WEB DAY 2026

DIY nell'era dell'AI: addio Vercel

Alessandro Romano
Senior Engineering Manager
Mollie

improove

By the end of this talk...

You will know **how to build** — from scratch:

- 🔔 Observability: crash alerts without Datadog
- 📊 Analytics: visitor tracking without Google
- 🚀 Deploy: CI/CD without Vercel
- 🛡️ Security: bot blocking without Cloudflare

On a €4 VPS. With AI as your pair programmer.

Who Am I? 👋

Alessandro Romano

- Senior Engineering Manager at Mollie 💳
- Father of 2
- "I like to understand how things work" 🔧

When did you last configure a server?

Last year?

5 years ago?

Never?

What PaaS gave us ✅

- Zero server configuration
- Instant SSL, CDN, previews
- Focus on product, not infrastructure
- Push to `main`. Done.

But.

What we stopped seeing

- How an HTTP server routes traffic
- How SSL certificates are renewed
- How processes restart after a crash
- How logs are collected and stored
- How deployments actually happen

Fear of what you don't understand.

The rules changed.

AI-assisted development made it possible to **experiment faster** than ever.

AI-Assisted Development

Not just faster code.

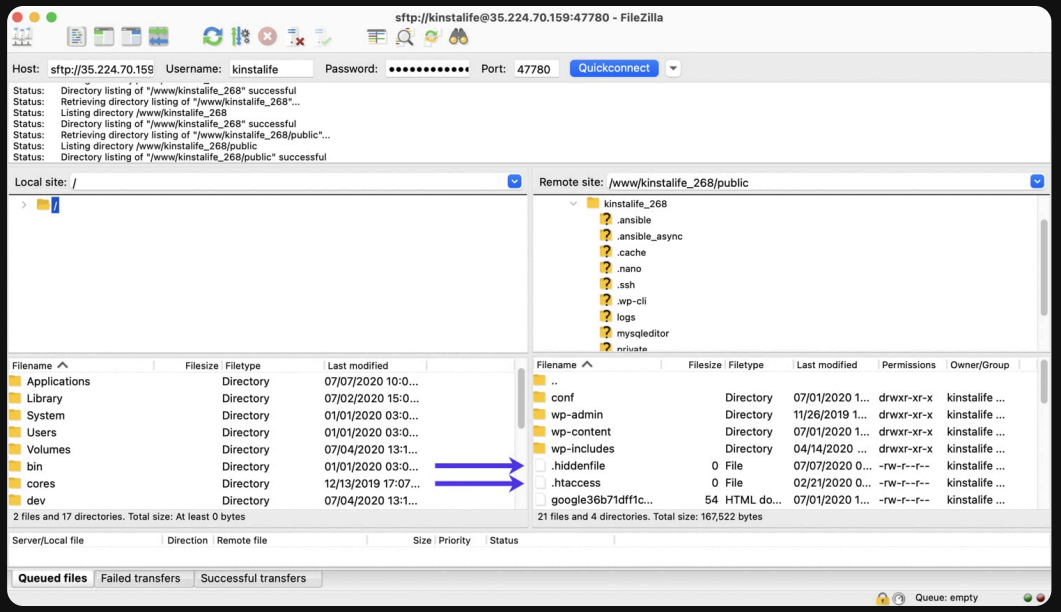
Lower activation energy for learning new things.

Explore and deep dive.

I've been here before.

