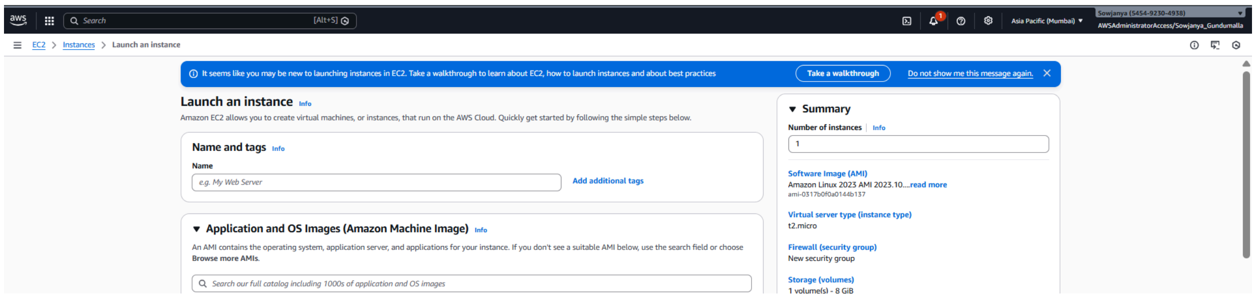
Mydomain: studiomoonbear.com
Loadbalancer: http://travelmemory-alb-1207660565.ap-south-1.elb.amazonaws.com/

BACKEND:

- The Travel Memory backend repository was cloned onto the EC2 instance and dependencies were installed.
- Environment variables were configured to connect the backend to MongoDB Atlas and define the application port.
- The backend server was started and verified to be running on the configured port.
- PM2 was used to manage the backend process to ensure continuous operation.
- Nginx was configured as a reverse proxy to forward incoming HTTP requests to the backend service.
- The backend API was successfully tested using the /hello endpoint.


Search [Alt+S]
Asia Pacific (Mumbai)
Sowjanya (5454-9230-4938)
AWSAdministratorAccess/Sowjanya_Gundumalla
EC2 > Instances > Launch an instance
Take a walkthrough
Do not show me this message again.
Info
Add additional tags

It seems like you may be new to launching instances in EC2. Take a walkthrough to learn about EC2, how to launch instances and about best practices

Launch an instance

Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags

Name: e.g. My Web Server

Application and OS Images (Amazon Machine Image)

An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search field or choose Browse more AMIs.

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

Summary

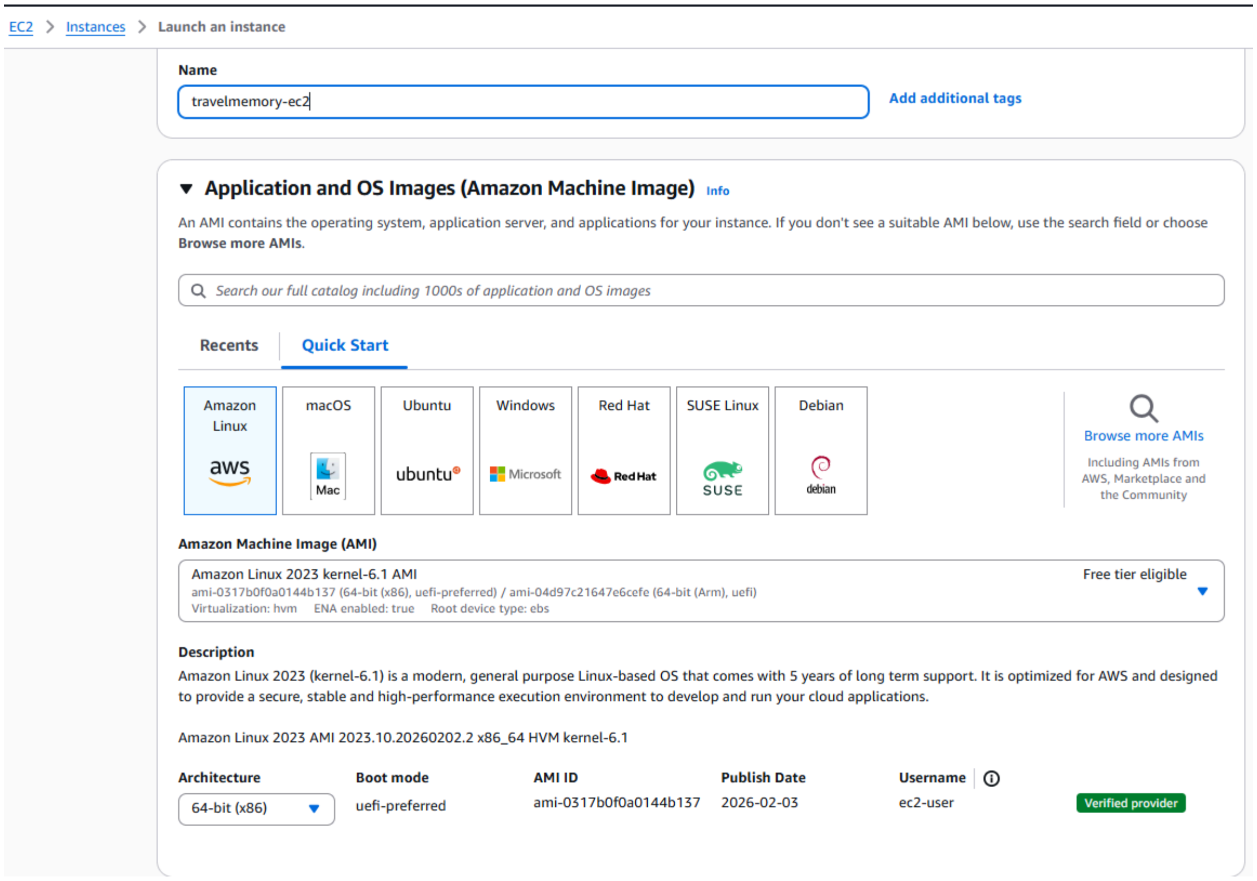
Number of instances: 1

Software Image (AMI): Amazon Linux 2023 AMI 2023.10....read more ami-0317b0f0a0144b137

Virtual server type (instance type): t2.micro

Firewall (security group): New security group

Storage (volumes): 1 volume(s) - 8 GiB



EC2 > Instances > Launch an instance

Name

travelmemory-ec2

Add additional tags

Application and OS Images (Amazon Machine Image) Info

An AMI contains the operating system, application server, and applications for your instance. If you don't see a suitable AMI below, use the search field or choose **Browse more AMIs**.

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

Recents | Quick Start

Amazon Linux | macOS | Ubuntu | Windows | Red Hat | SUSE Linux | Debian

Browse more AMIs

Including AMIs from AWS, Marketplace and the Community

Amazon Machine Image (AMI)

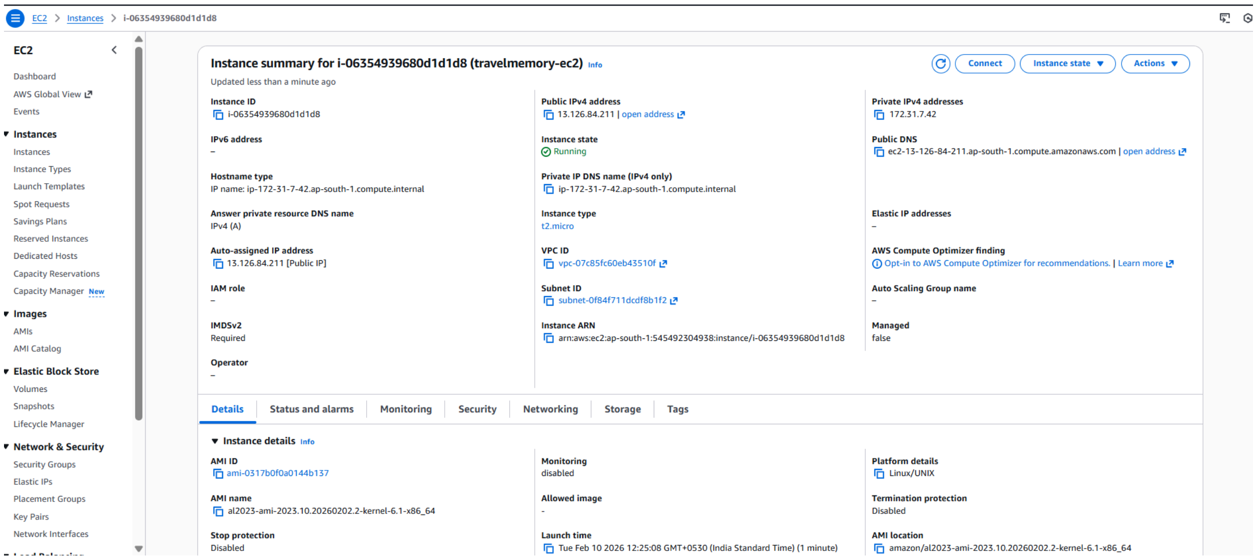
Amazon Linux 2023 kernel-6.1 AMI — Free tier eligible
ami-0317b0f0a0144b137 (64-bit (x86), uefi-preferred) / ami-04d97c21647e6cefe (64-bit (Arm), uefi)
Virtualization: hvm ENA enabled: true Root device type: ebs

Description

Amazon Linux 2023 (kernel-6.1) is a modern, general purpose Linux-based OS that comes with 5 years of long term support. It is optimized for AWS and designed to provide a secure, stable and high-performance execution environment to develop and run your cloud applications.

Amazon Linux 2023 AMI 2023.10.20260202.2 x86_64 HVM kernel-6.1

Architecture	Boot mode	AMI ID	Publish Date	Username	
64-bit (x86)	uefi-preferred	ami-0317b0f0a0144b137	2026-02-03	ec2-user	Verified provider



EC2 > Instances > i-06354939680d1d1d8

EC2

Dashboard
AWS Global View
Events

Instances: Instances, Instance Types, Launch Templates, Spot Requests, Savings Plans, Reserved Instances, Dedicated Hosts, Capacity Reservations, Capacity Manager New

Images: AMIs, AMI Catalog

Elastic Block Store: Volumes, Snapshots, Lifecycle Manager

Network & Security: Security Groups, Elastic IPs, Placement Groups, Key Pairs, Network Interfaces

Instance summary for i-06354939680d1d1d8 (travelmemory-ec2) Info

Connect | Instance state | Actions

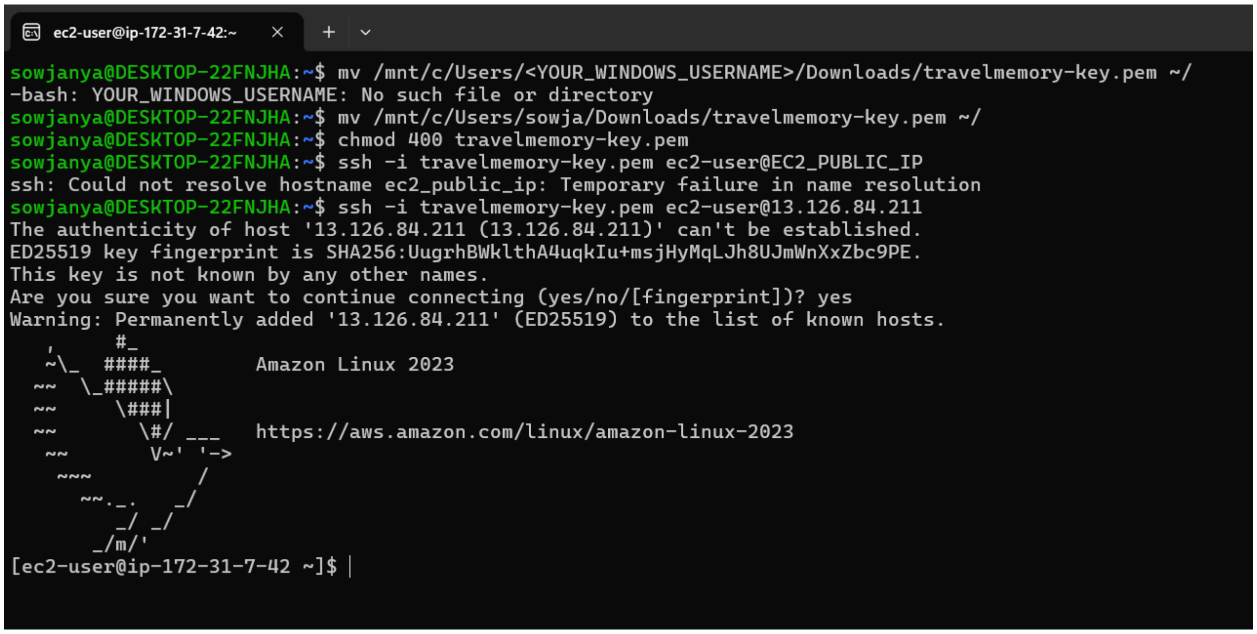
Updated less than a minute ago

Instance ID i-06354939680d1d1d8	**Public IPv4 address** 13.126.84.211 | open address	**Private IPv4 addresses** 172.31.7.42
IPv6 address –	**Instance state** Running	**Public DNS** ec2-13-126-84-211.ap-south-1.compute.amazonaws.com | open address
Hostname type IP name: ip-172-31-7-42.ap-south-1.compute.internal	**Private IP DNS name (IPv4 only)** ip-172-31-7-42.ap-south-1.compute.internal	
Answer private resource DNS name IPv4 (A)	**Instance type** t2.micro	**Elastic IP addresses** –
Auto-assigned IP address 13.126.84.211 [Public IP]	**VPC ID** vpc-07c85fc60eb43510f	**AWS Compute Optimizer finding** Opt-in to AWS Compute Optimizer for recommendations. | Learn more
IAM role –	**Subnet ID** subnet-0f84f711dcdf8b1f2	**Auto Scaling Group name** –
IMDSv2 Required	**Instance ARN** arn:aws:ec2:ap-south-1:545492304938:instance/i-06354939680d1d1d8	**Managed** false
Operator –		

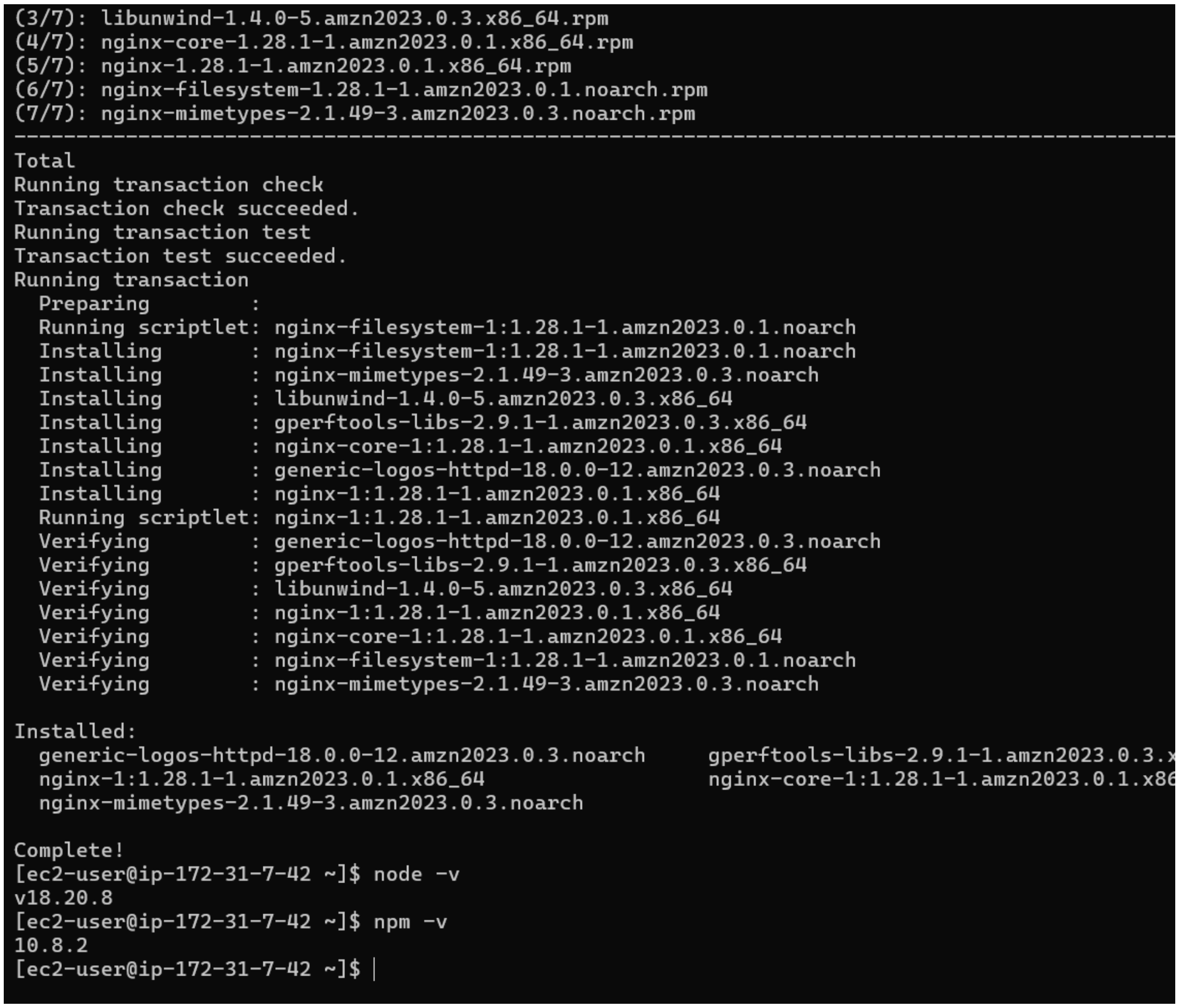
Details | Status and alarms | Monitoring | Security | Networking | Storage | Tags

