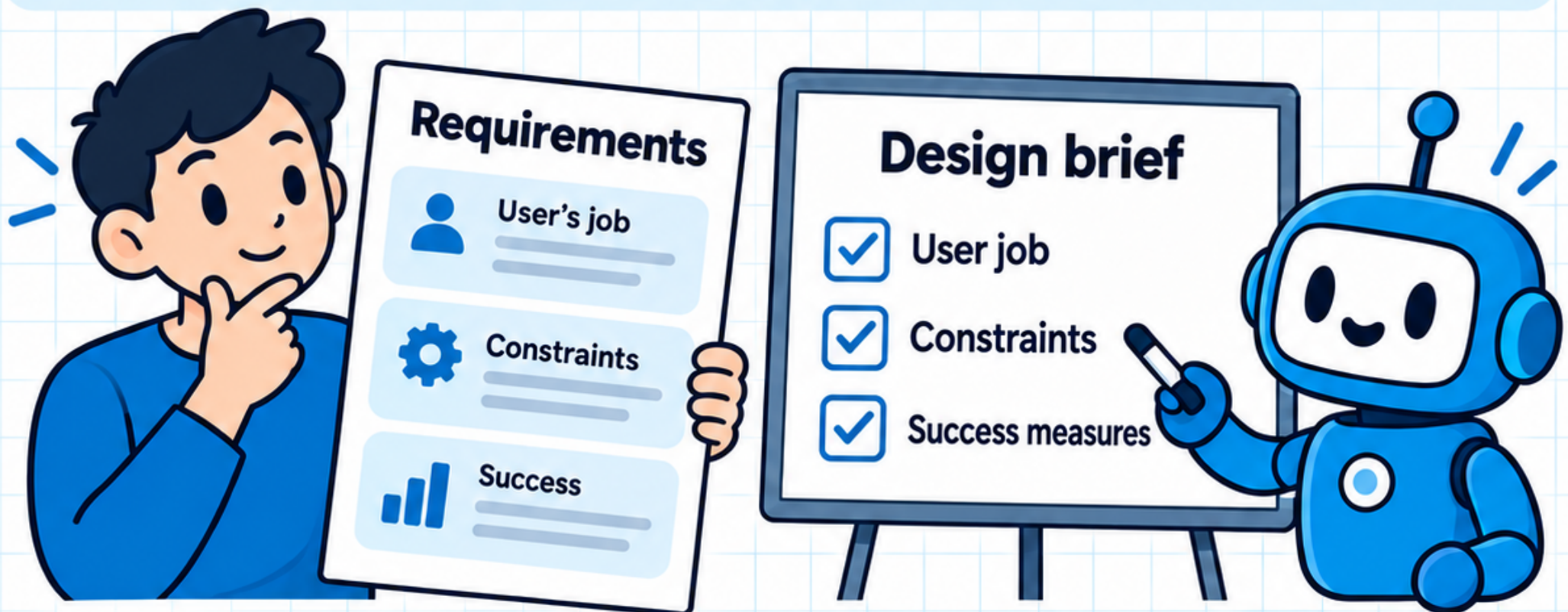


BEFORE YOU DRAW BOXES, DEFINE SUCCESS

Requirements and success metrics

A good design starts with the user's job, the constraints, and a way to measure the result.

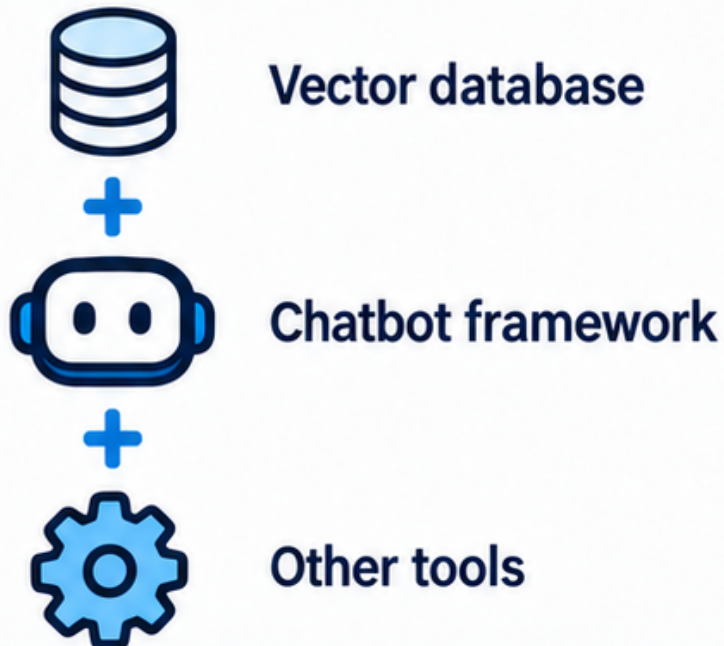


Today: write the brief for an AI policy assistant.

