BASIC MANUAL

コードを書かずに、SaaSを作る

Claude Code と Cloudflare を使った制作の進め方

実例：取引先ポータル ATRIUM

CONTENTS

01 何が変わったか

02 実例：取引先ポータル ATRIUM

03 道具の全体図

04 作り方の6ステップ

05〜08 各ステップの進め方

09 AIへの指示の出し方

10 落とし穴

11 作り始める前に

2026年11月

GarlondWorks

WHAT CHANGED

01 何が変わったか

専門のプログラマーでなくても、仕事で使えるWebサービス（SaaS＝インターネット越しに使うサービス）を作れるようになった。要るのはコードの知識より、「何を・誰のために・どこまで作るか」を決めて、AIに正しく伝える力。

人とAIの役割分担

人がやること	AIがやること
困りごとを見つける	決めた設計をコードにする
誰が使うかを決める	自分で動かして確かめる
作らないものを決める	エラーを読んで原因を探し、直す
出来上がりを触って、良し悪しを判断する	やったことを記録に残す

チャットAIと「作業するAI」の違い

日報アプリを作ったときも、何を作るかを決め、AIに伝え、出来上がりを試して直す、という流れで進めてきたと思う。このマニュアルの手順は、その流れを言葉にして並べ直したもの。道具が変わって違ってくるのは、AIが自分で手を動かせる範囲。チャットAIにもメモリやコードを試す機能はある。違いの芯は、自分のパソコンのファイルとコマンドに、AIの手が直接届くかどうか。Claude Code は、ファイルを直接読み書きし、コマンドを実行して結果を自分で確かめる（出典 2）。

	チャットAI	作業するAI（Claude Code）
作るファイル	人が貼り付けて運ぶ	AIが直接読み書きする
動かして確かめる	人が動かして、結果を伝える	AIがコマンドを実行して確かめる
エラーが出たら	人がエラー画面を貼って聞く	AIが実行結果を読んで直す
作業の前提	会話の中で伝え直す	決まったファイル（CLAUDE.md）を会話の始めに読む

このマニュアルの読み方

02 で実例の ATRIUM を紹介し、03 で道具をそろえる。04〜08 が作り方の本体で、各ステップに「ATRIUM ではどうしたか」と「AIへの指示の例」を並べた。09・10 はどの段階でも使う考え方、11 は作り始める前の確認。

CASE

02 実例：取引先ポータル ATRIUM

取引先ごとに「応接室」を用意して、資料・やりとり・やることを1か所にまとめるポータル。コードを書いたのはAI。人は、何を作るかを決め、出来上がりを触って判断し、Cloudflare の管理画面での設定を受け持った。

三層構造：下へ行くほど私的になる

応接室 自分 ＋ 招いた取引先	案件ごとの資料・伝言・やること。完成したものだけが出る
取引先の部屋 自分だけ	取引先ごとの台帳。作りかけの資料と内部メモはここに置く
執務室 自分だけ	全取引先の横断ビュー・予定・やること・ノート・名刺

作りかけは「取引先の部屋」で書き、完成したものだけを「応接室」に出す。応接室では、相手はこちらが置いた資料をゴミ箱に入れることはできるが、完全に消すことはできない（2026年10月時点・出典 3）。

名前は古代ローマの邸宅から借りた。アトリウムは主人が客を迎える広間で、その奥に記録を置く書斎がある。公から私へ段階的に移る配置が、この三層とそのまま重なった（出典 3）。

実際の応接室：取引先の画面

2026年11月2日に、GarlondWorks 側から開いた画面。開いてすぐ、案件・資料・相談の入口が1画面に並ぶ。



画面の見方

画面の場所	何ができるか
相談ロボ（上の帯）	応接室に置いた資料をもとに、AIに質問できる
案件	案件ごとに、資料・伝言・やることがまとまる。カードの下の数字は、それぞれの件数
最近の資料	置いた資料が新しい順に並ぶ。どの案件の資料か、誰がいつ置いたかも出る
お知らせ	相手の動きが、日付つきで自動で並ぶ
メモ	GarlondWorks から部屋への伝言。相手は読むだけ
メンバー	この部屋にもう入った人



