Create an Ubuntu EC2 instance (AWS Console)

1. Sign in to the AWS Management Console → **EC2** → **Instances** → **Launch instances**.
2. **Name and OS**
   - Name: `multi-cloud` (any name)
   - AMI: choose **Ubuntu Server 22.04 LTS** (or 20.04 LTS)
3. **Instance type**
   - `t2.micro` (free tier) or `t3a.small` / `t3.small` depending on need.
4. **Key pair (SSH)**
   - Select existing key pair or **Create new key pair** → download `.pem` file → save securely.
5. **Network settings / Security group (Inbound rules)** — at minimum add:
   - SSH — TCP 22 — Source: *your IP* (choose “My IP”)
   - (Optional for your app) Custom TCP — 3000, 3001, 3002 — Source: your IP (or 0.0.0.0/0 if public testing).
   - **Security tip:** restrict to your IP where possible.
6. Storage: default 8–20 GB is fine.
7. Tags: optional.
8. Review & Launch → confirm key pair → **Launch**

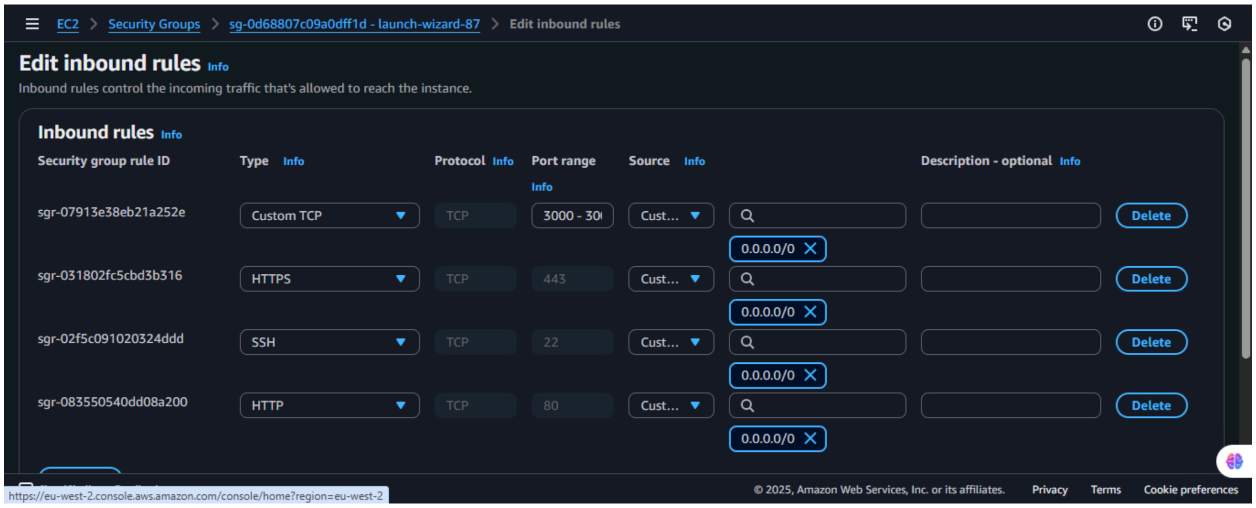
```
# 1. Update and install prerequisites
sudo apt update
sudo apt install -y ca-certificates curl gnupg lsb-release

# 2. Add Docker's official GPG key and repo
sudo mkdir -p /etc/apt/keyrings
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmour -o
/etc/apt/keyrings/docker.gpg
echo \
  "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] \
  https://download.docker.com/linux/ubuntu \
  $(lsb_release -cs) stable" | \
  sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

# 3. Install Docker Engine, CLI, and containerd
sudo apt update
sudo apt install -y docker-ce docker-ce-cli containerd.io docker-compose-plugin

# 4. Start & enable Docker
sudo systemctl enable --now docker

# 5. Add ubuntu user to docker group (so you can run docker without sudo)
sudo usermod -aG docker $USER
# apply group change in current shell without logout (optional)
newgrp docker
```



EC2 > Security Groups > sg-0d68807c09a0dff1d - launch-wizard-87 > Edit inbound rules

