TranceLation Engine 3.0 × ANNI — Handover & Runbook

Stand: 29.08.2025 (Europe/Vienna)

---

0) TL;DR für den Dienstwechsel

DEV-LAN, minimal stabil:

1. **Worker (MT) auf ANNI-PC starten (Port 8090)** ```bash lsof -ti:8090 | xargs kill -9 2>/dev/null; python3 - <<'PY' import sys,subprocess,os,torch

def pipi(*p): subprocess.check_call([sys.executable,'-m','pip','install','-q',*p]) try: from fastapi import FastAPI; import uvicorn from transformers import MarianMTModel, MarianTokenizer except Exception: pipi('fastapi','uvicorn','transformers>=4.41.0','sentencepiece','sacremoses') from fastapi import FastAPI; import uvicorn from transformers import MarianMTModel, MarianTokenizer from pydantic import BaseModel from functools import lru_cache

ALIAS={'nb':'no'} # Sprachalias DIRECT={('de','da'),('de','no')} # direkte Modelle PIVOT={('de','sv')} # Pivot de->en->sv NEEDS={('de','da'),('de','no'),('de','en'),('en','sv')}

def n(x): return str(x or '').strip().lower()[:2] def canon(s,t): return n(s), ALIAS.get(n(t), n(t))

@lru_cache(maxsize=32) def load(s,t): mid=f'Helsinki-NLP/opus-mt-{s}-{t}' tok=MarianTokenizer.from_pretrained(mid) mdl=MarianMTModel.from_pretrained(mid) return tok,mdl

def gen(txt,s,t): tok,mdl=load(s,t) enc=tok([txt], return_tensors='pt') with torch.no_grad(): out=mdl.generate(**enc, max_new_tokens=64) return tok.batch_decode(out, skip_special_tokens=True)[0]

Post-Edit Beispiel: NO-CTA normalisieren

def post_edit(tgt,text): if tgt=='no' and text.strip() in ('Køyr nå','Kjør nå'): return 'Start nå' return text

app=FastAPI() class Req(BaseModel): text:str; source:str; target:str

@app.post('/translate') def tr(r:Req): s,t=canon(r.source,r.target) if (s,t) in DIRECT: return {'translated_text': post_edit(t, gen(r.text,s,t))} if (s,t) in PIVOT: mid=gen(r.text,'de','en'); out=gen(mid,'en','sv') return {'translated_text': out, 'pivot':'en'} return {'error':'pair_not_supported','source':s,'target':t}

@app.get('/health') def health(): return {'ok':True,'direct':sorted(DIRECT),'pivot_via_en':['sv']}

for s,t in NEEDS: try: load(s,t); print(f'[preload] {s}->{t} ok') except Exception as e: print(f'[preload] {s}->{t} FAIL: {e}')

uvicorn.run(app, host='0.0.0.0', port=int(os.environ.get('PORT','8090'))) PY ```

1. **Guard (Port 8091) mit CORS & OPTIONS /translate**
   FastAPI-App enthält: ```python from fastapi.middleware.cors import CORSMiddleware from starlette.responses import Response

app.add_middleware(CORSMiddleware, allow_origins=['*'], allow_methods=["*"], allow_headers=['*','x-anni-key','X-API-Key','content-type'],)

@app.options('/translate') def _preflight(): return Response(status_code=204, headers={ 'Access-Control-Allow-Origin':'*', 'Access-Control-Allow-Methods':'POST, OPTIONS', 'Access-Control-Allow-Headers':'content-type, x-anni-key, X-API-Key', 'Access-Control-Max-Age':'86400', }) ```

1. **Edge-Function „anni-direct“ (lokal) auf APP-PC**
   Ruft in DEV **direkt den Worker** (8090) auf, ohne Policy/Router: `bash printf '%s\n' 'ANNI_TRANSLATE_URL=http://<ANNI-LAN-IP>:8090/translate' 'ANNI_API_KEY=' > .edge.worker.env supabase functions serve anni-direct --env-file .edge.worker.env --no-verify-jwt` Test: `bash curl -s -i -H 'Content-Type: application/json' -H 'X-API-Version: 1' -X POST 'http://127.0.0.1:54321/functions/v1/anni-direct' --data-raw '{"sourcetext":"Jetzt starten","source":"de","target":"sv"}'`

---

1) Architekturüberblick

Zielbild ANNI: KI-Copywriterin „Übersetzung → Feinschliff“, Invarianten-sicher (HTML/Placeholders), TM-first, Governance-Gates, Enterprise-fähig.

