Engineer

Your software as tools for AI agents

Expose selected software operations as described tools for AI agents, with permissions, contract checks and a planned rollout.

Give AI agents a defined way to use the business software you already run. We identify suitable operations, describe their inputs and results, and apply permission checks before exposing them as tools. Changes are planned alongside the existing application, with testing and a rollout agreed for its operational constraints.

Turn the service into a result

Make “Your software as tools for AI agents” serve a business outcome. We can connect the technical work to the users, operations and decisions it needs to support. Your existing software opened up as tools an AI agent can call, without rewriting anything that already works.

Explore our services Get in touch →

Examine the relevant work

Use the related work on this page to examine our approach. Ask about the tradeoffs and the conditions behind each result. Designed a JSON passthrough from Postgres to API — JSON straight from the database to the browser

Designed a JSON passthrough from Postgres to API Get in touch →

Unpicked: the rules out of the screens

The rules that run a business hide in the software: in a form’s validation, in a stored procedure, in a report nobody dares to change, in the head of the person who built it. We find them, write them down and move each into one layer — often the database itself — where every caller gets the same answer. The screen, the agent and the partner integration then use the same implemented rules instead of maintaining separate versions.

Exposed: described tools an agent can call

Each operation an agent should have is offered as a tool named for the work — book a slot, quote a price, release an order — with typed inputs, errors an agent can act on, and idempotent calls so a retry never books twice. The description is an OpenAPI document generated from the code, and an MCP server offers the same operations to the agents that speak it. Authentication is OAuth, so an agent acts for a user without a password changing hands, and only within that user's rights.

Roll out operations in controlled steps

We expose selected operations alongside the existing application and compare the new path with the current behavior. Slow work can move onto a queue where appropriate. Dependencies, migration needs, maintenance windows and rollback are assessed before rollout. Existing screens remain available where the agreed architecture allows; any interruption is planned with the people who run the service.

An agent believes **every word of a tool's description** — so the description has to come from the code.

A person notices when a manual is out of date; an agent acts on it.

Generate the contract and check for drift

Where the architecture supports it, we generate the description from the database and API definitions and compare the published contract with the implementation. Checks identify mismatches in the covered operations and make compatibility changes visible in review. The tests and their coverage are part of the handover; a generated description alone does not prove every behavior correct.

The work behind it

We have opened up large systems this way. We generated an API contract from the database out, with a guard at every hop, and migrated an API of 760 operations at 100% parity with its published description. We designed a JSON passthrough from PostgreSQL to the API, built a platform in clean layers with its business rules in one place, moved slow work onto a job queue, and built the operator back office that runs it without database access.

How an engagement runs

It starts with a list: what your people do in the software every day, ranked by how often and how much it costs them. We take the first few operations, find their rules, and expose them as described tools behind tests and permissions, running beside the screens rather than instead of them. Then the next few. Your team keeps the code, the tests and the description; the tools can be called by an agent built for your trade, by the assistants your staff already use, or by a partner's system. It is bought like the rest of our work, on the terms in what it costs.

If your software only works when somebody clicks it, tell us what it is.

The software you run, made into tools agents can call

Unpicked

The rules hidden in screens, forms and stored procedures found, written down and moved into one layer.

Exposed

Each operation offered as a typed, described tool over an API or an MCP server, acting only within its user's rights.

Kept running

A scoped integration can expose one operation at a time alongside existing screens; application changes are agreed after reviewing its interfaces.

Ask about your software

Your software already does **what an agent needs** — it has to be asked in words an agent can read.

It starts with a list of what your people do in it every day, ranked by what it costs them.

Ask where to start What it costs →

Scope the work before commissioning it

Describe the workflow, the existing interfaces and the users who need the change. We can define a testable delivery and a handover that lets your team maintain it.

See what it costs Get in touch →

Talk through your requirements

Tell us what you need from “Your software as tools for AI agents”, including the deadline and the team that will maintain it. We can discuss a practical way forward.

Explore our services

welcome@engineer.company +45 50 10 61 84 Copenhagen, Denmark

Take this page with you

Contact card

Carefully Crafted With Love by D.

CVR 45969703 · Tagensvej 105C, 3. th, 2200 København N

Copyright © 2025 – present · Engineer ApS · All rights reserved.

Printed from https://engineer.company/ai-tools/