zamp | CANDIDATE GUIDE

AI Solutions Associate — Case Study

Everything you need to know to do your best work · 2026

🎉 **Congratulations — you've moved to the next round**

This document has everything you need for the case study. Read it fully before you start. Good luck.

OVERVIEW

What this is — and why we do it this way

This is a take-home case study, not a whiteboard exercise. You have **one week** to pick a real operational problem, build a working automated process that actually runs, and submit it. If your build moves forward, you'll demo it live in the follow-up interview.

We work with companies on hard, messy operational problems — invoice processing, vendor onboarding, GTM execution. An AI Solutions Associate's job is to understand those problems quickly, design an automated process that handles real inputs end-to-end, and explain it clearly to the people who need to act on it. This case study replicates that exactly.

We care about whether your process actually works, the judgment behind your design choices, and how you explain it to a non-technical buyer.

> THE CORE QUESTION WE'RE ASKING
>
> Given a real operational problem, a week, and access to any AI tools you want — can you build a process that actually runs, handles real inputs, and deals with edge cases gracefully?

SCHEDULE

How the week runs

DAY 1

Pick your problem

Read all three problem statements and pick one. Reply to your hiring coordinator with your choice — subject: **"ASA Case Study — [Your Name] — PS-1"** (or PS-2, PS-3).

DAYS 2–6

Build

Build a working automated process that handles real inputs end-to-end. Wire up the logic, test it on your happy path and the edge cases you've defined. The process should actually run — not just look like it does.

DAY 7

Submit

Send your hiring coordinator two things: a link to your process (live and runnable), and a link to your 5-minute demo video.

AFTER SUBMISSION

Interview + live demo

If your submission moves forward, we'll schedule a call. You'll run your process live — a happy path and the edge cases you've prepared — and walk us through your decisions. No slides. The process runs in real time.

WHAT TO SUBMIT

Two things — a working process and a 5-minute demo video

Send both to your hiring coordinator on Day 7. That's the full submission.

⚙️

Deliverable 1 — A working automated process · live and runnable

Build a process that actually executes — not a mockup or a dashboard that simulates one. It should accept real inputs (a PDF, a form submission, a CSV row, a prospect name), run through your logic, and produce a real output. Use any stack you want: n8n, Make, Zapier, Cursor, Lovable, plain Python, anything. What matters is that it runs live and you can demonstrate it in the interview. The UI is part of the grade — we expect an intuitive, well-designed interface with a live run view (showing each stage as it executes) and a dashboard (showing history, status, and outputs across runs). This is what we'll look at together during the demo.

🎥

Deliverable 2 — A 5-minute demo video · Loom or any screen recording

Record yourself walking through your process. Show the happy path running live, then run at least one edge case. Narrate what's happening and why — what the process is doing at each step, the decisions it's making, and anything interesting about how you built it. Five minutes max. No editing needed, no slides. Just you, your screen, and your process running.

LIVE DEMO — IN THE FOLLOW-UP INTERVIEW

If your submission moves forward, you'll run your process live during the interview — a happy path plus the edge cases you've designed and prepared. We'll watch it execute in real time, then talk through your decisions. No slides. The process runs, we ask questions, you explain your thinking.

PROBLEM STATEMENTS

Pick one of three — read all before you choose

Each problem reflects work Zamp actually does. No right answer. The one that gives you an idea is the right one.

PS-1 · FINANCE / AP

Invoice processing — from PDF to decision

A mid-size company receives hundreds of vendor invoices every month — all by email, all as PDFs. Someone on the AP team opens each one, finds the matching purchase order in a spreadsheet, checks whether the numbers line up, and decides what to do with it. It's repetitive, it's slow, and it's the kind of work where a tired person on a Friday afternoon makes expensive mistakes.

The invoices themselves are messy. Different vendors format them differently. Some are clean PDFs with machine-readable text. Some are scanned images. Line items might be itemised or bundled. Tax might be embedded or separated. PO references might be explicit or implied. And sometimes critical information is just missing — no invoice number, no date, no clear total.

On the other side is a procurement system with its own logic — approved vendors, PO amounts, tolerance thresholds, duplicate detection rules. Matching an invoice to a PO isn't always obvious. A vendor might split a single PO into multiple invoices. Amounts might be close but not exact. The process has to make a decision — and be able to explain it.

> Build a process that takes an invoice as input and produces a clear, reasoned decision as output — with everything that happened in between visible. You decide what to extract, how to validate, what rules matter, and what the output looks like.

EDGE CASES — YOU DEFINE THEM

Design and build 2–4 edge cases of your own. They should be realistic scenarios where the process has to behave differently from the happy path — situations that reveal how flexible and deliberate your logic is. Don't pick trivial ones. The edge cases you choose tell us how well you understand the problem.

PS-2 · OPERATIONS

Vendor onboarding — from submission to approval

Before a company can pay a vendor, someone has to verify they're legitimate. That means collecting the right information — company details, banking info, tax registration, compliance documents — and checking that it's complete, consistent, and credible. It sounds straightforward. In practice, it's a mess.

Vendors submit incomplete forms. They attach the wrong documents. The name on the submission doesn't match the name on the bank account. The tax ID format is wrong for the country they claim to be in. Some submissions look fine on the surface but have subtle inconsistencies that only become visible when you cross-reference the fields. And when something's missing, someone has to track down the vendor, explain what's needed, and wait.

Procurement teams deal with this across dozens of new vendors every quarter, alongside everything else they're managing. The review is manual, the follow-up is manual, and the only audit trail is whatever's in someone's inbox. A bad onboarding — a vendor who slips through with inconsistent details — can cause payment fraud, compliance issues, or both.