2006. A PHP site. A dream.


```
1 <?php echo "Hello, World!"; ?>
```



FileZilla

Drag & Drop

Pray 🙏

My tower PC*

DynDNS

From Home 🏠

My new playground: aleromano.com

1 month paternity leave

€4/month Hetzner VPS

No PaaS & No SaaS

4 pillars

1. **Observability**
2. **Analytics**
3. **Deploy**
4. **Security**

1 / Observability

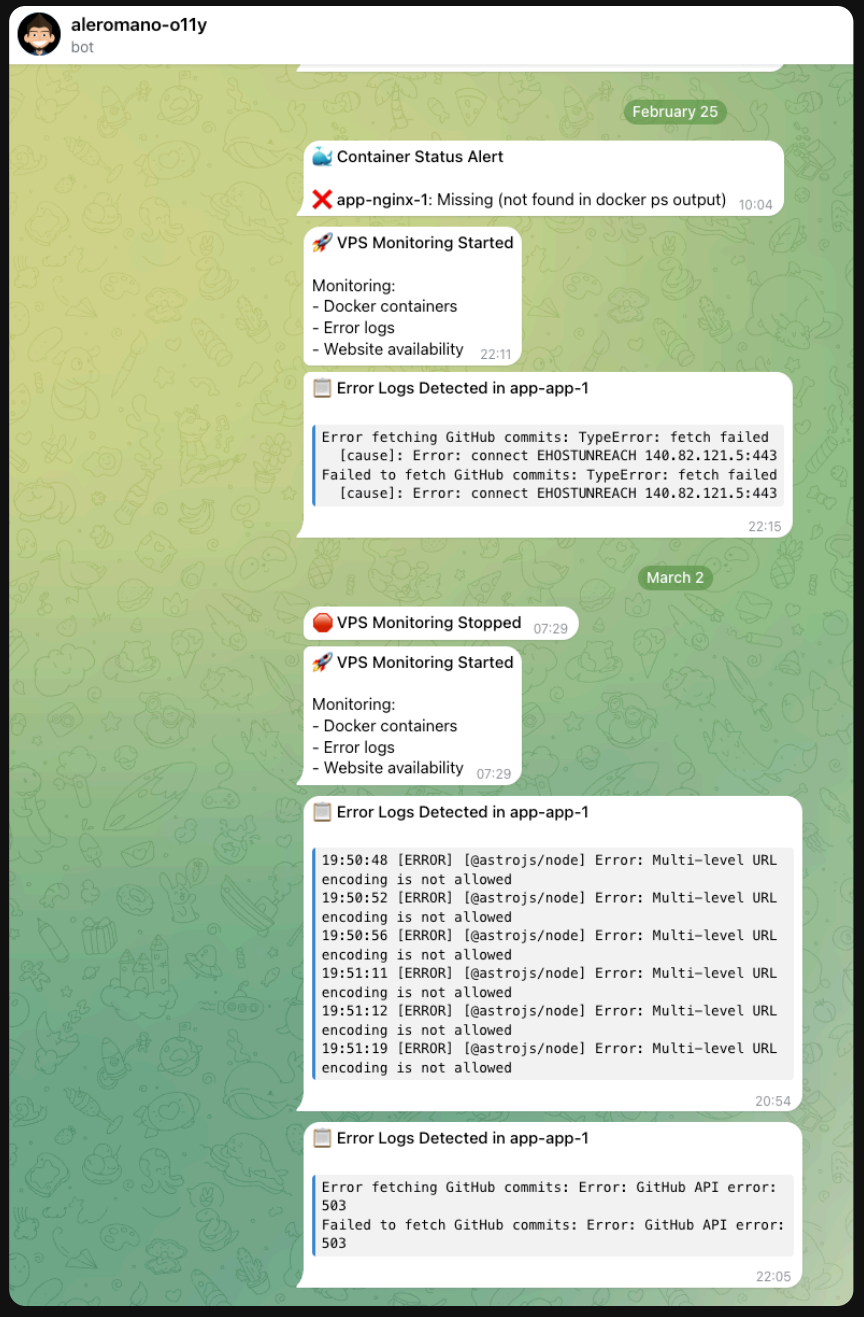
systemd + Node.js + Telegram

What it monitors

```
const CONFIG = {
  containerCheckInterval: 60000,  // every 1 min
  logCheckInterval:       300000, // every 5 min
  websiteCheckInterval:   180000, // every 3 min

  website: { url: 'https://aleromano.com' },
  containers: ['app-app-1', 'app-nginx-1', 'app-smtp-relay-1'],
  telegram: {
    botToken: process.env.TELEGRAM_BOT_TOKEN,
    chatId:   process.env.TELEGRAM_CHAT_ID,
  },
};
```

Alerts → Telegram

```
const since = Math.floor((Date.now() - CONFIG.logCheckInterval) / 10
const cmd = `docker logs --since ${since} ${containerName} 2>&1 \
  | grep -i "error|exception|fatal" | tail -n 10`;

exec(cmd, (_, stdout) => {
  if (stdout.trim())
    sendTelegramNotification(
      `📋 <b>Errors in ${containerName}</b>\n\n<pre>${stdout}</pre>
    );
});
```


aleromano-o11y
bot
February 25
Container Status Alert
app-nginx-1: Missing (not found in docker ps output) 10:04
VPS Monitoring Started
Monitoring:
- Docker containers
- Error logs
- Website availability 22:11
Error Logs Detected in app-app-1
Error fetching GitHub commits: TypeError: fetch failed
[cause]: Error: connect EHOSTUNREACH 140.82.121.5:443
Failed to fetch GitHub commits: TypeError: fetch failed
[cause]: Error: connect EHOSTUNREACH 140.82.121.5:443
22:15
March 2
VPS Monitoring Stopped 07:29
VPS Monitoring Started
Monitoring:
- Docker containers
- Error logs
- Website availability 07:29
Error Logs Detected in app-app-1
19:50:48 [ERROR] [@astrojs/node] Error: Multi-level URL encoding is not allowed
19:50:52 [ERROR] [@astrojs/node] Error: Multi-level URL encoding is not allowed
19:50:56 [ERROR] [@astrojs/node] Error: Multi-level URL encoding is not allowed
19:51:11 [ERROR] [@astrojs/node] Error: Multi-level URL encoding is not allowed
19:51:12 [ERROR] [@astrojs/node] Error: Multi-level URL encoding is not allowed
19:51:19 [ERROR] [@astrojs/node] Error: Multi-level URL encoding is not allowed
20:54
Error Logs Detected in app-app-1
Error fetching GitHub commits: Error: GitHub API error: 503
Failed to fetch GitHub commits: Error: GitHub API error: 503
22:05

What is systemd?

