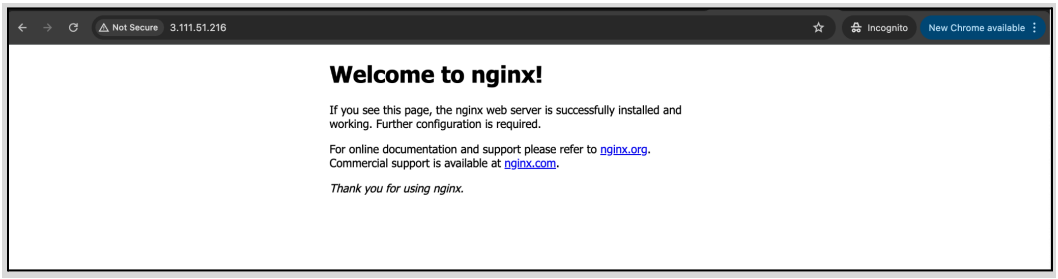
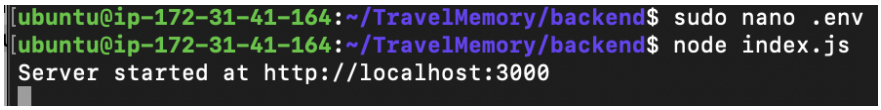
Travel Memory Application Deployment

- **Instance Configuration**
  1. Update the package management tool (apt)
     sudo apt update
     Sudo apt-get update
  2. Install the nginx server
     sudo apt install nginx



  3. Install the nodejs
     curl -sL https://deb.nodesource.com/setup_18.x -o/tmp/nodesource_setup.sh
     sudo bash /tmp/nodesource_setup.sh
     sudo apt install nodejs

- **Backend Configuration**
  1. Clone the “TravelMemory” project from git.
     git clone https://github.com/UnpredictablePrashant/TravelMemory
  2. Navigate to the Backend directory of the project and create .env file
     cd TravelMemory/backend
     sudo nano .env
  3. Add the URI and PORT details in .env file and save it.
     MONGO_URI='mongodb+srv://Yash:<password>@cluster0.swmtw8d.mongodb.net/?retryWrites=true&w=majority&appName=Cluster0'
     PORT=3000
  4. Install the npm libraries and run the backend.
     npm install
     node index.js
  5. Backend started running at port 3001.



  6. Edit Security group rules to allow port 3000 for public access.

- **Frontend Configurations**
    1. Navigate to the src/frontend Directory in TravelMemory app
       `cd TravelMemory/frontend/src`
    2. Open the url.js file and provide the backend url.
       `http://<ip address>:3000`
       Note:- replace ip address with your instance ip address

       ```
       export const baseUrl = "http://13.208.164.76:3000"
       ```
    3. Edit Security group rules to allow port 3001 for public access of the website.

    4. Install the npm libraries and run the Frontend.
       `cd ..`
       `npm install`
       `npm start`
    5. Frontend started at port 3000

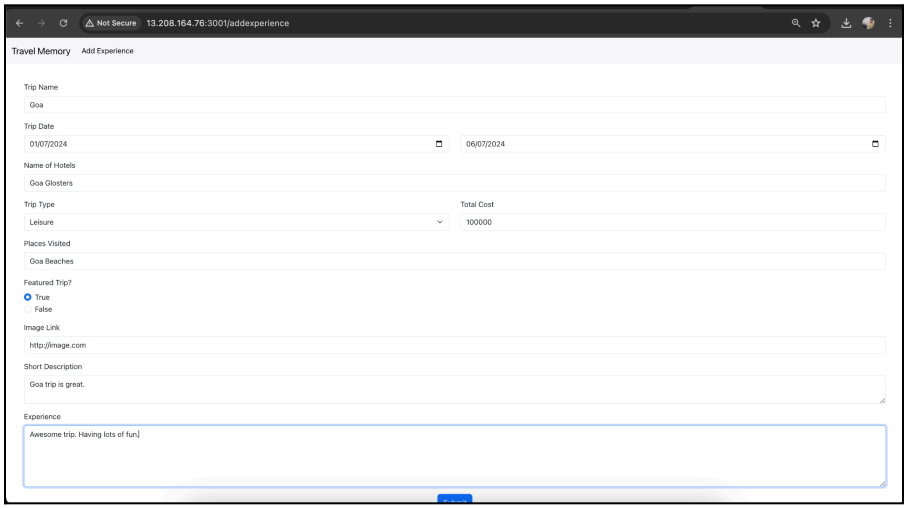
       

- **Configuring the path forwarding**
    1. Navigate to sites-available path in nginx and add the proxy pass details in the default named file.
       `sudo nano /etc/nginx/sites-available/default`

```
        server {
                listen 80 default_server;
                listen [::]:80 default_server;
                server_name _;
                location / {
                        #try_files $uri $uri/ =404;
                        proxy_pass http://localhost:3001;
                        proxy_http_version 1.1;
                        proxy_set_header Upgrade $http_upgrade;
                        proxy_set_header Connection "upgrade";
                        proxy_set_header Host $host;
                        proxy_cache_bypass $http_upgrade;
                }
        }
```

