🏭 GhibliKitchen Production Prompt (DE & ZH)

Dieser Prompt erzeugt automatisch **vollständige Wochenpläne** – eine **deutsche** Datei und eine **chinesische** Datei – im identischen Design und Verhalten wie die Referenz **„Woche-4-2025-10-20“**.

- Ausgabedateien (immer beide Varianten):
- `src/plans/Woche-{{WEEK_NR}}-{{START_DATE}}.de.jsx`
- `src/plans/Woche-{{WEEK_NR}}-{{START_DATE}}.zh.jsx`
- Ziel: Jede Woche soll sich **optisch, strukturell und funktional 1:1 gleich** verhalten.
- Die Dateien sind **selbstständig rendernde React-Komponenten** (kein Mount im File), und verwenden unsere bestehenden Utils (`exportPDFById`, `exportHTMLById`, `buildEmbedCss`) und i18n-Helfer (`UI`, `pickText`, `pickList`).

🎯 Ziel & UI-Layout (fix)

- **Tabs:** „Kochbuch“ (A4 quer) und „Einkaufsliste“ (A4 hoch).
- **Top-Bar Buttons:** „PDF erzeugen“, „HTML exportieren“, „Drucken“. **Kein** „HTML öffnen“.
- Nach PDF-Erzeugung erscheint ein **Download-Link** unter dem jeweiligen Tab-Inhalt.
- Jede Rezeptseite ist **eine** Seite (1 Rezept = 1 Seite), linkes Panel ≤1/3 (span 4), Rezept rechts ≥2/3 (span 8).
- **Cover-Seite:** Zweispaltig mit Flex: links Info/Upload (cardPanelStyle), rechts **Wochenübersicht** (cardMainStyle). Die Wochenübersicht ist **Pflicht**.
- **Wochenübersicht:** 7 Blöcke (Mo–So), je Block 3 Kacheln (Frühstück/Mittag/Abend); je Kachel Titel, „🌾 KH …“ (aus `target`), bei Frühstück/Abend „💊“.
- **Wochentag-Überschrift:** pro Rezept **oberhalb des Rezepttitels im rechten Hauptteil** (nicht als eigener H-Tag), kleine Zeile.
- **DALL·E-Prompts** NICHT rendern (nur als String im Code).
- **Bilder-Uploads** (Cover + je Rezept) via FileReader, persistiert in `localStorage`.
- Über jedem Rezept: **„{Wochentag} – {Morgen|Mittag|Abend}“**.
- **Metformin-Reminder:** Frühstück/Abend ✅, Mittag ❌.

- Jedes Rezept enthält eine **Kurzbeschreibung** mit Ursprung + „inspiriert von …“.
- **Kurz-Story (neutral):** Direkt **unter** dem Rezepttitel im rechten Hauptteil, sachlich (Region/Anlass/Saison, z. B. „kommt aus …, beliebt im …“), **Schriftgröße: 12**. Kein übertriebener „Ghibli“-Stil.
- Wochenübersicht oben zeigt pro Tag drei Kacheln (F/M/A) mit Titel, Ziel („🌾 KH …“) und 💊-Icon, wenn Reminder aktiv ist.

🧱 Fixe Metadaten & Basisstruktur

In BEIDEN Dateien (DE & ZH) identisch – nur Texte sind übersetzt:

```
export const meta = {
  title: "Woche {{WEEK_NR}}",
  startDate: "{{START_DATE}}",
  id: "woche-{{WEEK_NR}}-{{START_DATE}}"
};
const FILE_BASE = "Woche {{WEEK_NR}} {{START_DATE}}";
```

UI-Titel:

- Hauptseite: `GhibliKitchen – Woche {{WEEK_NR}}`
- Liste: `GhibliKitchen – Einkaufsliste – Woche {{WEEK_NR}}`

Farben & Styles:

```
const COLORS = {
  pageBg:"#FAF7F1",
  text:"#111827",
```

```
  border:"rgba(0,0,0,.10)",
  panelBG70:"rgba(255,255,255,.70)",
  panelBG80:"rgba(255,255,255,.80)",
  white:"#FFFFFF",
  emerald:"#059669",
  amber:"#f59e0b",
  sky:"#0284c7",
  neutral:"#404040",
  indigo:"#4f46e5",
  btnShadow:"0 6px 20px rgba(0,0,0,.12)",
};
const cardPanelStyle = {
  background: COLORS.panelBG70,
  borderRadius: 18,
  padding: 20,
  boxShadow: COLORS.btnShadow,
  border: `1px solid ${COLORS.border}`,
};
const cardMainStyle = {
  background: COLORS.white,
  borderRadius: 18,
  padding: 22,
  boxShadow: COLORS.btnShadow,
  border: `1px solid ${COLORS.border}`,
};
```

DALL·E-Prompt-Header & Helper:

```
const PROMPT_HEADER =

  "Ultra-clean cookbook photo, soft daylight, top-down, pastel
background, visible steam, pregnancy-safe (no raw fish or raw egg),
mild Asian home cooking (JP/CN/KR), family-friendly";

const buildPrompt = (a, b) => `${a}\n${b}`;
```

🧰 i18n-Helfer (Verwendung)

- Importiere: `import { UI } from "../i18n-ui";`
- Importiere: `import { pickText, pickList } from "../i18n-data";`
- **Wichtig:** Alle Felder in `DATA` können **String** ODER `{ de, zh }` sein.
- `pickText(v, lang)` gibt korrekt DE/ ZH zurück (Fallback auf `de`).
- `pickList(v, lang)` akzeptiert `Array` ODER `{ de:[], zh:[] }`.

