Graded Project on Travel Memory Application Deployment

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

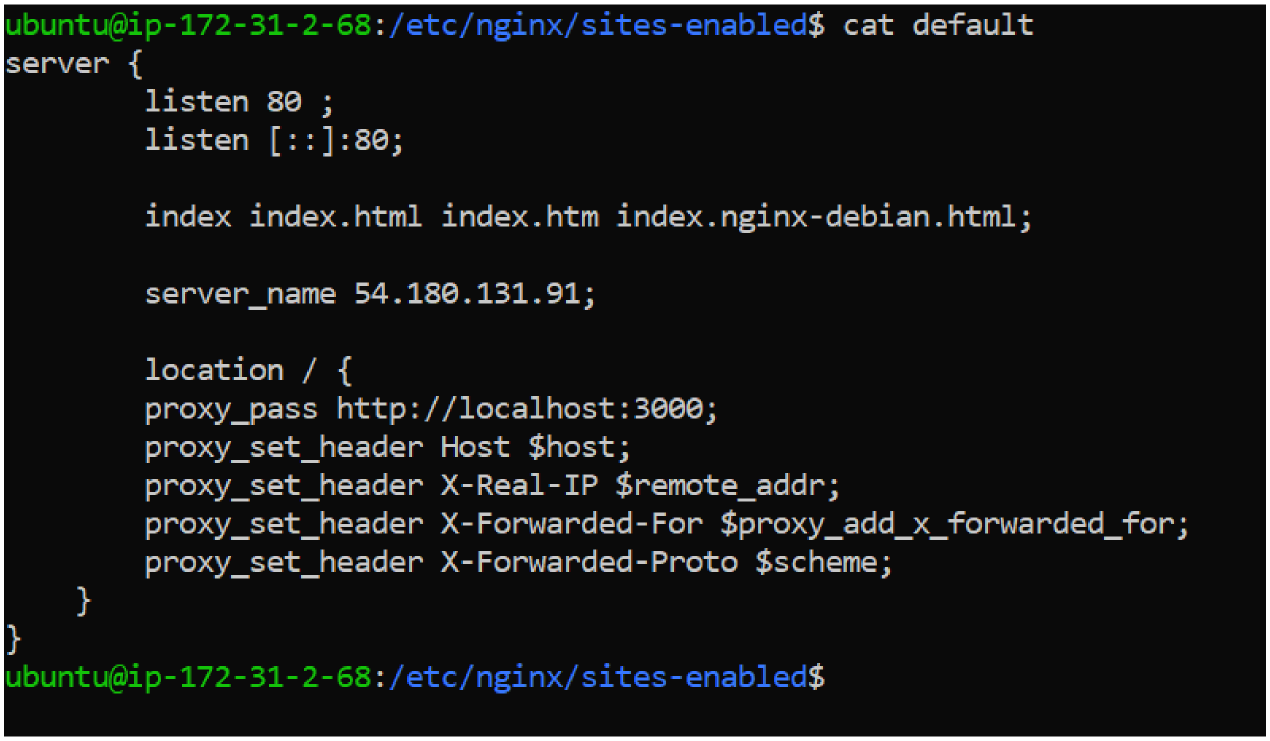
2. Frontend and Backend Connection:

