

EINE ANLEITUNG VON SABINE KALUSCHA

Verkaufsseiten ohne *Tool-Abo*

Die Schritt-für-Schritt-Anleitung. Wie du Verkaufsseiten, Landingpages und ganze Funnels mit Claude Code, GitHub und Cloudflare Pages baust und ohne monatliche Tool-Gebühr betreibst.

Was diese Anleitung *leistet*

Diese Anleitung beschreibt ein vollständiges, erprobtes Setup, mit dem du Webseiten selbst baust und veröffentlichst. Sie ist aus dem realen Aufbau meiner eigenen Seite sabinekaluscha.com entstanden und gibt genau die Schritte wieder, die dort funktioniert haben.

Du brauchst keinen Hintergrund in der Softwareentwicklung. Du solltest bereit sein, dich einmal in drei neue Werkzeuge einzuarbeiten und mit einer Kommandozeile zu arbeiten, wenn dir Schritt für Schritt gesagt wird, was zu tun ist. Die ehrliche Einordnung der Lernkurve lautet: Die erste eingestellte Seite kostet dich ein bis zwei konzentrierte Nachmittage. Danach ist jede weitere Seite eine Sache von Minuten, weil das Fundament steht und sich nur noch der Inhalt ändert.

Was diese Anleitung nicht leistet: Sie ersetzt keinen Designer für ein aufwendiges Markenbild und keinen Entwickler für komplexe Anwendungen mit Login, Datenbank oder Zahlungslogik. Für klassische Verkaufsseiten, Landingpages, Freebie-Seiten und einfache Funnels reicht sie vollständig aus, und genau dafür ist sie gedacht.

Das Video zur Anleitung

Begleitend zeige ich die einzelnen Schritte am Bildschirm in einem Video. Es lohnt sich, das einmal neben dieser PDF laufen zu lassen, weil man manche Handgriffe schneller versteht, wenn man sie sieht. Du findest es unter youtu.be/EmDAUsUGJo8.

Diese PDF hat zwei Leser

Du liest dieses Dokument als Mensch, der Schritt für Schritt vorgehen will. Es ist zugleich als Kontext-Dokument für eine KI-Assistenz geschrieben. Wer diese PDF an Claude oder ein anderes KI-System übergibt, stellt damit eine vollständige Beschreibung des Systems bereit: welcher Technik-Stack verwendet wird, welche Konventionen für Dateinamen und Verzeichnisse gelten, in welcher Reihenfolge die Schritte ablaufen. Eine KI, die diese Seiten gelesen hat, kann ohne Rückfrage innerhalb dieses Systems weiterarbeiten und zum Beispiel eine neue Seite anlegen, die zu den hier beschriebenen Konventionen passt.

Aus diesem Grund werden alle Begriffe einmal sauber benannt, alle Abkürzungen einmal ausgeschrieben und alle Datei-Pfade vollständig angegeben. Für dich als menschlichen Leser bedeutet das nur, dass an einzelnen Stellen etwas mehr Erklärung steht als unbedingt nötig. Für die KI-Lesart ist genau diese Genauigkeit der Punkt.

Die drei Werkzeuge im *Überblick*

Das gesamte Setup besteht aus drei Werkzeugen, die jeweils eine klar abgegrenzte Aufgabe haben. Wenn du verstehst, welches Werkzeug wofür zuständig ist, wird der Rest der Anleitung selbsterklärend.

Claude Code — der Ort, an dem die Seite entsteht

Claude Code ist ein Programm der Firma Anthropic, das auf deinem eigenen Rechner in einem Terminalfenster läuft. Ein Terminal ist das Textfenster, in dem du Befehle eintippst, statt mit der Maus zu klicken. Du beschreibst Claude Code in normaler Sprache, welche Seite du haben willst, und Claude Code schreibt die dafür nötigen Dateien direkt in einen Ordner auf deinem Rechner. Die Seite besteht am Ende aus HTML- und CSS-Dateien, also aus Text, den ein Browser anzeigen kann.

GitHub — der Ort, an dem die Seite gespeichert wird

GitHub ist ein Online-Dienst, der Ordner mit vollständiger Versionsgeschichte speichert. Ein solcher Ordner heißt Repository, kurz Repo. Jede Änderung an einer Datei wird als nachvollziehbarer Schritt festgehalten, sodass du jederzeit zu einem früheren Stand zurückkehren kannst. Das Repository ist die zentrale Wahrheit darüber, wie deine Webseite gerade aussieht.

