**Graded Project on Travel Memory Application Deployment:**

**Introduction:**

The TravelMemory application has been developed using the MERN stack. Your challenge is to deploy this application on an Amazon EC2 instance. This will provide you with hands-on experience in deploying full-stack applications, working with cloud platforms, and ensuring scalable architecture.

**Project Repository:**

Access the complete codebase of the TravelMemory application from the provided GitHub link: https://github.com/UnpredictablePrashant/TravelMemory

**Objective:**

- Set up the backend running on Node.js.

- Configure the front end designed with React.

- Ensure efficient communication between the front end and back end.

- Deploy the full application on an EC2 instance.

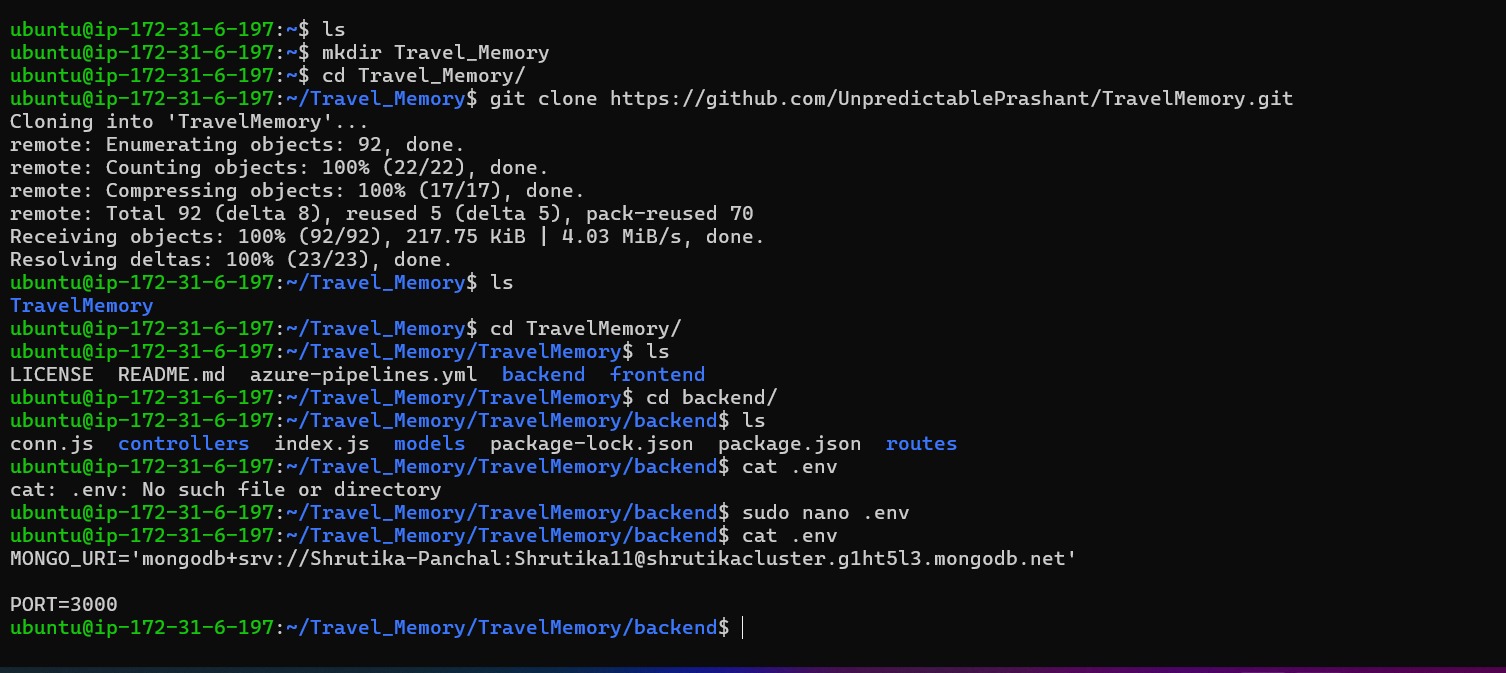
- Facilitate load balancing by creating multiple instances of the application.

- Connect a custom domain through Cloudflare.

**Tasks:**

1. Backend Configuration:

- Clone the repository and navigate to the backend directory.

 - The backend runs on port 3000. Set up a reverse proxy using nginx to ensure smooth deployment on EC2.

