

# TravelMemory application has been developed using the MERN stack

## Tasks:

### 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.

### EC2 instance creation:

EC2 > Instances > Launch an instance

**Name and tags** [Info](#)

Name

TM-aks-1

Add additional tags

**▼ Application and OS Images (Amazon Machine Image)** [Info](#)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Q Search our full catalog including 1000s of application and OS images

Recents

My AMIs

Quick Start

Amazon Linux

aws

macOS

Mac

Ubuntu

ubuntu

Windows

Microsoft

Red Hat

Red Hat

SUSE Linux

SUSE

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

**Amazon Machine Image (AMI)**

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type

Free tier eligible

ami-0866a3c8686eaeeba (64-bit x86) / ami-0325498274077fac5 (64-bit (Arm))

Virtualization: hvm ENA enabled: true Root device type: ebs

**Description**

Ubuntu Server 24.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (<http://www.ubuntu.com/cloud/services>).

Canonical, Ubuntu, 24.04, amd64 noble image

Architecture

64-bit (x86)

AMI ID

ami-0866a3c8686eaeeba

Username

ubuntu

Verified provider

**▼ Summary**

Number of instances

1

Software Image (AMI)

Canonical, Ubuntu, 24.04, amd64...[read more](#)

ami-0866a3c8686eaeeba

Virtual server type (instance type)

t3.micro

Firewall (security group)

Aks-SecurityGroup

Storage (volumes)

1 volume(s) - 8 GiB

Free tier: In your first year includes 750 hours of t2.micro (or t3.micro in the Regions in which t2.micro is unavailable) instance usage on free tier AMIs per month, 750 hours of public IPv4 address usage per month, 30 GiB of EBS storage, 2 million IOs, 1 GiB of snapshots, and 100 GiB of bandwidth to the internet.

Cancel

Launch instance

Preview code

Launch an instance

Instance type

t3.micro

Family: t3 2 vCPU 1 GiB Memory Current generation: true  
On-Demand Ubuntu Pro base pricing: 0.0139 USD per Hour  
On-Demand SUSE base pricing: 0.0104 USD per Hour  
On-Demand Linux base pricing: 0.0104 USD per Hour  
On-Demand RHEL base pricing: 0.0392 USD per Hour  
On-Demand Windows base pricing: 0.0196 USD per Hour

☒ All generations

[Compare instance types](#)

Additional costs apply for AMIs with pre-installed software

▼ Key pair (login) [Info](#)

You can use a key pair to securely connect to your instance. Ensure that you have access to the selected key pair before you launch the instance.

Key pair name - required

EC2-AMI-Aks-HV

[Create new key pair](#)

▼ Network settings [Info](#)

[Edit](#)

Network [Info](#)

vpc-09f02049d6176fe30 | dev\_stage\_vpc

Subnet [Info](#)

subnet-01874c4512136bd62 | az-1

Auto-assign public IP [Info](#)

Disable

Firewall (security groups) [Info](#)

A security group is a set of firewall rules that control the traffic for your instance. Add rules to allow specific traffic to reach your instance.

☐ Create security group

☒ Select existing security group

Common security groups [Info](#)

Select security groups

Aks-SecurityGroup sq-0bc060d9865615fcb

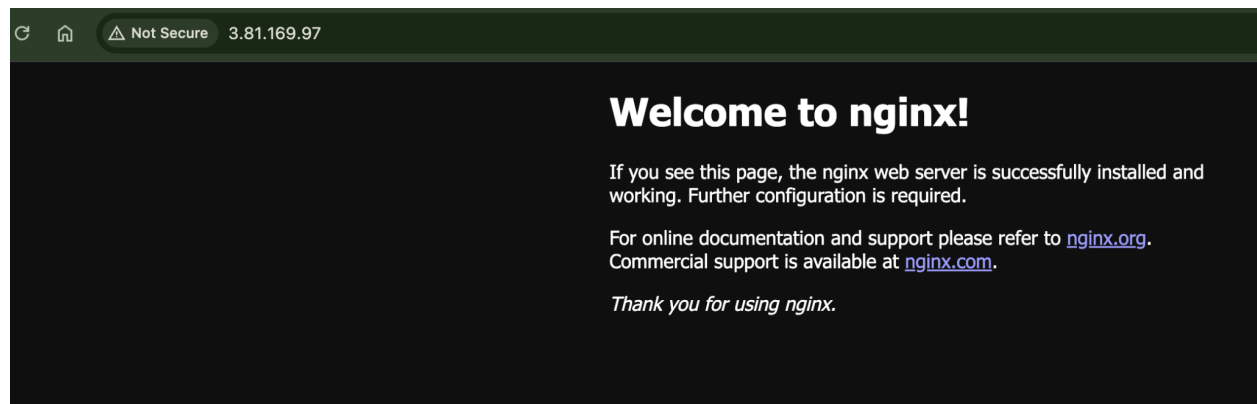
[Compare security group rules](#)

