

THE SHIP METHOD



CODE off



CUSTOMERS on

CODE OPTIONAL

How Anyone Can Build AI Apps That
Make Real Money

Code is optional.
Customers aren't.

KACPER JANIK
OSKAR MROCZEK

CODE OPTIONAL

How Anyone Can Build AI Apps That Make Real Money

THE SHIP METHOD

Kacper Janik
Oskar Mroczek

Copyright

Code Optional: How Anyone Can Build AI Apps That Make Real Money

Copyright © 2026 Kacper Janik and Oskar Mroczek. All rights reserved.

No part of this book may be copied, shared or resold without written permission from the authors, except for short quotations in reviews.

First edition, 2026.

This book is for information and education. It is not legal, financial, tax or security advice. The results described in it are examples, not promises: most apps never earn much, and yours may not either. Revenue figures are labelled by source, as explained in Chapter 1. Anna and SlotFill are fictional illustrations.

Product names and trademarks belong to their owners. The authors are not affiliated with the companies named in this book.

AI tools, prices and rules change quickly. The information here was accurate to the best of the authors' knowledge at the time of writing.

Introduction. The SHIP Method

For most of the history of software, there was a wall between people with ideas and people who could build them. On one side stood millions of nurses, shop owners, teachers, marketers, coaches, receptionists and accountants, each of whom knew a problem better than any engineer ever would. On the other side stood a small group of people who could write code.

If you were on the wrong side of that wall, your options were limited. Learn to code, which takes years. Hire someone, which takes money and the ability to judge their work. Or give up, which is what most people did.

That wall has fallen. Today you can describe an app in plain English and watch an AI build it in front of you. Ordinary people with no technical background are launching real products and, in some cases, earning real money from them.

But here is what the excited headlines leave out. Most apps never earn much. According to RevenueCat, which processes subscriptions for tens of thousands of apps, most apps fail to reach even \$1,000 in monthly revenue within their first two years. More than 14,000 new apps launch every month. The tools have made building easy for everyone, which means building alone is no longer an advantage for anyone.

This book exists because of that gap: between how easy it has become to build an app and how hard it still is to build one that people pay for.

What This Book Promises

By the time you finish this book, you will be able to:

- **Find a problem worth solving**, one that people already spend money, time or frustration on.
- **Prove that people will pay** before you build anything, in about a week.
- **Build a first version with AI**, without writing code, and make it safe enough for paying customers.
- **Get your first hundred users** without an advertising budget, and set up ways for new ones to find you.
- **Price it, measure it and decide its future** with a handful of honest numbers.

The finish line of this book is not a finished app. It is your **first paying customer**, and a clear plan for what comes next.

Who This Book Is For

This book is written for people who do not code and do not plan to learn. Employees with an idea they cannot stop thinking about. Freelancers and small business owners who see a tool their industry is missing. People who want a second income, or a way out of a job, and are tired of hype.

If you are a programmer, you will find the building chapters simple, but the rest of the book, which is about customers, distribution and money, may be exactly what you have been missing.

The No Hype Contract

Books about making money online have a reputation problem, and they earned it. Many are built on screenshots nobody can

check, stories that only include winners and promises nobody can keep. This book makes you four promises instead:

1. **Every revenue figure is labelled.** You will always know whether a number was verified, reported by the media, reported by a tool's maker, or reported by the founder alone.
2. **Base rates come with success stories.** When you read about someone who made \$30,000 a month, you will also read how rare that is.
3. **Failures and risks are included.** Security problems, viral spikes that faded, companies that were praised and then investigated. You learn as much from them as from the winners.
4. **Illustrations are marked.** Throughout the book you will follow Anna, a front desk manager at a physiotherapy clinic who builds an app called SlotFill. Anna is a fictional illustration, and every time she appears, the book says so. Her steps are real methods; her numbers are examples.

The SHIP Method

The book follows one simple method, named after what you are going to do: ship. Every letter is one stage, and every stage has two chapters.