Edit inbound rules Info

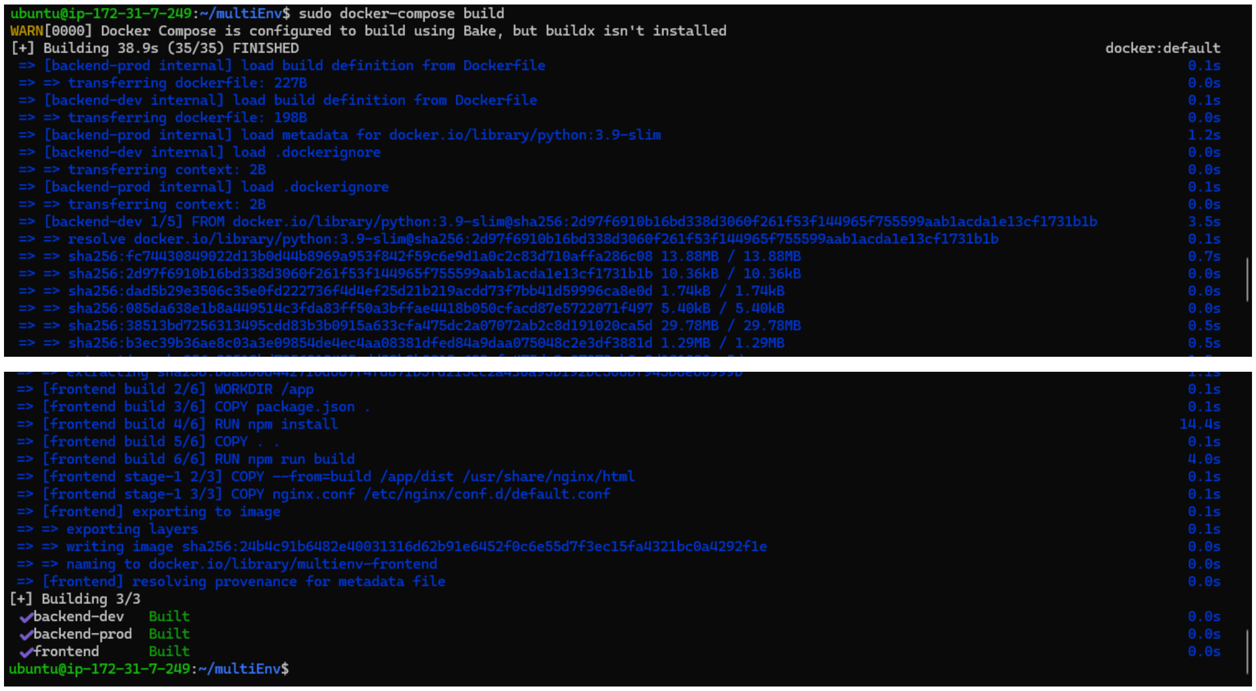
Inbound rules control the incoming traffic that's allowed to reach the instance.

Inbound rules Info

Security group rule ID	Type	Protocol	Port range	Source		Description - optional	
sgr-07913e38eb21a252e	Custom TCP	TCP	3000 - 30	Cust...	0.0.0.0/0		Delete
sgr-031802fc5cbd3b316	HTTPS	TCP	443	Cust...	0.0.0.0/0		Delete
sgr-02f5c091020324ddd	SSH	TCP	22	Cust...	0.0.0.0/0		Delete
sgr-083550540dd08a200	HTTP	TCP	80	Cust...	0.0.0.0/0		Delete

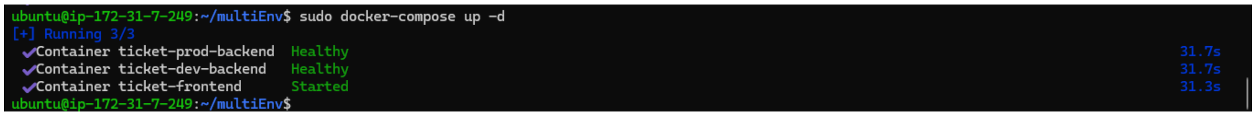
https://eu-west-2.console.aws.amazon.com/console/home?region=eu-west-2

© 2025, Amazon Web Services, Inc. or its affiliates. Privacy Terms Cookie preferences

