KI-gestütztes Lebenslauf-Parsing

Ein Prompt-Engineering-Workflow für Recruiting: vom Lebenslauf zu strukturierten Kandidatendaten

Portfolio-Projekt · Prompt Engineering & Datenextraktion · Erstellt mit Claude / GPT-kompatiblen LLMs

Kurzfassung: Dieser Workflow liest unstrukturierte Lebensläufe — egal in welchem Format — und gibt einheitliche, maschinenlesbare Kandidatendaten als JSON zurück: Kontaktdaten, Berufserfahrung, Ausbildung, Skills und Sprachen. Ein eingebautes Bonus-Feature erlaubt zusätzlich die *anonymisierte* Ausgabe für ein faires, vorurteilsfreies Vorscreening.

1. Das Problem

Recruiter und Personaldienstleister erhalten täglich Lebensläufe in völlig unterschiedlichen Layouts — als PDF, Word, ein- oder mehrspaltig, in verschiedenen Sprachen. Die relevanten Daten von Hand in ein Bewerbermanagement-System (ATS) zu übertragen, kostet 5–10 Minuten pro Bewerbung und führt zu uneinheitlichen Datensätzen, die sich kaum durchsuchen oder vergleichen lassen.

Ziel: Aus beliebigem Lebenslauf-Text automatisch ein einheitliches Kandidatenprofil erzeugen, das sich filtern, sortieren und mit Stellenanforderungen abgleichen lässt — robust gegenüber den vielen Formaten, in denen Menschen ihre Lebensläufe schreiben.

VORHER

- 5–10 Min. manuelles Erfassen pro Lebenslauf
- Uneinheitliche, schwer durchsuchbare Datensätze
- Datumsangaben und Skills in jedem CV anders

NACHHER

- Sekunden pro Lebenslauf, formatunabhängig
- Einheitliches, filterbares Kandidatenprofil
- Optional anonymisiert für faires Screening

2. Der Lösungsansatz

Wie schon beim Rechnungs-Workflow sind Eingabe und Speicherung technische Hülle — der eigentliche Wert liegt im Extraktions-Prompt. Die Besonderheit hier: Lebensläufe sind deutlich freier und unstrukturierter als Rechnungen. Es gibt keine festen Felder, keine garantierte Reihenfolge. Der Prompt muss also Bedeutung erkennen, nicht nur Positionen ablesen.

Schritt	Aufgabe	Werkzeug
1 · Eingabe	Lebenslauf-Text bereitstellen (aus PDF/Word extrahiert oder per OCR)	Textextraktion / OCR
2 · Extraktion	Der LLM interpretiert den Text und gibt strukturiertes JSON nach festem Schema zurück	**Engineered Prompt**
3 · Ausgabe	JSON ins ATS / in eine durchsuchbare Datenbank übernehmen	JSON-Validierung

3. Das Kernstück: Der Extraktions-Prompt

Jede Regel im Prompt löst ein konkretes Problem, das in echten Lebensläufen auftritt — von „heute" als Enddatum bis zu doppelt genannten Skills.

```
Du bist ein präziser Assistent zum Parsen von Lebensläufen.
Deine Aufgabe: Extrahiere die unten definierten Felder aus dem
bereitgestellten Lebenslauf-Text und gib AUSSCHLIESSLICH ein gültiges
JSON-Objekt zurück – keine Erklärungen, kein Markdown, kein Fließtext.

REGELN:
1. Halte dich exakt an Schema und Feldnamen.
2. Steht ein Wert nicht eindeutig im Text, setze ihn auf null.
   Leite keine Werte ab und erfinde nichts (auch keine Berufsjahre).
3. Daten im Format JJJJ-MM. Ist nur ein Jahr genannt, nutze JJJJ.
   Für ein laufendes Verhältnis ("heute", "aktuell", "present")
   setze "end_date" auf "present".
4. "skills": eine bereinigte, duplikatfreie Liste einzelner
   Fähigkeiten. Fasse keine Sätze, sondern einzelne Begriffe.
5. "work_experience" und "education": je ein Eintrag pro Station,
   in zeitlich absteigender Reihenfolge (neueste zuerst).
6. "total_experience_years": nur ausfüllen, wenn es im Text klar
   hervorgeht oder eindeutig aus den Zeiträumen berechenbar ist;
   sonst null. Runde auf ganze Jahre.
7. "summary": eine neutrale Zusammenfassung in 1–2 Sätzen, falls
   ein Profil/Über-mich-Abschnitt vorhanden ist; sonst null.
8. Gib keine Felder aus, die nicht im Schema stehen.

SCHEMA:
{
  "full_name": string | null,
  "email": string | null,
  "phone": string | null,
  "location": string | null,
  "summary": string | null,
  "total_experience_years": number | null,
  "skills": [ string ],
  "work_experience": [
    { "company": string, "title": string,
      "start_date": "JJJJ-MM" | null,
      "end_date": "JJJJ-MM" | "present" | null,
      "description": string | null }
  ],
  "education": [
    { "institution": string, "degree": string | null,
      "field": string | null,
      "start_year": number | null, "end_year": number | null }
  ],
  "languages": [ { "language": string, "level": string | null } ]
}

LEBENSLAUF-TEXT:
"""
```

```
{hier wird der extrahierte Lebenslauf-Text eingefügt}
"""
```

Warum dieser Prompt so gebaut ist

Regel	Welches reale Problem sie löst
„present" als Enddatum	Lebensläufe schreiben „heute", „bis jetzt", „current". Ein einheitlicher Wert macht laufende Anstellungen maschinell erkennbar.
Skills als bereinigte Liste	Erst eine saubere, duplikatfreie Liste erlaubt späteres Filtern nach „Python", „Excel" usw.
Berufsjahre nur wenn belegbar	Verhindert die häufigste Halluzination: eine erfundene Jahreszahl, die das Ranking verfälscht.
Neueste Station zuerst	Sorgt für konsistente Reihenfolge — egal wie der Bewerber sortiert hat.
Nur JSON, nichts erfinden	Macht die Ausgabe direkt ins ATS übernehmbar und vermeidet Fehlentscheidungen durch erfundene Daten.