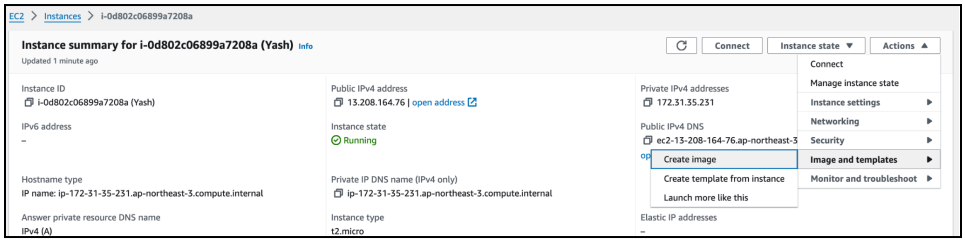
2. Delete the default file in sites-enabled and create a soft link of default file in sites-available.
   `sudo rm -rf /etc/sites-enabled/default`
   `sudo ln -s /etc/nginx/sites-available/default /etc/sites-enabled/default`

- **Full Application Working**
  1. Open the port 3001 of your app and click on Add Experience and fill the form.

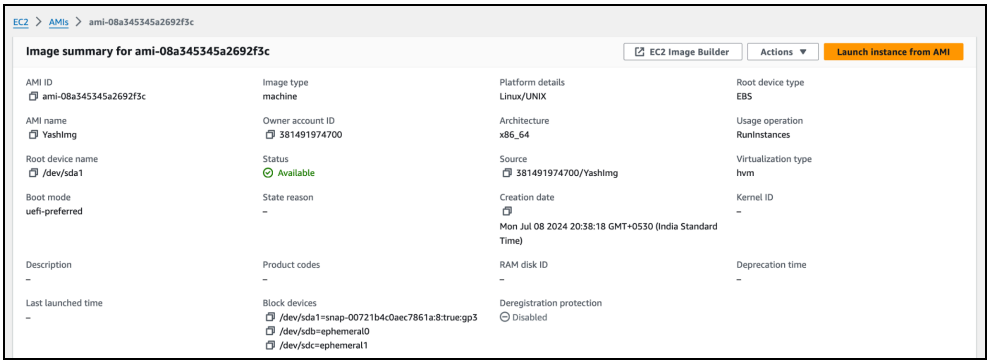


  2. Navigate to the Home Page of your App and you can see a card created for your trip.

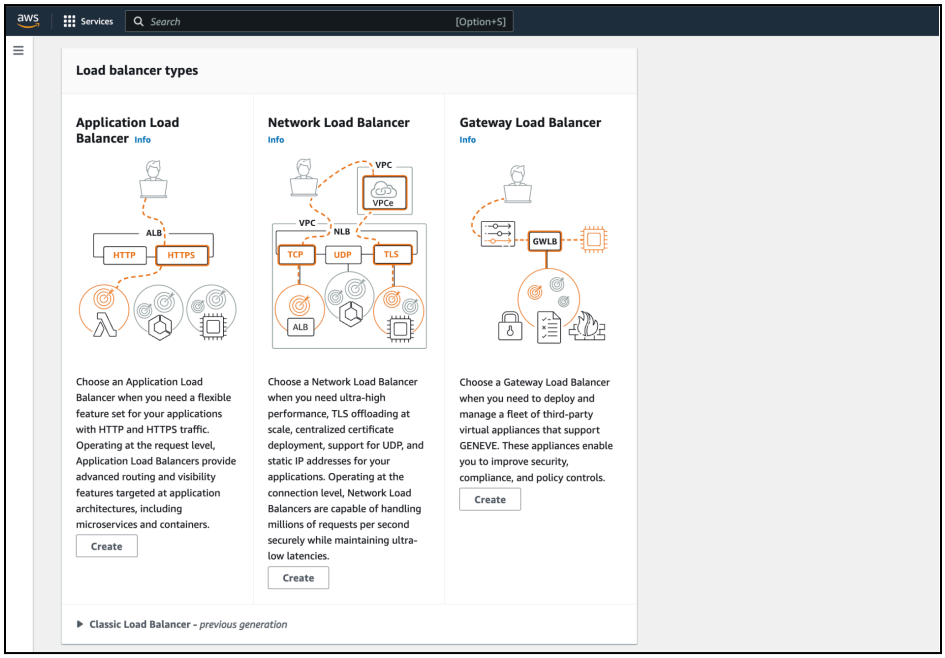
- **Load Balancer**
  1. Create the AMI image of the running Instance.