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

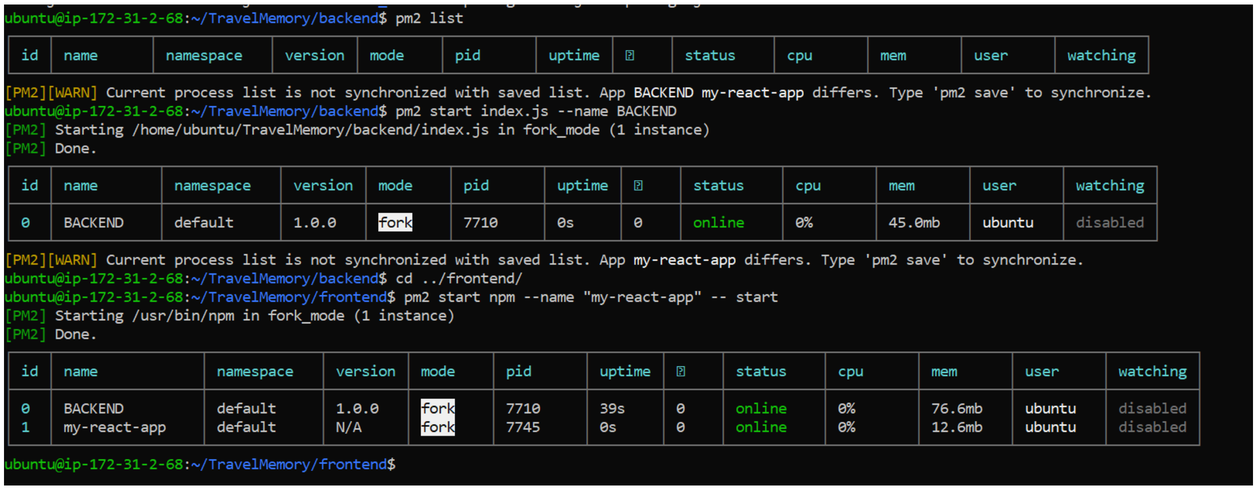
- Update the file to ensure the front end communicates effectively with the backend.

Below are the list of the commands used

```
sudo apt-get update
sudo apt install -y nodejs
sudo npm install pm2 -g
git clone https://github.com/UnpredictablePrashant/TravelMemory
cd TravelMemory/
cd backend/
nano .env
MONGO_URI='mongodb+srv://neehar:*****@cluster0.zrsxpuv.mongodb.net/TravelMemory'
PORT=3001
npm install
pm2 start index.js --name BACKEND
cd ..
cd frontend/
cd src/
nano url.js
export const baseUrl = "http://54.180.131.91:3001"
cd ..
pm2 start npm --name "my-react-app" -- start
curl http://localhost:3000
curl http://localhost:3001/trip
cd /etc/nginx/sites-enabled/
```



```
ubuntu@ip-172-31-2-68:/etc/nginx/sites-enabled$ cat default
server {
        listen 80 ;
        listen [::]:80;

        index index.html index.htm index.nginx-debian.html;

        server_name 54.180.131.91;

        location / {
        proxy_pass http://localhost:3000;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
        proxy_set_header X-Forwarded-Proto $scheme;
    }
}
ubuntu@ip-172-31-2-68:/etc/nginx/sites-enabled$
```

