

ARCHITECTURE • UX • AGENTS • TESTS • OPERATIONS

Building Platforms with AI Agents

A Hands-On Guide from Idea to a Safe Pilot Launch

WITH A RUNNABLE COMPANION PROJECT



Elton Marku

Building Platforms with AI Agents

A Hands-On Guide from Idea to a Safe Pilot Launch

{<3}

First English edition • 2026

Publication details

Building Platforms with AI Agents

A Hands-On Guide from Idea to a Safe Pilot Launch

Author: Elton Marku

Publisher: Entwicklerherz GmbH

Am Hauptbahnhof 16, 60329 Frankfurt am Main, Germany

Represented by managing director Elton Marku

Commercial register: Amtsgericht Frankfurt am Main, HRB 105176

Contact: kontakt@entwicklerherz.de

Website: entwicklerherz.de

© 2026 Entwicklerherz GmbH. All rights reserved. Use of the accompanying practice materials is subject to the terms supplied with those materials.

First English edition, 2026. Based on the German edition's editorial cutoff of October 5, 2026. Translation: October 7, 2026.

Imprint: Independently published

ISBN (paperback): 9798160921488

This book was created with the support of AI tools, including an AI-generated English translation. Selection, contextualization, and editorial responsibility rest with the author. Provider interfaces and features may change; the text identifies the scope and limits of specific checks. A prompt does not guarantee a particular result.

Auftragsplatz, its example businesses, and its characters are fictional. Real projects are identified as such. Product names belong to their respective owners; their mention does not imply a partnership or endorsement. Some genuine screenshots and the original companion starter retain German interface labels, explained in English where needed. An optional English starter follows the same learning path at entwicklerherz.de/en/buch.

Who's sitting beside you?

I am Elton Marku, a software developer with a German university degree in computer science (Diplom-Informatiker), and managing director of Entwicklerherz GmbH in Frankfurt am Main. What excites me about platforms is that moment when an idea becomes a place where people can actually do something together.

Before coding agents joined me at the computer, I programmed these applications largely myself. Later came no-code tools; today I work extensively with AI agents. I care not only about the first attractive screens, but also about the questions behind them: who may see what? Will the input survive? What happens when a service fails?

Professional platforms such as Fachzirkel and SchemaCommunity grew from that practice. Just as important were projects that progressed technically but did not continue commercially. Finished software and a working business are two different things. That experience belongs on my workbench as much as a successful launch does.

In this book, imagine me sitting beside you. We build something tangible, try it, and improve it step by step. You do not have to adopt my favorite tools or go back to university first.

Occasionally, I ask what you would expect after a click. It is not a knowledge test. Your answer turns a technical task into a picture of how your platform should work. That is exactly the picture you can give the agent.

I bring the techniques and experience. You bring the reason your platform should exist. Let's start there.

Contents

- Prologue 2**
 - You know the people. Now they need the right place. 3
- Part I - Understand and Build Your First Feature 8**
 - 1. Your Platform in One Sentence 9
 - 2. Your Coding Agent: Tools and Collaboration 13
 - 3. Start the Workbench and Open the Form 20
 - 4. Turn Input into a Preview 26
- Part II - Build the First Complete Platform Journey 31**
 - 5. Turn the Big Idea into Workable Steps 32
 - 6. Project Memory for People and Agents 37
 - 7. Edit a Preview Without Losing the Previous Version 45
 - 8. Design the Private Draft as a User Journey 48
 - 9. Save the Private Draft for Later 57
 - 10. Two Accounts Turn a Demo into a Platform 70
 - 11. Turn the Draft into a Traceable Decision 76
- Part III - From the Core Journey to an Operable Pilot 84**
 - 12. Let the Browser Handle Repetition 85
 - 13. Move the Preview Safely into the Cloud 91
 - 14. From an Error to a Lasting Fix 97
 - 15. Keep a Running Platform Easy to Change 102
 - 16. Approve a Limited Pilot 108
- Part IV - Four Optional Extensions 113**
 - 17. A Reason to Return 115
 - 18. Show Trust Without Overpromising 120
 - 19. Build a paywall without trusting the browser 125
 - 20. Follow a sensitive data path from end to end 133
- Part V — Real systems, transferable patterns 141**
 - 21. Three platforms, three useful shortcuts 142
- Part VI — Decide and apply 149**
 - 22. Keep building, change direction, or stop 150
 - 23. Your 30/60/90-day platform roadmap 154

OPENING

Prologue

You know the people. Now they need the right place.