START WITH A JOB, NOT A TOOL

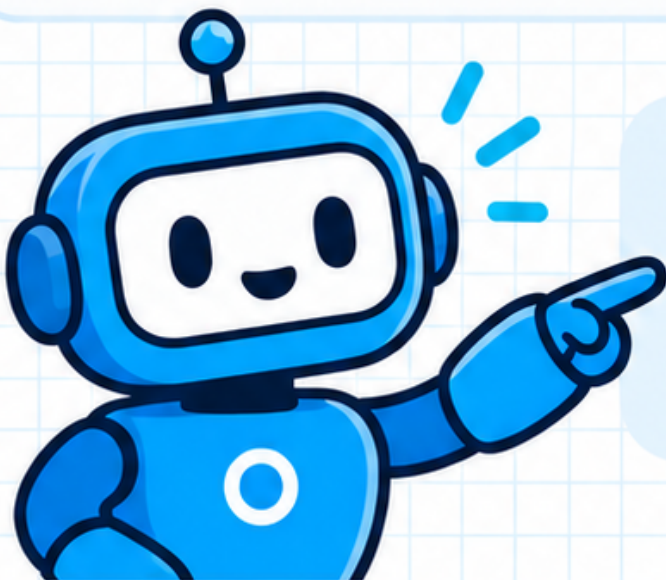
Weak brief:

“Build a chatbot with a vector database.”



Useful brief:

“Help employees find the correct travel policy and its source.”



The second brief tells us who needs help and what a useful result looks like.
The tool choice comes later.

FUNCTIONAL REQUIREMENTS DESCRIBE BEHAVIOR

What must the system do?