Once instances are running and checks are passed. Install services required and clone git repository:

- `sudo apt update -y`
- `sudo apt install git`
- `git clone https://github.com/UnpredictablePrashant/TravelMemory.git`
- `sudo apt install -y nginx`
- `sudo apt install nodejs`
- `sudo apt install npm`

**Update nginx config as below and validate via new public IP:**

```
server {  
    listen 80;  
    server_name 3.81.169.97;  
    location / {  
        proxy_pass http://127.0.0.1:3000;  
        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;  
    }  
}
```



## Setup Mongodb for this project

- create new database user
- save credentials
- copy connection string for backend folder, file named **.env**:

```
ubuntu@ip-10-0-0-218:~/TravelMemory/backend$ cat .env
PORT=3331
MONGO_URI='mongodb+srv://akshaythebest:aZwsDtRwhc0P3kjB@akshay-1-tm.macm0.mongodb.net/tm-akshay'
# URI / DB
```

Install npm and run “node index.js”

```
ubuntu@ip-10-0-0-218:~/TravelMemory/backend$
ubuntu@ip-10-0-0-218:~/TravelMemory/backend$ npm instal

added 117 packages, and audited 118 packages in 4s

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

13 vulnerabilities (3 low, 1 moderate, 8 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-10-0-0-218:~/TravelMemory/backend$
ubuntu@ip-10-0-0-218:~/TravelMemory/backend$
ubuntu@ip-10-0-0-218:~/TravelMemory/backend$ node index.js
Server started at http://localhost:3331
```

Verify the status by replacing localhost to EC2 public IP address.

```
ubuntu@ip-10-0-0-218:~/TravelMemory/backend$ cat index.js
const express = require('express')
const cors = require('cors')
require('dotenv').config()

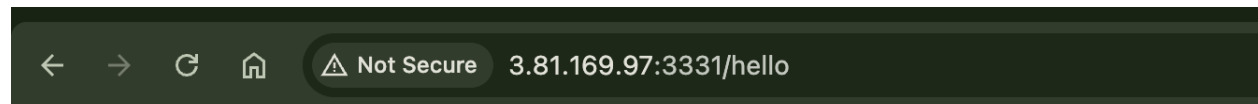
const app = express()
PORT = process.env.PORT
const conn = require('./conn')
app.use(express.json())
app.use(cors())

const tripRoutes = require('./routes/trip.routes')

app.use('/trip', tripRoutes) // http://3.81.169.97:3001/trip --> POST/GET/GET by ID

app.get('/hello', (req,res)=>{
  res.send('Hello World!')
})

app.listen(PORT, ()=>{
  console.log(`Server started at http://3.81.169.97:${PORT}`)
})
ubuntu@ip-10-0-0-218:~/TravelMemory/backend$
```



Hello World!

**The backend is working now.....**

## 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.

Note— Perform the same installation and git clone as performed on backend server as page no 3 on this doc

Apply changes to nginx folder `/etc/nginx/sites-available/default` as did on the backend server.

```
server {  
    listen 80;  
    server_name 3.88.48.97;  
    location / {  
        proxy_pass http://127.0.0.1:3000;  
        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;  
    }  
}
```

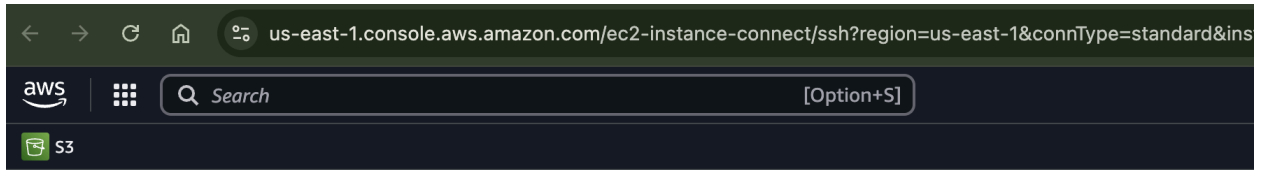
- Modify file *frontend/src/url.js* to point to the backend server IP address:

```
ubuntu@ip-10-0-0-19:~$  
ubuntu@ip-10-0-0-19:~$ cat TravelMemory/frontend/src/url.js  
export const baseUrl = process.env.REACT_APP_BACKEND_URL || "http://3.81.169.97:3331";  
ubuntu@ip-10-0-0-19:~$  
ubuntu@ip-10-0-0-19:~$
```