Instance details Info

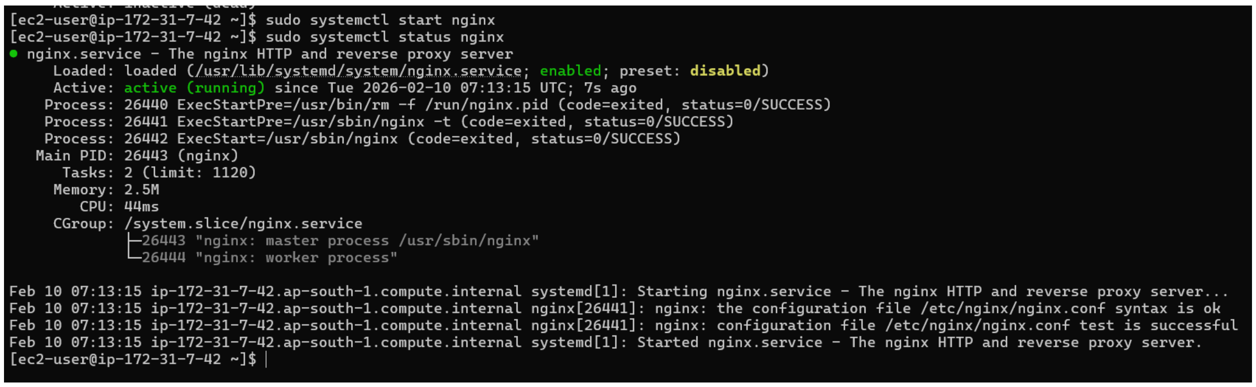
AMI ID ami-0317b0f0a0144b137	**Monitoring** disabled	**Platform details** Linux/UNIX
AMI name al2023-ami-2023.10.20260202.2-kernel-6.1-x86_64	**Allowed image** -	**Termination protection** Disabled
Stop protection Disabled	**Launch time** Tue Feb 10 2026 12:25:08 GMT+0530 (India Standard Time) (1 minute)	**AMI location** amazon/al2023-ami-2023.10.20260202.2-kernel-6.1-x86_64



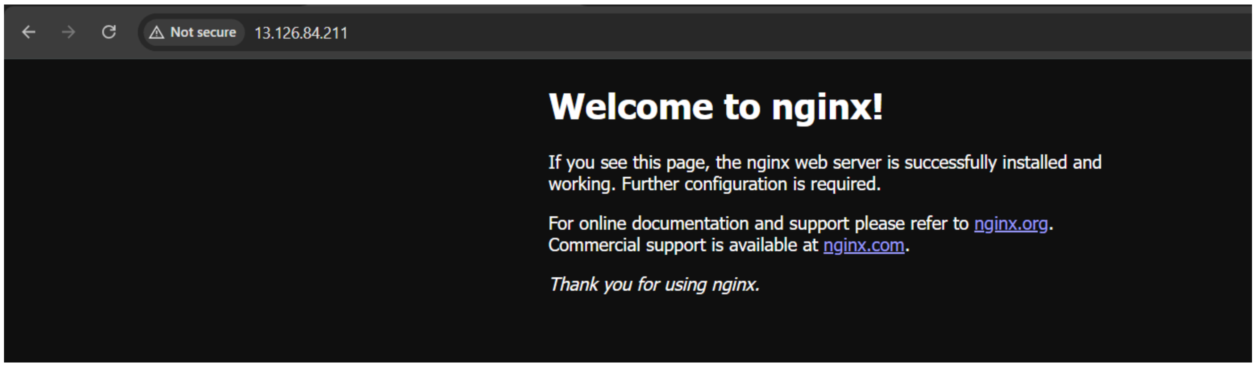
```
sowjanya@DESKTOP-22FNJHA:~$ mv /mnt/c/Users/<YOUR_WINDOWS_USERNAME>/Downloads/travelmemory-key.pem ~/
-bash: YOUR_WINDOWS_USERNAME: No such file or directory
sowjanya@DESKTOP-22FNJHA:~$ mv /mnt/c/Users/sowja/Downloads/travelmemory-key.pem ~/
sowjanya@DESKTOP-22FNJHA:~$ chmod 400 travelmemory-key.pem
sowjanya@DESKTOP-22FNJHA:~$ ssh -i travelmemory-key.pem ec2-user@EC2_PUBLIC_IP
ssh: Could not resolve hostname ec2_public_ip: Temporary failure in name resolution
sowjanya@DESKTOP-22FNJHA:~$ ssh -i travelmemory-key.pem ec2-user@13.126.84.211
The authenticity of host '13.126.84.211 (13.126.84.211)' can't be established.
ED25519 key fingerprint is SHA256:UugrhBWklthA4uqkIu+msjHyMqLJh8UJmWnXxZbc9PE.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added '13.126.84.211' (ED25519) to the list of known hosts.
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-31-7-42 ~]$
```



```
(3/7): libunwind-1.4.0-5.amzn2023.0.3.x86_64.rpm
(4/7): nginx-core-1.28.1-1.amzn2023.0.1.x86_64.rpm
(5/7): nginx-1.28.1-1.amzn2023.0.1.x86_64.rpm
(6/7): nginx-filesystem-1.28.1-1.amzn2023.0.1.noarch.rpm
(7/7): nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch.rpm
--------------------------------------------------------------------------------------------
Total
Running transaction check
Transaction check succeeded.
Running transaction test
Transaction test succeeded.
Running transaction
  Preparing        :
  Running scriptlet: nginx-filesystem-1:1.28.1-1.amzn2023.0.1.noarch
  Installing       : nginx-filesystem-1:1.28.1-1.amzn2023.0.1.noarch
  Installing       : nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch
  Installing       : libunwind-1.4.0-5.amzn2023.0.3.x86_64
  Installing       : gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64
  Installing       : nginx-core-1:1.28.1-1.amzn2023.0.1.x86_64
  Installing       : generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch
  Installing       : nginx-1:1.28.1-1.amzn2023.0.1.x86_64
  Running scriptlet: nginx-1:1.28.1-1.amzn2023.0.1.x86_64
  Verifying        : generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch
  Verifying        : gperftools-libs-2.9.1-1.amzn2023.0.3.x86_64
  Verifying        : libunwind-1.4.0-5.amzn2023.0.3.x86_64
  Verifying        : nginx-1:1.28.1-1.amzn2023.0.1.x86_64
  Verifying        : nginx-core-1:1.28.1-1.amzn2023.0.1.x86_64
  Verifying        : nginx-filesystem-1:1.28.1-1.amzn2023.0.1.noarch
  Verifying        : nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch

Installed:
  generic-logos-httpd-18.0.0-12.amzn2023.0.3.noarch     gperftools-libs-2.9.1-1.amzn2023.0.3.x
  nginx-1:1.28.1-1.amzn2023.0.1.x86_64                  nginx-core-1:1.28.1-1.amzn2023.0.1.x86
  nginx-mimetypes-2.1.49-3.amzn2023.0.3.noarch

Complete!
[ec2-user@ip-172-31-7-42 ~]$ node -v
v18.20.8
[ec2-user@ip-172-31-7-42 ~]$ npm -v
10.8.2
[ec2-user@ip-172-31-7-42 ~]$ |
```



```
     Active: inactive (dead)
[ec2-user@ip-172-31-7-42 ~]$ sudo systemctl start nginx
[ec2-user@ip-172-31-7-42 ~]$ sudo systemctl status nginx
● nginx.service - The nginx HTTP and reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled)
     Active: active (running) since Tue 2026-02-10 07:13:15 UTC; 7s ago
    Process: 26440 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited, status=0/SUCCESS)
    Process: 26441 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
    Process: 26442 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
   Main PID: 26443 (nginx)
      Tasks: 2 (limit: 1120)
     Memory: 2.5M
        CPU: 44ms
     CGroup: /system.slice/nginx.service
             ├─26443 "nginx: master process /usr/sbin/nginx"
             └─26444 "nginx: worker process"

Feb 10 07:13:15 ip-172-31-7-42.ap-south-1.compute.internal systemd[1]: Starting nginx.service - The nginx HTTP and reverse proxy server...
Feb 10 07:13:15 ip-172-31-7-42.ap-south-1.compute.internal nginx[26441]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
Feb 10 07:13:15 ip-172-31-7-42.ap-south-1.compute.internal nginx[26441]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Feb 10 07:13:15 ip-172-31-7-42.ap-south-1.compute.internal systemd[1]: Started nginx.service - The nginx HTTP and reverse proxy server.
[ec2-user@ip-172-31-7-42 ~]$ |
```



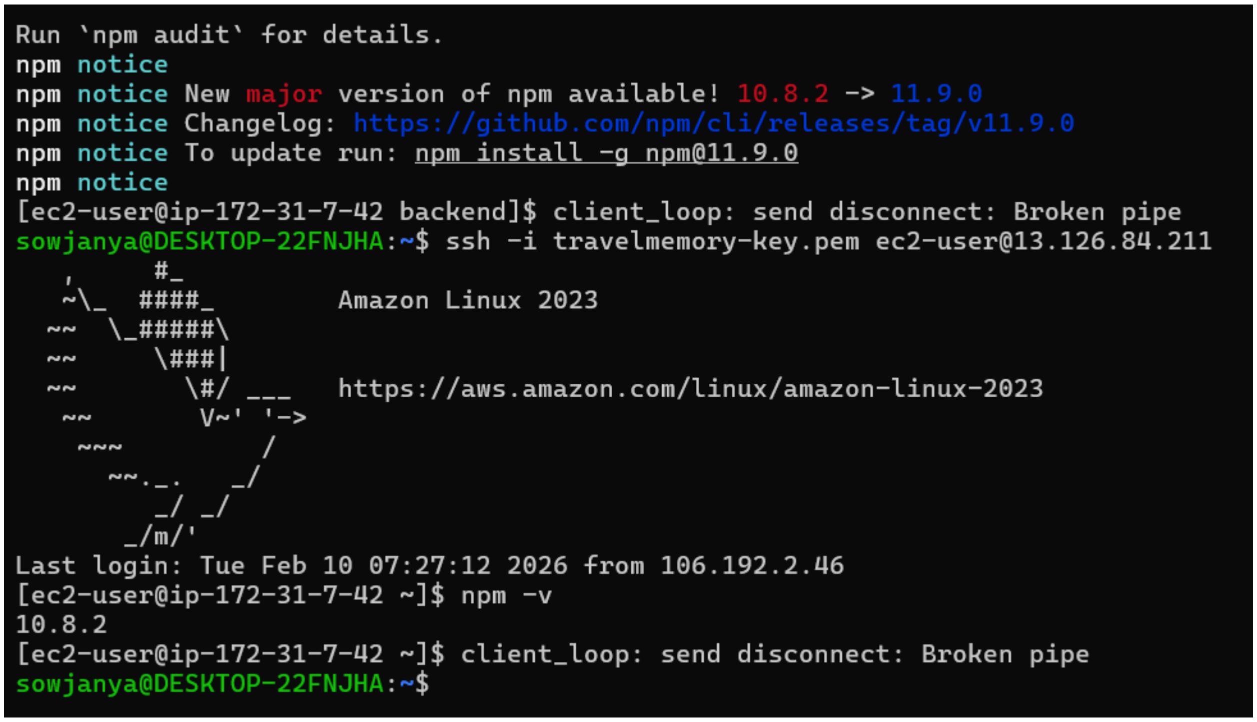
Not secure 13.126.84.211

Welcome to nginx!

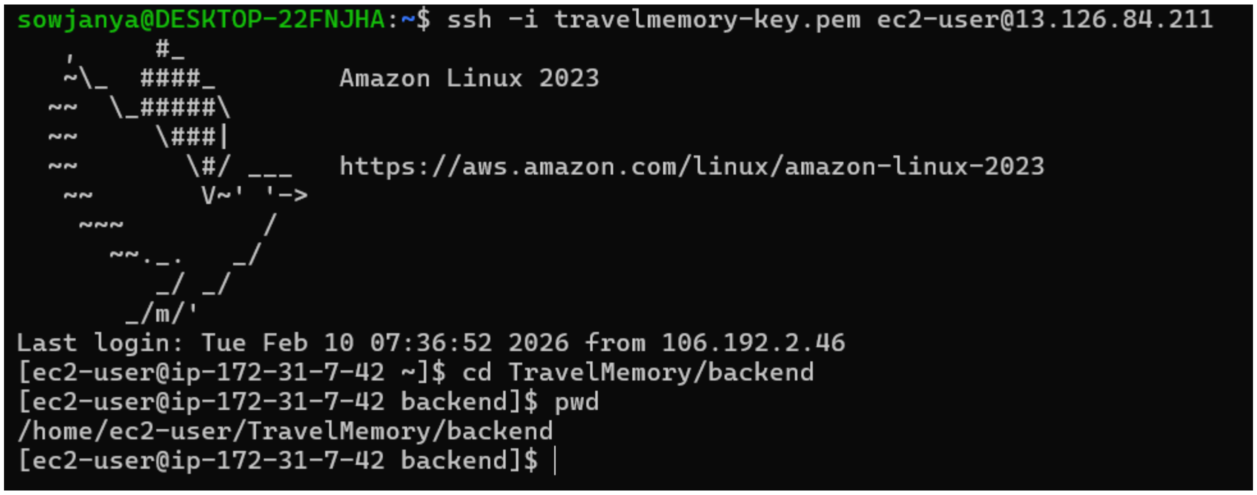
If you see this page, the nginx web server is successfully installed and working. Further configuration is required.

For online documentation and support please refer to nginx.org.
Commercial support is available at nginx.com.

