**CASE STUDY 2:**

**Problem Statement:** You have been asked to set up a Blue/Green (B/G) solution for a mission-critical web application running on AWS. It should consist of **two separate environments**. The blue environment contains Amazon EC2 instances in an Auto Scaling group that runs the current production version of the application. The green environment contains EC2 instances in another Auto Scaling group that runs the new version of the application. Each Auto Scaling group is behind its own Application Load Balancer (ALB), so you can configure **two alias records as endpoints** in Amazon Route 53 and use a **routing policy** to gradually shift traffic from the ALB for the blue environment to the ALB for the green environment. Route 53 endpoint must be accessible from the internet through a domain name and **HTTPS protocol only**. Setup this combination to achieve B/G deployment and explain how different **backend database endpoints** will be handled in this situation. Use proper tool to build an infrastructure as code (IaC) solution.

SOLUTION:

Blue/Green deployment is used to minimize downtime and risks during software updates. It works by running two identical environments: the "Blue" represents the current live version, while the "Green" hosts the new version. Traffic is initially directed to the Blue environment, and after successful testing, it is switched to Green. This method allows seamless rollbacks if issues arise, ensuring smooth application updates without disrupting users.

Here’s an explanation of each block in your Terraform script:

**Provider Block**  
Defines AWS as the cloud provider and sets the region dynamically using a variable.

**Security Group**  
Creates a security group named "web-sg" for EC2 instances, allowing HTTP (80) and SSH (22) access while enabling outbound traffic.

**ALB Target Groups**  
Defines two target groups, "blue" and "green," for the load balancer, each handling HTTP traffic on port 80.

**Application Load Balancer (ALB)**  
Creates an ALB that distributes traffic across instances. It’s linked to public subnets and a security group.

**Listener for HTTPS**  
Sets up an HTTPS listener on ALB using an SSL certificate, forwarding traffic to the Blue target group by default.

**EC2 Instances - Blue**  
Launches two EC2 instances in public subnets, installs Apache, and serves a website from a downloaded template.