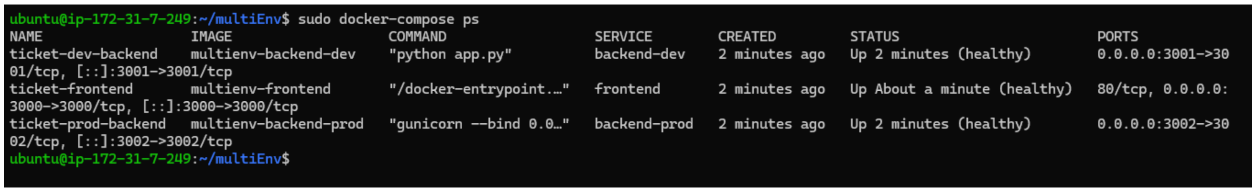


```
ubuntu@ip-172-31-7-249:~/multiEnv$ sudo docker-compose build
WARN[0000] Docker Compose is configured to build using Bake, but buildx isn't installed
[+] Building 38.9s (35/35) FINISHED                                                                                    docker:default
 => [backend-prod internal] load build definition from Dockerfile                                                                0.1s
 => => transferring dockerfile: 227B                                                                                             0.0s
 => [backend-dev internal] load build definition from Dockerfile                                                                 0.1s
 => => transferring dockerfile: 198B                                                                                             0.0s
 => [backend-prod internal] load metadata for docker.io/library/python:3.9-slim                                                  1.2s
 => [backend-dev internal] load .dockerignore                                                                                    0.0s
 => => transferring context: 2B                                                                                                  0.0s
 => [backend-prod internal] load .dockerignore                                                                                   0.1s
 => => transferring context: 2B                                                                                                  0.0s
 => [backend-dev 1/5] FROM docker.io/library/python:3.9-slim@sha256:2d97f6910b16bd338d3060f261f53f144965f755599aab1acda1e13cf1731b1b    3.5s
 => => resolve docker.io/library/python:3.9-slim@sha256:2d97f6910b16bd338d3060f261f53f144965f755599aab1acda1e13cf1731b1b          0.1s
 => => sha256:fc74430849022d13b0d44b8969a953f842f59c6e9d1a0c2c83d710affa286c08 13.88MB / 13.88MB                                 0.7s
 => => sha256:2d97f6910b16bd338d3060f261f53f144965f755599aab1acda1e13cf1731b1b 10.36kB / 10.36kB                                 0.0s
 => => sha256:dad5b29e3506c35e0fd222736f4d4ef25d21b219acdd73f7bb41d59996ca8e0d 1.74kB / 1.74kB                                   0.0s
 => => sha256:085da638e1b8a449514c3fda83ff50a3bffae4418b050cfacd87e5722071f497 5.40kB / 5.40kB                                   0.0s
 => => sha256:38513bd7256313495cdd83b3b0915a633cfa475dc2a07072ab2c8d191020ca5d 29.78MB / 29.78MB                                 0.5s
 => => sha256:b3ec39b36ae8c03a3e09854de4ec4aa08381dfed84a9daa075048c2e3df3881d 1.29MB / 1.29MB                                   0.5s
```

```
 => [frontend build 2/6] WORKDIR /app                                                                                            0.1s
 => [frontend build 3/6] COPY package.json .                                                                                     0.1s
 => [frontend build 4/6] RUN npm install                                                                                        14.4s
 => [frontend build 5/6] COPY . .                                                                                                0.1s
 => [frontend build 6/6] RUN npm run build                                                                                       4.0s
 => [frontend stage-1 2/3] COPY --from=build /app/dist /usr/share/nginx/html                                                     0.1s
 => [frontend stage-1 3/3] COPY nginx.conf /etc/nginx/conf.d/default.conf                                                        0.1s
 => [frontend] exporting to image                                                                                                0.1s
 => => exporting layers                                                                                                          0.1s
 => => writing image sha256:24b4c91b6482e40031316d62b91e6452f0c6e55d7f3ec15fa4321bc0a4292f1e                                     0.0s
 => => naming to docker.io/library/multienv-frontend                                                                             0.0s
 => [frontend] resolving provenance for metadata file                                                                            0.0s
[+] Building 3/3
 ✔backend-dev   Built                                                                                                            0.0s
 ✔backend-prod  Built                                                                                                            0.0s
 ✔frontend      Built                                                                                                            0.0s
ubuntu@ip-172-31-7-249:~/multiEnv$
```

