

AWS · EC2 · 2026 EDITION

AWS EC2 Setup Guide

Launch a server on the AWS Free plan, connect with Tabby, and run n8n with Docker Compose and HTTPS.

- Written for the new AWS Free plan (accounts made on or after 15-Jul-2025)
- Every command is copy-paste ready
- Works on Windows, macOS and Linux

Updated 30-Sep-2026

Companion to the Grounded AI Lab video **Free Cloud VPS in 2026: AWS vs Oracle vs Google Cloud**

Free-tier limits change often. Every section links to the official page, so check it before you rely on a number. Prices are approximate.

Contents

[01](#) What changed in 2026

[02](#) Prerequisites

[03](#) Create your AWS account

[04](#) Open the EC2 dashboard

[05](#) Launch an EC2 instance

[06](#) Storage and network

[07](#) Security group and key pair

[08](#) Connect with Tabby

[09](#) Install Docker and Docker Compose

[10](#) Deploy n8n with Docker Compose

[11](#) HTTPS with nip.io and Certbot

[12](#) Migrate existing n8n workflows

[13](#) Public IP vs Elastic IP

[14](#) Credits and cost control

[15](#) Monitoring: RAM, CPU and disk

[16](#) Troubleshooting

[17](#) Cheat sheet

What changed in 2026

AWS replaced the old 12-month free tier for new accounts on **15-Jul-2025**. Most tutorials online still describe the old one, so check which model your account is on before you follow anything.

Your account was created	What you get
On or after 15-Jul-2025 (Free plan)	\$100 credit at sign-up, plus up to \$100 more for finishing getting-started tasks (such as launching an EC2 instance or creating a budget). The Free plan lasts 6 months or until the credit runs out, whichever comes first. Then the account closes unless you upgrade to a paid plan. Always-free services, like Lambda's 1 million requests a month, keep going.
Before 15-Jul-2025 (legacy free tier)	750 hours a month of t2.micro or t3.micro for 12 months from sign-up, plus 30 GB of EBS storage and 750 hours of public IPv4. Nothing changes for you until those 12 months end.

What this means in practice

On the Free plan, your server, its disk and its public IP all draw from your credit. Small servers stretch it for the full 6 months; big ones burn it in weeks. Section 14 shows the maths.

This guide walks you through:

- Creating an AWS account on the Free plan
- Launching an EC2 server that fits inside your credit
- Connecting securely with the Tabby SSH client
- Installing Docker and Docker Compose
- Running n8n with HTTPS
- Moving existing n8n workflows across

Why Docker Compose?

Plain Docker is fine for one container. Compose puts your whole stack in one `docker-compose.yml`: one command starts it, and volumes, networks and settings are written down, so moving servers later is easy.

02 Prerequisites

You need	Details
Computer	Windows 10/11, macOS or Linux
Email address	A real one. Disposable addresses get flagged.
Credit or debit card	Used to verify you. Prepaid cards are often rejected.
Phone number	For the verification call or SMS
Tabby	Free SSH client from tabby.sh (any SSH client works)
Basic terminal skills	You'll paste and run commands

03 Create your AWS account

Already have one? Skip to section 4.

3.1 Sign up

- 1 Go to **aws.amazon.com** and click **Create an AWS Account**.
- 2 Enter your email and an account name, then verify the email with the 6-digit code.
- 3 Set a strong root password (use a password manager).
- 4 Choose **Personal** as the account type unless it's for a business, and fill in your contact details.

3.2 Choose your plan

- 5 When asked, pick the **Free plan**. You can upgrade to a paid plan any time; you can't go back.
- 6 Add your card. AWS may place a small temporary hold to verify it, which is released.
- 7 Verify your phone number by SMS or voice call.
- 8 Choose **Basic support (free)** and finish sign-up.

After sign-up

Activation can take a few minutes. Then sign in at **console.aws.amazon.com**. Turn on MFA for the root user straight away: search **IAM** → **Add MFA**.

04 Open the EC2 dashboard

- 1 In the console's top search bar, type **EC2** and open it.
- 2 Pick your region in the top-right corner. It's where your server lives, so choose one near your users.

