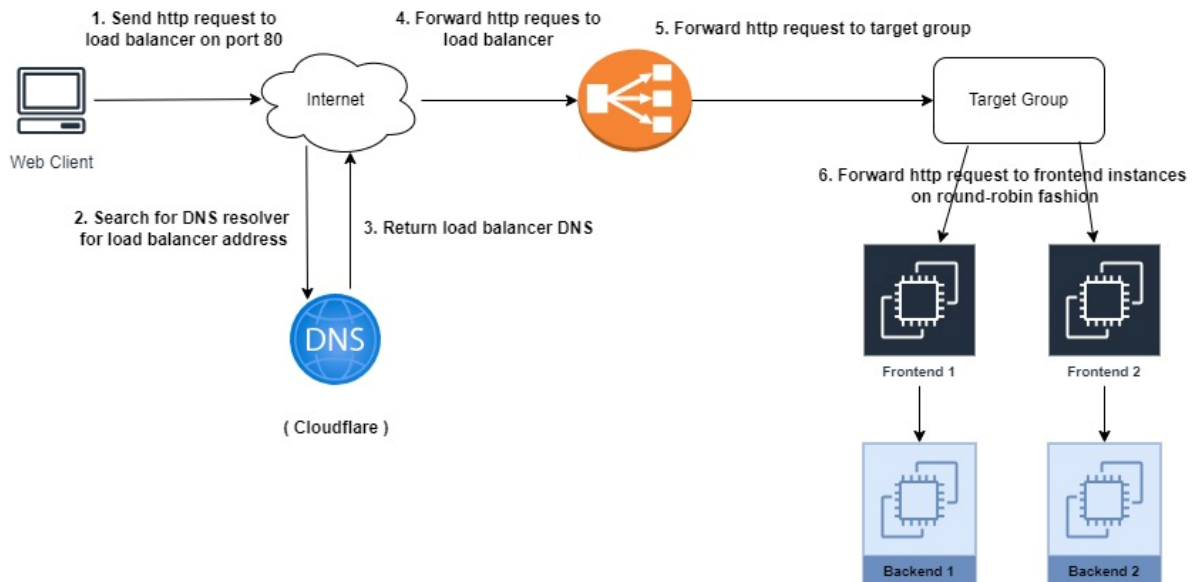


Travel memory application

Architecture Diagram:



Steps for deployment process:

Step 1.a:

Launch the instance (sonal_backend1) and clone the repository using below command:

git clone <https://github.com/UnpredictablePrashant/TravelMemory.git>

Step 1.b:

Create the .env file in the TravelMemory/backend folder and add below details

```
MONGO_URI='mongodb+srv://sonal:sonal1189@cluster-sonal.0ktone2.mongodb.net/MERN'
PORT=3001
```

Step 1.c

Start the backend server with command “npm install” and “node index.js”

This will start the backend server at port no 3001

```

ubuntu@ip-172-31-4-177:~$ cd TravelMemory/
ubuntu@ip-172-31-4-177:~/TravelMemory$ cd backend/
ubuntu@ip-172-31-4-177:~/TravelMemory/backend$ npm install

up to date, audited 118 packages in 3s

13 packages are looking for funding
  run `npm fund` for details

9 vulnerabilities (5 moderate, 3 high, 1 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
ubuntu@ip-172-31-4-177:~/TravelMemory/backend$ node index.js
Server started at http://localhost:3001

```

Step 1.d

Set up a reverse proxy using nginx with below configuration:

```

ubuntu@ip-172-31-3-129:~$ cd TravelMemory/
ubuntu@ip-172-31-3-129:~/TravelMemory$ sudo apt-get install nginx
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
nginx is already the newest version (1.24.0-2ubuntu7).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
ubuntu@ip-172-31-3-129:~/TravelMemory$ cd /etc/nginx/sites-available/
ubuntu@ip-172-31-3-129:/etc/nginx/sites-available$ ls
default
ubuntu@ip-172-31-3-129:/etc/nginx/sites-available$ nano default

```

```

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

server_name _;

location / {
    # First attempt to serve request as file, then
    # as directory, then fall back to displaying a 404.
    # try_files $uri $uri/ =404;
    proxy_pass http://localhost:3000;
    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;
}

# pass PHP scripts to FastCGI server
#

