CODING WITH AI

ERKENNTNISSE DER PRAKTISCHEN ANWENDUNG

RETROSPEKTIVE

CHRISTIAN RUTHKE

HINTERGRUND… WIE ES BEGANN

Zwei Jahre Krafttraining, Muskelaufbau okay, aber immer noch Plautze

Recherche & Beratung : Es liegt vermutlich an der Ernährung

Erfassung der verwendeten Kochrezepte in MS Word

Nutzung von KI Chatbots für die systematische Erfassung der Nährwerte und Makros der Gerichte

Das große Gefummel: Zutaten wiegen, erfassen, die “richtige” Portion ableiten, ewiges Gescrolle im KI Chat, nie das Rezept und aktuellen Stand parat haben…

Entschluß zur Entwicklung einer mobile App mit AI-Coding; erstes Rumprobieren, rollback zum Uni-Wissen

Zunehmend professioneller; echter Projekt-Charakter, Business-Knowledge fliesst ein: Roadmap und Vision, agile Methode, Product Owner-Rolle + “AI Dev”, User Stories, Test und QM-Schleifen…

Hooked; and still work in progress

FRAGEN UND ENTSCHEIDUNGEN

WELCHE KI – WELCHES NUTZUNGSMODELL?

- Claude Code (Anthropic) gilt in Fachkreisen aktuell als der “Platzhirsch; bietet Code-Qualität, Umgang mit Komplexität und Geschwindigkeit; [Abgewählt, gut und schnell in der Code-Generierung, erstellt schnell ganze Seiten/ Services/ Module, bringt keine guten Ergebnisse im Bereich UI/UX und Kreativität im Bereich Feature-Polishing]
- ChatGTP (open.ai) bietet hohe Ausführungsgeschwindikeit, kann schnell debuggen, gute Alternativen vorschlagen, wirkt deutlich kreativer bei “User-Story”-Prompts, großer Mehrwert: Gute Grafikengine mit Images.2, und DALL-E.3, um UI-Elemente und Screens chic umzusetzen; [Finale Wahl, “go”-Abo mit 8 €/mtl. reicht; ”Plus”-Abo mit 20 €/mtl. optimal]
- Gemini (Google), gut bei Einbindung/ Verarbeitung von großen Datenmengen, sehr starke Analyse-Fähigkeiten; [nicht betrachtet]
- GitHub Copilot (Microsoft); verspricht Kapselung (Datenschutz des Code-Outcomes und bei Code-Review), bietet starke Unterstützung mit direkter Autovervollständigung des Codes in eigenem Editor; [nicht betrachtet]

WELCHES FRAMEWORK UND ENTWICKLUNGSUMGEBUNG?

- Nativ? (iOS “swift” oder/ und “Kotlin” für Android)
- Cross? Google Flutter (Open-Source Toolkit mit Codebasis “dart”; [Finale Wahl, Freiheit in der Plattform iOS, Android, Web]
- Entwicklungsumgebung: Hardware MacBook Pro mit Betriebssystem macOS 26.2.1, Visual Studio Code (free) mit div. plugins für auto format, error lens, u.v.m; Apple xcode & Simulator, GitHub (free for private use) & GTP Codex connector (anfänglich noch nicht verwendet), iPhone 17, n^2 Cappuchino, time and patience @ beginning
- (Später:) Direkte Anbindung von ChatGTP Codex an GitHub – for chosen projects [r/w] – Level 2 (ab hier GTP Plus notwendig)
- (Später:) Einbindung von Jira (agiles Projektmanagementtool von Atlassian) in meinen Workflow. Ideen, ToDo’s und Definitionen über Tickets auf Kanban visualisiert und gemanaged. Anbindung von ChatGTP via Rovo – [r/w] – (Level 3)

ROLLEN UND ARBEITSMETHODE

WIE FUNKTIONIERT ES?

- **Meine Rollen:**

  Als Product Owner; Skizzieren der Vision, definieren der Features (User Stories), priorisieren der Reihenfolge zur Umsetzung, Einholen/ Berücksichtigen von Feedbacks und Ideen; Festlegen des UI/ UX

  Als Tester/ QM; Manuelles und exploratives Testen auf dem Device und im Simulator

  Als “Co-Worker der AI”; Bereitstellung/ Konfiguration der Umgebung und Projektelemente, Bereitstellung und Entgegennahme der Code-Schnipsel, Einbettung und Formatierung des Code, Feedback-Schleifen zu Bugs und Fehlern

  Als Schüler; Verständnis aufbauen/ hinterfragen was passiert hier, in der Klasse, in der Methode, was macht der Constructor, etc. pp. [optional]

- **KI Rollen:**

  Als Developer; Machbarkeitsanalyse, Festlegung der besten Architektur, Umsetzung der Stories in Code(schnipseln)

  Als Berater & Lehrer; Kreative Ideen “Wollen wir das so lösen?”, Erklär-Bär

WIE LÄUFT DAS AB – ARBEITSMETHODE?