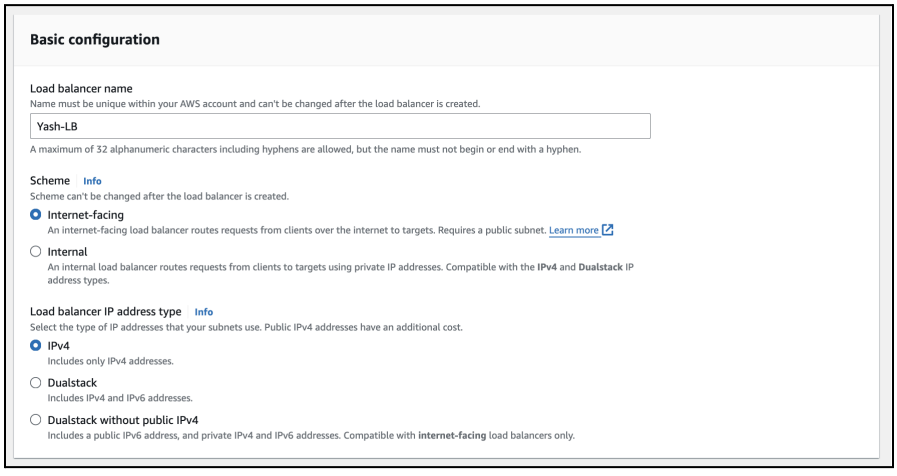
  2. Provide Image name and keep everything as default.