**Target Group Attachment - Blue**  
Associates Blue EC2 instances with the Blue target group for load balancing.

**EC2 Instances - Green**  
Launches two EC2 instances similar to Blue, but with a different website template.

**Target Group Attachment - Green**  
Associates Green EC2 instances with the Green target group.

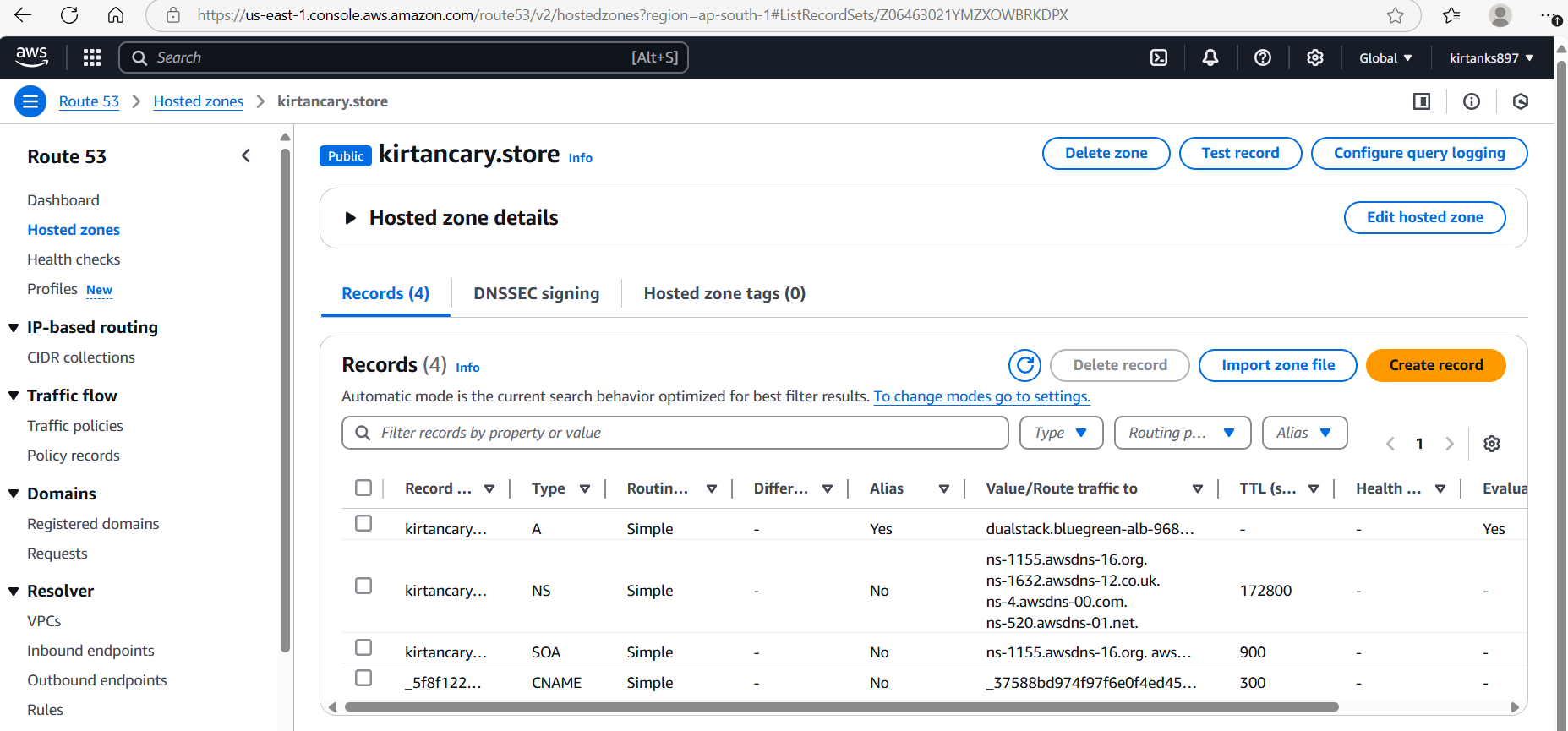
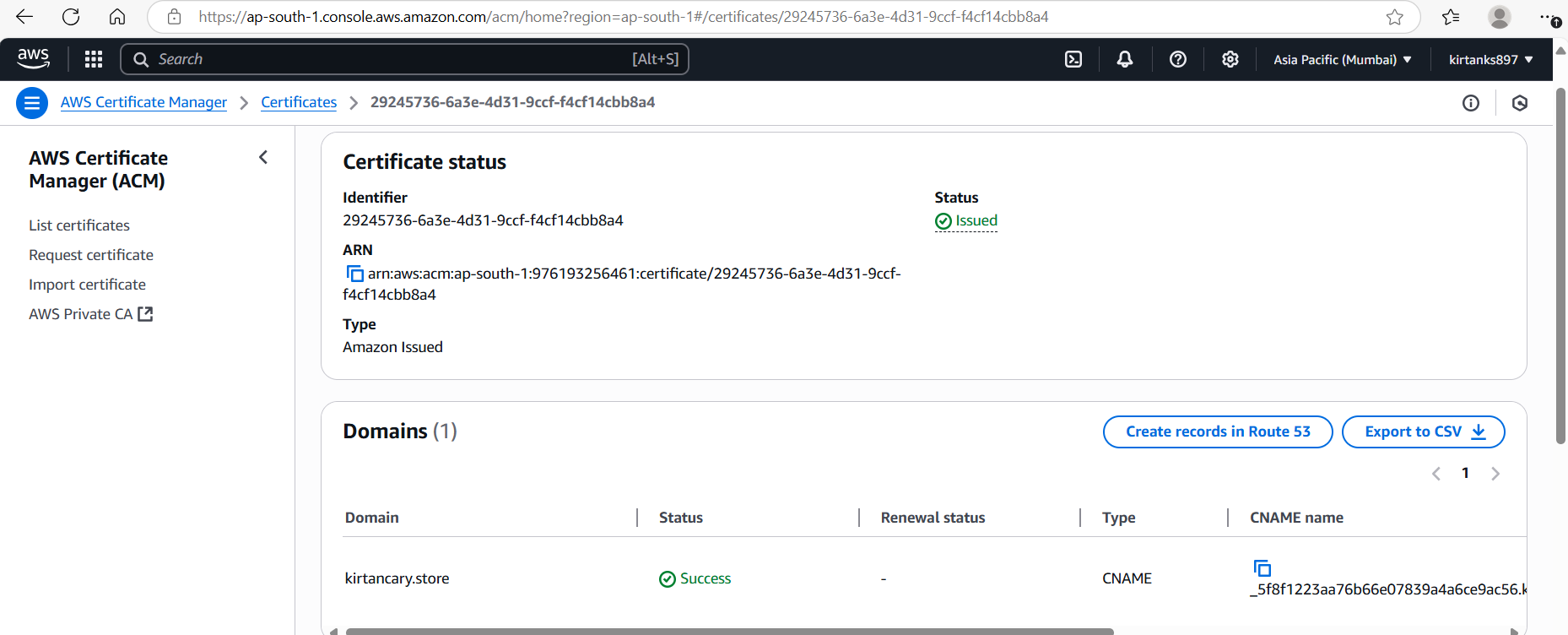
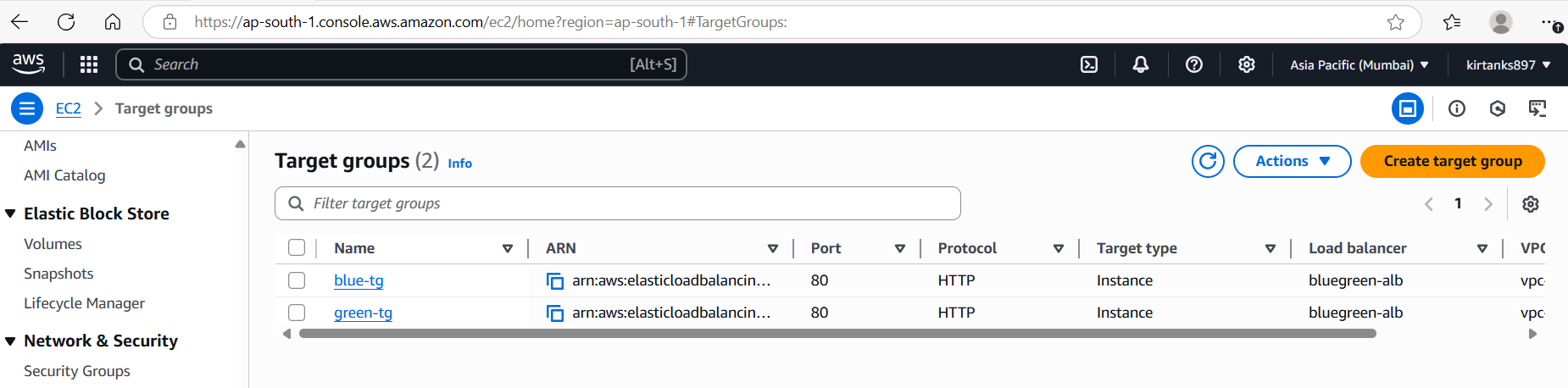