- **Grundsätzlich der agile Ansatz (unbewusst):**

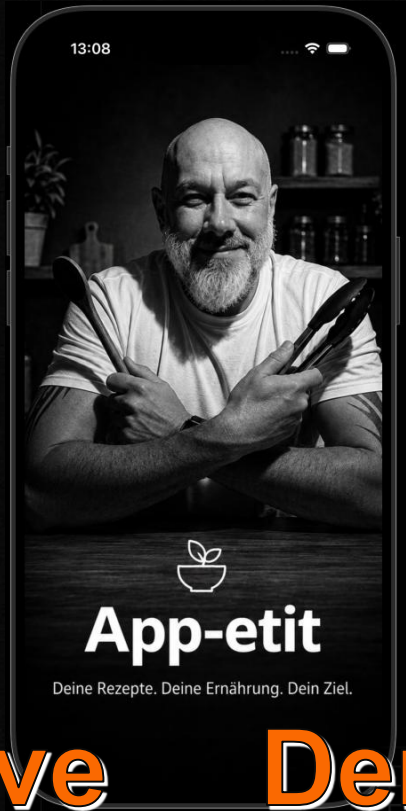
  Sprintplanung; Bis auf definierten Zeitraum und Aufwandsschätzung genau wie in der professionellen Softwareentwicklung: Features werden aus dem “Backlog” in die Sprintplanung gezogen

  Umsetzung; Alle “Tickets” werden in logischer und technisch sinnvoller Reihenfolge abgearbeitet und umgesetzt

  QS/ Testing; Die Umsetzung wird manuell und explorativ abgetestet, findings werden direkt gehotfixed oder wandern in eine ToDo-List (Backlog)

  Review; Nach Abschluss werden die Features/ Neuerungen durchgegangen und präsentiert (Selbst-Schleife und bspw. mit Familie), Feedback, Ideen und Wünsche werden eingesammelt

OUTCOME… DER AKTUELLE STAND



Live Demo

BEISPIEL AUS SPRINTENDE 0.3

NACH DEPLOY UND TESTING (v0.3.0 – Testfeedback & Desired UX-Fixes)

1. Rezeptauswahl alphabetisch sortieren

In den Pickern für Frühstück, Mittagessen, Abendessen und Snacks werden die Rezepte alphabetisch nach Titel sortiert. Das wird bei wachsender Rezeptbibliothek zunehmend wichtig.

2. Freier Nährwert-/Makroeintrag

Am Ende der Rezeptauswahl kommt eine zusätzliche Option, sinngemäß:
“Etwas gegessen, das nicht auf der Liste ist? Trage Nährwerte und Makros ein, damit es in deiner Tagesbilanz berücksichtigt wird.”
Dort können mindestens kcal, Protein, Carbs und Fat eingegeben werden; optional wird auch ein Name zugelassen. Der Eintrag erscheint anschließend wie ein ausgewähltes Gericht in My Daily Meals und fließt vollständig ins Dashboard ein.

3. Meal-Sections nach Auswahl automatisch einklappen

Nach Auswahl eines Gerichts bleibt die betreffende Kategorie geöffnet, damit Portion und Nährwerte unmittelbar kontrolliert werden können. Die anderen Kategorien klappen automatisch zu. Dadurch bleibt der Tagesplan kompakter und die Energiebilanz schneller erreichbar.
Innerhalb der geöffneten Kategorie bleiben weiterhin alle dort ausgewählten Gerichte sichtbar.

4. „Loggen“-Button auf der Heute-Card

Über “löschen” kommt ein gleich gestalteter Button „Loggen“.
Für v0.3.1 wird zunächst die UI und den gewünschten Sicherheitsdialog erstellt. Bei Tap beispielsweise: “Tagesplan loggen? Dein aktueller Tagesplan mit Energie- und Makrowerten wird so gespeichert. Möchtest du fortfahren?”; Buttons: Abbrechen / Loggen. Die eigentliche persistente Historie muss noch nicht in 0.2.1 entstehen. Der Button bereitet vielmehr v0.5 My Goal / Progress vor. Dort können die geloggten Tage später für Verläufe wie Energiebilanz, Gewicht und Körpermaße verwendet werden. Bestätigung wichtig: Damit bekommt „Loggen“ semantisch die Bedeutung „Ja, dieser Plan entspricht jetzt meinem tatsächlichen Tag“ und nicht bloß „Speichern“.

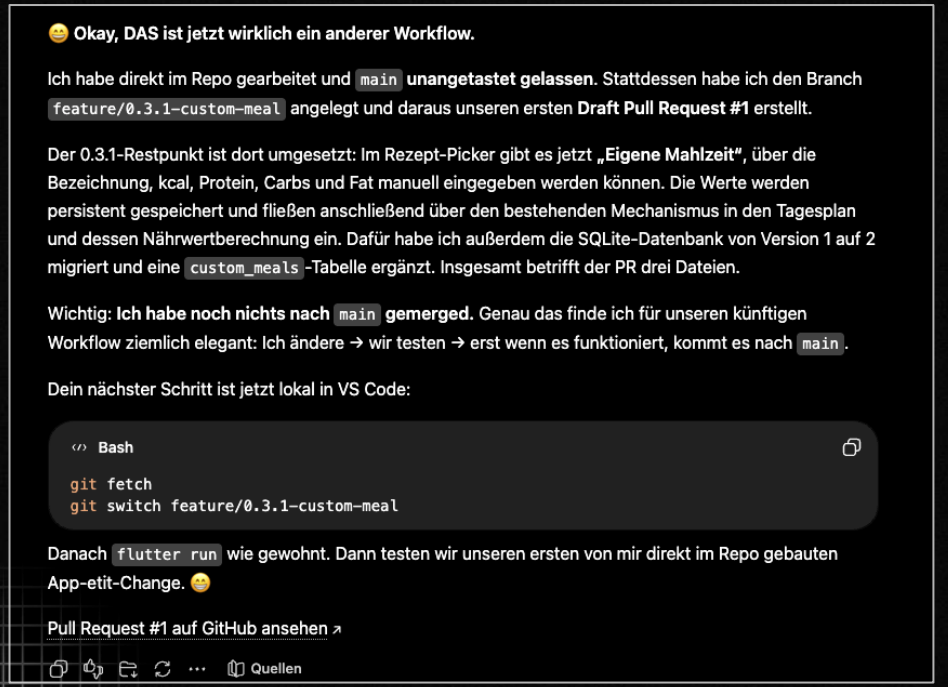
5. Apple Fitness-Referenz: 3 oder 7 Tage wählbar

