



Idea to **MVP** in 1 Day

A compressed, hour-by-hour path to a working, demo-able mobile app MVP — for when you don't have weeks, you have today, and you just need to know if the idea is worth building further.

THE PROMISE (AND THE FINE PRINT)

By the end of today, you'll have a working MVP running on a simulator or a real device, proving one core flow — something you can put in front of 2–3 people and get a real reaction to.

It does **not** mean App Store submission, production-hardened code, pixel-perfect design, or anything close to “done.” A 1-day MVP is a question you're asking your users, not an answer you're giving them.



01 · THE PLAN

Budget your **day**

Four blocks, eight hours. Treat these as budgets, not targets — if design is eating into hour 4, that's a signal to cut scope, not to keep polishing.

HOURS	WHAT YOU'RE DOING
0-1	Scope the idea down to one core user flow
1-3	Design fast — wireframes and a minimal design language, not a full design file
3-7	Build fast — React Native + Firestore, prompt-and-go
7-8	Test the one flow, package a demo, get feedback from 2-3 people

WHAT YOU'RE NOT DOING TODAY

Say this out loud before you start — it's the thing that will save you the most time: today is not the day for App Store review, payment processing, a polish pass on the UI, or thinking about scale. If any of those start creeping into your plan, you've drifted off the 1-day path — cut it and move on.



Cut to one **flow**

Every idea has exactly one thing at its center that has to work for it to mean anything. Find that flow and cut everything else. Ask yourself: if a stranger used only this one flow, would they understand what the app is for?

THE RUTHLESS FEATURE CUT CHECKLIST

- Authentication beyond the bare minimum. **Concretely:** skip Firebase Auth entirely, hardcode `const TEST_UID = 'test-user-1'` in one constants file, and use it everywhere a real user ID would go. Real auth is a Day-2 job and swapping it in later is a find-and-replace on one constant.
- Settings, preferences, or customization screens
- Notifications, reminders, or any background/async behavior
- Social features — sharing, comments, likes, friend lists
- Anything that only matters once you have real user data (analytics, recommendations, history)
- A second or third screen that doesn't sit directly on the core flow's path

COPY-PASTE · ONE-PARAGRAPH MVP SPEC

I'm building an MVP for [one-sentence idea]. The single core flow is: the user [does X], which results in [Y]. It's a mobile app for iOS, built in React Native. For today, I only need this one flow working end to end — no auth beyond a placeholder, no settings, no extra screens. The user should be able to [action] and immediately see [result].



Design in **hours**, not days

This block leans on the workflow from Designing UI with AI, compressed — you're skipping the manual-polish pass entirely today.

Skip Figma Design polish

Take the Figma Make wireframe as-is, or skip wireframing altogether and prompt Claude Code directly for styled React Native components. A 1-day MVP needs a screen clear enough to prove the flow works — not a polished design file.

Grab real reference screens

Don't generate a design from a blank prompt. Pull one or two real screens from Mobbin that solve a similar problem, and feed those in as reference. Grounding beats generating, especially under a time crunch.

SET A MINIMAL DESIGN LANGUAGE BEFORE YOU PROMPT

You don't need a full brand kit today, but you do need three decisions made before you start prompting, or the AI will invent a slightly different palette and spacing on every screen it generates:

- One primary color
- One font
- One spacing unit (8px is a safe default)



Ship the **s+ta+ck** that iterates fast

The longest block of the day, leaning directly on The AI Coding Stack.

Default: React Native + Firestore — here's why the alternatives are wrong for today specifically.

INSTEAD OF	USE THIS	BECAUSE
Native Swift	React Native	Xcode's build times eat hours you don't have, and agents write RN more reliably — see <i>The AI Coding Stack</i> , which sets out the evidence for that claim in full.
Supabase	Firestore	Pay-per-use, zero-setup — you're writing data within minutes, with no infrastructure decisions to make today.
Figma Design	Figma Make, or skip	Same trade as the design block: a structured design file is worth building when screens will be maintained for months. Today, a raw wireframe — or a reference screen and a good prompt — gets you to code faster.

COPY-PASTE · TERMINAL · PROJECT TO WRITING DATA

1. Scaffold the app (TypeScript + file-based routing)

```
npx create-expo-app@latest my-mvp --template default
cd my-mvp
```

2. Create a Firebase project at console.firebase.google.com,

```
# add an iOS app with your bundle ID, download
# GoogleService-Info.plist into the project root.
```

3. Install the SDK

```
npx expo install @react-native-firebase/app \
  @react-native-firebase/firestore
npx expo install expo-build-properties
```

4. Run it on a device (native modules => not Expo Go)

```
npx expo run:ios --device
```

Firestore needs no schema and no migrations — the first `.collection('items').add({...})` creates the collection. Start rules in test mode today; locking them down is a Day-2 job, and one you must not forget if this MVP ever meets a real user.



