

DER SAMWER-TRICK MIT GPT-6 ASTRA: EINE MARKT-VALIDIERTE APP VERDIENT

5 PROMPTS

TOOL: CHATGPT + HIGGSFIELD

STAND 2026-09-17

Du baust eine App nach, die in einem anderen Land schon Geld verdient. Mit diesem Trick sind die Samwer-Brüder groß geworden. GPT-6 Astra wertet App-Store-Daten und Screenshots aus, schreibt das Reverse-Engineering (die Zerlegung, wie die fremde App genau funktioniert) und baut die App, die Higgsfield API (die Schnittstelle, über die deine App bei Higgsfield Bilder bestellt) liefert die Bilder. Am Ende steht der komplette Bau-Prompt für TrueYou, meinen Foto-Enhancer für Dating-Profile. Und vorher die Warnung vor dem Umsatzfehler, der dich um Faktor 40 danebenliegen lässt.

WAS DU BRAUCHST

ChatGPT Plus für 23 Euro im Monat. Darin ist GPT-6 Astra enthalten, das Modell, das die Analyse schreibt und die App baut.

Ein Konto bei [Higgsfield](#) mit API-Zugang. Die API hat keine Abo-Gebühr, sie rechnet über Credits pro erzeugtem Bild ab. Wie viel dich eine Generierung kostet, lässt du dir anzeigen, bevor du einen Preis für deine App festlegst.

Google Chrome auf dem Rechner, ein iPhone mit einem kostenlosen Apple Account und ein leerer Ordner mit dem Namen deiner App, in dem alle Screenshots landen.

01 EINE APP FINDEN, DIE SCHON GELD VERDIENT

Du suchst keine Idee. Du suchst einen Beweis. Zwei Seiten zeigen dir, welche Apps gerade Umsatz machen, beide ohne Anmeldung durchsuchbar:

AppTracker unter [appstoretracker.com](#), dazu die kostenlose Chrome-Erweiterung **AppTracker Watchlist for App Store Revenue**. Sie legt Umsatz- und Download-Schätzungen direkt über jede App-Store-Seite. Nach der Installation erscheint das Symbol oben rechts neben der Adressleiste.

AppMagic unter [appmagic.rocks/top-charts/apps](#). Kategorie wählen, nach Umsatz sortieren.

Schau auf beiden nach, die Schätzungen weichen voneinander ab. Filtere nach kürzlich gelaunchten Apps in anderen Ländern, die gut performen, also Apps, die es in Deutschland noch nicht oder kaum gibt.

Prüf dann drei Fragen: Verdient die App bereits Geld? Ist sie technisch einfach, also im Kern ein Upload, eine KI-Verarbeitung und ein Ergebnis? Gibt es in den Rezensionen einen Nutzungszweck, für den niemand eine eigene App gebaut hat?

DER FEHLER, DER DICH UM FAKTOR 40 DANEBENLIEGEN LÄSST

Lies bei jeder Umsatzzahl mit, **auf welchen Zeitraum sie sich bezieht**. In den Listen, die gerade herumgereicht werden, stehen häufig Gesamtumsatz seit Launch und Monatsumsatz nebeneinander, ohne dass dabei steht, was was ist. Wer eine kumulierte Zahl für den Monatsumsatz hält, verrechnet sich um ein Vielfaches und baut die App am Ende für einen Markt, den es so nicht gibt.

Der zweite Punkt: Manche Quellen nennen Netto-Zahlen, also nach dem Anteil von Apple und Google, andere Brutto. Stell nie zwei Zahlen nebeneinander, ohne zu wissen, welche davon welche ist.

01

ALLE DATEN AUS DEM APP STORE ZIEHEN

Abtippen musst du nichts. Astra liest später Preise, Probephase und Nutzungszwecke selbst aus den Screenshots.

Öffne die App-Store-Seite der App in Chrome über apps.apple.com/us/app/ plus App-Name. So siehst du den Store des Landes, in dem die App gut läuft.

Screenshots in dieser Reihenfolge und benenne die Dateien genau so:

01-store-kopf.png mit Name, Untertitel, Bewertung und Rang

02-store-screenshots-1.png bis **02-store-screenshots-6.png**, jedes Vorschaubild einzeln

03-store-beschreibung.png mit dem kompletten Beschreibungstext

04-store-inapp-kaeufer.png mit dem Abschnitt In-App-Käufe und allen Preisen

05-store-neuheiten.png mit der Versionshistorie, sie verrät, was zuletzt gebaut wurde

Jetzt die Rezensionen: Öffne die App im App Store auf dem iPhone, tippe auf die Bewertung und dann auf Alle anzeigen. Sortiere nach Hilfreichste und screenshotte die ersten 20 Rezensionen als **06-rezension-hilfreich-01.png** und weiter. Sortiere danach nach Neueste und screenshotte 20 weitere als **07-rezension-neu-01.png** und weiter.

TIPP

Chrome zeigt auf der Store-Seite nur einige Rezensionen. Die vollständige Liste mit Sortierung gibt es in der App-Store-App auf dem iPhone. Deshalb kommen die Rezensionen vom iPhone und der Rest aus Chrome. Die Screenshots vom iPhone schickst du per AirDrop in deinen Ordner am Rechner.

DIE APP LADEN UND JEDE SEITE SCREENSHOTTEN

Die Store-Seite zeigt, was die App verspricht. Erst die App selbst zeigt, wie der Weg vom Öffnen bis zur Bezahlung läuft. Dafür lädst du die App aus dem Land, in dem sie erfolgreich ist.

Öffne auf dem iPhone die Einstellungen, tipp oben auf deinen Namen, dann auf Medien & Käufe, dann auf Account anzeigen. Tipp auf Land/Region und auf Land oder Region ändern. Wähl das Land, akzeptier die Nutzungsbedingungen zweimal und gib eine Zahlungsmethode ein. Apple verlangt vorher, dass dein Guthaben aufgebraucht ist und keine laufenden Abos mehr auf dem Account liegen.

Installier die App und öffne sie. Mach ab dem ersten Bildschirm einen Screenshot von jeder Seite, auf dem iPhone mit Seitentaste und Lauter-Taste gleichzeitig. Auch von Zwischenmeldungen, Ladebildschirmen und Pop-ups.

Geh den kompletten Weg: Onboarding, Zugriffsabfragen, Upload, Auswahl, Verarbeitung, Ergebnis, Paywall, Einstellungen. Bei der Paywall screenshotten und auf Schließen tippen, nichts kaufen. Wenn es eine Probephase gibt, screenshote den Bildschirm mit dem Preis nach der Probe.

Benenne die Screenshots mit Nummer und Seite, also **10-app-onboarding-1.png**, **11-app-zugriff-fotos.png**, **12-app-upload.png**, **13-app-stilauswahl.png**, **14-app-verarbeitung.png**, **15-app-ergebnis.png**, **16-app-paywall.png**, **17-app-einstellungen.png**. Übertrag alles per AirDrop in deinen Ordner am Rechner.

TIPP

Falls du dein Hauptkonto nicht umstellen willst: Leg auf dem iPhone unter Einstellungen, dein Name, Abmelden einen zweiten Apple Account nur für den fremden Store an. Beim Erstellen wählst du das Zielland und als Zahlungsmethode Keine. Kostenlose Apps lässt du damit laden, dein Hauptaccount bleibt unverändert.

ASTRA DAS SEITENPROTOKOLL SCHREIBEN LASSEN

Im Reel sind es zwei Sätze, und genau diese zwei Sätze sind die Prompts. Damit sie etwas Brauchbares liefern, muss Astra vorher verstehen, was die App kann und wie der Nutzer durch sie läuft. Diesen Kontext baust du jetzt auf, und Astra macht die Arbeit dabei selbst.