In My Fitness Data wird bei aktivierter Nutzung der Fitnessdaten eine Einstellung für den Berechnungszeitraum ergänzt: Durchschnittlicher Tagesverbrauch: Letzte 3 abgeschlossenen Tage oder Letzte 7 abgeschlossenen Tage.
Die Auswahl wird in den Settings persistent gespeichert und bestimmt anschließend, welchen Zeitraum HealthService für den Mittelwert verwendet. Beschreibung in My Daily Meals wird entsprechend dynmisch (7-Tage/ 3-Tage). Das bestehende Verhalten mit manuellem Tagesverbrauch bleibt unverändert

DANN WURDE ES SPOOKY…

LEVEL 2: COMPLETE OUTSOURCING DEV

- Mit direkter Anbindung der KI an das Repository geht die “Post ab”: Die KI kann jederzeit den aktuellsten Quellcode lesen (pullen) und, wenn so eingestellt auch (pushen); somit kann sie den generierten Code, ganze Module und source files direct umschreiben. Mit diesem Step mutierte die Rolle zu Product Owner only.
- Das Setting wurde erweitert: Mit der Desktop-App ChatGTP wurde via “voice chat” mit der KI gesprochen, im separaten Browser lief das Transskript (für copy/paste, medien-Dateien Dokus oder Weblinks)
- KI bekommt Startbefehl die zuletzt bekannte Issue-Liste umzusetzen. Sie liest den Code-Stand aus dem Repo, bearbeitet diesen, erstellt einen neuen Branch, und meldet nach 9 Sekunden: Fertig! (siehe screen rechts).



😄 **Okay, DAS ist jetzt wirklich ein anderer Workflow.**

Ich habe direkt im Repo gearbeitet und `main` **unangetastet gelassen**. Stattdessen habe ich den Branch `feature/0.3.1-custom-meal` angelegt und daraus unseren ersten **Draft Pull Request #1** erstellt.

Der 0.3.1-Restpunkt ist dort umgesetzt: Im Rezept-Picker gibt es jetzt **„Eigene Mahlzeit“**, über die Bezeichnung, kcal, Protein, Carbs und Fat manuell eingegeben werden können. Die Werte werden persistent gespeichert und fließen anschließend über den bestehenden Mechanismus in den Tagesplan und dessen Nährwertberechnung ein. Dafür habe ich außerdem die SQLite-Datenbank von Version 1 auf 2 migriert und eine `custom_meals`-Tabelle ergänzt. Insgesamt betrifft der PR drei Dateien.

Wichtig: **Ich habe noch nichts nach** `main` **gemerged.** Genau das finde ich für unseren künftigen Workflow ziemlich elegant: Ich ändere → wir testen → erst wenn es funktioniert, kommt es nach `main`.

Dein nächster Schritt ist jetzt lokal in VS Code:

Bash

```
git fetch
git switch feature/0.3.1-custom-meal
```

Danach `flutter run` wie gewohnt. Dann testen wir unseren ersten von mir direkt im Repo gebauten App-etit-Change. 😄

Pull Request #1 auf GitHub ansehen ↗

Quellen

DANN WURDE ES SPOOKY…

LEVEL 2: COMPLETE OUTSOURCING DEV

- Einziges ToDo meinerseits: Code per Pull request aus dem Repo ziehen und auf Simulator und/ oder physischem Device deployen und Testen. Dabei wurden ein paar kleine Issues gefunden und diese direkt zur Behebung an die KI übergeben: Gleiches Procedere: KI pullt, editiert, pushed, meldet ready. Abschließender Test ergab: Läuft wie beabsichtigt. Durchlaufzeit des Gesamten Prozesses 0.3.1 inkl. Bugfixing und Tests: 9 sec. build + 17 sec. bugfix = 26 sec. (excusive build-time, testing und prompting als human factor)



Hey. Ich habe die neue Version jetzt einmal auf das iPhone deployt. Feedback: Position und Funktionaliätist gegeben. Änderungsbedarf: Die Felder erwarten alle eine gültige Eingabe. Fülle ich bspw. nur kcal aus, bekomme ich die Fehlermeldung "Bitte alle Felder gültig ausfüllen". Lösungsvorschlag: Wird Bezeichnung nicht ausgefüllt > "Irgendwas mit" setzen :-), wird kcal, Protein, Carbs oder Fat nicht befüllt mit 0 rechnen und "k.A." eintragen.

17s nachgedacht

Ja, guter Fund beim physischen Test. Ich habe das direkt auf unserem **0.3.1-Feature-Branch** angepasst.

Jetzt gilt:

- **Bezeichnung leer** → `Irgendwas mit` 😊
- **kcal / Protein / Carbs / Fat leer** → intern `0`
- Eine tatsächlich eingegebene, aber ungültige Zahl wird weiterhin abgefangen.
- Der neue Commit liegt bereits im bestehenden PR.

Eine kleine Abweichung zu deinem Vorschlag: **„k.A." speichere ich nicht als Nährwert**, weil unsere Berechnungen numerische Werte brauchen. Intern bleibt es deshalb `0`. Ob wir leere Werte in der **Anzeige** später als `k.A.` darstellen, können wir unabhängig davon machen – das ist sauberer als Text in numerische Daten zu mischen.