You open your platform. Someone needs help; someone else has a useful answer. They find each other without calling you, searching a spreadsheet, or hunting for the right message in a group chat.

A small moment. But it is exactly why you wanted to build the thing.

Perhaps you have a professional community in mind, where useful knowledge does not disappear between birthday greetings. Perhaps customers need to submit documents, club members want to organize something, or people want to do business together. You know the need. What is missing is the path from “This ought to exist” to “Give it a try.”

Who we are building for

This book is for **people who want to build or develop their own platform with AI agents**. You know the people and the problem you want to solve. You do not need professional programming experience. You should, however, be comfortable installing applications, opening a project folder, and using a browser. On a company computer, you will need the appropriate permissions.

Maybe you finally want to try an idea of your own. Maybe you already run a platform and want to initiate changes yourself. Either way, we are working on the same question: how can you and an agent turn what you want into something people can actually use? You describe the goal, the agent handles much of the technical work, and we look together at what comes out.

If you already develop software, you can skip the setup. The same questions remain interesting: how do I break a request into small, workable steps? How do I stop an agent from building past my goal?

How does a first feature grow into a reliable workflow? Technical explanations appear where they help with those decisions. The main route is neither a programming course nor a collection of terminal commands.

From an idea to work on a running project

A platform connects people through shared activities: one member asks a question and another answers; someone submits a proposal and receives a decision. What looks obvious in the browser requires design, data storage, access rules, and ongoing operation behind the scenes.

These are also the building blocks of social networks: people publish content, respond to one another, and decide who may see something. This book gives you foundations and practical techniques for building your own community or specialized social network. We make those possibilities tangible through a manageable example; additional features and large user numbers bring further work later.

AI agents let you take part in that work yourself: describe a need, try the implementation, and use the result to think further. You will encounter the technical details along the way. For a question such as “Is my post still private, or can the group already read it?”, your understanding of the product is what matters. The agent can ask questions and show options; you can then decide on a basis that makes sense to you.

Before you run your first task, we will therefore clarify what a coding agent can do and how to work with one. Then we build. The first click already belongs to your future platform; we simply make the first step small enough to try straight away.

Our shared workbench: Auftragsplatz

You may have posted a classified ad or looked for help before. Our learning project, **Auftragsplatz**, is a service marketplace: small businesses describe what they need, and specialists respond with a proposal. The request might be for product photos, shop signage, or help

with a website. The platform, businesses, and people in the example are fictional.



We are building the marketplace, not the services it advertises

We are building the path from a request through a proposal to a reply. Whatever service the business and specialist subsequently agree on is delivered outside Auftragsplatz.

- **Lea runs the bicycle repair shop Radwerk Nordhafen.** The phone keeps ringing. She is looking for “Online appointments for the bicycle repair shop.” This means separate scheduling software: a calendar on her shop’s website where customers can book available repair appointments.
- **Mara is a freelance web developer.** She finds the request on Auftragsplatz and explains how she would help Lea. She prepares that text on the marketplace and sends it to Lea.
- **You build and run Auftragsplatz.** Your application keeps a record of their exchange. The scheduling software is the reason for the request, not a feature of the marketplace. Just as a classifieds site does not manufacture bicycles, we are not building a workshop calendar here.

Lea’s request is ready as an example. We start with **Mara’s response to it**: a button opens a form, and her input becomes a preview. Later, the proposal can be saved, submitted, and answered. Whether the request concerns a calendar or product photos makes no difference to those features.