Öffne ChatGPT, wähl GPT-6 Astra als Modell, starte einen neuen Chat und zieh die Store-Screenshots in das Eingabefeld. Pro Nachricht passen 20 Dateien. Bei mehr Screenshots schickst du mehrere Nachrichten hintereinander und schreibst dazu **Teil 1 von 3, warte mit der Analyse bis alle Teile da sind**. Mit der letzten Nachricht schickst du die App-Screenshots zusammen mit diesem Prompt.

PROMPT 01: ASTRA SCHREIBT DAS SEITENPROTOKOLL

Ich habe dir Screenshots einer erfolgreichen App hochgeladen, die ich nachbauen will. Dateien mit 01 bis 07 kommen aus dem App Store, Dateien ab 10 aus der installierten App, in der Reihenfolge, in der ich sie beim Benutzen gesehen habe.

Erstelle fuer jeden Screenshot ab Nummer 10 ein Seitenprotokoll als Tabelle mit diesen Spalten: Dateiname, Seite, Was der Nutzer hier sieht, Was passiert bei Tap oder Eingabe, Wohin geht es danach, Auffaelligkeiten. Schreibe Button-Texte in Anführungszeichen und genau so, wie sie auf dem Screenshot stehen. In die Spalte Auffaelligkeiten kommt alles, was einen Nutzer stoeren oder ueberraschen wuerde, zum Beispiel keine Tipps zur Fotoqualitaet, Werbung beim Warten, vorausgewaehltes Jahresabo.

Lies ausserdem aus den Store-Screenshots alle Preise mit Zeitraum und Probephase woertlich heraus, und aus den Rezensionen jeden Zweck, fuer den die Leute die App nutzen, mit der Anzahl der Nennungen.

Bestaetige mir zuerst, wie viele Screenshots du gesehen hast. Stelle mir danach zu jeder Seite hoechstens eine Frage, wo du dir nicht sicher bist, wohin ein Button fuehrt oder was eine Seite bedeutet. Ich antworte in normalen Saetzen, du traegst die Antworten ins Protokoll ein.

WENN DAS NICHT KLAPPT

Der Chat wird mit den vielen Bildern langsam oder bricht ab

Teil die Screenshots auf zwei Chats auf: Im ersten lässt du nur das Seitenprotokoll und die Rezensions-Auswertung schreiben, kopierst das Ergebnis als Text heraus und fügst es im zweiten Chat oben als Kontext ein. Die Prompts bleiben Wort für Wort dieselben, nur die Bilder bleiben im ersten Chat.

WENN DAS NICHT KLAPPT

Falls GPT-6 Astra in deinem normalen Chat nicht in der Modellauswahl steht: Bei ChatGPT Plus liegt Astra zum Start in ChatGPT Work und in Codex. Beide erreichst du über dieselbe Anmeldung, im Menü der ChatGPT-App oder unter chatgpt.com. Dort wählst du Astra aus und schickst denselben Prompt mit denselben Screenshots.

DEINE POSITIONIERUNG FESTLEGEN

Astra antwortet mit der Anzahl der Screenshots, dem Protokoll, den Preisen und den Nutzungszwecken, danach kommen ein paar Rückfragen. Du antwortest so, wie du es in der App gesehen hast, in normalen Sätzen.

Jetzt entscheidest du, für wen du baust und was du anders machst. Das ist die Stelle, an der aus einer Kopie ein eigenes Produkt wird. Füll die eckigen Klammern aus. Wenn du bei einem Punkt nichts weißt, schreib **weiss ich nicht**, dann schlägt Astra etwas vor.

VORLAGE: DEINE POSITIONIERUNG

Jetzt meine Positionierung. Merke dir das Seitenprotokoll, die Preise und die Nutzungszwecke von gerade eben, sie gehoeren zum Kontext fuer alles Weitere.

App-Name der Vorlage: [NAME DER APP]

Land, in dem sie gut laeuft: [LAND]

Geschaetzter Umsatz: [BETRAG und ZEITRAUM, so wie es auf deinem Screen steht]

Meine Positionierung:

Zielgruppe: [zum Beispiel Menschen zwischen 22 und 40, die auf Tinder, Hinge und Bumble aktiv sind und mehr Matches wollen]

Kernversprechen: [zum Beispiel In vier Minuten Profilfotos, die aussehen wie ein Profi-Shooting und trotzdem wie du]

Markt: [zum Beispiel DACH, Web-App]

Was ich weglassen: [zum Beispiel alle Video-Funktionen, Retusche alter Familienfotos]

Was ich neu dazunehmen: [zum Beispiel Vorschau jedes Fotos im Tinder-Layout, automatischer Vergleich der Fotos mit Bewertung]

Preisidee: [zum Beispiel 20 Dollar im Monat]

Name meiner App: [DEIN APP-NAME]

Wenn dir fuer die Analyse noch etwas fehlt, frag mich jetzt, bevor wir weitermachen.

ASTRA DIE APP REVERSE-ENGINEEREN LASSEN

Wenn Astra keine Fragen mehr hat, schickst du den ersten Prompt aus dem Reel. Der erste Satz ist der Auftrag wörtlich aus dem Video, der Rest legt fest, in welcher Reihenfolge und Tiefe Astra antwortet. Setz im ersten Satz deine eigene Nische ein.

Astra spielt dir danach eine komplette Analyse aus. Lies sie einmal durch und schreib in den Chat, was du anders siehst. Astra arbeitet es ein.

PROMPT 02: REVERSE-ENGINEERING, DER ERSTE PROMPT AUS DEM REEL

Reverse-Engineere mir diese App mit dem idealen Kunden, seinen Pain Points, der Marktlücke, zugeschnitten auf einen Foto-Enhancer fuer Dating-Apps.

Gehe in genau dieser Reihenfolge vor:

1. Funktionsinventar: Liste jede Funktion, die auf den Screenshots sichtbar ist, mit Screenshot-Nummer. Markiere, welche Funktion hinter der Paywall liegt.
2. Nutzerfluss: Beschreibe den Weg vom ersten Oeffnen bis zum bezahlten Ergebnis Schritt fuer Schritt, mit jedem Screen, jedem Button-Text und jeder Zwischenmeldung.
3. Monetarisierung: Preise, Probezeit, Abo-Logik, an welcher Stelle die Paywall kommt und mit welchem Versprechen.
4. Rezensionen-Auswertung: Werte alle Rezensionen-Screenshots aus. Nenne die fuenf haeufigsten Gruende fuer Begeisterung und die fuenf haeufigsten Beschwerden, jeweils mit Zitat. Sag klar, wofuer die Leute die App wirklich nutzen.
5. Idealer Kunde: Ein konkretes Profil mit Alter, Situation, Ausloeser fuer den Download, Erwartung an das Ergebnis und Zahlungsbereitschaft.
6. Pain Points: Die fuenf wichtigsten Schmerzpunkte dieses Kunden, sortiert nach Dringlichkeit, jeweils mit dem Beleg aus den Rezensionen.
7. Marktlücke: Was die App fuer diesen Kunden nicht loest und warum das eine Chance ist.
8. Zuschnitt auf meine Nische: Wie die App aussehen muss, damit sie nur fuer diesen einen Zweck gebaut ist. Welche Funktionen fliegen raus, welche kommen neu dazu, wie sich Onboarding, Sprache und Preis aendern.
9. Feature-Liste fuer den Nachbau: Eine priorisierte Liste mit Muss, Soll und Kann.

Schreibe alles auf Deutsch, konkret und ohne Fuellsaetze. Wo dir Informationen fehlen, sag das und frag nach, statt zu raten.