Region	Code	Good for
Asia Pacific (Mumbai)	ap-south-1	India
Asia Pacific (Hyderabad)	ap-south-2	India
Asia Pacific (Singapore)	ap-southeast-1	South-East Asia
US East (N. Virginia)	us-east-1	US; most services
Europe (Ireland)	eu-west-1	Europe

05 Launch an EC2 instance

- 1 On the EC2 dashboard, click **Launch instance**.
- 2 Under **Name and tags**, name it, e.g. `n8n-server`.
- 3 Under **Application and OS Images**, choose **Ubuntu** → **Ubuntu Server 24.04 LTS, 64-bit (x86)**.
- 4 Under **Instance type**, pick one marked **Free tier eligible** (table below).

Instance type	vCPU	RAM	Approx. cost if left on 24/7*	Use it for
t3.micro	2	1 GB	~\$7.60 / month	Learning, light n8n (add swap: section 15)
t3.small	2	2 GB	~\$15 / month	n8n with real workflows (recommended)
c7i-flex.large	2	4 GB	~\$62 / month	Short experiments only
m7i-flex.large	2	8 GB	~\$70 / month	Short experiments only

*us-east-1 on-demand Linux prices, Sep-2026; Mumbai is similar. These come out of your credit, not your card. The t4g.micro and t4g.small (ARM) types are cheaper still, but need the ARM (arm64) Ubuntu image.

Legacy accounts (before 15-Jul-2025)

Pick **t2.micro** or **t3.micro**: they're the only types covered by your 750 free hours.

06 Storage and network

6.1 Storage

- 1 Under **Configure storage**, change the default 8 GB to **20 GB**, type **gp3**. That's about \$1.60 a month from your credit (legacy accounts get 30 GB free).

6.2 Optional: update on first boot

Expand **Advanced details** and paste this into **User data**:

```
#!/bin/bash
apt-get update -y
apt-get upgrade -y
```

6.3 Network settings

- VPC: default
- Subnet: no preference
- Auto-assign public IP: **Enable**

Your public IP costs a little

AWS charges about \$0.005 an hour (~\$3.60 a month) for every public IPv4 address. On the Free plan it comes out of your credit. Without a public IP you can't SSH in from home, so keep it.

07 Security group and key pair

7.1 Create a key pair

- 1 Under **Key pair (login)**, click **Create new key pair**.
- 2 Name it (e.g. `ec2-key`), type **RSA**, format **.pem**.
- 3 Click **Create key pair**. The file downloads once. AWS does not keep a copy.

Guard your .pem file

Lose it and you can't log in: you'd have to rebuild the server. Keep it in `~/.ssh/` and back it up to a password manager or an encrypted drive. Never commit it to Git.

On macOS or Linux, lock down its permissions:

```
chmod 400 ~/.ssh/ec2-key.pem
```

7.2 Inbound rules

Type	Port	Source	Why
SSH	22	My IP	Your terminal access; only from your IP
HTTP	80	Anywhere	Needed once for the free HTTPS certificate
HTTPS	443	Anywhere	Secure web traffic
Custom TCP	5678	Anywhere	The n8n editor

- 4 Review everything and click **Launch instance**. It's ready in 1–2 minutes.

08 Connect with Tabby

- 1 In **EC2** → **Instances**, select your server and copy its **Public IPv4 address**.
- 2 Open Tabby → + → **SSH connection**, and fill in the table below.
- 3 Click **Save**, then **Connect**. Type `yes` to trust the server the first time.

Field	Value
Host	Your public IPv4 address
Port	22
Username	<code>ubuntu</code>
Auth method	Private key → select your .pem file

You're in

You'll see a prompt like `ubuntu@ip-172-31-xx-xx:~$`.

09 Install Docker and Docker Compose

Run these on the server. They install Docker Engine and the Compose v2 plugin from Docker's own repository.

```
sudo apt-get update -y
sudo apt-get install -y ca-certificates curl
sudo install -m 0755 -d /etc/apt/keyrings
sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg -o /etc/apt/keyrings/docker.asc
sudo chmod a+r /etc/apt/keyrings/docker.asc
echo "deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.asc] \
    https://download.docker.com/linux/ubuntu $(. /etc/os-release && echo "$VERSION_CODENAME")
    stable" | \
    sudo tee /etc/apt/sources.list.d/docker.list > /dev/null
sudo apt-get update -y
sudo apt-get install -y docker-ce docker-ce-cli containerd.io docker-buildx-plugin docker-
compose-plugin
```

Run Docker without `sudo`, start it on boot, and check it works:

```
sudo usermod -aG docker $USER
newgrp docker
sudo systemctl enable --now docker
docker --version && docker compose version
docker run hello-world
```

