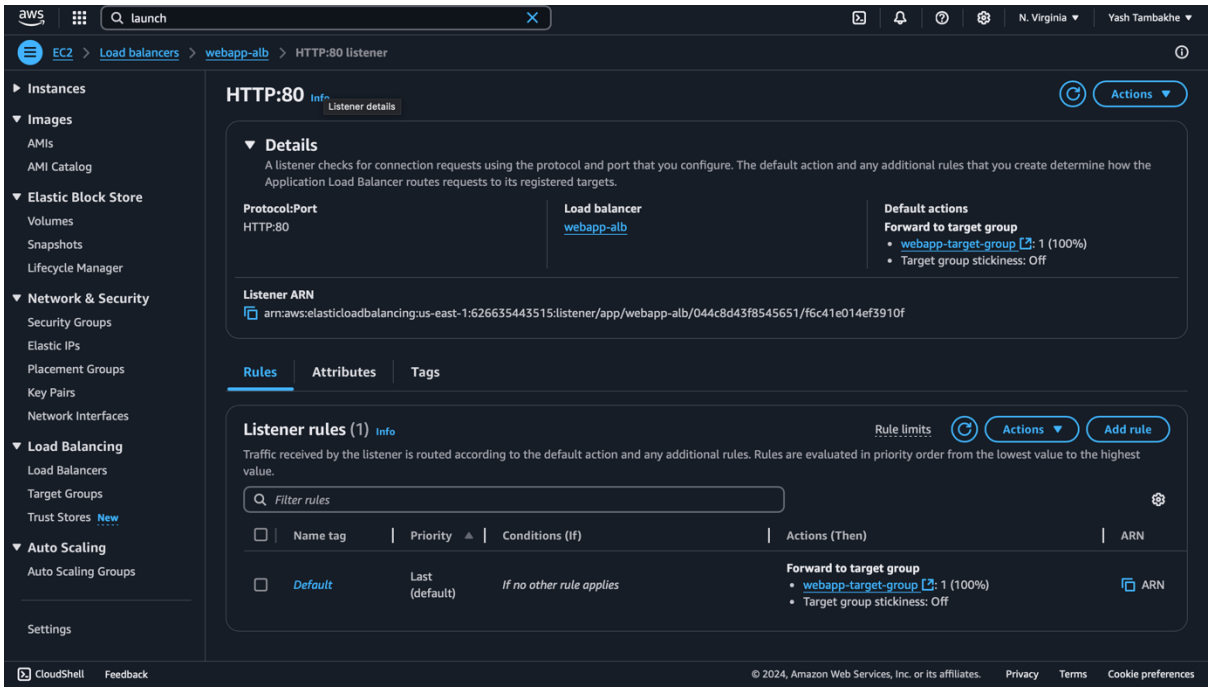
The screenshot shows the target group settings, including the health check configurations. It displays details such as the protocol, port, health check path, and thresholds for healthy and unhealthy targets, confirming the setup to monitor the health of instances in the target group.



The screenshot shows the configuration of the webapp-alb, an internet-facing Application Load Balancer in the webapp-vpc.