DEN HIGGSFIELD-ZUGANG HOLEN

Das ist die einzige Stelle im Guide, an der du selbst etwas in eine Datei tippst, denn Zugangsdaten gehören nie in einen Chat.

Wenn du noch kein Konto hast, legst du es bei [Higgsfield](https://higgsfield.ai) an. Den API-Zugang selbst findest du danach unter open.higgsfield.ai. Dort legst du einen API-Key an und bekommst eine Key-ID und ein Secret, also zwei Werte, nicht einen. Beides zeigt Higgsfield nur einmal an, lass das Fenster offen. Lade ein kleines Guthaben auf, die API zieht pro fertigem Bild Credits ab.

Leg in deinem Projektordner eine Datei mit dem Namen `.env` an (eine einfache Textdatei, in der Programme geheime Zugangsdaten nachschlagen) und trag die beiden Werte hinter die Gleichheitszeichen ein:

```
HF_API_KEY_ID=deine-key-id
```

```
HF_API_KEY_SECRET=dein-secret
```

Speichern, Datei schließen. Leg im selben Ordner eine Datei `.gitignore` an, in der die Zeile `.env` steht. Schick die `.env` niemals jemandem und lad sie nirgendwo hoch.

ZUGANGSDATEN GEHÖREN AUF DEN SERVER

Higgsfield schreibt in der eigenen Dokumentation: *API credentials grant access to your account and credits. Use them only in server-side code and never commit them to source control.* Auf Deutsch: Die Zugangsdaten laufen nur über deinen kleinen Server, nie im Code, den der Browser deiner Nutzer lädt. Wer die Seite aufmacht, kann sonst deinen Key auslesen und auf deine Rechnung generieren.

HIER IST DER PROMPT, DER DIE GESAMTE APP BAUT

Alles, was du bis hierhin gesammelt hast, steckt in diesem einen Prompt: der Aufbau, jeder Bildschirm, jeder Text wörtlich, die Farben, die Logik und die Anbindung an die Higgsfield-API. Du kopierst ihn in den Astra-Chat, schickst ihn ab und bekommst eine einzelne HTML-Datei zurück, die du per Doppelklick im Browser öffnest.

Das Einzige, was darin fehlt, ist dein eigener API-Key. Der steht nie im Prompt und nie im Code, sondern nur in deiner `.env` aus Schritt 7. Der Prompt sagt Astra, dass es die Zugangsdaten von dort holen soll, also trag deinen Key einmal in die Datei ein und der Rest läuft.

DER BAU-PROMPT: DIE KOMPLETTE APP IN EINEM STÜCK

Das hier ist der Prompt, mit dem TrueYou entstanden ist, unverändert. Kopier ihn in den Astra-Chat und schick ihn ab. Astra schreibt daraus eine einzelne HTML-Datei, die du im Browser öffnest.

Zwei Dinge kannst du vorher anpassen: den Namen der App und die sechs Stile. Alles andere lässt du so, die exakten Farbwerte, die drei Schriftrollen und die wörtlichen Bildschirmtexte sind der Grund, warum das Ergebnis nicht nach Baukasten aussieht.

Wenn du lieber deine eigene Nische baust, schick statt dessen den Satz aus dem Reel: **Mach mir daraus einen kompletten Bauplan in genau dieser Form, aber fuer meine Positionierung** und häng den Prompt unten als Vorlage an.

PROMPT 03: DER KOMPLETTE BAU-PROMPT (TRUEYOU)

Build "TrueYou" - a single-page mobile web prototype

1. PRODUCT SUMMARY

TrueYou is a mobile-app prototype for AI dating photos, rendered inside an iPhone 15 Pro frame on a near-black page so it can be screen-recorded. The user uploads 3-5 selfies, picks at least two photo "vibes", watches a cinematic generating screen with a giant mono percentage counter and ALL-CAPS status lines, and receives a 3-column grid of generated variants (3 per style), each with a picture-in-picture thumbnail of the source selfie and a score pill. From there the user can preview any photo inside pixel-faithful mockups of Tinder, Hinge and Bumble, and the app can run an automatic A/B test that pits the variants against each other with LIKE/NOPE stamps, an Elo rating, a live vote ticker and a confetti winner screen. Generation is simulated on a <canvas> (each style is a distinct color grade of the user's own selfie) so the whole thing works offline; an optional real-generation path calls an AI image-generation API with identity references and falls back to the simulation on any failure.

2. TECH CONSTRAINTS

One self-contained HTML file: inline CSS and vanilla JS, no framework, no bundler, no build step. Opening the file in Chrome must work.

Fonts from Google Fonts: Archivo Black (400), Inter (400, 500, 600, 700, 800), Space Mono (400, 700), with display=swap. Fallbacks: Archivo Black → -apple-system, sans-serif; Inter → -apple-system, BlinkMacSystemFont, sans-serif; Space Mono → ui-monospace, 'SF Mono', monospace.

No network needed for the default mode. Images stay in memory as data URLs; nothing is uploaded anywhere.

All screens live in the same DOM and are switched by toggling classes; no routing.

Document title: TrueYou - AI Dating Photos. lang="en". Viewport meta width=device-width, initial-scale=1.

Respect prefers-reduced-motion: reduce by setting every animation and transition duration to .01ms.

3. DESIGN SYSTEM

3.1 Color tokens

ink #000 - phone background, text on white buttons

paper #fff - primary text, primary buttons, borders of "selected" things

smoke #17181a - raised surfaces (selected cards, tab track, failed cards)

smoke2 #1f2023 - reserved secondary surface

line #2a2b2f - all 1px hairline borders and dividers

steel #8a8f98 - secondary text (subtitles, descriptions)

dim #55585f - tertiary text (step labels, captions, fineprint)

like #2bd672 - LIKE stamp, Tinder heart

nope #ff4458 - NOPE stamp, Tinder X, blinking record dot, error text

bumble yellow #ffc629 - Bumble accents, offline badge

page background #050505 - the page behind the phone

dashed placeholder color #3a3c41 - dashed borders of empty/upload frames and unchecked circles

Tinder super-like blue #35a8ff - star button in the Tinder mockup

Hinge paper #f7f4ef, Hinge ink #201a17, Hinge muted #98908a

dark-on-yellow #201a00 - text on bumble-yellow surfaces

3.2 Type roles - three roles, never mixed within one element

Display (Archivo Black, weight 400) - page headlines and the logo only. H1 hero: 54px, line-height .98, letter-spacing -1.5px. H2 screen title: 25px, line-height 1.08, letter-spacing -.4px, margin-bottom 10px. Logo "TrueYou." 18px, letter-spacing -.5px.

UI (Inter) - body copy, buttons, card titles. Subtitle .sub: 14.5px, weight 500, line-height 1.5, color steel, margin-bottom 20px. Primary button label: 16px, weight 800, letter-spacing -.2px. Ghost button label: 14px, weight 700. Card title: 15px weight 800 letter-spacing -.3px; card description 11.5px weight 500 steel. Tips: 13px weight 600.

Toast: 13.5px weight 800.

Data / mono (Space Mono, weight 700 unless stated) - every number, label, status, counter, ticker, stamp, tag. Eyebrow: 10px, letter-spacing 1.6px, color steel, preceded by a 6px round dot in nope-red that blinks (keyframe: 50% opacity .2, 1.3s ease infinite), inline-flex, gap 8px. Step label ("STEP 1/3"): 10.5px, letter-spacing 1.6px, color dim. Fineprint: 9.5px, letter-spacing 1.8px, uppercase, color dim, centered. Data strip numbers: 17px letter-spacing -.5px; data strip labels 9px letter-spacing 1.4px uppercase dim. Giant counter: 96px, letter-spacing -6px, line-height 1, font-variant-numeric: tabular-nums; its "%" is a <sup> at 30px, letter-spacing 0, vertical-align: 42px, color steel. Status line under the counter: 11px, letter-spacing 2.4px, uppercase. Score pill: 9.5px tabular. Style tag: 8px letter-spacing 1px uppercase. Photo counter bar: 10px letter-spacing 1.6px uppercase dim. Mini frame captions: 9px letter-spacing 1.4px uppercase dim.