スマホで開いたとき

同じ部屋を、スマホの幅で開いた画面。カードが縦1列に並び、右下に「AIに相談」のボタンが出る。現場にいても、スマホから資料を確かめたり、相談したりできる。

パソコンとスマホで別々の画面を作ったわけではない。1つの画面が、幅に合わせて並び方を変えている。

作った順番

時期	やったこと
2026年9月20日	VISION.md を書き、入口の鍵（Access）が本当に効くかを最小の仕組みで確かめた（出典 3）
2026年9月21日	設計から画面・デザインまでを一巡させた（出典 3）
2026年9月23日	実際に使う環境に公開した（出典 3）
2026年10月上旬	実際の取引先とのやりとりで使い始めた（出典 3）
2026年10月〜	使いながら、ノート・名刺・通知メールなどを足していった（出典 3）

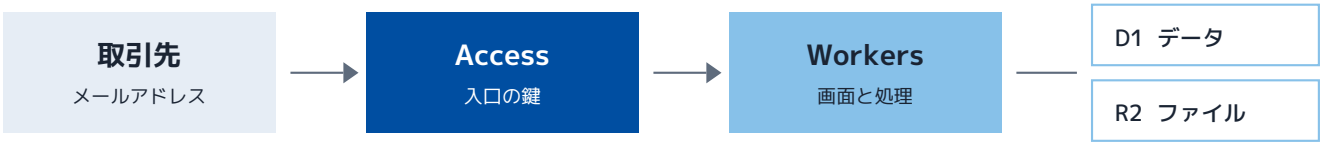
TOOLS

03 道具の全体図

作る係は Claude Code、置き場所は Cloudflare。サーバーを自分で用意したり管理したりする必要はない。

道具	役割	メモ
Claude Code	作業するAI。指示を聞いてファイルを書き、動かして確かめる	デスクトップアプリ版は有料のClaudeプランが必要（出典 2）。料金はこのマニュアルには載せていない（巻末を参照）
Workers	画面と処理を動かす場所	Cloudflare の中で動く。サーバーの管理は要らない
D1	データベース	取引先・案件・やること・伝言などを入れる
R2	ファイル置き場	資料・画像を入れる。使い始めるときに、従量課金の契約で有効にする必要があった（無料枠あり・出典 3）
Access	入口の鍵。招いたメールアドレスだけを通す	相手はアカウントを作らず、メールに届く確認コードで入れる。無料プランは50ユーザーまで（出典 1）

取引先が中に入るまで



招いていないアドレスは、Access の手前で止まる。アプリの中身には一度も届かない。

なぜ Google Apps Script（GAS）ではなかったか

2026年9月に判断した時点では、GAS のWebアプリに「特定の外部ユーザーだけを通す」という権限の細かさが標準では無かった。それだと、自分で書いた条件分岐（if 文）が唯一の壁になる。Access なら、誰を通すかの判定がアプリのコードの外側で行われる。コードに不具合があっても扉は開かない（出典 3）。

管理画面の設定は、人が受け持つ

AIがふだん使う Cloudflare のコマンドからは、Access を設定できない。そのため ATRIUM では、招く相手のアドレスの追加や R2 の有効化を、人が Cloudflare の管理画面で行った。ログイン画面の見た目など一部は、別に発行した鍵（APIトークン）を使ってAIが設定した（出典 3）。

SIX STEPS

04 作り方の6ステップ

コードを書く前の 1・2・3 に一番時間を使う。ここが固まっていれば、4 以降はAIが速く進めてくれる。

1
困りごとを言葉にする

人　誰が何に困るか一文で言う
AI　質問して言葉にする

2
VISION.md を書く

人　作らないもの・完成を決める
AI　聞き取ってファイルにする

3
器を選ぶ

人　比べた結果から1つ選ぶ
AI　同じ軸で候補を比べる

4
小さく作って確かめる

人　合格の条件を決め、触る
AI　最小の形で作って試す

5
見た目のルールを先に書く

人　参考と方向を決める
AI　参考を実測してルールにする

6
公開して守る

人　公開してよいかを決める
AI　安全面の点検と控えの仕組み

どのステップでも、合格かどうかを決めるのは人。AIが手を動かす 4〜6 でも、出来上がりを触って「これでいい」と言うまでは終わらない。

器は「何を守りたいか」で選ぶ（STEP 3 の考え方）

ATRIUM で一番守りたかったのは「取引先Aの人に、取引先Bの資料も存在も見せない」こと。だから権限の守り方を一番の判断軸にして、Cloudflare を選んだ。作るものが変われば、選ぶ器も変わる。