Gesundheits- & Küchenregeln

- Küchenmix: **CN/JP/KR dominant** (mind. 6/7 Tage), **max. 1** IT-Gericht.
- Pro Rezept (2 Personen): **60–90 g KH** gesamt; Protein-Hinweis optional (**20–40 g p. P.**).
- Diabetes (frühes Stadium; Metformin 2× täglich (früh und abend)): pro Mahlzeit (2 Pers.) **60–90 g KH gesamt** (≈30–45 g p. P.), ballaststoffbetont; **Protein 20–40 g p. P.** Metformin: reine Erinnerung „mit der Mahlzeit“ (kein Med-Rat).
- Garmethoden: **Dämpfen, Sieden, Schmoren**; wenig Öl; Zwiebel/Knoblauch sparsam & gut gegart; **Säure mild**; Algen/Jod **sparsam**.
- Schwangerschaft: **nichts Rohes**; alles **durchgaren** (Eier vollständig gestockt); quecksilberarme Fische (Lachs/Kabeljau/Seelachs/Wolfsbarsch); Hygiene; **Sojasauce natriumarm**; **Jod (Wakame/Kombu) sparsam**.

Gastritis:

- Nur bei explizit „gastritis-konform“ → **streng** (Schärfe/zu sauer/fettig meiden, schonend garen, wenig Öl, warm).
- Standard (**balanced**), wenn NICHT angefordert: mild würzen, nicht zu scharf; vorsichtiges Wok/Anbraten/Grillen mit wenig Öl erlaubt; milde Säure moderat; Zwiebel/Knoblauch maßvoll & gut gegart; Chili optional separat.
- **Non-Strict Checks:** „Gastritis“ wird **ohne „✓“** angegeben (nur erläuternder Text, z. B. „Gastritis – mild …“).
- Titel: **Deutsch + Originalname + Schriftzeichen**.

🗂️ Datenmodell (21 Rezepte)

- Genau **21 Rezepte**: 7 Tage × 3 (Frühstück `-f`, Mittag `-m`, Abend `-a`).
- Rezept-Objekt (DE & ZH strukturell identisch):

```
type Recipe = {

  id: "mo-f" | "mo-m" | "mo-a" | "di-f" | ... | "so-a";

  title: string | { de: string; zh?: string };

  desc: string | { de: string; zh?: string };

  story: string | { de: string; zh?: string };

  target: string | { de: string; zh?: string }; // z. B. "≈70 g KH
gesamt (2 P.) · Protein ≈20 g p. P."

  ingredients: string[] | { de: string[]; zh?: string[] }; // ≥ 5
Einträge

  steps: string[] | { de: string[]; zh?: string[] };       // ≥ 3
Einträge

  checks: string | { de: string; zh?: string };

  swaps: string | { de: string; zh?: string };

  side:  string | { de: string; zh?: string };

  remind: boolean; // Frühstück/Abend: true, Mittag: false

  prompt: string;  // buildPrompt(PROMPT_HEADER, "...")

}
```

Tagesreihenfolge:

```
const DAYS_ORDER = ["mo","di","mi","do","fr","sa","so"];

const DAY_NAME_DE = { mo:"Montag", di:"Dienstag", mi:"Mittwoch",
do:"Donnerstag", fr:"Freitag", sa:"Samstag", so:"Sonntag" };

const DAY_NAME_ZH = { mo:"周一", di:"周二", mi:"周三", do:"周四", fr:"周五
", sa:"周六", so:"周日" };

const groupByDay = (arr) => {

  const map = { mo:[],di:[],mi:[],do:[],fr:[],sa:[],so:[] };

  arr.forEach((r)=> map[r.id.split("-")[0]].push(r));

  Object.values(map).forEach((list)=>

    list.sort((a,b)=> ["f","m","a"].indexOf(a.id.split("-")[1]) -
["f","m","a"].indexOf(b.id.split("-")[1]))

  );

  return map;

};
```

📋 Rezeptkarte (Pflicht-Layout)

- Linkes Info-Panel:
- Upload (nur in UI sichtbar, `print:hidden`)
- (kleines) Bild
- Kurzbeschreibung, Ziel, Checks, Beilage
- **Reminder-Badge** („💊 Metformin…“) wenn `remind === true`
- Rechter Hauptbereich:
- Breadcrumb in **sky** (Tag + Meal)
- **Titel** (h2)
- **Story** (kurz)