Cloudflare Pages — der Ort, an dem die Seite live geht

Cloudflare Pages ist ein kostenloser Hosting-Dienst. Er beobachtet dein Repository bei GitHub und veröffentlicht die Seite automatisch neu, sobald sich dort etwas ändert. Cloudflare Pages stellt außerdem ohne Zusatzaufwand ein gültiges SSL-Zertifikat bereit, also die Verschlüsselung, die ein Schloss-Symbol im Browser und das https-Präfix in der Adresse erzeugt.

Wie die drei zusammenspielen



Der entscheidende Punkt an dieser Kette: Du arbeitest nur ganz links, bei Claude Code. Den Weg über GitHub zu Cloudflare Pages legt die Seite nach der einmaligen Einrichtung von selbst zurück. Ein einziger Befehl genügt, und wenige Minuten später ist die Änderung im Netz.

Was dieser Stack ersetzt

Die meisten Selbstständigen kommen aus einem von drei Lagern. Die folgende Übersicht zeigt, was beim Wechsel jeweils wegfällt und was dafür dazukommt.

Bisheriges System	Was wegfällt	Was dazukommt
Funnel-Software Funnelcockpit, ClickFunnels, Builderall, Perspective Funnel	Die monatliche Grundgebühr, die Bindung an einen Anbieter, die Begrenzung auf dessen Vorlagen und das Risiko, dass deine Seiten verschwinden, wenn der Vertrag endet.	Volle Kontrolle über jede Zeile der Seite, freie Gestaltung, keine laufenden Kosten außer der Domain, dauerhafter Besitz der Dateien im eigenen Repository.
Webseiten-Baukasten Wix, Squarespace, Webflow, Jimdo	Das Abo, die Ladezeit-Nachteile vieler Baukästen und die Schwierigkeit, deine Inhalte sauber zu exportieren, wenn du umziehen willst.	Sehr schnelle, statische Seiten, sauberes HTML für gute Auffindbarkeit bei Suchmaschinen und KI-Systemen, ein Export-Problem entsteht gar nicht erst, weil dir die Dateien gehören.
WordPress mit Hosting klassische Installation bei einem Hoster	Plugin-Updates, Sicherheits-Patches, die angreifbare Datenbank und die Hosting-Rechnung. Eine statische Seite hat keine Datenbank, die man übernehmen könnte.	Wartungsarmut, weil es keine Software-Updates auf einem Server gibt, und ein deutlich kleineres Angriffsziel, weil keine dynamischen Bestandteile laufen.

Was bleibt, egal woher du kommst

Deine Domain bleibt deine Domain. Du kannst sie bei deinem bisherigen Anbieter registriert lassen. Dein E-Mail-Marketing-Tool bleibt ebenfalls bestehen, in diesem Setup ist das Quentn. Es ändert sich der Ort, an dem die Seiten gebaut und ausgeliefert werden, und der Weg, wie Besucher auf deine Formulare gelangen.

Ein ehrlicher Hinweis zu den Kosten: Claude Code wird über ein Abonnement oder über die Programmierschnittstelle von Anthropic bezahlt. Diese Anleitung trägt den Titel „ohne Tool-Abo“, weil das wiederkehrende Abo für Funnel- und Baukasten-Software entfällt. Die Nutzung einer KI-Assistenz ist eine bewusste Investition in das Werkzeug, mit dem du die Seiten baust, und ersetzt mehrere Spezial-Abos zugleich.

Das erstmalige *Setup*

Diese Schritte machst du genau einmal. Danach ist das Fundament eingerichtet und du arbeitest nur noch am Inhalt. Jeder Schritt nennt am Ende, woran du erkennst, dass er geklappt hat.

SCHRITT 1

Konto bei GitHub anlegen

Öffne github.com und erstelle ein kostenloses Konto. GitHub ist der Online-Speicher mit Versionsgeschichte, in dem alle Dateien deiner Webseite liegen werden. Wähle einen Benutzernamen, den du dauerhaft behalten möchtest, weil er später in technischen Pfaden auftaucht.

Geklappt, wenn: du nach der Anmeldung dein leeres GitHub-Profil unter github.com/dein-name siehst.

SCHRITT 2

Das Repository erstellen

Ein Repository auf GitHub ist ein Ordner mit Versionsgeschichte, in dem alle Dateien deiner Webseite liegen. Du erstellst es im GitHub-Web-Interface über die Schaltfläche „New Repository“ und wählst den Sichtbarkeits-Status „private“, damit nur du Zugriff hast. Nenne das Repository nach einem klaren, gleichbleibenden Muster, zum Beispiel `deinname-website`.