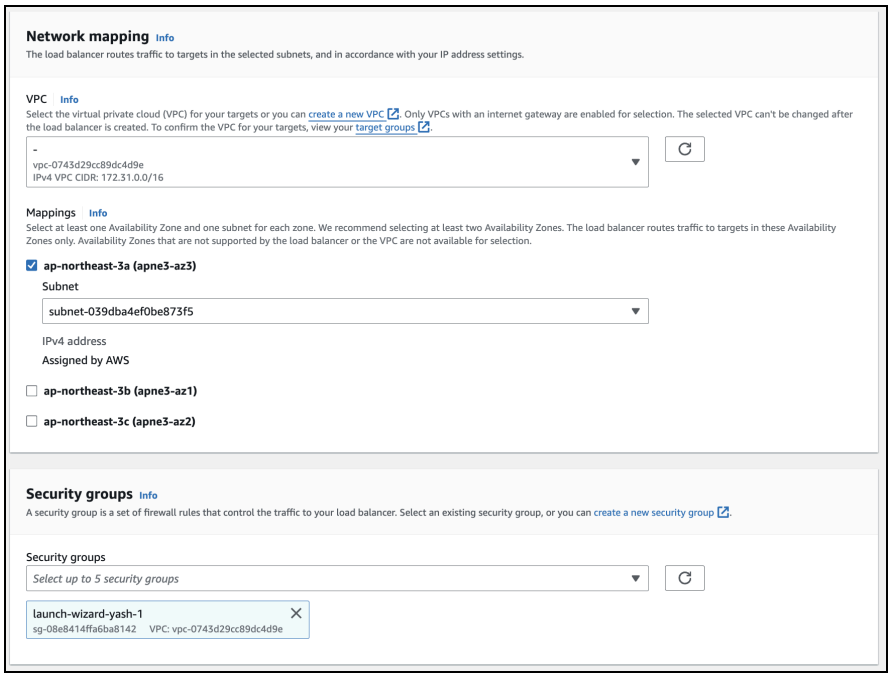


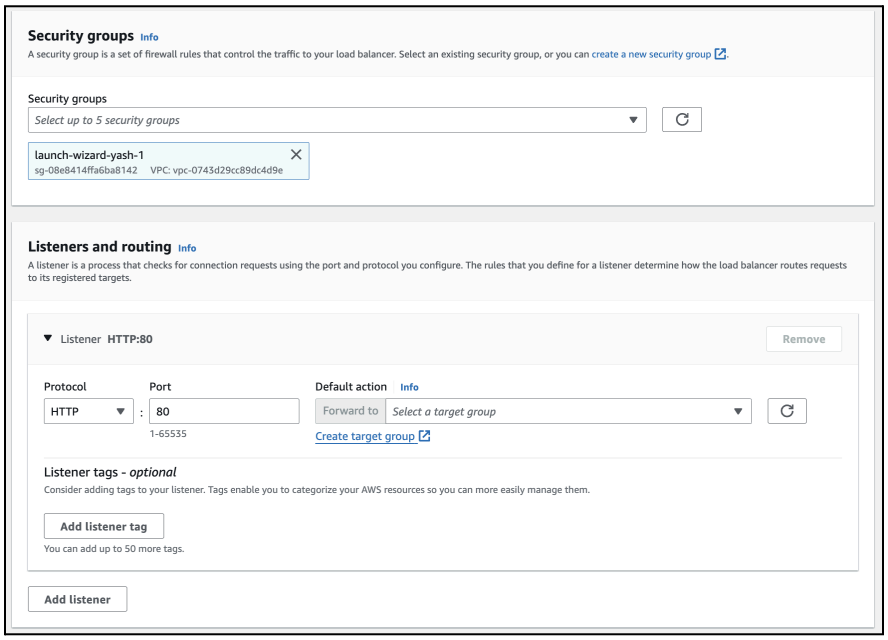
3. Create a new Application Load Balancer



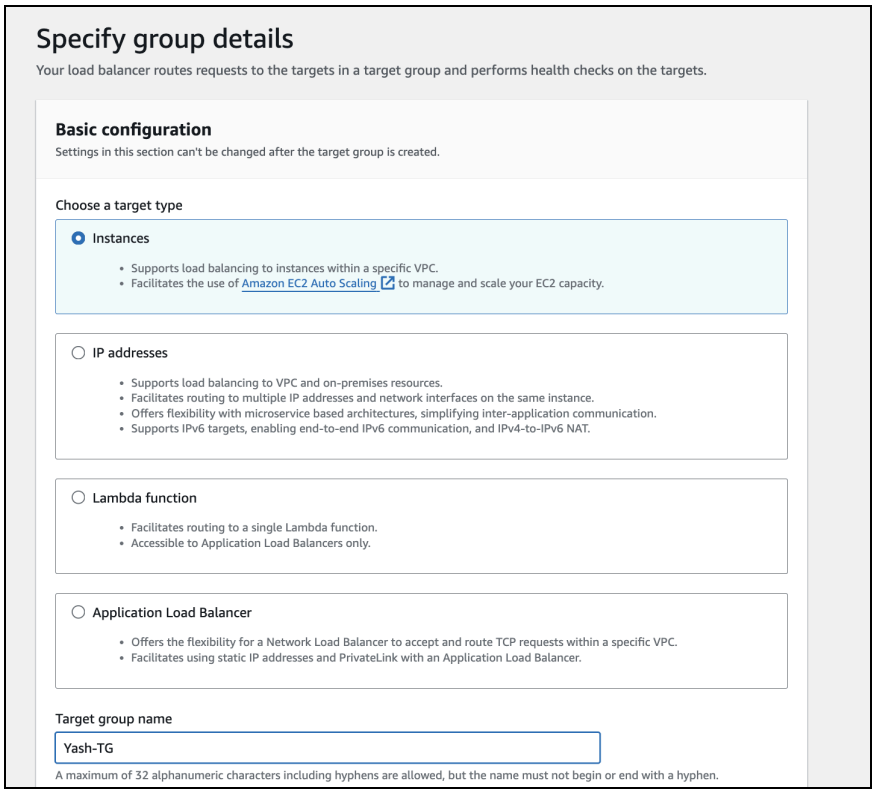
4. Provide a unique name and select the configurations as per below Screenshots.