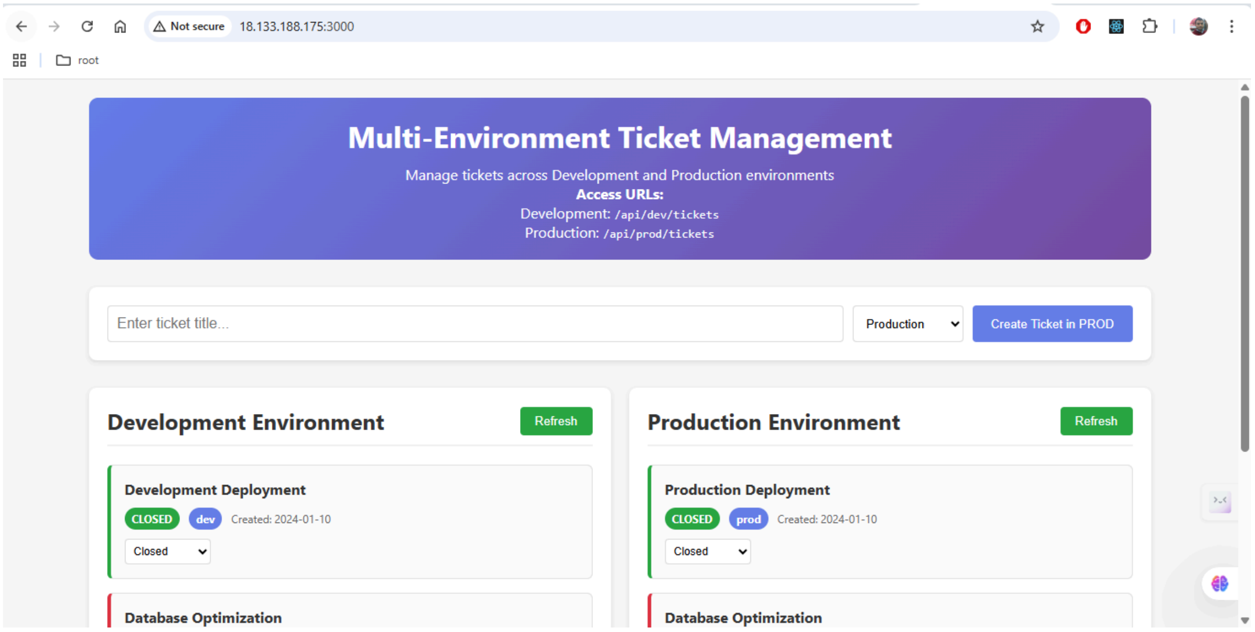


```
ubuntu@ip-172-31-7-249:~/multiEnv$ sudo docker-compose up -d
[+] Running 3/3
 ✔Container ticket-prod-backend  Healthy                                                                                        31.7s
 ✔Container ticket-dev-backend   Healthy                                                                                        31.7s
 ✔Container ticket-frontend      Started                                                                                        31.3s
ubuntu@ip-172-31-7-249:~/multiEnv$
```



```
ubuntu@ip-172-31-7-249:~/multiEnv$ sudo docker-compose ps
NAME                  IMAGE                   COMMAND                      SERVICE        CREATED           STATUS                          PORTS
ticket-dev-backend    multienv-backend-dev    "python app.py"              backend-dev    2 minutes ago     Up 2 minutes (healthy)          0.0.0.0:3001->30
01/tcp, [::]:3001->3001/tcp
ticket-frontend       multienv-frontend       "/docker-entrypoint.…"       frontend       2 minutes ago     Up About a minute (healthy)     80/tcp, 0.0.0.0:
3000->3000/tcp, [::]:3000->3000/tcp
ticket-prod-backend   multienv-backend-prod   "gunicorn --bind 0.0…"       backend-prod   2 minutes ago     Up 2 minutes (healthy)          0.0.0.0:3002->30
02/tcp, [::]:3002->3002/tcp
ubuntu@ip-172-31-7-249:~/multiEnv$
```

Frontend Dashboard: http://localhost:3000


Not secure 18.133.188.175:3000

root

Multi-Environment Ticket Management

Manage tickets across Development and Production environments
Access URLs:
Development: `/api/dev/tickets`
Production: `/api/prod/tickets`