Geklappt, wenn: das Repository in deinem GitHub-Konto sichtbar ist und als „private“ markiert ist.

SCHRITT 3

Node.js installieren

Node.js ist eine Laufzeitumgebung, die dein Rechner braucht, um die Seite lokal zu bauen und in einer Vorschau anzuzeigen. Lade die aktuelle Version mit Langzeitunterstützung von nodejs.org herunter und installiere sie. Für dieses Setup ist mindestens Version 22.12 nötig.

Geklappt, wenn: der Terminal-Befehl `node --version` eine Nummer ausgibt, die mit 22 oder höher beginnt.

SCHRITT 4

Claude Code installieren und anmelden

Folge der offiziellen Installationsanleitung von Anthropic für Claude Code. Nach der Installation startest du Claude Code in einem Terminalfenster und meldest dich mit deinem Anthropic-Konto an. Ab jetzt kannst du in normaler Sprache mit dem Werkzeug sprechen.

Geklappt, wenn: Claude Code im Terminal startet und auf eine erste Testfrage antwortet.

SCHRITT 5

Das Projekt lokal anlegen

Lege das Projekt in einem eigenen Ordner an, zum Beispiel `~/Projekte/deiname-website`. Die Tilde steht für dein persönliches Benutzerverzeichnis. Bitte Claude Code, dort ein Astro-Projekt einzurichten. Astro ist das System, das aus deinen Inhalten fertige, statische Webseiten erzeugt. Statisch bedeutet, dass die Seiten ausgeliefert werden, ohne dass im Hintergrund eine Datenbank läuft, was sie schnell und wartungsarm macht.

Geklappt, wenn: der Befehl `npm run dev` im Projektordner einen lokalen Server startet und du die Seite im Browser unter `localhost:4321` siehst.

SCHRITT 6

Das lokale Projekt mit GitHub verbinden

Bitte Claude Code, das lokale Projekt mit deinem GitHub-Repository zu verbinden und den ersten Stand hochzuladen. Das Hochladen heißt Push. Der Standard-Zweig, auf dem die veröffentlichte Seite liegt, heißt `main`. Merke dir diesen Namen, weil Cloudflare Pages später genau diesen Zweig beobachtet.

Geklappt, wenn: du die Dateien des Projekts im GitHub-Web-Interface deines Repositorys siehst.

SCHRITT 7

Cloudflare-Konto anlegen und Pages verbinden

Erstelle ein kostenloses Konto bei cloudflare.com. Lege im Bereich „Workers & Pages“ ein neues Pages-Projekt an und verbinde es mit deinem GitHub-Repository. Wähle als zu beobachtenden Zweig `main` und als Build-Befehl den Astro-Build. Cloudflare Pages baut die Seite ab jetzt bei jeder Änderung automatisch neu.

Geklappt, wenn: Cloudflare Pages einen ersten Build abschließt und dir eine Vorschau-Adresse mit der Endung `.pages.dev` gibt, unter der die Seite erreichbar ist.