The SHIP method: four letters, nine chapters, one paying business

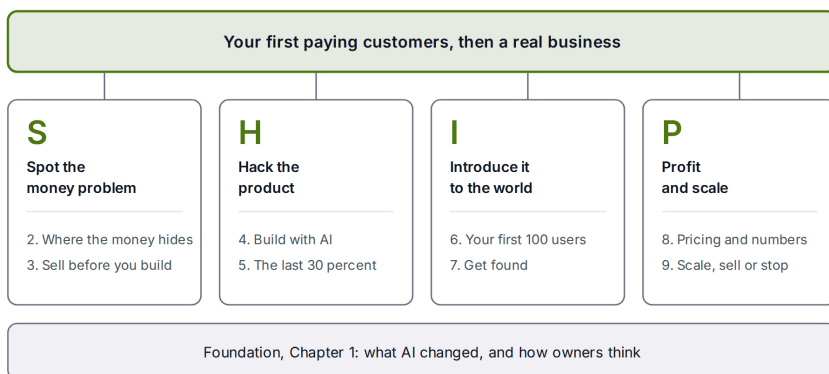


Figure 0.1 · The SHIP method

- **Chapter 1, the foundation**, explains what AI changed, what it did not change, and why the people who win think like owners rather than builders.
- **S: Spot the money problem.** Chapter 2 shows you where money hides and gives you a scorecard to choose the right problem. Chapter 3 shows you how to sell the promise before you build, and how to tell real interest from polite compliments.
- **H: Hack the product.** Chapter 4 turns your plan into a first version with AI tools. Chapter 5 covers the last 30 percent that separates a demo from a product: testing, security, costs, payments and rules.
- **I: Introduce it to the world.** Chapter 6 gets you your first hundred users by borrowing attention from people who already have it. Chapter 7 makes sure new users can find you in app stores, search engines and AI assistants.
- **P: Profit and scale.** Chapter 8 covers pricing and the five numbers that tell you the truth. Chapter 9 helps you decide whether to scale, sell, fix or close.

Notice the proportions. Building, the part most people think is the whole job, takes two chapters out of nine. Finding the right problem, proving demand, reaching people and making money take the rest. That is not an accident. It is where the difference between hobbies and businesses really lies.

The 1-30-90 Rule

One rule runs through the whole book. Write it somewhere you will see it every day:

One problem. Thirty days to your first paying user. Ninety days to decide whether to grow it or close it.

One problem, because focus beats talent. *Thirty days*, because a deadline turns a dream into a plan, and because with today's tools, a month is enough to find, test, build and sell a first version to someone. *Ninety days*, because you deserve an honest answer, and because time spent on an app that will not work is time taken from the one that will.

The Conclusion turns this rule into a day by day plan.

How to Use This Book

Every chapter follows the same pattern, so you always know where you are:

- **A story and a law.** Each chapter opens with a real case and one rule in bold, the idea to remember if you forget everything else.
- **The method.** Step by step, with figures you can photograph and keep.
- **Prompts.** Ready to copy instructions for AI tools, numbered through the book and collected in Appendix B.

- **A worked example.** Anna applies the chapter to her own business.
- **Mistakes to avoid.**
- **Ship It Now.** A short checklist of tasks. This is the most important part of every chapter. Reading changes nothing. Doing changes everything.

One more thing. AI tools change every few months. Names, prices and features in this book were accurate when it was written, but some will be out of date by the time you read it. That is why the book teaches principles, not button clicks, and why the current list of recommended tools is kept up to date for readers of the Code Optional newsletter, as Appendix C explains.

The wall is down. Let's see what you build on the other side of it.

Notes

- Base rates for apps: [TechRepublic on RevenueCat 2025](#); [RevenueCat, State of Subscription Apps 2026](#)

Chapter 1. The Great Unlock

“The hottest new programming language is English.”

Andrej Karpathy, AI researcher, January 2023

In February 2025, a growth marketer in Brazil sat down to build an app.

Sabrine Matos had no technical training. She had never shipped software. What she had was a problem she understood better than most engineers ever would: women meeting strangers from dating apps had no simple way to check who they were about to meet.

She opened an AI app builder called Lovable and started describing what she wanted, in plain words.

Forty five days later, Plinq was live. Within three months it reportedly had more than 10,000 users and an annual revenue run rate of about \$456,000.

Read that again. A marketer with no technical training, a problem worth solving and a laptop.

Now ask yourself one honest question: **what was stopping you?**

For most of the history of software, the answer was simple. You could not build it. You needed a programmer, and programmers were expensive, busy and hard to judge. Good ideas died in notebooks because the people who had them could not turn them into products.