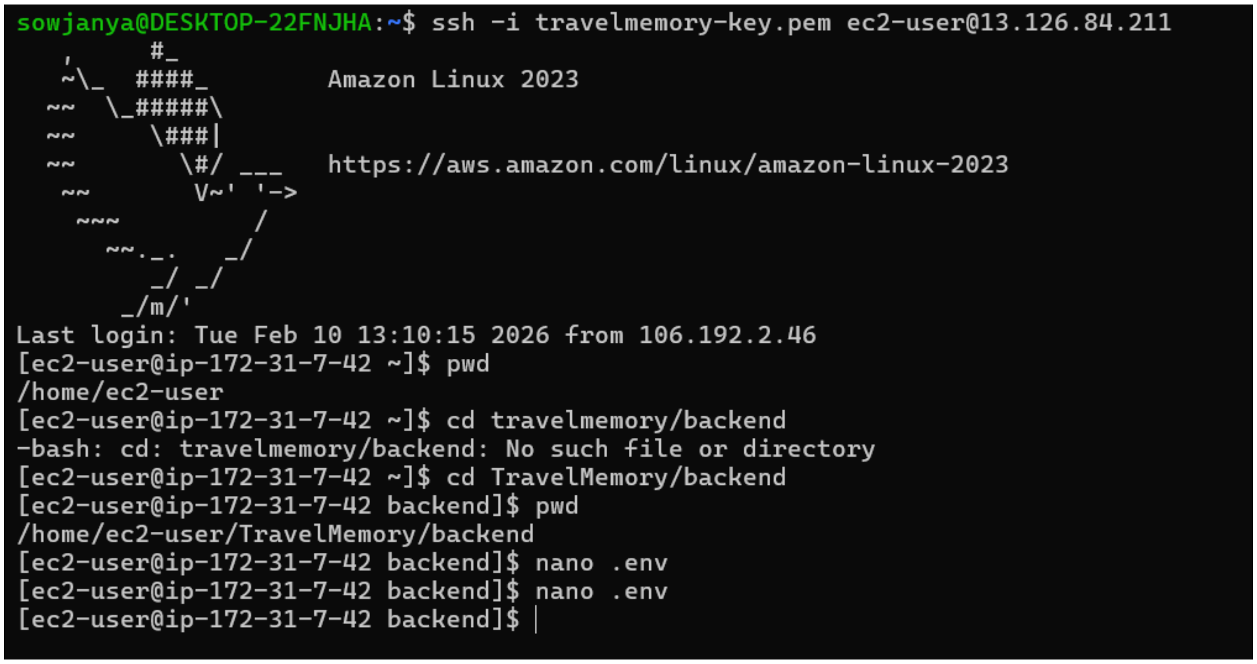
Thank you for using nginx.



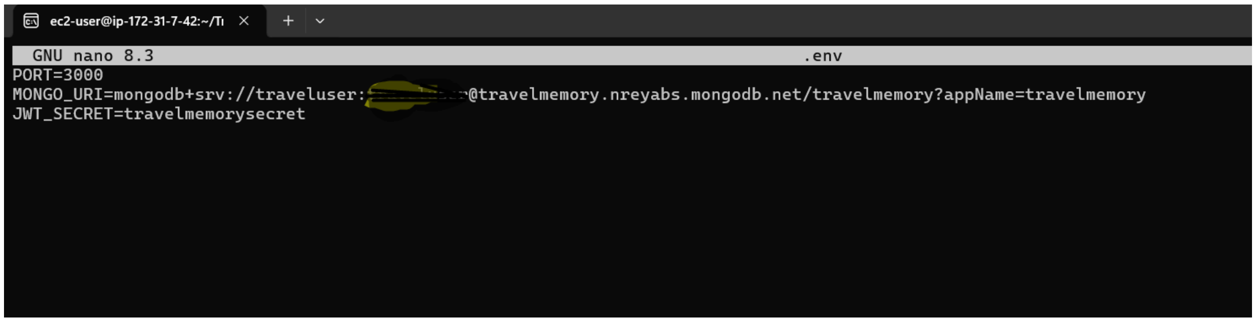
```
Run `npm audit` for details.
npm notice
npm notice New major version of npm available! 10.8.2 -> 11.9.0
npm notice Changelog: https://github.com/npm/cli/releases/tag/v11.9.0
npm notice To update run: npm install -g npm@11.9.0
npm notice
[ec2-user@ip-172-31-7-42 backend]$ client_loop: send disconnect: Broken pipe
sowjanya@DESKTOP-22FNJHA:~$ ssh -i travelmemory-key.pem ec2-user@13.126.84.211
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Tue Feb 10 07:27:12 2026 from 106.192.2.46
[ec2-user@ip-172-31-7-42 ~]$ npm -v
10.8.2
[ec2-user@ip-172-31-7-42 ~]$ client_loop: send disconnect: Broken pipe
sowjanya@DESKTOP-22FNJHA:~$
```



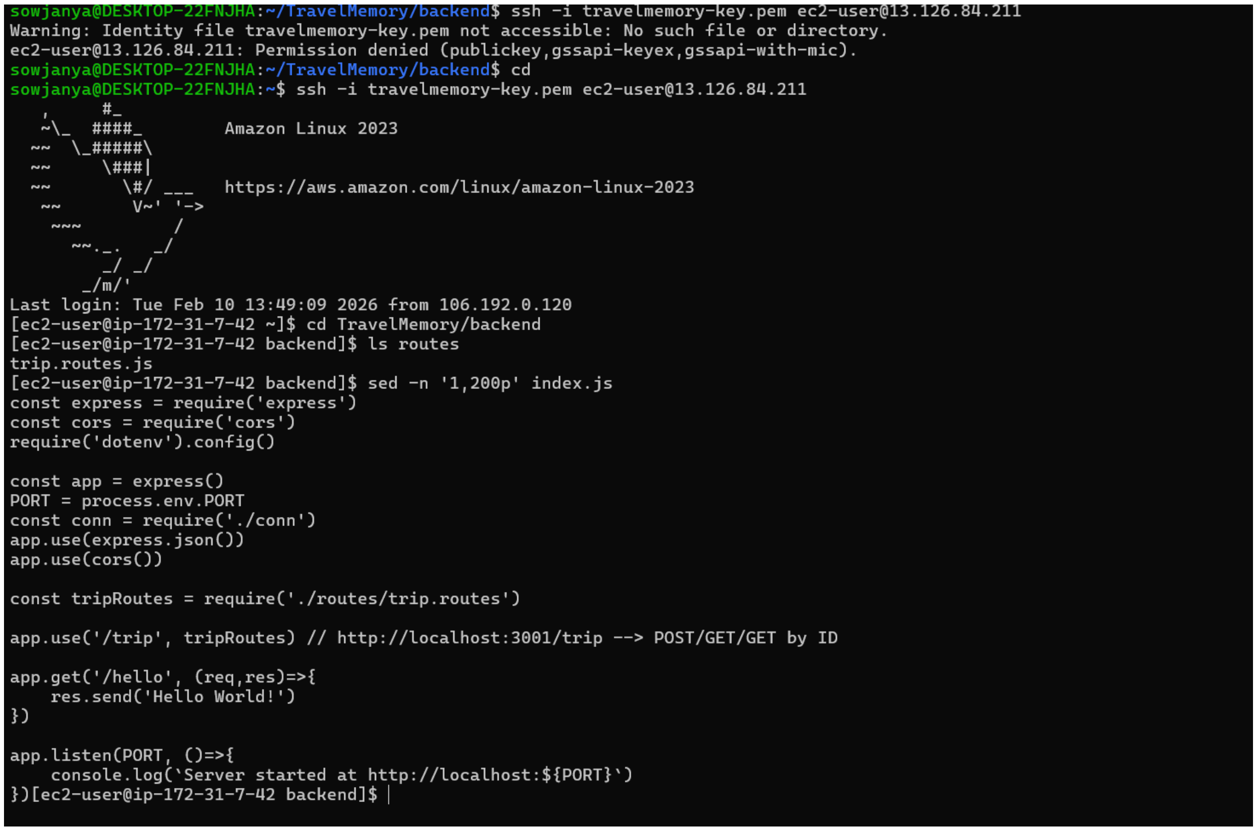
```
sowjanya@DESKTOP-22FNJHA:~$ ssh -i travelmemory-key.pem ec2-user@13.126.84.211
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Tue Feb 10 07:36:52 2026 from 106.192.2.46
[ec2-user@ip-172-31-7-42 ~]$ cd TravelMemory/backend
[ec2-user@ip-172-31-7-42 backend]$ pwd
/home/ec2-user/TravelMemory/backend
[ec2-user@ip-172-31-7-42 backend]$ 
```



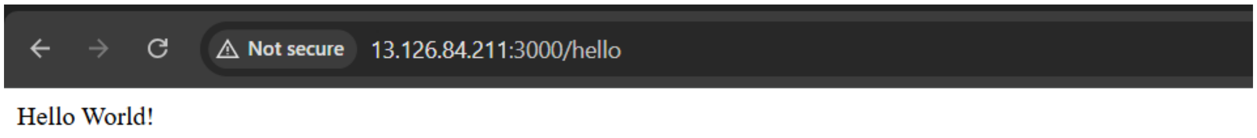
```
sowjanya@DESKTOP-22FNJHA:~$ ssh -i travelmemory-key.pem ec2-user@13.126.84.211
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Tue Feb 10 13:10:15 2026 from 106.192.2.46
[ec2-user@ip-172-31-7-42 ~]$ pwd
/home/ec2-user
[ec2-user@ip-172-31-7-42 ~]$ cd travelmemory/backend
-bash: cd: travelmemory/backend: No such file or directory
[ec2-user@ip-172-31-7-42 ~]$ cd TravelMemory/backend
[ec2-user@ip-172-31-7-42 backend]$ pwd
/home/ec2-user/TravelMemory/backend
[ec2-user@ip-172-31-7-42 backend]$ nano .env
[ec2-user@ip-172-31-7-42 backend]$ nano .env
[ec2-user@ip-172-31-7-42 backend]$ 
```



```
  GNU nano 8.3                                                             .env
PORT=3000
MONGO_URI=mongodb+srv://traveluser:[redacted]@travelmemory.nreyabs.mongodb.net/travelmemory?appName=travelmemory
JWT_SECRET=travelmemorysecret
```

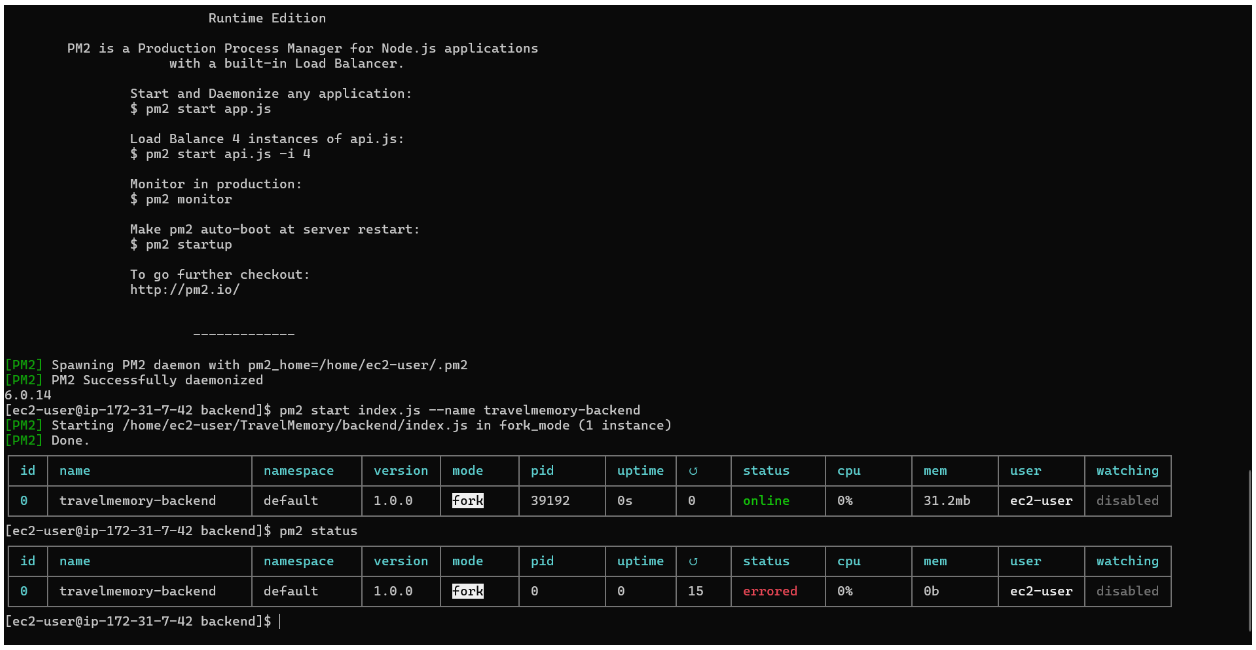


```
sowjanya@DESKTOP-22FNJHA:~/TravelMemory/backend$ ssh -i travelmemory-key.pem ec2-user@13.126.84.211
Warning: Identity file travelmemory-key.pem not accessible: No such file or directory.
ec2-user@13.126.84.211: Permission denied (publickey,gssapi-keyex,gssapi-with-mic).
sowjanya@DESKTOP-22FNJHA:~/TravelMemory/backend$ cd
sowjanya@DESKTOP-22FNJHA:~$ ssh -i travelmemory-key.pem ec2-user@13.126.84.211
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Tue Feb 10 13:49:09 2026 from 106.192.0.120
[ec2-user@ip-172-31-7-42 ~]$ cd TravelMemory/backend
[ec2-user@ip-172-31-7-42 backend]$ ls routes
trip.routes.js
[ec2-user@ip-172-31-7-42 backend]$ sed -n '1,200p' 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://localhost: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://localhost:${PORT}`)
})[ec2-user@ip-172-31-7-42 backend]$
```

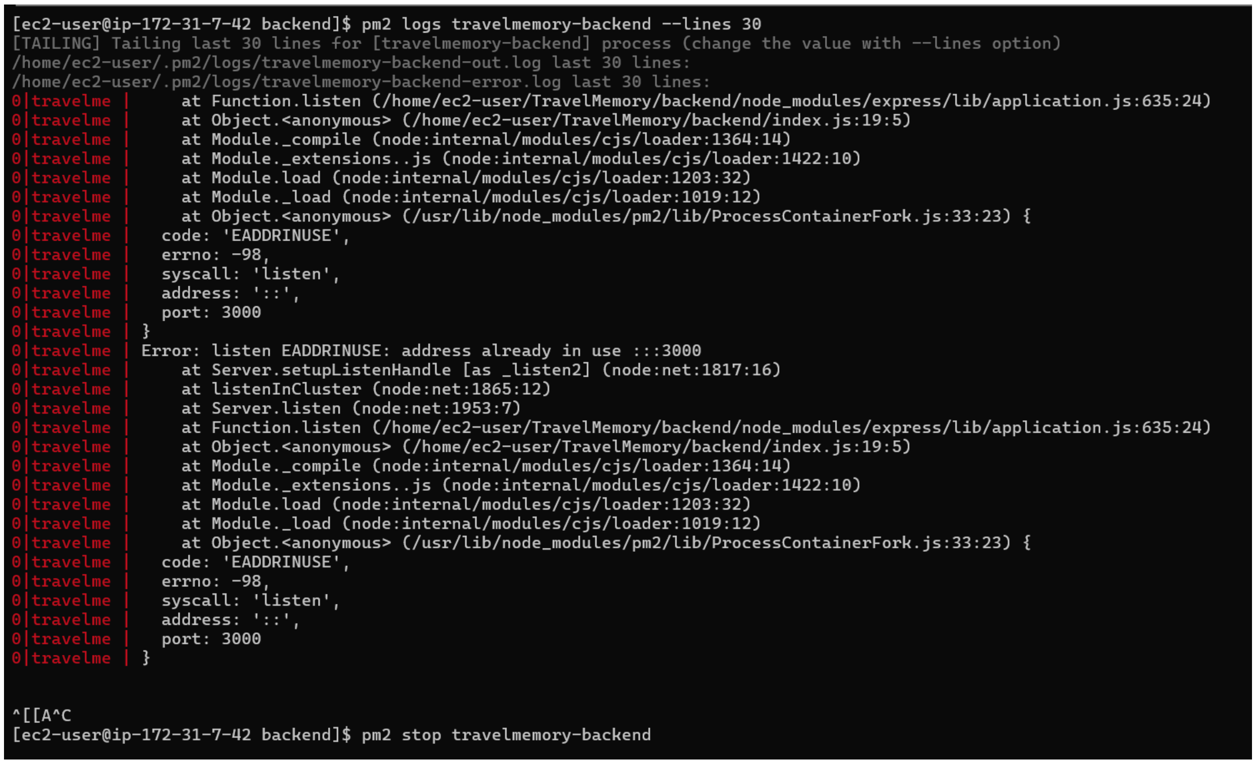


13.126.84.211:3000/hello

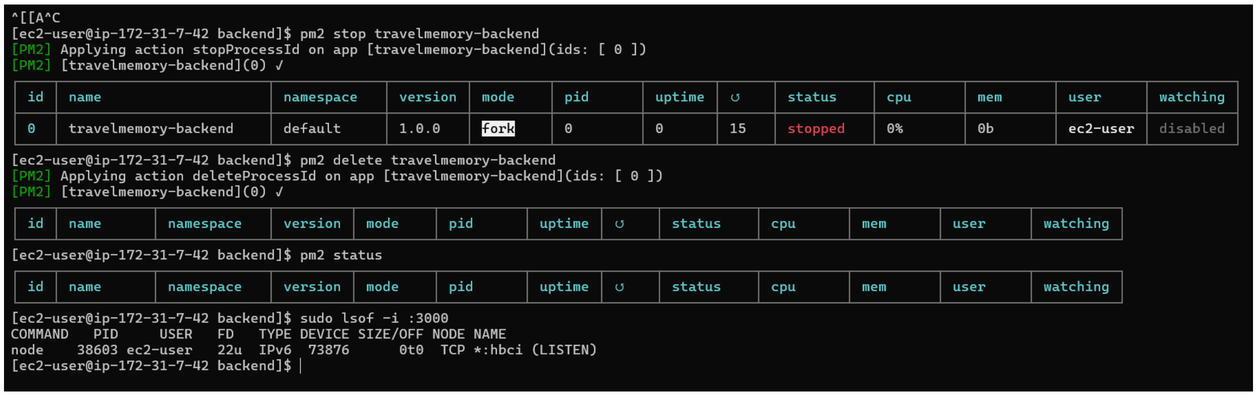
Hello World!