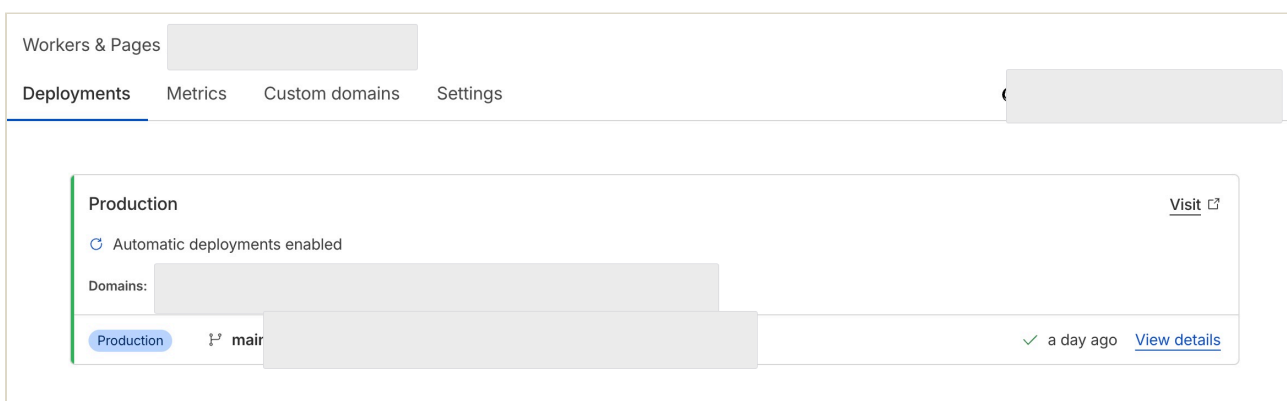
SCHRITT 8

Domain und DNS bei Cloudflare einrichten

DNS steht für Domain Name System und ist das Telefonbuch des Internets, das deinen Domainnamen mit dem richtigen Server verbindet. In diesem Setup verwaltet Cloudflare das DNS, während die Domain weiterhin bei ihrem ursprünglichen Registrar registriert bleibt. Die Hauptadresse der Seite ist `www.sabinekaluscha.com`. Die Adresse ohne `www`, die sogenannte Apex-Domain, leitet per dauerhafter Weiterleitung mit dem Statuscode 301 sauber auf die `www`-Adresse um. Die genauen Werte gibt Cloudflare im Dashboard vor.

Geklappt, wenn: deine Domain im Browser die neue Seite zeigt und ein Schloss-Symbol mit gültiger `https`-Verbindung erscheint.

Der Screenshot zur Orientierung



Das Cloudflare-Pages-Dashboard mit dem verbundenen Repository, dem Zweig `main` und einem grün abgeschlossenen Production-Build. Diese Ansicht zeigt, dass die Verbindung zwischen GitHub und Cloudflare Pages steht.

Das Fundament steht

An diesem Punkt ist die einmalige Einrichtung abgeschlossen. Du hast ein Konto bei GitHub, ein Repository, eine lokale Arbeitskopie auf deinem Rechner, eine Verbindung zu Cloudflare Pages und eine Domain mit gültiger Verschlüsselung. Alles, was jetzt folgt, ist Arbeit am Inhalt.

Halte dir die drei zentralen Befehle bereit, die du im Projektordner immer wieder brauchst. Sie sind bewusst kurz, damit sie schnell in Fleisch und Blut übergehen.

```
# lokale Vorschau starten, erreichbar unter localhost:4321
npm run dev

# die Seite einmal testweise komplett bauen
npm run build

# Änderungen hochladen – Cloudflare baut danach automatisch live
git push
```

Die wichtigste Konvention

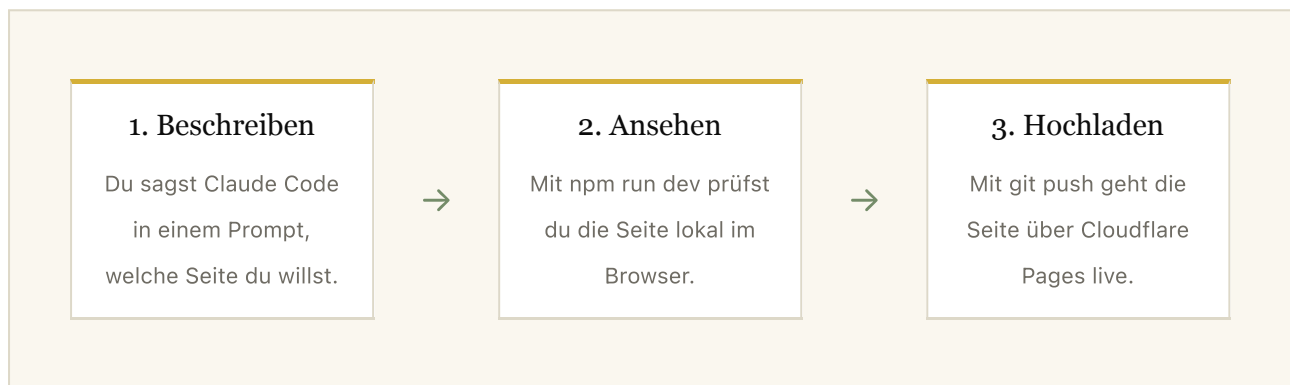
Alles, was im Repository im Ordner `public/` liegt, wird unter der Adresse deiner Domain direkt erreichbar. Eine Datei `public/anleitung.pdf` erscheint zum Beispiel unter `www.deinedomain.de/anleitung.pdf`. Diese Regel brauchst du gleich, wenn es um Downloads geht.

Die erste Seite *bauen*

Eine neue Seite zu bauen folgt immer demselben Dreischritt: Du beschreibst sie Claude Code, eine Datei entsteht im Projekt, die Änderung wird hochgeladen. Danach ist die Seite live.