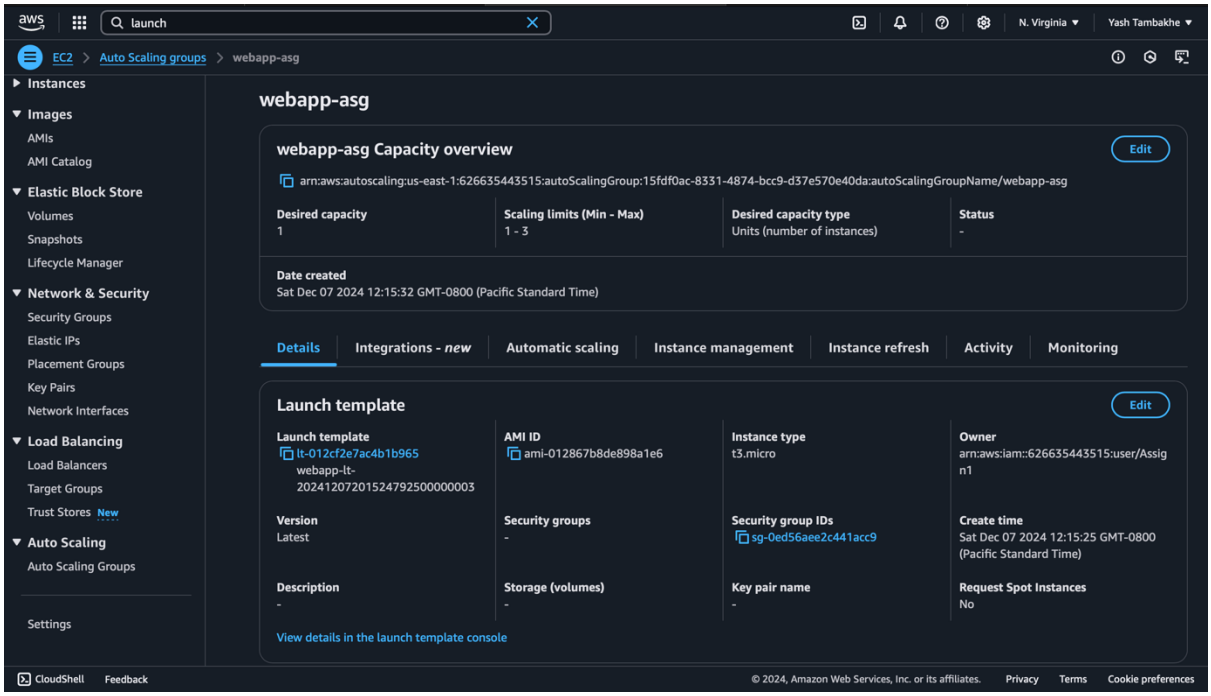


The screenshot shows the Load Balancer listener configuration, where a listener on port 80 is set up to forward HTTP traffic to the designated target group. This setup ensures that incoming web traffic is directed appropriately to the resources in the target group.

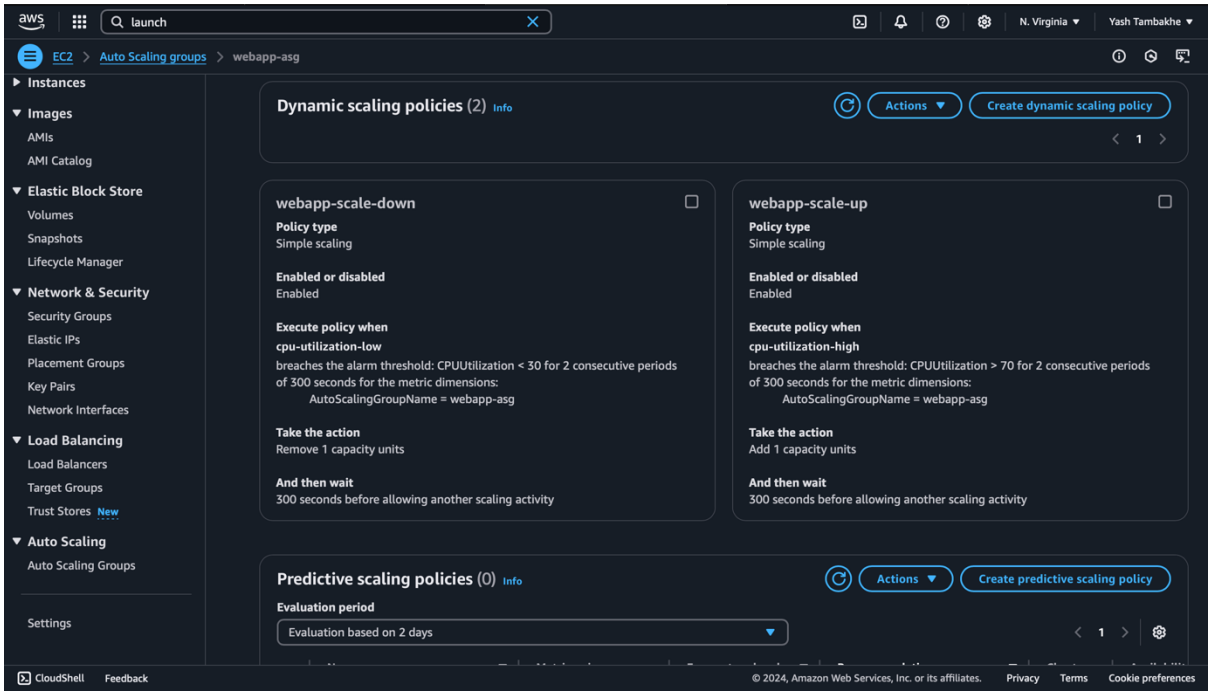


Step 6: Implement Auto Scaling

The screenshot displays the Auto Scaling Group (ASG) settings, including the configured instance counts. It shows details such as the desired, minimum, and maximum number of instances, confirming the ASG's capacity management to maintain optimal resource levels based on demand.