> Build a process that takes a vendor submission as input and produces a clear status as output — approved, pending, or rejected — with the reasoning visible. For anything not approved, the process should communicate back what's needed. You decide what the submission looks like, what rules to apply, and how the output is structured.

EDGE CASES — YOU DEFINE THEM

Design and build 2–4 edge cases of your own. They should be realistic scenarios where the process has to behave differently from the happy path — situations that reveal how flexible and deliberate your logic is. Don't pick trivial ones. The edge cases you choose tell us how well you understand the problem.

PS-3 · GTM

Personalised outreach — from prospect to draft

B2B sales lives or dies on first contact. A message that sounds generic gets deleted. A message that references something real — a recent hire, a funding round, a problem the company is publicly grappling with — gets a reply. The gap between the two is research: finding the signal, understanding why it matters to this specific person, and translating it into something worth reading.

The problem is time. An SDR working a list of 200 prospects can't spend 20 minutes researching each one. So they don't. They swap in the company name, write something passable, and send it at scale. The reply rate reflects that.

But the signal is there. It's in LinkedIn activity, company news, job postings, funding announcements, executive interviews, recent press. For most prospects, there's enough to write something that sounds like it was written for them — if someone had the time to find it and the judgment to know which thread to pull. That's the job you're automating. A rep names a target. The process does the research, identifies the most relevant hook, and surfaces a draft that's actually worth reviewing — not just a template with the name filled in. The human stays in the loop before anything sends.

> Build a process that takes a prospect as input and produces a personalised outreach draft as output — grounded in something real about that person, ready for a human to review before it goes anywhere. You decide how to find signal, how to judge what's relevant, and how to turn it into a message worth sending.

EDGE CASES — YOU DEFINE THEM

Design and build 2–4 edge cases of your own. They should be realistic scenarios where the process has to behave differently from the happy path — situations that reveal how flexible and deliberate your logic is. Don't pick trivial ones. The edge cases you choose tell us how well you understand the problem.

GUIDANCE

How to approach the week

We've watched a lot of people do this kind of challenge. The ones who do well aren't the ones who build the most — they're the ones who make the best choices about what to build, get it actually working, and can explain it clearly. Here's what we'd suggest.

1 **Map the process on paper before you build anything**

Write out every step: what's the input, what happens at each stage, what are the decision points, what's the output. Do this before touching any tools. The clearer your process map, the easier the build. Candidates who skip this step usually end up with something that works on the happy path but falls apart on anything unexpected.

2 **Get the happy path working first, then add edge cases**

Don't try to handle everything at once. Build the clean path end-to-end — input comes in, process runs, correct output comes out. Once that works reliably, add your edge cases one at a time. A process that handles 3 scenarios well beats one that half-handles ten.

3 **Use AI tools — and be deliberate about it**

We use AI at every stage of our work. Use it to help you build faster, generate test inputs, write extraction logic, draft outputs. There's no extra credit for doing things manually. What matters is that your process works and you understand why each tool is there — and that you can speak to it fluently in your video and in the interview.

4 **Prepare your demo runs before the interview**

You'll run the process live during the interview — a happy path and the edge cases you've built. Have your test inputs ready and rehearsed. Know exactly what you're going to run and in what order. A demo that goes smoothly shows you know your build. One that breaks live, even on a minor thing, is hard to recover from.

5 **Submit something that runs, even if it's not perfect**

A process that executes end-to-end — even if it only handles the happy path — is always better than one that's 80% built and can't actually run. We can have a great conversation about limitations and what you'd build next. We can't have one about something that doesn't work at all.

WHAT A STRONG SUBMISSION LOOKS LIKE

A process that actually runs — takes a real input, executes the logic, produces a clear output — with an intuitive UI and dashboard you can point to. Edge cases that are handled deliberately, not ignored. A demo video that shows it running and explains the thinking behind it. And a live interview demo where the process runs and you can talk fluently about every decision you made.

FAQ

Questions we get asked a lot

If your question isn't here, email your hiring coordinator — they'll get back to you same day.

Can I ask questions about the problem statement?

Yes — but treat ambiguity as part of the exercise. Make an assumption, note it somewhere you can reference in the live pitch, and move on.

What tools can I use to build the process?

Anything — n8n, Make, Zapier, Cursor, Lovable, plain Python, Retool, custom code. Same for AI: ChatGPT, Claude, Gemini, any model or API. The only requirement: the process must actually run live during the interview. No Zamp or Pace — use tools that exist independently.

Does the process need to use real data?

No — create your own test inputs. For PS-1, make a few realistic invoice PDFs and a PO dataset. For PS-2, create vendor submission forms or JSON. For PS-3, use real public information about real companies (or make up plausible ones). The inputs need to be realistic enough that the process logic is meaningful.

Do I need to build a UI?

You need something to show run status and outputs during the demo — a simple dashboard, a structured log, even a clean console output is fine. You don't need a polished product UI. What matters is that the interviewer can see the process running and understand the outputs.

What if I need more time?

Email your hiring coordinator before the deadline — don't go silent. If something genuinely went wrong, say so. If you underestimated scope, submit what you have and be honest about what's left.

What if I have no finance or AP background?

That's fine — most of us didn't either. Think about the person behind the workflow and build for them. Good product instincts transfer.

What happens after the interview + debrief?

We'll follow up within a few days. The debrief is a conversation about your thinking, not a second test. No gotcha.