In diesem Setup liegt jede Seite als eigene Datei im Ordner `src/pages/`. Der Dateiname bestimmt die Adresse. Eine Datei `src/pages/danke.astro` ist später unter `/danke` erreichbar. Gestaltungsregeln liegen in einer zugehörigen Datei im Ordner `src/styles/`. Diese Konvention gibst du Claude Code einmal mit, und von da an hält es sie ein.

Der immer gleiche Ablauf



Ein Prompt ist die Anweisung, die du Claude Code gibst. Je klarer du beschreibst, was auf der Seite stehen soll und welchen Zweck sie hat, desto näher ist das Ergebnis an deiner Vorstellung. Die folgenden drei Beispiele zeigen denselben Aufbau für drei verschiedene Seitentypen, damit du das Muster auf eigene Seiten übertragen kannst.

Beispiel 1 — Eine Landingpage für ein Freebie

Bau eine neue Seite unter `src/pages/freebie.astro`.
Zweck: Anmeldung für mein kostenloses E-Book.
Aufbau: kurze Überschrift, ein Absatz mit dem Nutzen, drei Stichpunkte zum Inhalt, darunter mein Anmeldeformular. Verwende den gleichen ruhigen Stil und dasselbe Layout wie meine bestehenden Seiten.
Lege die Gestaltung in `src/styles/freebie.css` ab.

Beispiel 2 — Eine Verkaufsseite für ein Produkt

Bau eine Verkaufsseite unter `src/pages/kurs.astro`.
Zweck: Verkauf meines Online-Kurses.
Aufbau: Überschrift mit dem Ergebnis, ein Absatz zur Ausgangslage des Lesers, vier Module als Liste, ein Abschnitt zu mir, ein klarer Kaufknopf, darunter Antworten auf häufige Fragen. Ruhiger Ton, kein Pathos. Gestaltung in `src/styles/kurs.css`.

Beispiel 3 — Eine einfache Über-mich-Seite

Bau eine Seite unter `src/pages/ueber-mich.astro`.
Zweck: Wer ich bin und wofür ich stehe.
Aufbau: ein einleitender Absatz, mein beruflicher Weg in drei Abschnitten, ein Portraitbild aus `public/images/`, am Ende ein Hinweis auf meinen Newsletter. Gleicher Stil wie der Rest der Seite.

An den drei Beispielen erkennst du das wiederkehrende Muster: Du nennst den Datei-Pfad, den Zweck der Seite, den gewünschten Aufbau in klaren Bestandteilen und einen Hinweis auf den Stil. Genau diese vier Angaben braucht Claude Code, um eine Seite zu erzeugen, die zu deinem bestehenden Auftritt passt.

Praktischer Hinweis

Sieh dir die Seite nach jedem Prompt zuerst lokal mit `npm run dev` an. Korrekturen formulierst du als Folge-Prompt, etwa „Mach die Überschrift kürzer und das Formular schmaler“. Erst wenn die Seite dir gefällt, lädst du sie mit `git push` hoch.

Live nehmen und Änderungen *machen*

Das Veröffentlichen läuft nach der Einrichtung ohne dein Zutun. Du musst nur verstehen, was im Hintergrund passiert, damit du im seltenen Fehlerfall ruhig bleiben kannst.

Wie das automatische Veröffentlichen funktioniert

Sobald du `git push` ausführst, landen deine geänderten Dateien im GitHub-Repository. Cloudflare Pages beobachtet den Zweig `main` und bemerkt die Änderung innerhalb von Sekunden. Es startet einen Build, baut also aus deinen Dateien die fertige Seite, und stellt das Ergebnis weltweit bereit. Von deinem Befehl bis zur sichtbaren Änderung im Netz vergehen in der Regel ein bis drei Minuten.

Eine bestehende Seite ändern

Eine schon veröffentlichte Seite zu ändern ist kein Sonderfall. Du bittest Claude Code, die betreffende Datei anzupassen, prüfst die Änderung lokal und führst wieder `git push` aus. Es gibt keine erneute Einrichtung, weil das Fundament bestehen bleibt. Die Versionsgeschichte bei GitHub sorgt dafür, dass du jederzeit zu einem früheren Stand zurückkehren kannst, falls eine Änderung dir doch nicht gefällt.

Wenn etwas nicht funktioniert

Drei Fehlerbilder treten am häufigsten auf, und alle drei lassen sich ruhig eingrenzen.