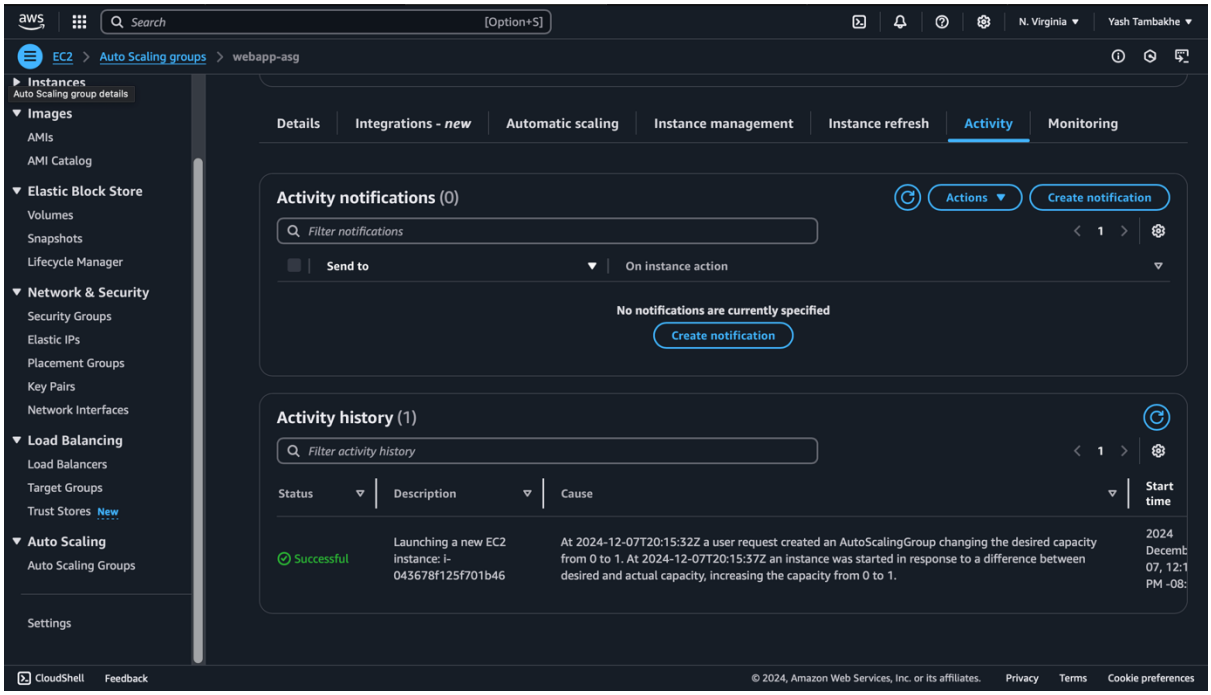


The screenshot shows the CloudWatch alarm configurations used for scaling policies. It details the metrics monitored, such as CPU utilization, and specifies the thresholds that trigger scaling actions, ensuring that resources are adjusted automatically to maintain performance and efficiency.