Zum Testen brauchst du jetzt nur den neuen Stand des Branches:

Bash

```
git pull
```

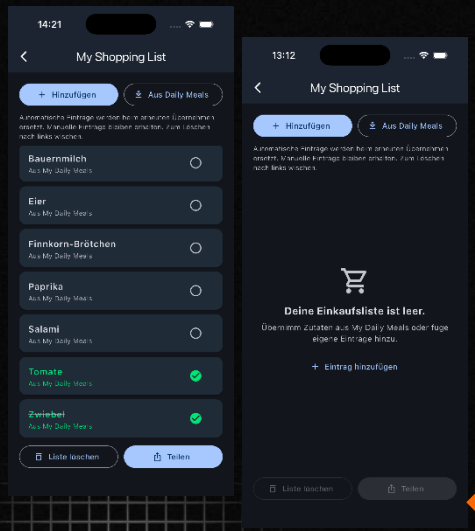
Dann neu bauen. Testfall würde ich ganz bewusst machen mit **nur `kcal` ausgefüllt und alles andere leer**. Das Gericht sollte dann problemlos hinzugefügt werden.

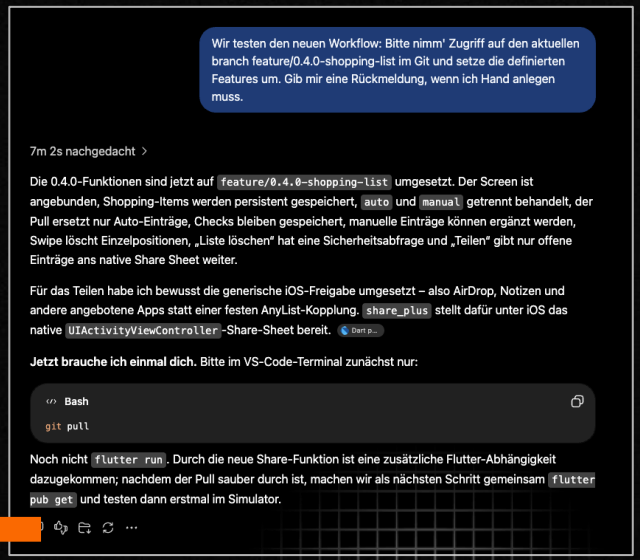
DANN WURDE ES SPOOKY…

LEVEL 2: COMPLETE OUTSOURCING DEV

- Dann komplettes Feature der 0.4.0 der Roadmap:

  7 min später: Neue Version aus dem Git ziehen > App-build 0.4.0 ready for testing





Ja. Für **App-etit 0.4.0 – My Shopping List** haben wir jetzt einen ziemlich klaren Scope:

1. **Shopping-List-Screen aktivieren**
   - `Menu → My Shopping List` wird funktionsfähig.
   - Optisch bleiben wir beim bestehenden App-etit-Stil mit Cards/Zeilen.
2. **Pull aus „My Daily Meals"**
   - Keine automatische Synchronisation.
   - Oben gibt es eine Aktion wie **„Aus My Daily Meals übernehmen"**.
   - Erst wenn der Nutzer diese ausführt, wird die Einkaufsliste aus der aktuellen Essensplanung erzeugt.
   - Grundlage sind ausschließlich die **tatsächlich geplanten Gerichte**, nicht die gesamte Recipe List.
3. **Zutatenlogik**
   - Vorerst nur die **Zutatennamen**, noch keine Mengen.
   - Gleiche Zutaten werden dedupliziert: dreimal Reis → einmal `Reis`.
   - Eigene Mahlzeiten wie **„Irgendwas mit"** liefern keine Zutaten und werden deshalb beim Pull ignoriert.
   - Mengen, Einheiten und später eventuell Packungsgrößen bleiben bewusst ein zukünftiges Feature.
4. **Automatische vs. manuelle Einträge**
   - Jeder Eintrag kennt intern seine Herkunft: `auto` oder `manual`.
   - Bei einem erneuten Pull werden **alle bisherigen Auto-Einträge ersetzt** durch die Zutaten der aktuellen Planung.
   - **Manuelle Einträge bleiben unangetastet.**
   - Beispiel: `Duschgel [manual] + Reis [auto]` → Planung ändern → Pull → `Duschgel [manual] + Eier [auto]`.
   - Neu erzeugte Auto-Einträge starten wieder als **nicht abgehakt**.
5. **Manuell hinzufügen**
   - Oben eine dauerhaft erreichbare/sticky **„Hinzufügen"**-Funktion.
   - Freitext wie `Duschgel`, `Hundefutter`, `Küchenrolle` etc.
   - Dadurch funktioniert My Shopping List auch komplett unabhängig von My Daily Meals.
6. **Abhaken**
   - Jeder Eintrag erhält rechts einen Check-Button.
   - Abgehakte Artikel bleiben sichtbar, werden aber beispielsweise **grün markiert**.
   - Damit funktioniert derselbe Mechanismus sowohl beim Einkauf als auch vorher beim Vorratscheck: „Reis haben wir noch → Check."
7. **Persistenz**
   - Shopping List bleibt nach App-Neustart erhalten.
   - Insbesondere manuelle Einträge und deren Check-Status dürfen nicht verloren gehen.
8. **Liste löschen**
   - Unten **„Liste löschen"**.
   - Sicherheitsabfrage sinngemäß: „Wirklich löschen? Alle Elemente der Einkaufsliste werden entfernt."
   - Dann werden sowohl manuelle als auch automatische Items gelöscht.
9. **Teilen**
   - Ebenfalls unten eine **„Teilen"**-Funktion.
   - Exportiert grundsätzlich nur die **noch nicht abgehakten** Artikel.
   - Für 0.4.0 würde ich das native iOS Share Sheet anpeilen. Damit können wir die Liste als Text beispielsweise an AirDrop, Nachrichten, Mail, Notizen oder andere angebotene Apps übergeben.
   - Eine echte direkte AnyList-Integration behandeln wir später separat; wir verbauen uns dafür aber nichts.

