■ IMMEDIATE FIX STEPS

PSScript Backend Recovery Guide

Server: 74.208.184.195
Generated: January 14, 2026

Current Status Overview

Component	Status	Details
SSH Access	■ BLOCKED	Port 22 timeout
Backend API	■ DOWN	Error 1033
Frontend	■ UP	Cloudflare Tunnel
Cloudflare Tunnel	■ ACTIVE	Routing traffic
SSL Certificate	■ VALID	Until Feb 23, 2026

Server Information

Server IP	`74.208.184.195`
SSH User	`root`
SSH Password	`xyyCbL6G`
Application URL	`https://psscript.morloksmaze.com`
Project Directory	`/opt/psscript`

Step 1: Access Hosting Provider Console

CRITICAL: SSH is blocked. You must access via your hosting provider's console.

Provider	Console Access
DigitalOcean	Droplets → Select droplet → "Console" button
Linode	Linodes → Select instance → "Launch LISH Console"
Vultr	Instances → Select instance → View Console
AWS EC2	EC2 Dashboard → Instances → Connect → "EC2 Instance Connect"
Hetzner	Cloud Console → Servers → Select server → "Console"
OVH	Manager → Servers → Select server → "KVM/IPMI"

Login Credentials:

Username	**`root`**
Password	**`xyyCbL6G`**

Step 2: Fix SSH Access

Run these commands in the console:

```
# Check SSH service status
    systemctl status sshd
# Start SSH if not running
    systemctl start sshd

    systemctl enable sshd
# Check firewall status
    ufw status
# Allow SSH through firewall
    ufw allow 22/tcp

    ufw reload
# Test from your local machine
    ssh root@74.208.184.195
```

■ Success: SSH connection should work without timeout

Step 3: Check Docker Containers

Navigate to project and check containers:

```
# Navigate to project
    cd /opt/psscript

# Check server resources
    top -bn1 | head -20

    free -h

    df -h

# Check all containers
    docker ps -a

# Expected: 8 containers running
# frontend, backend, postgres, redis, ai-service,
# cloudflared, pgadmin, redis-commander
```

Container	Port	Description
`frontend`	3000	React/Vite application
`backend`	4000	Node.js API (CRITICAL)
`postgres`	5432	PostgreSQL database
`redis`	6379	Redis cache
`ai-service`	5000	AI processing
`cloudflared`	-	Cloudflare Tunnel
`pgadmin`	5050	Database admin UI
`redis-commander`	8081	Redis admin UI

Step 4: Fix Backend Service (CRITICAL)

Backend troubleshooting steps:

# Step 1: Check backend logs

```
docker logs backend --tail 50
```

# Step 2: Look for common errors:
# - EADDRINUSE (port conflict)
# - ECONNREFUSED postgres (database issue)
# - Cannot find module (missing dependencies)
# Step 3: Restart backend

```
docker compose restart backend
```

# Step 4: Wait for startup

```
sleep 30
```

# Step 5: Test health locally

```
curl http://localhost:4000/api/health
```

# Step 6: Test via tunnel

```
curl https://psscript.morloksmaze.com/api/health
```

■ Expected Response (Success):

```
{"status":"healthy","database":"connected","redis":"connected"}
```

If Still Not Working - Restart All Services:

```
docker compose restart
```

```
sleep 60
```

```
docker compose ps
```

Step 5: Verification Checklist

After completing fixes, verify all these items:

□	Check	Test Command	Expected Result
✓	**SSH Works**	`ssh root@74.208.184.195`	No timeout
✓	**Containers Up**	`docker ps`	8 containers "Up"
✓	**Backend OK**	`curl localhost:4000/api/health`	Returns 200 OK
✓	**API Tunnel OK**	`curl https://psscript.morloksmaze.com/api/health`	No Error 1033
✓	**Frontend Loads**	`Visit https://psscript.morloksmaze.com`	Page loads
✓	**Auth Works**	`Complete Cloudflare Access login`	Login succeeds
✓	**App Functional**	`Test core features`	No console errors

Common Issues & Solutions

Issue: Backend Container Not Running

```
docker ps -a | grep backend
docker logs backend --tail 100
docker compose up -d backend
```

Issue: Database Connection Failed

```
docker ps | grep postgres
docker logs postgres --tail 50
docker compose restart postgres
sleep 15
docker compose restart backend
```

Issue: Out of Memory/Disk Space

```
free -h
df -h
docker system prune -f
systemctl restart docker
cd /opt/psscript && docker compose up -d
```

Quick Command Reference

Category	Command
Status	`docker compose ps`
Status	`docker ps -a`
Logs	`docker logs backend --tail 50`
Logs	`docker compose logs -f`
Restart	`docker compose restart backend`
Restart	`docker compose restart`
Health	`curl localhost:4000/api/health`
Health	`curl https://psscript.morloksmaze.com/api/health`
Resources	`docker stats --no-stream`
Resources	`free -h && df -h`
Emergency	`docker compose down && docker compose up -d`

Generated by Claude AI | January 14, 2026