The init system that Linux runs **first** on boot.

- Starts and stops services
- Restarts them on crash
- Manages dependencies between services
- Replaces cron, nohup, screen, forever, pm2

The systemd unit file

```
[Unit]
Description=VPS Observability Daemon
After=network.target docker.service
Requires=docker.service

[Service]
Type=simple
User=deploy
Group=docker
ExecStart=/usr/bin/node /opt/scripts/monitor.js
Restart=always
RestartSec=10
Environment=TELEGRAM_BOT_TOKEN=...
Environment=TELEGRAM_CHAT_ID=...
StandardOutput=append:/var/log/observability.log
```

But what if you need graphs?

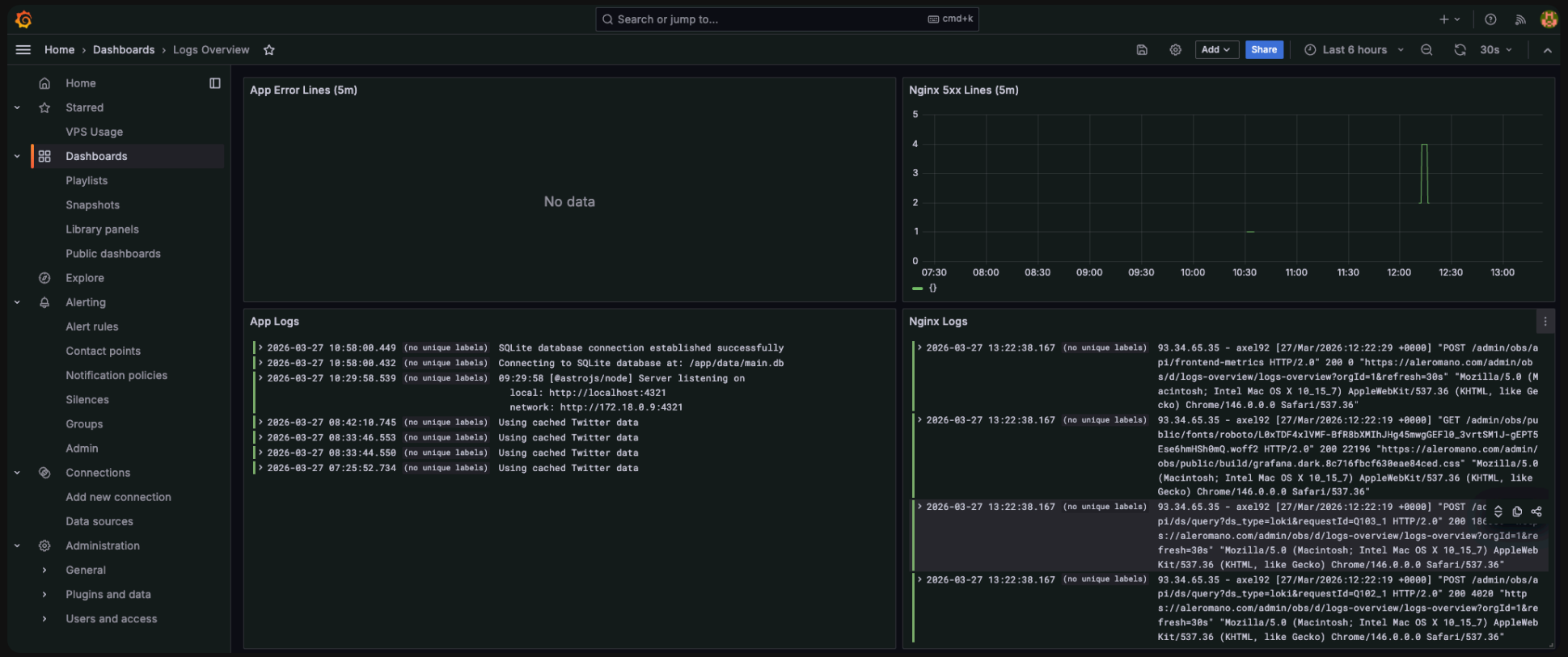
Dashboards? Retention? Alerting rules?