Enter ticket title... | Production | Create Ticket in PROD

Development Environment
Refresh

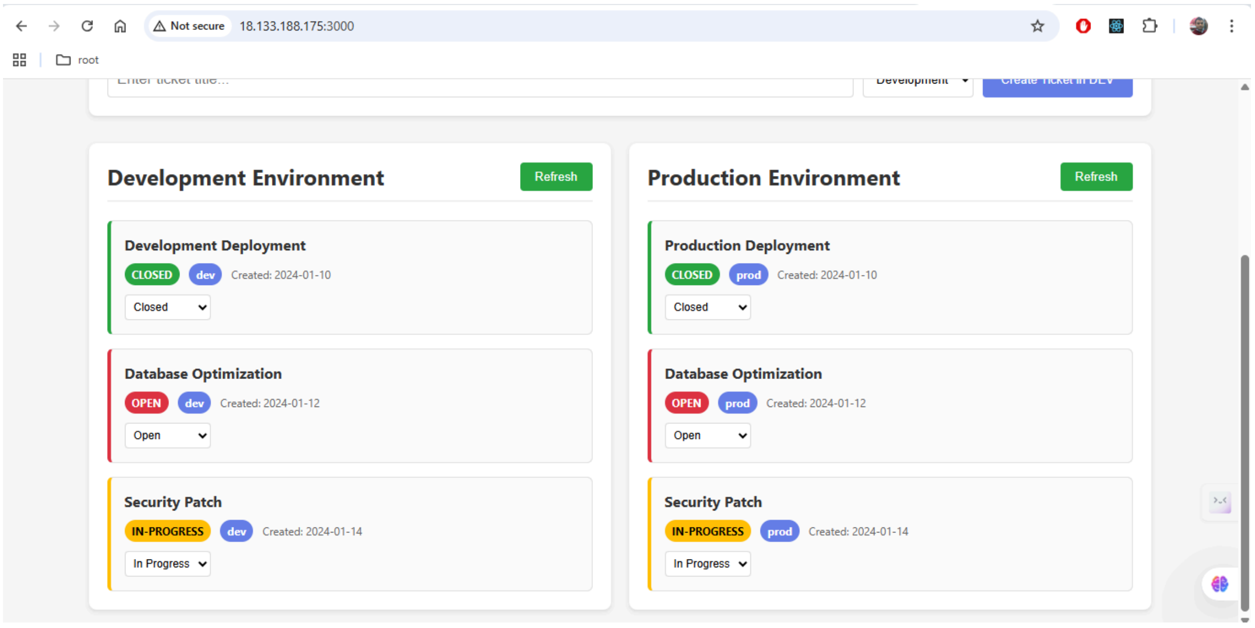
Development Deployment
CLOSED dev Created: 2024-01-10
Closed

