AI-04491 SSL Certificate Error

Everything was working for months and now I'm getting this error message:

docmost.gravelcloud.com sent an invalid response.ERR_SSL_PROTOCOL_ERROR

`ERR_SSL_PROTOCOL_ERROR` happens when your browser tries to make an encrypted HTTPS connection to your server, but the server replies with unencrypted HTTP data (or drops the TLS handshake).

Since Docmost was working fine and suddenly stopped, the root cause usually boils down to **an expired SSL certificate**, **a broken reverse proxy**, or **a port/protocol mismatch**.

Step 1: Check Your Reverse Proxy & Certificates (Most Likely)

If you're running Docmost behind **Nginx Proxy Manager**, **Caddy**, **Traefik**, or **Cloudflare**:

1. **Check for an Expired Let's Encrypt Certificate**
   Auto-renewal scripts often silently fail.
   - If using **Nginx Proxy Manager** or **Traefik**, log into the dashboard and check if the SSL certificate for `docmost.gravelcloud.com` expired. Try manually forcing a certificate renewal.
   - If using **Cloudflare**, check your SSL/TLS mode under *SSL/TLS* > *Overview*. If set to **Flexible** or **Off**, set it to **Full (strict)**.
2. **Verify HTTP vs. HTTPS Target Ports**
   A common cause is a reverse proxy configured to forward HTTPS traffic to an HTTPS target port when Docmost is only listening on plain HTTP inside its Docker container.
   - Ensure your proxy routes `[https://docmost.gravelcloud.com](https://docmost.gravelcloud.com)` to `http://<DOCMOST_CONTAINER_IP>:3000` (using **HTTP**, not HTTPS, on the backend side).
3. **Check Container Logs**
   If you run Docmost via Docker Compose, check the logs for your proxy and Docmost:
   Bash

```
docker logs --tail 100 <proxy-container-name>
docker logs --tail 100 <docmost-container-name>
```

   Look for warnings about failing ACME/Let's Encrypt challenges, renewal errors, or port binding conflicts.

Step 2: Test Direct Access & Other Devices

To rule out local browser/network issues:

- **Try Incognito / Private Mode or another browser** (e.g., Firefox or Edge). Browsers aggressively cache HTTPS redirects (HSTS).
- **Test with `curl` in your terminal** to see what the server is actually returning:
  Bash

```
curl -Iv https://docmost.gravelcloud.com
```

  - If it responds with `curl: (35) error:1408F10B:SSL routines:ssl3_get_record:wrong version number`, your reverse proxy is sending unencrypted HTTP traffic back to an HTTPS request.

Step 3: Check Environment Variables

In your `docker-compose.yml` or `.env` file for Docmost, verify the `APP_URL` variable:

YAML

```
APP_URL="https://docmost.gravelcloud.com"
```

If this was recently changed to `http://` or modified, Docmost and your proxy may be generating conflicting redirect loops.

How is Docmost hosted?

If these steps don't resolve it immediately, let me know:

1. **How is Docmost deployed?** (e.g., Docker Compose on a VPS, Synology NAS, Unraid)
2. **Which reverse proxy or SSL manager are you using?** (e.g., Nginx Proxy Manager, Cloudflare, Caddy, Traefik, NPM)

I renewed all SSL certificates, is there a way to set SSL Certificates to not expire?

Here's the short answer: **No, you can't set public SSL certificates to never expire.** In fact, public web browsers and standards bodies will reject any standard SSL certificate valid for longer than 398 days (~13 months).

However, you **can set up automatic background renewal** so you never have to manually renew a certificate again.

Why Public SSL Certificates Must Expire

The 398-day cap (and upcoming proposals to shorten it to 90 days) exists for two practical reasons:

- **Security:** If a private key is leaked or compromised, an expiring certificate ensures the breach has a strict time limit.
- **Algorithmic Agility:** Expiring certificates force the web to regularly update to modern encryption standards and revoked-certificate lists.

The Real Solution: Automated Certificate Management

Instead of manual renewals, you can automate certificate issuance using the **ACME protocol** (the system Let's Encrypt uses).

How you set this up depends on your reverse proxy:

1. Nginx Proxy Manager (NPM)

NPM handles Let's Encrypt auto-renewals built-in via a background `cron` job. If your certificates expired unexpectedly, check:

- **Port 80 Access:** Let's Encrypt requires port `80` (HTTP) open to solve the HTTP-01 challenge when renewing. If port 80 is blocked by a firewall or router, auto-renewal will fail silently.
- **DNS Challenge:** If port 80 cannot be exposed to the internet, switch NPM to use **DNS-01 Challenges** (using an API key from Cloudflare, Namecheap, etc.), which allows full auto-renewal behind firewalls.

2. Caddy

If you want a truly "set it and forget it" setup, **Caddy Server** automatically provisions, manages, and silently renews SSL certificates via Let's Encrypt/ZeroSSL out of the box with zero extra setup.

3. Cloudflare Origin Certificates (If using Cloudflare Proxy)

If `docmost.gravelcloud.com` is proxied through Cloudflare (orange cloud icon active):

- You can issue a **Cloudflare Origin CA Certificate** on your server.
- Cloudflare Origin certificates can be set to be valid for **up to 15 years**.
- *Note:* This certificate only encrypts traffic between your server and Cloudflare; Cloudflare handles the public HTTPS certificate for visitors automatically.