One prompt, one good **pass**

Everything you decided in hours 0–3 goes into a single prompt. Optimize for one strong first pass rather than five rounds of refinement — each round costs you twenty minutes you don't have.

COPY-PASTE · THE HANDOFF PROMPT

Build this MVP screen (or screens) in React Native with Expo Router and TypeScript.

SPEC: [paste your one-paragraph MVP spec from Hour 0–1]

DESIGN LANGUAGE:

- Primary color: [hex]
- Font: [name]
- Spacing unit: 8px, scaling 8/16/24/32
- Reference screens: [attach or link the Mobbin screens]

DATA: Firestore via @react-native-firebase/firestore.

Collection [name], documents shaped like:

```
{ [field]: [type], createdAt: timestamp }
```

No auth today — use a hardcoded TEST_UID constant everywhere a user id is needed.

CONSTRAINTS:

- One flow only: [action] → [result]. No settings, no onboarding, no extra screens.
- Include the loading and empty states — I'm demoing this.
- No placeholder/lorem copy. Realistic text and lengths.

Build the whole thing, then tell me how to run it.

Skip EAS — but know the catch

Building and signing for distribution is a same-day time sink. **The catch:** Expo Go cannot load native modules — Firebase, RevenueCat, notifications and StoreKit simply don't exist there, and the errors look like unrelated bugs.

So which do you need?

If your one flow is UI and local state only, stay in Expo Go — it's the fastest loop there is. The moment it touches Firestore or any native module, spend the 20 minutes on `npx expo run:ios --device` once, and iterate against that build for the rest of the day.



Get a real **reaction**

Before you show this to anyone, confirm the one thing that actually matters
— then get it in front of people.

SMOKE TEST CHECKLIST

- Does the core flow work start to finish, without crashing?
- Does it work on a real device, not just the simulator, if you have one available?
- Is the result of the flow actually visible to the user when they finish it?
- Does it survive a cold start — force-quit the app, reopen, run the flow again?
- Does it handle the empty state, before any data exists? That's the first thing a new tester sees.

IF SOMETHING FAILS WITH AN HOUR LEFT

Don't start debugging. With one hour left, the goal changes from "fix it" to "still get a real reaction today."

- **It crashes:** `git restore` back to whatever last worked and demo that, even if it's a smaller flow. A working smaller thing beats a broken bigger one.
- **It works, but only sometimes:** record a screen capture of a successful run right now, while it works. That recording is your demo — you can still show it to people tonight.
- **One step is broken mid-flow:** fake that step. Hardcode the result and move on; you're testing whether people want the outcome, not whether your code computes it.
- **Nothing works:** the honest answer is that today's question got answered differently — you learned the scope was too big. Cut it in half and the next attempt is a much better one.

Screen recording

Record yourself running through the core flow on the simulator or device — the fastest way to share something concrete with people who aren't in front of your machine.

Expo Go link

If your testers have Expo Go installed, share the project link directly and let them run it live on their own device. No build, no App Store, no waiting.

THE ONE-QUESTION FEEDBACK LOOP

Show the flow to 2-3 people and ask exactly one question: **"Where did you hesitate?"** Hesitation points to confusion, unclear affordances, or a flow that doesn't match how people actually think about the problem.



06 · RECAP

Don't drift

Three ways a 1-day MVP quietly turns into a 3-day MVP.

- a Scope creep.** The number one killer of a 1-day timeline. It rarely shows up as one big decision — it's a dozen small “well, while I'm in here” additions that each cost 10 minutes and together cost your whole afternoon.
- b Chasing design polish instead of validating the flow.** It's tempting to spend an extra hour making a screen look right, especially once it's close. The question today is whether the flow works and whether people understand it — not whether the shadows are exactly right.
- c Picking Supabase, Swift, or other tooling this guide is built to avoid.** Legitimate choices on a longer timeline — but under a 1-day constraint, any setup time spent on infrastructure or native tooling is time you don't get back.

If the MVP validates

- 1 Idea to App Store** — the combined guide. Start here: its **Validation** and **Product Spec** stages are the natural next stop, because “2–3 people reacted well” is not yet demand, and the spec is what every later stage reads from.
- 2 Designing UI with AI** — the full design workflow: a real design language, multiple variations, accessibility checks, and the manual polish pass you skipped today.
- 3 The AI Coding Stack** — the full build workflow: proper Figma Design files, a real database and security rules, EAS builds, and the path to an App Store launch.

Today's MVP was built to answer one question fast. Everything from here is about building the real thing.