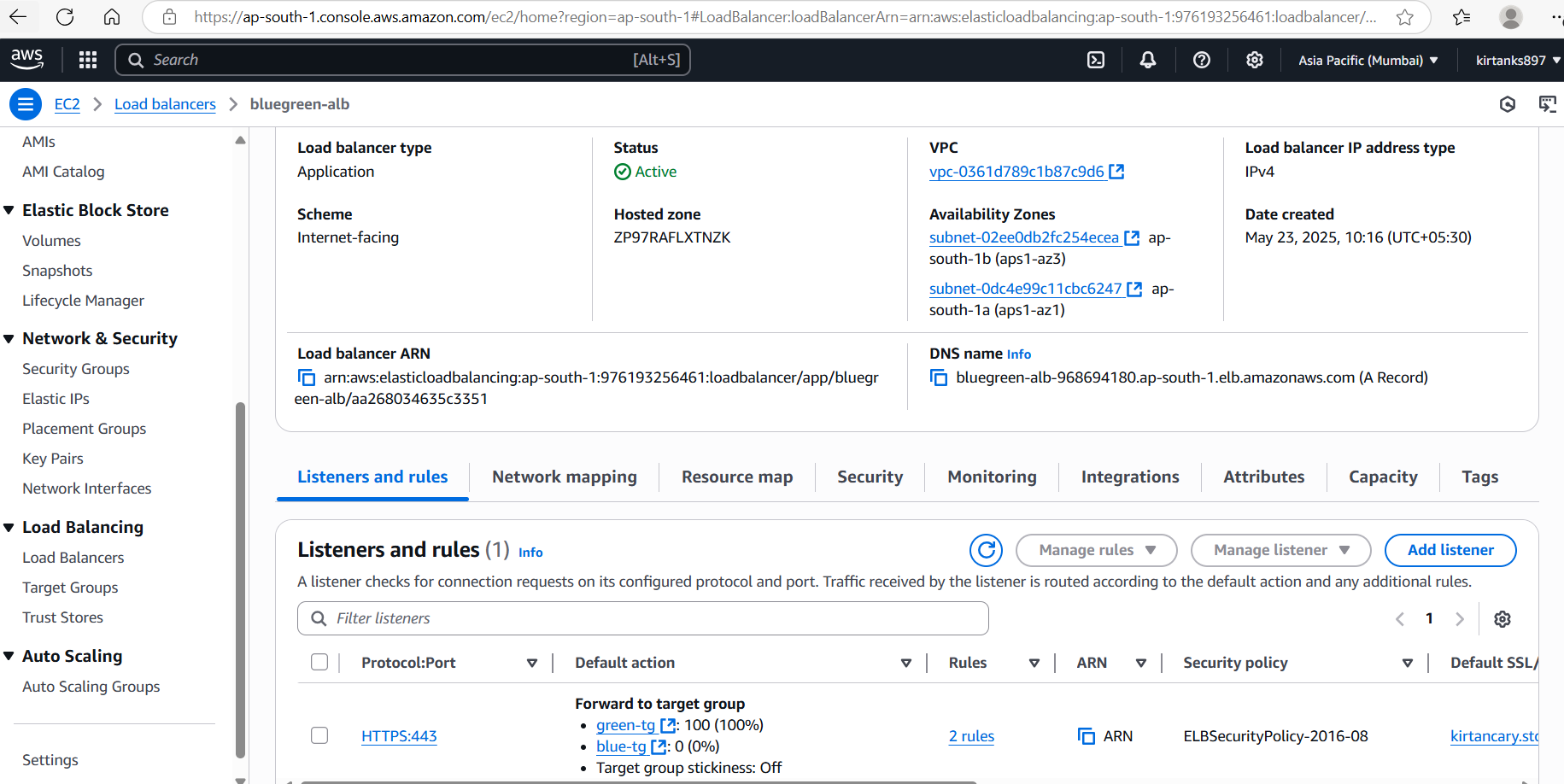
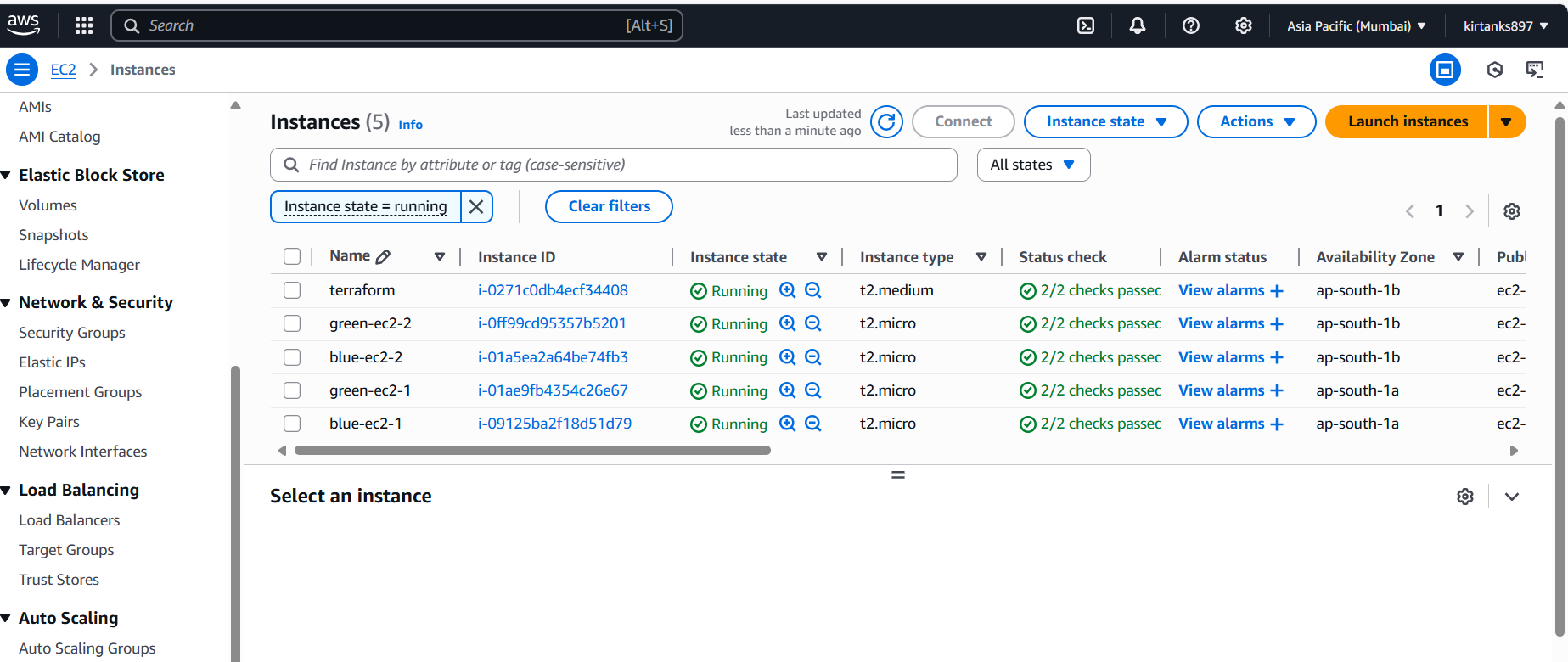
**Listener Rule for Green Path**  
Defines a condition to forward traffic to the Green target group when requests match the "/green/\*" path.