STEP 1 ・ 2

05 困りごとを言葉にして、VISION.md を書く

STEP 1 機能ではなく、困りごとから書く

「ポータルを作りたい」「アプリを作りたい」から始めると、AIは世の中にある一般的なものを作ってしまう。誰が、どんな場面で、何に困っているのか。そこから始めると、作るべきものと作らなくていいものが自然に分かれる。

ATRIUM では

出発点は「ポータルを作りたい」ではなく、「クライアントと同じものを見る場所がない」という困りごとだった。資料はメール添付、進捗は口頭とチャット、最新版がどれかは双方の記憶頼み。前に使っていたポータルは大企業の社員向けで、実際に使っていたのは4つの機能だけだった（出典 3）。

STEP 2 VISION.md：AIに毎回読ませる「北極星」

VISION.md は、作るものの目的と方針を書いたファイル。Claude Code は、プロジェクトに置いた CLAUDE.md を会話（セッション）の始めに読む（出典 2）。CLAUDE.md に「VISION.md を読む」と書いておけば、何日たっても同じ地点に戻れる。会話が長くなると、AIは目の前のつじつま合わせに流れやすい。ファイルが、それを引き戻す役になる。

書くこと	中身
核心の問題	誰が、何に困っているか
解決するもの	何を作れば、その困りごとが消えるか
設計の原則	迷ったときに立ち返る決まりごと（例：権限は仕組みで守る）
作らないもの	やらないと決めたこと。これが無いと機能が膨らみ続ける
完成の定義	何ができたら「できた」と言えるか

AIへの指示の例

取引先と資料を共有するポータルを作りたい。まだコードは書かないで。まず VISION.md を一緒に作りたい。

困っているのは、資料がメール添付でばらばらになり、どれが最新か分からなくなること。使うのは自分と、取引先の担当者。取引先は10社くらい。

私に質問しながら、核心の問題・解決するもの・設計の原則・作らないもの・完成の定義を埋めていって。できたら、CLAUDE.md に「最初に VISION.md を読む」と書いておいて。

STEP 3 · 4

06 器を選んで、小さく作って確かめる

STEP 3 候補はAIに比べさせ、決めるのは自分

どの仕組みで作るかは、AIに複数の候補を同じ軸で比べさせると判断しやすい。軸は自分で指定する。AIのおすすめは参考にしつつ、最後は VISION.md の「核心の問題」に一番効くものを選ぶ。

AIへの指示の例

この VISION.md を実現する仕組みを、GAS・Cloudflare・もう1つ別の候補で比べて。

比べる軸は、権限の守り方、月々の費用、自分で面倒を見られるか、の3つ。表にして、おすすめとその理由も書いて。専門用語には一言ずつ説明を付けて。

STEP 4 一番怖いところから、小さく試す

いきなり全部を作らない。うまくいかなかったら計画ごと崩れる部分を先に見つけて、そこだけを最小の形で試す。そして、試す前に「何が起きたら合格か」を言葉で決めておく。合格の条件が無いと、AIは「動きました」で止まってしまう。

ATRIUM では

画面を作る前に、入口の鍵（Access）だけを最小の仕組みで確かめた。合格の条件は3つ。招いたアドレスで入れる／招いていないアドレスでは中身が見えない／ログインせずに外から開いても中身が返ってこない。3つとも実際に試して通ってから、本体に進んだ（出典 3）。

データベースも、実際にデータを入れて決まりごとを確かめた。その途中で、内部向けの資料が1回の更新で応接室に出てしまう穴が見つかり、データベース側で拒否する仕組みを足して塞いだ（出典 3）。