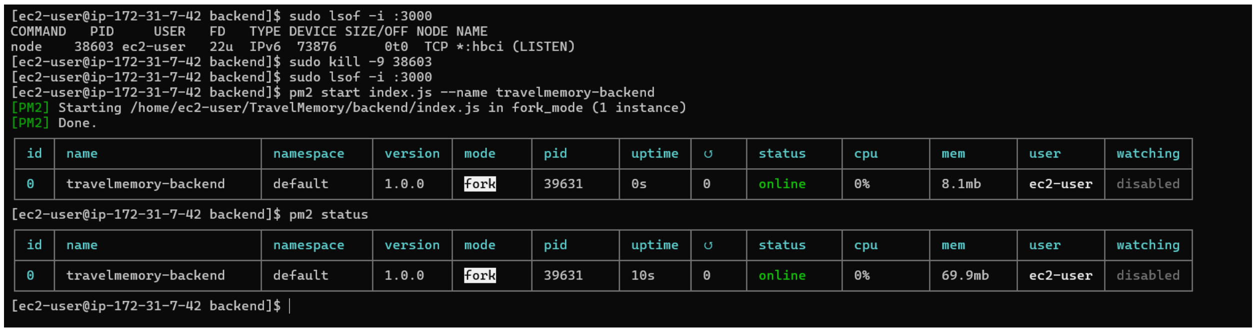
```
                          Runtime Edition

        PM2 is a Production Process Manager for Node.js applications
                     with a built-in Load Balancer.

                Start and Daemonize any application:
                $ pm2 start app.js

                Load Balance 4 instances of api.js:
                $ pm2 start api.js -i 4

                Monitor in production:
                $ pm2 monitor

                Make pm2 auto-boot at server restart:
                $ pm2 startup

                To go further checkout:
                http://pm2.io/


                        -------------

[PM2] Spawning PM2 daemon with pm2_home=/home/ec2-user/.pm2
[PM2] PM2 Successfully daemonized
6.0.14
[ec2-user@ip-172-31-7-42 backend]$ pm2 start index.js --name travelmemory-backend
[PM2] Starting /home/ec2-user/TravelMemory/backend/index.js in fork_mode (1 instance)
[PM2] Done.
│ id │ name                 │ namespace │ version │ mode │ pid   │ uptime │ ↺  │ status  │ cpu │ mem    │ user     │ watching │
│ 0  │ travelmemory-backend │ default   │ 1.0.0   │ fork │ 39192 │ 0s     │ 0  │ online  │ 0%  │ 31.2mb │ ec2-user │ disabled │
[ec2-user@ip-172-31-7-42 backend]$ pm2 status
│ id │ name                 │ namespace │ version │ mode │ pid   │ uptime │ ↺  │ status  │ cpu │ mem    │ user     │ watching │
│ 0  │ travelmemory-backend │ default   │ 1.0.0   │ fork │ 0     │ 0      │ 15 │ errored │ 0%  │ 0b     │ ec2-user │ disabled │
[ec2-user@ip-172-31-7-42 backend]$ |
```



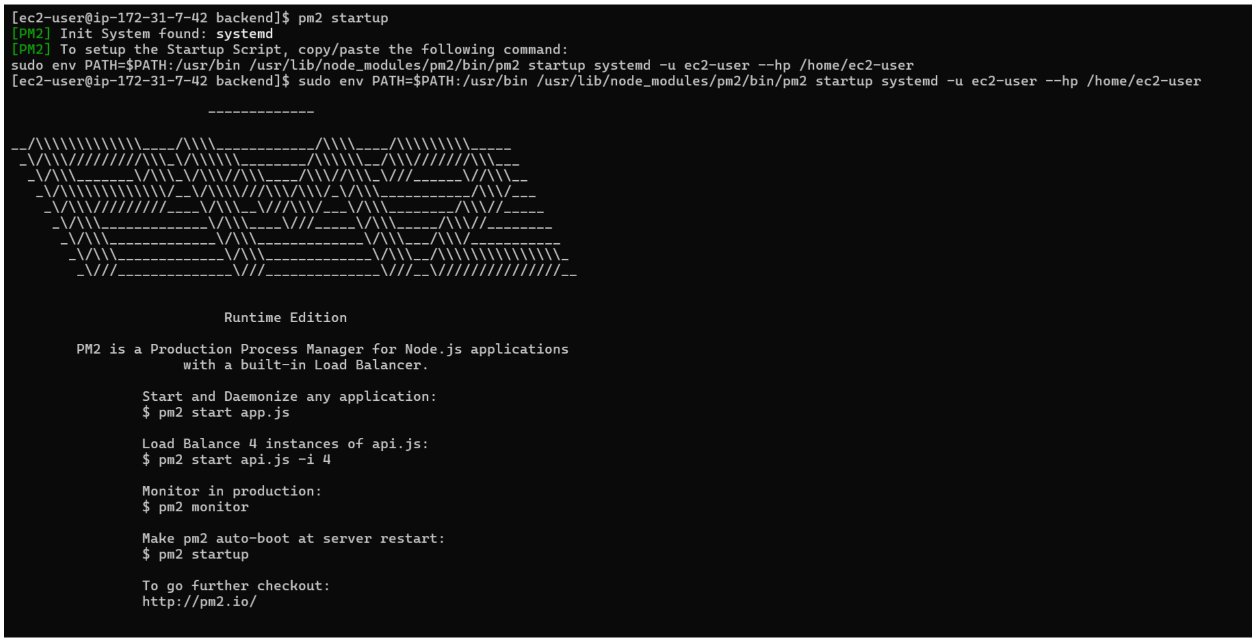
```
[ec2-user@ip-172-31-7-42 backend]$ pm2 logs travelmemory-backend --lines 30
[TAILING] Tailing last 30 lines for [travelmemory-backend] process (change the value with --lines option)
/home/ec2-user/.pm2/logs/travelmemory-backend-out.log last 30 lines:
/home/ec2-user/.pm2/logs/travelmemory-backend-error.log last 30 lines:
0|travelme |     at Function.listen (/home/ec2-user/TravelMemory/backend/node_modules/express/lib/application.js:635:24)
0|travelme |     at Object.<anonymous> (/home/ec2-user/TravelMemory/backend/index.js:19:5)
0|travelme |     at Module._compile (node:internal/modules/cjs/loader:1364:14)
0|travelme |     at Module._extensions..js (node:internal/modules/cjs/loader:1422:10)
0|travelme |     at Module.load (node:internal/modules/cjs/loader:1203:32)
0|travelme |     at Module._load (node:internal/modules/cjs/loader:1019:12)
0|travelme |     at Object.<anonymous> (/usr/lib/node_modules/pm2/lib/ProcessContainerFork.js:33:23) {
0|travelme |   code: 'EADDRINUSE',
0|travelme |   errno: -98,
0|travelme |   syscall: 'listen',
0|travelme |   address: '::',
0|travelme |   port: 3000
0|travelme | }
0|travelme | Error: listen EADDRINUSE: address already in use :::3000
0|travelme |     at Server.setupListenHandle [as _listen2] (node:net:1817:16)
0|travelme |     at listenInCluster (node:net:1865:12)
0|travelme |     at Server.listen (node:net:1953:7)
0|travelme |     at Function.listen (/home/ec2-user/TravelMemory/backend/node_modules/express/lib/application.js:635:24)
0|travelme |     at Object.<anonymous> (/home/ec2-user/TravelMemory/backend/index.js:19:5)
0|travelme |     at Module._compile (node:internal/modules/cjs/loader:1364:14)
0|travelme |     at Module._extensions..js (node:internal/modules/cjs/loader:1422:10)
0|travelme |     at Module.load (node:internal/modules/cjs/loader:1203:32)
0|travelme |     at Module._load (node:internal/modules/cjs/loader:1019:12)
0|travelme |     at Object.<anonymous> (/usr/lib/node_modules/pm2/lib/ProcessContainerFork.js:33:23) {
0|travelme |   code: 'EADDRINUSE',
0|travelme |   errno: -98,
0|travelme |   syscall: 'listen',
0|travelme |   address: '::',
0|travelme |   port: 3000
0|travelme | }


^[[A^C
[ec2-user@ip-172-31-7-42 backend]$ pm2 stop travelmemory-backend
```



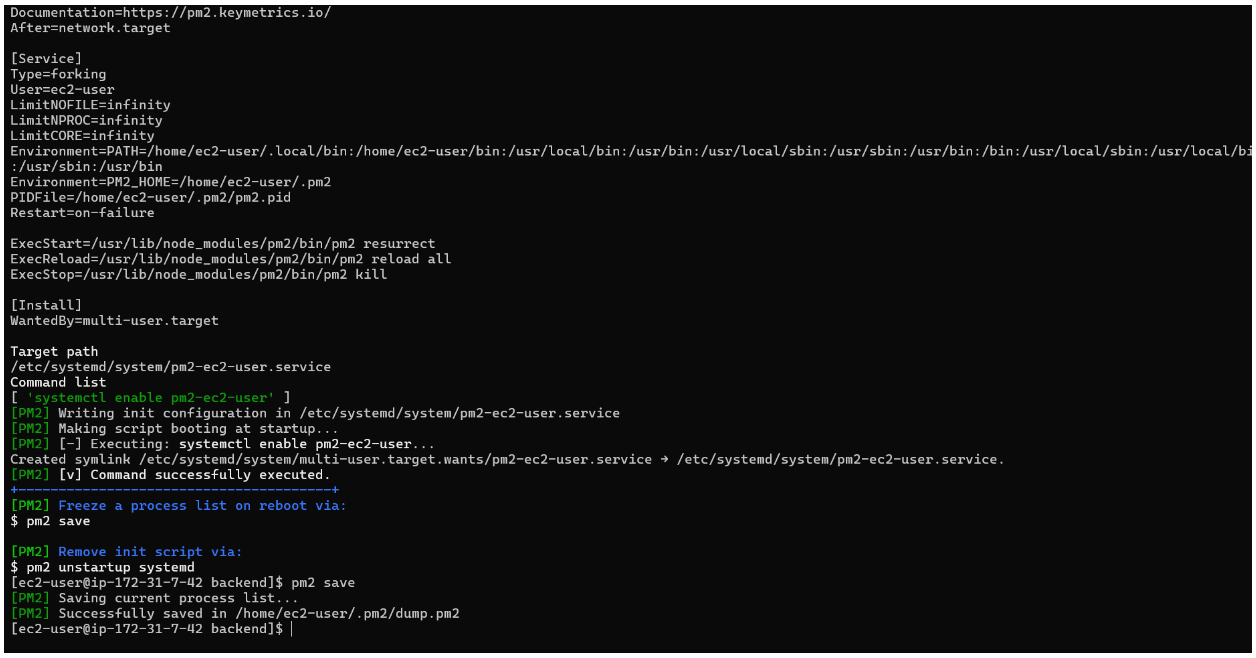
```
^[[A^C
[ec2-user@ip-172-31-7-42 backend]$ pm2 stop travelmemory-backend
[PM2] Applying action stopProcessId on app [travelmemory-backend](ids: [ 0 ])
[PM2] [travelmemory-backend](0) ✓
│ id │ name                 │ namespace │ version │ mode │ pid │ uptime │ ↺  │ status  │ cpu │ mem │ user     │ watching │
│ 0  │ travelmemory-backend │ default   │ 1.0.0   │ fork │ 0   │ 0      │ 15 │ stopped │ 0%  │ 0b  │ ec2-user │ disabled │
[ec2-user@ip-172-31-7-42 backend]$ pm2 delete travelmemory-backend
[PM2] Applying action deleteProcessId on app [travelmemory-backend](ids: [ 0 ])
[PM2] [travelmemory-backend](0) ✓
│ id │ name │ namespace │ version │ mode │ pid │ uptime │ ↺ │ status │ cpu │ mem │ user │ watching │
[ec2-user@ip-172-31-7-42 backend]$ pm2 status
│ id │ name │ namespace │ version │ mode │ pid │ uptime │ ↺ │ status │ cpu │ mem │ user │ watching │
[ec2-user@ip-172-31-7-42 backend]$ sudo lsof -i :3000
COMMAND   PID     USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
node    38603 ec2-user   22u  IPv6  73876      0t0  TCP *:hbci (LISTEN)
[ec2-user@ip-172-31-7-42 backend]$ |
```



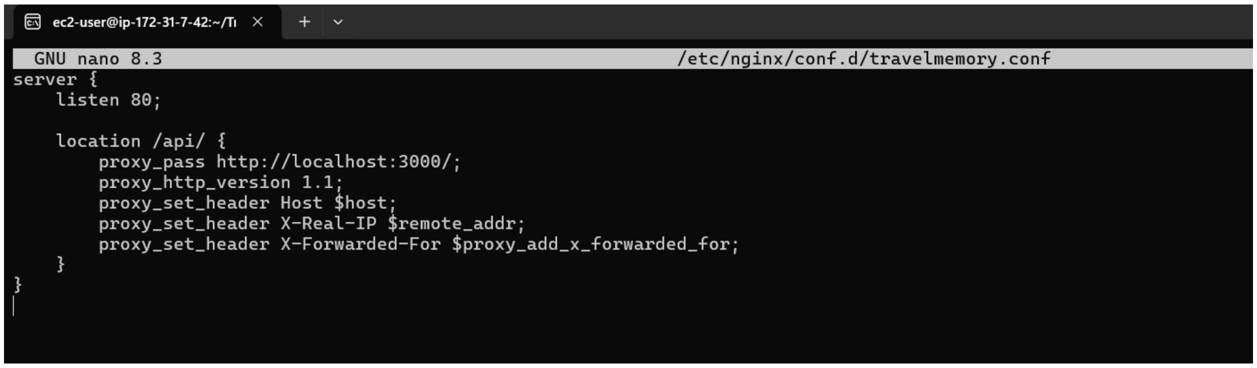
```
[ec2-user@ip-172-31-7-42 backend]$ sudo lsof -i :3000
COMMAND   PID     USER   FD   TYPE DEVICE SIZE/OFF NODE NAME
node    38603 ec2-user   22u  IPv6  73876      0t0  TCP *:hbci (LISTEN)
[ec2-user@ip-172-31-7-42 backend]$ sudo kill -9 38603
[ec2-user@ip-172-31-7-42 backend]$ sudo lsof -i :3000
[ec2-user@ip-172-31-7-42 backend]$ pm2 start index.js --name travelmemory-backend
[PM2] Starting /home/ec2-user/TravelMemory/backend/index.js in fork_mode (1 instance)
[PM2] Done.
│ id │ name                 │ namespace │ version │ mode │ pid   │ uptime │ ↺ │ status │ cpu │ mem    │ user     │ watching │
│ 0  │ travelmemory-backend │ default   │ 1.0.0   │ fork │ 39631 │ 0s     │ 0 │ online │ 0%  │ 8.1mb  │ ec2-user │ disabled │
[ec2-user@ip-172-31-7-42 backend]$ pm2 status
│ id │ name                 │ namespace │ version │ mode │ pid   │ uptime │ ↺ │ status │ cpu │ mem    │ user     │ watching │
│ 0  │ travelmemory-backend │ default   │ 1.0.0   │ fork │ 39631 │ 10s    │ 0 │ online │ 0%  │ 69.9mb │ ec2-user │ disabled │
[ec2-user@ip-172-31-7-42 backend]$ |
```