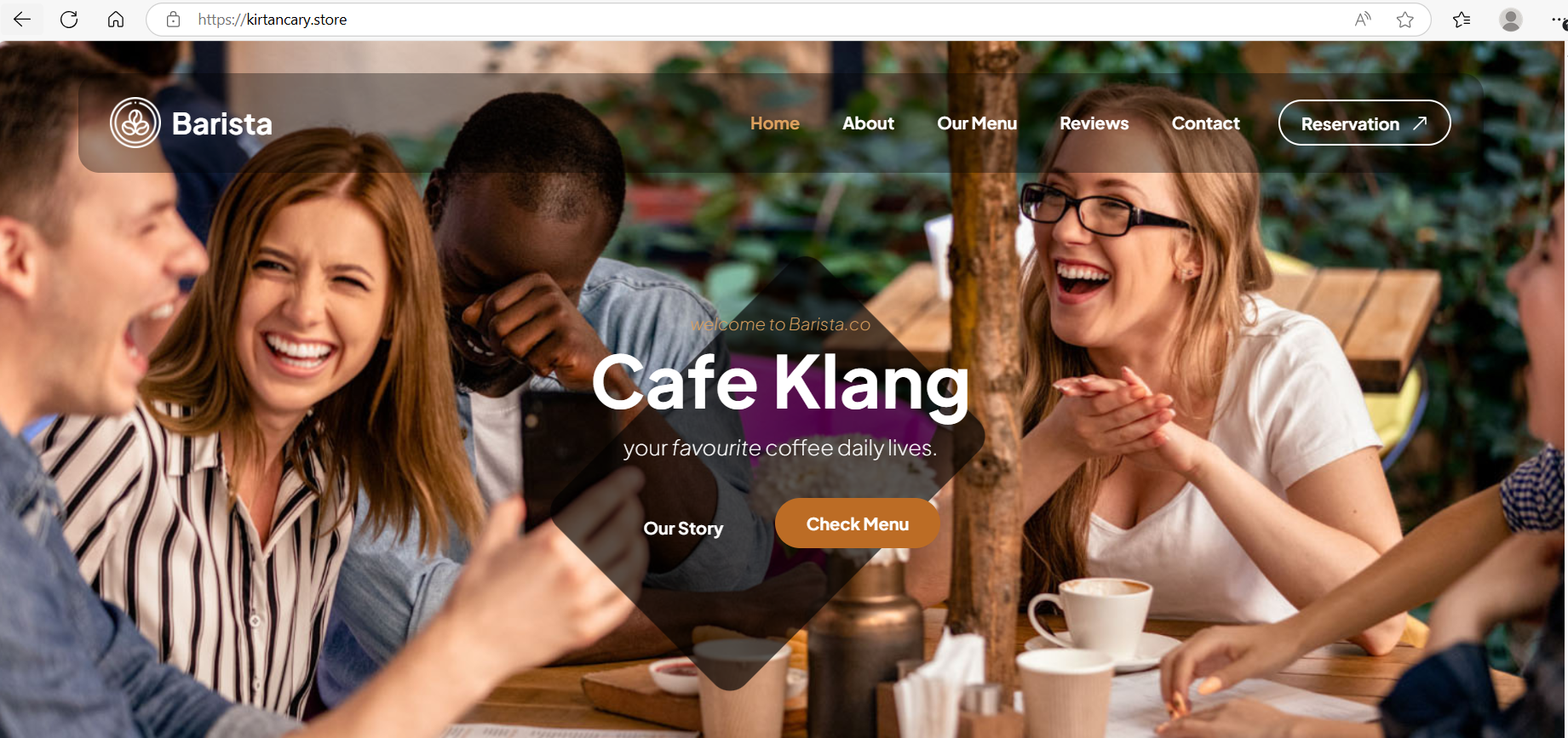
**ACM and route53 hosted zone**

ACM is used to issue an SSL certificate, enabling secure HTTPS connections for the application. The HTTPS listener in ALB uses this certificate and forwards traffic to the Blue target group by default.