**Install Nginx :**

*Commands used :*

sudo apt-get update

sudo apt install nginx

systemctl status nginx.service

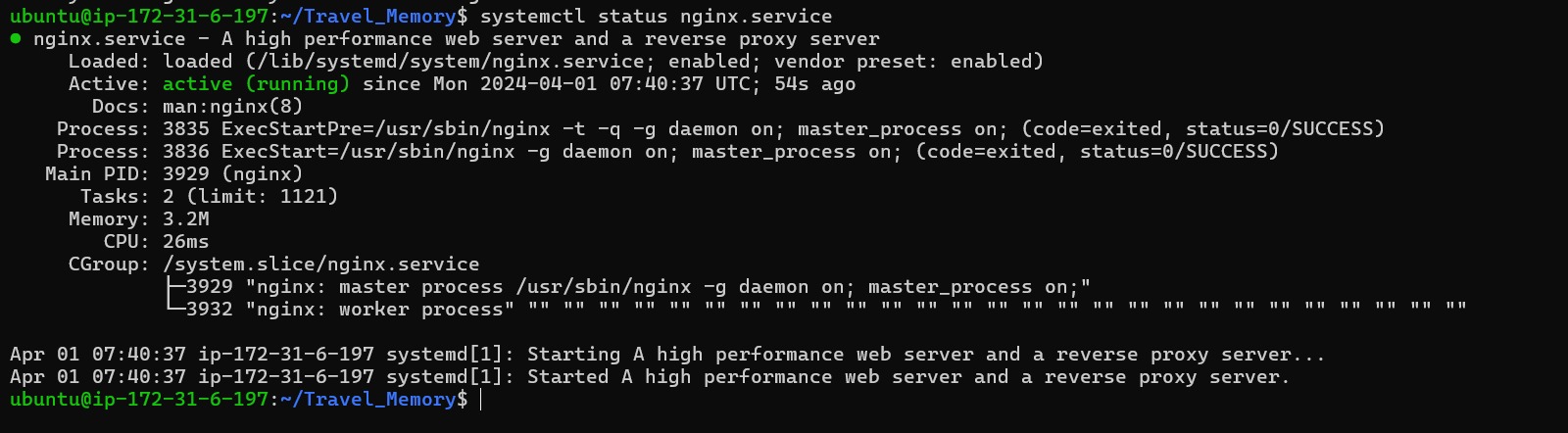
cd /etc/nginx/sites-enabled/

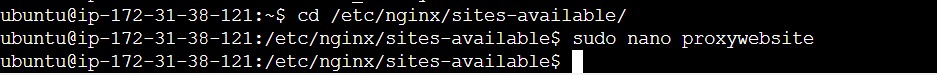
sudo nano proxywebsite

sudo ln -s /etc/nginx/sites-available/proxywebsite /etc/nginx/sites-enabled/

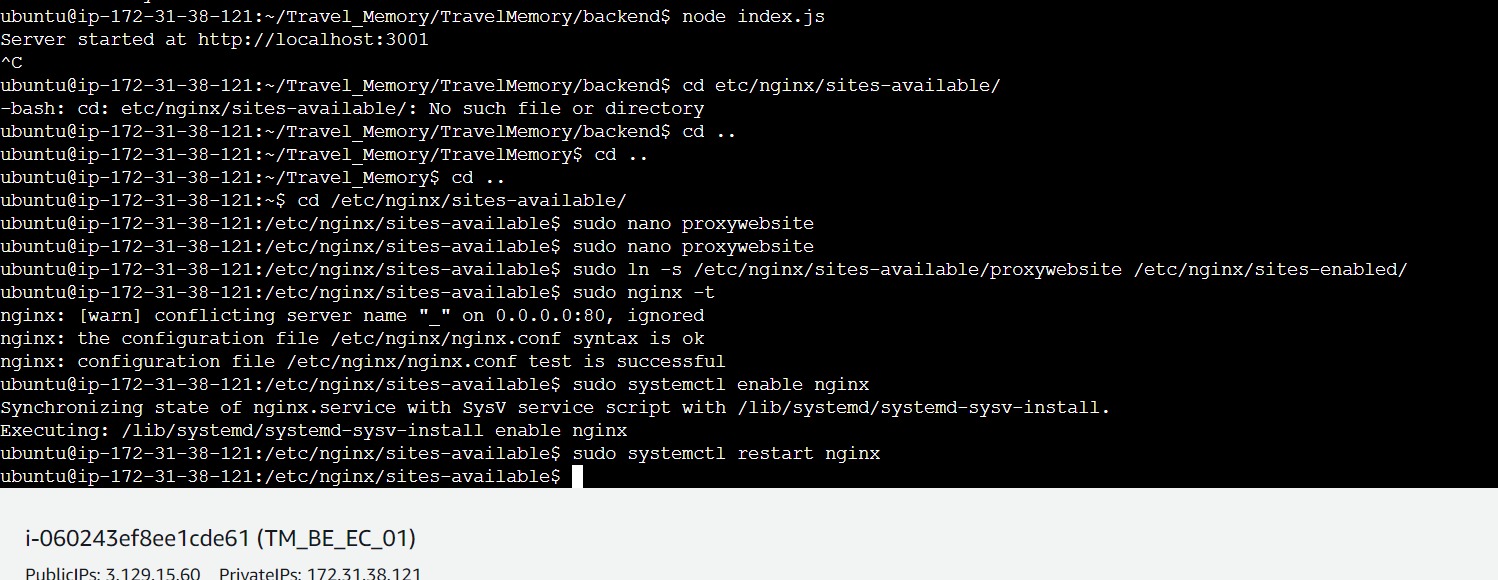
sudo nginx -t