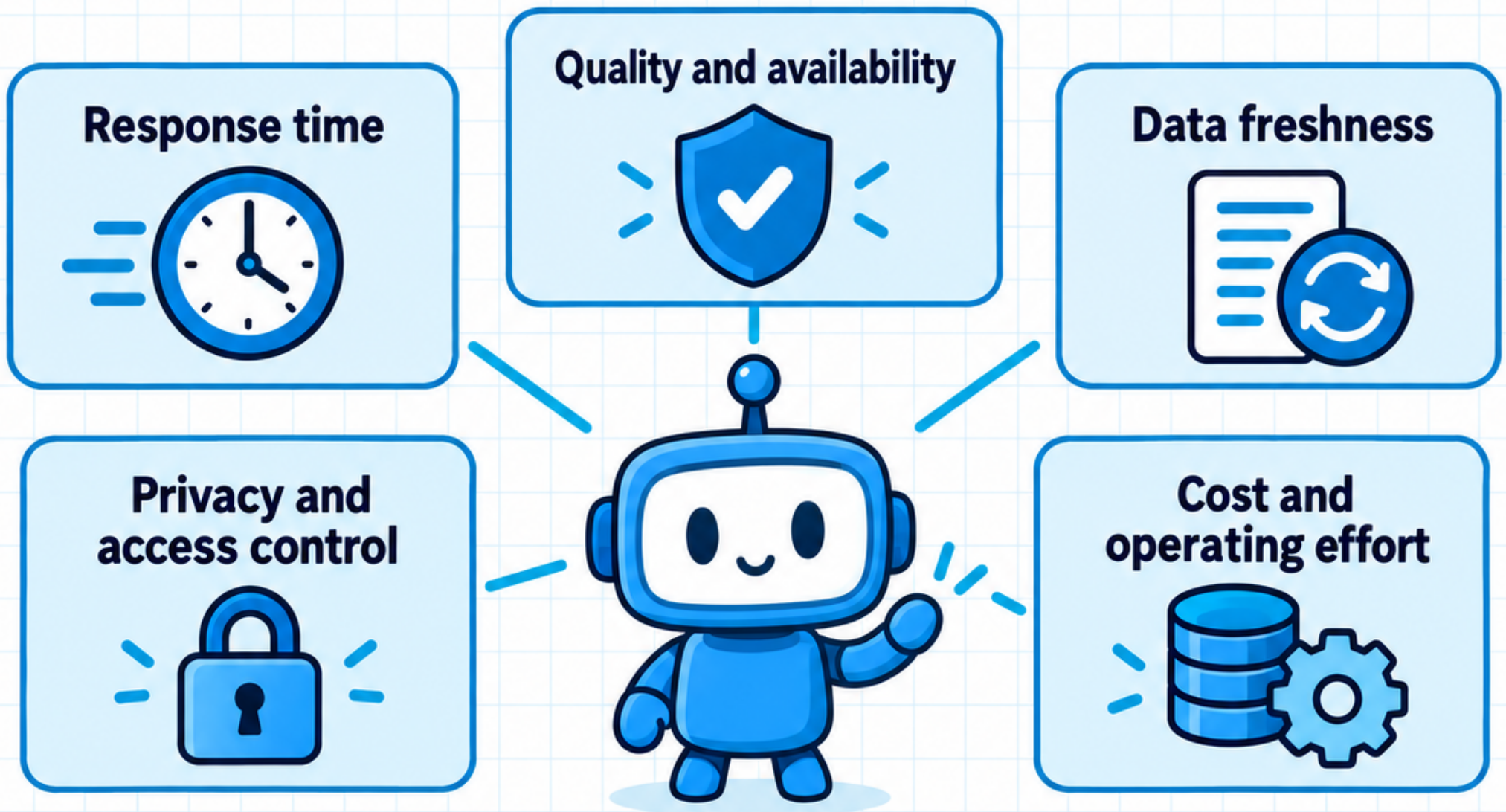


Each behavior needs an observable result.



NONFUNCTIONAL REQUIREMENTS SHAPE THE DESIGN

How well must the system work?