```

Step 2.a: Launch the frontend instance (sonal_frontend1)

Step 2.b : Install frontend server using command “npm install”

```
ubuntu@ip-172-31-3-129:~$ cd TravelMemory/  
ubuntu@ip-172-31-3-129:~/TravelMemory$ cd frontend/  
ubuntu@ip-172-31-3-129:~/TravelMemory/frontend$ npm install  
  
up to date, audited 1504 packages in 12s  
  
235 packages are looking for funding  
  run `npm fund` for details  
  
19 vulnerabilities (9 moderate, 9 high, 1 critical)  
  
To address issues that do not require attention, run:  
  npm audit fix  
  
To address all issues (including breaking changes), run:  
  npm audit fix --force  
  
Run `npm audit` for details.  
ubuntu@ip-172-31-3-129:~/TravelMemory/frontend$ cd src  
ubuntu@ip-172-31-3-129:~/TravelMemory/frontend/src$ nano url.js
```

Step 2.c: Update url.js with backend IP

```
GNU nano 7.2  
export const baseUrl = "http://52.78.28.89:3001"
```

Step 2.d Start the frontend server with command “npm start”

```
^C  
ubuntu@ip-172-31-3-129:~/TravelMemory/frontend$  
  
ubuntu@ip-172-31-3-129:~/TravelMemory/frontend$ npm start
```

This will start the frontend server at port no-3000

```
Compiled successfully!

You can now view frontend in the browser.

Local:      http://localhost:3000
On Your Network: http://172.31.3.129:3000

Note that the development build is not optimized.
To create a production build, use npm run build.

webpack compiled successfully
```

Step3: Create two AMI images from above instances.

Step4: Create a set of backend and frontend instances (sonal_backend2 and sonal_frontend2)

Step5: Create a target group “TG-FE-SONAL” and register frontend instances as below:

EC2 > Target groups > TG-FE-SONAL

TG-FE-SONAL

Actions ▾

Details
arn:aws:elasticloadbalancing:ap-northeast-2:975050024946:targetgroup/TG-FE-SONAL/696ddb0650cd43c

Target type Instance	Protocol : Port HTTP: 80	Protocol version HTTP1	VPC vpc-0f22c13329dc40837
IP address type IPv4	Load balancer MERN-LB-SONAL		

2 Total targets	2 Healthy 0 Anomalous	0 Unhealthy	0 Unused	0 Initial	0 Draining
--------------------	-----------------------------	----------------	-------------	--------------	---------------

► Distribution of targets by Availability Zone (AZ)
Select values in this table to see corresponding filters applied to the Registered targets table below.

Targets | Monitoring | Health checks | Attributes | Tags

Registered targets (2) Info
Target groups route requests to individual registered targets using the protocol and port number specified. Health checks are performed on all registered targets according to the target group's health check settings. Anomaly detection is automatically applied to HTTP/HTTPS target groups with at least 3 healthy targets.

Anomaly mitigation: Not applicable 🔄 Deregister Register targets

Filter targets

<input type="checkbox"/>	Instance ID	Name	Port	Zone	Health status	Health status details	Launch...	Anomaly detection result
<input type="checkbox"/>	i-033e78624f94c57b0	sonal_frontend2	80	ap-northeast-2a	Healthy	-	July 14, 2...	Normal
<input type="checkbox"/>	i-0892e9d95ced1fc04	sonal_frontend1	80	ap-northeast-2a	Healthy	-	July 14, 2...	Normal

Step4: Created load balancer MERN_LB_SONAL and added listener “TG-FE-SONAL”

MERN-LB-SONAL

Details

Load balancer type Application	Status Active	VPC vpc-0f22c13329dc40837	Load balancer IP address type IPv4
Scheme Internet-facing	Hosted zone ZWKZPGT148KDX	Availability Zones subnet-0dc085f68a4254e66 ap-northeast-2b (apne2-az2) subnet-05c5c244dc8e4409a ap-northeast-2c (apne2-az3) subnet-07295e6abb2cf2426 ap-northeast-2a (apne2-az1) subnet-022c9b6354b90eb1a ap-northeast-2d (apne2-az4)	Date created July 10, 2024, 18:30 (UTC+05:30)
Load balancer ARN arn:aws:elasticloadbalancing:ap-northeast-2:975050024946:loadbalancer/app/MERN-LB-SONAL/ca8b2dca4c6a119c		DNS name MERN-LB-SONAL-1170111301.ap-northeast-2.elb.amazonaws.com (A Record)	