sudo systemctl restart nginx

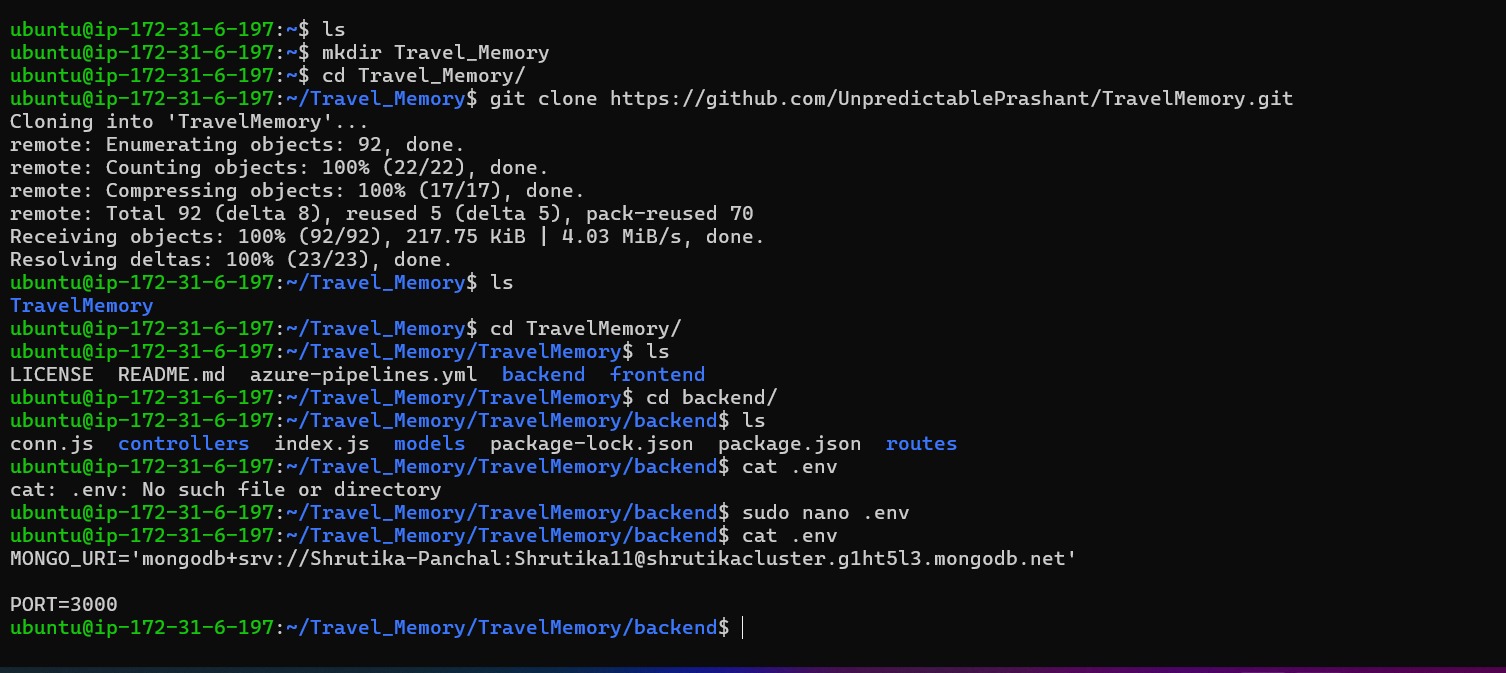


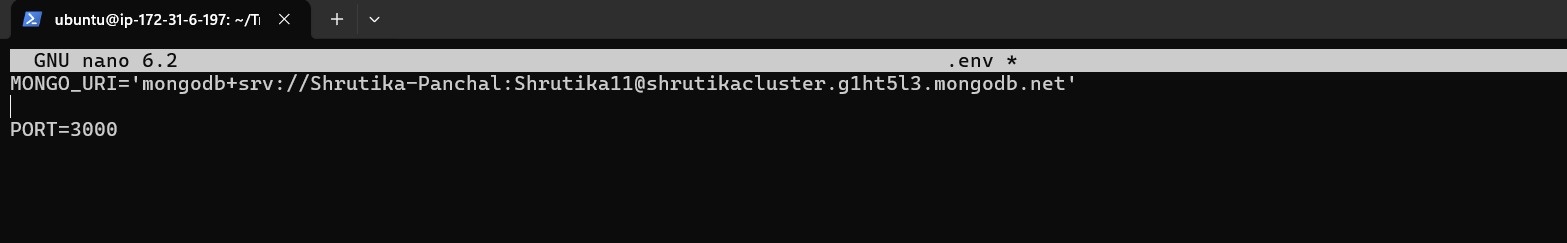






- Update the .env file to incorporate database connection details and port information.





**Node Installation :**

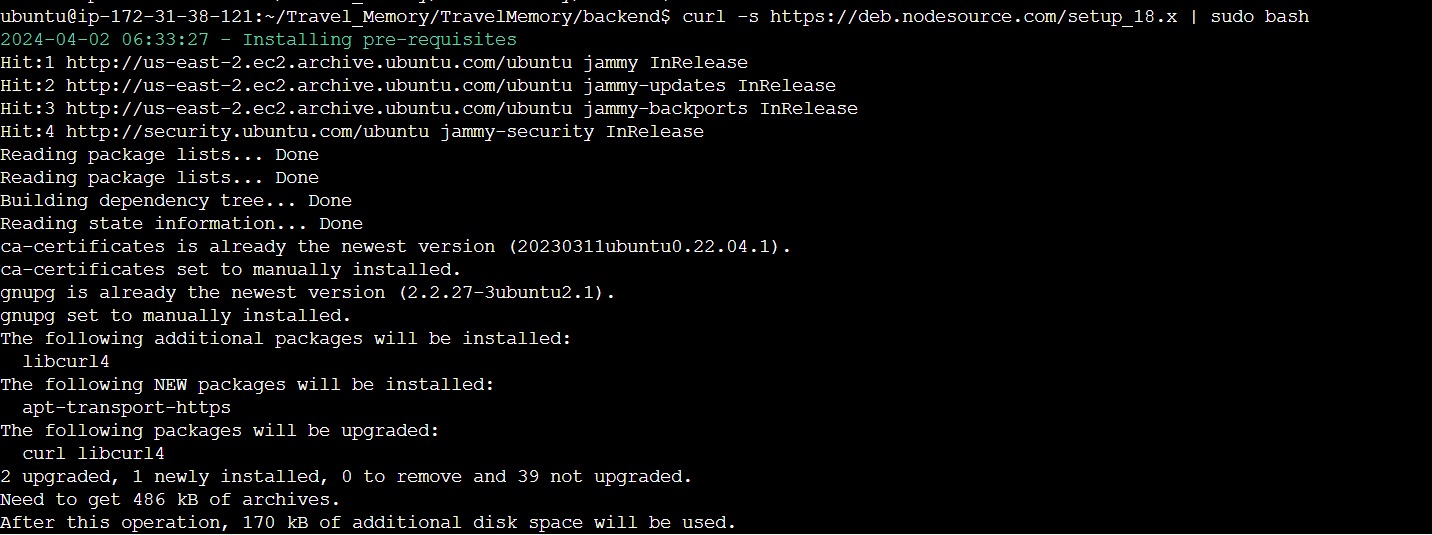
*Commands used :*

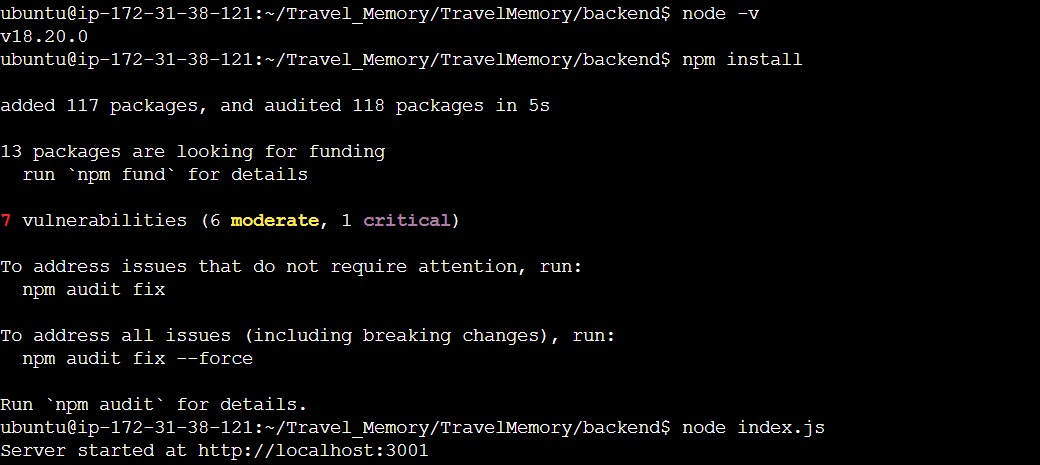
curl -s https://deb.nodesource.com/setup\_18.x | sudo bash