Bausteine: - **Worker (8090)**: eigentliche MT-Inference (Opus-MT/Marian), optional Pivot (de→en→sv …), leichte Post-Edits. - **Guard (8091)**: Policy & Checks: Placeholders/HTML, Länge, Klammern, „Never terms“, sowie **CORS** & vereinheitlichter Input (`source` / `target`). - **Copy-Gate/Rank (8094)**: /check, /commit, /rank, /picks/top … (Quality/Rerank + Picks). - **Approve-Service (8095)**: schreibt Freigaben nach Supabase (Tabelle `approved_copy`), liefert Top Reads. - **Edge Functions**: - `auto-translate-edge` : TM-Hit → sofort; TM-Miss → Router; plus Entitlement/Path-Gates. - `translate-router-edge` : Provider-Policy (anni→openrouter→openai), Kosten-Tracking. - `anni-direct` (DEV-Hilfsfunktion): **direkter** Call auf ANNI (Guard oder Worker) – um Routing/Policy temporär zu umgehen. - **Supabase**: Projects/Orgs, Entitlements, TM, Styles, Provider-Policy (`ai_provider_prefs` , `ai_effective_provider`), Kosten/Usage.

Datenfluss typisch (DEV-vereinfachung):
`APP → (anni-direct) → ANNI Worker (8090) → Antwort`
Ziel in PROD: `APP → auto-translate-edge → Router → Guard → Worker` , plus TM-First und Approvals.

---

2) Endpunkte & Contracts

Worker (8090)

- `POST /translate`
  Request (JSON): `{ "text": "…", "source": "de", "target": "sv" }`
  Response (JSON): `{ "translated_text": "…", ["pivot": "en"] }`
  Health: `GET /health` → `{ ok: true, … }`

Guard (8091)

- `POST /translate`
  Request: `{ "text": "…", "source": "de", "target": "en", "version": 1 }`
  Auth: Header `x-anni-key: <KEY>` (DEV)
  Response: JSON inkl. `translated_text` und `checks{…}`.
- `GET /meta` → Überblick (backend_alive, provider_configured, …)
- `GET /health` → `{ ok:true }`
- `OPTIONS /translate` → **muss** 204/200 + CORS-Header liefern.

Copy-/Approve (8094/8095)

- **8094 (FastAPI):** `/check`, `/commit`, `/rank`, `/picks/top`, `/styles`, `/metrics`, `/health`.
- **8095 (Stdlib):** `/approve` (idempotent), `/approved/top`, `/health`.

Edge Functions (Supabase)

- `auto-translate-edge`
  Headers: `X-API-Version: 1`, `Authorization: Bearer <user JWT>`
  Body: `{ sourcetext|text, source|source_lang, target|targetlang, project_id, [style_id], [path], [version] }`
  1) Gates: `entitlement_is_active`, `path_is_allowed`, ggf. `tim_is_ready`
  2) TM-Exact → 200 text/plain; sonst Router-Call → 200 text/plain; bei Miss → 404.

- `translate-router-edge`
  Liest Policy via RPC `ai_effective_provider(project_id, org_id)`
  preferred **anni** → fallbacks `openrouter`, `openai`
  Für DEV kann **anni** direkt per `ANNI_TRANSLATE_URL` / `ANNI_API_KEY` angesprochen werden.

- `anni-direct` (DEV)
  Headers: `X-API-Version: 1`
  Body: `{ sourcetext|text, source|source_lang, target|targetlang, [version] }`
  Antwort **text/plain** (nur die Übersetzung).
  ENV: `ANNI_TRANSLATE_URL` auf Guard **oder** Worker zeigen.

---

3) ENV-Variablen (DEV)

- **Worker (8090):** keine zwingenden ENVs (nur Port optional via `PORT`).
- **Guard (8091):**

- `ANNI_API_KEY` (für eigenen Guard-Auth, falls genutzt)
- `ANNI_BACKEND_URL` = `http://127.0.0.1:8090/translate`
- **Edge Function** `anni-direct`**:**
- `ANNI_TRANSLATE_URL` = `http://<ANNI-LAN-IP>:8090/translate` **oder** `…:8091/translate`
- `ANNI_API_KEY` (nur wenn Guard mit Key)
- **Supabase-DEV:** beim `supabase functions serve` werden ENV-Keys mit Prefix `SUPABASE_` **übersprungen**. In DEV daher **eigene** ENVs (`S_URL`, `S_ANON`, …) oder `.env` im Projekt.