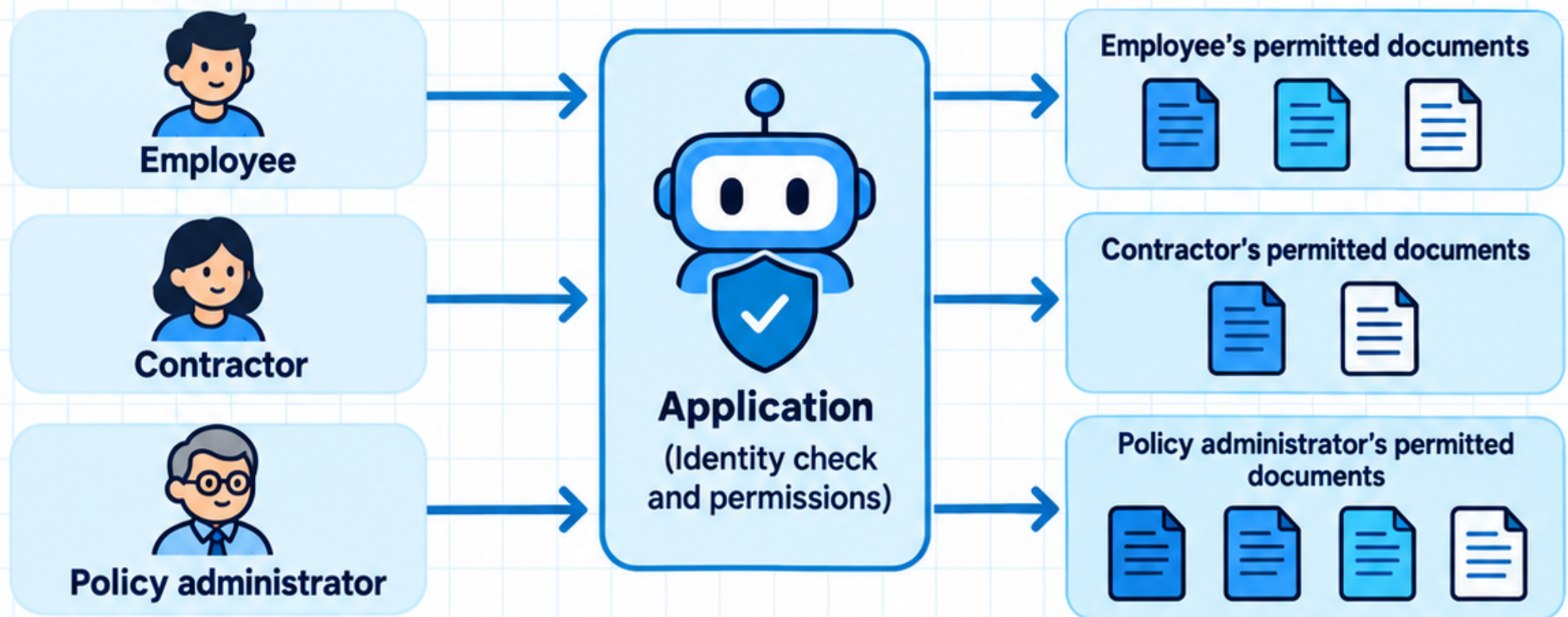


**“Fast, accurate, and cheap” is a wish.
A design brief defines how each property will be measured.**

NAME THE USERS AND THEIR BOUNDARIES

An employee, a contractor, and a policy administrator can have different permissions.

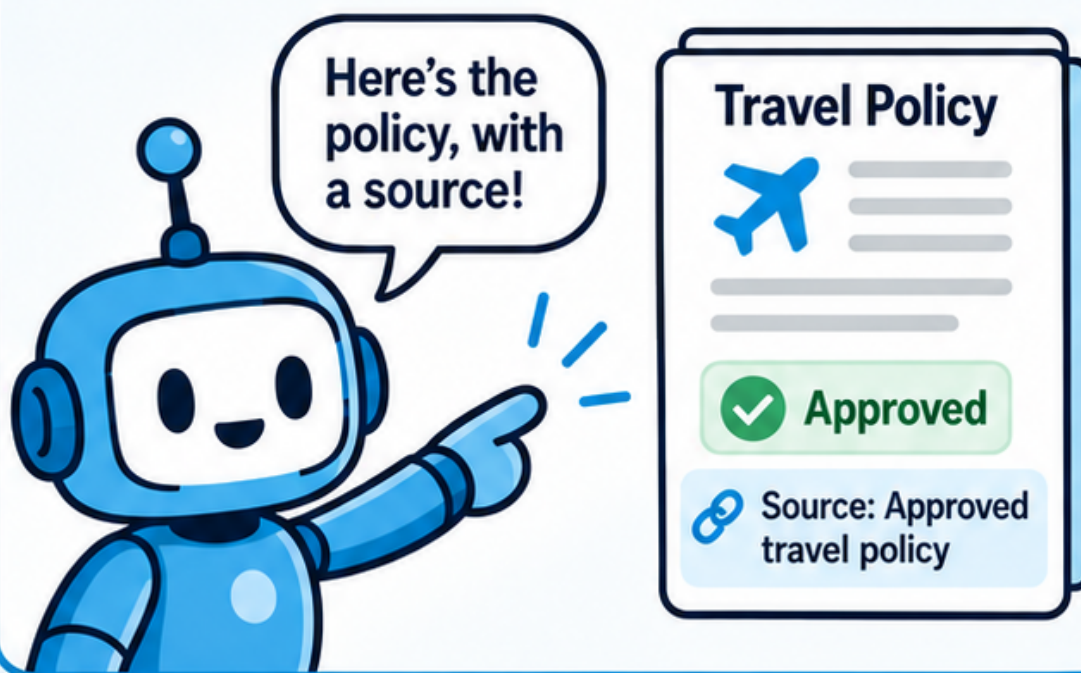
The same question does not imply access to the same documents.