sudo apt-get install nodejs -y

npm install

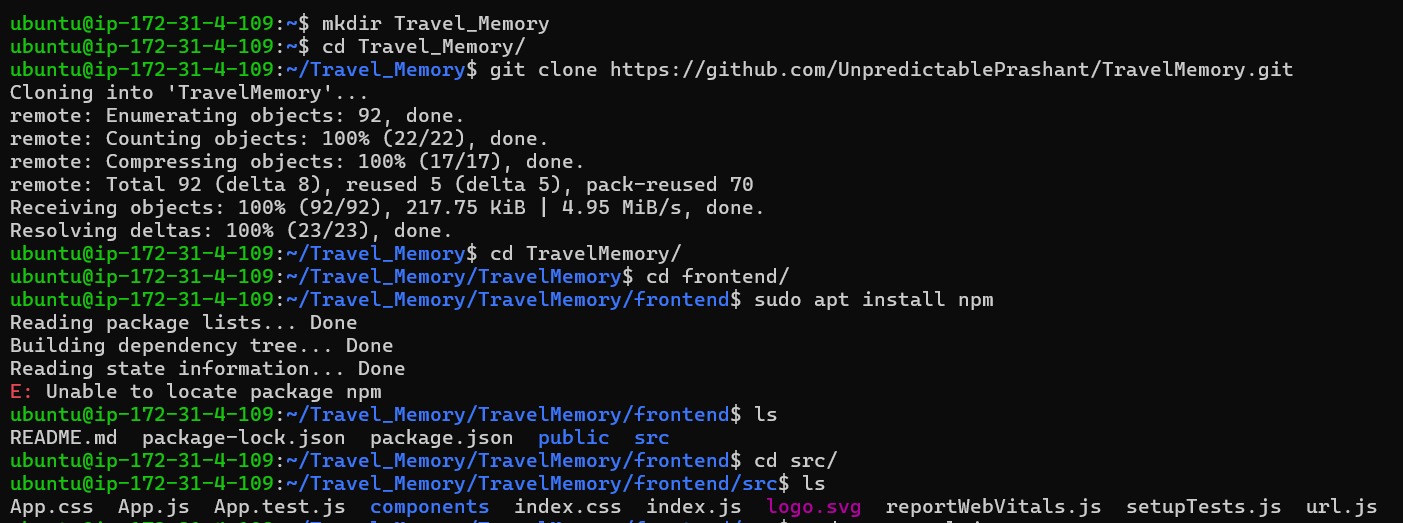
node index.js



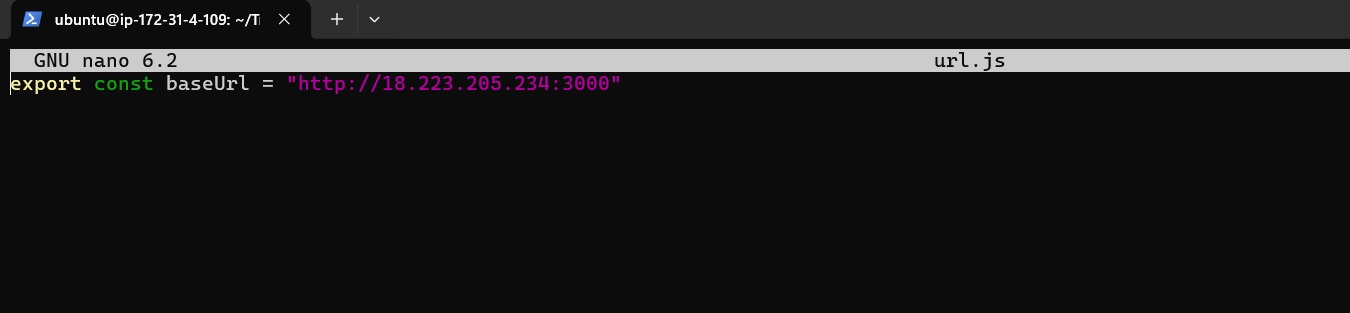


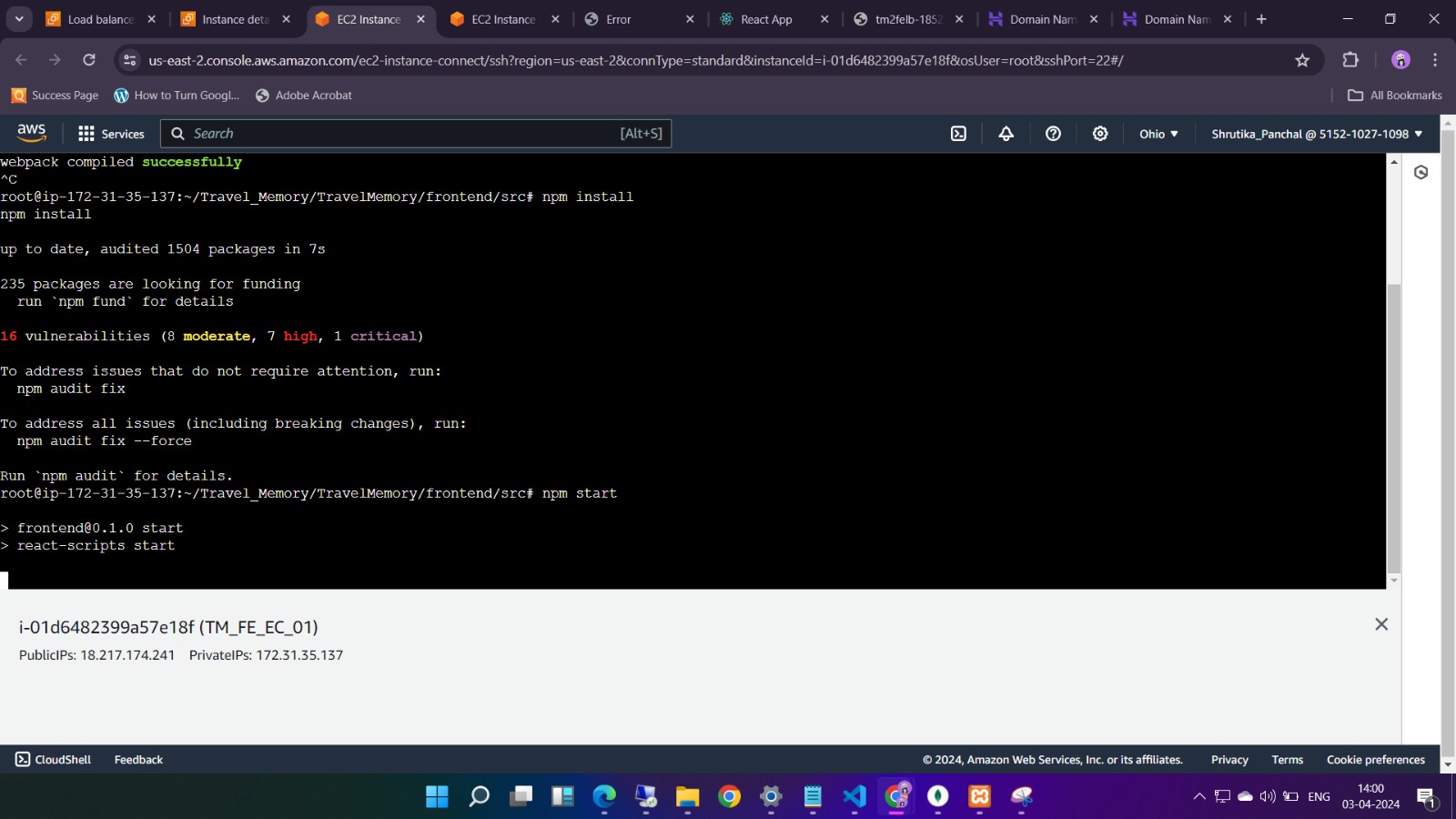
2. Frontend and Backend Connection:

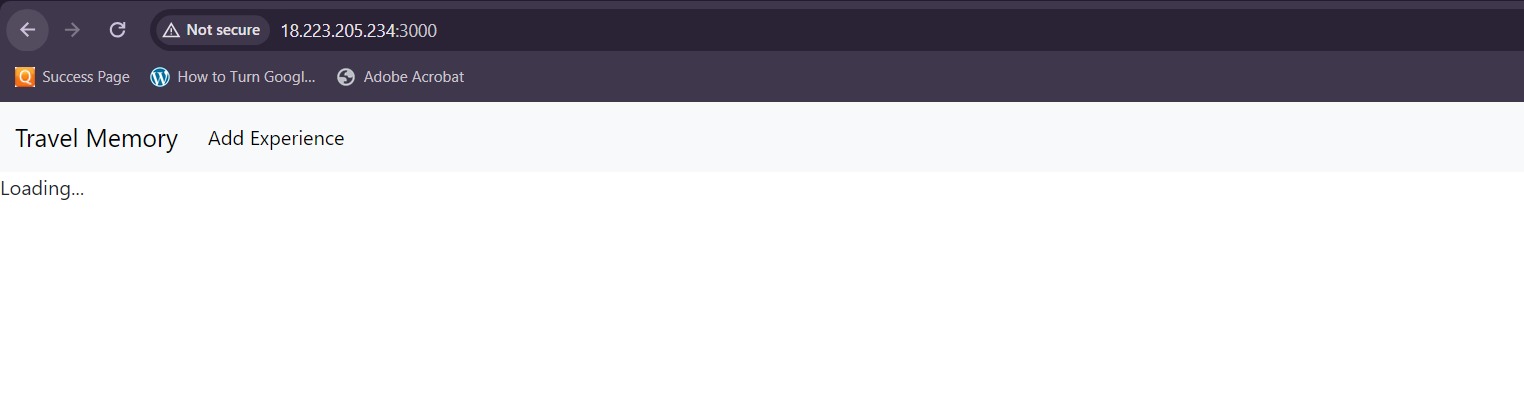
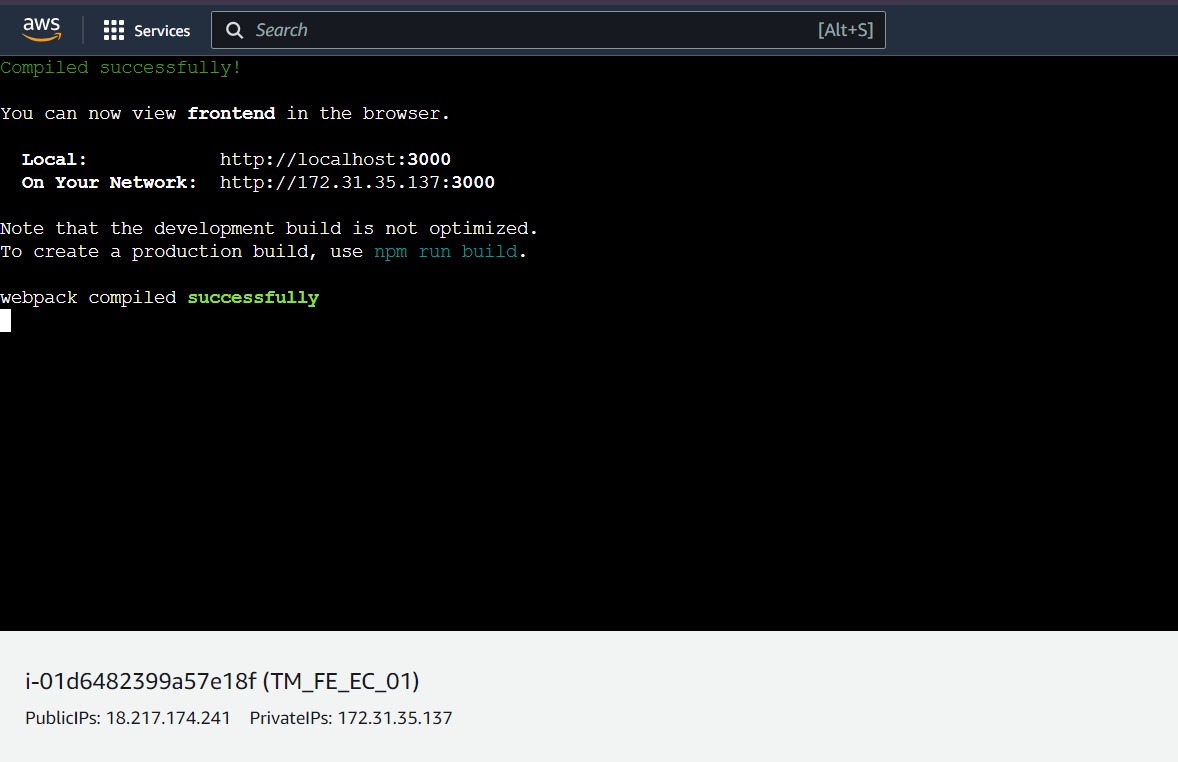
- Navigate to the urls.js in the frontend directory.