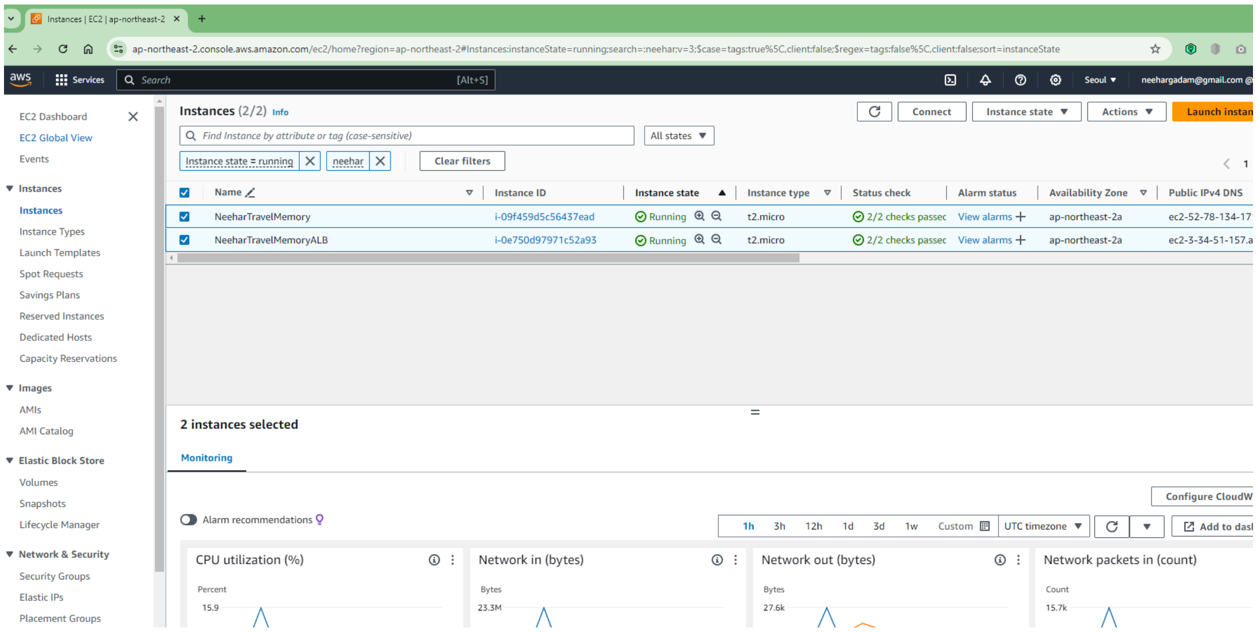


```
ubuntu@ip-172-31-2-68:~/TravelMemory/backend$ pm2 list
│ id │ name      │ namespace  │ version │ mode     │ pid      │ uptime │ ▯  │ status    │ cpu      │ mem      │ user     │ watching │

[PM2][WARN] Current process list is not synchronized with saved list. App BACKEND my-react-app differs. Type 'pm2 save' to synchronize.
ubuntu@ip-172-31-2-68:~/TravelMemory/backend$ pm2 start index.js --name BACKEND
[PM2] Starting /home/ubuntu/TravelMemory/backend/index.js in fork_mode (1 instance)
[PM2] Done.
│ id │ name       │ namespace   │ version │ mode     │ pid      │ uptime │ ▯  │ status    │ cpu      │ mem      │ user     │ watching │
│ 0  │ BACKEND    │ default     │ 1.0.0   │ fork     │ 7710     │ 0s     │ 0  │ online    │ 0%       │ 45.0mb   │ ubuntu   │ disabled │

[PM2][WARN] Current process list is not synchronized with saved list. App my-react-app differs. Type 'pm2 save' to synchronize.
ubuntu@ip-172-31-2-68:~/TravelMemory/backend$ cd ../frontend/
ubuntu@ip-172-31-2-68:~/TravelMemory/frontend$ pm2 start npm --name "my-react-app" -- start
[PM2] Starting /usr/bin/npm in fork_mode (1 instance)
[PM2] Done.
│ id │ name            │ namespace   │ version │ mode     │ pid      │ uptime │ ▯  │ status    │ cpu      │ mem      │ user     │ watching │
│ 0  │ BACKEND         │ default     │ 1.0.0   │ fork     │ 7710     │ 39s    │ 0  │ online    │ 0%       │ 76.6mb   │ ubuntu   │ disabled │
│ 1  │ my-react-app    │ default     │ N/A     │ fork     │ 7745     │ 0s     │ 0  │ online    │ 0%       │ 12.6mb   │ ubuntu   │ disabled │

ubuntu@ip-172-31-2-68:~/TravelMemory/frontend$
```