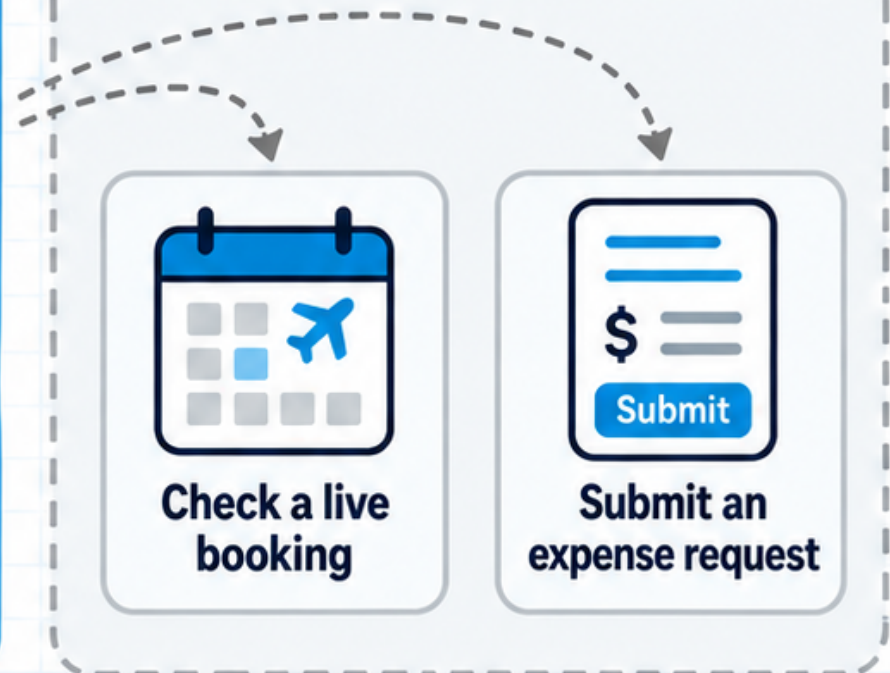
Resolve identity in the application. Carry the user's allowed scope into retrieval, tools, and stored results.

STATE THE SCOPE OF VERSION 1

V1: explain approved travel policies with sources.



Later: check a live booking or submit an expense request.



**Those additions need new data, permissions, and failure handling.
A clear first version lets us measure one useful job before adding more responsibilities.**



New data



Permissions



Failure handling

DESCRIBE TRAFFIC WITH UNITS

Illustrative peak:
120 requests
per minute.

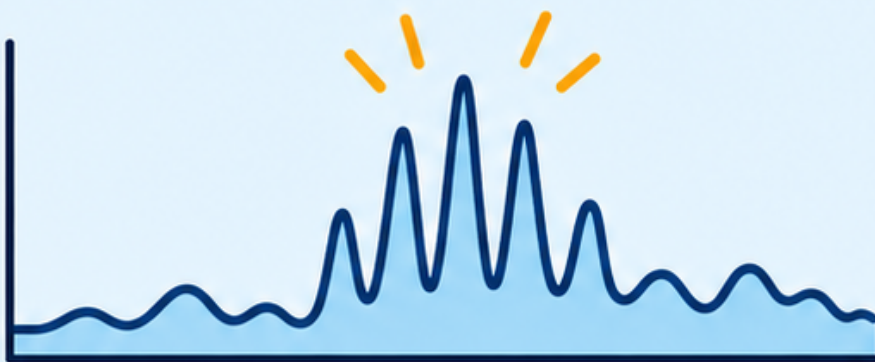


$120 \div 60 = 2$
requests per second.



Ask about bursts, simultaneous users, and request length too.
Two short questions and two large document analyses can
create very different workloads.

Teaching example



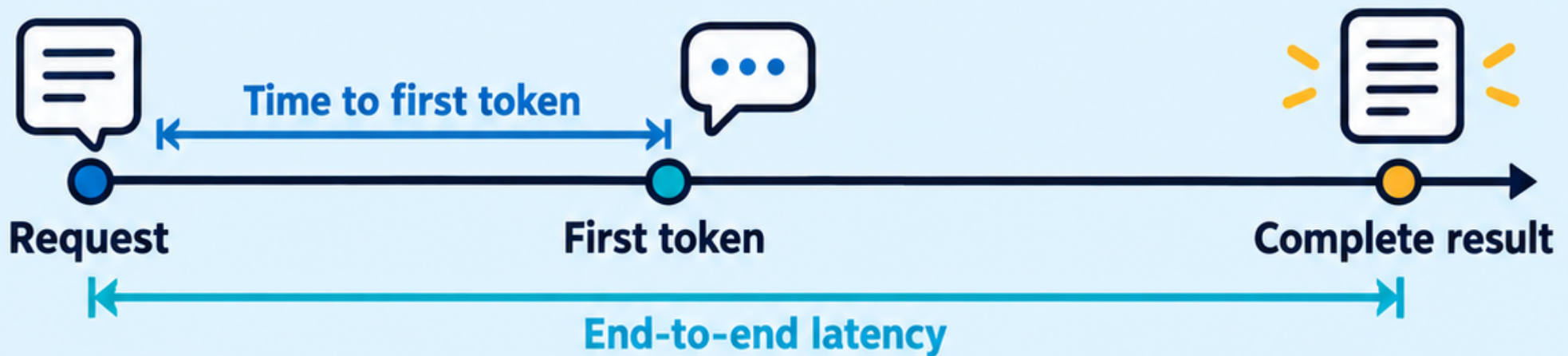
Two short
questions



Two large
document analyses



DEFINE WHICH LATENCY YOU MEAN



Time to first token:
when text starts
arriving.