That wall is gone. This chapter has two halves. The first shows what really changed, and what did not. The second shows how to think about it: not like a builder, but like an owner. People who get both halves right are the ones who get paid.

The Law of the Unlock

Building an app is no longer the hard part. Getting people to pay for it is.

This is the most important sentence in this chapter, and maybe in this book. Everything that follows rests on it.

AI has made building cheap, fast and open to anyone who can describe a problem clearly. That is the good news. The other half is that it is now cheap, fast and open to everyone else too. When everybody can build, building stops being an advantage. What remains scarce is the thing that was always scarce: a problem people care about, and a way to reach them.

So here is the deal this book makes with you. You will learn to build, because you must. But you will spend just as much time learning to find the right problem and the right customers, because that is where the money is.

The Numbers Nobody Puts on the Cover

Most books in this genre would stop at Sabrina's story and tell you that you could be next. This one will show you the full picture, because you cannot make good decisions with half of it.

Start with how big the unlock really is. Lovable, the builder Sabrina used, reported about 1 million new projects every week by June 2026, and said its users are primarily non technical. Its annual revenue run rate grew from \$400 million in February 2026

to \$600 million in September 2026. Those are not engineers. Those are people like you, building.

Now look at the other side. RevenueCat, a company that processes subscriptions for tens of thousands of apps, publishes some of the best data on what apps actually earn. Its 2025 report covered more than 75,000 apps. The finding was blunt: **most apps never reach \$1,000 in monthly revenue within their first two years.** Even the top 5 percent of new apps were earning around \$8,880 two years in. The bottom quarter earned \$19 or less.

The 2026 report, covering more than 115,000 apps, added two numbers that matter for anyone building with AI. More than 14,000 new apps launch every month. And AI apps earned about 41 percent more per user in their first year than other apps, but their monthly plans retained users 36 percent worse over twelve months.

Put those facts side by side and the shape of the opportunity becomes clear:

- **The tools work.** Ordinary people are building real products at a scale that did not exist two years ago.
- **The market is crowded.** Thousands of new apps compete for attention every month.
- **Most apps earn almost nothing.** Not because they are badly built, but because nobody needs them or nobody finds them.
- **AI apps can earn more per user.** But they must work harder to keep those users.

This is not a reason to give up. It is a reason to play differently from the crowd. The crowd starts by building. You will start by finding a problem that people already pay to solve, and you will plan how to reach them before you write your first prompt.

That is my promise to you, and my contract with you: no hype, no fantasy income screenshots, no stories without sources. Every number in this book tells you where it came from, so you can judge it yourself.

Three Walls That Fell, Two That Did Not

For decades, three walls stood between an idea and a product. In the space of about two years, all three came down.

Wall 1: Skill

You used to need to speak the computer's language. Now the computer speaks yours.

In February 2025, Andrej Karpathy gave this new way of working a name: *vibe coding*. You describe what you want, the AI writes the code, you look at the result and ask for changes. By the end of the year, Collins Dictionary had made it the Word of the Year. Even professionals changed how they work: a quarter of the startups in Y Combinator's Winter 2025 batch reported codebases that were 95 percent written by AI.

The skill that matters now is not syntax. It is clarity. **If you can explain a problem clearly to a smart stranger, you can explain it to an AI.**

Wall 2: Cost

A first version of an app used to mean hiring a freelancer or an agency and paying for months of work before you knew whether anyone wanted it. Today the first version costs a tool subscription and your time. That changes the math of risk completely. When a test is cheap, you can afford to run many of them.

Wall 3: Time

Sabrine went from idea to live product in 45 days. Hasaam Bhatti, whose story you will meet in a moment, reported building his first version in 48 hours. A cycle that once took a quarter of a year can now fit into a weekend.

Speed is not just convenience. It means you can learn from real users while your competitors are still writing specifications.

The walls that are still standing

Two walls did not move, and most beginners walk straight into them.

The demand wall. AI can build anything you describe. It cannot tell you whether anyone will pay for it. That is your job, and Part I of this book is about doing it well.

The trust wall. A builder gets you to a working demo fast. It does not guarantee a product that is safe to put in front of paying customers. In July 2026, Zapier’s review of the leading builders concluded that they get projects only about “70% of the way” to launch, because of bugs and security issues. A 2025 study of 1,645 apps made with Lovable found 170 with flaws that could let strangers access data they should never see. The last 30 percent is where amateurs get hurt, and Chapter 5 will show you how to cross it safely.