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



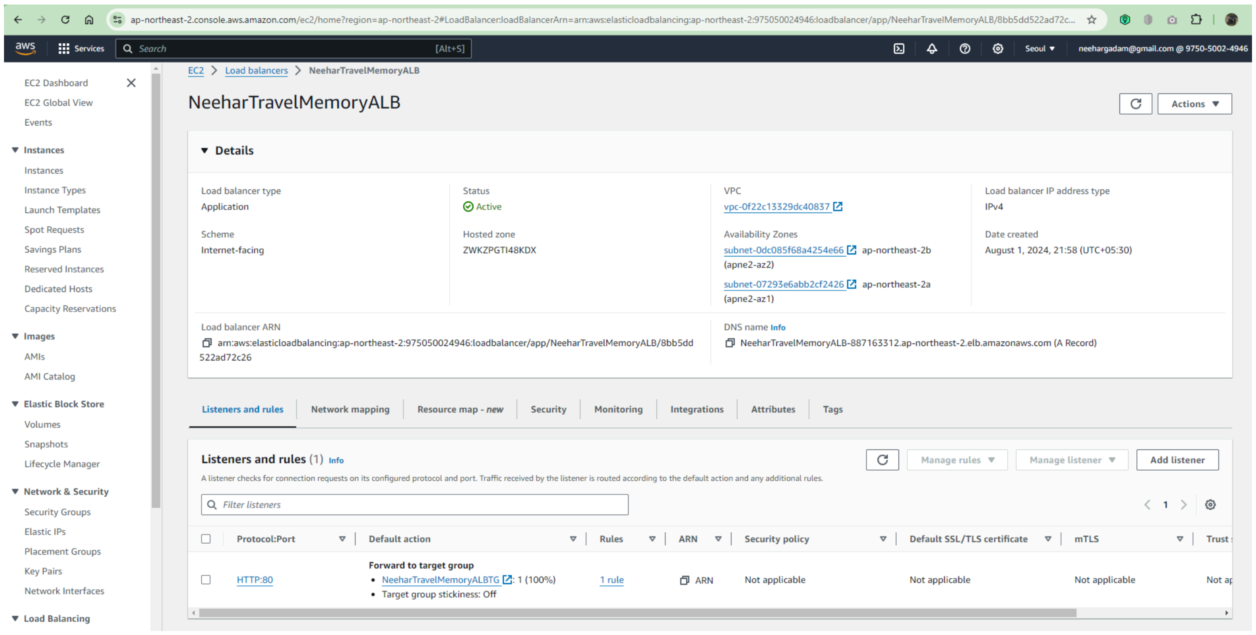
Instances (2/2) Info

Filters: Instance state = running, neehar

Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS
NeeharTravelMemory	i-09f459d5c56437ead	Running	t2.micro	2/2 checks passec	View alarms +	ap-northeast-2a	ec2-52-78-134-17
NeeharTravelMemoryALB	i-0e750d97971c52a93	Running	t2.micro	2/2 checks passec	View alarms +	ap-northeast-2a	ec2-3-34-51-157.a

2 instances selected

Monitoring



EC2 > Load balancers > NeeharTravelMemoryALB

NeeharTravelMemoryALB

Details

Load balancer type: Application
Status: Active
VPC: vpc-0f22c13329dc40837
Load balancer IP address type: IPv4
Scheme: Internet-facing
Hosted zone: ZWKZPGTI48KDX
Availability Zones: subnet-0dc085f68a4254e66 ap-northeast-2b (apne2-az2); subnet-07293e6abb2cf2426 ap-northeast-2a (apne2-az1)
Date created: August 1, 2024, 21:58 (UTC+05:30)
Load balancer ARN: arn:aws:elasticloadbalancing:ap-northeast-2:975050024946:loadbalancer/app/NeeharTravelMemoryALB/8bb5dd522ad72c26
DNS name: NeeharTravelMemoryALB-887163312.ap-northeast-2.elb.amazonaws.com (A Record)

Listeners and rules (1)

A listener checks for connection requests on its configured protocol and port. Traffic received by the listener is routed according to the default action and any additional rules.