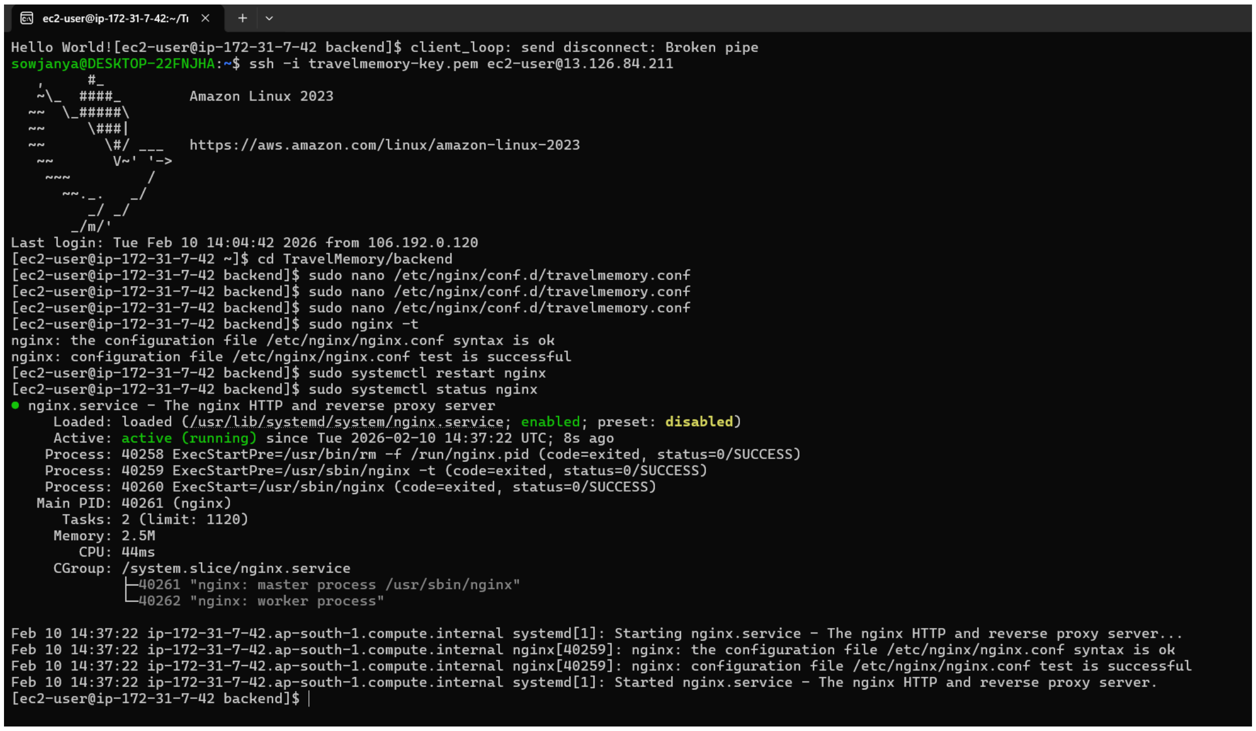
```
[ec2-user@ip-172-31-7-42 backend]$ pm2 startup
[PM2] Init System found: systemd
[PM2] To setup the Startup Script, copy/paste the following command:
sudo env PATH=$PATH:/usr/bin /usr/lib/node_modules/pm2/bin/pm2 startup systemd -u ec2-user --hp /home/ec2-user
[ec2-user@ip-172-31-7-42 backend]$ sudo env PATH=$PATH:/usr/bin /usr/lib/node_modules/pm2/bin/pm2 startup systemd -u ec2-user --hp /home/ec2-user

                        -------------

__/\\\\\\\\\\\\\____/\\\\____________/\\\\____/\\\\\\\\\_____
 _\/\\\/////////\\\_\/\\\\\\________/\\\\\\__/\\\///////\\\___
  _\/\\\_______\/\\\_\/\\\//\\\____/\\\//\\\_\///______\//\\\__
   _\/\\\\\\\\\\\\\/__\/\\\\///\\\/\\\/_\/\\\___________/\\\/___
    _\/\\\/////////____\/\\\__\///\\\/___\/\\\________/\\\//_____
     _\/\\\_____________\/\\\____\///_____\/\\\_____/\\\//________
      _\/\\\_____________\/\\\_____________\/\\\___/\\\/___________
       _\/\\\_____________\/\\\_____________\/\\\__/\\\\\\\\\\\\\\\_
        _\///______________\///______________\///__\///////////////__


                          Runtime Edition

        PM2 is a Production Process Manager for Node.js applications
                     with a built-in Load Balancer.

                Start and Daemonize any application:
                $ pm2 start app.js

                Load Balance 4 instances of api.js:
                $ pm2 start api.js -i 4

                Monitor in production:
                $ pm2 monitor

                Make pm2 auto-boot at server restart:
                $ pm2 startup

                To go further checkout:
                http://pm2.io/
```



```
Documentation=https://pm2.keymetrics.io/
After=network.target

[Service]
Type=forking
User=ec2-user
LimitNOFILE=infinity
LimitNPROC=infinity
LimitCORE=infinity
Environment=PATH=/home/ec2-user/.local/bin:/home/ec2-user/bin:/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/usr/bin:/bin:/usr/local/sbin:/usr/local/bi
:/usr/sbin:/usr/bin
Environment=PM2_HOME=/home/ec2-user/.pm2
PIDFile=/home/ec2-user/.pm2/pm2.pid
Restart=on-failure

ExecStart=/usr/lib/node_modules/pm2/bin/pm2 resurrect
ExecReload=/usr/lib/node_modules/pm2/bin/pm2 reload all
ExecStop=/usr/lib/node_modules/pm2/bin/pm2 kill

[Install]
WantedBy=multi-user.target

Target path
/etc/systemd/system/pm2-ec2-user.service
Command list
[ 'systemctl enable pm2-ec2-user' ]
[PM2] Writing init configuration in /etc/systemd/system/pm2-ec2-user.service
[PM2] Making script booting at startup...
[PM2] [-] Executing: systemctl enable pm2-ec2-user...
Created symlink /etc/systemd/system/multi-user.target.wants/pm2-ec2-user.service → /etc/systemd/system/pm2-ec2-user.service.
[PM2] [v] Command successfully executed.
+---------------------------------------+
[PM2] Freeze a process list on reboot via:
$ pm2 save

[PM2] Remove init script via:
$ pm2 unstartup systemd
[ec2-user@ip-172-31-7-42 backend]$ pm2 save
[PM2] Saving current process list...
[PM2] Successfully saved in /home/ec2-user/.pm2/dump.pm2
[ec2-user@ip-172-31-7-42 backend]$
```



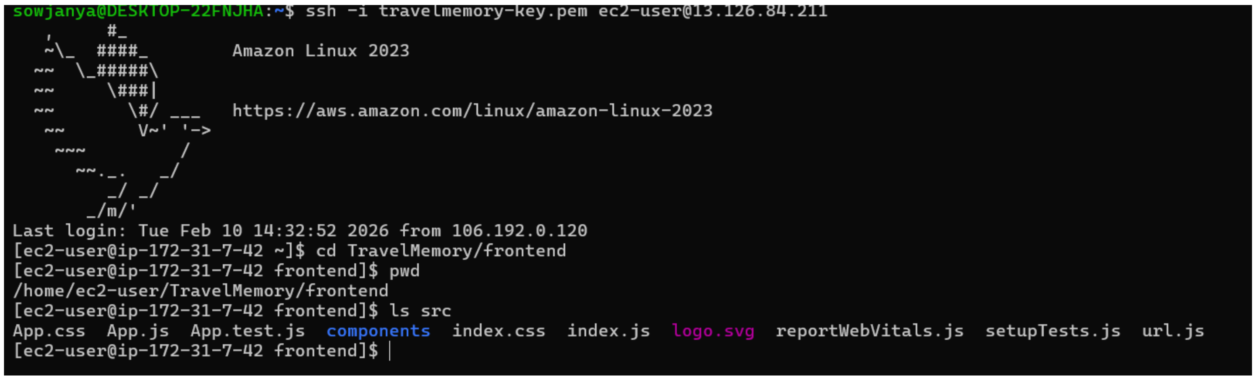
```
ec2-user@ip-172-31-7-42:~/Tı
 GNU nano 8.3                                        /etc/nginx/conf.d/travelmemory.conf
server {
    listen 80;

    location /api/ {
        proxy_pass http://localhost:3000/;
        proxy_http_version 1.1;
        proxy_set_header Host $host;
        proxy_set_header X-Real-IP $remote_addr;
        proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    }
}
```



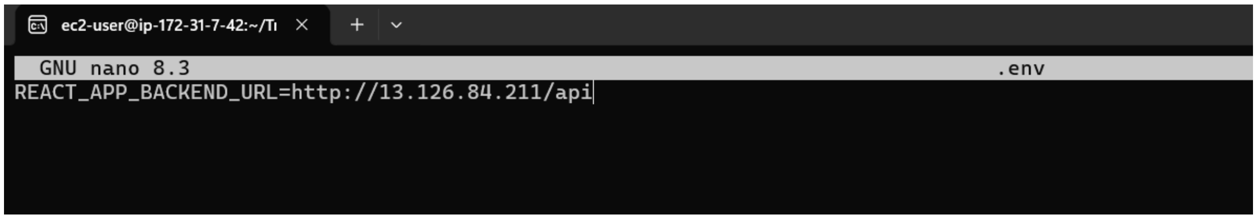
```
Hello World![ec2-user@ip-172-31-7-42 backend]$ client_loop: send disconnect: Broken pipe
sowjanya@DESKTOP-22FNJHA:~$ ssh -i travelmemory-key.pem ec2-user@13.126.84.211
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Tue Feb 10 14:04:42 2026 from 106.192.0.120
[ec2-user@ip-172-31-7-42 ~]$ cd TravelMemory/backend
[ec2-user@ip-172-31-7-42 backend]$ sudo nano /etc/nginx/conf.d/travelmemory.conf
[ec2-user@ip-172-31-7-42 backend]$ sudo nano /etc/nginx/conf.d/travelmemory.conf
[ec2-user@ip-172-31-7-42 backend]$ sudo nano /etc/nginx/conf.d/travelmemory.conf
[ec2-user@ip-172-31-7-42 backend]$ sudo nginx -t
nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
nginx: configuration file /etc/nginx/nginx.conf test is successful
[ec2-user@ip-172-31-7-42 backend]$ sudo systemctl restart nginx
[ec2-user@ip-172-31-7-42 backend]$ sudo systemctl status nginx
● nginx.service - The nginx HTTP and reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled)
     Active: active (running) since Tue 2026-02-10 14:37:22 UTC; 8s ago
    Process: 40258 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited, status=0/SUCCESS)
    Process: 40259 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
    Process: 40260 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
   Main PID: 40261 (nginx)
      Tasks: 2 (limit: 1120)
     Memory: 2.5M
        CPU: 44ms
     CGroup: /system.slice/nginx.service
             ├─40261 "nginx: master process /usr/sbin/nginx"
             └─40262 "nginx: worker process"

Feb 10 14:37:22 ip-172-31-7-42.ap-south-1.compute.internal systemd[1]: Starting nginx.service - The nginx HTTP and reverse proxy server...
Feb 10 14:37:22 ip-172-31-7-42.ap-south-1.compute.internal nginx[40259]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
Feb 10 14:37:22 ip-172-31-7-42.ap-south-1.compute.internal nginx[40259]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Feb 10 14:37:22 ip-172-31-7-42.ap-south-1.compute.internal systemd[1]: Started nginx.service - The nginx HTTP and reverse proxy server.
[ec2-user@ip-172-31-7-42 backend]$ |
```

FRONTEND:

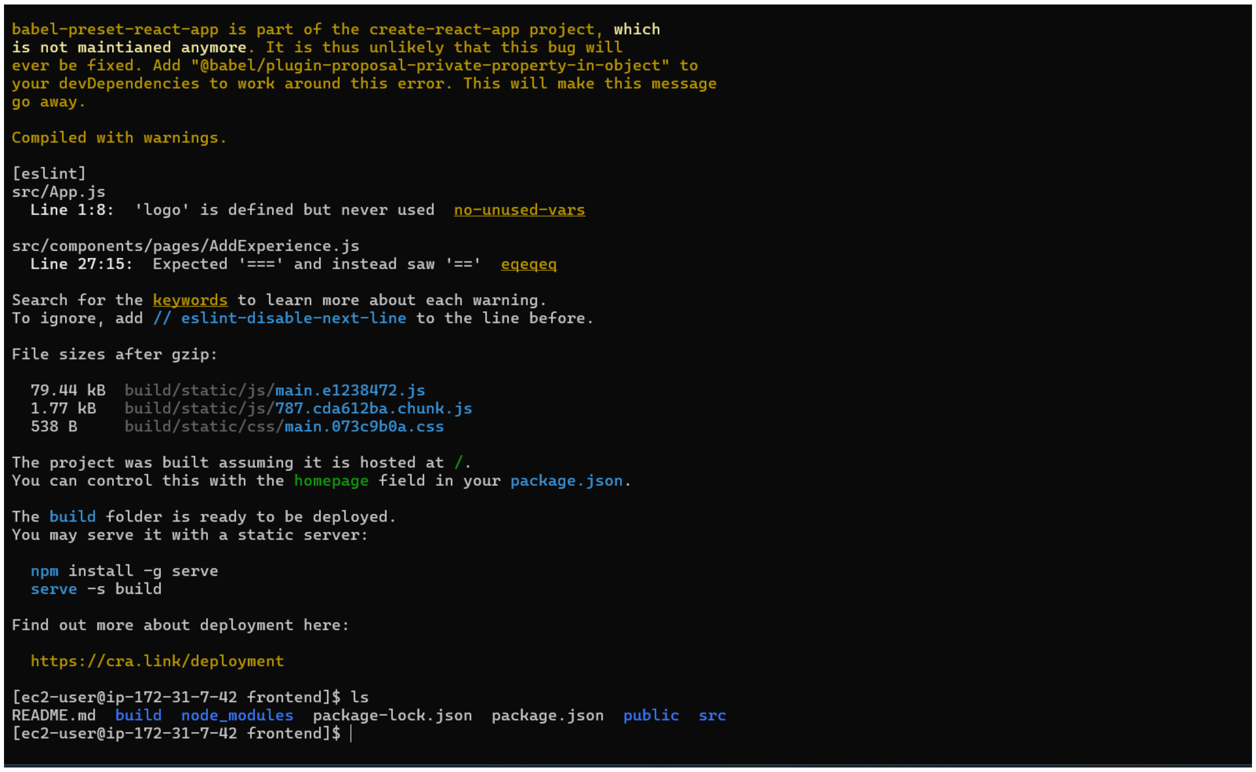
- The frontend application was configured to connect to the deployed backend API.
- A production build of the React application was generated for deployment.
- Nginx was configured to serve the React production build files.
- The frontend application was successfully accessed via the EC2 public IP address.
- Frontend and backend integration was verified through successful API communication.