3.3 Stage and iPhone 15 Pro frame

html, body height 100%; body background #050505, font-family Inter, color white, overflow: hidden.

Stage: height: 100vh; display: grid; place-items: center.

Phone: position: relative; width: 393px; height: 852px; max-height: 96vh; aspect-ratio: 393/852; background: #000; border-radius: 54px; border: 1px solid #222; box-shadow: 0 0 10px #0c0c0c, 0 0 12px #262626, 0 40px 90px rgba(0,0,0,.85); overflow: hidden; display: flex; flex-direction: column.

Dynamic Island: absolute, top 12px, centered, 118x34px, background #000, radius 20px, box-shadow: 0 0 1px #161616, z-index 50.

Status bar: absolute top 0, full width, height 54px, z-index 48, flex space-between, padding: 14px 32px 0, 15px weight 600 (Inter). Left: the text 9:41 (pinned, never a live clock). Right: two inline SVGs 7px apart - a 4-bar cellular signal (18x12, bars 3px wide at x=0/5/10/15 with heights 5/7/9.5/12, radius 1, currentColor) and a battery (25x12: outer rect 21x11 at .5,.5 radius 3.5 stroke currentColor opacity .4 no fill; inner rect 16x8 at 2,2 radius 2 filled; nub 2x4 at 23,4 radius 1 opacity .4).

Home indicator: absolute bottom 8px, centered, 140x5px, radius 3px, white at opacity .9, z-index 50.

Camera flash overlay: absolute inset 0, background white, opacity 0, pointer-events none, z-index 55. When triggered it plays camflash .65s ease: 0% opacity 0 → 12% opacity 1 → 100% opacity 0.

Small screens (max-width: 500px): stage becomes display: block; the phone becomes width: 100vw; height: 100vh; max-height: none; border-radius: 0; border: none; box-shadow: none; aspect-ratio: auto; island and home indicator hidden.

3.4 Film grain

A pseudo-element covering the whole phone (position: absolute; inset: 0; z-index: 46; pointer-events: none) with opacity: .045 and mix-blend-mode: overlay, whose background image is an inline SVG data URI: a 140x140 tile containing an feTurbulence type="fractalNoise" baseFrequency=".9" numOctaves="2" filter applied to a full-tile <rect> at opacity=".5". It sits above all screens but below the status bar, island, flash and home indicator.

3.5 Screens and transitions

Every screen is position: absolute; inset: 0; padding: 74px 22px 34px; display: flex; flex-direction: column.

Inactive: opacity: 0; transform: translateX(36px); pointer-events: none. Active: opacity 1, translateX(0), pointer-events auto. Transition: opacity .34s ease, transform .42s cubic-bezier(.22,1,.36,1).

When leaving, the outgoing screen gets an "exit-left" state (translateX(-36px)) that is removed 500ms later, so screens slide right-to-left going forward.

Top nav row (.nav): flex, space-between, align center, margin-bottom 16px, min-height 34px. Back button: 34x34 circle, border: 1px solid line, transparent, white left-chevron at 20px, padding-bottom: 3px.

CTA area at the bottom of a screen: margin-top: auto; display: flex; flex-direction:

column; gap: 8px; padding-top: 14px.

3.6 Shared components

Primary button: full width, padding: 18px, no border, radius 999px, white background, black text (Inter 16/800). :active scales to .965 (transition transform .12s). :disabled opacity .25. Optional glow variant adds box-shadow: 0 0 44px rgba(255,255,255,.15). Ghost button: full width, padding 13px, transparent, steel text, Inter 14/700, radius 999px.

Toast: absolute, bottom 92px, centered, white background, black text, padding: 12px 20px, radius 999px, box-shadow: 0 12px 34px rgba(0,0,0,.55), z-index 60, white-space: nowrap. Hidden state: opacity 0 and translateY(20px); shown: opacity 1, translateY(0); transition all .35s cubic-bezier(.22,1,.36,1). Auto-hides after 2200ms; a new toast resets the timer.

Data strip: flex row, border-top and border-bottom 1px line, cells flex: 1; padding: 12px 4px; text-align: center separated by 1px line right borders (none on the last).

Each cell: a mono number on top (17px) and a mono uppercase label below (9px dim).

Stamp (the LIKE / NOPE / WINNER mark): Space Mono 700, uppercase, letter-spaced, colored text with a matching 1.5px solid border, background rgba(0,0,0,.7), rotated -10deg. Mini size: 9px, letter-spacing 1px, padding: 2px 5px, radius 5px. Large size: 28px, letter-spacing 4px, padding: 6px 14px, radius 10px, border 3px. LIKE and WINNER use like-green; NOPE uses nope-red.

Pop-in keyframe (pop): from { transform: scale(.7); opacity: 0 } - used for upload cells (.35s) and result cards (.4s), easing cubic-bezier(.22,1.4,.36,1).

Offline badge (only in the offline build): absolute top 62px right 18px, z-index 49, mono 9px letter-spacing 1.5px, text #201a00 on bumble-yellow, padding: 4px 8px, radius 6px, text OFFLINE DEMO.

4. SCREENS (in order) - copy is verbatim

Screen 1 - Onboarding

Top row (flex, space-between): logo TrueYou. (display 18px) on the left; on the right an eyebrow with blinking red dot: AI PHOTOS - SEEN IN-APP.

Hero block (margin: auto 0, column, gap 16px): H1 (display 54px): 10x your / matches. (line break before "matches"; the "10x your" uses a non-breaking space). Subtitle: AI photos that look exactly like you - on your best day. Then see them right inside Tinder before anyone else does.

Two mini frames side by side (flex, gap 14px, margin-top 8px), each 86x114px radius 16px: a "raw" frame with 1.5px dashed #3a3c41, rotated -5deg, containing a selfie emoji at 24px in dim color, caption below (mono 9px, 22px under the frame) your selfie; between them an arrow at 17px, dim, weight 900; a "result" frame with 2px solid #fff, background smoke, rotated 4deg, box-shadow: 0 14px 40px rgba(0,0,0,.6), in its top-left corner (6px, 6px) a 24x32 pip rectangle radius 7px 1.5px solid #fff background #26272b, a mini LIKE stamp hanging off the bottom-right (bottom 8px, right -12px), caption your best shot.

Data strip with three cells: 10x / matches; 4 min / to photos; 3 apps / previewed.

Margin-bottom 4px.

CTA area: primary button Generate my Tinder photos → goes to Upload. Fineprint below: Tinder - Hinge - Bumble.

Screen 2 - Upload

Nav: back button (→ Onboarding) and step label STEP 1/3.

H2: Upload 3-5 selfies.

Subtitle: Clear face, different angles, good light. The AI learns your unique features - your uploads stay private.

Upload grid: 3 columns, gap 9px. Each filled cell: aspect 3/4, radius 14px, 1px solid line, image object-fit: cover, pop-in animation, and a remove button top-right (6px/6px): 22px circle, background rgba(0,0,0,.75), white cross at 11px. The last cell is the "add" cell: 1.5px dashed #3a3c41, transparent, column-centered, gap 4px, a fullwidth plus at 24px white weight 300 and the mono caption add photo (8.5px, letter-

spacing 1.4px, uppercase). It is a <label> for a hidden <input type="file" accept="image/*" multiple>. The add cell disappears once 5 photos are present. Tips block (margin-top 18px, column, gap 8px, 13px weight 600 steel; the negative tip in dim): Chest-up, looking at camera / Neutral and smiling shots / No group photos or heavy filters.