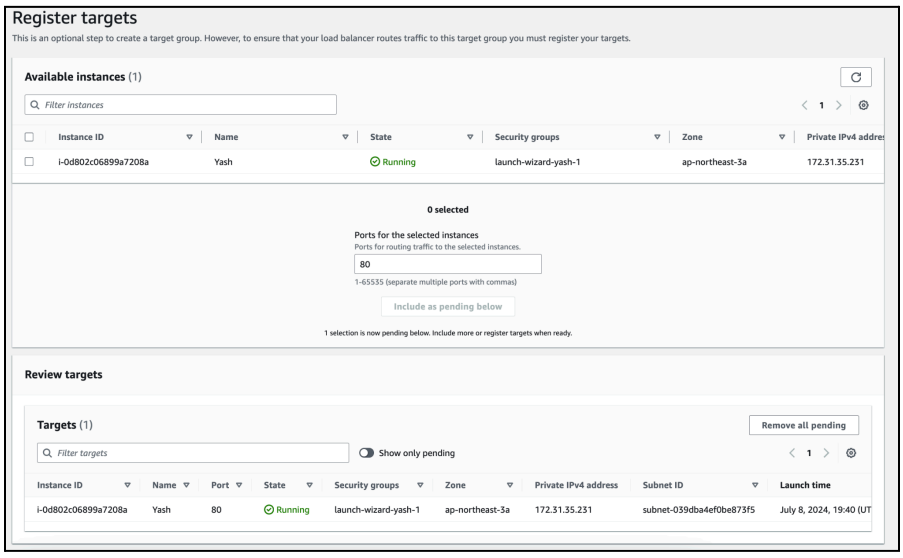




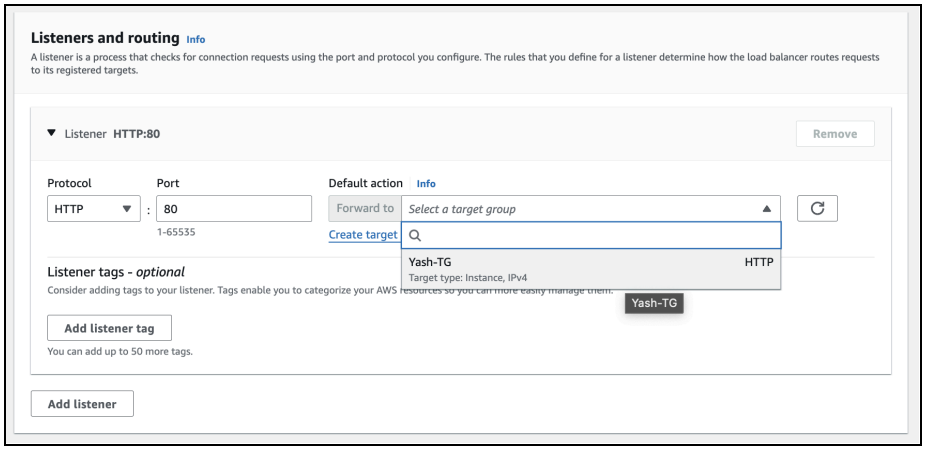
5. Select create new Target Group and choose Instance type and provide a unique name and click on next.



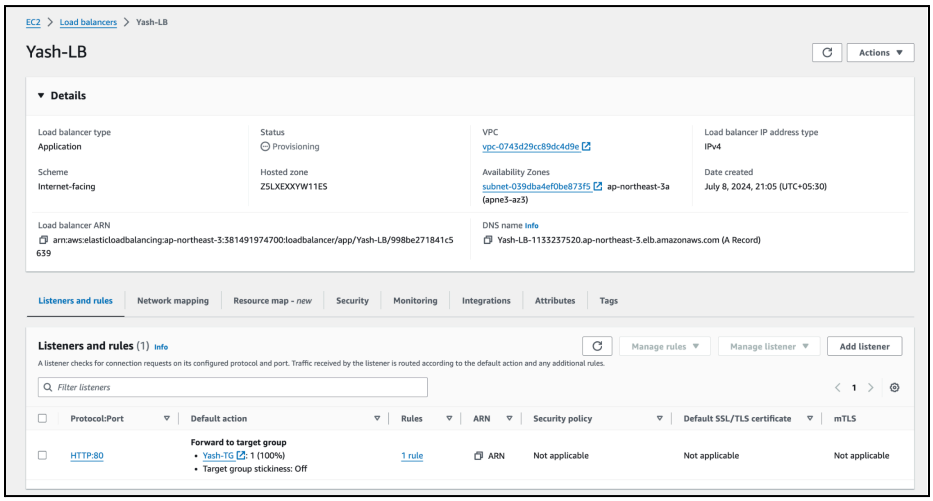
6. Select the instance in Register targets and click on “include as pending below” then click on create target group.



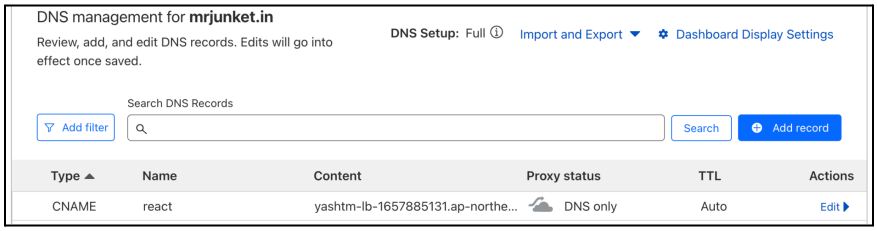
7. Select the Target group you created in listeners and routing then click on create load balancer.