- **Die Änderung erscheint nicht.** Sieh im Cloudflare-Pages-Dashboard nach, ob der letzte Build grün abgeschlossen ist. Ein roter Build zeigt im Protokoll, das Build-Log genannt wird, die Zeile mit der Ursache. Gib diese Zeile Claude Code, das den Fehler in der Regel direkt erklärt und behebt.
- **Die Seite sieht lokal richtig aus, online aber nicht.** Meist liegt das an einer Datei, die lokal existiert und noch nicht hochgeladen wurde. Ein erneutes `git push` löst das.
- **Die Domain zeigt eine Fehlermeldung statt der Seite.** Das deutet auf eine DNS-Einstellung hin. Prüfe im Cloudflare-Dashboard, ob die Einträge für deine Domain unverändert auf das Pages-Projekt zeigen.

CLOUDFLARE PAGES · BUILD-LOG · DEPLOYMENT FEHLGESCHLAGEN

```
12:04:01 Cloning repository...
12:04:03 Installing dependencies (npm install)
12:04:18 Running build command: npm run build
12:04:21 > astro build
12:04:24
error Unexpected "}" in src/pages/danke.astro:18:2

12:04:24 Build failed with 1 error.
12:04:24 Failed: build command exited with code 1
```

So sieht ein fehlgeschlagener Build aus. Die rot markierte Zeile nennt Datei und Stelle, hier `src/pages/danke.astro` in Zeile 18. Genau diese Zeile kopierst du im Fehlerfall zu Claude Code, das den Fehler in der Regel direkt erklärt und behebt.

Umzug von einem bestehenden *System*

Dieses Kapitel ist die Checkliste für den Wechsel. Wenn du sie der Reihe nach abarbeitest, bleibt deine Seite während des Umzugs erreichbar und deine E-Mails laufen ohne Unterbrechung weiter.

DNS-Einträge sauber umstellen

Im DNS gibt es verschiedene Eintragstypen. Ein A-Record verbindet deine Domain mit der Zahlenadresse eines Servers. Ein CNAME-Record verweist eine Adresse auf eine andere Adresse, zum Beispiel die www-Adresse auf das Pages-Projekt. Ein MX-Record legt fest, welcher Server deine E-Mails empfängt. Für die Webseite ändern sich der A-Record oder der CNAME-Record. Die MX-Records für deine E-Mail bleiben unverändert. Stelle die Web-Einträge erst um, wenn die neue Seite über die Cloudflare-Vorschau-Adresse nachweislich funktioniert, damit es keine Lücke gibt, in der die Domain ins Leere zeigt.

Achte auf den TTL-Wert jedes Eintrags. TTL steht für Time to Live und gibt an, wie lange ein alter Wert zwischengespeichert bleibt. Setze die TTL einige Stunden vor dem Umzug auf einen niedrigen Wert, damit die Umstellung schnell überall ankommt und der Umzug planbar wird.

E-Mail-Setup nicht vergessen

Wer den DNS-Anbieter wechselt, verliert leicht die MX-Records aus dem Blick, weil sie nichts mit der Webseite zu tun haben. Geh in dieser Reihenfolge vor: Notiere zuerst alle bestehenden E-Mail-Einträge deines Anbieters, also MX-Records und die zugehörigen Echtheits-Einträge SPF, DKIM und DMARC. Übertrage sie vollständig in die neue DNS-Verwaltung. Stelle danach die Web-Einträge um. Schicke dir zum Schluss eine Test-E-Mail an deine eigene Adresse und antworte darauf, damit du Empfang und Versand bestätigt hast.

Bestehende Seiten archivieren oder umleiten

Wenn alte Adressen bei Suchmaschinen ranken, entscheide bewusst, welche du weiterführst und welche du auf eine neue Seite lenkst. Eine dauerhafte Weiterleitung mit dem Statuscode 301 gibt den Rang der alten Adresse an die neue weiter. Bei Cloudflare richtest du solche Weiterleitungen über eine Redirect Rule ein, die Anfragen an eine alte Adresse automatisch an die passende neue Adresse schickt.

SSL-Zertifikate

Cloudflare Pages stellt das SSL-Zertifikat automatisch und kostenlos bereit. Falls bei deinem alten Anbieter ein Zertifikat kostenpflichtig war, kannst du diese Position nach dem Umzug streichen, weil die Verschlüsselung jetzt ohne Zusatzkosten Teil des Hostings ist.