Deploy n8n with Docker Compose

```
mkdir -p ~/n8n/certs && cd ~/n8n
nano docker-compose.yml
```

Paste this, replacing every `YOUR_IP` with your server's IP (e.g. `54.161.22.33`):

```
services:
  n8n:
    image: n8nio/n8n:latest
    container_name: n8n
    restart: unless-stopped
    ports:
      - "5678:5678"
    environment:
      - N8N_HOST=YOUR_IP.nip.io
      - N8N_PORT=5678
      - N8N_PROTOCOL=https
      - NODE_ENV=production
      - WEBHOOK_URL=https://YOUR_IP.nip.io:5678/
      - N8N_EDITOR_BASE_URL=https://YOUR_IP.nip.io:5678/
      - N8N_SECURE_COOKIE=true
      - N8N_SSL_KEY=/certs/privkey.pem
      - N8N_SSL_CERT=/certs/fullchain.pem
      - GENERIC_TIMEZONE=Asia/Kolkata
    volumes:
      - n8n_data:/home/node/.n8n
      - ./certs:/certs
volumes:
  n8n_data:
```

Save and exit: **Ctrl+O**, **Enter**, **Ctrl+X**. Don't start it yet: it needs the certificate from section 11.

What's nip.io?

A free DNS service that turns an IP into a hostname: `54.161.22.33.nip.io` points to `54.161.22.33`. Certificates and Google sign-in need a hostname; they reject bare IPs.

Command	What it does
<code>docker compose up -d</code>	Start n8n in the background
<code>docker compose ps</code>	Is it running?
<code>docker compose logs -f</code>	Live logs
<code>docker compose restart</code>	Restart n8n
<code>docker compose pull && docker compose up -d</code>	Update n8n
<code>docker compose down</code>	Stop and remove the container (data stays)
<code>docker compose down -v</code>	Deletes your data volume too. Careful.

11 HTTPS with nip.io and Certbot

Let's Encrypt gives you a free certificate. It checks you control the host over port 80, which you opened in section 7.

```
sudo apt-get install -y certbot
sudo certbot certonly --standalone -d YOUR_IP.nip.io \
  --non-interactive --agree-tos --email you@example.com
sudo cp /etc/letsencrypt/live/YOUR_IP.nip.io/{fullchain,privkey}.pem ~/n8n/certs/
sudo chmod 644 ~/n8n/certs/*
cd ~/n8n && docker compose up -d && docker compose logs -f
```

Open https://YOUR_IP.nip.io:5678 and create your n8n owner account.

11.1 Make renewals reach n8n

Fixed in this edition

Certificates last 90 days. Certbot renews them automatically, but only in `/etc/letsencrypt`: the copies n8n reads in `~/n8n/certs` would go stale and n8n would stop loading after ~90 days. This hook copies the renewed files and restarts n8n every time.

```
sudo tee /etc/letsencrypt/renewal-hooks/deploy/n8n-certs.sh > /dev/null <<'EOF'
#!/bin/bash
cp /etc/letsencrypt/live/YOUR_IP.nip.io/{fullchain,privkey}.pem /home/ubuntu/n8n/certs/
chmod 644 /home/ubuntu/n8n/certs/*
cd /home/ubuntu/n8n && docker compose restart n8n
EOF
sudo chmod +x /etc/letsencrypt/renewal-hooks/deploy/n8n-certs.sh
sudo certbot renew --dry-run
```

12 Migrate existing n8n workflows

Your workflows, credentials and history live in the `.n8n` folder. On the **old** server:

```
tar -czf ~/n8n-backup.tar.gz ~/.n8n
ls -lh ~/n8n-backup.tar.gz
```

Copy the file to the new server with Tabby's built-in SFTP (same host, key and user as SSH), into `/home/ubuntu/`. Then on the **new** server:

```
cd ~/n8n && docker compose down
tar -xzf ~/n8n-backup.tar.gz -C /
VOL=$(docker volume inspect n8n_n8n_data --format '{{.Mountpoint}}')
sudo cp -r /home/ubuntu/.n8n/. "$VOL"
sudo chown -R 1000:1000 "$VOL"
docker compose up -d
```

Log in with your OLD account

Don't create a new owner account: the backup already holds your old login. If your old n8n used a custom `N8N_ENCRYPTION_KEY`, set the same key on the new server or your saved credentials won't decrypt.

- Test each workflow on the new server.
- Reconnect OAuth credentials (Google etc.); tokens often expire.
- Point third-party webhooks at the new URL.
- Only then stop the old server.