CTA: one primary button. With fewer than 3 photos it is disabled and reads Add 3 more photos, Add 2 more photos, or Add 1 more photo. With 3-5 photos it is enabled and reads Continue → Styles.

Screen 3 - Styles

Nav: back (→ Upload), STEP 2/3.

H2: Pick your vibes.

Subtitle: We shoot you in every style you pick - then preview each shot right in Tinder, Hinge and Bumble.

Vertical list of six cards (column, gap 8px, overflow-y: auto). Card: flex row, align center, gap 14px, radius 16px, padding: 13px 16px, transparent, 1px solid line, transitions border-color .18s, background .18s, transform .12s, :active scale .98. Contents: emoji (22px, 30px wide, centered), text block (h3 + p), and a 24px check circle on the right (1.5px solid #3a3c41, containing a check at 12px weight 900 in transparent color). Selected state: border white, background smoke, check circle filled white with black check.

The six styles: Casual - Golden hour, relaxed, approachable. Outdoor - Sunset trail, adventurous energy. Travel - Rooftop abroad, vacation vibe. Editorial - B&W studio, magazine quality. Athletic - Moody gym light, disciplined. Night out - Neon bokeh, cinematic city.

CTA: primary button, disabled with fewer than 2 selected reading Pick at least 2 styles; otherwise Generate N photos where N = selected styles x 3. Click → Generating.

Screen 4 - Generating

Content vertically centered and text-centered (column, align center, gap 10px):

Eyebrow in white (dot still blinks red): SHOOTING

Giant counter: the number (starts at 0) followed by the superscript %.

Status line (mono 11px, letter-spacing 2.4px), initial text UPLOADING YOUR PHOTOS; changes over time through this exact sequence, each swap fading out (opacity 0), replacing text after 250ms and fading back in (transition .3s): UPLOADING YOUR PHOTOS / MAPPING YOUR FACE / LEARNING YOUR ANGLES / SETTING UP THE SHOOT / SHOOTING YOUR SETS / FIXING THE LIGHT / KEEPING IT REAL / RENDERING PHOTOS / DONE (only at 100%).

Caption (12px weight 600 dim): usually takes about a minute

Progress bar: 190x2px, background #222, margin-top 20px, white fill whose width transitions .5s ease.

At 100% the camera flash fires, then the app moves to Results.

Screen 5 - Results

Nav: step label STEP 3/3 (no back button).

H2 initially Your photos are ready, but once the grid is rendered it is replaced by N photos ready. If the real-generation path produced failures: N photos ready - M failed. Subtitle: Practically indistinguishable from a pro shoot - and it's still you. Tap any photo to see it live in the apps.

Grid: 3 equal columns, gap 7px, overflow-y: auto, padding-bottom 8px. Card: aspect 3/4, radius 13px, 1px solid line, pointer cursor, pop-in .4s with animation-fill-mode: backwards and a delay of index x 55ms, :active scale .96. Overlays: PiP of the source selfie (top 6px, left 6px, 28x38px, radius 7px, 1.5px solid #fff, object-fit: cover, box-shadow: 0 4px 12px rgba(0,0,0,.6), background #222); score pill (top 6px, right 6px, white background, black text, padding: 3px 7px, radius 999px, mono 9.5px, one decimal); style tag (bottom 6px, left 6px, rgba(0,0,0,.75) with backdrop-filter: blur(8px), padding: 3px 7px, radius 999px, mono 8px uppercase).

Failed card variant (real path only): column-centered on smoke background, padding: 8px, gap 7px, static style tag, a mono 8px nope-red reason line, and a small pill button Retry. While retrying the reason line reads retrying... and the button is hidden.

CTA: primary button with glow See them in Tinder → opens the Preview on the first photo. Ghost button Download all photos. Tapping any card opens the Preview on that photo.

Screen 6 - In-app preview

padding-bottom: 26px. Nav row (gap 10px): back button (→ Results) and a segmented control filling the rest: track background smoke, radius 999px, padding: 3px; three buttons flex: 1; padding: 8px 0, radius 999px, Inter 12.5/800, steel text, transparent; the active one is white with black text. Labels: Tinder (default), Hinge, Bumble. Stage: flex: 1; position: relative; min-height: 0. Two invisible tap zones over the stage, each 34% wide, spanning the full height (left: -8px / right: -8px, z-index 20) for previous/next. Left/Right arrow keys do the same while this screen is active. Navigation wraps around. The mockup fills the stage: position: absolute; inset: 0; border-radius: 24px; overflow: hidden; border: 1px solid line, entering with mockIn .3s ease (from { opacity: 0; transform: scale(.97) }). Profile used everywhere: name Alex, age 25, distance 2 km away. Tinder: background #111; the photo covers the card; 4 story segments across the top (top 10px, left/right 10px, gap 4px, each flex: 1; height: 3px; border-radius: 2px; background: rgba(255,255,255,.35), the first solid white); bottom gradient 46% tall linear-gradient(to top, rgba(0,0,0,.88), transparent); info block left/right 16px, bottom 92px: Alex at 27px/800 letter-spacing -.5px followed by 25 at 24px/400 with 6px left margin in rgba(255,255,255,.85); below it 2 km away at 13px/500 rgba(255,255,255,.75), margin-top 4px; action row centered at bottom 20px, gap 16px, circles with background: rgba(20,20,20,.55); backdrop-filter: blur(6px), weight 800: cross 54px in nope-red with 1.5px nope border at 20px, star 42px in #35a8ff with matching border at 15px, heart 54px in like-green with matching border at 19px. Hinge (light UI): background #f7f4ef, color #201a17, column, padding: 14px. Name Alex in Georgia, 'Times New Roman', serif 24px/700 letter-spacing -.3px, margin: 2px 4px 10px. Photo panel flex: 1, radius 14px, image cover, with a 48px white circle bottom-right (12px/12px) containing a heart at 20px in #201a17, box-shadow: 0 6px 18px rgba(0,0,0,.25). Below, a prompt card: white, radius 14px, padding: 14px 16px, margin-top 10px, box-shadow: 0 2px 10px rgba(0,0,0,.05); a label (10.5px/700, letter-spacing .6px, uppercase, #98908a) and an answer (Georgia 16.5px, margin-top 5px, line-height 1.35). The prompt pair is picked by photo index modulo 4 from: My simple pleasures / Golden hour walks and a proper flat white; The way to win me over / Show up with snacks and a good playlist; A life goal of mine / Shoot a photo on every continent; I go crazy for / Rooftops at sunset and window seats. Bumble: background #111, photo cover; wordmark bumble top 12px left 14px, 15px/800 letter-spacing -.3px in bumble-yellow with text-shadow: 0 1px 8px rgba(0,0,0,.5); bottom gradient 42% linear-gradient(to top, rgba(0,0,0,.85), transparent); info left 16px bottom 26px max-width 60%: Alex, 25 26px/800 letter-spacing -.5px followed by a yellow check at 15px, margin-left 7px, vertical-align: 3px; below, 12.5px/500 rgba(255,255,255,.75): <Style name> - 2 km away. Actions stacked at right 14px bottom 22px, gap 12px, 50px circles at 19px/800: cross on rgba(255,255,255,.92) in #777, heart on bumble-yellow in #201a00. Bottom bar (margin-top 12px, centered, gap 16px, mono 10px uppercase dim): chevrons at 15px steel around the counter text PHOTO 1/12 - CASUAL. Absolutely on the right, a 36px circular download button (1px solid line, background smoke, white arrow at 15px, :active scale .92) that saves the current photo and shows the toast Saved to Downloads.