8. Load Balancer successfully created.

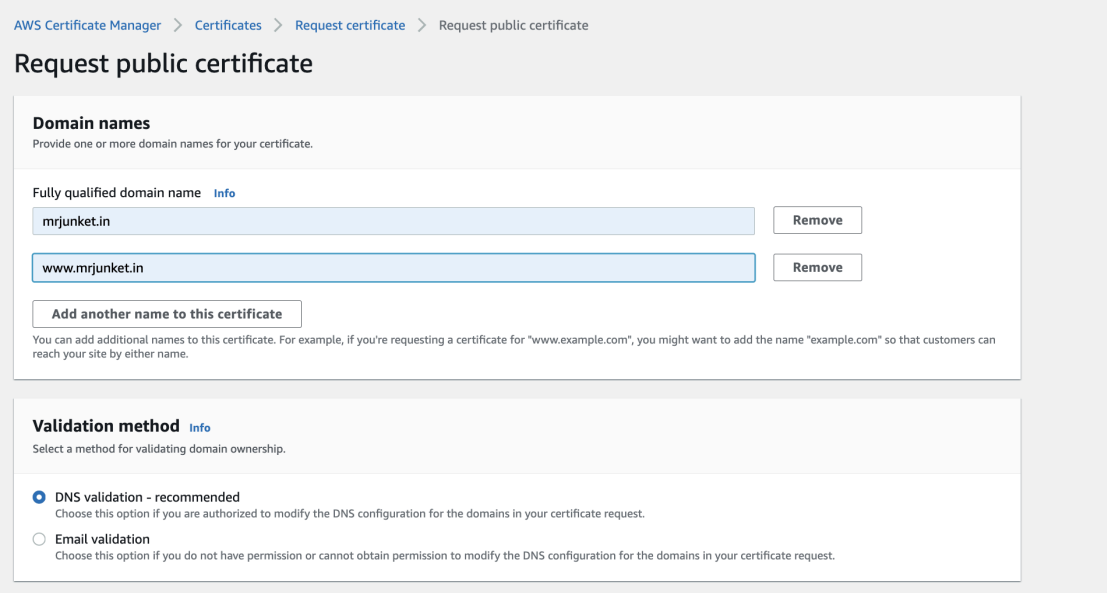


- **Configuring Domain Setup**
  1. Created a Cname record and added the DNS name from load balancer in content as shown below.

