Prompt Pattern Pack (Engineer Edition)

Sofort nutzbare Muster für Claude oder Codex im Engineering-Alltag

2026-04-07

Dieses Pack hilft dir, aus KI-Experimenten einen wiederholbaren Workflow zu machen. Alle Muster sind auf reale Engineering-Tasks ausgerichtet.

Template 1: Component Scaffold (bounded)

Ziel

Neue Komponente mit klarem Scope erzeugen, ohne Architekturdrift.

Prompt-Template

```
Kontext:
- Repo: [repo]
- Datei: [zielpfad]
- Stack: [z.B. React + TypeScript]
- In Scope: [x, y]
- Out of Scope: [a, b]

Aufgabe:
Erstelle [Komponente] mit Props [liste].
Halte bestehende API-Verträge stabil.

Akzeptanz:
1) rendert [A]
2) verändert [B] bei Interaktion
3) enthält "Reset"-Pfad

Output:
1) Plan (max 5 Punkte)
2) Code
3) 1 Testfall
4) Risiko-Hinweis
```

Template 2: Refactor ohne Verhaltensänderung

Ziel

Lesbarkeit verbessern, ohne Produktverhalten zu ändern.

Prompt-Template

```
Lies [datei].
Refaktoriere nur innerhalb der markierten Funktion [name].

Grenzen:
- Keine neue Abhängigkeit
- Keine Signaturänderung
- Keine API-Änderung
- Diff maximal [x] Zeilen
```

```
Output:
1) Plan
2) Refactor-Vorschlag
3) Warum Verhalten gleich bleibt
4) Test-Hinweis
```

Template 3: Test-Edge-Cases

Ziel

Schwächen in Randfällen absichern.

Prompt-Template

```
Lies [komponente] und [testdatei].
Bestehende Tests sind zu oberflächlich.

Füge Tests hinzu für:
1) Null-/Empty-Fall
2) Sortier-/Filter-Randfall
3) Pagination-/Boundary-Fall

Vorgaben:
- Beschreibungen aus User-Sicht
- Bestehende Tests nicht umschreiben
- Nur neue Fälle ergänzen
```

Template 4: Kleine Hook-Extraktion

Ziel

Wiederverwendbare Logik isolieren, UI-Verhalten beibehalten.

Prompt-Template

```
Lies [datei].
Extrahiere [logik] in Hook [hook-name] unter [pfad].

Muss gleich bleiben:
- Props der Komponente
- sichtbares Verhalten im DOM
- Timing/Seiteneffekte

Output:
1) Hook-Code
2) angepasste Komponente
3) 1 Test für den Hook
4) Risiko-Notiz
```

Task-Routing-Regeln

- KI-first bei bekannten Mustern (Scaffold, kleine Refactors, Tests).
- Manual-first bei Architekturfragen, Multi-Modul-Änderungen, Incident-Hotfixes.
- Eskalation: Nach 2 fehlgeschlagenen Iterationen Aufgabe manuell lösen.

Review-Checkliste vor Commit

- [ ] Scope eingehalten (In/Out of Scope)
- [ ] Keine themenfremden Änderungen
- [ ] Akzeptanzkriterien erfüllt
- [ ] Tests vorhanden oder begründet
- [ ] Risikoquelle benannt