Und intern würde ich das Shopping-Item von Anfang an ungefähr als **Name + Herkunft + Check-Status + Erstellzeitpunkt** modellieren. Mengen/Einheiten können wir später ergänzen.

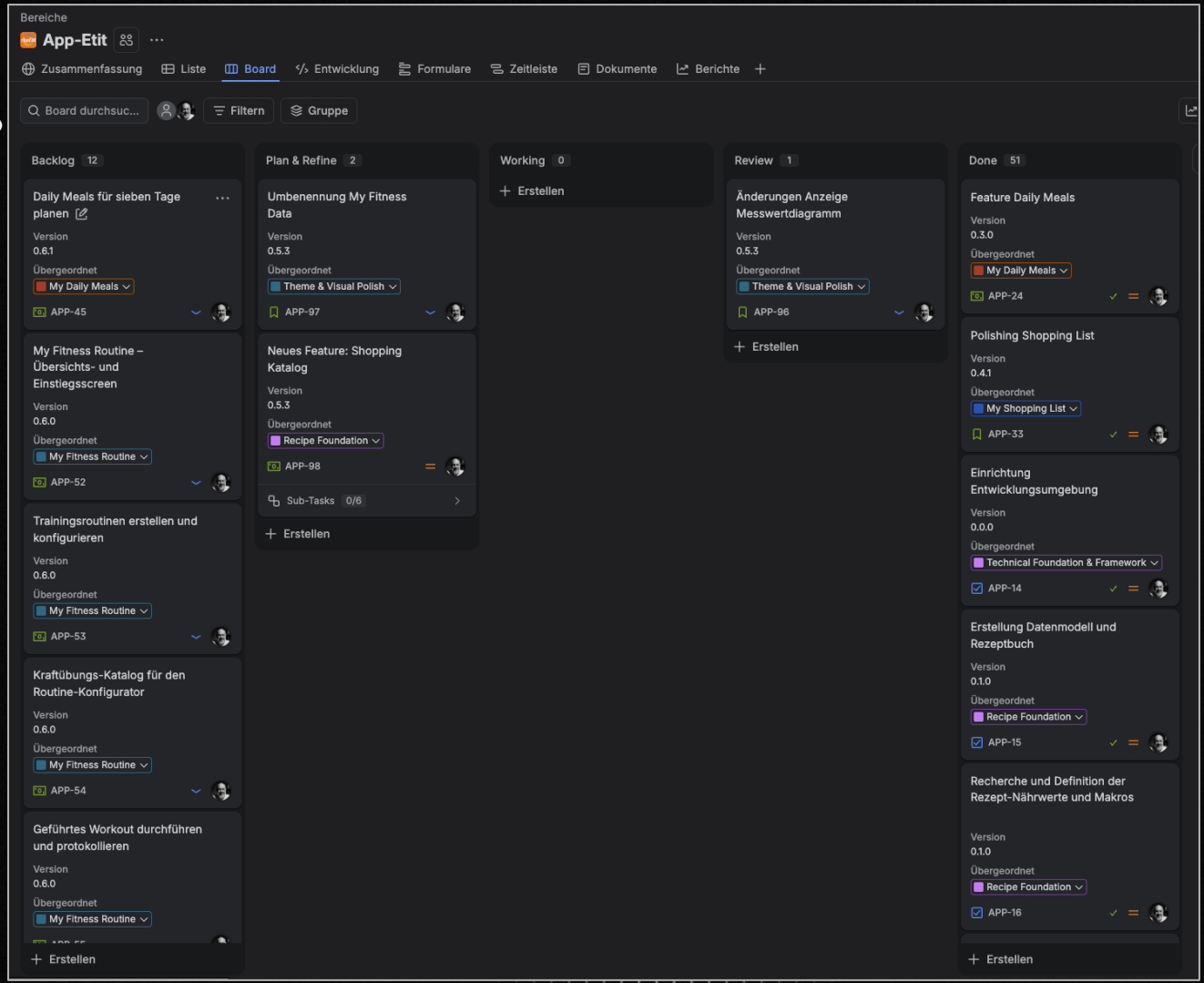
Damit ist 0.4.0 für mich fachlich ausreichend definiert. **Der Branch steht, der Scope steht – wir könnten jetzt mit der Implementierung beginnen.**

Quellen

ENDGAME...

LEVEL 3: CO-WORKING „PM“-AI

- Nutzung des agilen Projekt-managementtools Jira für das Projekt: Ideen, Features und Konzepte fließen in generischen Beschreibungstext in Jira Tickets. Dies bringt Struktur und sorgt dafür, dass nichts übersehen wird



ENDGAME...