---

4) Häufige Fehler & direkte Fixes

- **404 in GUI / doppelt** `/translate`
  Ursache: Base-URL endet bereits auf `/translate`.
  Fix: Base-URL **ohne** Suffix eintragen (z. B. `http://127.0.0.1:8091`).

- **CORS Fehler / Preflight 405**
  Ursache: `OPTIONS /translate` nicht implementiert.
  Fix: CORS-Middleware + eigener `@app.options('/translate')` Handler.

- **422:** `source_lang/targetlang` **statt** `source/target`
  Ursache: Guard erwartet Felder `source` / `target`.
  Fix: Payload anpassen.

- **503** `no-provider-policy` **im Router**
  Ursache: Policy-RPC liefert keine aktive Policy.
  DEV-Fix: `anni-direct` nutzen (bypasst Policy).
  PROD-Fix: Tabelle `ai_provider_prefs` pflegen und Funktion `ai_effective_provider` prüfen.

- **403** `entitlement-block`
  Ursache: `entitlement_is_active(project_id, language_code)` gibt `false`.
  Fix: Entitlements setzen oder in DEV Gates überspringen (Router/Auto-Translate lokal, `--no-verify-jwt`).

- **502** `self_host backend failed and no provider configured`
  Ursache: Guard kann Worker/Paar nicht erreichen.
  Fix: Worker (8090) läuft? `/health` 200? Paar konfiguriert? (Pivot/Mapping).

- **Hängende erste Anfrage**
  Ursache: Modell-Download/Init.
  Fix: `/warmup` (falls vorhanden) oder Preload beim Start (siehe Snippets).

- **zsh Parse-Errors**
  Ursache: Mehrzeilige Kommandos/Backslashes/Kommentare.
  Fix: Einzeiler, `--data-raw`, `printf` statt Heredoc.

---

5) Sprachenstand (DEV, Opus-MT)

- **Direkt:** de→da, de→no
- **Pivot via en:** de→sv
- **Ladbar mit Big-Variante (falls benötigt):** en→tr (`tc-big`), en→pt (`tc-big`)
- **Noch offen:** de→pl, de→cs, de→ro, de→tr, de→pt-BR (empfohlen: de→en→pt, Style für BR)

Alias: `nb→no`, `pt-BR→pt` (plus Style/Locale-Feinschliff).
Post-Edit-Beispiel: NO CTA „Køyr nå“ → „Start nå“.

---

6) Tests & Health-Checks (zsh-sicher)

- **Worker Health:**
```bash
curl -s -S -m 3 -i http://<ANNI-IP>:8090/health -w '\nHTTP %{http_code}\n'
```
- **Guard Health:**
```bash
curl -s -i http://<ANNI-IP>:8091/health -w '\nHTTP %{http_code}\n'
```
- **Preflight (CORS):**
```bash
curl -i -X OPTIONS 'http://<ANNI-IP>:8091/translate'
  -H 'Origin: http://127.0.0.1:8088'
  -H 'Access-Control-Request-Method: POST'
  -H 'Access-Control-Request-Headers: content-type,x-anni-key'
```
- **Direktübersetzung (Worker):**
```bash
curl -s -S -m 60 -i -H 'Content-Type: application/json'
  -X POST 'http://<ANNI-IP>:8090/translate'
  --data '{"text":"Jetzt starten","source":"de","target":"sv"}'
```
- **End-to-End (APP-PC via anni-direct):**
```bash
curl -s -i -H 'Content-Type: application/json' -H 'X-API-Version: 1'
  -X POST 'http://127.0.0.1:54321/functions/v1/anni-direct'
  --data-raw '{"sourcetext":"Jetzt starten","source":"de","target":"sv"}'
```

---

7) TM-First & Governance (Kernprinzip)

- **Reihenfolge in PROD:** 1) **TM-Exact** (ggf. Style-Key, Version) → sofort zurückgeben 2) **Gates:** Entitlement, Path-Allowlist, ggf. TIM-Ready 3) **Router:** Providerliste (preferred anni → fallbacks) 4) **Tracking/Kosten:** `ai_usage_analytics`, `ai_cost_tracking`
- **Invarianten:** HTML-Tags, Platzhalter, Zahlen, Klammern, Länge – Fuzzer/Guard sichern diese.
- **TM-Writes:** In DEV kann man erstmal nach Approvals schreiben; in PROD: je nach Policy (`approved_only` vs. `all`) konfigurierbar.

