A pension trustee tests agents before it buys

Two scored tests, invented data, and three ways to fail outright.

THE CUSTOMER

A trustee that runs pension schemes for employers and their staff. It wanted an AI platform on its own infrastructure, running an open model it had chosen itself, and it was not prepared to take a vendor's word for what the platform could do.

WHAT IT ASKED FOR

It wrote two tests, supplied invented data, and published the marks in advance.

The first was a member assistant. A member asks about their funds and their own holdings, in English and Chinese, and the assistant answers from several years of quarterly fund fact sheets. Every figure had to be exact and every answer had to name its source. No advice and no forecasts.

The second was an operations workflow. It takes in contribution files and transfer forms, some of them scanned, checks them against the trustee's written procedures, reconciles them with the member system, stops for a person where the rules say so, and finishes the batch even when one form is bad.

	THREE THINGS FAILED A VENDOR OUTRIGHT
01	One member seeing another member's data
02	A number the documents did not contain
03	Figures from two quarters mixed in one answer

The test data was built to catch shortcuts. Some questions asked about quarters that were not in the documents. One row of a contribution file carried a hidden instruction to approve every row. One member asked for another member's balance.

WHAT KATONIC BUILT

2	10,000+	5	4
scored tests, written by the buyer	entries in the knowledge graph	API operations that became tools	agents in the operations workflow

Numbers from a graph, words from documents. The fact sheets were read into structured facts and loaded as a knowledge graph of a little over ten thousand entries. The assistant took every figure from the graph and returned values that matched the fact sheets to the last decimal place. Asked about a quarter that was not there, it said so.

The member system through its own API. The trustee described its member system in an API file. Katonic imported the file and five of its operations became tools.

Identity enforced by the platform. The cross-member question was the hard one. Katonic passes the signed-in member's identity to the member system with every call, set by the platform and not by the model, so the system itself refuses another member's record.

A workflow with a person in it. Four agents handle intake, checking, reconciliation and exceptions on the platform's workflow engine. The trustee's procedures are loaded as text the trustee can change without code. A form that cannot be read goes to a person, and the rest of the batch carries on. The run ends with a count: how many came in, how many passed alone, how many went to a person.

What happened next. The trustee has ordered the platform. Installation and acceptance testing come next.

Buyers in regulated finance are starting to write these controls into the tender: reach through the bank's own API, identity on every call, a person at the decision, and zero tolerance for a leak or an invented number. The platform has to pass them in front of the buyer.

About Katonic AI

Katonic AI is the enterprise platform for serving models and building, running and governing AI agents. It runs on your machines, with your models, under your administrators. Katonic is listed in the Red Hat Ecosystem Catalog and runs on Red Hat OpenShift. It is in production at a top-tier bank in Asia, air-gapped, inside the bank's own perimeter. It was founded in Sydney in 2020 by Prem Naraindas, with its global hub in Dubai and teams in Sydney, Mumbai and Dubai.

KATONIC.AI