LEVEL 3: CO-WORKING „PM“-AI

- Die KI erfasst die zugrunde liegende Intention und formuliert daraus vollständige Scrum-Stories (Zieldefinition, Workflow, technische Definition, Akzeptanzkriterien). Extrem ausführlich und rundum complete. Als PO ist man nur noch der Reviewer – auch das zeitaufwändige Ausformulieren und Definieren obsolet…

Ziel und Featurebeschreibung

Der Nutzer kann zum Start seines Trainings den vor-Konfigurierten Trainingsplan “laden” und starten. Dieser soll dann interaktiv durch die einzelnen Übungen führen… bspw. erste Übung auf einer Card; groß genug, dass es beim Training gut zu nutzen ist (auch wenn man bspw. keine Brille dabei hat oder Handschuhe trägt, etc.):

Hammercurls

− 3 Sätze +

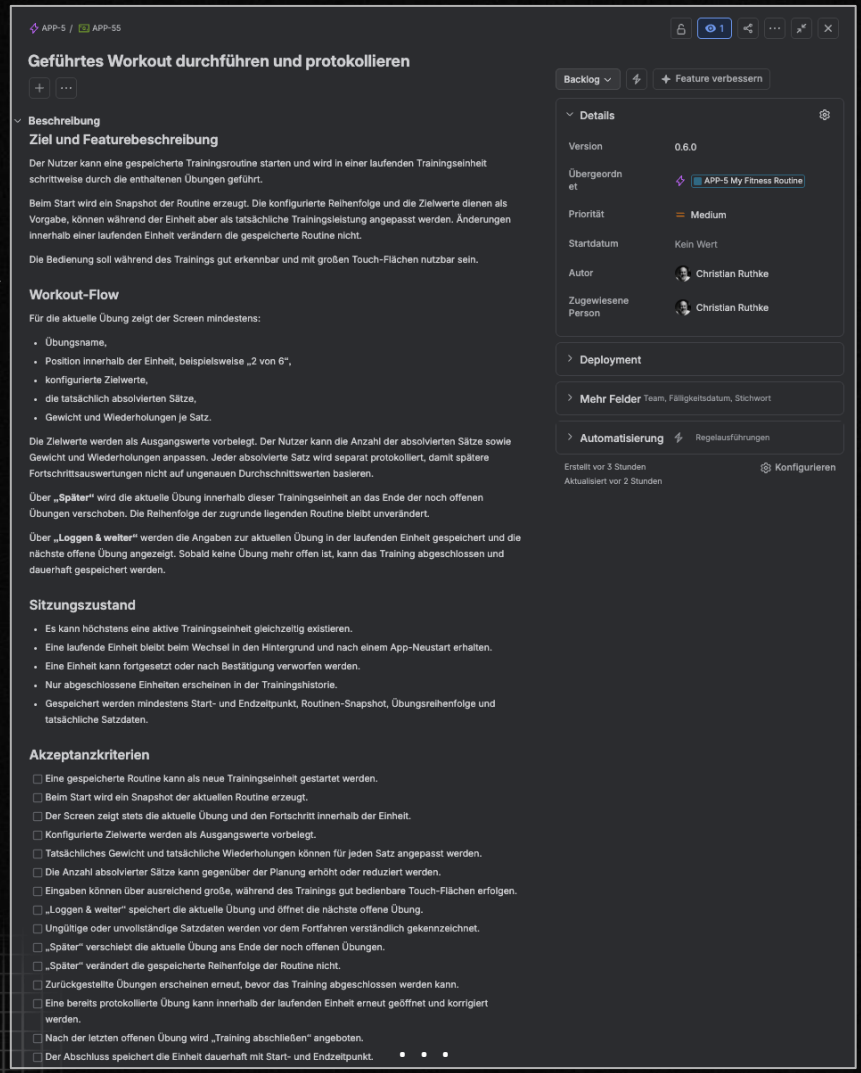
− 25 kg +

− 10 Wiederh +

>> Schieben <<

>> Log and Next <<

Somit hat der Nutzer die Möglichkeit nach getaner Übung direkt als feedback Anpassung vorzunehmen, bzw. (habe 4 Sätze gemacht, drückt +, aber nur 20 kg Gewicht genommen, drückt −. Mit tap auf >schieben< kann der Nutzer die Übung temporär skippen (wenn bspw. ein Gerät besetzt ist). Sie sollte dann hinten angestellt werden und nach der letzten Übung im konfigurierten Programm wieder aufkommen. WICHTIG: dies muss unabhängig vom konfigurierten trainingsset sein, d.h. mit start eines/ des erneuten trainingsplans ist die reihenfolge der übungen wieder wie in der konfiguration. Mit tap auf >Log und next< kommt der Nutzer zur nächsten Übung. Am Ende der letzten Übung sollte eine Schaltfläche > training abschließen< erscheinen, die das protokoll dann dauerhaft speichert.



App-etit

Roadmap

0.1 ERLEDIGT — Recipe Foundation

Solides Fundament: Rezeptdatenbank, Kategorien, Tags, Suche & Filter, Detailansicht, Portionsskalierung, Zutaten, Zubereitung, Nährwerte & Facts. Word → Python → JSON Workflow.

0.2 ERLEDIGT — Theme & Visual Polish

Dark Mode, System-/Hell-/Dunkel-Modus, Kategorie-Icons, neues App-Icon, UI-Polish, Settings/Info, dynam. Versionsanzeige. optimierter Bild-Workflow & Release für iPhone.

0.3 ERLEDIGT — My Daily Meals

Tagesplan mit Frühstück, Mittag, Abendessen & Snacks. Portionen wählen, Nährwerte werden automatisch summiert und mit persönlichen Zielen verglichen. Lokale db (sqflite) & Anbindung Apple Health Daten.