Database Optimization

Production Environment
Refresh

Production Deployment
CLOSED prod Created: 2024-01-10
Closed

Database Optimization


Not secure 18.133.188.175:3000

root

Enter ticket title... | Development | Create Ticket in DEV

Development Environment
Refresh

Development Deployment
CLOSED dev Created: 2024-01-10
Closed

Database Optimization
OPEN dev Created: 2024-01-12
Open

Security Patch
IN-PROGRESS dev Created: 2024-01-14
In Progress

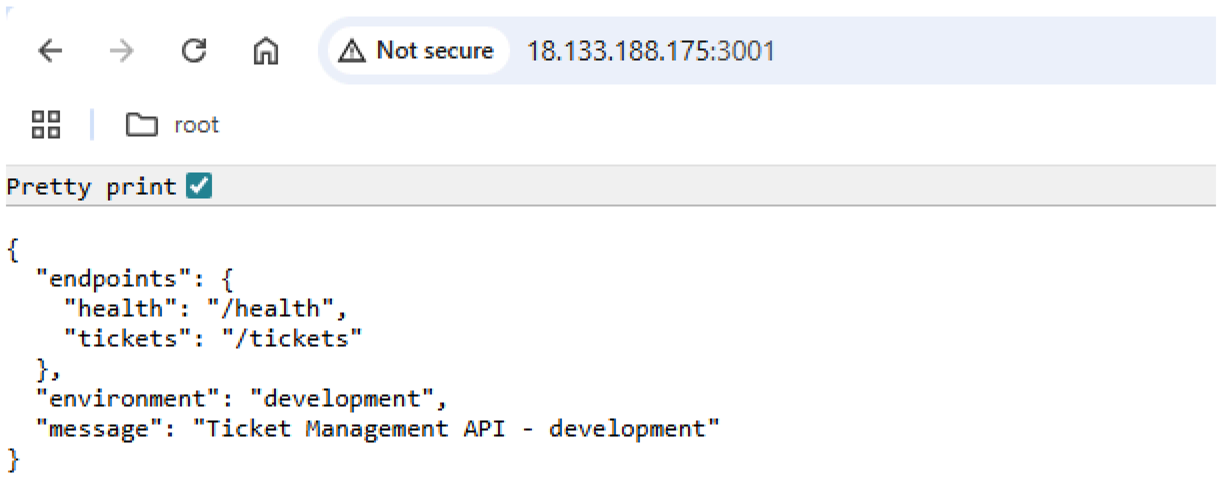
Production Environment
Refresh

Production Deployment
CLOSED prod Created: 2024-01-10
Closed

Database Optimization
OPEN prod Created: 2024-01-12
Open

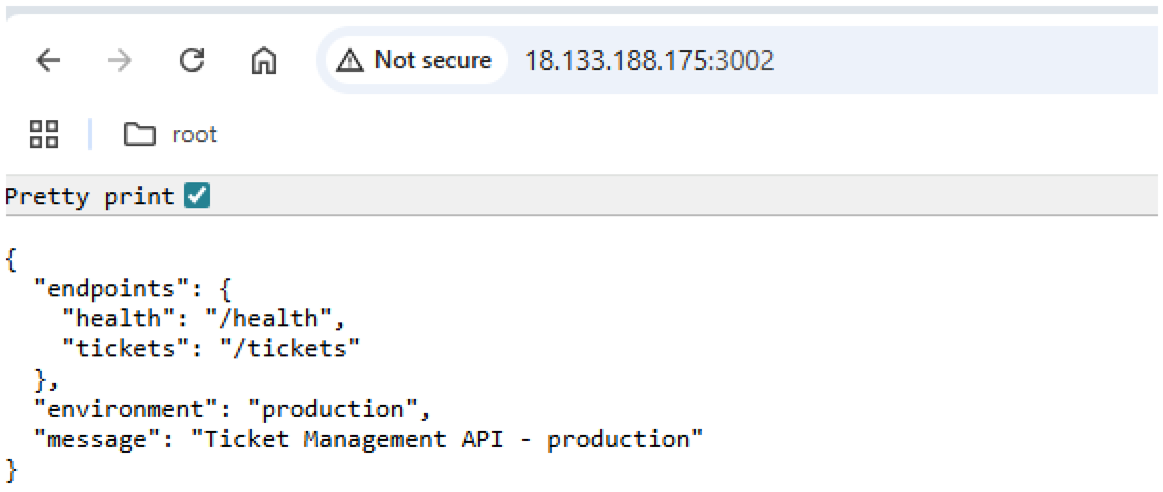
Security Patch
IN-PROGRESS prod Created: 2024-01-14
In Progress

Development API Direct: http://localhost:3001



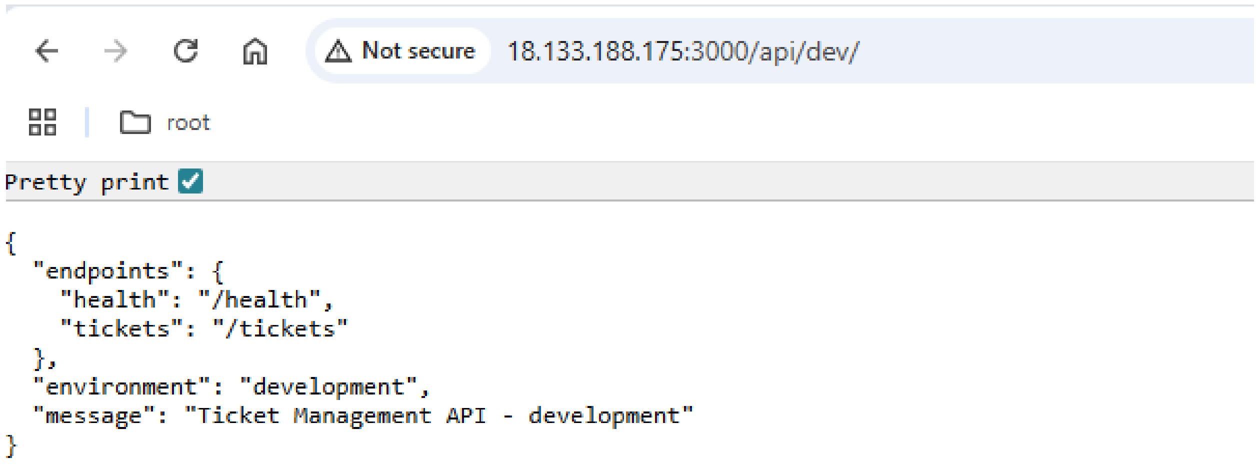
```
{
  "endpoints": {
    "health": "/health",
    "tickets": "/tickets"
  },
  "environment": "development",
  "message": "Ticket Management API - development"
}
```