- **Zutaten (2 Personen)** – Liste
- **Schritte** – geordnete Liste
- **Swaps** – Satz

Sichere Render-Guards (immer nutzen, besonders in ZH):

```
const asList = (v, lang) => {
  try {
    const out = pickList(v, lang);
    return Array.isArray(out) ? out : [];
  } catch { return []; }
};
const safeText = (v, lang) => {
  try { const s = pickText(v, lang); return (s ?? "").toString(); }
  catch { return ""; }
};
```

🛒 Einkaufsliste (Auto-Summen)

- Parser für `Zutat Menge Einheit` (g|ml|l|EL|TL|Stück)
- Einheiten-Normalisierung: `l → ml`
- **4 Gruppen:**

1) Protein/Fisch/Tofu

2) Gemüse/Pilze

3) Reis/Nudeln/Sättigung

4) Algen/Brühen/Würze

- Ausgabe sortiert (alphabetisch), Mengen `Math.round`.

🧪 Tests (am Dateiende)

Jede Datei enthält **Tests()** mit:

- `DATA.length === 21`
- IDs eindeutig
- Lunch hat **keinen** Reminder, Frühstück/Abend **müssen** Reminder haben
- Jede Zutatenliste ≥ 5, jeder Step ≥ 3
- `LIST_SUMMARY` hat **4 Gruppen**
- Konsolen-Log bei Erfolg.

🧭 Interaktive UI-Elemente (Top-Bar)

- Tabs (State: `"kochbuch"` | `"einkauf"`)
- Drei Buttons:
- **PDF erzeugen** → `exportPDFById("cookbook-root" | "list-root", ...)`
- **HTML exportieren** → `exportHTMLById(...)`
- **Drucken** → `window.print()`
- Nach Export: Blob/URL unterhalb des Tabs als Download-Link anzeigen.

🌐 Sprache

- DE-Datei: `lang`-State initial `"de"`; ZH-Datei: initial `"zh"`.

- Texte via `UI[lang]` + `pickText/pickList`.
- **Kein** zusätzlicher Language-Switcher im Top-Bar nötig (Sidebar übernimmt).

✅ Output-Anforderung

Erzeuge **zwei komplette, lauffähige JSX-Dateien** (DE & ZH) gemäß obiger Vorgaben.

Keine Erklärtexte, nur die beiden Datei-Inhalte.

Benutze exakt die Referenz-Stile von *Woche-4-2025-10-20* (IDs, Klassen, Farben, Struktur).

Dateinamen:

- `src/plans/Woche-{{WEEK_NR}}-{{START_DATE}}.de.jsx`
- `src/plans/Woche-{{WEEK_NR}}-{{START_DATE}}.zh.jsx`

📦 Beispiel-Header (nur Referenz, nicht duplizieren)

```
import React, { useEffect, useMemo, useRef, useState } from "react";
import { exportPDFById, exportHTMLById } from "../utils/exporters";
import { buildEmbedCss } from "../utils/embedCss";
import { UI } from "../i18n-ui";
import { pickText, pickList } from "../i18n-data";
```

🚦 Qualitäts-Checkliste (bei Generierung)

- 21 Rezepte, IDs korrekt (mo|di|…|so)-(f|m|a)
- Keine leeren Felder: title, desc, story, target, ingredients[≥5], steps[≥3], checks, swaps, side
- Reminder-Regel passt
- Wochenübersicht zeigt 7×3 Karten (Titel + 🌾 KH + 💊 ggf.)
- PDF- und HTML-Export funktionieren (getrennte Orientierung)
- Keine experimentellen CSS-Farben (`color-mix`, `oklab` etc.)

🧪 Mini-Test-Code (am Ende jeder JSX)

```
function Tests() {
  try {
    if (DATA.length !== 21) throw new Error("DATA length must be 21");
    const ids = new Set(DATA.map((r) => r.id));
    if (ids.size !== 21) throw new Error("IDs not unique");
    DATA.forEach((r) => {
      const isLunch = /-m$/.test(r.id);
      if (isLunch && r.remind) throw new Error("Mittag darf keinen
Reminder haben");
      if (!isLunch && !r.remind) throw new Error("Frühstück/Abend
brauchen Reminder");
      if (!Array.isArray(r.ingredients) || r.ingredients.length < 5)
throw new Error(`Zutaten zu wenig: ${r.id}`);
      if (!Array.isArray(r.steps) || r.steps.length < 3) throw new
Error(`Steps zu wenig: ${r.id}`);
    });
    const groups = Object.keys(LIST_SUMMARY);
```

```
    if (groups.length !== 4) throw new Error("LIST_SUMMARY groups
missing");

    console.log("[GhibliKitchen] All tests passed (JSX).");

  } catch (e) {

    console.error("[GhibliKitchen] Tests failed:", e);

  }

}
```