Tracking und Analytics

Wenn du Google Analytics, den Meta-Pixel oder ein vergleichbares Werkzeug eingebunden hast, nimm die zugehörigen Code-Schnipsel mit ins neue System. Solche Schnipsel gehören in den Kopfbereich der Seiten, der im Quelltext `<head>` heißt. Du musst das nicht von Hand in jede Datei kopieren. Bitte Claude Code mit einer klaren Anweisung darum, etwa „Füge diesen Tracking-Code in den head-Bereich aller Seiten ein“, und es trägt den Code an der zentralen Stelle ein, von der alle Seiten erben.

Quentn oder anderes E-Mail-Marketing-Tool

Deine Formulare bindest du im neuen System direkt als HTML ein. Quentn stellt für jedes Formular eine Empfangsadresse bereit, an die das Formular die eingetragenen Daten sendet. Der Code dafür hat diese Form:

```
<form action="https://deinname.eu-3.quentn-site.com/public/forms/[FORMULAR-ID]/raw/submit"
method="post">
  <div>
    <label>Vorname</label>
    <input value="" type="text" name="first_name">
  </div>
  <div>
    <label>E-Mail</label>
    <input value="" type="email" name="mail">
  </div>
  <button type="submit">Eintragen</button>
</form>
```

Den Platzhalter `[FORMULAR-ID]` ersetzt du durch die Nummer deines Formulars. Diese Nummer findest du im Quentn-Backend in der Formular-Übersicht, wenn du das gewünschte Formular öffnest und dir den Einbettungs-Code anzeigen lässt. Die Feldnamen `first_name` und `mail` müssen exakt so stehen bleiben, weil Quentn die Daten anhand dieser Namen den richtigen Feldern zuordnet. Die Beschriftungen und Knopftexte darfst du frei ändern. Wenn du Claude Code den Code und die Formular-Nummer gibst, baut es das Formular passend in die gewünschte Seite ein.

Datenschutz und Impressum

Diese Pflichtseiten dürfen beim Umzug nicht verloren gehen. Sie liegen im neuen Setup als eigene Seiten unter `src/pages/datenschutz.astro` und `src/pages/impressum.astro` und sind im Fußbereich jeder Seite verlinkt. Für die rechtssichere Formulierung gibt es etablierte Generatoren, deren Ergebnis du in diese Seiten übernimmst.

Mein eigener Umzug von *Funnelcockpit*

Die vorige Checkliste ist allgemein gehalten. Hier kommt mein eigener Weg, so wie ich tatsächlich von Funnelcockpit auf dieses Setup gewechselt bin. Die Reihenfolge der Schritte ist dabei wichtiger als jeder Schritt für sich allein.

1. **Bestandsaufnahme zuerst.** Ich habe alles aufgeschrieben, was in Funnelcockpit lag: die Seiten, den angebundenen Blog mit seinen Artikeln, sämtliche Bilder und vor allem alle DNS-Einträge inklusive der für meine E-Mail. Diese Liste wurde später mein Sicherheitsnetz.
2. **Inhalte aus der Hand des Anbieters holen.** Ich habe die Hauptseite als Entwurf gesichert und alle Bilder lokal gespeichert, damit nichts mehr an Funnelcockpit hängt, sobald ich kündige. Die Bilder liegen seitdem im Repository und werden mit der Seite ausgeliefert.
3. **Beim Blog ausgewählt statt alles geschleppt.** Aus dem alten Blog habe ich nur die Artikel mit echtem Wert übernommen, jeweils als Markdown mit den zugehörigen Bildern. Den Rest habe ich bewusst zurückgelassen.
4. **Neu gebaut mit Claude Code und Astro.** Die Hauptseite habe ich auf Basis meines eigenen Funnelcockpit-Entwurfs neu bauen lassen, die Bestelllinks wieder eingesetzt und eine freundliche Fehlerseite für die Umbauphase ergänzt.
5. **Erst die Vorschau, dann die Domain.** Die neue Seite lief zunächst nur unter der Cloudflare-Vorschau-Adresse. Erst als dort wirklich alles stimmte, habe ich die Domain umgezogen, damit es keine Minute gibt, in der die Seite ins Leere zeigt.
6. **E-Mail vor der Webseite.** Vor dem Umstellen habe ich alle E-Mail-Einträge notiert: die MX-Records zu meinem Postfach-Anbieter, die SPF-Zeile und die DKIM-Einträge für meinen Versanddienst und für Quentn, dazu DMARC. Beim Abgleich gegen den automatischen Scan des neuen Anbieters fehlten vier Einträge, die ich von Hand nachgetragen habe. Ohne die vorab erstellte Liste wäre mir das durchgerutscht.
7. **Subdomains in Ruhe gelassen.** Meine Subdomains für die App und die Akademie zeigen weiterhin direkt auf ihre eigenen Server. Die habe ich bewusst unangetastet gelassen, damit dort nichts kippt, während ich vorne umbau.
8. **SSL kam von allein.** Cloudflare stellt das Zertifikat automatisch bereit. Das Zertifikat aus dem alten Setup brauchte ich danach nicht mehr und konnte die Position einsparen.
9. **Getestet, dann gekündigt.** Erst als die Seite live war, die https-Verbindung stand, eine Test-E-Mail an mich selbst ankam und der Quentn-Status grün war, habe ich den Funnelcockpit-Vertrag beendet.