AIへの指示の例

本体を作る前に、入口の鍵だけを試す最小の仕組みを作って。

合格の条件は3つ。招いたアドレスで入れること。招いていないアドレスでは中身が見えないこと。ログインしないで外から開いても、中身が返ってこないこと。

3つとも実際に試して、結果を見せて。どれかが通らなければ、先に進まないで報告して。

STEP 5

07 見た目のルールを、画面より先に書く

画面を作ってから色を直すと、直すたびに全部の画面に手が入る。先にルールを1つのファイルに書けば、方向を変えるときもそのファイルを直すだけで済む。

ATRIUM では

コンセプトは「説明書が要らないこと」。デジタルが苦手な人でも、覚えずに使える画面を目指した（出典 3）。

実在するサイトを2つ選び、色・書体・形は片方から、配置と余白はもう片方から、と軸ごとに実際の値を測って借りた。それを DESIGN.md と、色を定義した1つのファイルにまとめてから画面を作った（出典 3）。

4案

最初の1日（2026年9月21日）に試した見た目の方向の数。4案を行き来しても、作り直したのは色を決めた1ファイルだけだった（出典 3）

最初に書いたルール（2026年9月21日時点・抜粋）

ルール	理由
鮮やかな色は1つだけ。文字には使わず「面」で使う	明るい色は白地の文字にすると読めない。面にして上に黒い文字を置く
角の丸みはごく小さく、影は付けない	境目は罫線と余白で作る
書体はゴシックだけ	明朝を混ぜない。読み込めない環境でも明朝に落ちないようにする
状態を色だけで伝えない	色が分かりにくい人もいる。形と言葉でも示す

その後も、配色や画面の骨格は使いながら何度も見直している（出典 3）。ルールのファイルは、一度書いたら終わりではなく、方向を変えるたびに先に直す場所になる。

AIへの指示の例

画面を作る前に DESIGN.md を作りたい。参考にしたいサイトはこの2つ。色と書体はこっち、配置と余白はこっちから借りたい。

実際のサイトから色・文字の大きさ・余白の値を測って、ルールにまとめて。色の定義は1つのファイルに集めて、あとからそこだけ直せば全画面が変わるようにして。

STEP 6

08 公開して、守る

人を招く前に、「誰が何をできてしまうか」の目で全体を点検する。公開したあとも、データの控えと利用量を見張り続ける。

守ること	やり方	ATRIUM では
見せてはいけないものが見えないか	取引先のふりをして、見えてはいけないものを試させる	取引先として試したら、他の取引先の存在が推測できる穴が見つかり、塞いだ（10で詳しく・出典 3）
データが消えても戻せるか	控え（バックアップ）を自動で取り、一度は戻す練習をする	週1回、自動で控えを取っている。ただし戻す手順はまだ試していない（出典 3）
無料枠を使い切らないか	利用量のアラート（通知）を設定しておく	アラートで、データベースの読み取りが無料枠の78%に達したと分かり、読み方を直した（出典 3）

AIへの指示の例

公開する前に、取引先Aの人のふりをして、取引先Bの資料や、Bが存在すること自体に届く方法がないかを試して。

URLを書き換える、番号を順に試す、などの攻め方も含めて。

見つけたら、直す前にまず一覧にして報告して。

公開・お金・個人情報は、人が最後に決める

AIは点検も修正もできる。でも、公開してよいか、お金がかかる変更をするか、誰を招くかは、人が決めて承認する。ATRIUM でも、実際に使っている環境への反映は、人がOKを出してから行っている（出典 3）。

HOW TO ASK

09 AIへの指示の出し方

指示に入れるのは4つ。何のためか、誰が使うか、やらないこと、何ができたら完成か。

入れること	なぜ要るか
目的	何のための機能かが分かると、AIが細部を正しく選べる
誰が使うか	パソコンが得意な人と苦手な人では、正解の画面が違う
やらないこと	書かないと、AIは親切心で機能を足してしまう
完成の条件	「動きました」で止まらせず、実際に操作して確かめさせる

同じお願いの、悪い例と良い例

悪い例

資料をアップできるようにして。

良い例

応接室に、取引先の担当者が資料を置けるようにしたい。

使うのはパソコンが苦手な人なので、ボタンは1つで迷わない形にして。

こちらが置いた資料を、相手が完全に消せる作りにはしないで。

完成の条件：取引先として入って資料を置ける／自分が置いたものは引き取れる／こちらが置いた資料は、相手からは完全に消せない。この3つを実際に操作して確かめてから報告して。