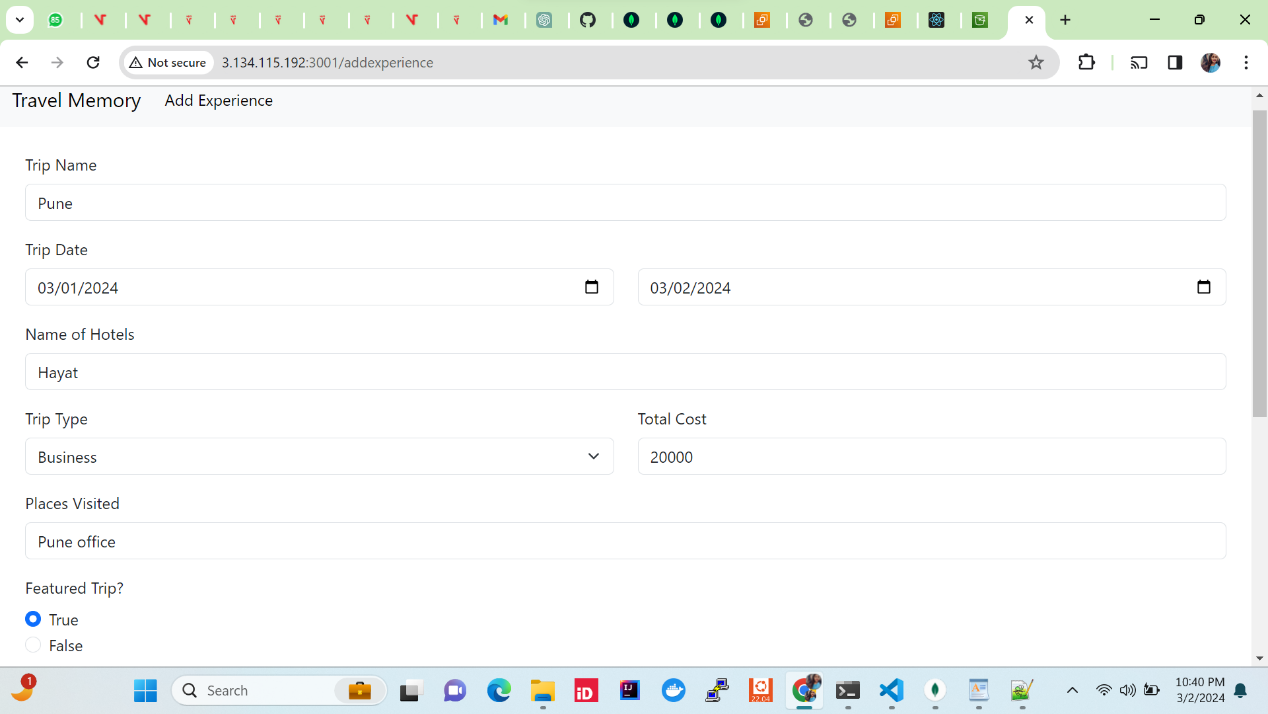


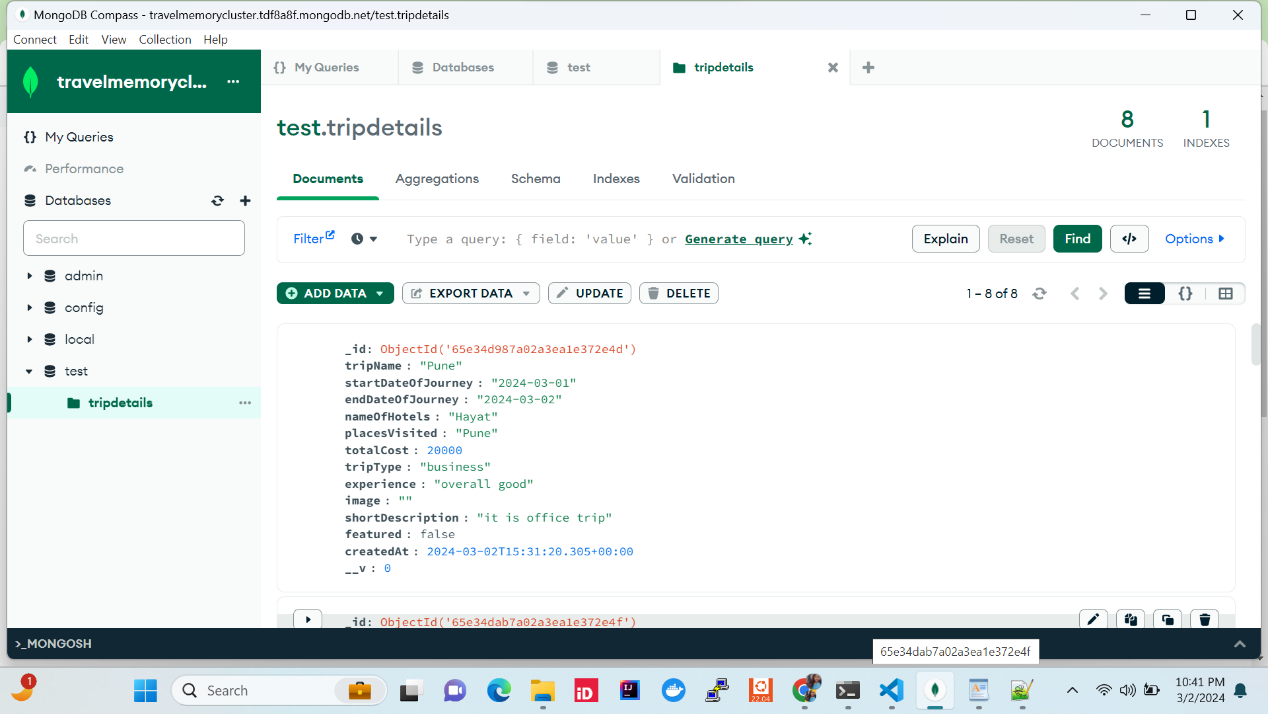
- Update the file to ensure the frontend communicates effectively with the backend.

