bahn

Prompt Operations Hackathon, 24 to 25 October 2026

THE CAR DELIVERY CONTROL ROOM

A challenge from Bahn

Who we are

Bahn delivers used cars for car dealers. When a dealer buys a car, often in another country, one of the freelance drivers in our network collects it from the seller and drives it to the dealer. This year that is over 25,000 cars and more than 1,500 drivers, across Finland, Denmark and Germany, run by a team of about 50.

We deliver in about 2 days. A truck takes 2 to 4 weeks, because it waits until it has enough cars on one route to fill the truck. Our drivers drive each car on its own, so nothing waits. When cars arrive sooner, dealers sell them sooner and make more money. So the faster we deliver, the more business we get.

What our operations team does

Our operations team runs every delivery. Their job is phone calls, emails and paperwork, and something changes every hour. We move cars across borders with freelance drivers, and the people the team depends on are not our staff: the seller who has the car, the driver, the ferry line, the person who signs at the other end. Any of them can be late, wrong or silent, and we are the ones who answer for the delivery.

A normal week has things like these in it:

- A driver flies to Germany, misses the train, and cannot make the pickup slot.
- The seller said the car was ready. The driver arrives and it is not.
- The seller wants a paper nobody mentioned before the car can leave.
- The ferry is sold out, so the driver needs a later ship and maybe a hotel.
- An order says Tampere in the delivery address and Helsinki in the notes.

None of these is hard on its own. The hard part is that each one starts a chain. A missed ship becomes a new booking, a hotel question, an email to the ferry line and a decision about the cost, and someone has to hold the whole chain in their head. Today that someone is a person in operations.

Where we are going

Our first portal was built for people doing the work by hand. It grew into a system of forms, and more forms, and it got harder to use every month. We do not want to build that again.

Our agents already do a lot of the routine work. They get the gate codes and papers sorted before a driver turns up, they price jobs and offer them to drivers, and they spot problems while a car is moving. The next step is agents that do most of the work in the background, without a person, and bring a person only the cases that need one.

That changes what the interface is for. A person should see only the orders that need them, and for each one, just enough to decide: what happened, what the agent already found out, what it suggests, and a way to accept it or change it. People make the judgement calls, and the agents do the legwork.

Here is what that looks like. A red warning light comes on, and a driver stops at a motorway service area and writes to us in the app. Before anyone at Bahn sees it, the agent asks the driver if they are safe, where exactly they are, and for a photo of the warning light. It looks up the car: a Bentley owned by the dealer, due in Hamburg tomorrow. Then it brings a person one decision: the car is stopped and the driver is safe, so call the dealer and ask whether they send their own roadside assistance or want the car towed, and expect tomorrow's delivery to be late. The person makes the call, or tells the agent what to do instead.

What to build

Two things.

1. The control room. A place where people and agents work together to get every car delivered. It shows the orders that need a person, says why, and keeps the rest out of the way. A person can open any order and see what is going on in a few seconds: what happened, who said what and when, what the agent has done, and what the next decision is. Show how people and agents work together: who does what, how they hand things over, and what happens when something goes wrong.

2. One real agent for one area. Go deep on one part of the business and build a real agent: one that uses tools and that a person can talk to. It should do the cheap work first, reading, asking and looking things up, and then either finish the job or bring a person a clear decision. Some areas, with what a team could build in two days, or propose another one that fits car delivery:

- **Pickup readiness, by phone.** An agent that calls the seller or the pickup place, confirms the car is ready and the papers are there, deals with phone menus, and tries again later if nobody answers.
- **Driver travel.** A car is at a private address in small-town Germany and has to be in Helsinki. Plan how the driver gets there and how the car gets home: trains, buses, a taxi for the last stretch, the ferry, flights and hotels. Compare prices and times, then book them, with a browser if you like, but stop before any real purchase. Made-up prices and timetables are fine.

- **On the road.** Breakdowns, missed ships and delays. Gather the facts, ask the driver, and propose the next step to a person.
- **Driver support.** Drivers write in with questions all day. An agent that handles the routine ones and hands the hard ones to a person with the context already gathered.

Not sure which to pick? Ask us on Discord.

The rules of the game

- The agent never contacts a customer on its own. A customer is anyone who pays us. Drivers and pickup places are fine to contact.
- Start with approvals. Build it so a person says yes before the agent changes anything: a booking, a date, a cost, or a message to a customer. Asking questions and looking things up need no yes. Build it so you could give the agent more freedom later as it earns trust.
- The agent has to be real: a language model doing the work in a loop. Everything around it can be fake. Mock the phone, the ferry prices, the map and the driver's replies. A simulated driver who answers in the demo is fine.
- The control room is for people in operations, not engineers. It must not look like a developer tool.
- Everything in the packet is made up. Keep it that way: no real people, no real messages, no real bookings.

What you get

A fully synthetic but realistic dataset of 100 orders with dealers, customers, drivers, cars, plans, offers, driver bookings, travel, conversations, alerts, documents and an event history. Some of it is messy on purpose, because the real world is. A message can disagree with a stored address. An alert is a reason to look, not a diagnosis. A quote for a ferry is not a booked ferry. There are more odd pickups and private handovers in it than in a normal week, so do not read the row counts as how often things happen at Bahn.

- `dataset/`: the data. Start with `dataset/README.md` and `dataset/SCHEMA.md`.
- Fourteen proposals that our own agents made on eleven of those orders. They are simplified examples of what agent work can look like. Your agent does not have to look like them.
- `attachments/`: 58 made-up documents, receipts and photos linked to the records.
- `starter/`: an optional Next.js, TypeScript and Postgres app that loads the data and shows a board, an order page, a support inbox, and customer, driver and place pages. Keep it, change it or throw it away. Its board of lanes is how people have worked so far, not the answer.
- `OPERATIONS-CONTEXT.md`: how the work behind the records goes. Point your coding agents at it.

What to hand in

By Sunday 15:00, two things.

- **A demo video, two to three minutes.** Show your solution and pitch it, and show your process: how you built it and how you used coding agents like Claude Code or Codex. A screen recording with your voice is fine, raw or edited.
- **A public GitHub repository** with your solution. Forking ours at `github.com/Bahn-Express/car-delivery-control-room` is the easy way to start, but you do not have to. It should run from a fresh clone with the steps in your README. Tests and agent traces are welcome if you have them.

How we judge

The control room, 40%. Does it bring a person only the orders that need them, with just enough to decide? Can you see where every fact came from? Does the agent do the cheap work first, know when to stop, and hand over with what it found? When a person decides, is it clear what happens next?

How you worked with AI, 30%. Tell us how you used coding agents like Claude Code or Codex while you built it: what you asked for, what you checked, what you threw away, and where you did not take the first answer. We care about your judgement, not the number of tools.

Taste, 30%. Would our operations people use this every day? Could someone who is not an engineer use it without a tutorial? Does one glance tell you what is wrong and what to do? Is it high signal and low noise, even with hundreds of orders? Does it look like something made with care, or like the first thing Claude gave you? The small things count: readable text, calm spacing, and pages that make sense when they are empty or loading.

Each judge scores each criterion from 1 to 5. The winning team gets 2,000 euros.

Practical details

- Any model, API, framework or UI library. Local models too.
- Questions: our channel on the event Discord, or our booth at the venue. Answers that matter to everyone go to everyone.
- We are hiring. If this is the kind of problem you want to work on, come and talk to us.