Screen 7 - Automatic A/B test

This screen starts on its own: after the Results grid has finished popping in plus a 2.4 s dwell, the app auto-advances here. Tapping a result card or See them in Tinder before that cancels the auto-advance and opens the Preview; returning to Results re-arms the timer.

No nav. Top: eyebrow with blinking red dot LIVE A/B TEST; below it the H2 Finding your best shot; subtitle Your photos are being matched against real swipe data. No taps needed.

Ticker (mono 10px, letter-spacing 1.6px, dim, uppercase, centered): MATCHING VS 250,412 HISTORICAL SWIPES. The number counts up live: it starts at 250,412 and increases by a random 300-900 on every vote, formatted with thousands separators.

Arena: two photo cards side by side (gap 10px, each aspect 3/4, radius 16px, 1px solid line, image cover, style tag bottom-left as on Results). Each vote stamps one card LIKE and the other NOPE (large stamp, centered on the card, entering with from { transform: scale(1.6) rotate(-10deg); opacity: 0 } to scale(1) rotate(-10deg) over .25s with the overshoot easing). The losing card also dims to opacity .45. Below each card a mono 10.5px line shows that photo's current Elo rating with tabular numerals.

Live percentage bar (full width, 2px tall, background #222, margin-top 14px): the white fill shows the current leader's win rate as a percentage; a mono 9.5px caption to its right reads e.g. LEADER 71%.

Vote counter under the bar, mono 9px dim uppercase: VOTE 7/16.

Finale: after the last regular vote, the two highest-rated photos meet in one final match; the caption above the arena briefly changes to mono FINAL - TOP 2 BY ELO. After that match, the flash fires and the app moves to the Winner screen.

Screen 8 - Winner

Eyebrow RESULT (dot blinks). H2 Your best shot. Subtitle This is the photo that gets the most likes. Set it as your main and watch what happens.

Hero photo: the winner, centered, width 58% of the screen, aspect 3/4, radius 18px, 2px solid #fff, box-shadow: 0 24px 60px rgba(0,0,0,.7), rotated 2deg, with the source-selfie PiP top-left and a large WINNER stamp over its bottom-right corner (bottom -10px, right -14px).

Confetti: 90 rectangles (4x8px, radius 1px) in white, like-green, nope-red and bumble-yellow, spawned along the top edge with random x, falling with slight horizontal drift and rotation over 2.6-3.4 s, starting when the screen becomes active, playing once.

Data strip with three cells: +38% / match uplift; 1612 / elo score (the winner's actual rating); 94% / confidence. The uplift is $\text{round}((\text{winnerElo} - 1500) / 1500 \times 500)$ clamped to 12-64, prefixed with + and %; confidence is $\text{round}(100 \times \text{winnerWins} / \text{winnerMatches})$ clamped to 80-98.

CTA: primary button with glow Set as my Tinder photo → opens the Preview on the winning photo with the Tinder tab active. Ghost button Download all photos.

5. LOGIC AND BEHAVIOUR

5.1 State and navigation

State: selfies[] (data URLs), styles (ordered set of style keys casual | outdoor | travel | formal | gym | night), photos[] (objects { id, url, src, style, score } where src is the selfie the variant was made from), preview index, preview app (tinder | hinge | bumble), and A/B state (ratings, match list, vote index). Navigation is a single go(screenId) that removes active from the current screen, adds the exit-left state for 500ms, and activates the target.

5.2 Upload

On file selection, take at most 5 minus current count files, read each as a data URL and append. Reset the input value so the same file can be chosen again. Re-render the grid on every change; remove buttons splice by index. Button label and disabled state per Screen 2.

5.3 Style selection

Clicking a card toggles its key in the set and its selected class; the CTA updates per Screen 3. Order of generation follows the order in which styles were selected.

5.4 Generating progress

On entering the screen: progress = 0, message index = 0. A timer every 350ms adds $(\text{random}() \times 2.2 + 0.5)$ to progress, multiplied by 0.25 once progress > 80, capped at 94. The counter shows $\text{floor}(\text{progress})$; the bar width is progress%.

Message index = $\text{min}(\text{floor}(\text{progress} / 12.5), 7)$; when it changes, swap text with the fade described above.

When the photos are ready, stop that timer and run a finishing timer every 60ms adding 4 until 100. At 100: status text DONE, fire the flash, and 620ms later navigate to Results.

Simulation timing: build the mock photos immediately, then wait 6.5-10 s ($6500 + \text{random}() \times 3500$ ms) before finishing, so the counter visibly climbs and stalls in the 80s before snapping to 100.

5.5 Simulated generation (canvas)

For each selected style (in selection order) produce 3 variants. Variant v (0-2) of the i -th photo uses source selfie selfies[($i + v$) mod selfies.length]. Render onto a 600x800 canvas:

Zoom factor $1 + v \times 0.14$. Cover-scale the image ($\max(600/w, 800/h) \times \text{zoom}$), draw it centered horizontally at $x = (600 - \text{sw}) / 2$ and vertically at $y = (800 - \text{sh}) / 2.4$ so faces stay in frame at higher zoom.

Before drawing, set the canvas 2D filter to the style's filter string plus a per-variant brightness($1 + (v - 1) \times 0.05$):

casual (warm): brightness(1.08) contrast(1.05) saturate(1.15) sepia(.18), vignette alpha .25

outdoor (teal-orange): brightness(1.1) contrast(1.12) saturate(1.25) hue-rotate(-6deg), vignette .15

travel (vivid film): brightness(1.12) contrast(1.08) saturate(1.45), vignette .2, plus grain

formal (B&W): grayscale(1) contrast(1.35) brightness(1.02), vignette .45

gym (moody): brightness(.92) contrast(1.3) saturate(.85) hue-rotate(8deg), vignette .5

night (cool neon): brightness(.88) contrast(1.2) saturate(1.3) hue-rotate(-14deg), vignette .55

Reset the filter, then paint a radial vignette centered at (300, 400) from inner radius $800/3$ (transparent) to outer radius $800/1.1$ (rgba(0,0,0,<vignette alpha>)).

If the style has grain, scatter 4000 1x1px white points at random positions with alpha $\text{random}() \times 0.05$.

Export as JPEG at quality 0.9 (data URL). Each photo gets a score $(6.8 + \text{random}() \times 2.7)$.toFixed(1).

5.6 Preview

Opening the preview sets the index, renders the mockup for the active app, and navigates. Changing the tab re-renders in place. Prev/next zones and arrow keys step with wrap-around (skipping failed photos if any). The counter text updates on every render.

5.7 Elo A/B simulation

Every photo starts at rating 1500; K-factor 32. Each photo has a hidden strength = its displayed score.

Schedule: 16 regular matches built from a shuffled round-robin so every photo appears at least twice and no pair repeats until all pairs are used, followed by 1 final between the two highest-rated photos at that moment.

Cadence per match: cards slide in (mockIn keyframe), 700ms later the vote lands: pick the winner with probability $1 / (1 + 10^{((s_B - s_A) / 1.2)})$ where s is the hidden strength, stamp LIKE/NOPE, update both Elo ratings ($\text{expected} = 1 / (1 + 10^{((r_B - r_A) / 400)})$, $r \pm= K \times (\text{result} - \text{expected})$), tick the ticker (+300-900), update the percentage bar and VOTE $n/16$ counter; 650ms later the next match begins. Total run about 24 s.