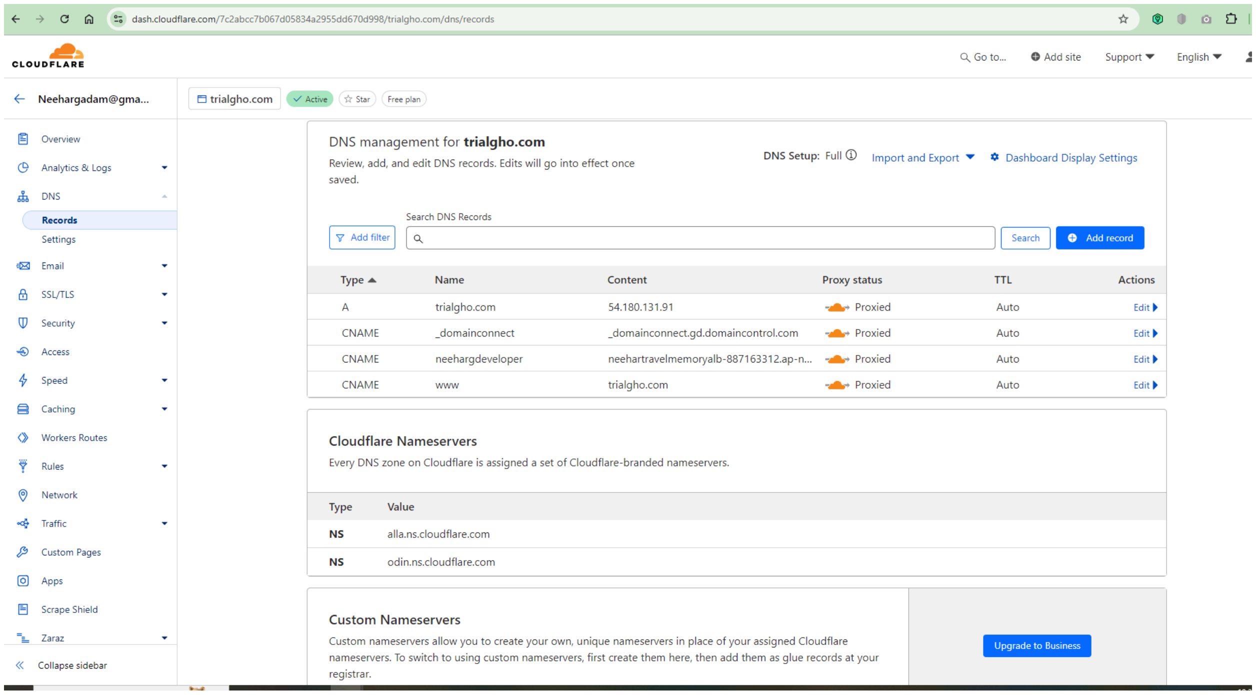
Protocol:Port	Default action	Rules	ARN	Security policy	Default SSL/TLS certificate	mTLS
HTTP:80	Forward to target group: NeeharTravelMemoryALBTG: 1 (100%); Target group stickiness: Off	1 rule	ARN	Not applicable	Not applicable	Not applicable

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 front end.



DNS management for **trialgho.com**

Review, add, and edit DNS records. Edits will go into effect once saved.

DNS Setup: Full

Type	Name	Content	Proxy status	TTL	Actions
A	trialgho.com	54.180.131.91	Proxied	Auto	Edit
CNAME	_domainconnect	_domainconnect.gd.domaincontrol.com	Proxied	Auto	Edit
CNAME	neehargdeveloper	neehartravelmemoryalb-887163312.ap-n...	Proxied	Auto	Edit
CNAME	www	trialgho.com	Proxied	Auto	Edit