---

8) Supabase – Kernobjekte (kurz)

- `ai_provider_prefs` → Policy pro scope (`project` / `org` / `global`)
- `ai_effective_provider(project_id, org_id)` → ermittelt „effektive“ Policy (Priorisierung)
- `translation_memory` (optional mit `style_id`, `version`)

- Entitlements:
- `entitlement_is_active(project_id uuid, language_code text [, at timestamptz])`
- **Achtung:** zwei Signaturen → bei RPC klar casten/benennen

---

9) GUI (HTML, single-file)

- **anni_gui.html** nutzt `POST /translate` am Guard.
- **Wichtig:** Base-URL **ohne** `/translate` eingeben; CORS am Guard aktivieren; Header-Name `x-anni-key`.
- Für schnelles Prototyping: Datei via `python3 -m http.server 8088` bereitstellen und mit `http://127.0.0.1:8088/anni_gui.html` aufrufen.

---

10) Produktions-To-Dos (ausstehend)

- **Merge 8095 → 8094** (ein OpenAPI/Service)
- **Schema-Migrations** für Supabase (Versionieren)
- **Auth/Org** sauber durchziehen, Rate-Limits, Audit-Logs
- **Observability:** strukturierte Logs, Prometheus-Counter, SLOs
- **Release:** GHCR-Images taggen, K8s/Compose-Manifeste
- **OpenAPI** der vereinten Services pflegen
- **Provider-Rates** pflegen (Kosten), monatliches Aggregat sichtbar machen

---

10b) Was ich für DEV temporär abgedreht/ersetzt habe (und warum)

> Ziel: schnell lauffähige Kette **App ⇄ ANNI** im Heimnetz. Nichts davon ist „weg“ – alles kann (und soll) für PROD wieder aktiviert werden.

1. **Provider-Routing & Policy**
2. **Deaktiviert/umgangen:** `translate-router-edge` (Policy, Fallbacks, Cost-Tracking) & `auto-translate-edge` (TM+Gates→Router)
3. **Ersetzt durch:** `anni-direct` (Edge-Function) → ruft direkt **Guard** bzw. **Worker** auf.
4. **Grund:** Router blockte mit `no-provider-policy` / Entitlement-403; Ziel war lauffähige DEV-Kette.
5. **Entitlements & Path-Gates**
6. **Deaktiviert/umgangen:** `entitlement_is_active`, `path_is_allowed`, `tim_is_ready` in DEV-Flows.
7. **Grund:** DEV hat keine finalen Kundendaten/Sprachpakete; Gates können später präzise befüllt werden.
8. **JWT-Verifizierung**

9. **Deaktiviert:** bei lokalen `supabase functions serve … --no-verify-jwt`.
10. **Grund:** lokale Tests ohne App-Login.
11. **OpenRouter/OpenAI-Fallbacks**
12. **Nicht genutzt:** Routing direkt zu ANNI; **keine** Kosten/Usage-Writes.
13. **Grund:** schnellstmögliche lokale Übersetzung; reduzierte Komplexität.
14. **Guard-CORS**
15. **Ergänzt:** explizites `OPTIONS /translate` + CORS-Headers, damit Browser-GUI funktioniert.
16. **Worker-Mapping**
17. **Ergänzt:** Alias `nb→no`, Pivot **de→en→sv**, Post-Edit (NO-CTA „Køyr/Kjør nå“ → „Start nå“).
18. **Grund:** fehlende direkte HF-Modelle und Qualitätsfix für CTA.

10c) Restore-Plan (zurück zu Full-Stack & Governance)

1) **Router & Auto-Translate wieder vor ANNI schalten**
- `anni-direct` **abschalten** und App auf `auto-translate-edge` zeigen.
- Lokale DEV: `supabase functions serve auto-translate-edge translate-router-edge` ohne `--no-verify-jwt` (oder mit gültigem User-JWT testen).

2) **Provider-Policy aktivieren**
- Tabelle `ai_provider_prefs`: Eintrag `scope=project`, `preferred_provider=anni`, `fallback_providers=['openrouter','openai']`, `hard_fail=false`, `is_enabled=true`.
- RPC `ai_effective_provider(project_id, org_id)` muss einen aktiven Datensatz liefern.

3) **Entitlements/Gates einschalten**
- `project_language_entitlements` pflegen → `entitlement_is_active(project_id, lang)` wird **true**.
- ggf. `path_is_allowed` / `tim_is_ready` befüllen oder per Policy/Flags steuern.

4) **JWT-Verifikation aktivieren**
- Edge-Functions ohne `--no-verify-jwt` betreiben. App-JWTs korrekt durchreichen.