The walls to building fell. The walls to selling did not.

FELL IN THE LAST TWO YEARS

Skill

Describe it in plain words.
AI writes the code.

Cost

A first version costs a
subscription and your time.

Time

Idea to live product in
days or weeks, not months.

STILL STANDING

Demand

AI cannot tell you whether anyone will pay.
That is your job. See Part I.

Trust

Builders get projects about 70% of the way.
The last 30% is on you. See Chapter 5.

Figure 1.1 · Three walls fell, two still stand

The top row is why you can start today. The bottom row is why most people who start never get paid.

Proof, With the Labels On

Every revenue figure in this book carries one of four labels, so you always know how much weight it can bear:

- **Verified:** confirmed by a payment platform or an independent data source.
- **Media reported:** published by a news outlet that did its own reporting.
- **Vendor reported:** published by the company that sells the tool. Useful, but it is also marketing.
- **Founder reported:** shared by the founders themselves. Often honest, never audited.

Sabrina's numbers at the start of this chapter are *vendor reported*: they come from Lovable's own blog. Keep that in mind as you meet three more builders.

Hasaam Bhatti: the deal that mattered more than the code

Hasaam describes himself as a non technical founder. He built Launch Fast, a research tool for Amazon sellers, using an AI coding tool and a few standard services for hosting and data. He reported having a first version in 48 hours, \$10,000 in monthly recurring revenue by day 30, \$21,800 by day 90 and about \$30,000 a few months later.

The interesting part is not the 48 hours. It is what he did before launch. He made a deal with an existing coaching community for Amazon sellers: a share of the business in exchange for access to its members. On day one, his app was in front of exactly the people who had the problem it solved.

Label: founder reported, Indie Hackers, June 2026.

Therese Waechter: the small business owner

Therese runs Otto's Grotto, an online shop. She needed a wholesale catalog for her business and built it herself with an AI tool instead of hiring a developer. She told Business Insider that it helped her double her revenue.

Her app was never meant to be a startup. It was a tool that made her existing business earn more. That is a path many readers of this book will find faster and safer than launching something new.

Label: founder reported, via Business Insider, August 2025.

Pieter Levels: the spike that is not a business

In early 2025, Pieter Levels built a simple flight game in the browser with heavy help from AI. In 17 days it reached a revenue run rate of about \$1 million a year, largely from sponsors paying to appear inside the game.

It is a famous story, and it is easy to learn the wrong lesson from it. Pieter is an experienced developer with a very large following online. The game took off in large part because so many people were already watching what he built. Even he was careful about it: “Whether the MRR is truly sustainable for a year we can only guess.”

Label: founder reported, levels.io, March 2025.

The pattern

Look at what these stories have in common. In each one, **the build was fast, but the reach decided the result**. Hasaam borrowed a community. Pieter brought an audience. Therese already had customers. Sabrina picked a problem she understood deeply and knew who had it.

None of them won because their code was better. They won because the right people saw the product at the right moment. Remember that when you are tempted to spend another week polishing a feature.

From Builder to Owner

That is the first half of the picture: the walls, the numbers, the people. The second half is about you.

Before you read on, answer three questions. Be honest. Nobody is watching.

1. Do you believe you need to learn to code before you can build something real?
2. Do you believe you need savings, a loan or an investor before you can start?
3. Do you believe you need a brilliant, original idea that nobody has had before?

If you answered yes to even one of them, keep reading carefully. Those three beliefs have stopped more people than any lack of talent, time or money. They feel like common sense. In the age of AI, they are simply out of date.

The people who make money with apps do not think like programmers, and they do not think like inventors. They think like **owners**. An owner does not ask “what can I build?” An owner asks “who has a problem, what is it costing them, and what would they pay to make it go away?”

That question rests on a simple rule, the second one in this chapter:

Nobody pays for your idea or your code. People pay to get rid of a problem.

Your idea is worth nothing until someone with a problem says “yes, I want that.” Your code is worth nothing until it removes that problem better than what they use today. The only thing that turns either of them into money is a customer. Almost every beginner behaves as if the opposite were true. They guard their idea like a secret, polish their app for months and only then go looking for someone to sell it to. Owners do it the other way round.