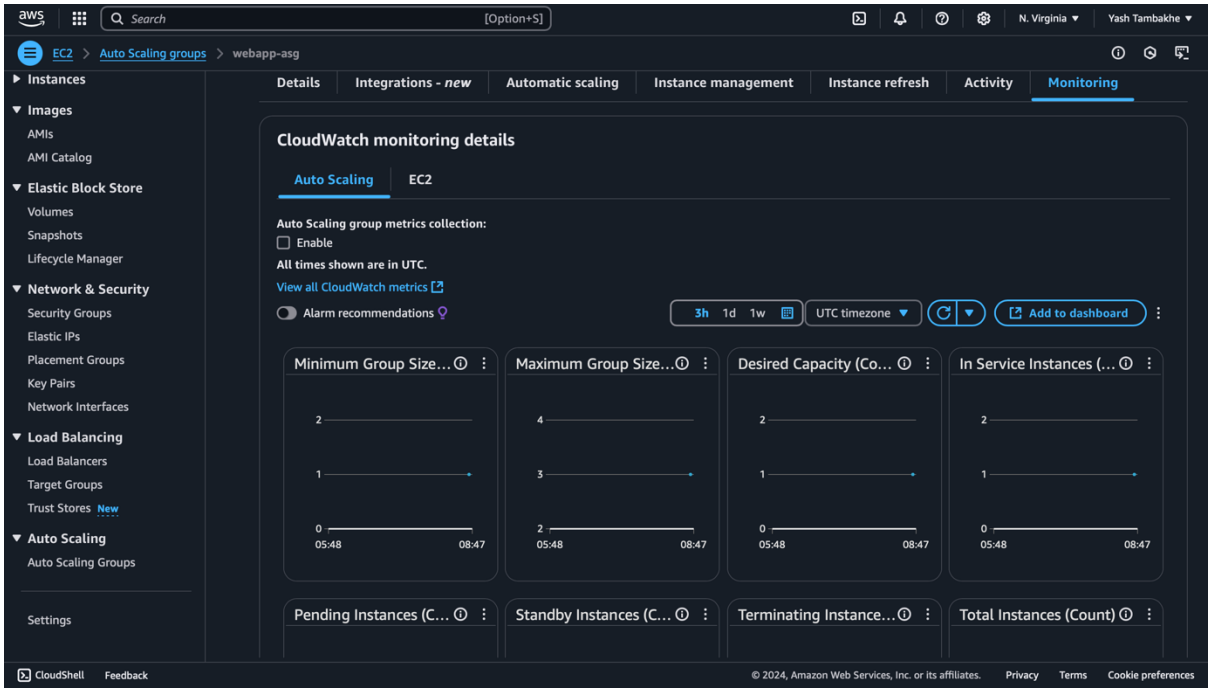


Step 7: Testing and Validation

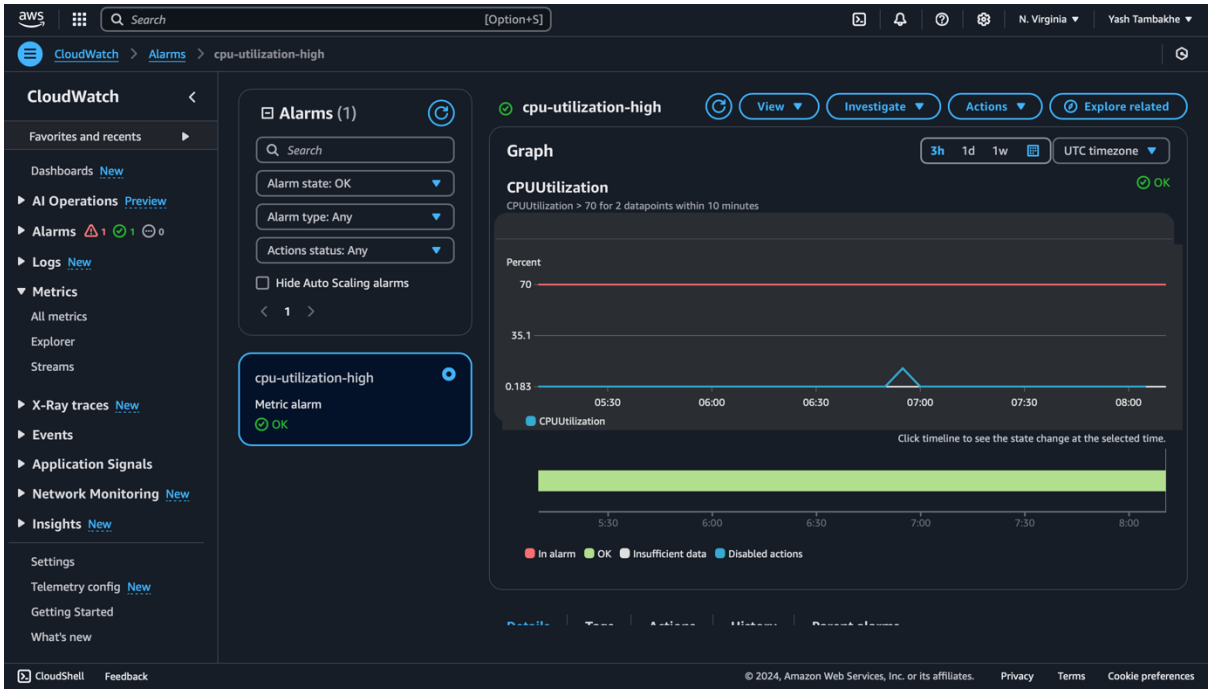
The screenshot captures the scaling activity of instances in the Auto Scaling Group (ASG) during load testing. It shows instances being scaled up to handle increased demand and scaled down when the load decreases, demonstrating the dynamic resource adjustment based on the defined scaling policies.