Listeners and rules (1)

Protocol/Port	Default action	Rules	ARN	Security policy	Default SSL/TLS certificate	mTLS	Trust store
HTTP:80	Forward to target group • TG-FE-SONAL 1 (100%) • Target group stickiness: Off	1 rule	ARN	Not applicable	Not applicable	Not applicable	Not applicable

Step5: Domain Setup with Cloudflare:

1. Created a CNAME record pointing to the load balancer DNS.

CNAME	maketravelmemory	mern-lb-sonal-1170111301.ap-north...	DNS only	Auto	Edit
-------	------------------	--------------------------------------	----------	------	------

2. Create two A records pointing to frontend instances DNS.

A	frontend2	35.10.95.105	DNS only	Auto	Edit
A	frontend1	52.78.200.134	DNS only	Auto	Edit

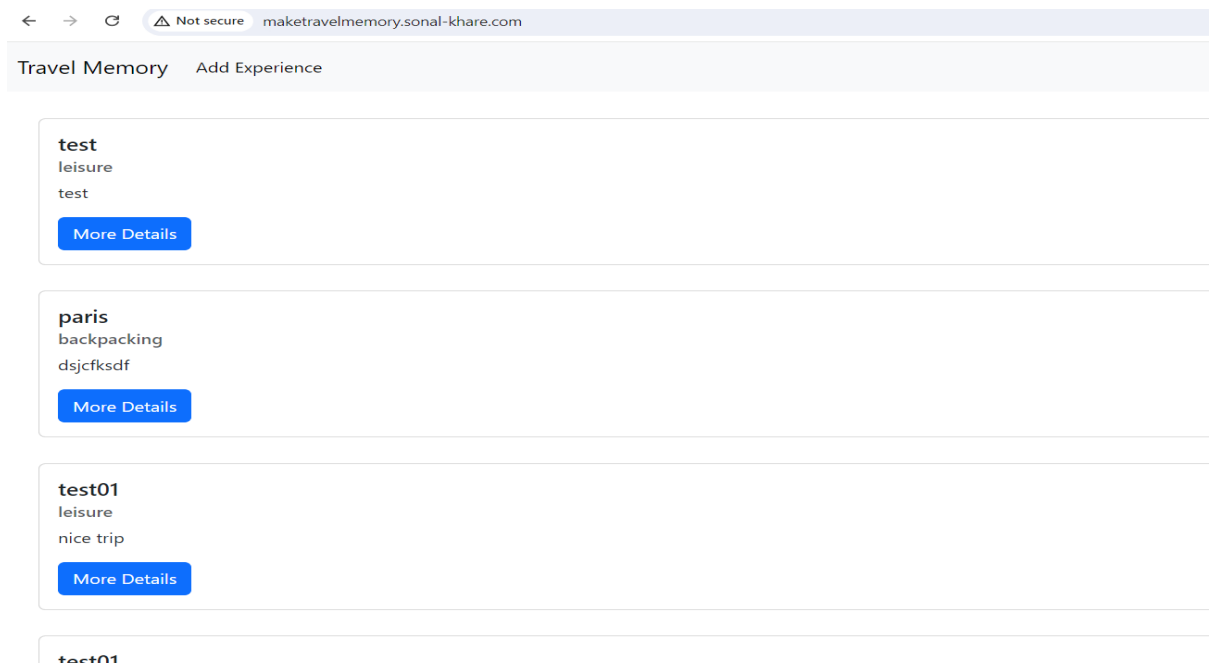
Step6: Send <http://maketravelmemory.sonal-khare.com> via browser or any web client multiple times:

Load balancer will redirect the request to front end instances in round robin manner.

a) First it will send the request to first frontend server.



b) Second request will be send to the second frontend server.



<https://github.com/sonalbatch5/TravelMemory/>