TranceSpell® — Spezifikation v1.0

Komponente: Eigenständiger Service (on-prem)
Ziel: Rechtschreib-Erkennung (*Detection-Only*) in Quelltexten, invariant-sicher, sprach-agnostisch integrierbar
Port (Default): 8096
Status: v1.0 (MVP, produktionsreif für Detection-Only)

---

1) Zweck & Positionierung

TranceSpell® erkennt Rechtschreibfehler in **Quelltexten**, ohne Änderungen am Text vorzunehmen. Der Service läuft getrennt von Guard/TranceCreate, ist jedoch kompatibel und kann **früh in der Kette** (vor Transcreation) ausgeführt werden.
Nicht-Ziele v1.0: keine Grammatik, keine Autokorrektur, keine GUI-Integration.

Kernwerte - *Detection-Only:* Es werden **Issues** gemeldet; der Text bleibt unverändert. - *Invariant-sicher:* `{{placeholders}}`, `{tokens}`, HTML, URLs, Emojis, Zahlen bleiben unberührt (Masking). - *Sprach-agnostisch orchestriert:* Hunspell genutzt, wo verfügbar; ansonsten pyspellchecker; andere Sprachen „unsupported“ (kein Fehler).

---

2) Architekturüberblick

- **ts_server.py** — FastAPI App, Endpoints, Pydantic Models, CORS.
- **ts_core.py** — Masking, Tokenisierung, Engine-Auswahl, Prüf-Logik.
- **config/trancespell.json** — Hunspell-Pfade, Aliase, Limits.

Engines 1. **Hunspell** (voll): wenn `.aff` / `.dic` vorhanden
2. **pyspellchecker** (basic): fallback für bestimmte westliche Sprachen
3. **unsupported**: keine Engine → leere Issues + Hinweis

Auto-Discovery (optional v1.1, kompatibel): Scannt übliche Hunspell-Verzeichnisse + konfigurierte Pfade; `/languages` gruppiert `full/basic/unsupported`.

---

3) API-Spezifikation

GET `/health`

Response

```
{
  "ok": true,
  "ready": true,
  "engine": "hunspell|pyspell",
  "langs": ["de","en", "…"],
```

```
  "trace": {"full": 2, "basic": 6, "unsupported": 10}
}
```

GET `/languages`

Response

```
{
  "langs": {"full": ["de","en"], "basic": ["fr","es"], "unsupported":
["ja","th"]},
  "aliases": {"de-DE":"de","en-US":"en","iw":"he","in":"id","pt-BR":"pt","zh-
CN":"zh","zh-TW":"zh"},
  "paths": {"hunspell": ["/usr/share/hunspell","/usr/local/share/hunspell"]}
}
```

POST `/check`

Request

```
{
  "lang": "de-DE",
  "text": "<button>Jetz registrieren</button> 🙂 {{COUNT}}"
}
```

Response

```
{
  "issues": [
    {"start": 8, "end": 12, "token": "Jetz", "suggestions": ["Jetzt"],
"rule": "spell"}
  ],
  "masked": true,
  "trace": {"lang": "de", "engine": "hunspell|pyspell", "checked_tokens": 2,
"issues": 1, "elapsed_ms": 12}
}
```

Hinweise - `start` / `end` sind Offsets im **Originaltext** (nach Masking korrekt remappt). - Bei *unsupported* Sprachen: `issues: []`, `trace.note: "lang_not_supported_for_spell"`.

---

4) Masking & Invarianten (kompatibel zu Guard/TC)

Geschützte Spans (werden nicht geprüft & nicht gezählt): - `{{…}}` (double-brace placeholders) - `{token}` (single-brace tokens) - **HTML-Tags** `<…>` (Inhalt darf geprüft werden; Tags selbst sind geschützt) - **URLs**: `https?://…` - **Emojis** (Unicode Emoji präsentiert als ein Graphem) - **Zahlen** & Zahlenbereiche (z. B. `1990–2014`, `1 234,56`)

Tokenisierung: nur außerhalb geschützter Spans.

---

5) Sprach-Handling & Engine-Auswahl

- **Normalisierung:** `lang_normalize("de-DE") → "de"`; Aliase z. B. `iw→he`, `in→id`, `pt-BR→pt`, `zh-*→zh`.
- **Priorität:** Hunspell (falls `aff+dic`) → pyspellchecker → unsupported.
- **Caching:** Engines werden je Sprache wiederverwendet (Performance).

---

6) Konfiguration (`config/trancespell.json`)

```
{
  "dictionaries": {
    "de": {"aff": "/usr/local/share/hunspell/de_DE.aff", "dic": "/usr/local/share/hunspell/de_DE.dic"},
    "en": {"aff": "/usr/local/share/hunspell/en_US.aff", "dic": "/usr/local/share/hunspell/en_US.dic"}
  },
  "aliases": {"de-DE": "de", "en-US": "en", "iw": "he", "in": "id", "pt-BR": "pt", "zh-CN": "zh", "zh-TW": "zh"},
  "max_suggestions": 5,
  "timeout_ms": 8000,
  "hunspell_paths": ["/usr/share/hunspell", "/usr/local/share/hunspell"]
}
```

Empfehlung: Wörterbücher systemweit installieren; TS findet sie per Auto-Discovery.

---

7) Qualität, Telemetrie & Grenzen

- **Qualität:** Korrekte Offsets, keine Maskenverletzungen, stabile Engine-Wahl je Sprache.
- **Telemetrie (`trace`):** Engine, Sprache, `checked_tokens`, `issues`, Dauer.
- **Grenzen:** CJK/Thai ohne Wortgrenzen → aktuell `unsupported` (Detection nicht zuverlässig ohne Segmentierung).

---

8) Tests & Abnahme (Smoke)

A) DE Detection — `<button>Jetz registrieren</button> 🙂 {{COUNT}}`
Erwartet: 1 Issue „Jetz“ → „Jetzt“, Masken ok, Offsets korrekt.

B) Unsupported — `lang=ja`
Erwartet: `issues: []`, `trace.note` gesetzt.

C) Invarianten — Text mit `{app}`, `<a href="https://x.y">`, Zahlenbereich
Erwartet: keine Issues innerhalb Masken; keine Textänderung.

Exit: 0 bei PASS.

---

9) Betrieb & Monitoring

- **Port:** 8096 (empfohlen).
- **Readiness:** `/health` (zählt Full/Basic/Unsupported).
- **Konfiguration:** beim Start geladen; Pfade für Hunspell konfigurierbar.
- **Logging:** Requests, Engine-Wahl, Fehlerfälle; Metriken in `trace` der Responses.

---

10) Integrations-Roadmap

- **Guard** (optional): Vor-Check auf Quelltexte; Issues als Anreicherung in Metadaten zurückgeben.
- **TranceCreate** (optional): Pre-Stage „SpellCheck“, nur Reporting; Kunde entscheidet UI-seitig, wie Issues angezeigt werden.
- **GUI** (optional): Tab „SpellCheck“, Filter nach Sprache/Typ, CSV/JSON-Export.

---

11) Versionierung & Kompatibilität

- **Version:** TranceSpell® v1.0
- **Abwärtskompatibel:** Detection-Only; spätere Erweiterungen (Auto-Discovery, zusätzliche Engines) ohne Breaking Changes geplant.

---

12) Markenhinweis

TranceSpell® ist eine Produktbezeichnung von TranceLate.it FlexCo.