4_DEVOPS_DOCUMENTATION —

Complete step by step from connecting to EC2 via PuTTY all the way to testing the final working app, Security Group ports, DuckDNS setup, cloning repos, building frontend, starting backend, common issues and fixes, all important file locations on EC2.

DevOps Documentation

ChourangiHealth Video Consultation Platform

Infrastructure Overview

Component	Details
Cloud Provider	AWS EC2
Instance	Ubuntu (t2 or t3 series)
Public IP	18.130.149.212
Domain	chourangi.duckdns.org
SSL	Let's Encrypt (free, auto-renewing)
Web Server	Nginx (inside OpenVidu Docker container)
Backend Runtime	Python Uvicorn on port 8000
Media Server	OpenVidu v2.32 (Docker)

Step 1 — Connect to EC2 Server via PuTTY

You need the .ppk private key file to connect.

1. Open PuTTY
2. In Host Name field enter: ubuntu@18.130.149.212
3. In the left panel go to Connection > SSH > Auth > Credentials
4. Browse and select your .ppk private key file
5. Click Open

6. Accept the server fingerprint if prompted

You will be logged in as the ubuntu user but our OpenVidu files are under root. Switch to root:

sudo su

You will now be at root@ip-172-31-33-5 prompt.

Step 2 — EC2 Security Group Configuration

The following inbound rules must be configured in AWS Console under EC2 > Security Groups:

Port	Protocol	Source	Purpose
22	TCP	0.0.0.0/0	SSH access
80	TCP	0.0.0.0/0	HTTP and Let's Encrypt verification
443	TCP	0.0.0.0/0	HTTPS web traffic
4443	TCP	0.0.0.0/0	OpenVidu WebSocket signaling
3478	TCP	0.0.0.0/0	TURN server
3478	UDP	0.0.0.0/0	STUN server
40000-57000	UDP	0.0.0.0/0	WebRTC media streams
10000-20000	UDP	0.0.0.0/0	Additional media traffic

All rules should also have IPv6 equivalents with source ::/0.

Step 3 — Domain Setup with DuckDNS

DuckDNS provides free subdomains. Our domain is chourangi.duckdns.org.

To set up or update:

1. Go to https://www.duckdns.org
2. Login with Google account
3. Create subdomain: chourangi
4. Set current IP to: 18.130.149.212
5. Click Update IP

If the EC2 instance IP changes (because no Elastic IP is assigned) you must update DuckDNS with the new IP. This is why assigning an AWS Elastic IP to the EC2 instance is recommended for production.

Step 4 — OpenVidu Configuration for Production

Edit the OpenVidu environment file:

nano /opt/openvidu/.env

Set these values:

DOMAIN_OR_PUBLIC_IP=chourangi.duckdns.org
CERTIFICATE_TYPE=letsencrypt
LETSENCRYPT_EMAIL=ashutosh@xtensible.in
OPENVIDU_SECRET=openvidu-v2-32

Save with Ctrl+X then Y then Enter.

Disable the default OpenVidu Call app in docker-compose.yml:

sed -i 's/WITH_APP=true/WITH_APP=false/' /opt/openvidu/docker-compose.yml

---

Step 5 — Install Node.js on EC2

Node.js version 18 is required to build the React frontend:

curl -fsSL https://deb.nodesource.com/setup_18.x | sudo -E bash -
sudo apt-get install -y nodejs

Verify installation:

node --version
npm --version

Step 6 — Clone Repositories from GitHub

Navigate to the working directory:

cd /home/ubuntu
mkdir test-1
cd test-1

Clone frontend (important: use -b master flag to get the correct branch):

git clone -b master https://github.com/AshuuuPatil/openvidu_videocall_microservice_frontend.git frontend

Clone backend:

git clone -b master
https://github.com/AshuuuPatil/openvidu_videocall_microservice_backend.git backend

When prompted for credentials:

- Username: AshuuuPatil
- Password: Use GitHub Personal Access Token (not account password)

To generate a GitHub Personal Access Token:

1. Go to https://github.com/settings/tokens
2. Click Generate new token (classic)
3. Name it ec2-deploy
4. Check the repo scope
5. Click Generate token and copy it immediately

Step 7 — Build and Deploy Frontend

Install dependencies and build:

```
cd /home/ubuntu/test-1/frontend
npm install
npm run build
```

Copy build files to OpenVidu Nginx serving directory:

```
cp -r build/* /opt/openvidu/custom-layout/
```

Create the Nginx routing configuration:

```
nano /opt/openvidu/custom-nginx-locations/frontend.conf
```

Paste this content:

```
location / {
    root /opt/openvidu/custom-layout;
    index index.html;
    try_files $uri $uri/ =404;
}

location /index.html {
    root /opt/openvidu/custom-layout;
}

location /api/ {
    proxy_pass http://172.17.0.1:8000;
    proxy_set_header Host $host;
    proxy_set_header X-Real-IP $remote_addr;
}
```

Save with Ctrl+X then Y then Enter.