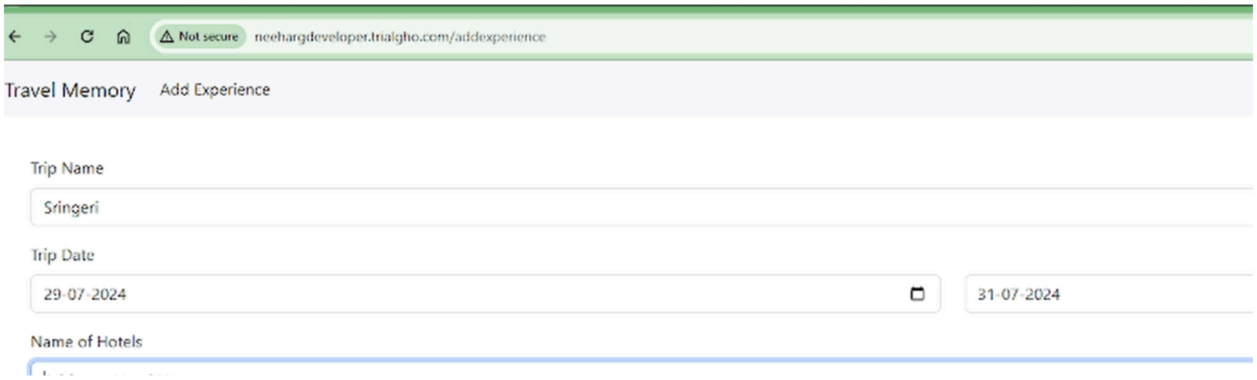
Cloudflare Nameservers

Every DNS zone on Cloudflare is assigned a set of Cloudflare-branded nameservers.

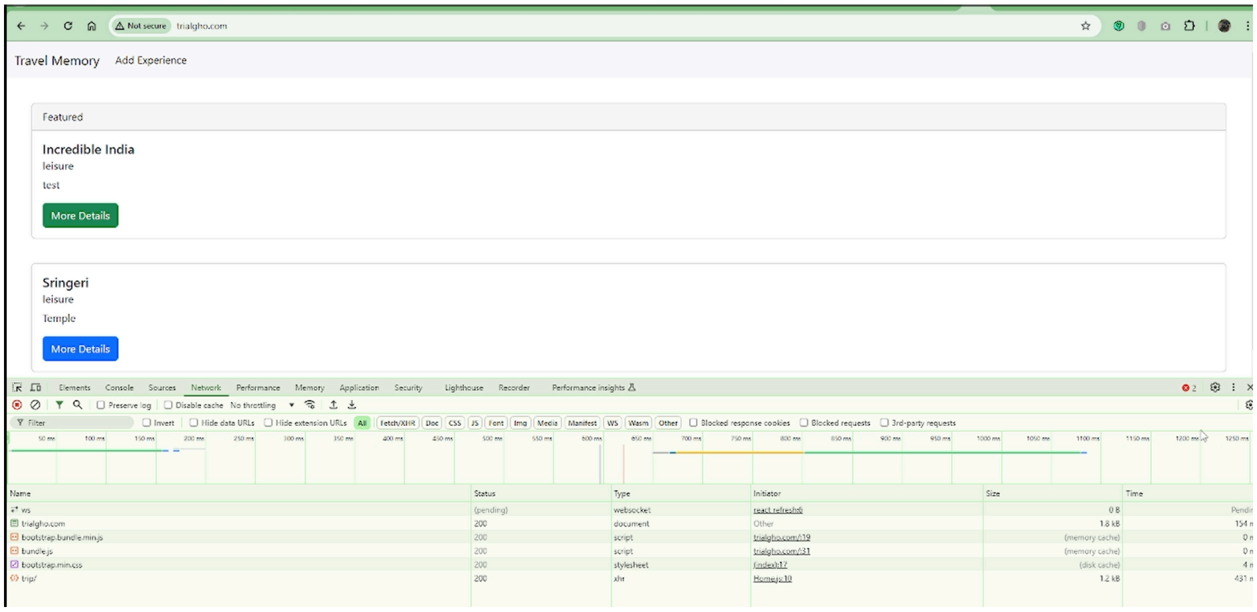
Type	Value
NS	alla.ns.cloudflare.com
NS	odin.ns.cloudflare.com

Custom Nameservers

Custom nameservers allow you to create your own, unique nameservers in place of your assigned Cloudflare nameservers. To switch to using custom nameservers, first create them here, then add them as glue records at your registrar.