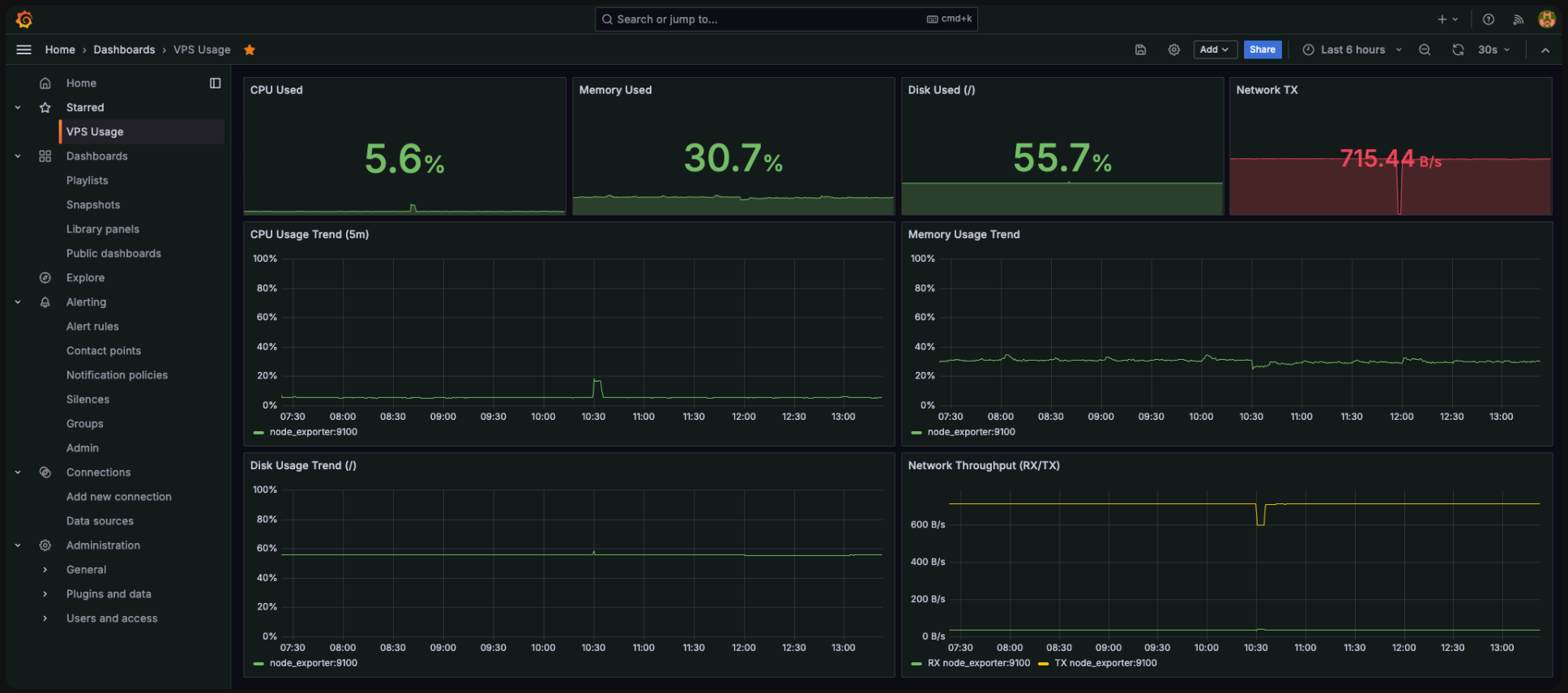
You need Prometheus + Loki + Grafana.

Sounds scary?

Grafana stack in docker-compose

```yaml
services:
  grafana:
    image: grafana/grafana:latest
    ports: ["3000:3000"]
    volumes: [grafana-data:/var/lib/grafana]

  prometheus:
    image: prom/prometheus:latest
    volumes: [./prometheus.yml:/etc/prometheus/prometheus.yml]

  loki:
    image: grafana/loki:latest
```