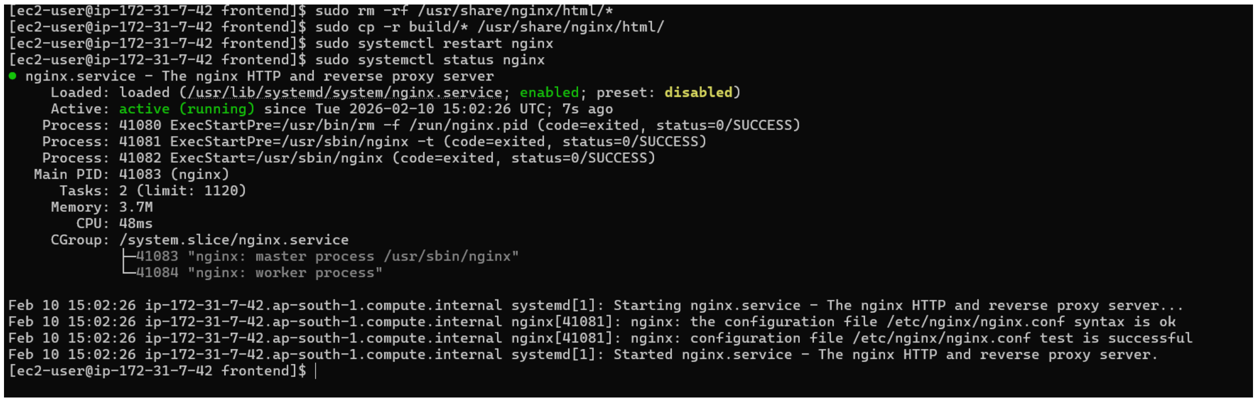
```
sowjanya@DESKTOP-22FNJHA:~$ ssh -i travelmemory-key.pem ec2-user@13.126.84.211
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Tue Feb 10 14:32:52 2026 from 106.192.0.120
[ec2-user@ip-172-31-7-42 ~]$ cd TravelMemory/frontend
[ec2-user@ip-172-31-7-42 frontend]$ pwd
/home/ec2-user/TravelMemory/frontend
[ec2-user@ip-172-31-7-42 frontend]$ ls src
App.css  App.js  App.test.js  components  index.css  index.js  logo.svg  reportWebVitals.js  setupTests.js  url.js
[ec2-user@ip-172-31-7-42 frontend]$ |
```



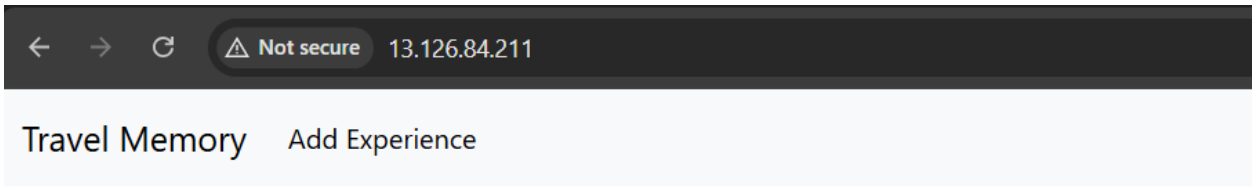
```
  GNU nano 8.3                                                  .env
REACT_APP_BACKEND_URL=http://13.126.84.211/api
```



```
babel-preset-react-app is part of the create-react-app project, which
is not maintianed anymore. It is thus unlikely that this bug will
ever be fixed. Add "@babel/plugin-proposal-private-property-in-object" to
your devDependencies to work around this error. This will make this message
go away.

Compiled with warnings.

[eslint]
src/App.js
  Line 1:8:  'logo' is defined but never used  no-unused-vars

src/components/pages/AddExperience.js
  Line 27:15:  Expected '===' and instead saw '=='  eqeqeq

Search for the keywords to learn more about each warning.
To ignore, add // eslint-disable-next-line to the line before.

File sizes after gzip:

  79.44 kB  build/static/js/main.e1238472.js
  1.77 kB   build/static/js/787.cda612ba.chunk.js
  538 B     build/static/css/main.073c9b0a.css

The project was built assuming it is hosted at /.
You can control this with the homepage field in your package.json.

The build folder is ready to be deployed.
You may serve it with a static server:

  npm install -g serve
  serve -s build

Find out more about deployment here:

  https://cra.link/deployment

[ec2-user@ip-172-31-7-42 frontend]$ ls
README.md  build  node_modules  package-lock.json  package.json  public  src
[ec2-user@ip-172-31-7-42 frontend]$
```

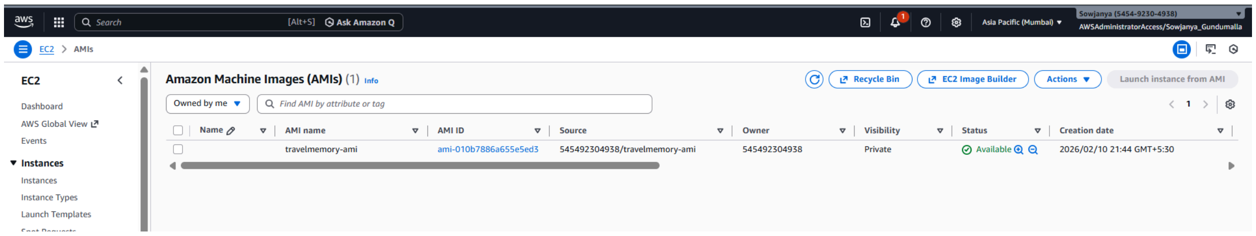


```
[ec2-user@ip-172-31-7-42 frontend]$ sudo rm -rf /usr/share/nginx/html/*
[ec2-user@ip-172-31-7-42 frontend]$ sudo cp -r build/* /usr/share/nginx/html/
[ec2-user@ip-172-31-7-42 frontend]$ sudo systemctl restart nginx
[ec2-user@ip-172-31-7-42 frontend]$ sudo systemctl status nginx
● nginx.service - The nginx HTTP and reverse proxy server
     Loaded: loaded (/usr/lib/systemd/system/nginx.service; enabled; preset: disabled)
     Active: active (running) since Tue 2026-02-10 15:02:26 UTC; 7s ago
    Process: 41080 ExecStartPre=/usr/bin/rm -f /run/nginx.pid (code=exited, status=0/SUCCESS)
    Process: 41081 ExecStartPre=/usr/sbin/nginx -t (code=exited, status=0/SUCCESS)
    Process: 41082 ExecStart=/usr/sbin/nginx (code=exited, status=0/SUCCESS)
   Main PID: 41083 (nginx)
      Tasks: 2 (limit: 1120)
     Memory: 3.7M
        CPU: 48ms
     CGroup: /system.slice/nginx.service
             ├─41083 "nginx: master process /usr/sbin/nginx"
             └─41084 "nginx: worker process"

Feb 10 15:02:26 ip-172-31-7-42.ap-south-1.compute.internal systemd[1]: Starting nginx.service - The nginx HTTP and reverse proxy server...
Feb 10 15:02:26 ip-172-31-7-42.ap-south-1.compute.internal nginx[41081]: nginx: the configuration file /etc/nginx/nginx.conf syntax is ok
Feb 10 15:02:26 ip-172-31-7-42.ap-south-1.compute.internal nginx[41081]: nginx: configuration file /etc/nginx/nginx.conf test is successful
Feb 10 15:02:26 ip-172-31-7-42.ap-south-1.compute.internal systemd[1]: Started nginx.service - The nginx HTTP and reverse proxy server.
[ec2-user@ip-172-31-7-42 frontend]$
```



LOAD BALANCER:

- An Amazon Machine Image (AMI) was created from the configured EC2 instance.
- Multiple EC2 instances were launched from the custom AMI to enable horizontal scaling.
- A Target Group was created and EC2 instances were registered successfully.
- Health checks confirmed that all instances were healthy and ready to receive traffic.
- An Application Load Balancer was configured to distribute incoming requests across instances.
- The application was successfully accessed using the Load Balancer DNS endpoint.



Amazon Machine Images (AMIs) (1) Info

Name	AMI name	AMI ID	Source	Owner	Visibility	Status	Creation date
	travelmemory-ami	ami-010b7886a655e5ed3	545492304938/travelmemory-ami	545492304938	Private	Available	2026/02/10 21:44 GMT+5:30

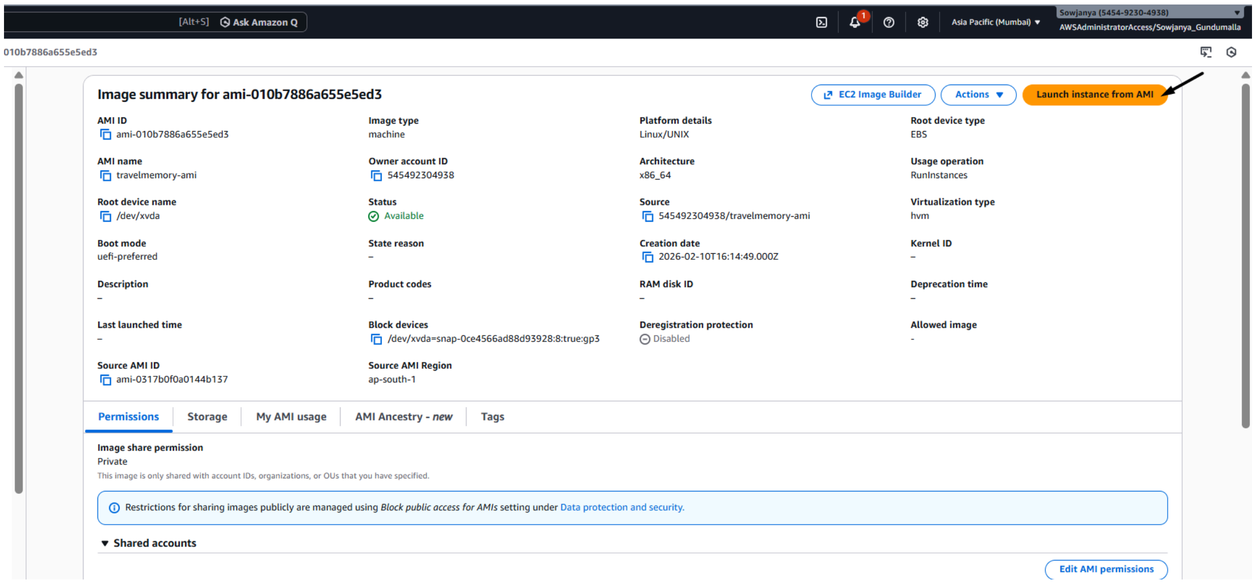


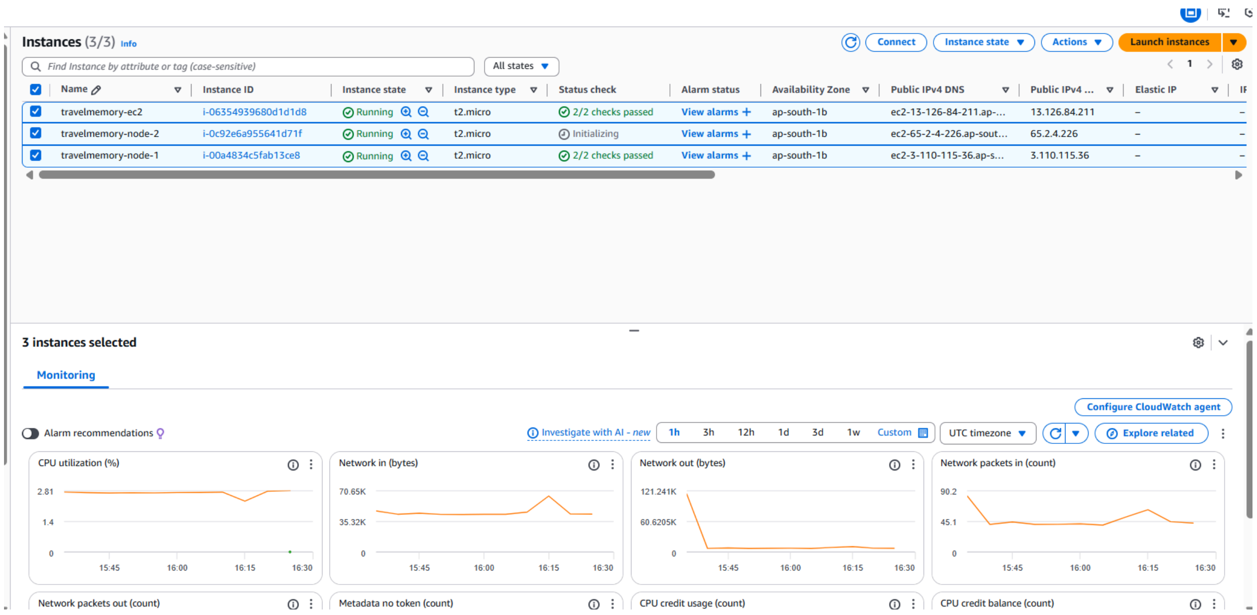
Image summary for ami-010b7886a655e5ed3

AMI ID ami-010b7886a655e5ed3	Image type machine	Platform details Linux/UNIX	Root device type EBS
AMI name travelmemory-ami	Owner account ID 545492304938	Architecture x86_64	Usage operation RunInstances
Root device name /dev/xvda	Status Available	Source 545492304938/travelmemory-ami	Virtualization type hvm
Boot mode uefi-preferred	State reason –	Creation date 2026-02-10T16:14:49.000Z	Kernel ID –
Description –	Product codes –	RAM disk ID –	Deprecation time –
Last launched time –	Block devices /dev/xvda=snap-0ce4566ad88d93928:8:true:gp3	Deregistration protection Disabled	Allowed image -
Source AMI ID ami-0317b0f0a0144b137	Source AMI Region ap-south-1		

Image share permission
Private
This image is only shared with account IDs, organizations, or OUs that you have specified.

Restrictions for sharing images publicly are managed using *Block public access for AMIs* setting under Data protection and security.

Shared accounts



Instances (3/3) Info

Connect | Instance state | Actions | Launch instances

Find Instance by attribute or tag (case-sensitive) | All states

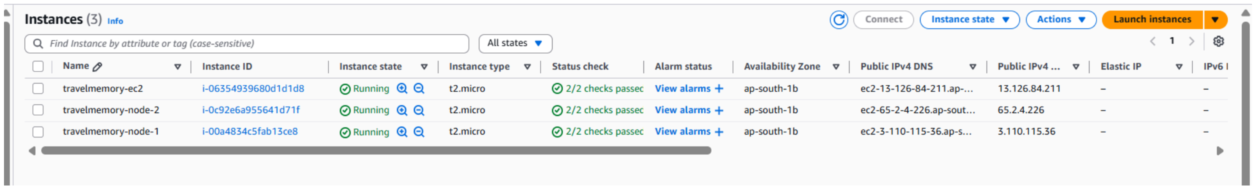
✓	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP	IF
✓	travelmemory-ec2	i-06354939680d1d1d8	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1b	ec2-13-126-84-211.ap-...	13.126.84.211	–	–
✓	travelmemory-node-2	i-0c92e6a955641d71f	Running	t2.micro	Initializing	View alarms +	ap-south-1b	ec2-65-2-4-226.ap-sout...	65.2.4.226	–	–
✓	travelmemory-node-1	i-00a4834c5fab13ce8	Running	t2.micro	2/2 checks passed	View alarms +	ap-south-1b	ec2-3-110-115-36.ap-s...	3.110.115.36	–	–

3 instances selected

Monitoring

Configure CloudWatch agent

Alarm recommendations | Investigate with AI - new | 1h 3h 12h 1d 3d 1w Custom | UTC timezone | Explore related

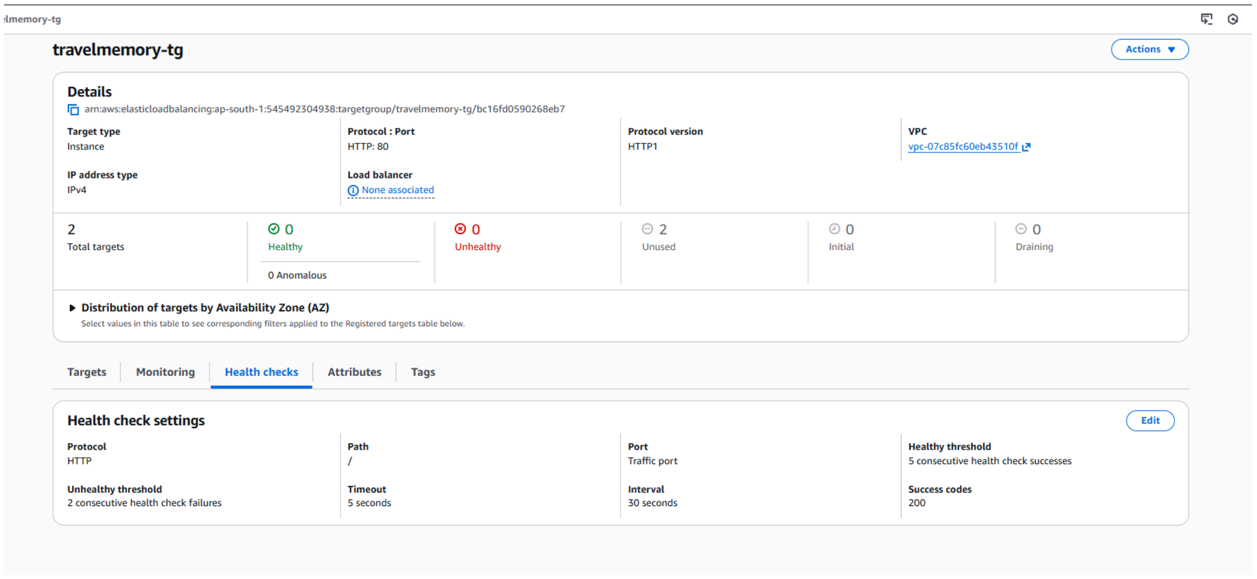


Instances (3) Info

Connect | Instance state | Actions | Launch instances

Find Instance by attribute or tag (case-sensitive) | All states

	Name	Instance ID	Instance state	Instance type	Status check	Alarm status	Availability Zone	Public IPv4 DNS	Public IPv4 ...	Elastic IP	IPv6 I
	travelmemory-ec2	i-06354939680d1d1d8	Running	t2.micro	2/2 checks passec	View alarms +	ap-south-1b	ec2-13-126-84-211.ap-...	13.126.84.211	–	–
	travelmemory-node-2	i-0c92e6a955641d71f	Running	t2.micro	2/2 checks passec	View alarms +	ap-south-1b	ec2-65-2-4-226.ap-sout...	65.2.4.226	–	–
	travelmemory-node-1	i-00a4834c5fab13ce8	Running	t2.micro	2/2 checks passec	View alarms +	ap-south-1b	ec2-3-110-115-36.ap-s...	3.110.115.36	–	–



lmemory-tg

travelmemory-tg

Actions

Details

arn:aws:elasticloadbalancing:ap-south-1:545492304938:targetgroup/travelmemory-tg/bc16fd0590268eb7