End-to-end latency:
when the complete
result is ready.

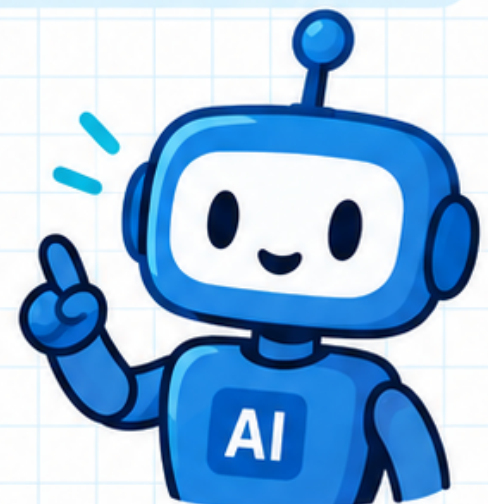


p95 latency:
95% of measured
requests finish
within this time.



Hypothetical target

Example target: p95 complete answers
under 5 seconds at the stated peak load.



MEASURE COST PER SUCCESSFUL TASK

Suppose a test run costs \$3 and completes 60 tasks successfully.

Illustrative calculation

\$3

60

$$\text{\$3} \div 60 = \text{\$0.05}$$

per successful task.



Include failed attempts and retries in the spend.
State whether the calculation includes model calls,
search, hosting, and human review.

Cost scope (to declare)



Model calls



Search



Hosting



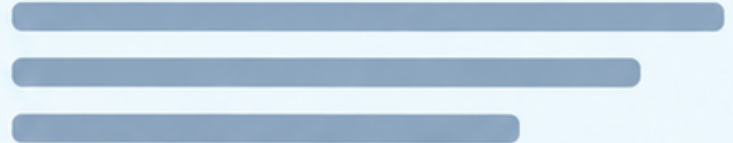
Human review

DEFINE QUALITY FOR THIS PARTICULAR JOB

For the policy assistant, a successful answer must:

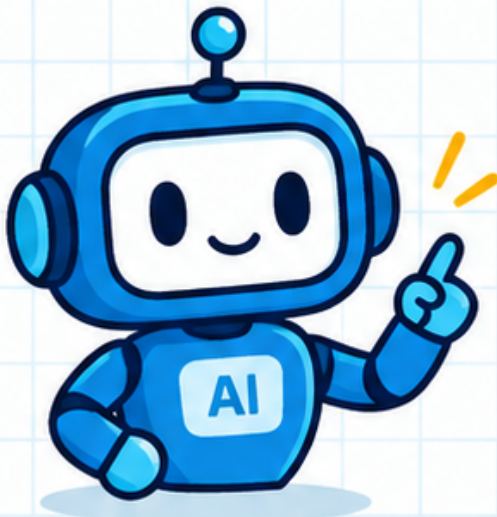
- ✓ Address the question.
- ✓ Match the applicable source.
- ✓ Cite the correct document.
- ✓ Avoid disallowed information.