Trip Name	Sringeri
Trip Date	29-07-2024 / 31-07-2024
Name of Hotels	

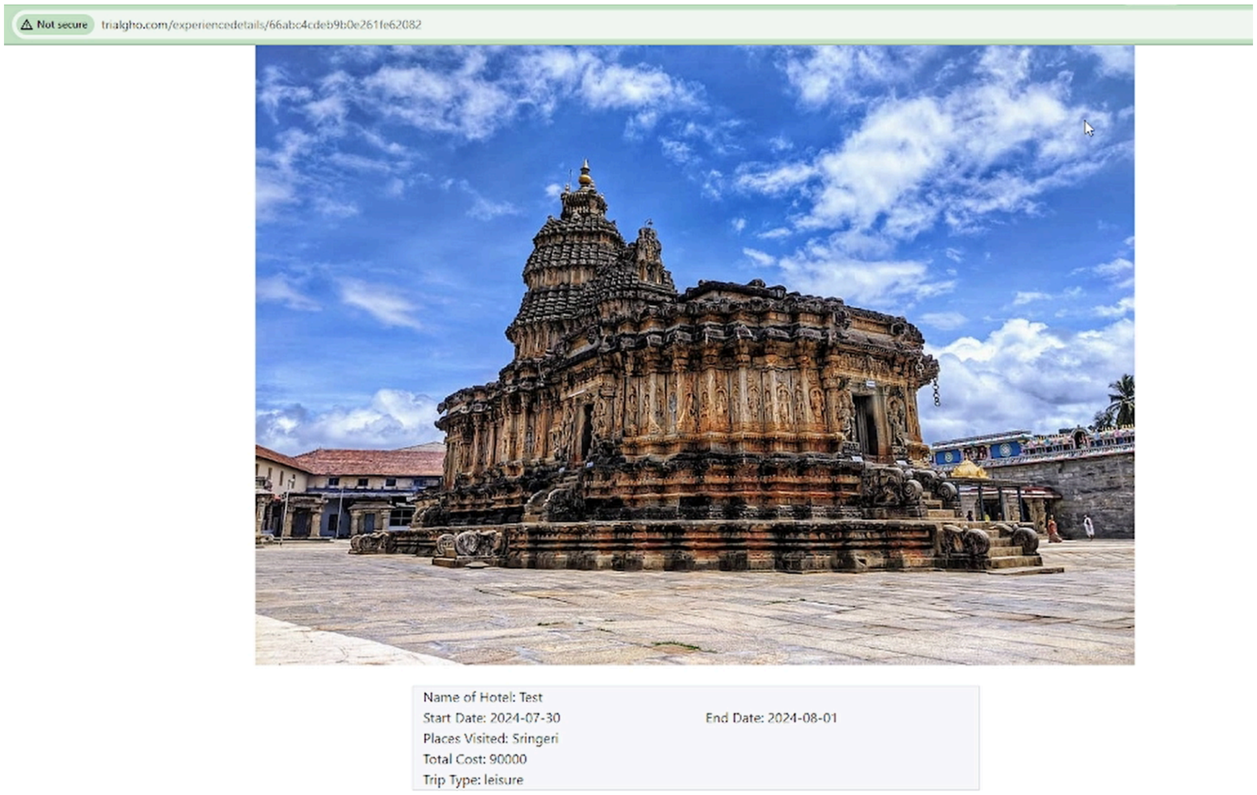


Featured

Incredible India
leisure
test
More Details

Sringeri
leisure
Temple
More Details

```
Name                  Status     Type        Initiator          Size             Time
ws                    (pending)  websocket   react refresh:6               0 B   Pendi
trialgho.com          200        document    Other                        1.8 kB  154 n
bootstrap.bundle.min.js 200      script      trialgho.com/:19   (memory cache)    0 n
bundle.js             200        script      trialgho.com/:31   (memory cache)    0 n
bootstrap.min.css     200        stylesheet  (index):17           (disk cache)    4 n
trip/                 200        xhr         Home.js:10                   1.2 kB  431 n
```