Sudo nano url.js



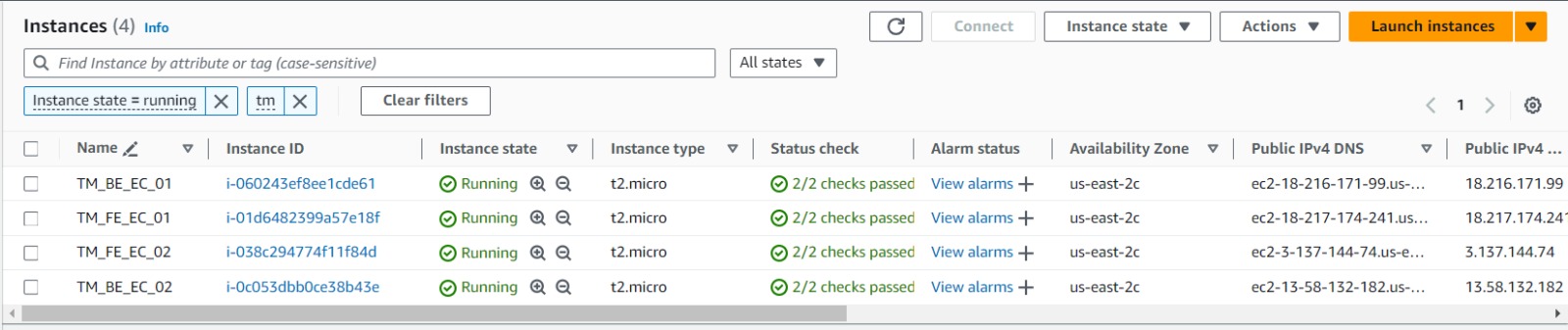




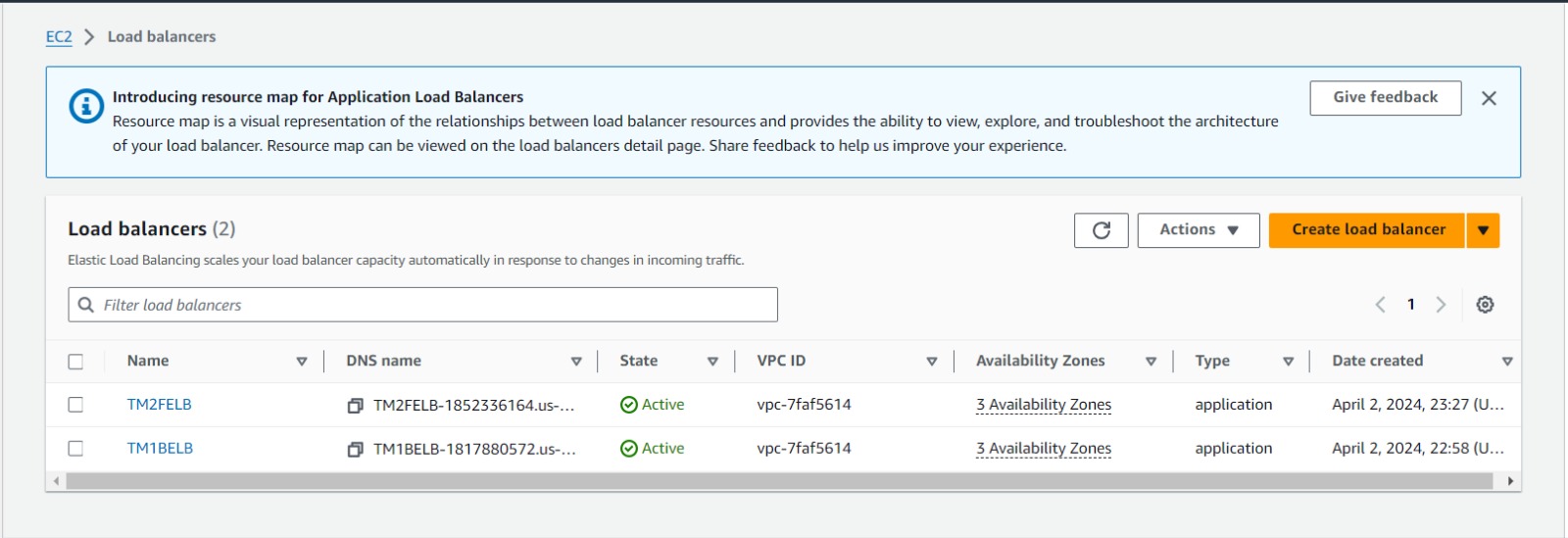


3. Scaling the Application:

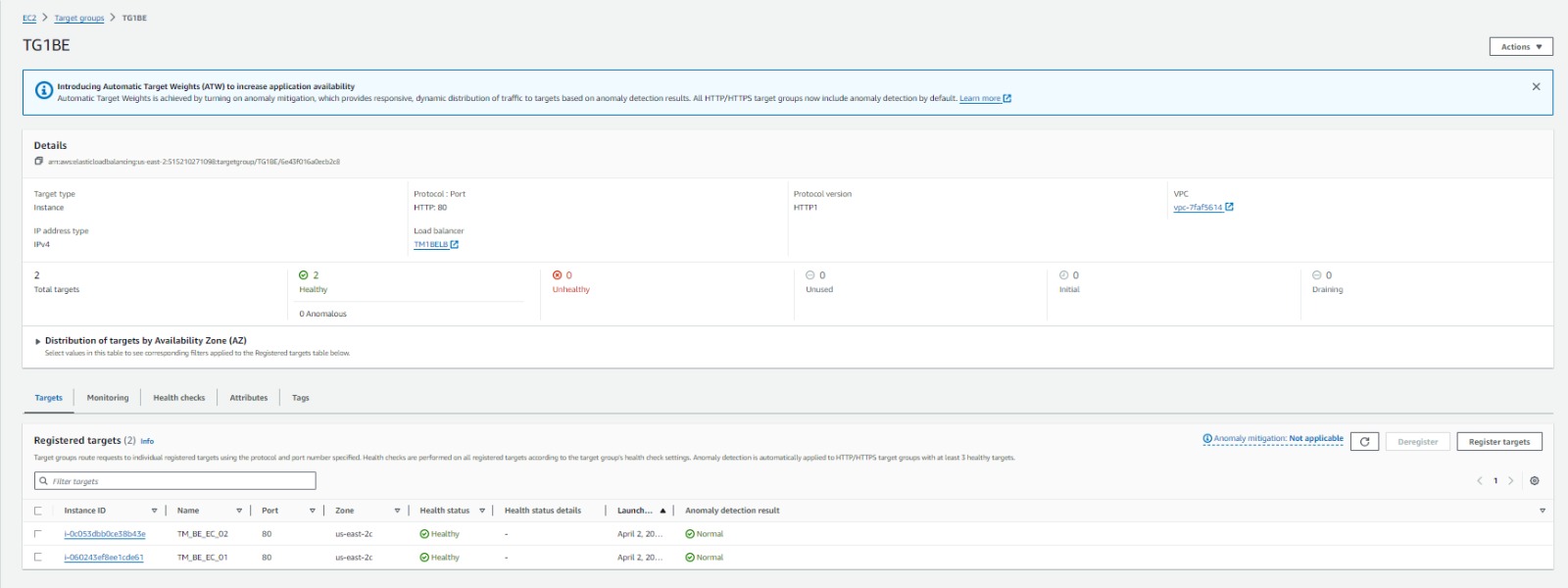
- Create multiple instances of both the frontend and backend servers.