Search or jump to... cmd+k
Home › Dashboards › Logs Overview
Add
Share
Last 6 hours
30s
Home
Starred
VPS Usage
Dashboards
Playlists
Snapshots
Library panels
Public dashboards
Explore
Alerting
Alert rules
Contact points
Notification policies
Silences
Groups
Admin
Connections
Add new connection
Data sources
Administration
General
Plugins and data
Users and access
App Error Lines (5m)
No data
Nginx 5xx Lines (5m)
5
4
3
2
1
0
07:30
08:00
08:30
09:00
09:30
10:00
10:30
11:00
11:30
12:00
12:30
13:00
{}
App Logs
2026-03-27 10:58:00.449 (no unique labels) SQLite database connection established successfully
2026-03-27 10:58:00.432 (no unique labels) Connecting to SQLite database at: /app/data/main.db
2026-03-27 10:29:58.539 (no unique labels) 09:29:58 [@astrojs/node] Server listening on
local: http://localhost:4321
network: http://172.18.0.9:4321
2026-03-27 08:42:10.745 (no unique labels) Using cached Twitter data
2026-03-27 08:33:46.553 (no unique labels) Using cached Twitter data
2026-03-27 08:33:44.550 (no unique labels) Using cached Twitter data
2026-03-27 07:25:52.734 (no unique labels) Using cached Twitter data
Nginx Logs
2026-03-27 13:22:38.167 (no unique labels) 93.34.65.35 - axel92 [27/Mar/2026:12:22:29 +0000] "POST /admin/obs/api/frontend-metrics HTTP/2.0" 200 0 "https://aleromano.com/admin/obs/d/logs-overview/logs-overview?orgId=1&refresh=30s" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/146.0.0.0 Safari/537.36"
2026-03-27 13:22:38.167 (no unique labels) 93.34.65.35 - axel92 [27/Mar/2026:12:22:19 +0000] "GET /admin/obs/public/fonts/roboto/L0xTDF4xlVMF-BfR8bXMIhJHg45mwgGEFl0_3vrtSM1J-gEPT5Ese6hmHSh0mQ.woff2 HTTP/2.0" 200 22196 "https://aleromano.com/admin/obs/public/build/grafana.dark.8c716fbcf630eae84ced.css" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/146.0.0.0 Safari/537.36"
2026-03-27 13:22:38.167 (no unique labels) 93.34.65.35 - axel92 [27/Mar/2026:12:22:19 +0000] "POST /admin/obs/api/ds/query?ds_type=loki&requestId=Q103_1 HTTP/2.0" 200 18
s://aleromano.com/admin/obs/d/logs-overview/logs-overview?orgId=1&refresh=30s" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/146.0.0.0 Safari/537.36"
2026-03-27 13:22:38.167 (no unique labels) 93.34.65.35 - axel92 [27/Mar/2026:12:22:19 +0000] "POST /admin/obs/api/ds/query?ds_type=loki&requestId=Q102_1 HTTP/2.0" 200 4020 "https://aleromano.com/admin/obs/d/logs-overview/logs-overview?orgId=1&refresh=30s" "Mozilla/5.0 (Macintosh; Intel Mac OS X 10_15_7) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/146.0.0.0 Safari/537.36"


Search or jump to...
cmd+k
Home › Dashboards › VPS Usage
Add
Share
Last 6 hours
30s
Home
Starred
VPS Usage
Dashboards
Playlists
Snapshots
Library panels
Public dashboards
Explore
Alerting
Alert rules
Contact points
Notification policies
Silences
Groups
Admin
Connections
Add new connection
Data sources
Administration
General
Plugins and data
Users and access
CPU Used
5.6%
Memory Used
30.7%
Disk Used (/)
55.7%
Network TX
715.44 B/s
CPU Usage Trend (5m)
100%
80%
60%
40%
20%
0%
07:30 08:00 08:30 09:00 09:30 10:00 10:30 11:00 11:30 12:00 12:30 13:00
node_exporter:9100
Memory Usage Trend
100%
80%
60%
40%
20%
0%
07:30 08:00 08:30 09:00 09:30 10:00 10:30 11:00 11:30 12:00 12:30 13:00
node_exporter:9100
Disk Usage Trend (/)
100%
80%
60%
40%
20%
0%
07:30 08:00 08:30 09:00 09:30 10:00 10:30 11:00 11:30 12:00 12:30 13:00
node_exporter:9100
Network Throughput (RX/TX)
600 B/s
400 B/s
200 B/s
0 B/s
07:30 08:00 08:30 09:00 09:30 10:00 10:30 11:00 11:30 12:00 12:30 13:00
RX node_exporter:9100
TX node_exporter:9100

2 / Analytics

TypeScript + SQLite + no cookies 

Client side: fire and forget

```
function sendEvent(event: AnalyticsEvent): void {
  fetch('/api/analytics/collect', {
    method: 'POST',
    body: JSON.stringify(event),
    headers: { 'Content-Type': 'application/json' },
    keepalive: true, // survives tab close
  }).catch(() => {}); // never break the UX
}

// Respects DNT and Global Privacy Control
if (!shouldRespectPrivacy()) {
  sendPageView();
  initClickTracking();
  initTimeTracking();
}
```

Server side: privacy by design

```
1 // Hash = SHA256(IP + UserAgent + Date + Salt)
2 // Rotates daily — no cross-day tracking, no cookies
3 function generateVisitorHash(ip: string, userAgent: string): string {
4   const today = new Date().toISOString().split('T')[0];
5   const input = `${ip}|${userAgent}|${today}|${HASH_SALT}`;
6   return createHash('sha256').update(input).digest('hex').substring(0
7 }
```

SQLite: the database you already have

- A **file** on disk — zero config, zero server
- Add it to Node.js with `better-sqlite3`
- Fast enough for millions of rows
- Full SQL: joins, aggregations, indexes

The right tool for a personal site.