Target type Instance	**Protocol : Port** HTTP: 80	**Protocol version** HTTP1	**VPC** vpc-07c85fc60eb43510f
IP address type IPv4	**Load balancer** None associated		

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

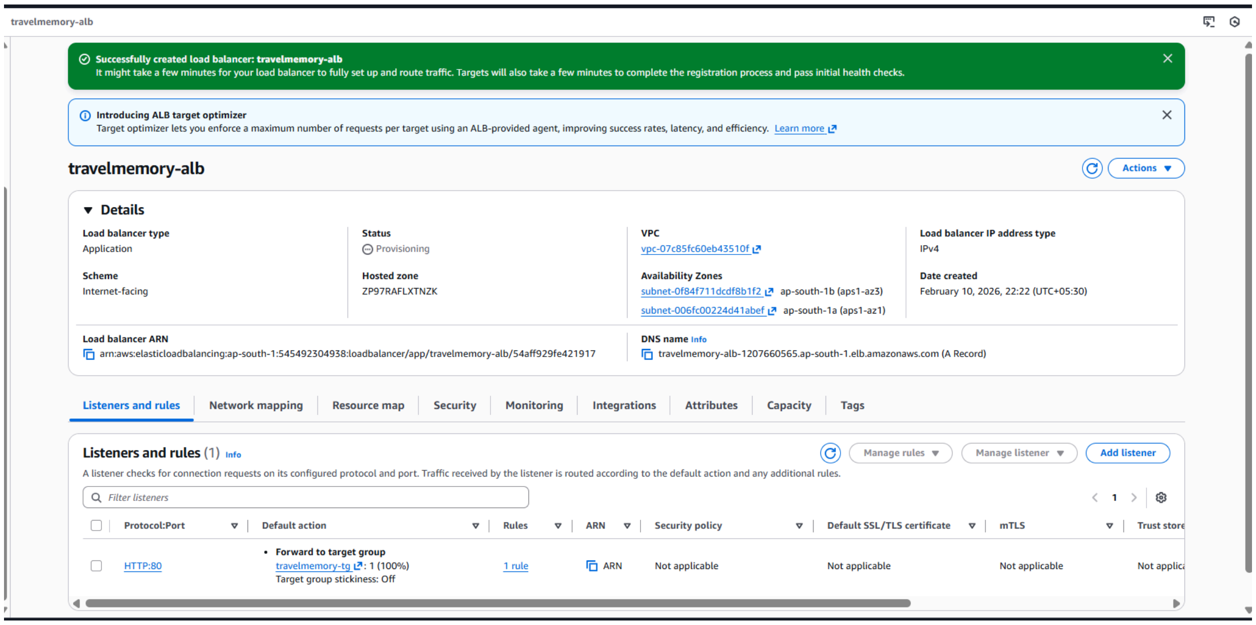
Distribution of targets by Availability Zone (AZ)

Select values in this table to see corresponding filters applied to the Registered targets table below.

Targets | Monitoring | Health checks | Attributes | Tags

Health check settings Edit

Protocol HTTP	**Path** /	**Port** Traffic port	**Healthy threshold** 5 consecutive health check successes
Unhealthy threshold 2 consecutive health check failures	**Timeout** 5 seconds	**Interval** 30 seconds	**Success codes** 200


travelmemory-alb

Successfully created load balancer: travelmemory-alb
It might take a few minutes for your load balancer to fully set up and route traffic. Targets will also take a few minutes to complete the registration process and pass initial health checks.

Introducing ALB target optimizer
Target optimizer lets you enforce a maximum number of requests per target using an ALB-provided agent, improving success rates, latency, and efficiency. Learn more

travelmemory-alb

Details

Load balancer type Application	**Status** Provisioning	**VPC** vpc-07c85fc60eb43510f	**Load balancer IP address type** IPv4
Scheme Internet-facing	**Hosted zone** ZP97RAFLXTNZK	**Availability Zones** subnet-0f84f711dcdf8b1f2 ap-south-1b (aps1-az3) subnet-006fc00224d41abef ap-south-1a (aps1-az1)	**Date created** February 10, 2026, 22:22 (UTC+05:30)

Load balancer ARN
arn:aws:elasticloadbalancing:ap-south-1:545492304938:loadbalancer/app/travelmemory-alb/54aff929fe421917

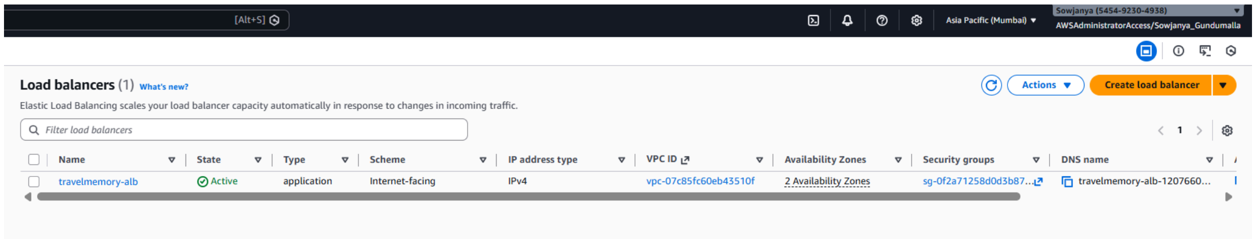
DNS name Info
travelmemory-alb-1207660565.ap-south-1.elb.amazonaws.com (A Record)

Listeners and rules | Network mapping | Resource map | Security | Monitoring | Integrations | Attributes | Capacity | Tags

Listeners and rules (1) Info

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.

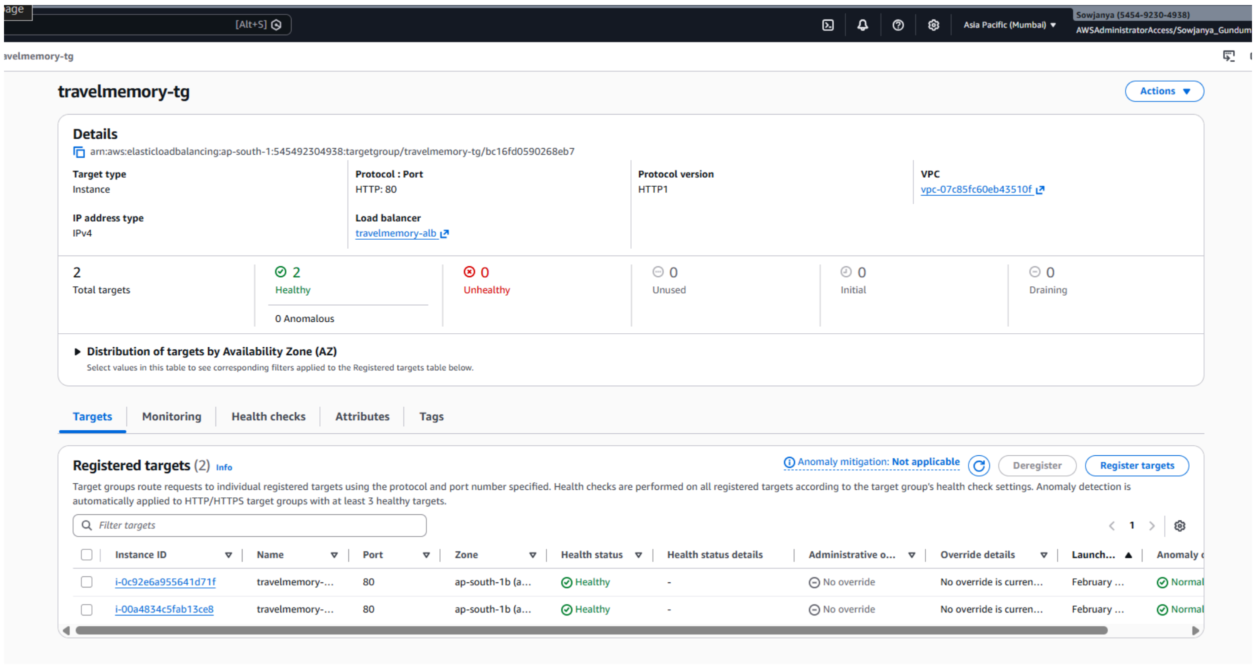
Protocol:Port	Default action	Rules	ARN	Security policy	Default SSL/TLS certificate	mTLS	Trust store
HTTP:80	• **Forward to target group** travelmemory-tg: 1 (100%) Target group stickiness: Off	1 rule	ARN	Not applicable	Not applicable	Not applicable	Not applica


[Alt+S]
Asia Pacific (Mumbai)
Sowjanya (5454-9230-4938)
AWSAdministratorAccess/Sowjanya_Gundumalla

Load balancers (1) What's new?

Elastic Load Balancing scales your load balancer capacity automatically in response to changes in incoming traffic.

Name	State	Type	Scheme	IP address type	VPC ID	Availability Zones	Security groups	DNS name
travelmemory-alb	Active	application	Internet-facing	IPv4	vpc-07c85fc60eb43510f	2 Availability Zones	sg-0f2a71258d0d3b87...	travelmemory-alb-1207660...


age
[Alt+S]
Asia Pacific (Mumbai)
Sowjanya (5454-9230-4938)
AWSAdministratorAccess/Sowjanya_Gundum
avelmemory-tg

travelmemory-tg

Details

arn:aws:elasticloadbalancing:ap-south-1:545492304938:targetgroup/travelmemory-tg/bc16fd0590268eb7

Target type Instance	**Protocol : Port** HTTP: 80	**Protocol version** HTTP1	**VPC** vpc-07c85fc60eb43510f
IP address type IPv4	**Load balancer** travelmemory-alb		

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.

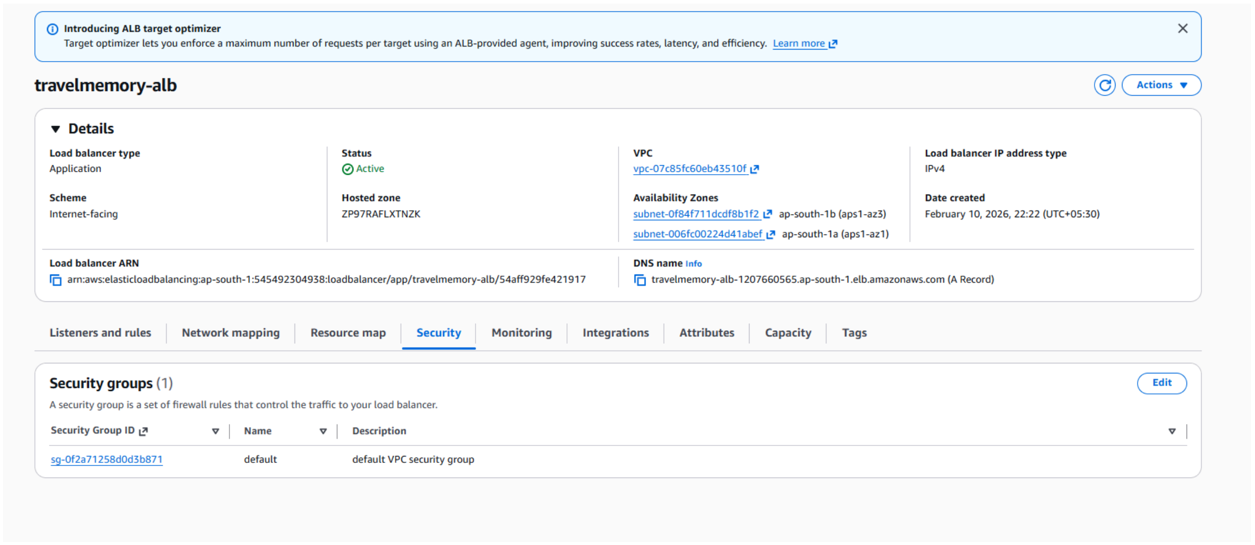
Targets | Monitoring | Health checks | Attributes | Tags

Registered targets (2) Info

Anomaly mitigation: Not applicable

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.

Instance ID	Name	Port	Zone	Health status	Health status details	Administrative o...	Override details	Launch...	Anomaly
i-0c92e6a955641d71f	travelmemory-...	80	ap-south-1b (a...	Healthy	-	No override	No override is curren...	February ...	Normal
i-00a4834c5fab13ce8	travelmemory-...	80	ap-south-1b (a...	Healthy	-	No override	No override is curren...	February ...	Normal


Introducing ALB target optimizer
Target optimizer lets you enforce a maximum number of requests per target using an ALB-provided agent, improving success rates, latency, and efficiency. Learn more
Actions
Listeners and rules
Network mapping
Resource map
Security
Monitoring
Integrations
Attributes
Capacity
Tags
Edit

travelmemory-alb

Details

Load balancer type Application	**Status** Active	**VPC** vpc-07c85fc60eb43510f	**Load balancer IP address type** IPv4
Scheme Internet-facing	**Hosted zone** ZP97RAFLXTNZK	**Availability Zones** subnet-0f84f711dcdf8b1f2 ap-south-1b (aps1-az3); subnet-006fc00224d41abef ap-south-1a (aps1-az1)	**Date created** February 10, 2026, 22:22 (UTC+05:30)

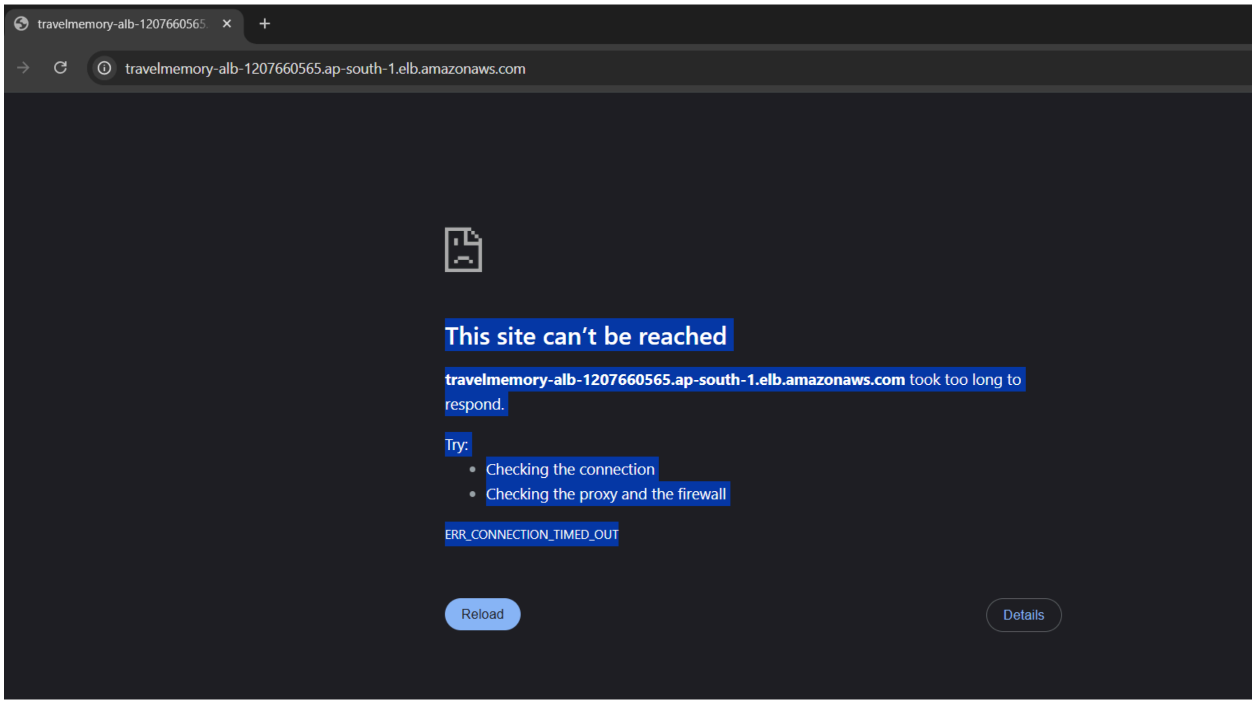
Load balancer ARN
arn:aws:elasticloadbalancing:ap-south-1:545492304938:loadbalancer/app/travelmemory-alb/54aff929fe421917

DNS name Info
travelmemory-alb-1207660565.ap-south-1.elb.amazonaws.com (A Record)

Security groups (1)

A security group is a set of firewall rules that control the traffic to your load balancer.