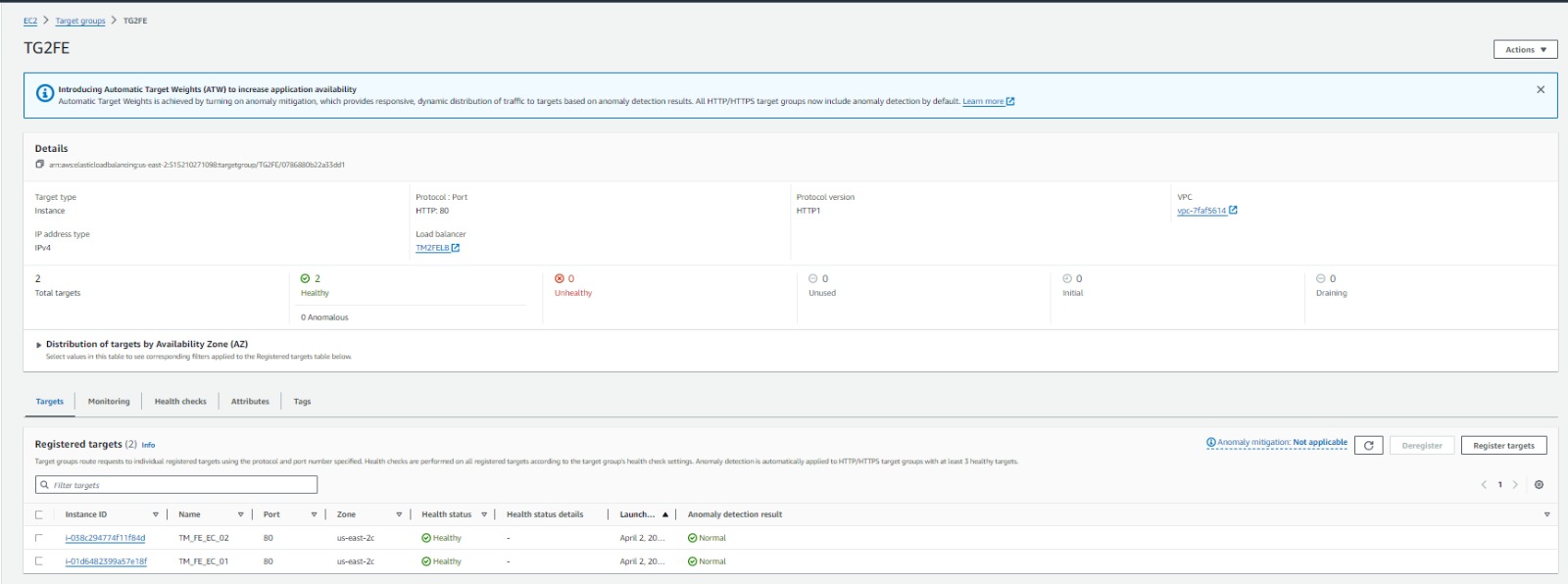
- Add these instances to a load balancer to ensure efficient distribution of incoming traffic.



Backend Target group :



Frontend Target Group :



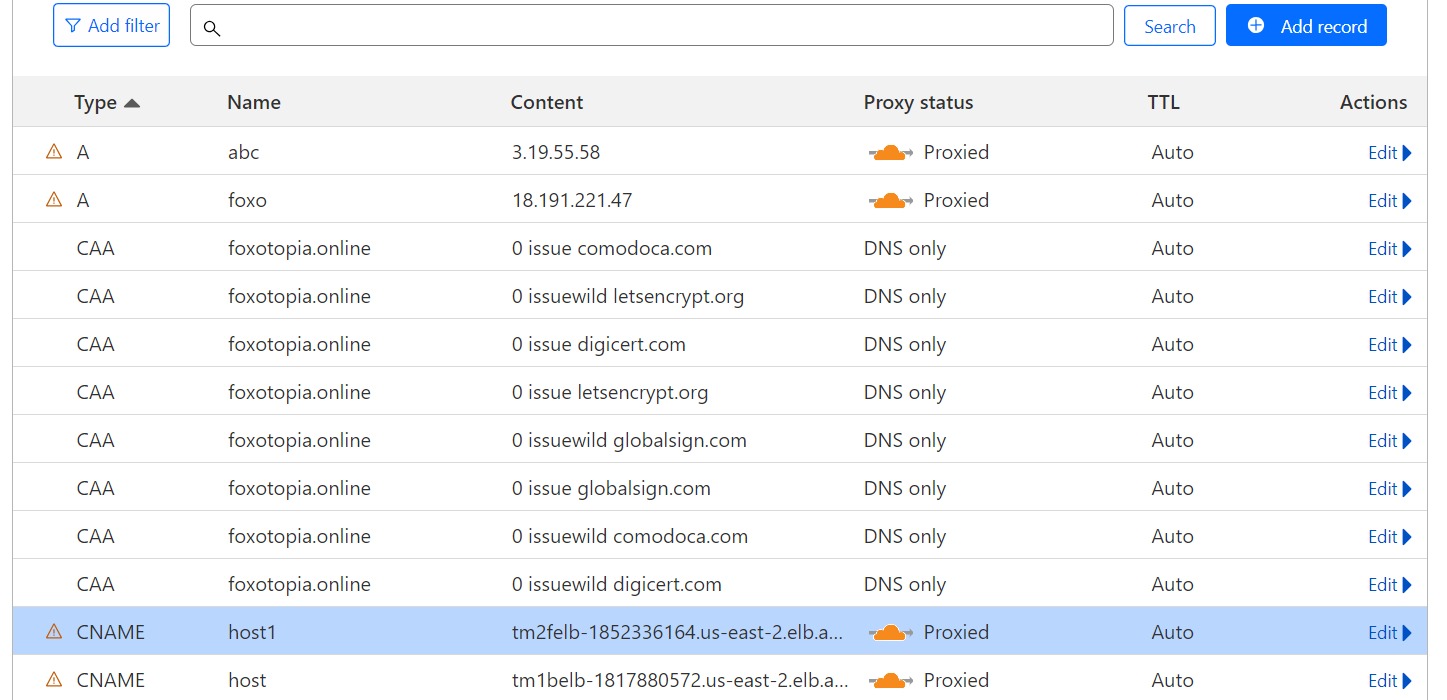
4. Domain Setup with Cloudflare:

- Connect your custom domain to the application using Cloudflare.

- Create a CNAME record pointing to the load balancer endpoint.

- Set up an A record with the IP address of the EC2 instance hosting the frontend.

A record :



CNAME record :