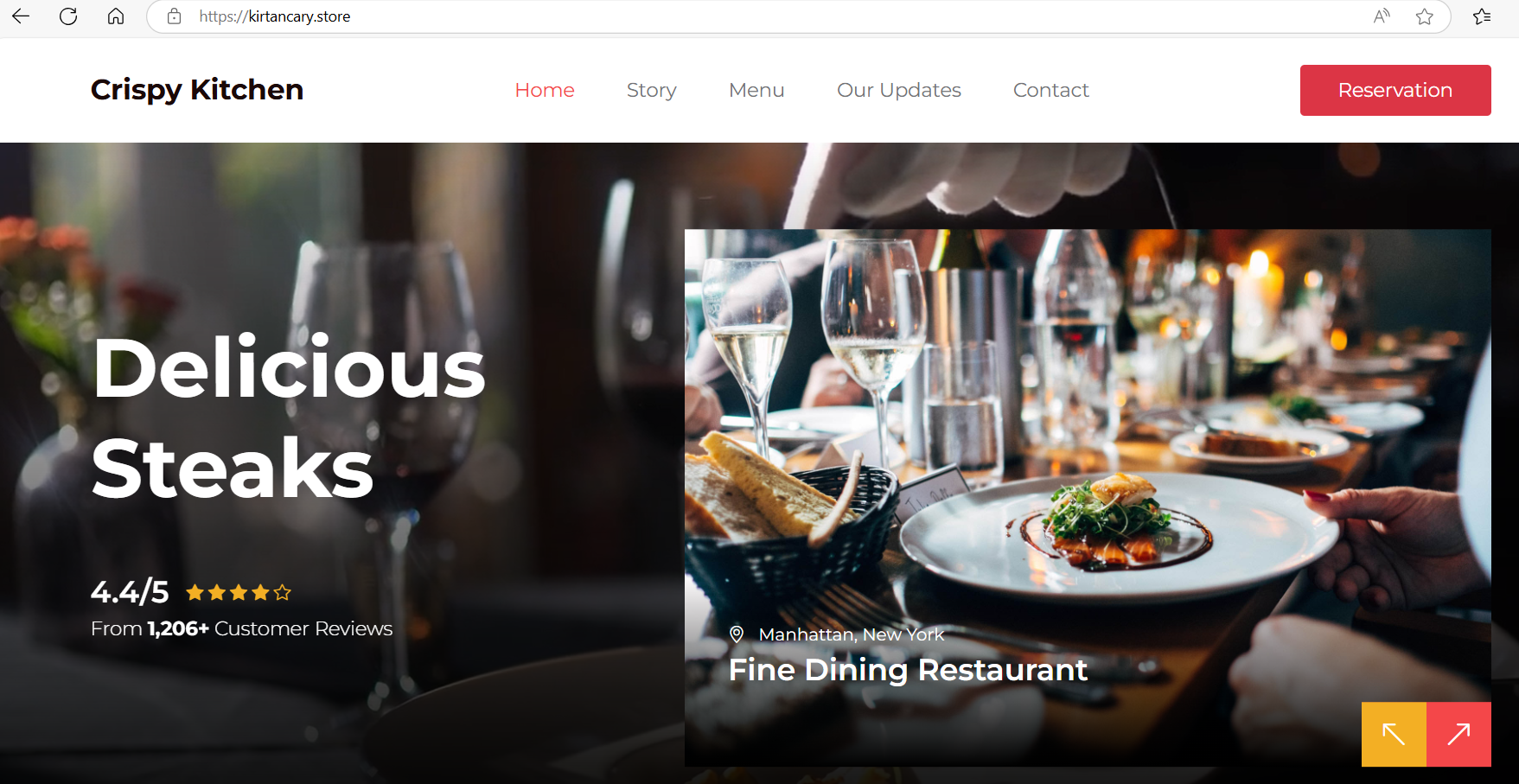
Route 53 is then configured with a hosted zone to manage DNS records, ensuring users can access the application with a custom domain.

**SCREENSHOTS:**





**Blue Environment**



**Green environment**