The schema

```sql
CREATE TABLE analytics_visits (
  id            INTEGER PRIMARY KEY AUTOINCREMENT,
  visitor_hash  TEXT    NOT NULL,
  path          TEXT    NOT NULL,
  referrer      TEXT,
  timestamp     INTEGER NOT NULL
);

CREATE INDEX idx_visits_path      ON analytics_visits(path);
CREATE INDEX idx_visits_timestamp ON analytics_visits(timestamp);

CREATE TABLE analytics_events (
  id           INTEGER PRIMARY KEY AUTOINCREMENT,
  visitor_hash TEXT    NOT NULL,
  event_type   TEXT    NOT NULL,
```

Querying: top pages & unique visitors

```sql
-- Top pages in the last 7 days
SELECT path, COUNT(*) AS views
FROM analytics_visits
WHERE timestamp > strftime('%s', 'now', '-7 days')
GROUP BY path
ORDER BY views DESC;

-- Unique visitors today
SELECT COUNT(DISTINCT visitor_hash) AS visitors
FROM analytics_visits
WHERE timestamp > strftime('%s', 'now', 'start of day');
```

aleromano.com/admin/analytics

Alessandro Romano

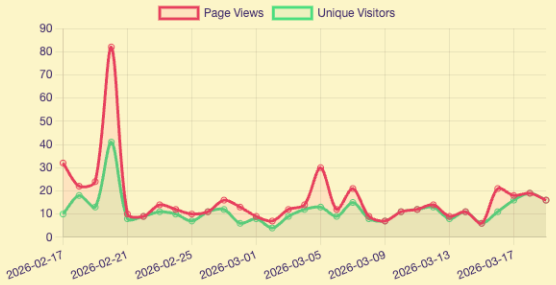
HOME ABOUT BLOG CONTACT

Analytics Dashboard

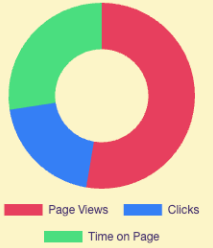
7d 30d 90d

Total Page Views	Unique Visitors	Total Events	Avg. Time on Page
513	364	461	27m 5s

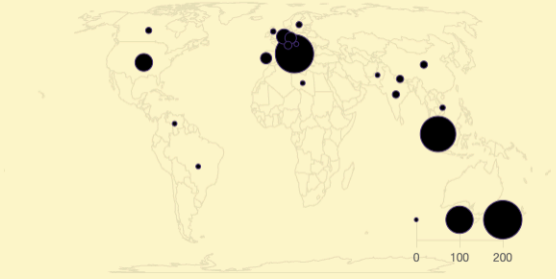
Daily Traffic



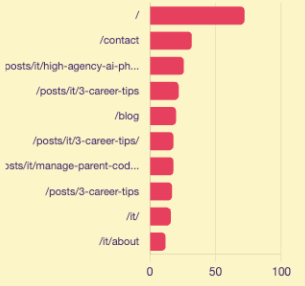
Event Breakdown



Visitors by Country



Top Pages



3 / Deploy

GitHub Actions + Docker 🐳

The Dockerfile

```dockerfile
FROM node:20-alpine AS base
WORKDIR /app

# Copy only package files first → cache layer
COPY package.json package-lock.json ./

FROM base AS build-deps
RUN npm install
FROM base AS prod-deps
RUN npm install --omit=dev

FROM build-deps AS build
COPY . .
RUN npm run build

FROM base AS runtime
```

docker-compose: wiring it all together

- One file defines every service — app, nginx, SMTP relay
- Handles networking, volumes, and restart policies
- Two files: `docker-compose.yml` (base) + `docker-compose.prod.yml` (overrides)
- A single command brings the whole stack up or down

The docker-compose.yml

```yaml
services:
  app:
    image: ghcr.io/aleromano92/aleromano.com:latest
    restart: unless-stopped
    volumes:
      - analytics-data:/app/data

  nginx:
    image: nginx:alpine
    restart: unless-stopped
    ports:
      - "80:80"
      - "443:443"
    volumes:
      - ./nginx.conf:/etc/nginx/nginx.conf:ro
```

GitHub Actions: CI/CD in 3 steps

Triggered on every push to `main`

- 1️⃣ Checkout code & authenticate to GHCR
- 2️⃣ Build Docker image & push to registry
- 3️⃣ SSH into Hetzner & restart the stack

Free for public repos.