Security Group ID	Name	Description
sg-0f2a71258d0d3b871	default	default VPC security group


travelmemory-alb-1207660565.
travelmemory-alb-1207660565.ap-south-1.elb.amazonaws.com
Reload
Details

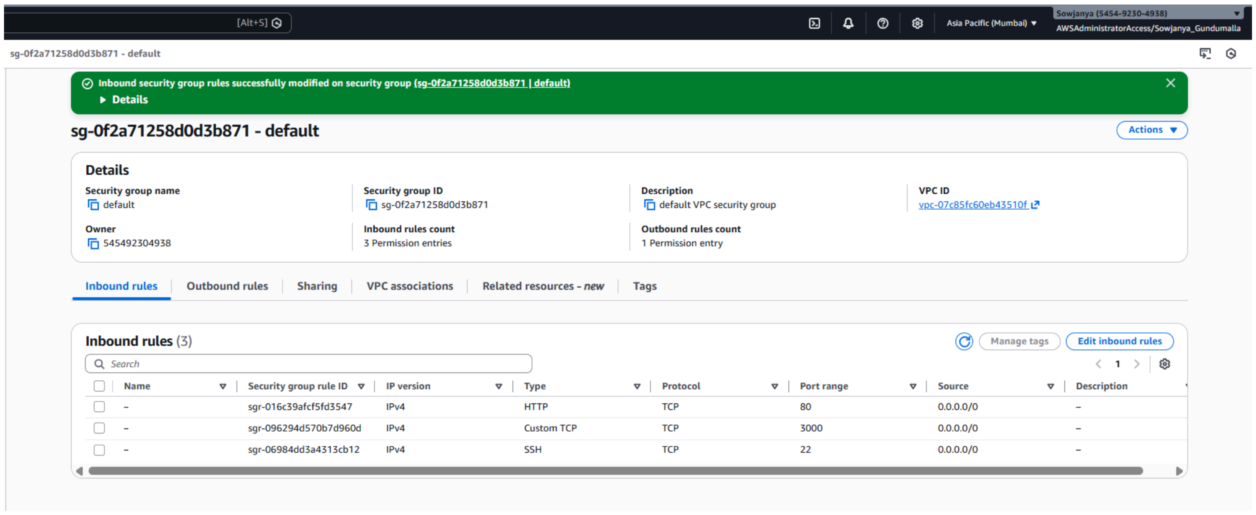
This site can’t be reached

travelmemory-alb-1207660565.ap-south-1.elb.amazonaws.com took too long to respond.

Try:

- Checking the connection
- Checking the proxy and the firewall

ERR_CONNECTION_TIMED_OUT



sg-0f2a71258d0d3b871 - default

Inbound security group rules successfully modified on security group (sg-0f2a71258d0d3b871 | default)

Details

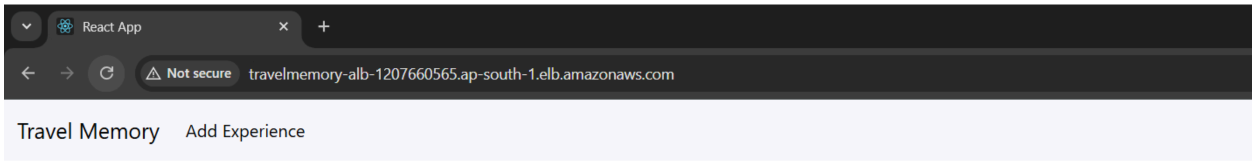
sg-0f2a71258d0d3b871 - default

Details

Security group name	Security group ID	Description	VPC ID
default	sg-0f2a71258d0d3b871	default VPC security group	vpc-07c85fc60eb43510f
Owner	Inbound rules count	Outbound rules count	
545492304938	3 Permission entries	1 Permission entry	

Inbound rules (3)

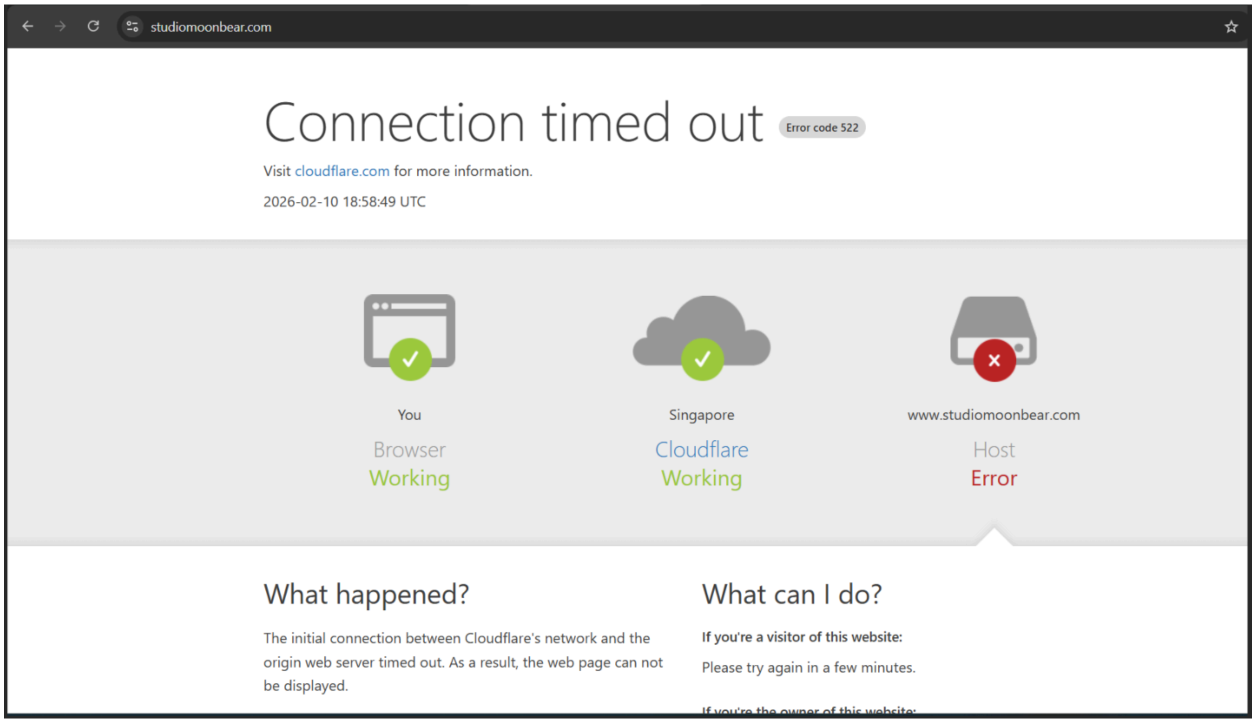
Name	Security group rule ID	IP version	Type	Protocol	Port range	Source	Description
–	sgr-016c39afcf5fd3547	IPv4	HTTP	TCP	80	0.0.0.0/0	–
–	sgr-096294d570b7d960d	IPv4	Custom TCP	TCP	3000	0.0.0.0/0	–
–	sgr-06984dd3a4313cb12	IPv4	SSH	TCP	22	0.0.0.0/0	–



Cloudflare and Domain Configuration:

Mydomain: studiomoonbear.com
Loadbalancer: http://travelmemory-alb-1207660565.ap-south-1.elb.amazonaws.com/

- The custom domain was added to Cloudflare for DNS management.
- Nameservers were updated at the domain registrar to point to Cloudflare.
- A CNAME record was configured to route the www subdomain to the Load Balancer.
- An A record was configured to map the root domain to the EC2 public IP address.
- DNS propagation was verified and the application was successfully accessed via the custom domain.



studiomoonbear.com

Connection timed out Error code 522

Visit cloudflare.com for more information.

2026-02-10 18:58:49 UTC

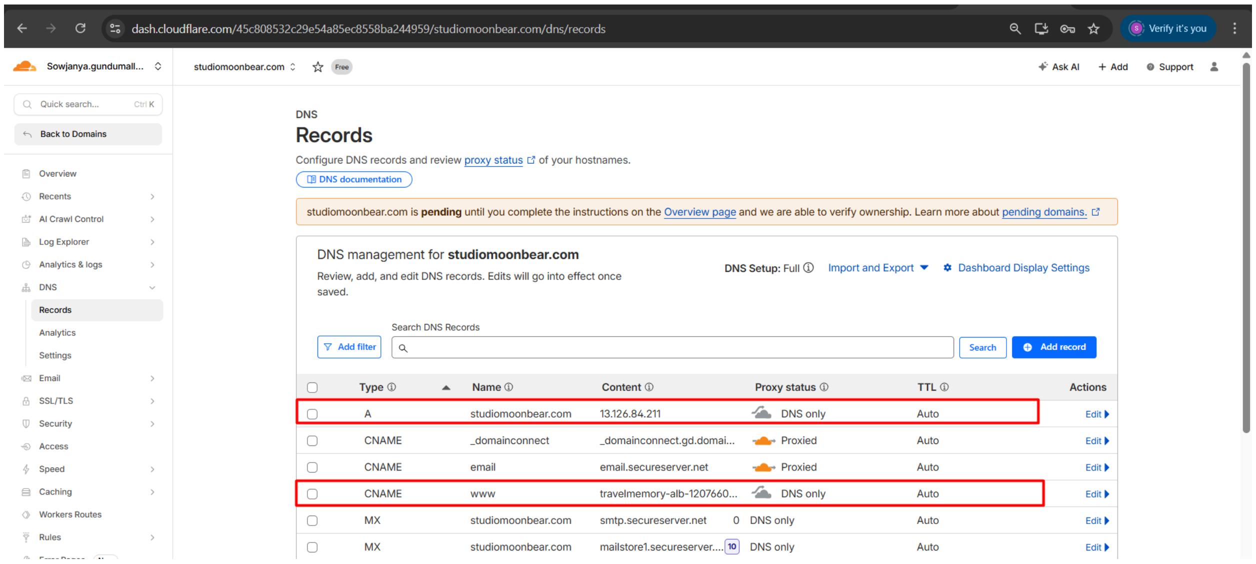
What happened?

The initial connection between Cloudflare's network and the origin web server timed out. As a result, the web page can not be displayed.

What can I do?

If you're a visitor of this website:

Please try again in a few minutes.



dash.cloudflare.com/45c808532c29e54a85ec8558ba244959/studiomoonbear.com/dns/records

DNS

Records

Configure DNS records and review proxy status of your hostnames.

DNS documentation

studiomoonbear.com is **pending** until you complete the instructions on the Overview page and we are able to verify ownership. Learn more about pending domains.

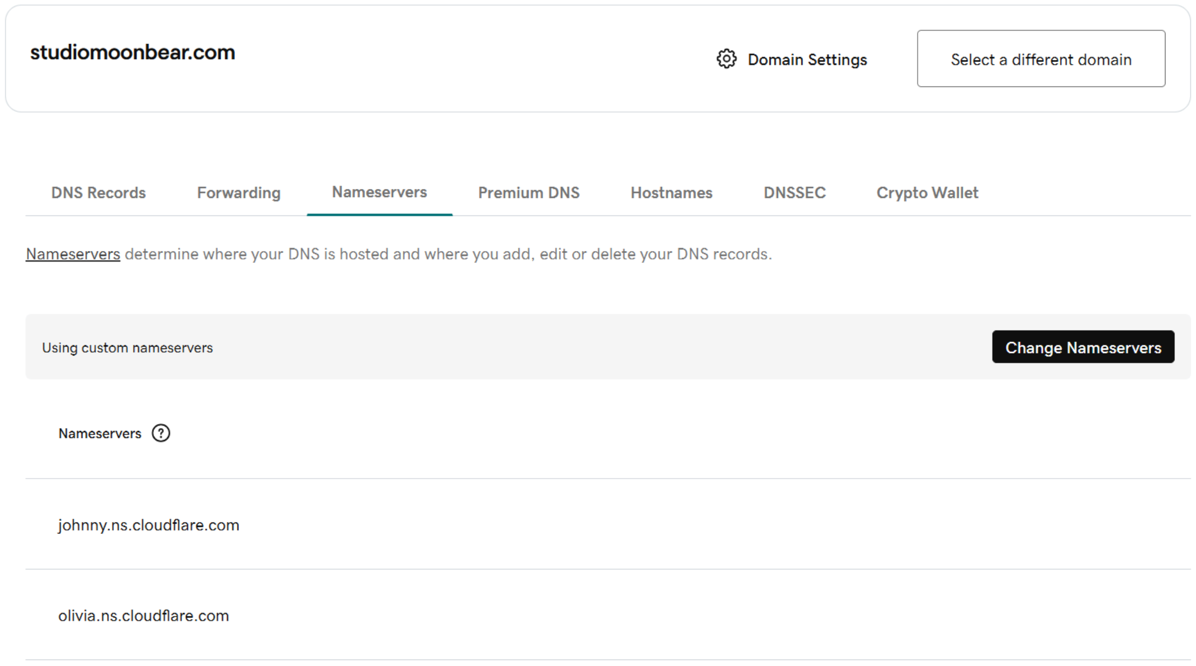
DNS management for **studiomoonbear.com**

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

DNS Setup: Full   Import and Export   Dashboard Display Settings

Search DNS Records

Type	Name	Content	Proxy status	TTL	Actions
A	studiomoonbear.com	13.126.84.211	DNS only	Auto	Edit
CNAME	_domainconnect	_domainconnect.gd.domai...	Proxied	Auto	Edit
CNAME	email	email.secureserver.net	Proxied	Auto	Edit
CNAME	www	travelmemory-alb-1207660...	DNS only	Auto	Edit
MX	studiomoonbear.com	smtp.secureserver.net 0	DNS only	Auto	Edit
MX	studiomoonbear.com	mailstore1.secureserver.... 10	DNS only	Auto	Edit

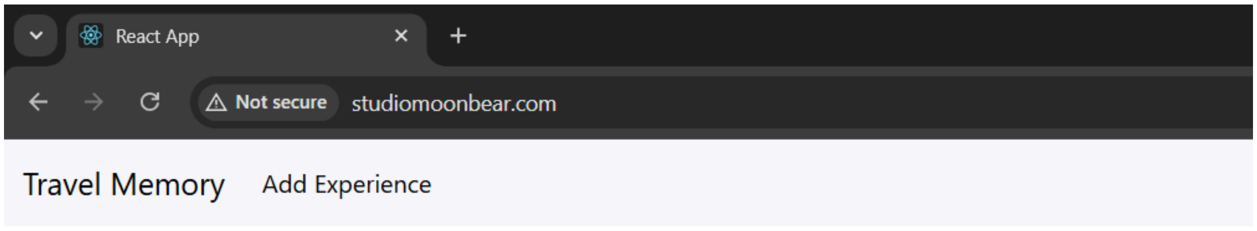


studiomoonbear.com

Nameservers determine where your DNS is hosted and where you add, edit or delete your DNS records.

Using custom nameservers

Nameservers
johnny.ns.cloudflare.com
olivia.ns.cloudflare.com



studiomoonbear.com

Travel Memory   Add Experience