Leader win rate = wins / matches of the currently highest-rated photo, shown as an integer percent. Winner = the photo with the highest rating after the final; ties go to the one with more wins.

5.8 Downloads

Download all photos triggers one download per photo, 400ms apart, named trueyou-01-casual.jpg, trueyou-02-casual.jpg, ... (two-digit index, style key), via a temporary anchor with download set to the data URL, then shows the toast Saved N photos to

Downloads. Single-photo download from the preview uses the same naming and the toast Saved to Downloads. Remote URLs (real path) are fetched to a blob first and the object URL revoked 5 s later.

5.9 Optional real generation (AI image API with identity reference)

Implement behind a single feature switch; the default build never calls the network. On first open of the "live" build, ask for an API key with a native prompt() (message: TrueYou needs an API key. / Paste it as KEY_ID:SECRET. / Cancel = offline demo mode.), validate its shape, retry up to 3 times with an alert on bad format, store it in localStorage, and fall back to the simulation when cancelled. Never embed a key in the file.

Flow: upload each selfie to the API's presigned upload endpoint (request an upload URL with the MIME type, PUT the bytes with the returned headers, keep the returned public URL). Then for every (style, variant) pair submit one image-to-image / identity-reference generation request with: the prompt = style prompt + variant framing hint + enhancement suffix (Appendix A), up to 4 selfie URLs as references, aspect ratio 3:4, one image at 2K resolution, JPEG output. Poll the job's status URL every 2 s for the first minute, then every 5 s, up to 6 minutes; succeed on completed/succeeded, fail on failed/nsfw/canceled.

Concurrency: 2 worker loops over the task list; submissions staggered at least 1200 ms apart; retry POSTs up to 4 times on network errors, 429 and 5xx with backoff 2 s / 5 s / 12 s. A moderation (nsfw) failure is retried once.

Per-variant failures do not abort the run: they become failed cards with a short reason (not enough credits, flagged by moderation, timed out, server error (5xx), network error, or the first 60 chars of the error) and a Retry button that regenerates only that variant (toast Variant regenerated on success). If every variant fails, show toast API unavailable - demo mode and fall back to the simulation (with a 4 s wait so the counter still animates).

Extract the result URL defensively from images[], results[] or outputs[] entries that may be strings, {url}, {image_url} or {image:{url}}.

The Generating screen behaves identically in both modes: the progress animation is cosmetic and finishes when the job completes.

6. ACCEPTANCE CRITERIA

The page shows a 393x852 (max 96vh) black phone with 54px radius, the three-ring shadow, a 118x34 island, the pinned 9:41 status bar, a 140x5 home indicator and a visible but subtle noise grain over the whole phone; the page behind it is #050505.

Onboarding reads exactly 10x your matches., shows the three-cell data strip 10X / MATCHES, 4 MIN / TO PHOTOS, 3 APPS / PREVIEWED, and the button Generate my Tinder photos with TINDER - HINGE - BUMBLE beneath.

With 0/1/2 photos the upload button is at 25% opacity and reads Add 3 more photos / Add 2 more photos / Add 1 more photo; at 3 it becomes Continue; at 5 the dashed add cell disappears; removing a photo re-enables it.

The styles button reads Pick at least 2 styles until two cards are selected, then Generate 6 photos (and Generate 9 photos for three), and selected cards turn to a white border on #17181a with a filled check.

The generating screen shows a 96px Space Mono counter with a small grey %, cycles through the eight ALL-CAPS status lines in order, stalls in the 80s-94, then snaps to 100, shows DONE, flashes white, and lands on the results grid.

The results grid has exactly 3 cards per selected style, each visibly graded differently per style, each with a 28x38 white-bordered PiP of the source selfie top-left, a white score pill (6.8-9.5) top-right and an uppercase style tag bottom-left; the heading reads N photos ready.

Tapping a card opens the preview showing that photo inside the Tinder mockup; the Hinge tab switches to a cream light UI with a serif name and prompt card; the Bumble tab shows the yellow bumble wordmark; the bar reads PHOTO n/N - STYLE; tapping the right third advances and wraps.

Left alone on the results screen, the A/B test starts by itself, stamps LIKE/NOPE on alternating pairs about once a second while the MATCHING VS 250,412 HISTORICAL SWIPES

number climbs, ends with a FINAL - TOP 2 BY ELO match, and the winner screen shows the top-rated photo with a WINNER stamp, confetti, the three stats and the buttons Set as my Tinder photo and Download all photos.

APPENDIX A - prompts for the real-generation path

Compose each request as STYLE_PROMPT + VARIANT_HINT + ENHANCE.

ENHANCE (appended to every prompt):

SUBJECT: the exact same person from the reference selfies - 100% face accuracy, instantly recognizable - and the SAME AGE as in the references. Do NOT age the subject. GROOMING: healthy even skin with real pores, groomed brows, a stylish textured haircut. REALISM: an authentic photograph, never AI-looking - no waxy over-smoothed skin, no plastic retouching, no oversaturated HDR glow, no uncanny perfect symmetry. Keep small real-world imperfections: stray hairs, fabric wrinkles, natural skin texture, true-to-life color and contrast. Keep the subject's real body shape as it appears in the references.

VARIANT_HINT by variant index:

0 - FRAMING: classic medium shot, natural confident expression.

1 - FRAMING: tighter chest-up crop, subject looking straight into the lens, confident half-smile.

2 - FRAMING: three-quarter angle from slightly below, subject glancing off-camera, more relaxed pose in a different spot of the same scene.

STYLE_PROMPT by style key:

casual - A straight-on medium shot of the person from the reference selfies sitting relaxed on sun-warmed concrete steps of a quiet European side street at golden hour.

WARDROBE: open sand-colored linen overshirt over a fitted white ribbed cotton tank, relaxed washed-blue denim, clean minimal leather sneakers, thin silver chain. LIGHT: hard low sun from camera-left rakes across the scene, long crisp shadows on the steps, warm rim glow in the hair. CAMERA: 35mm street-photography lens, f/2, ISO 200, eye-level framing, moderate background blur. PALETTE: muted earth tones. Candid lifestyle energy, as if shot on a flagship smartphone, subtle digital grain.

outdoor - A medium-full shot of the person from the reference selfies standing on a rocky coastal trail at sunset, wind moving the hair, confident gaze just past the lens. WARDROBE: fitted dark-olive technical jacket over a white tee, tapered hiking trousers. LIGHT: low golden sun as key from behind-left, natural bounce fill, glowing atmospheric haze over the ridgeline. CAMERA: 50mm lens, f/2.8, ISO 100, slightly low hero angle. PALETTE: cinematic teal-and-orange grade, crisp contrast. Adventurous, effortless energy.

travel - A candid three-quarter shot of the person from the reference selfies leaning on the stone parapet of a rooftop terrace above a Mediterranean old town at late afternoon, mid-laugh, completely off-guard. WARDROBE: open cream linen shirt with sleeves rolled, tailored light trousers, vintage sunglasses hooked in the collar. LIGHT: soft warm directional sun, sea glitter in the background bokeh. CAMERA: 35mm film look, f/2.8, visible fine grain, slight halation on highlights. PALETTE: vivid Kodak-style warmth. Vacation spontaneity, magazine travel-story feel.

formal - A black-and-white editorial studio portrait of the person from the reference selfies, seated on a simple stool against a deep charcoal seamless backdrop, torso angled, eyes locked into the lens. WARDROBE: tailored charcoal blazer over a fine merino crew-neck, no tie, minimal watch. LIGHT: hard rim light carving the jawline from behind-right, large soft key at 45 degrees, deep controlled shadows. CAMERA: medium-format look, 80mm, f/5.6, razor-sharp eyes, subtle film grain. High-contrast monochrome, magazine-cover composition with generous negative space.