Step 1: checkout & authenticate

```yaml
- name: Checkout repository
  uses: actions/checkout@v4

- name: Log in to GitHub Container Registry
  uses: docker/login-action@v3
  with:
    registry: ghcr.io
    username: ${{ github.actor }}
    password: ${{ secrets.GITHUB_TOKEN }}
```

Step 2: build & push to GHCR

```yaml
- name: Build and push Docker image
  uses: docker/build-push-action@v5
  with:
    context: .
    push: true
    tags: ghcr.io/${{ github.repository }}:latest
    cache-from: type=gha
    cache-to: type=gha,mode=max
```

Step 3: SSH into Hetzner & deploy

```yaml
- name: Deploy to Hetzner
  uses: appleboy/ssh-action@master
  with:
    host: ${{ secrets.HETZNER_HOST }}
    key: ${{ secrets.HETZNER_SSH_KEY }}
    script: |
      cd ~/app
      git reset --hard origin/main
      docker pull ghcr.io/${{ github.repository }}:latest
      docker-compose -f docker-compose.yml \
        -f docker-compose.prod.yml down
      docker-compose -f docker-compose.yml \
        -f docker-compose.prod.yml up -d
      docker image prune -f
```

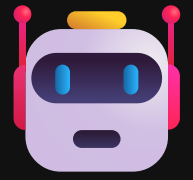
4 / Security

nginx rules, no Cloudflare 🛡️

Open your access logs.

```
188.245.236.17 "GET /wp-admin/setup-config.php" 444
45.33.32.156   "GET /phpmyadmin/index.php" 444
91.212.166.22  "GET /.env" 444
185.220.101.4  "GET /?id=1 UNION SELECT username,password FROM users"
```

Your site runs Astro. They're looking for WordPress.

The internet is mostly bots. 

- .env hunters looking for leaked secrets
- WordPress scanners (your site runs Astro)
- SQL injection on a static blog
- Vulnerability scanners probing every port

Cloudflare hides this from you.

nginx shows you the truth.

444 — the silent drop

```
# Requests via IP (not domain) → connection closed, no response
server {
    listen 80 default_server;
    listen 443 default_server;
    ssl_reject_handshake on;
    return 444;
}

# WordPress/phpmyadmin hunters
location ~* /(wp-admin|phpmyadmin|cpanel) {
    return 444;
}

# PHP probes on a Node.js server
location ~* \.(php|asp|jsp|cgi)$ {
```

SQL injection on a static blog 😂

```
location ~* \?(.*)(select|union|insert|drop|delete)(.*)$ {
    return 444;
}
```

Someone tried to DROP TABLE my blog posts.