Four Myths That Keep Beginners Stuck

Every myth below was once true, or close to it. That is what makes them dangerous.

Myth 1: “I have to learn to code first”

Ten years ago this was a fair point. Today it is the most expensive delay you can choose. You have already seen why: the skill wall has fallen.

What you do need to learn is different and much faster: how to describe a feature clearly, how to test whether it works, and how to decide what not to build. You will practise all three in Part II. None of them requires a computer science degree.

Myth 2: "I need money to start"

Even before AI, this myth was weaker than it looked. For *The \$100 Startup*, Chris Guillebeau studied more than 1,500 founders who built businesses that started with less than \$1,000.

Today the cost of a first version has fallen further: a tool subscription and your evenings. What you are really short of is not money. It is **attention**: yours, which you must protect, and your customers', which you must earn. Remember Hasaam Bhatti. His key move was not raising cash. He gave a community a share of his business in exchange for access to its members.

Myth 3: "I need a genius idea"

Look at the businesses you met in this chapter. A research tool for Amazon sellers. A wholesale catalog for a small online shop. A safety check for women meeting strangers. None of them was a new invention. Each solved a known, painful problem for a clearly defined group of people.

The genius is not in the idea. It is in the fit between a problem and the people who have it. In fact, if nobody else is solving your problem, be suspicious. Competitors are often the best proof that people pay for a solution.

Myth 4: "If I build it, they will come"

This is the most expensive myth of all, because you only discover it is false after you have built the thing. More than 14,000 new apps launch every month, and most never reach \$1,000 in monthly

revenue. Nobody is waiting for your app. Nobody is searching for it. **If you do not bring people to it, nobody comes.**

That is why Part III of this book, getting your app in front of people, is as long as Part II, building it.

Swap each myth for the owner's version

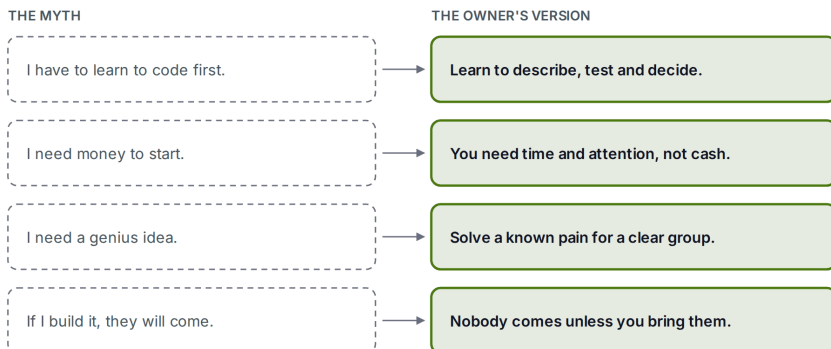


Figure 1.2 · The myth and the owner's version

Take a photo of this figure. Every time you catch yourself thinking like the left column, switch to the right.

Builders Draw a Line, Owners Draw a Loop

The difference between a builder and an owner is not talent. It is the shape of how they work.

A builder works in a straight line. Idea, then months of building alone, then polishing, then launch, then hope. The first honest feedback arrives at the very end, after most of the time and energy is already spent. If the answer is “nobody wants this,” it is too late to change much.

An owner works in a loop, and goes around it again and again:

Builders draw a line. Owners draw a loop.

THE BUILDER'S LINE: FEEDBACK ONLY AT THE END



THE OWNER'S LOOP: FEEDBACK EVERY TURN

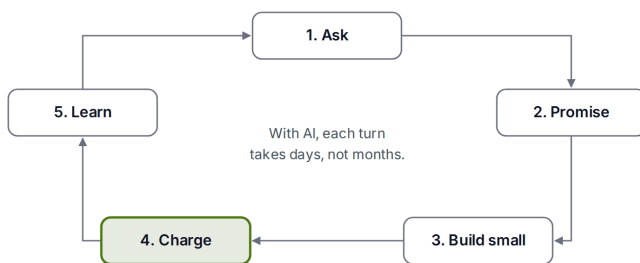


Figure 1.3 · The builder's line and the owner's loop

1. **Ask.** Talk to people who have the problem. Not about your idea, about their day: what annoys them, what they have tried, what they pay for today.
2. **Promise.** Describe the solution in one sentence and watch their reaction. “So it would do X for you?” A shrug is an answer too.
3. **Build small.** Build only the smallest version that keeps the promise. With AI, that can mean days.
4. **Charge.** Ask for money early. This step is highlighted for a reason: **compliments are free, payment is proof.** Someone who pays has told you the truth.
5. **Learn.** Look at what people used, what they ignored and why they paid or did not. Then go back to step one with better questions.