Run below from frontend path:

Npm install

Npm start



```
Compiled successfully!

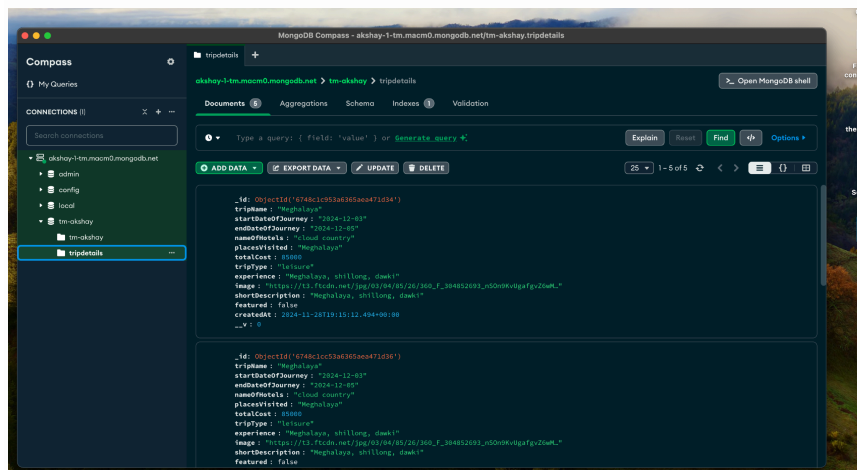
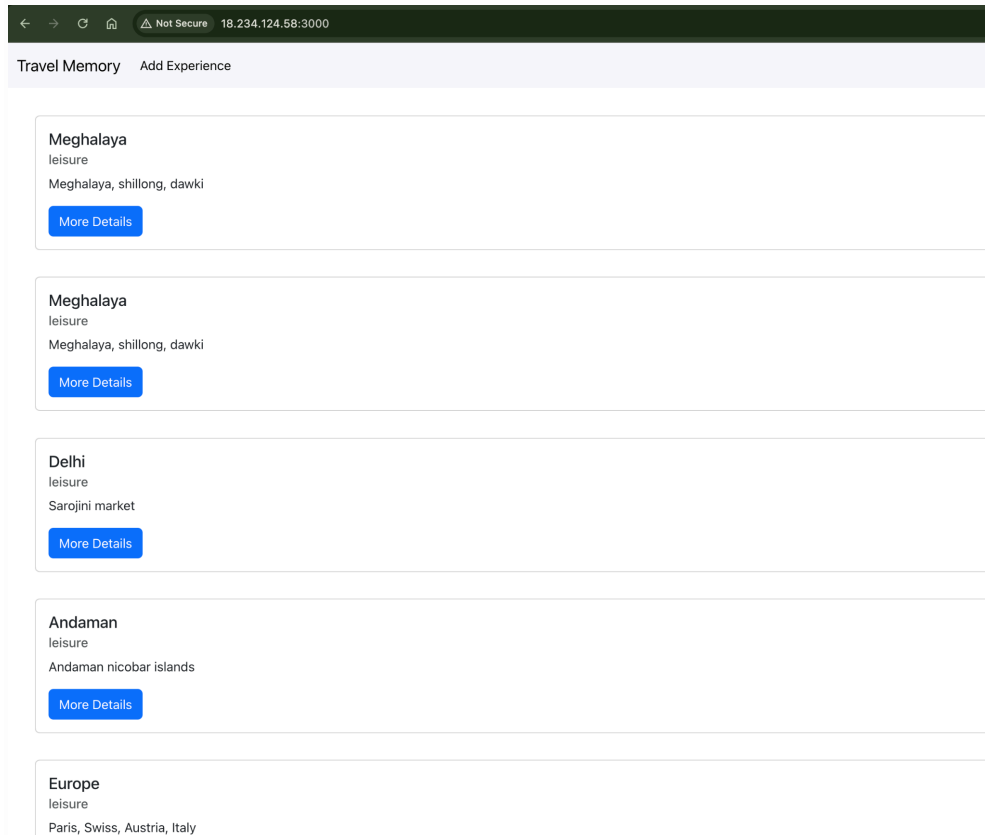
You can now view frontend in the browser.

Local:      http://localhost:3000
On Your Network:  http://10.0.0.105:3000

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

webpack compiled successfully
```

Enter the data in Add experience and save or submit .Create files should appear in mongodb database:



Frontend is connecting to the backend server now.