Production API Direct: http://localhost:3002



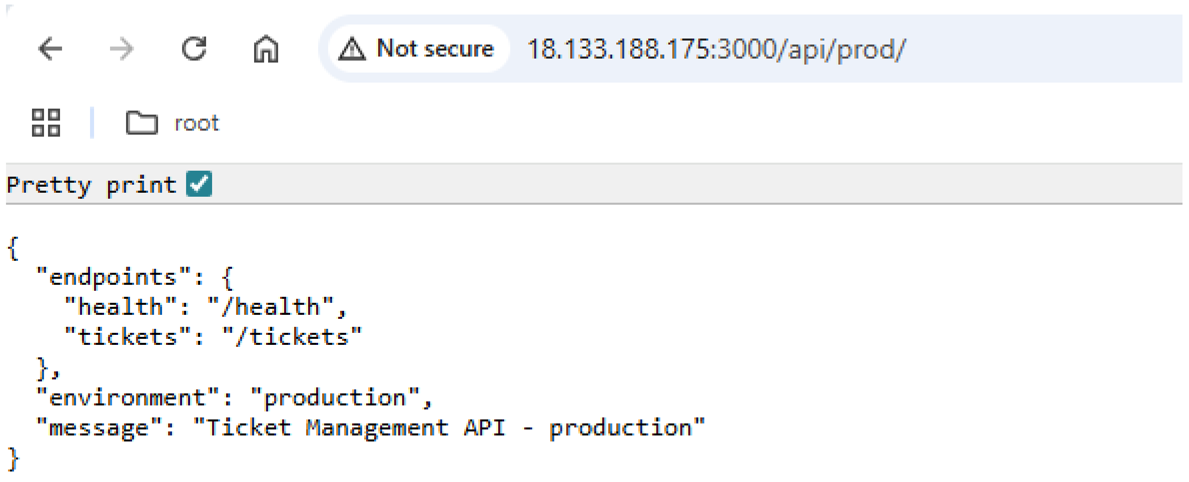
```
{
  "endpoints": {
    "health": "/health",
    "tickets": "/tickets"
  },
  "environment": "production",
  "message": "Ticket Management API - production"
}
```

Development API via Proxy: http://localhost:3000/api/dev

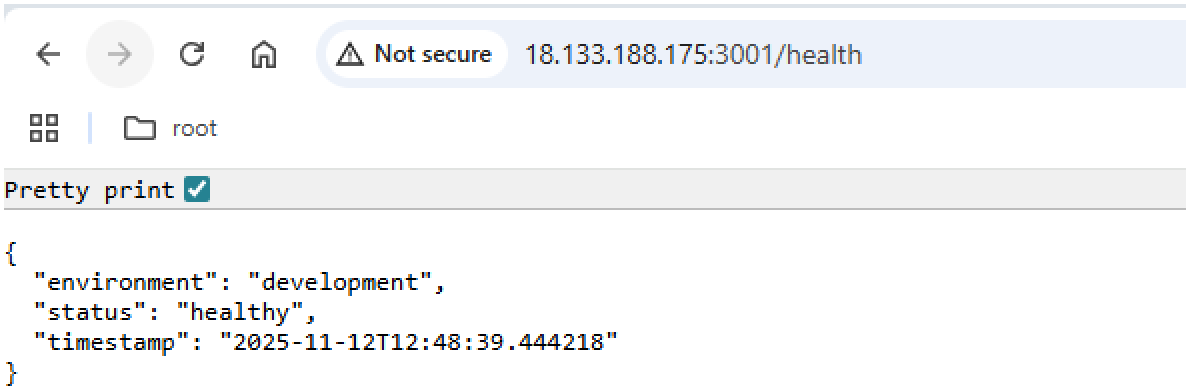


```
{
  "endpoints": {
    "health": "/health",
    "tickets": "/tickets"
  },
  "environment": "development",
  "message": "Ticket Management API - development"
}
```

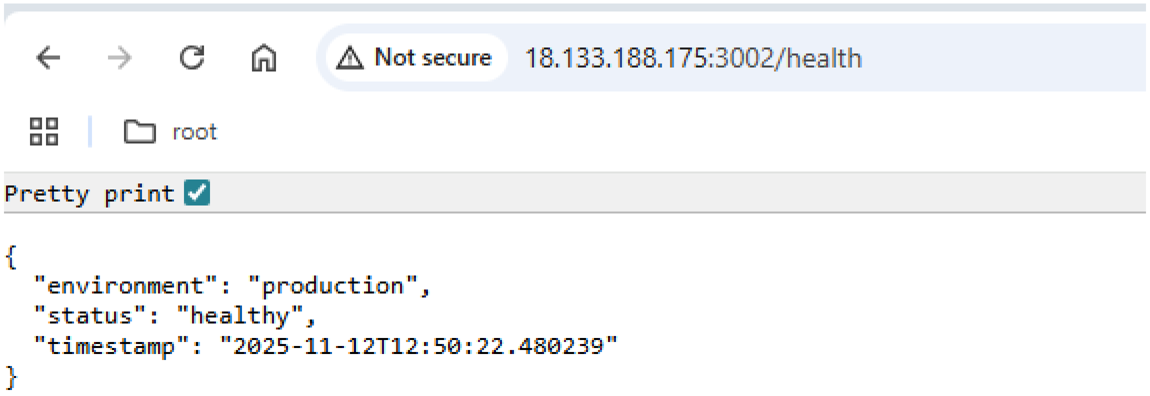
Production API via Proxy: http://localhost:3000/api/prod