As you build along, you create reusable building blocks. Mara’s private proposal could be a draft post in your community; Lea’s answer could be a reply to a classified ad. We adapt the rules to the purpose. Your learning page starts almost empty; the technical framework and reference pages are ready for comparison.

The route through the whole book

Chapters 1 to 4: understand the basics and get something running.

You learn how a platform fits together, choose an agent, and learn to describe a task. Then you start the workbench. A button becomes a form, and the form becomes a preview you can check. By the end, your own small feature responds to your input.

Chapters 5 to 8: build further with intent. You break down larger requests, record the project state, and add editing and cancellation. Only now do we improve the user journey and usability on small screens. We are improving something that already works.

Chapters 9 to 11: an interface becomes a platform. Mara gets a real account; her proposal is stored and belongs to her. A second account makes access rules visible. Submission and Lea's decision follow. Now one person's actions genuinely change what another person can do.

Chapters 12 to 16: release with confidence. Automated browser tests take over repeated checks. You deploy the tested version to a cloud preview, fix errors, and keep the project maintainable. This becomes a limited pilot for real users.

Chapters 17 to 20: extend deliberately. Notifications, meaningful trust signals, a paywall, and a sensitive AI data flow extend the platform. Not every idea needs all of them. Concrete additions show how to connect more services without damaging the existing journey.

Chapters 21 to 23: apply it to your own project. Real platform examples provide useful shortcuts. You decide what to keep building, change, or stop, and plan your next 30, 60, and 90 days.

The thread throughout is Mara's response to Lea's request: **start, view, edit, save, submit, respond, and operate**. New technology joins us when it enables one of those actions. Search, profiles, and publishing further requests can follow; for the first steps, Lea's request is already there.

How to work with this book

To build along, use the companion package **auftragsplatz-workshop-v0.5.3.zip**. It contains the prepared project framework and reference

pages for comparing your results. Keep it on your computer. You will find the book companion at entwicklerherz.de/en/buch. The exercises in this edition use version 0.5.3 so that your starting point matches the instructions.

The starter and some genuine screenshots use German interface labels. Where a label matters for a step, this edition explains its English meaning alongside the original wording. Product names, filenames, and commands stay unchanged so that what you read still matches what you open.

To begin, you need your computer, a browser, and access to Codex or Claude Code. One agent is enough. Cloud accounts, a database, and a payment provider arrive in their respective chapters; they are not hidden entry requirements for the first exercise.

The practical steps in the main text take you to the next visible result. A **closer look** offers additional technical understanding, not a hidden prerequisite. Important product promises, however, are something we check together: if the screen says “private,” the text really must remain invisible to the wrong account.

One question is enough for the next page: who wants to do something on your platform that helps someone else? Keep that person in mind. They will help us notice whether we are building the right feature.

PART I

Understand and Build Your First Feature

1. Your Platform in One Sentence

"I want to build a community." Perhaps you can already picture the people: finding one another, asking questions, and helping each other. Let's begin with one such encounter. What could the first person do that creates something useful for a second?

Before we open the tools, we will capture that moment in one sentence. It does not need to contain a complete business idea. It just needs to show us where to start.

More than a website with a login

Think of a group organizing something together. The question is in the chat, the answer is in an email, and the current list is probably called "final_new_2." A platform brings those activities together. Think of it as a **digital meeting place with a memory and ground rules**: people do things together, and later they can still see what happened.

More precisely, an **online platform** connects several participants, their content, and their actions under shared rules. What one person does changes what others can see or do. That shared state survives closing the browser.

In a community, a question meets an answer; in a marketplace, an offer meets an inquiry. In an internal portal, a request leads to an approval. What connects them is less the login than what people do together afterward.

Our sentence for Auftragsplatz is:

On Auftragsplatz, Lea publishes a request. Mara responds with a proposal, and Lea accepts or declines.