### 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.

### LoadBalancer creation

**Summary**  
Review and confirm your configurations. [Estimate cost](#)

**Basic configuration** [Edit](#)

FrontendLB-Aks

- Internet-facing
- IPv4

**Security groups** [Edit](#)

- Aks-SecurityGroup  
[sg-0bc060d9865615fcb](#)

**Network mapping** [Edit](#)

VPC: [vpc-09f02049d6176fe30](#)  
dev\_stage\_vpc

- us-east-1a  
[subnet-01874c4512136bd62](#)  
az-1
- us-east-1b  
[subnet-08fa616f96d54dfc2](#)  
az-2

**Listeners and routing** [Edit](#)

- HTTP:80 defaults to  
[frontend-TGs](#)

**Service integrations** [Edit](#)

Amazon CloudFront + AWS Web Application Firewall (WAF): None  
AWS WAF: None  
AWS Global Accelerator: None

**Tags** [Edit](#)

None

**Attributes**

Certain default attributes will be applied to your load balancer. You can view and edit them after creating the load balancer.

**Creation workflow and status**

► **Server-side tasks and status**

After completing and submitting the above steps, all server-side tasks and their statuses become available for monitoring.

[Cancel](#)

Create load balancer

# Adding Targets to frontend LB:

EC2

Target groups

Create target group

Step 1

Specify group details

Step 2

Register targets

Register targets

This is an optional step to create a target group. However, to ensure that your load balancer routes traffic to this target group you must register your targets.

Available instances (2/5)

Filter instances

2 matches

frontend

and

aks

Clear filters

<

1

>

<input checked="" type="checkbox"/>	Instance ID	Name	State	Security groups	Zone	Private IPv4 addr
<input checked="" type="checkbox"/>	i-077d729f3f977941f	frontend-Aks-TM-1	Running	Aks-SecurityGroup	us-east-1a	10.0.0.19
<input checked="" type="checkbox"/>	i-0e75718edde510ba3	frontend-Aks-TM-2	Running	Aks-SecurityGroup	us-east-1a	10.0.0.105

2 selected

Ports for the selected instances

Ports for routing traffic to the selected instances.

80

1-65535 (separate multiple ports with comma)

Include as pending below

Review targets

**Purchased a domain and created CNAME to point frontend loadbalancer:**

The screenshot shows the BigRock web interface for managing DNS records. The browser address bar displays `myorders.bigrock.in/orders/manage/akshay2devops.online/domain`. The left sidebar contains navigation links: HOME, ORDERS, BILLING, MY BUSINESS, PARTNER BENEFITS, and TRANSACTIONS. The main content area is titled "MANAGE DNS RECORDS" for the domain `akshay2devops.online`. It features tabs for ALL RECORDS, A, AAAA, MX, CNAME, NS, TXT, SRV, and SOA. The CNAME tab is active, showing a table with one record:

NAME	IN	CNAME	SUB-DOMAIN/FULLY QUALIFIED DOMAIN*
akshay2devops.online		CNAME	frontendlb-aks-1343385166.us-east-1.elb.am...

Below the table, the TTL (Time to Live) is set to 11 hours. A note states: "These changes will take 4-6 hours to come into effect." Buttons for "Save" and "Cancel" are visible on the right.

**Validate the check on purchased domain name:**

2:42

🕒 📶 📶 📶 79%



⚠️ akshay2devops.online



13



## Travel Memory



### Meghalaya

leisure

Meghalaya, shillong, dawki

[More Details](#)

### Meghalaya

leisure

Meghalaya, shillong, dawki

[More Details](#)

### Delhi

leisure

Sarojini market

[More Details](#)

### Andaman

leisure

Andaman nicobar islands

[More Details](#)

### Europe

leisure

Paris, Swiss, Austria, Italy

[More Details](#)