Stolpersteine, die ich selbst hatte

Ein DKIM-Eintrag war doppelt angelegt, und erst nach dem Entfernen des überzähligen war die E-Mail-Signatur wieder eindeutig. Während der Bauphase gab es bewusst keinen Eintrag für die Adresse ohne www, die Seite lud also kurz nirgendwohin. Am Ende trägt die Reihenfolge alles: erst Vorschau prüfen, dann E-Mail sichern, dann DNS umstellen, ganz zuletzt kündigen.

Was geht damit, *was nicht*

Eine ehrliche Einordnung gehört dazu, damit du das Werkzeug für die richtigen Aufgaben einsetzt und für die falschen gar nicht erst.

Wofür dieses Setup sehr gut passt

- Verkaufsseiten und Landingpages mit Text, Bildern und einem Anmelde- oder Kaufknopf. Genau dafür ist es gebaut.
- Freebie-Seiten und mehrstufige Funnels aus mehreren verbundenen Seiten, etwa Eintrageseite, Bestätigungsseite und Dankeseite.
- Eine ruhige Hauptseite, eine Über-mich-Seite und ein Blog mit längeren Texten.
- Seiten, die schnell laden und bei Suchmaschinen und KI-Systemen gut auffindbar sein sollen, weil das ausgelieferte HTML sauber ist.

Wo die Grenzen liegen

- Ein geschützter Mitgliederbereich mit Login und persönlichen Daten braucht eine Anwendung mit Datenbank. Dafür ist ein darauf spezialisiertes Werkzeug oder ein Entwickler die bessere Wahl.
- Ein vollwertiger Onlineshop mit Lagerbestand, Versandlogik und Zahlungsabwicklung übersteigt den Zweck einer statischen Seite. Für einzelne Produkte reicht ein Kaufknopf, der zu einem Bezahlanbieter führt.
- Ein sehr aufwendiges, einzigartiges Markendesign mit aufwendigen Animationen profitiert von der Hand eines Designers. Das Setup liefert ein ruhiges, professionelles Bild, kein preisgekröntes Sonderdesign.

Der ehrliche Kern lautet: Für den größten Teil dessen, was eine Selbstständige an Seiten täglich braucht, trägt dieses Setup vollständig. Für die wenigen Fälle mit echter Anwendungslogik lohnt sich der Griff zu spezialisierter Software oder fachlicher Unterstützung, und das ist eine ruhige Entscheidung, keine Niederlage.

Ein ruhiger *Anfang*

Das Setup wirkt von außen größer, als es nach dem ersten Durchlauf bleibt. Sobald das Fundament einmal steht, ist eine neue Seite eine Sache von Minuten.

Du hast jetzt eine vollständige Beschreibung des Systems in der Hand. Du kennst die drei Werkzeuge und ihre Aufgaben, du kennst die einmalige Einrichtung, den immer gleichen Dreischritt beim Bauen einer Seite und die Checkliste für den Umzug von einem bestehenden System. Mehr brauchst du für den Anfang nicht.

Wenn du diese PDF einer KI-Assistenz gibst, kann sie auf dieser Grundlage direkt mit dir innerhalb des Systems arbeiten, weil alle Konventionen, Pfade und Werkzeuge hier benannt sind. Wenn du selbst anfängst, beginne mit einer einzigen kleinen Seite und geh den Dreischritt einmal vollständig durch. Das Vertrauen kommt mit dem ersten `git push`, der die eigene Seite ins Netz bringt.

Ich begleite diese Themen weiter und schreibe in ruhigen Abständen darüber, wie sich ein schlankes Web-Setup im Alltag einer Selbstständigen anfühlt und wo es sich lohnt. Wenn du das lesen möchtest, findest du den Newsletter auf sabinekaluscha.com. Es gibt keinen Druck und kein tägliches Dauerfeuer, nur gelegentliche, brauchbare Post.

*Hyggelige Grüße,
Sabine*