The loop feels slower at the start, because you are talking instead of building. It is much faster overall, because every turn removes a wrong guess. AI made the build step short. That makes the loop the natural way to work, not a luxury.

Your First Owner's Week

Mindset does not change by reading. It changes by doing a few things differently. Here is your first week as an owner: eight steps that fit into a week of evenings.

1. **Write your unfair knowledge list.** Take a blank page and list ten problems you know better than most people. Look at your job, your industry, your hobbies, your family, the groups you belong to. Sabrina's edge was not technical. It was knowing a problem from the inside.
2. **Mark the money.** Next to each problem, write who has it and what they pay today to deal with it: software, a service, their own time, a spreadsheet they hate. If nobody pays anything in any form, cross it out for now. Circle the three with the clearest money.
3. **Name your myths.** Write down which of the four myths you believed before this chapter, and copy the owner's version from Figure 1.2 next to each. Seeing your old belief in your own handwriting makes it easier to notice when it comes back.
4. **Lose your fear in one hour.** Open any AI app builder with a free plan and build something small, just for you. Do not aim for a business. Aim for the moment you realise you can do this. Use the first prompt below.
5. **Rehearse with AI.** Nervous about talking to strangers? Practise first with an AI chat assistant, using the second prompt below.
6. **Have three real conversations.** Find three people who have your strongest problem and talk to them for 15 minutes each. Do not pitch. Do not mention an app. Ask about their life, using the five questions below.

7. **Write your reach list.** List every group of people you can reach without paying for ads: communities you belong to, newsletters you know, creators you could approach, former colleagues. Hasaam's breakthrough was a deal with a community. Yours might be on this list.
8. **Write your promise and set your dates.** Finish this sentence: *"I help [who] to [get what result] without [the pain they hate]."* Then set your **1-30-90** dates: one problem, 30 days to your first paying user, 90 days to decide whether to grow it or close it. Put them in your calendar, together with fixed weekly hours for this work. Owners protect their time. Builders spend whatever is left over.

Prompt 1: your first build

Copy it, fill in the brackets and paste it into the builder:

```
Build a simple web app for [who it is for] that helps them [do one specific thing].  
The main screen should let the user [first action] and then see [result].  
Keep the design clean and mobile friendly. Do not add login yet.  
After you build it, list three things I should test.
```

Notice what this prompt does. It names the user, the job and the result. It keeps the scope small. And it asks the AI to help you test. That is the habit you will build on in Part II.

Prompt 2: your rehearsal partner

```
Act as [a specific type of person] who struggles with [the problem].  
I am going to interview you about it. Answer like a real, busy person:  
short, a bit skeptical, with concrete details from your daily life.  
Do not make it easy for me. After 5 questions, tell me which of my questions were leading or useless, and how to improve them.
```

The practice is useful, but it is not a substitute. An AI will never pay you. Only the real conversations count.

Five questions for real conversations

They work because they ask about the past, which people remember honestly, instead of the future, which people imagine generously:

- “Tell me about the last time [the problem] happened.”
- “What did you do about it?”
- “What have you tried before? What did it cost you?”
- “What is the most annoying part of it?”
- “If it went away tomorrow, what would that change for you?”

If you want to go deeper, Rob Fitzpatrick’s short book *The Mom Test* is the classic guide to asking questions that even people who love you cannot lie about.

A worked example

To see how the week fits together, imagine Anna. She runs the front desk at a small physiotherapy clinic. This is an illustration, not a real case, but every step is one you can copy.

Her unfair knowledge list includes “patients forget appointments,” “therapists rewrite the same exercise sheets by hand” and “the clinic loses track of who still owes for packages.” Marking the money, she notices that missed appointments cost the clinic real revenue every week, and that the clinic already pays for a booking system that does not solve it.