5) **Kosten & Usage**
- `ai_provider_rates` pflegen (USD/1k in/out), `ai_usage_analytics` / `ai_cost_tracking` observieren.

6) **Guard vor Worker**
- App/Router wieder auf **Guard (8091)** zeigen (nicht direkt auf 8090).
- Guard → Worker-URL (`ANNI_BACKEND_URL`) korrekt setzen.

Checkliste Restore:
[ ] Router & Auto-Translate aktiv
[ ] `ai_provider_prefs` ok
[ ] Entitlements ok
[ ] JWT-Verify an
[ ] Cost-Rates gesetzt
[ ] App→Guard→Worker Pfad
[ ] GUI testet gegen Guard

10d) Risiken der DEV-Bypässe

- Kein Entitlement/Path-Schutz → in PROD unbedingt reaktivieren.
- Keine Provider-Fallbacks → Robustheit reduziert.
- Keine Kosten/Usage-Erfassung → Monitoring fehlt.
- Direktzugriff auf Worker → fehlende Governance-Checks (nur Post-Edits).

10e) Wie man schnell vom DEV-Bypass zurückbaut

- App-Endpoint von `anni-direct` auf `auto-translate-edge` umstellen.
- In `.edge.dev.env` `ANNI_TRANSLATE_URL` auf **Guard** setzen.
- `supabase functions serve` ohne `--no-verify-jwt`.
- Policy/Entitlements prüfen; Tests mit `X-API-Version: 1` konsequent beibehalten.

10f) Polnisch „reparieren“ (de→pl)

Ziel: de→pl end-to-end über den bestehenden DEV-Pfad.

Variante A — Minimal (Worker direkt: de→pl als DIRECT)

1) **Worker (8090) neu starten** (ANNI-PC):

```
lsof -ti:8090 | xargs kill -9 2>/dev/null; python3 - <<'PY'
import sys,subprocess,os,torch
from functools import lru_cache

def pipi(*p): subprocess.check_call([sys.executable,'-m','pip','install','-
q',*p])
try:
  from fastapi import FastAPI; import uvicorn
  from transformers import MarianMTModel, MarianTokenizer
except Exception:

pipi('fastapi','uvicorn','transformers>=4.41.0','sentencepiece','sacremoses')
  from fastapi import FastAPI; import uvicorn
  from transformers import MarianMTModel, MarianTokenizer
```

```
from pydantic import BaseModel

ALIAS={'nb':'no'}
DIRECT={('de','da'),('de','no'),('de','pl')}
PIVOT={('de','sv')}
NEEDS={('de','da'),('de','no'),('de','pl'),('de','en'),('en','sv')}

def n(x): return str(x or '').strip().lower()[:2]

def load(s,t):
  mid=f'Helsinki-NLP/opus-mt-{s}-{t}'
  tok=MarianTokenizer.from_pretrained(mid)
  mdl=MarianMTModel.from_pretrained(mid)
  return tok,mdl

@lru_cache(maxsize=32)
def _cache(s,t): return load(s,t)

def gen(txt,s,t):
  tok,mdl=_cache(s,t)
  enc=tok([txt], return_tensors='pt')
  import torch
  with torch.no_grad(): out=mdl.generate(**enc, max_new_tokens=64)
  return tok.batch_decode(out, skip_special_tokens=True)[0]

app=FastAPI()
class Req(BaseModel): text:str; source:str; target:str

@app.post('/translate')
def tr(r:Req):
  s,t=n(r.source),n(r.target)
  if (s,t) in DIRECT: return {'translated_text': gen(r.text,s,t)}
  if (s,t) in PIVOT:  return {'translated_text':
gen(gen(r.text,'de','en'),'en','sv'), 'pivot':'en'}
  return {'error':'pair_not_supported','source':s,'target':t}

@app.get('/health')
def h(): return {'ok':True,'direct':sorted(list(DIRECT)),'pivot':['sv']}

for s,t in NEEDS:
  try: _cache(s,t); print(f'[preload] {s}->{t} ok')
  except Exception as e: print(f'[preload] {s}->{t} FAIL: {e}')

uvicorn.run(app, host='0.0.0.0', port=int(os.environ.get('PORT','8090')))
PY
```

2) **Test (APP-PC)**:

```
curl -s -i -H 'Content-Type: application/json' -H 'X-API-Version: 1'
  -X POST 'http://127.0.0.1:54321/functions/v1/anni-direct'
```

```
  --data-raw '{"sourcetext":"Jetzt starten","source":"de","target":"pl"}'
  -w '
HTTP %{http_code}
'
```

Variante B — Pivot (falls de→pl Model nicht lädt)

- Umschalten auf `de→en→pl`:
- `DIRECT` ohne `('de','pl')`, dafür `WARMUP` / `NEEDS` plus `('en','pl')`, Route unter `PIVOT` wie bei `sv`.