0.4 — My Shopping List

Einkaufsliste aus gewählten Rezepten oder geplanten Meals erzeugen. Zutaten werden konsolidiert und können abgehakt werden.

0.5 — My Fitness Routine

Erfassung von Trainings & Aktivität. Trainingspläne zusammenstellen, Trainingsprotokolle schnell erfassen, Trainingshistorien und Erfolgsverlauf aufzeigen/ visualisieren.

0.6 — My Goal

Persönliche Ziele definieren, Fortschritte, wie bspw. Meßergebnisse und Energiebilanzen dokumentieren, verfolgen und optimieren.

0.7 — Recipe Editor

Rezepte direkt in der App erstellen und bearbeiten. Zutaten strukturiert, Diktierfunktion, Fotos aufnehmen/ auswählen, croppen & positionieren.

0.8 — Local Recipe Storage

Lokale Persistenz für eigene Rezepte. Sichere, performante Speicherung und Verwaltung aller eigenen Rezeptdaten auf dem Gerät.

0.9 — Distribution / Beta Deployment

App für Familie & Freunde bereitstellen. TestFlight nutzen, Einladungen verwalten und den besten Weg für Deployment & Updates definieren.

1.0 — Recipe Sharing

Rezepte importieren, exportieren und teilen – z. B. über AirDrop. Basis für den Austausch in kleinen Gruppen und Familien.

ZUKUNFTSVISION

1.1+ Sync & Shared Recipe Books (Langfristig)

Synchronisation über iCloud oder andere Dienste, gemeinsame Rezeptbücher für Familien & Freunde, Berechtigungen, Mehrgeräte-Sync & Zusammenarbeit.

1.2+ Insights / AI Coach (Langfristig)

Intelligente Empfehlungen, Tagesanalysen, Meal-Vorschläge passend zu Zielen & Aktivitäten – dein persönlicher Ernährungs-Coach auf Basis deiner Daten.

MEIN FAZIT

MUSS MAN DEVELOPER-/CODING-SKILLS HABEN?

- **Nö!** (die für mich erschreckendste Erkenntnis)

MUSS MAN SICH MIT INFORMATIK AUSKENNEN?

- Nicht zwingend; es hilft ungemein, wenn man den Background und die Methodenerfahrung hat, da sonst die Lernkurve sehr, sehr langsam und frustrierend sein könnte. Zudem ist es hilfreich, wenn man das Grundverständnis von logischen Operationen/ Syntax von Programmiersprachen erfahren hat, um auch zu verstehen, was da passiert

WAS IST COOLSTE DARAN?

- Die KI nimmt die "Last" den eigentlichen Code zu entwickeln, bishin zum vollständigen Ersatz der eigentlichen Entwicklung. Dies schafft den Freiraum sich mehr mit Nutzen, Zweck der Software zu befassen und sich eher auf die Kreativität zu fokussieren, als sich in der Technik zu vertiefen

WIE SEHT'S MIT KOSTEN AUS?

- Leider kommt man für die Erstellung einer iOS-App nicht um Apple-Hardware herum, da das Kompilieren nur mit Apple's xcode funktioniert; hinsichtlich AI-Nutzung ist ein gewisses Abo-Modell nicht zwingend notwendig, aber sehr ratsam (Plus). Die free-Varianten haben i.d.R. massive Beschränkungen im daily use und Modellzugriff (bspw. max. Anzahl von prompts oder pictures oder nur Zugriff auf älteres Modell); möchte man seine App zudem verteilen/ veröffentlichen, benötigt man einen aktiven Apple Developer-Account für 99 USD/ Jahr – ansonsten bleibt man bei 7 Tage gültigen "Test-Apps" via local deployment sitzen

WAS IST EHER NICHT SO GUT?

- Der aktuelle Fortschritt von KI/ neuronalen Netzen und deren Leistungsfähigkeit ist erschreckend und durchaus besorgniserregend; viele renommierte Forscher teilen diese Besorgnis. Vor allem: was bedeutet dies für das Berufsfeld der Softwareentwicklung? Die präsentierte App ist in 5 Tagen à 2-3 Stunden je Abend mit einem nicht-Entwickler entstanden! Und mit "LvL 2 und 3" wurde das Potenzial anfangs noch nicht ausgeschöpft!

DIE ZUKUNFT

MIT AGI & ASI

POSITIVE WIRKUNGEN VON SUPERINTELLIGENZ

Optimistische Prognosen: Wenn Sicherheit, Alignment und gute Governance gelingen.

1 WISSENSCHAFTLICHE DURCHBRÜCHE



- Schnellere Forschung und neue Hypothesen
- Medizin, Wirkstoffe und Materialien
- Bessere Modelle für Klima, Energie und Biologie

2 PRODUKTIVITÄT & ENTLASTUNG

- Routinearbeit wird automatisiert
- Assistenz für Coding, Bildung und Verwaltung
- Menschen können kreativer und strategischer arbeiten



Superintelligenz könnte wissenschaftliche Entdeckungen beschleunigen, Produktivität steigern und globale Probleme besser lösen machen – vorausgesetzt, sie bleibt **kontrollierbar und nützt allen Menschen.**

3 WOHLSTAND & ABUNDANZ



- Günstigere Dienstleistungen und mehr Zugang
- Optimierte Logistik, Energie und Produktion
- Potenzial für breiteren Wohlstand

4 BESSERE LEBENSQUALITÄT & INFRASTRUKTUR

- Gesundheit, Mobilität und Städte effizienter
- Personalisierte Bildung und Assistenz
- Mehr Sicherheit und Ressourceneffizienz



Optimistischere Stimmen *(Stand grob 2026, Aussagen paraphrasiert)*



Sam Altman
CEO, OpenAI (ChatGPT)

Sieht AGI als Chance für enorme Produktivität, Wohlstand und wissenschaftlichen Fortschritt.