In her three conversations, with two clinic owners and one therapist, she hears the same thing: reminders exist, but patients ignore them, and nobody follows up. One owner says,

unprompted, that he would “pay tomorrow” for fewer empty slots.

Her promise becomes: *“I help small physiotherapy clinics fill cancelled slots without phoning patients one by one.”* Her reach list includes a regional association of physiotherapists and a group chat of clinic managers she already belongs to. Her 30 day goal: one clinic paying for a simple waitlist that offers freed slots to other patients automatically.

Anna has not written a line of code. She has already done the part that most builders skip.

Six Mistakes That Waste the Unlock

1. **Starting with the tool instead of the problem.** “I want to build something with AI” is not a business. “Clinics lose money on empty appointment slots” might be. Tools change every few months. Problems people pay to solve last for years.
2. **Believing the screenshot.** A revenue screenshot on social media is the most persuasive and least reliable number on the internet. Ask who reported it, whether it was verified and how many people tried the same thing and failed. You now know the base rate: most apps never reach \$1,000 a month.
3. **Pitching instead of listening.** The moment you describe your app, people start being polite. Keep your idea to yourself until you have heard about their problem in their own words.
4. **Counting compliments.** “That sounds amazing, I would totally use it” is not data. A pre order, a deposit or a real email on a waitlist is. Count commitments, not compliments.
5. **Mistaking the demo for the product.** A builder can make something that looks finished in an afternoon. Looking finished and being safe to take payments and personal data

are different things. Do not invite paying customers until you have crossed the last 30 percent.

6. **Waiting for the right moment.** You will never wake up feeling like a developer, and there is no week with enough free time. You do not need either. Book small, fixed blocks of time, protect them and start before you feel ready, because nobody ever does.

Ship It Now

Do these within the next seven days, before you read Chapter 2.

- ☐ Write your unfair knowledge list and mark the money next to each problem.
- ☐ Circle your top three problems.
- ☐ Write down the myths you believed, with the owner's version next to each.
- ☐ Spend one hour building a small app for yourself with Prompt 1.
- ☐ Rehearse once with Prompt 2.
- ☐ Have three 15 minute conversations with people who have your strongest problem.
- ☐ Write your reach list and your one sentence promise.
- ☐ Put your 1-30-90 dates and your weekly owner hours in your calendar.

In one sentence: AI has made building easy for everyone, so your advantage is no longer the code but the problem you choose, the people you can reach and your habit of asking before you build.

Chapter Notes

- Epigraph: [Andrej Karpathy on X, January 24, 2023](#)

- Sabrina Matos and Plinq (vendor reported): [Lovable blog](#), September 2025
- Lovable projects per week and users: [TechCrunch](#), June 9, 2026
- Lovable run rate of \$600 million: [TechCrunch](#), September 24, 2026
- RevenueCat 2025 report, 75,000+ apps: [TechRepublic](#)
- RevenueCat 2026 report, 115,000+ apps and 14,000+ new apps a month: [RevenueCat blog](#)
- Vibe coding, Collins Word of the Year, Y Combinator Winter 2025: [Wikipedia](#), [Vibe coding](#)
- “70% of the way” to launch: [Zapier](#), July 2026
- 170 of 1,645 Lovable apps with access flaws: [The Next Web](#)
- Hasaam Bhatti and Launch Fast (founder reported): [Indie Hackers](#), June 2026
- Therese Waechter and Otto’s Grotto (founder reported): [Business Insider via Yahoo](#), August 2025
- Pieter Levels and his flight game (founder reported): [levels.io](#); [404 Media](#), March 2025
- 1,500+ founders who started with less than \$1,000: [The \\$100 Startup](#)
- Further reading: [Rob Fitzpatrick](#), [The Mom Test](#)

Want the Rest?

This free sample ends here. The full book takes you from this chapter to your first paying customer and beyond:

- Chapters 2 and 3: where the money hides, and how to prove people will pay before you build.
- Chapters 4 and 5: building with AI, then crossing the last 30 percent safely.
- Chapters 6 and 7: your first 100 users, and how new ones find you.
- Chapters 8 and 9: pricing, the five numbers that matter, and what to do on day 90.
- All 16 prompts, the 90 day checklist, the launch checklist and a glossary.

Code Optional is available as an eBook. Look for it wherever you found this chapter.

Code is optional. Customers aren't.