13 Public IP vs Elastic IP

	Public IP (default)	Elastic IP
Changes when you stop/start?	Yes	No, it's fixed
Cost	~\$3.60 / month while in use	~\$3.60 / month, attached or not
Good for	Learning and testing	Anything with webhooks, OAuth or DNS

A new IP breaks your nip.io hostname, certificate and webhooks, so use an Elastic IP once n8n matters:

- 1 EC2 → **Elastic IPs** → **Allocate Elastic IP address** → **Allocate**.
- 2 Select it → **Actions** → **Associate Elastic IP address** → pick your instance → **Associate**. Tick **Allow reassociation** so you can move it to a rebuilt server later.
- 3 Update `YOUR_IP` in `docker-compose.yml`, re-run the certificate step (section 11), and restart n8n.
- 4 When you delete the server, **release** the Elastic IP too, or it keeps billing.

14 Credits and cost control

14.1 Set a budget before anything else

- 1 Search **Billing and Cost Management** → **Budgets** → **Create budget**.
- 2 Choose the **Zero spend budget** template, add your email, and create it. You'll get an email the moment anything costs real money.
- 3 Check your remaining credit and days any time under **Billing and Cost Management** → **Credits**.

14.2 How far \$100 goes (approximate)

Setup, running 24/7	Per month	6 months
t3.micro + 20 GB gp3 + public IP	~\$13	~\$78: fits the \$100 sign-up credit
t3.small + 20 GB gp3 + public IP	~\$21	~\$126: needs some of the extra \$100
m7i-flex.large + 20 GB gp3 + public IP	~\$75	Credit gone in 2–3 months

- **Stop** (don't terminate) the server when you're not using it: compute stops charging, the disk and IP don't.
- Two servers at once cost double. Delete what you don't use: instances, volumes, snapshots, Elastic IPs.
- Before month 6, decide: upgrade to a paid plan (you keep the server and pay for usage), or back up and let the account close.
- Want a fixed monthly price instead? AWS Lightsail is the simple next step.

15 Monitoring: RAM, CPU and disk

```
free -h      # RAM (look at "available")
df -h        # disk space
htop         # live CPU and RAM (q to quit)
docker stats # per-container usage (Ctrl+C to quit)
```

15.1 Add swap (strongly recommended on 1–2 GB servers)

```
sudo fallocate -l 2G /swapfile
sudo chmod 600 /swapfile
sudo mkswap /swapfile
sudo swapon /swapfile
echo '/swapfile none swap sw 0 0' | sudo tee -a /etc/fstab
free -h
```

16 Troubleshooting

Problem	Fix
SSH times out or is refused	Port 22 open to your current IP ? (Home IPs change: edit the rule.) Instance running with 2/2 checks passed? Username <code>ubuntu</code> ?
UNPROTECTED PRIVATE KEY FILE	<code>chmod 400 ~/.ssh/ec2-key.pem</code>
Instance type not eligible	Pick one marked Free tier eligible in the launch wizard (section 5).
n8n secure-cookie error	You're on plain HTTP. Finish section 11, or for a quick test set <code>N8N_SECURE_COOKIE=false</code> and restart.
docker: permission denied	<code>sudo usermod -aG docker \$USER</code> then <code>newgrp docker</code>
docker compose: not found	Use <code>docker compose</code> (a space), not <code>docker-compose</code> .
n8n won't open in the browser	Port 5678 open? <code>docker compose ps</code> running? <code>docker compose logs</code> errors? Using <code>https://</code> ?
n8n dies after ~90 days	The certificate copies went stale. Add the renewal hook in section 11.1.

17 Cheat sheet

What	Value / command
SSH (Tabby)	Host: your IP · Port 22 · User <code>ubuntu</code> · Key: your <code>.pem</code>
n8n URL	<code>https://YOUR_IP.nip.io:5678</code>
Start / stop n8n	<code>docker compose up -d</code> / <code>docker compose down</code>
Update n8n	<code>docker compose pull && docker compose up -d</code>
Logs	<code>docker compose logs -f</code>
Free up disk	<code>docker system prune -af</code> (removes unused images)
Open ports	<code>sudo ss -tlnp</code>
Reboot	<code>sudo reboot</code>

Your server is live. Now build something on it.

More practical AI builds every week on **Grounded [AI] Lab**: YouTube, Instagram and Facebook [@groundedailab](#). Official AWS free-tier page: aws.amazon.com/free