DevOps Assignment – Error-to-Solution Journey

This document bundles **all major errors faced during the DevOps Assessment project**, the **root causes**, and the **exact solutions applied**, aligned with the original *DevOps Assignment PDF* requirements.

---

1. Project Context (from Assignment PDF)

Goal: Deploy a full-stack Hello World application with: - **Frontend**: React (Vite) served via Nginx - **Backend**: Django REST API - **Containerization**: Docker & Docker Compose - **Infrastructure**: AWS EC2 - **CI/CD**: GitHub Actions

Expected Flow:

Browser → Nginx (Frontend) → `/api/*` → Nginx Proxy → Django Backend → JSON Response

---

2. Major Issues Faced & How They Were Solved

Issue 1: GitHub Actions – Docker permission denied

Error

```
permission denied while trying to connect to the Docker daemon socket
```

Root Cause

- GitHub Actions runner user did not have permission to access `/var/run/docker.sock`

Solution

- Ran GitHub Actions self-hosted runner **as Administrator** (temporary fix)
- Ensured Docker daemon was running

✅**Pipeline moved forward**

Issue 2: Deployment succeeded but website shows "Connection Failed"

Symptom

- GitHub Actions: ✅Success

- Browser: ❌ `Failed to connect to the backend`

Root Cause

- Frontend was calling:

```
http://localhost:8000/api/hello/
```

- In production, `localhost` refers to **browser**, not EC2 backend

Fix Applied

- Changed frontend API call to relative path:

```
axios.get('/api/hello/')
```

✅Correct approach for Nginx-proxied apps

Issue 3: API works inside EC2 but fails externally (HTTP 400)

Evidence

```
curl http://localhost/api/hello/    # 200 OK
curl http://EC2_IP/api/hello/       # 400 Bad Request
```

Root Cause

- Django `ALLOWED_HOSTS` was empty
- Django blocks unknown Host headers

Confirmation

```
<title>DisallowedHost at /api/hello/</title>
```

Issue 4: Editing Django settings inside container failed

Error

```
vi: not found
nano: not found
sudo: not found
```

Root Cause

- Minimal Docker image
- Containers are immutable

Important DevOps Lesson

> **Never patch running containers**

Issue 5: Environment variable not reflected in Django

Observation

```
echo $ALLOWED_HOSTS   # *
```

But inside Django:

```
ALLOWED_HOSTS = []
```

Root Cause

- Django `settings.py` did not read env vars

Final Correct Solution (Production-Grade)

1️⃣Modify `settings.py`

```
import os

ALLOWED_HOSTS = os.getenv("ALLOWED_HOSTS", "*").split(",")
DEBUG = os.getenv("DEBUG", "0") == "1"
```

2️⃣Ensure `.env.prod`

```
DEBUG=0
ALLOWED_HOSTS=*
```

3️⃣Docker Compose (Backend)

```
env_file:
  - /opt/devops-assessment/.env.prod
```

4️⃣Rebuild via Docker Compose

```
docker compose -f docker-compose.prod.yml up -d --force-recreate backend
frontend
```

Issue 6: Nginx reverse proxy working but Host header mismatch

Fix in Nginx config

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

Final Verification

Backend check

```
curl -i http://localhost/api/hello/
# 200 OK
```

External access

```
curl -i http://<EC2_PUBLIC_IP>/api/hello/
# 200 OK
```

Browser

✅Backend Online

```
Hello World from Django Backend!
```

---

3. CI/CD Validation

- GitHub Actions:
- Build images
- Push to Docker Hub
- SSH deploy to EC2

✅No manual fixes required after commit

---

4. Key DevOps Learnings (Interview Ready)

- Containers are immutable
- Environment-driven configuration is mandatory
- `localhost` never works in production frontend
- Django `ALLOWED_HOSTS` is a common production blocker
- CI/CD success ≠ application success

---

5. Final Architecture

```
User Browser
   ↓
Nginx (Frontend)
   ↓ /api
Nginx Proxy
   ↓
Django Backend
```

---

6. Final Status

‼️ **PROJECT COMPLETED SUCCESSFULLY**

- CI/CD ✅
- Frontend ✅
- Backend ✅
- AWS EC2 ✅
- Docker & Nginx ✅

This document can be directly used for: - Interview explanation - Assignment submission - GitHub README - Portfolio showcase