進め方のコツ

コツ	中身
一度に頼むのは1つ	まとめて頼むと、どこで崩れたか分からなくなる
「まだ作らないで」と言う	考えを固めたい段階では、先に質問させる
分からない言葉は聞き返す	専門用語はAIに説明させてよい。分かったふりで進めない
決めたことはファイルに残す	会話は流れて消える。VISION.md・DESIGN.md・作業の記録に書かせる

PITFALLS

10 落とし穴：ATRIUM で実際に起きたこと

どれも、画面を見ているだけでは気づけなかった。「大丈夫？」と聞くとAIは「大丈夫です」と返しやすい。「招いていない人が中を見る方法を3つ挙げて」のように、壊し方を考えさせる聞き方が効く。

落とし穴 1

エラーの言い分けが、取引先の一覧を漏らす

「存在しない＝404」「権限がない＝403」と返し分けると、外から部屋のURLの名前を順に試すだけで、どの取引先が存在するかが割り出せてしまう。取引先の一覧は営業上の秘密なので、どちらも404に統一した（出典 3）。

落とし穴 2

テストが「通った」のに、何も確かめていなかった

テスト用に付けた資料の番号の形が、アプリが受け付ける形と違っていた。そのため中身を調べる前に「見つかりません（404）」で門前払いされていて、権限の判定は一度も動いていなかった。それを「404だから安全」と読みかけた（出典 3）。合格したときは、なぜ合格したのかまで確かめる。

落とし穴 3

招待の漏れに、こちらが気づけない

Access は、許可していないアドレスにはそもそも確認コードを送らない。拒否した記録も残らない。相手は「コードが来ない」状態になり、こちらは連絡を待つしかない。招待は、データベースへの登録と Access の一覧への追加の両方が要る。1回の操作で両方に届く形が理想だが、ATRIUM ではまだ人の手順で補っている。招くときは Access の一覧にも人が足し、取引先を消す画面には「Access 側でも外す」という案内を出している（出典 3）。

落とし穴 4

無料枠が、知らないうちに減っていた

トップ画面を開いている間、全部の記録を毎回読み直す作りだった。Cloudflare のアラートで、データベースの読み取りが無料枠の78%に達したと分かった。そのままだと上限を超える見込みだったため、必要な行だけを読む作りに直した（出典 3）。

CHECKLIST

11 作り始める前に

チェックリスト

	確かめること
☐	困りごとを一文で言える
☐	使う人と、だいたいの人数が決まっている
☐	作らないものを3つ言える
☐	VISION.md があり、CLAUDE.md に「VISION.md を読む」と書いてある
☐	一番怖いところ（権限・お金・個人情報）がどこか分かっている
☐	最初に試すことの「合格の条件」を言葉にできる
☐	管理画面で人が操作する部分（鍵の設定・有料機能の有効化）を把握している

困りごとを見つけ、AIと形にし、使ってみて直す。この手順は、日報アプリづくりにも、次に作るものにも、そのまま当てはめられる骨組み。このマニュアルは、その手順に名前を付けて並べたもの。

SOURCES

出典と確認の範囲

出典 1	Cloudflare「Zero Trust / SASE plans」https://www.cloudflare.com/plans/sase-zero-trust/ 無料プランの料金と人数の上限（50ユーザー）を、料金表の全文で確認。2026年11月2日時点
出典 2	Anthropic「Claude Code overview」https://code.claude.com/docs/en/overview Claude Code の働き（ファイルの読み書き・コマンド実行）、CLAUDE.md を会話の始めに読むこと、デスクトップアプリに有料プランが要ることを、全文で確認。2026年11月2日時点
出典 3	GarlondWorks「ATRIUM 制作記録」（VISION.md・DESIGN.md・実装の記録・引き継ぎ台帳） 社内の一次記録。日付・設計判断・落とし穴・利用状況を全文で確認

確認の範囲：出典 1〜3 はいずれも全文を確認した。料金・無料枠・プランの中身は変わることがあるので、使い始める時点の公式ページが正になる。Claude の料金は、公式ページで確かめられなかったため、このマニュアルには金額を載せていない。