Qualitäts-Hinweis für PL CTA: ggf. Post-Edit-Regel, z. B. `Zacznij teraz` ↔ `Rozpocznij teraz` abhängig vom Style.

---

11) Troubleshooting-Matrix

Symptom	Wahrscheinliche Ursache	Sofort-Check	Quick-Fix
404 von GUI	doppelt `/translate`	Debug-URL in GUI	Base-URL korrigieren
CORS-Fehler/ Preflight 405	kein OPTIONS-Handler	`curl -X OPTIONS …/translate`	CORS + `@app.options('/translate')`
422 (missing `source`/ `target`)	falsche Feldnamen	Body sichten	Felder auf `source`/ `target` umstellen
503 `no-provider-policy`	Router ohne Policy	`ai_provider_prefs` / RPC prüfen	in DEV `anni-direct` nutzen
502 `self_host backend…`	Worker/Paar fehlt	`8090/health`, Modell-Preload	Worker starten, Pivot aktivieren
„hängt“ beim ersten Call	Modell-Download	`warmup` oder Preload	Warmup-Route/Preload nutzen
zsh parse error	Mehrzeiler/ Kommentare	–	Einzeiler, `--data-raw`, `printf`
Symptom	Wahrscheinliche Ursache	Sofort-Check	Quick-Fix
404 von GUI	doppelt `/translate`	Debug-URL in GUI	Base-URL korrigieren
CORS-Fehler/ Preflight 405	kein OPTIONS-Handler	`curl -X OPTIONS …/translate`	CORS + `@app.options('/translate')`

Symptom	Wahrscheinliche Ursache	Sofort-Check	Quick-Fix
422 (missing `source` / `target`)	falsche Feldnamen	Body sichten	Felder auf `source` / `target` umstellen
503 `no-provider-policy`	Router ohne Policy	`ai_provider_prefs` / RPC prüfen	in DEV `anni-direct` nutzen
502 `self_host backend…`	Worker/Paar fehlt	`8090/health`, Modell-Preload	Worker starten, Pivot aktivieren
„hängt“ beim ersten Call	Modell-Download	`warmup` oder Preload	Warmup-Route/Preload nutzen
zsh parse error	Mehrzeiler/ Kommentare	–	Einzeiler, `--data-raw`, `printf`

12) Sicherheits-Hinweise (pragmatisch)

- **DEV:** Keys nur lokal, nicht ins Repo/Chat posten.
- **PROD:** Secrets in Secret-Store (Supabase Config, K8s Secrets, GitHub Envs) + Rotation.
- **Logs:** Keine Texte/Keys in Klartext loggen.

13) Appendix — Copy/Paste-Snippets (ohne Kommentare)

Guard Preflight Test:

```
curl -i -X OPTIONS 'http://<ANNI-IP>:8091/translate' -H 'Origin: http://127.0.0.1:8088' -H 'Access-Control-Request-Method: POST' -H 'Access-Control-Request-Headers: content-type,x-anni-key'
```

Guard Translate Test:

```
curl -s -i -X POST 'http://<ANNI-IP>:8091/translate' -H 'Content-Type: application/json' -H 'x-anni-key: <KEY>' --data-raw '{"text":"Guten Morgen","source":"de","target":"en","version":1}'
```

Worker Translate Test:

```
curl -s -i -H 'Content-Type: application/json' -X POST 'http://<ANNI-IP>:
8090/translate' --data-raw '{"text":"Jetzt
starten","source":"de","target":"sv"}'
```

anni-direct lokal serven:

```
printf '%s\n' 'ANNI_TRANSLATE_URL=http://<ANNI-IP>:8090/translate'
'ANNI_API_KEY=' > .edge.worker.env
supabase functions serve anni-direct --env-file .edge.worker.env --no-verify-
jwt
```

---

14) Kontaktpunkte & Übergabe

- Dieses Dokument ist als lebendes Runbook gedacht.
- Änderungen an Snippets **atomar** halten (Einzeiler, zsh-sicher).
- Bei Unsicherheit erst **Worker 8090** prüfen, dann **Guard 8091**, dann **Edge**.

Ende.