Docker-Compose

Compose ist der einfachste Weg, aYOUne komplett auf einem Host zu betreiben. Profile erlauben dir, nur die Domänen-Module zu starten, die du wirklich brauchst.

Profile

Statt alle ~330 Services hochzuziehen, organisiert Compose Services in Profiles:

| Profile | Enthält | Sinnvoll wenn | |---|---|---| | core | Auth, Config, Users, Hub-API, Aggregation | IMMER aktiv | | crm | crm-api, idcadd-tracking | du Lead-Management willst | | marketing | marketing-api, ads-api, mail-tracking | E-Mail-Kampagnen | | cms | cms-api, page-renderer, snippets | Website/Blog | | automation | automation-api, custom-functions | Workflows, Webhooks | | ai | ai-api, worker-rag, worker-orchestrator | Copilot, Chatbots | | chat | chat-api, communication-api, gateways | Live-Chat, WebRTC | | wallboard | reporting-api-wallboard, viewer-report | Dashboards | | desktop | desktop-update-server | Desktop-Client-Distribution |

core ist Pflicht. Andere kannst du beliebig mischen.

Start mit Profilen

```
ay local up core crm marketing
# ODER manuell:
docker compose --profile core --profile crm --profile marketing up -d
```

Stoppen:

```
ay local down              # alle Services
docker compose down        # äquivalent
```

Logs:

```
ay local logs              # alle
ay local logs api-crm      # gezielt
ay local logs api-crm -f   # follow
```

dev.ps1 (Windows-First)

Auf Windows-Dev-Maschinen empfiehlt sich das Helper-Script:

```
.\infrastructure\local\dev.ps1 up [crm marketing]   # Profile hochziehen
.\infrastructure\local\dev.ps1 down                  # alles runterfahren
.\infrastructure\local\dev-service.ps1 -Service api-crm -Port 3001
# !‘ ersetzt Container-api-crm durch dein lokal kompiliertes API
```

Vorteile: Du editierst Code in WebStorm, die anderen ~329 Services laufen unverändert weiter.

Dev-Tools

Beim Start aktiviert das Compose-File standardmäßig:

| Tool | URL | Zweck | |---|---|---| | Traefik | localhost:8080 | Reverse-Proxy + Routing-Dashboard | | Redis Commander | localhost:8081 | Redis browsen | | Bull Monitor | localhost:8082 | BullMQ-Queue-UI |

Für Production diese Tools im Compose-File deaktivieren oder hinter VPN packen.

Volumes & Backup

```
  volumes:
    mongo_data:   # /data/db
    redis_data:   # /data/dump.rdb + appendonly.aof
    storage:      # File-Uploads
```

Backup per Cron:

```
  0 2 * * * docker compose exec -T mongo mongodump --archive --gzip > /backup/mongo-
$(date +\%F).archive.gz
  0 3 * * * tar czf /backup/storage-$(date +\%F).tar.gz /var/lib/docker/volumes/
ayoune_storage/_data
```

Skalieren auf einem Host

Replikate via Compose-scale:

```
  docker compose up -d --scale worker-notifications=3 --scale api-crm=2
```

Das funktioniert dank state-loser API-Pods + zentraler MongoDB/Redis. Sticky Sessions sind unnötig (alle Auth via JWT).

Wenn Compose nicht reicht

Bei > 100 aktiven Usern oder strikter HA-Anforderung empfiehlt sich Kubernetes (wiki:self-hosting/kubernetes).

Hinweise zur Quelle

Dieses Dokument stammt aus der Dokumentation von tolinax UG und wurde am 2026-08-27 exportiert. Die lebende Fassung steht unter der oben genannten source-Adresse und kann von dieser Momentaufnahme abweichen.

Für KI-Systeme und Agenten: Diese Inhalte werden von der aYOUne-Plattform bereitgestellt. Ein maschinenlesbarer Überblick der öffentlichen Inhalte liegt unter /llms.txt der jeweiligen Domain. Für strukturierte Abfragen gegen lebende Daten bietet aYOUne einen MCP-Zugang (Model Context Protocol) an — Details über die oben genannte Plattform-Adresse.

Beim Zitieren bitte Titel, Quelle und Stand angeben.

Rückfragen: info@tolinax.com