gym - An athletic portrait of the person from the reference selfies resting between sets in a modern concrete gym, forearms on a bar, chalk dust hanging in the air, focused expression. WARDROBE: fitted dark training tee, athletic shorts. LIGHT: moody directional topline, cool practical lamps in the background bokeh. CAMERA: 35mm, f/2, ISO 800, slight low angle. PALETTE: cool desaturated grade with warm skin tones

preserved. Disciplined, quietly intense sports-editorial mood.
night - A paparazzi-style night shot of the person from the reference selfies stepping out of a doorway onto a city sidewalk, caught mid-stride. WARDROBE: boxy black overshirt over a white tee, dark tapered trousers, chunky clean sneakers, thin chain. LIGHT: strong direct frontal flash - hard shadow thrown on the wall behind, glossy highlights on fabric and skin, wet asphalt reflecting neon signs beyond the flash falloff. CAMERA: 28mm, f/4, 1/125s, flash-balanced against magenta-and-blue city ambience. High contrast, raw candid editorial energy.

TIPP

Leg dir den Prompt als Datei `ONE_PROMPT.md` in deinen Projektordner. Damit lässt du die App jederzeit neu bauen, auch mit einem anderen Werkzeug, ohne die Analyse zu wiederholen. Ein Satz an Astra reicht dafür:
Speichere den kompletten Bau-Prompt als ONE_PROMPT.md zum Download.

01 ABNEHMEN UND STARTEN

Astra liefert dir eine HTML-Datei. Doppelklick, sie öffnet sich im Browser. Bevor du sie jemandem zeigst, gehst du die Abnahmekriterien aus Abschnitt 6 selbst durch. Dieser Prompt lässt dir die Beweise dafür liefern.

PROMPT 04: ABNAHME MIT BEWEISEN

Gehe jede Zeile der Acceptance Criteria aus Abschnitt 6 einzeln durch. Starte die App, führe den Klickweg fuer jedes Kriterium selbst aus und mache nach jedem Schritt einen Screenshot. Liefere mir eine Tabelle mit Kriterium, Ergebnis (erfüllt oder nicht erfüllt) und dem Dateinamen des Screenshots. Bei nicht erfüllt: behebe den Fehler, teste erneut und aktualisiere die Tabelle.

Zum Schluss: Bestaetige, dass in keiner Datei ausser .env Zugangsdaten stehen, dass .env in der .gitignore steht, und dass die App ohne Internetverbindung mit simulierten Bildern durchläuft.

WENN DAS NICHT KLAPPT

Die App gibt keine Bilder zurück

Lass dir den Status ausgeben. Die Higgsfield-API antwortet zeitversetzt: Du bekommst zuerst eine Auftragsnummer, und der Status geht danach auf completed, failed, nsfw oder canceled. Steht er dauerhaft auf failed, stimmen meistens die Zugangsdaten in der `.env` nicht.

BEVOR DU IN DIE APP STORES WILLST

Die Web-App darfst du sofort online stellen. Für den App Store und Google Play gelten eigene Regeln, und genau an ihnen scheitern KI-Apps am häufigsten. Drei davon betreffen dich direkt.

Apple verlangt in Richtlinie 5.1.2(i): *You must clearly disclose where personal data will be shared with third parties, including with third-party AI, and obtain explicit permission before doing so.* Das Selfie deines Nutzers geht an die

Higgsfield-API, also an einen Dritten. Sag das auf dem Bildschirm, bevor hochgeladen wird, und hol dir die Zustimmung. Nicht versteckt in der Datenschutzerklärung.

Apple verlangt in Richtlinie 4.2: *If your app is not particularly useful, unique, or “app-like,” it doesn’t belong on the App Store*. Genau deshalb sind die Stil-Voreinstellungen, das Onboarding, die In-App-Vorschau und der A/B-Test im Bau-Prompt oben drin. Sie sind das, was aus einer Anbindung eine App macht.

Google Play macht dich als Entwickler dafür verantwortlich, was deine App erzeugt, und verweist dabei auf die eigenen Richtlinien zu unangemessenen Inhalten. Teste vor dem Einreichen selbst, ob sich deiner App etwas Anstößiges entlocken lässt. Ein Melde-Button an jedem erzeugten Ergebnis ist die billigste Absicherung und kostet dich zehn Minuten Einbau.

DER EHRliche TEIL

Der Bau-Prompt oben gibt dir den funktionierenden Kern: Upload, Generierung, Ergebnis, Vorschau, angedeutete Bezahlschranke. Das läuft im Browser und das kannst du herzeigen.

Was nicht dabei ist: echte Bezahlung, denn Stripe oder RevenueCat brauchen ein Konto und eine Verifizierung. Eine Kostenbremse pro Nutzer, und die ist wichtiger als sie klingt, weil jede Generierung dich Credits kostet und ein einzelner Nutzer ohne Limit dein Guthaben leerziehen kann. Und der App Store selbst, mit nativer Hülle, Entwicklerkonto und Apple-Review. Das dauert Tage, nicht Minuten.

Und eine Sache zum Produkt: Dating-Fotos, die besser aussehen als der Mensch beim ersten Date, gehen nach hinten los. Der Bau-Prompt ist deshalb so geschrieben, dass die Person erkennbar bleibt. Wenn du an dieser Stelle schraubst, schraubst du an der Wiederkauftrate.

Plan realistisch: eine halbe Stunde bis zum ersten echten Bild, ein Wochenende bis zur laufenden Web-App, ein Monat bis in die Stores. Und prüf am Wochenende, ob jemand die App überhaupt will, bevor du Geld für das Entwicklerkonto ausgibst.

HÄUFIGE PROBLEME

ZWEI QUELLEN NENNEN GANZ UNTERSCHIEDLICHE UMSÄTZE

Prüf bei jeder Zahl, ob sie den Monatsumsatz oder den Gesamtumsatz seit dem Start meint, und ob sie netto (nach dem Anteil von Apple und Google) oder brutto ist. Vergleich nur Zahlen derselben Art.

WIE ES WEITERGEHT

Bevor du Geld in Entwicklerkonten steckst: Zeig die Web-App ein paar Leuten aus deiner Zielgruppe und schau, ob sie sie wirklich benutzen.

Mehr zur Higgsfield-Anbindung findest du im Guide [Apps mit Higgsfield bauen](#).

Denselben Weg mit Claude statt ChatGPT zeigt [Apps nachbauen mit Claude Code](#).

Fragen oder ein Ergebnis, das du zeigen willst? Komm in die [Community](#).

HINWEIS

Hinweis: Dieser Guide dient zu Informations- und Bildungszwecken. Alle Inhalte spiegeln den Stand bei Erstellung wider, Oberflächen, Funktionen und Preise der genannten Tools können sich ändern. Prüfe Ergebnisse von KI-Tools immer selbst nach. @GPTMarlon übernimmt keine Haftung für Entscheidungen auf Grundlage dieser Inhalte. Dieser Guide stellt keine Anlage-, Steuer-, Rechts- oder Gesundheitsberatung dar. Lass Ergebnisse bei Bedarf fachlich prüfen.

Namen, Logos, Texte, Grafiken und Fotos anderer Apps sind geschützt und werden nicht übernommen, du baust die Funktion nach, nicht das Erscheinungsbild. Rezensionen enthalten Namen echter Personen: Lad nur hoch, was du für die Analyse brauchst, und lösche die Screenshots danach. Deine Higgsfield-Zugangsdaten schickst du niemandem und lädst sie nirgendwo hoch.