Supported



Source: Policy Document

Unsupported

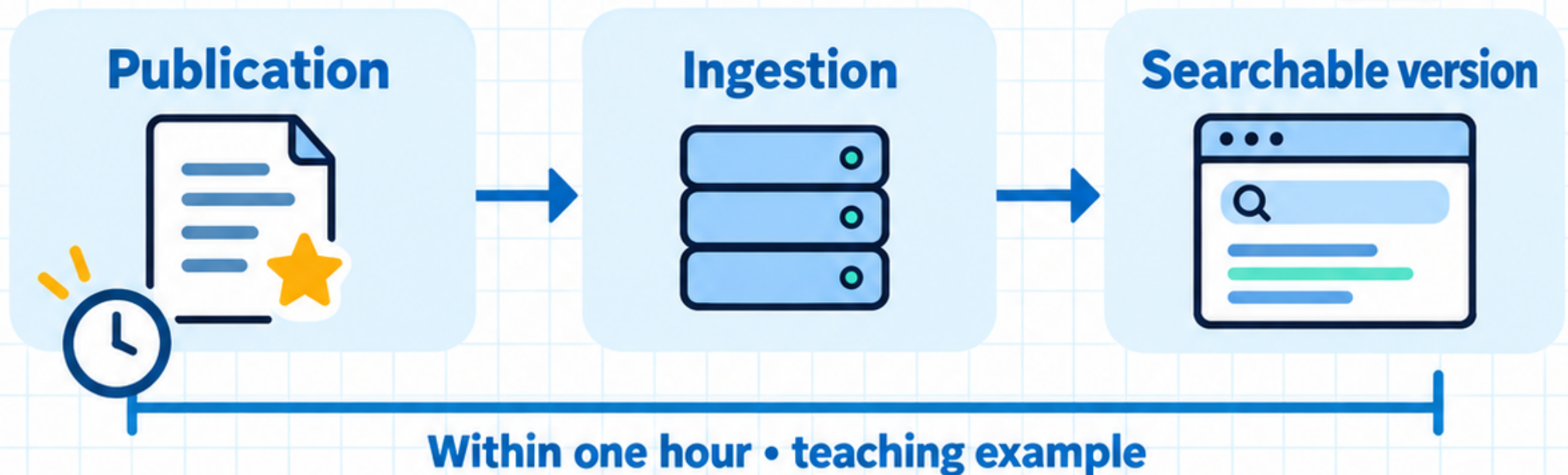
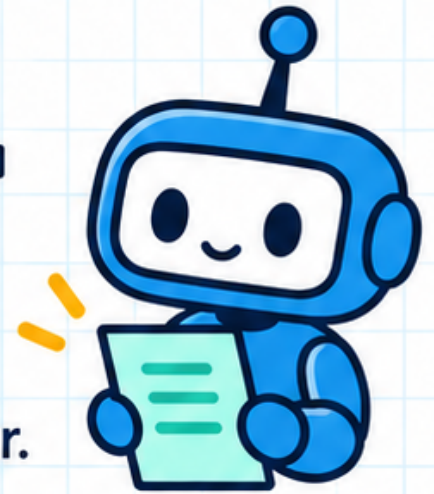


A fluent answer can fail this definition.
A correct handoff can pass a test that expects uncertainty.

GIVE FRESHNESS ITS OWN REQUIREMENT

“Use the latest policy” needs a time boundary.

Example: a published update becomes searchable within one hour.



Now the design needs change detection, index updates, and a way to stop old versions appearing. The freshness clock begins at a defined event.



Detect changes



Update index



Retire old versions

SPECIFY FAILURE BEHAVIOR



**No matching
source**



**explain the gap
or hand off.**



**Model
timeout**



**follow a limited
retry policy, then
show a fallback.**



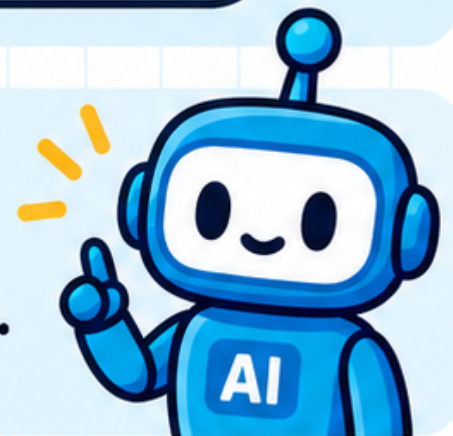
**Access
denied**



**return no
protected content.**



**These responses are requirements.
They belong in the tests and the architecture.**



DECIDE WHAT DATA CAN BE RETAINED

Questions, retrieved passages, and traces can contain private information.



Application records

 Expiry rule



Documents

 Expiry rule



Traces

 Expiry rule



Define what is stored, who can read it, when it expires, and how it is removed.



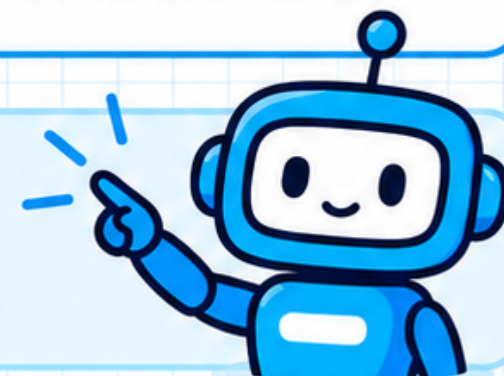
Useful debugging does not require keeping every payload forever. Retention rules must cover each store and telemetry destination.