Rate limiting without a WAF

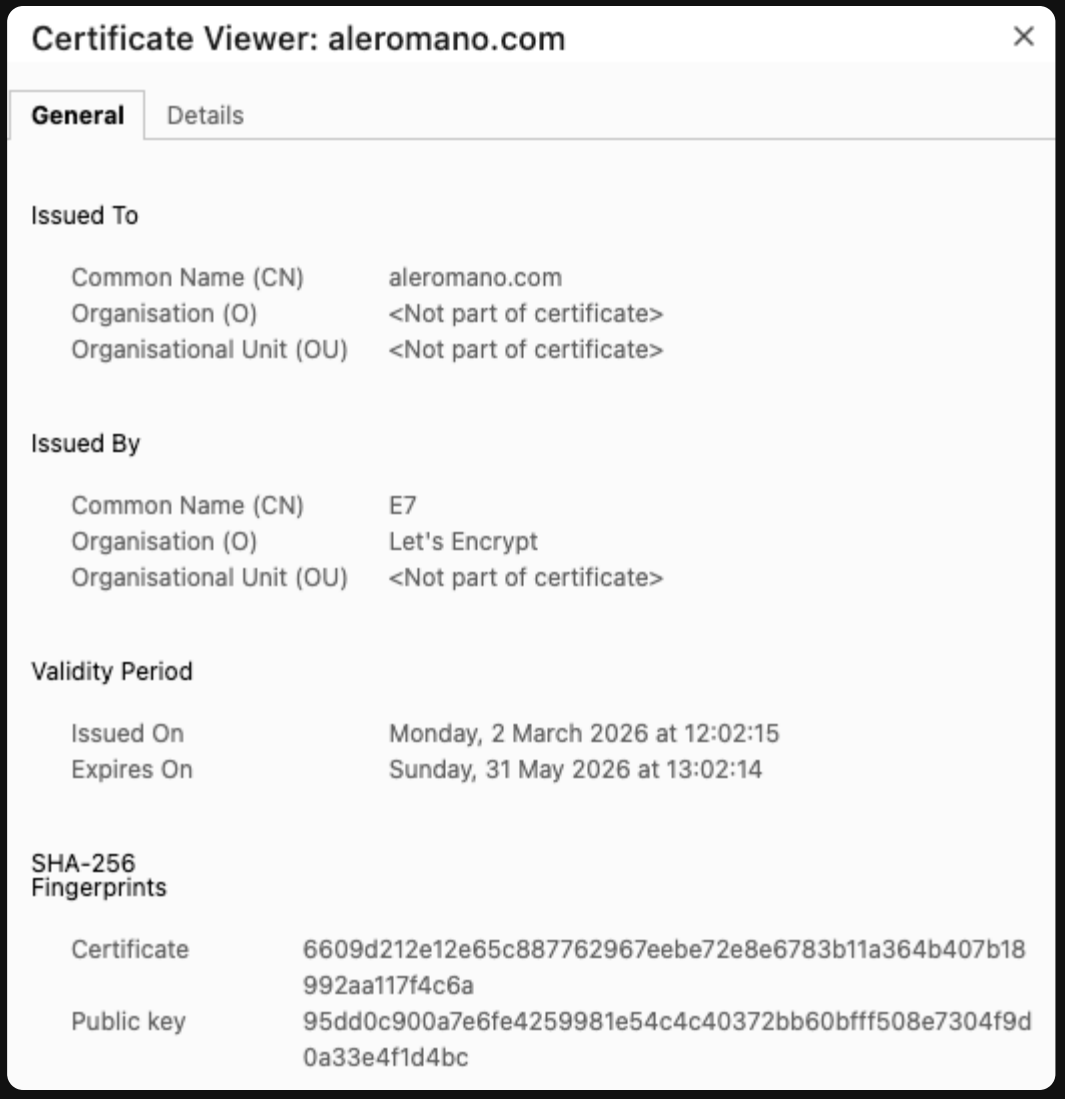
```
# 5 requests/minute per IP on /admin
limit_req_zone $binary_remote_addr zone=ADMIN:1m rate=5r/m;

location /admin {
    limit_req zone=ADMIN burst=2 nodelay;
    limit_req_status 429;
}
```

Enough for you. Nothing for a brute force attack.

Let's Encrypt: free SSL for everyone

- A non-profit Certificate Authority — trusted by all browsers
- Issues certificates in **seconds**, for free
- Valid 90 days — designed for automation
- Powers ~350 million active certificates

Before 2015: SSL cost €50–200/year per domain.


Certificate Viewer: aleromano.com
General
Details
Issued To
Common Name (CN) aleromano.com
Organisation (O) <Not part of certificate>
Organisational Unit (OU) <Not part of certificate>
Issued By
Common Name (CN) E7
Organisation (O) Let's Encrypt
Organisational Unit (OU) <Not part of certificate>
Validity Period
Issued On Monday, 2 March 2026 at 12:02:15
Expires On Sunday, 31 May 2026 at 13:02:14
SHA-256 Fingerprints
Certificate 6609d212e12e65c887762967eebe72e8e6783b11a364b407b18992aa117f4c6a
Public key 95dd0c900a7e6fe4259981e54c4c40372bb60bfff508e7304f9d0a33e4f1d4bc

certbot

```
# Get a certificate and auto-configure nginx
certbot --nginx -d aleromano.com -d www.aleromano.com

# certbot writes this into your nginx config:
# ssl_certificate     /etc/letsencrypt/live/aleromano.com/fullchain.p
# ssl_certificate_key /etc/letsencrypt/live/aleromano.com/privkey.pem

# Auto-renewal runs via a systemd timer — already active after instal
systemctl status certbot.timer
```

What you get without Cloudflare

- HTTPS with Let's Encrypt (free)
- Bot blocking at the nginx layer
- Rate limiting on sensitive routes
- Security headers: HSTS, CSP, X-Frame-Options
- Zero vendor dependency

Not better than Cloudflare.

But you built it. You understand it.

What we just built together

- 🔔 **Observability** — Node.js daemon + systemd + Telegram
- 📊 **Analytics** — TypeScript + SQLite + SHA256 hashing
- 🚀 **Deploy** — Dockerfile + GitHub Actions + Hetzner VPS
- 🛡️ **Security** — nginx 444 + rate limiting, no WAF

Less magic.

More understanding.

The opportunity

AI + fundamentals = **more work, not less.**

*For those who use AI **and** know the basics.*

What you can go build tomorrow

- 🔔 0-noise alerts
- 📊 Privacy-first analytics
- 🚀 Full CI/CD pipeline
- 🛡️ Bot & brute-force protection

Blog: aleromano.com | **X**: @_aleromano | **LinkedIn**: /in/alessandroromano92

WEB DAY 2026

Kudos

improove