We can work with that: Lea needs help; Mara offers it. The **request** describes Lea's need; the **proposal** is Mara's response. The desired

scheduling software remains the subject of the request. Our job is the exchange on the marketplace: both people should be able to return to the same conversation state.

Three building blocks behind a click

Mara wants to start her proposal today and continue tomorrow. For the Save button to keep its promise, three building blocks must work together. The pyramid shows them from the visible interface down to data storage. The arrows show the outward and return journey of an input:

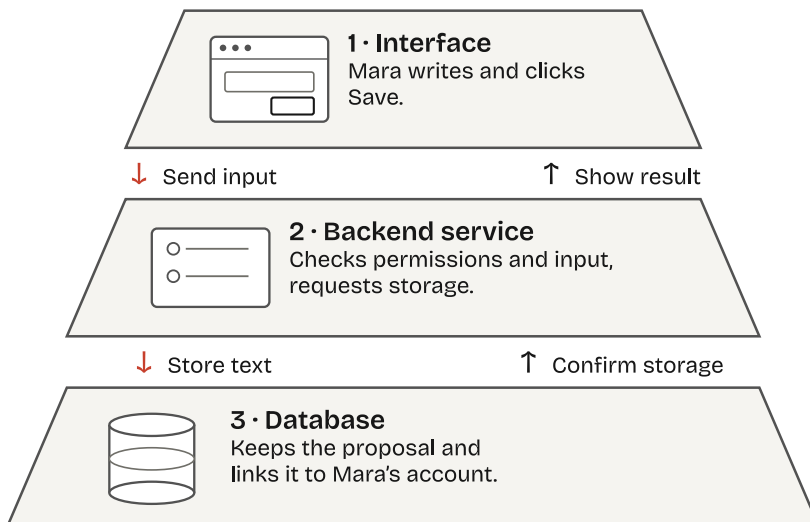


Figure 1: The three layers of Auftragsplatz. Input travels down; confirmation travels back up.

The interface is what Mara uses. She writes her text here and clicks Save. Fields, buttons, and cards can be reused as components. The interface, also called the frontend, forwards her input and later displays the result.

The service behind it processes her request. It runs on a server: a computer that handles requests from the application. This is where the rules live. Is Mara signed in? Does the draft belong to her? Is the required information present? If everything checks out, the service

arranges for the data to be saved. This part is also called the backend. If something is wrong, it returns a useful error message instead.

The database keeps the proposal. It records what Mara wrote and which account owns it. Only after a successful save does the service tell the interface “Saved.” When Mara returns tomorrow, the service checks her access, reads the text from the database, and sends it back to the interface. A form becomes a place where she can pick up her work.

What we build ourselves

You will meet these three tasks in community posts, classified ads, and customer inquiries too. **The shared item, its participants, and its visibility** remain the core questions: who may contribute, and who gets to see it?

These technical tasks do not have to live in three separate programs. A **tech stack** is the selection of tools used to implement them. To start, their cooperation matters; the tool names can wait.

External services join in when needed, for email or payments, for example. The application asks them to do something and processes their responses. You do not have to build a mail service or a bank to offer those features.

Auftragsplatz grows into this picture step by step. First we build only in the browser: Mara enters something and sees a preview. The complete storage path in the diagram arrives later, along with real accounts and access rules. We meet each building block when it has a concrete job to do.

Our first small building step

Lea’s request is already prepared for the introduction. Mara now gets a **Vorschlag vorbereiten** button: Prepare proposal. It opens two input fields. Next we build a preview so she can look at her text before anything is sent. We add registration and payment when the exchange needs them.

We keep building on that button. Later, the input is saved, submitted, and answered. You do not have to throw away your first result after the exercise; it becomes part of the platform, piece by piece.

For your own idea, consider the equivalent action: preparing a post, describing a need, or recording an inquiry. You do not need to type anything yet. In the next chapter, you will first meet the tool that helps us turn this wish into a working feature.