4. Worked Example

EINGABE (LEBENSLAUF-TEXT, GEKÜRZT)

```
Anna Beispiel
Musterweg 5, 44789 Bochum · anna.beispiel@email.de · 0151 2345678

Profil
Frontend-Entwicklerin mit Fokus auf barrierefreie Web-Apps.

Berufserfahrung
03/2023 – heute   Webentwicklerin, Studio Nord GmbH, Bochum
   Entwicklung von React-Komponenten und Pflege des Design-Systems.
09/2020 – 02/2023 Werkstudentin Frontend, NetCraft AG, Essen

Ausbildung
2019 – 2022  B.Sc. Medieninformatik, Hochschule Bochum

Kenntnisse: HTML, CSS, JavaScript, React, React, Figma, Barrierefreiheit
Sprachen: Deutsch (Muttersprache), Englisch (fließend)
```

AUSGABE (VOM WORKFLOW ERZEUGTES JSON)

```json
{
  "full_name": "Anna Beispiel",
  "email": "anna.beispiel@email.de",
  "phone": "0151 2345678",
  "location": "Bochum",
  "summary": "Frontend-Entwicklerin mit Fokus auf barrierefreie Web-Apps.",
  "total_experience_years": 5,
  "skills": ["HTML", "CSS", "JavaScript", "React", "Figma", "Barrierefreiheit"],
  "work_experience": [
```

```
    { "company": "Studio Nord GmbH", "title": "Webentwicklerin",
      "start_date": "2023-03", "end_date": "present",
      "description": "Entwicklung von React-Komponenten und Pflege des Design-
Systems." },
    { "company": "NetCraft AG", "title": "Werkstudentin Frontend",
      "start_date": "2020-09", "end_date": "2023-02", "description": null }
  ],
  "education": [
    { "institution": "Hochschule Bochum", "degree": "B.Sc.",
      "field": "Medieninformatik", "start_year": 2019, "end_year": 2022 }
  ],
  "languages": [
    { "language": "Deutsch", "level": "Muttersprache" },
    { "language": "Englisch", "level": "fließend" }
  ]
}
```

Beachte: „React" war im Text doppelt genannt — in der Ausgabe erscheint es nur einmal. „heute" wurde zu `"present"`, und alle Daten sind einheitlich formatiert.

5. Umgang mit Sonderfällen

Sonderfall	Verhalten des Workflows
Lücke im Lebenslauf	Wird nicht „aufgefüllt" — die Stationen bleiben wie angegeben, ohne erfundene Zeiträume.
Nur Jahreszahl (kein Monat)	Datum wird als `JJJJ` ausgegeben statt geraten.
Kein Profilabschnitt	`summary` wird `null` statt einer erfundenen Beschreibung.
Mehrsprachiger Lebenslauf	Inhalte werden im Originalwortlaut extrahiert; das Schema bleibt gleich.
Doppelte Skills	Werden zu einer bereinigten Liste zusammengeführt.

6. Bonus-Feature: Anonymisiertes Screening

Ein echter Mehrwert für faires Recruiting: Durch eine einzige zusätzliche Anweisung lässt sich derselbe Prompt so umstellen, dass er **persönliche Identitätsmerkmale weglässt** — Name, Foto, Adresse, Geburtsdatum, Geschlecht. So können Recruiter Kandidaten zunächst nur nach Qualifikation vergleichen und unbewusste Vorurteile reduzieren.

ZUSÄTZLICHE REGEL FÜR DEN ANONYMISIERTEN MODUS

```
ANONYMISIERUNG: Setze "full_name", "email", "phone" und "location"
immer auf null. Entferne aus allen anderen Feldern jeden Hinweis auf
Name, Geschlecht, Alter, Herkunft oder Adresse. Bewahre dabei alle
```

```
fachlich relevanten Informationen (Skills, Erfahrung, Ausbildung)
vollständig.
```

Das demonstriert ein zentrales Prinzip guten Prompt Engineerings: Ein gut strukturierter Basis-Prompt lässt sich mit minimalem Aufwand für verwandte Anwendungsfälle wiederverwenden, statt von Grund auf neu gebaut zu werden.

7. Eingesetzte Fähigkeiten

Prompt Engineering | Strukturierte Ausgaben (JSON) | Datenextraktion | Schema-Design | Datennormalisierung | Edge-Case-Handling | Wiederverwendbare Prompts | ChatGPT / Claude

8. Möglicher Ausbau

Der Workflow lässt sich erweitern um automatischen Abgleich gegen eine Stellenbeschreibung (Matching-Score), Batch-Verarbeitung ganzer Bewerbungseingänge sowie direkten Export ins ATS. Der Prompt bleibt das stabile Herzstück — er ist bewusst so entworfen, dass er mit unterschiedlichsten Lebenslauf-Formaten zurechtkommt und sich, wie das Anonymisierungs-Feature zeigt, leicht an neue Anforderungen anpassen lässt.