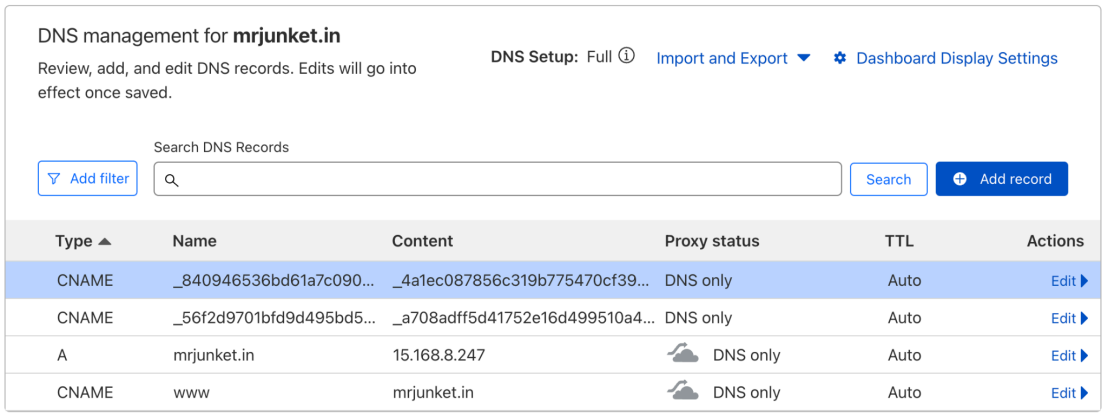


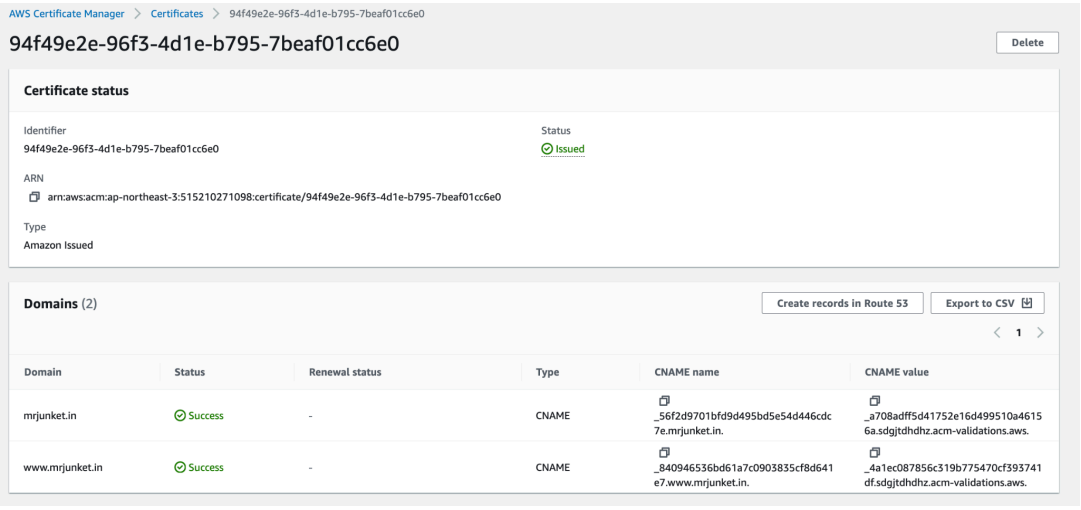
2. Open the react.mrjunket.in and you will see the application working.

- **Adding SSL Certificate to Load Balancer**
    1. Go to ACM (Amazon ACM) and request a new public certificate. Puth the Domain names as below and rest leave everything default.