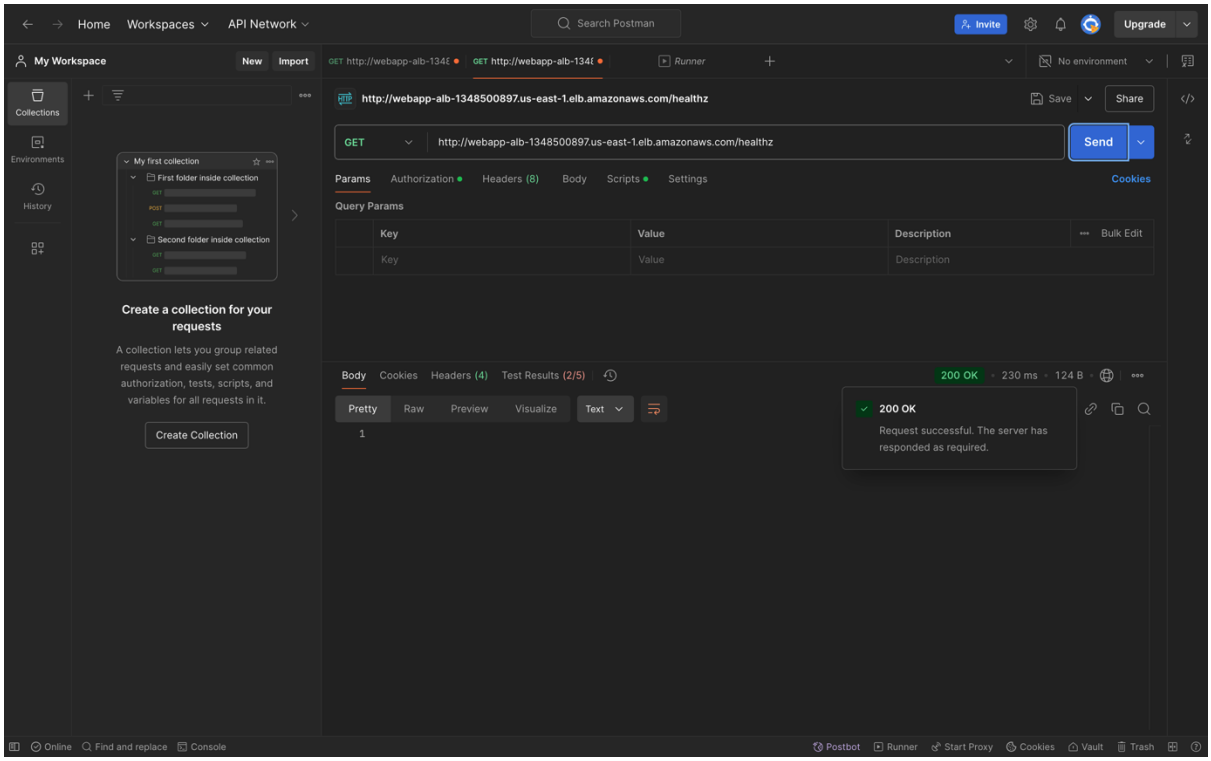
The screenshot displays CloudWatch monitoring details, showing key metrics such as CPU utilization, network traffic, and instance performance over time.



The screenshot shows the Amazon CloudWatch Alarms console monitoring a metric alarm named cpu-utilization-high, which tracks CPU utilization exceeding 70% for 2 data points within 10 minutes. The alarm is currently in the OK state, with a graph displaying CPU usage below the threshold.



The screenshot shows a GET request to the /healthz endpoint, which returned a 200 OK response. This indicates that the server is running and healthy.



The screenshot shows a successful POST request to create a user. The server responded with a 201 Created status, and the response body contains the user details in JSON format, confirming that the user was created successfully.