Note: 172.17.0.1 is the Docker bridge IP. This allows the Nginx container to reach the FastAPI backend running on the EC2 host at port 8000.

Step 8 — Setup and Start Backend

Install Python dependencies:

```
cd /home/ubuntu/test-1/backend
pip install -r requirements.txt --break-system-packages --ignore-installed
```

Update the backend environment file:

```
nano .env
```

Change OPENVIDU_URL to use the domain:

```
OPENVIDU_URL=https://chourangi.duckdns.org
```

Start the backend in background (no Ctrl+C needed):

```
nohup uvicorn app.main:app --host 0.0.0.0 --port 8000 > /tmp/backend.log 2>&1 &
```

Verify backend is running:

```
curl http://localhost:8000/health
```

Expected response:

```
{"status":"ok"}
```

Step 9 — Start OpenVidu

```
cd /opt/openvidu
./openvidu restart
```

Wait 2 to 3 minutes for OpenVidu to start and for Let's Encrypt certificate to be obtained. You will see in the logs:

```
OpenVidu is ready!
* OpenVidu Server URL: https://chourangi.duckdns.org/
* OpenVidu Dashboard: https://chourangi.duckdns.org/dashboard
```

Step 10 — Test the Application

Open browser and visit:

https://chourangi.duckdns.org

You should see the ChourangiHealth login page with a green padlock (SSL) and no security warnings.

Test the full flow:

1. Open two browser windows or two devices
2. In window 1: login as doctor with username doctor and password 123456, enter Calling ID 1, click Start Consultation
3. In window 2: login as patient with username user and password 123456, enter Calling ID 1, click Start Consultation
4. Both should see each other in the video call

Updating Code After Changes

Update Frontend

```
cd /home/ubuntu/test-1/frontend
git pull origin master
npm run build
cp -r build/* /opt/openvidu/custom-layout/
```

No restart needed. Browser refresh will load new files.

Update Backend

```
cd /home/ubuntu/test-1/backend
git pull origin master
pkill -f uvicorn
nohup uvicorn app.main:app --host 0.0.0.0 --port 8000 > /tmp/backend.log 2>&1 &
```

Monitoring and Logs

View backend logs:

tail -f /tmp/backend.log

View Nginx logs:

docker logs openvidu-nginx-1 --tail 50

View OpenVidu server logs:

docker logs openvidu-openvidu-server-1 --tail 50

Check all running Docker containers:

docker ps

Check if backend process is running:

ps aux | grep uvicorn

Common Issues and Fixes

Browser shows security warning

Cause: SSL certificate not set up. Fix: Ensure CERTIFICATE_TYPE=letsencrypt in .env and domain is correctly pointed to the server IP.

Login shows 504 Gateway Timeout

Cause: Backend is not running or Nginx cannot reach it. Fix: Check if backend is running with curl http://localhost:8000/health. If not running, start it again. Verify proxy_pass IP in frontend.conf is 172.17.0.1.

Video not showing for remote participant

Cause: streamCreated event fires before React renders the video element. Fix: Use setTimeout of 300ms before calling subscriber.addVideoElement.

WebSocket connection failed error in browser console

Cause: OpenVidu server unreachable from browser or SSL not set up. Fix: Ensure SSL is configured and all Security Group ports are open.

ERR_CONNECTION_REFUSED when visiting domain

Cause: OpenVidu not running or Nginx container crashed. Fix: Run docker ps to check containers, then ./openvidu restart.

409 error in OpenVidu session creation

This is not an error. It means the session already exists which is the expected behavior when the second participant joins. The backend handles this gracefully.

Important File Locations on EC2

Path	Purpose
/opt/openvidu/.env	OpenVidu configuration
/opt/openvidu/docker-compose.yml	Docker container configuration
/opt/openvidu/custom-layout/	React build files served by Nginx
/opt/openvidu/custom-nginx-locations/frontend.conf	Custom Nginx routing rules
/home/ubuntu/test-1/frontend/	Frontend source code
/home/ubuntu/test-1/backend/	Backend source code
/tmp/backend.log	Backend runtime logs