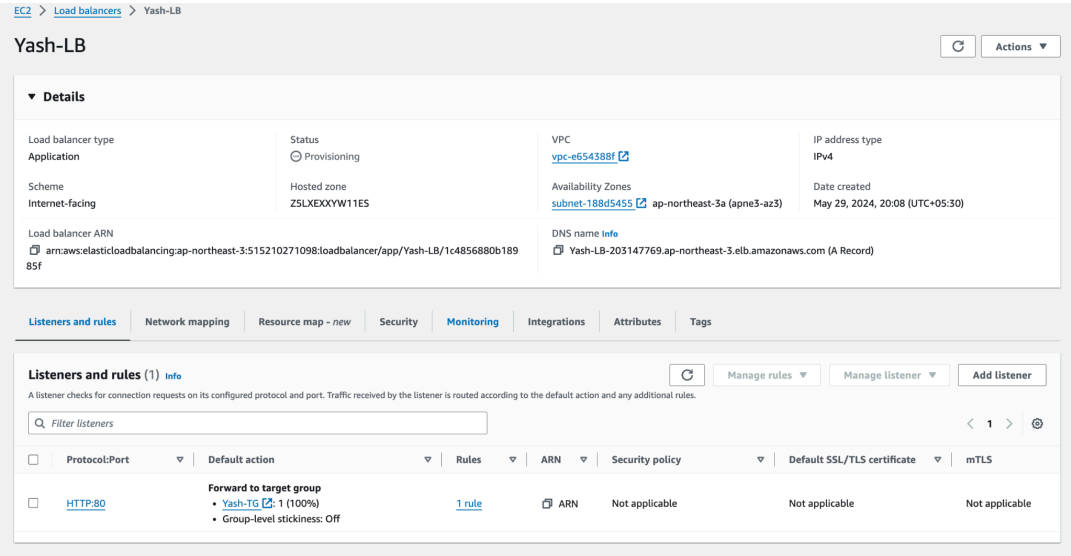


    2. Open the certificate issued to you and copy the cname record key value pair to cloudflare.

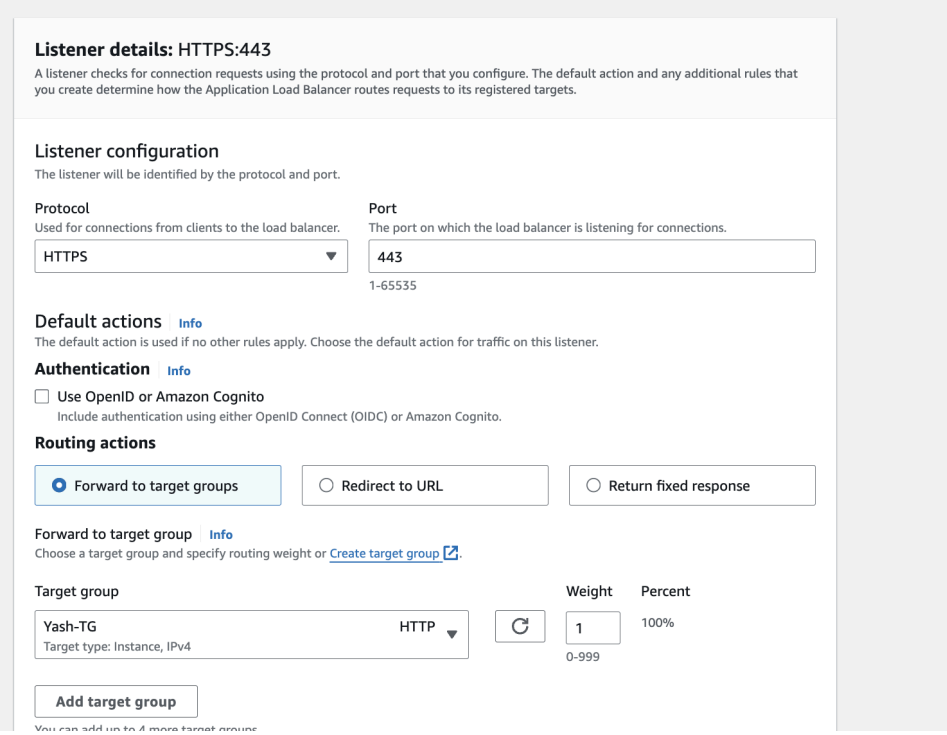




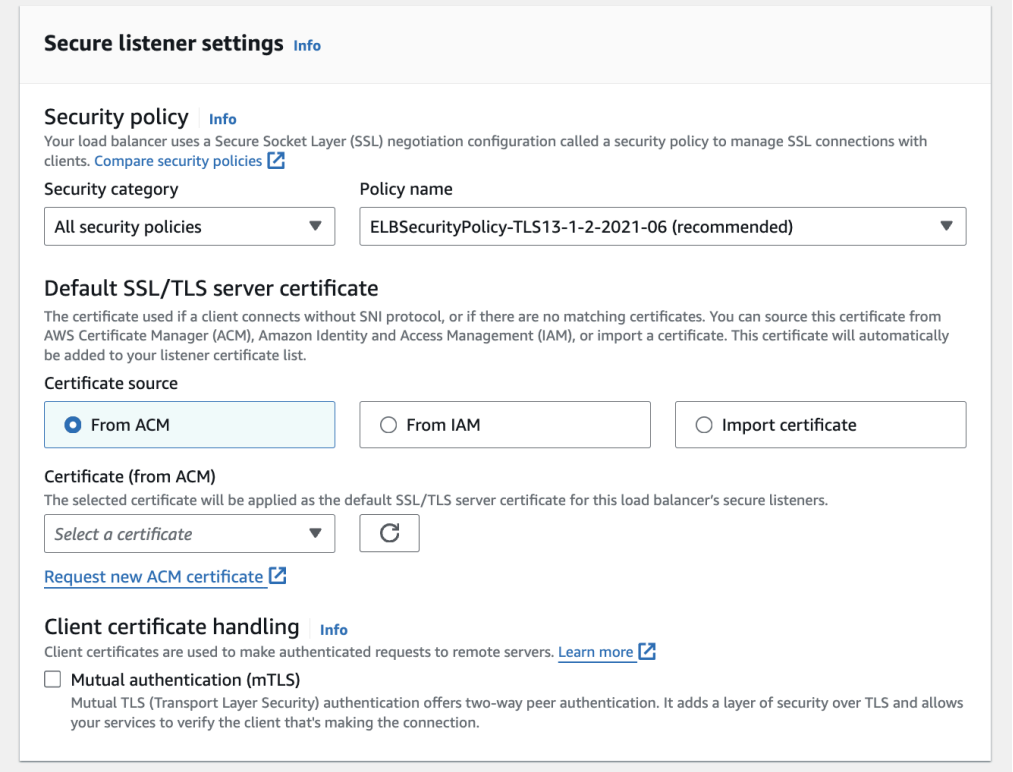
    3. You can see status as success. Then navigate to load balancer and click on add listener.



4. Select Https in listener configuration and the target group which you created.



5. Attach the certificate you created in the listener setting.



6. Navigate to your domain and you can see a secure connection over https.

- **Travel Memory Application Architecture**