TURN WISHES INTO AN ACCEPTANCE TABLE

Requirement		Evidence
	Correct policy	reviewed answer cases
	Private documents stay private	access-boundary tests
	Fast responses	latency under declared load
	Fresh policy	timed update test
	Affordable service	cost per successful task



Every important requirement needs a measurement method.



TOOLS HELP COLLECT THE EVIDENCE



PostgreSQL

store versioned test cases and outcomes.



Python or pytest

check expected behavior.



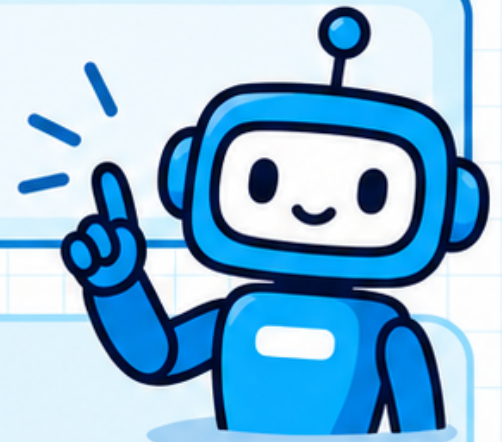
OpenTelemetry

record the timing of request steps.



Phoenix

inspect AI traces and compare evaluations.



Tools collect evidence.

The product team still defines what success means.

RESOLVE CONFLICTING REQUIREMENTS



More retrieved evidence can increase coverage and response time.



A smaller model can reduce cost and miss difficult questions.



Candidate A



Same task set



Quality



Latency

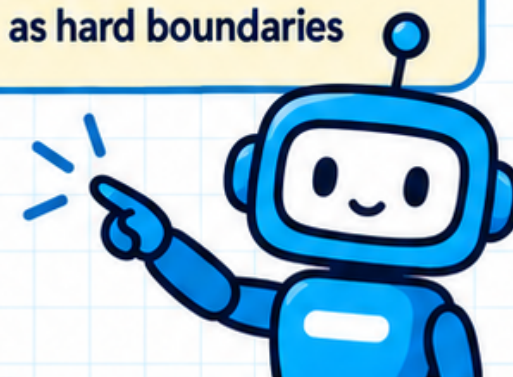


Cost



Permission requirements

as hard boundaries



Compare designs on the same task set. Keep permission requirements as hard boundaries. Decide acceptable quality, latency, and cost trade-offs from evidence.

WRITE A ONE-PAGE DESIGN BRIEF

Fill in:



User and job



Required behavior and V1 scope



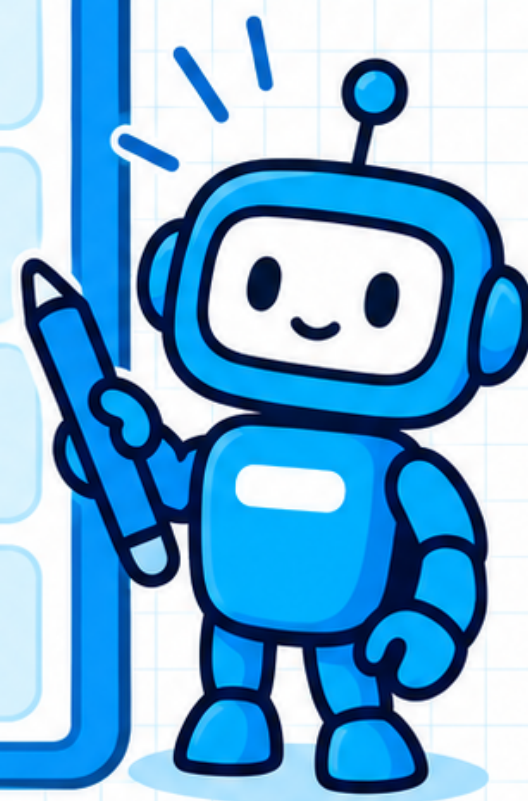
Data sources and permissions



Quality, latency, freshness,
and cost measures



Expected traffic and failure responses



**Label every unmeasured number as an assumption.
Give each requirement an owner.**

YOUR TURN

Design the brief for a support assistant or a nightly demand forecast.
Write three required behaviors, three measurable constraints,
and two failure responses.



3 required behaviors



3 measurable constraints



2 failure responses



For every metric, name its population or denominator and measurement method.



DAY 3: Choosing the right system shape



Save this checklist before choosing your stack.