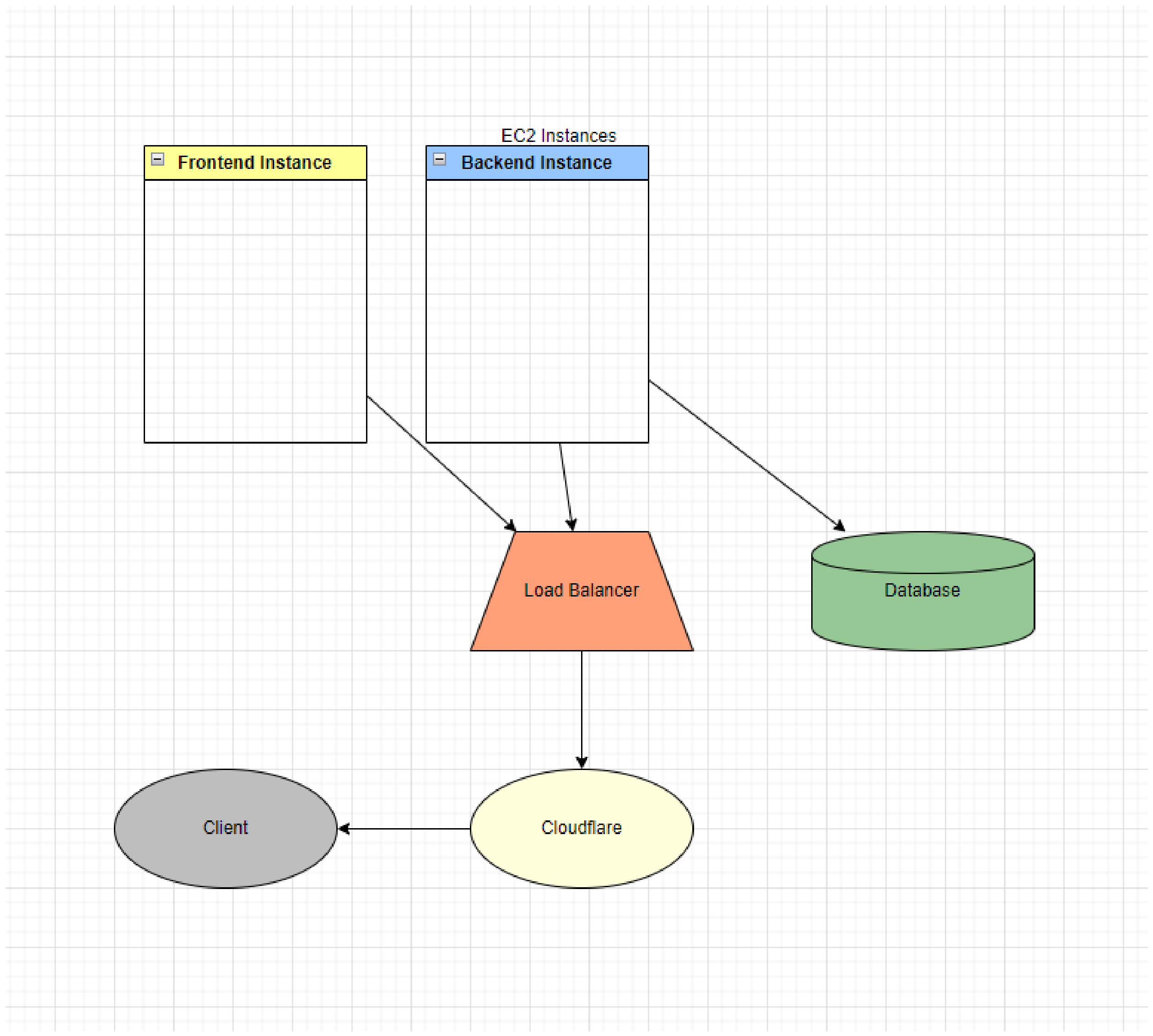


Name of Hotel: Test
Start Date: 2024-07-30 End Date: 2024-08-01
Places Visited: Sringeri
Total Cost: 90000
Trip Type: leisure

5. Documentation:

- Prepare comprehensive documentation detailing each step of the deployment process. Include relevant screenshots to make the process clear and reproducible.

- Design a deployment architecture diagram using [draw.io](https://www.draw.io/) to visualize the flow and connections.


EC2 Instances
Frontend Instance
Backend Instance
Load Balancer
Database
Client
Cloudflare