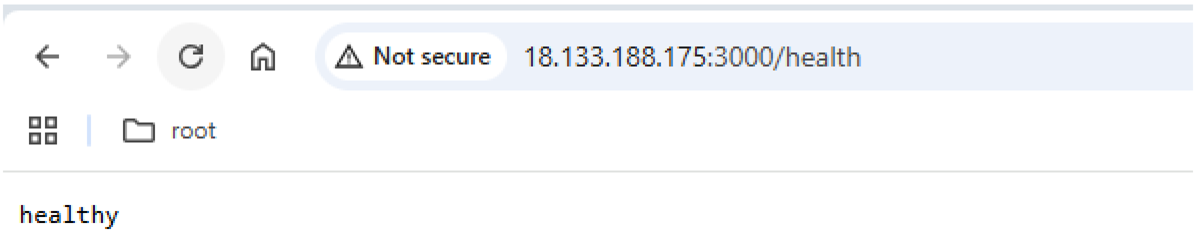
```
{
  "endpoints": {
    "health": "/health",
    "tickets": "/tickets"
  },
  "environment": "production",
  "message": "Ticket Management API - production"
}
```



```
{
  "environment": "development",
  "status": "healthy",
  "timestamp": "2025-11-12T12:48:39.444218"
}
```

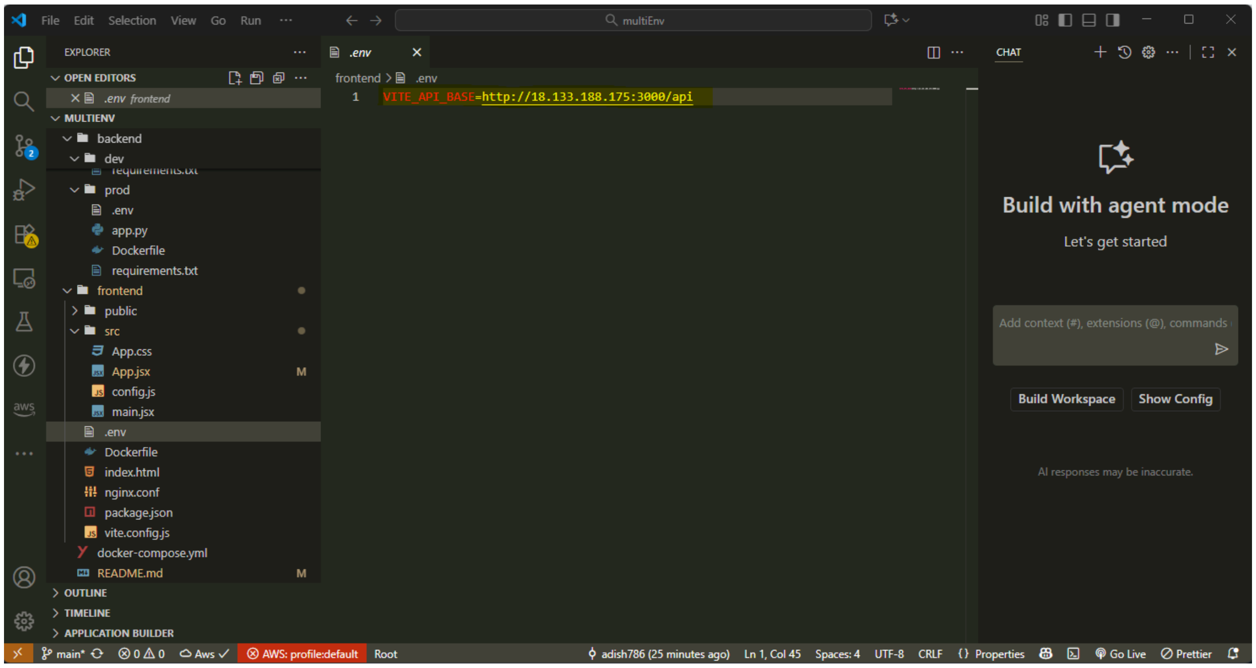


```
{
  "environment": "production",
  "status": "healthy",
  "timestamp": "2025-11-12T12:50:22.480239"
}
```



```
healthy
```

If you are using EC2 instance and connecting my Ip address then you need to update in .env file with respective public ip located in frontend



```
VITE_API_BASE=http://18.133.188.175:3000/api
```