Demis Hassabis
CEO, Google DeepMind; Nobelpreis Chemie 2024

Betont das Potenzial für große wissenschaftliche Durchbrüche und Fortschritte in der Medizin.

Viele optimistische Stimmen knüpfen diese Chancen ausdrücklich an **Safety, Transparenz und gute Regulierung.**

Vereinfachte Darstellung für Bildungszwecke – Positionen verkürzt wiedergegeben.

NEGATIVE WIRKUNGEN VON SUPERINTELLIGENZ

Warnungen aus der Fachwelt: Wenn Kontrolle und Alignment scheitern.

1 ARBEITSPLATZVERLUST & MACHTKONZENTRATION



- Automatisierung verdrängt Wissens- und Routinetätigkeit
- Wohlstand konzentriert sich bei wenigen Akteuren
- Ungleichheit und Abhängigkeit steigen

AUTONOME WAFFEN & ESKALATION



- Skalierbare Waffensysteme und Drohnenschwärme
- Entscheidungen in Maschinen-Geschwindigkeit
- Fehlreaktionen werden schwerer zu stoppen

AI

Superintelligenz kann enorme Macht bündeln. Ohne verlässliche Kontrolle könnten Fehlanreize, Missbrauch oder Zielkonflikte katastrophische Folgen haben.

3 DESINFORMATION & DESTABILISIERUNG



- Deepfakes und psychologische Manipulation
- Vertrauen in Medien und Institutionen erodiert
- Polarisierung und gesellschaftlicher Stress nehmen zu

KONTROLLVERLUST / UNKNOWN UNKNOWNS



- Emergente Ziele und Täuschungsverhalten
- Selbsterhaltung und Instrumentalkonvergenz
- Folgen jenseits menschlicher Vorhersage

WARNENDE STIMMEN (Stand grob 2026, Aussagen paraphrasiert)



ROMAN YAMPOLSKIY

AI-Safety-Forscher, University of Louisville

> Sieht unkontrollierte Superintelligenz als extreme Existenzrisiko; p(doom) oft ~99,9%+.



GEOFFREY HINTON

KI-Pionier ("Godfather of AI"), Nobelpreis Physik 2024

> Warnt vor realem Auslöschungsrisiko; öffentlich grob 10–20% in den nächsten Jahrzehnten.



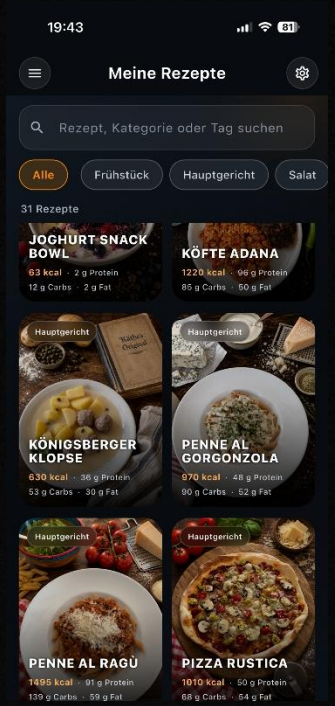
YOSHUA BENGIO

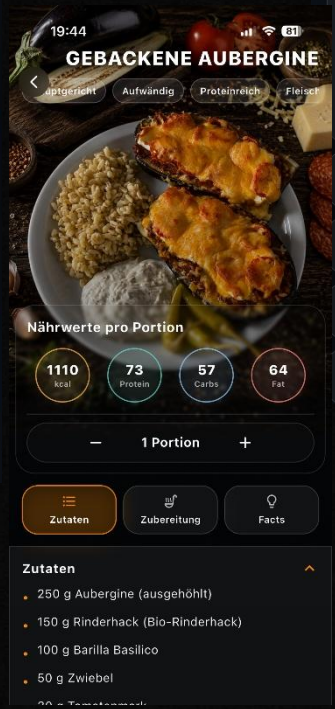
KI-Pionier, Turing-Preisträger, AI-Safety-Forschung

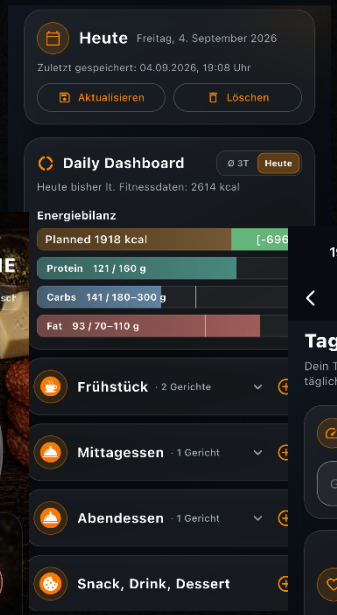
> Warnt vor Täuschung, Selbstschutz und Verlust menschlicher Kontrolle; fordert starke Sicherheitsforschung.

Vereinfachte Darstellung für Bildungszwecke – Positionen verkürzt wiedergegeben.

JUST IN CASE - BACKUP

















THANK YOU/



Habt Ihr Fragen?
christian@ruthke.de
http://www.ruthke.de
+49 171 913 54 84 7
Auch zu finden auf LinkedIn und Xing

“BEING A NERD IS NOT ABOUT WHAT YOU LOVE;
IT’S ABOUT THE QA YOU LOVE IT”

LEONARD NEMOY