

The payment reconciliation checklist

Run it against your own system. 35 questions, written for the person who has to answer for the numbers rather than for the person who wrote the code.

Answer each one yes or no. A no is not a defect on its own. A no you did not know about is, because every one of these becomes somebody's afternoon eventually, and usually on the day the month closes.

It comes out of building payment ingestion and allocation against M-Pesa Daraja and manual entry, so the mobile-money questions are specific. The rest apply to any system where money arrives and has to be matched to something, bank feeds included.

Money arriving

☐ Can a payment smaller than the amount due be accepted and allocated?

Partial payment is the normal case, not the exception. A system that only accepts the exact amount either rejects real money or silently records the wrong figure, and both are worse than holding a balance.

☐ Does the outstanding balance update from the payment, rather than being typed in?

If somebody keys the remaining balance by hand after each payment, the balance is an opinion. It should be the arithmetic of what was charged minus what was allocated, computed every time it is read.

☐ Can a payment larger than the amount due be accepted without failing?

Overpayment happens constantly: rounding up, paying two months at once, adding the transaction cost. Refusing it loses the money. Accepting it and ignoring the excess loses the customer's trust the first time they ask where it went.

☐ Is a credit balance a real, visible thing that carries forward?

Excess has to land somewhere with a name. If it is folded into the next invoice as a discount, nobody can tell later whether the customer overpaid or was charged less.

☐ Can a payment be accepted from a phone number that is not the account holder's?

A tenant's brother pays. A finance officer pays for the whole office. The paying number matching the registered number is the exception in Kenya, not the rule, so matching on it alone loses a real payment every week.

☐ **Is the account reference validated at the point of payment, not after?**

A reference typed into a Paybill prompt is the only thing tying that money to an account. Validating it while the payer is still on the phone is the difference between a correction and an investigation.

☐ **Is there a defined behaviour for a missing or unrecognised reference?**

Not a crash and not a silent default to the nearest match. A named state, with somebody responsible for it, is the answer. Guessing the account from the amount is how one customer's money pays another customer's bill.

☐ **Does a mistyped reference that resembles a real account get flagged rather than matched?**

Fuzzy matching feels helpful until it allocates to the wrong account and nobody notices for two months, because from the inside a wrong allocation looks exactly like a right one.

The callback path

☐ **Is the payment callback idempotent?**

The gateway will send the same callback twice. Not might: will. If the second one creates a second payment, the customer is credited twice and the ledger is wrong in a way that only surfaces at the bank reconciliation.

☐ **Is idempotency enforced by the database, not only by application code?**

A unique constraint on the gateway's transaction reference is what survives two requests arriving in the same millisecond. A check-then-insert in application code does not.

☐ **Does the callback endpoint return success before doing the work?**

A gateway retries on a slow response. If allocation, invoicing and notification all run before the response, a busy minute becomes a retry storm and every retry is another chance to double-post.

☐ **Are failed callbacks stored rather than dropped?**

A callback that arrives while the database is down should be on disk somewhere, waiting. If the only record of it was the HTTP request, that money arrived and your system never heard about it.

☐ **Can a missed callback be recovered without the customer telling you?**

Reconciling against the gateway's own transaction list on a schedule is what turns a missing callback into a five-minute gap instead of a support ticket next month.

☐ **Is a payment left in limbo by a timeout resolvable automatically?**

An STK push that times out leaves a request with no answer. The money may have moved. Querying the transaction status rather than assuming failure is the difference between a refund conversation and no conversation at all.

☐ **Is the gateway's transaction reference stored, in full, on every payment?**

It is the only identifier that exists on both sides. Without it, a dispute is two spreadsheets and a phone call.

Money going back out

☐ **Can an allocation be reversed without deleting anything?**

A reversal is an event that happened, and it belongs in the record. Deleting the original row makes the correction invisible and makes last month's report change after it was sent.

☐ **Is a refund distinguishable from a reversal in the ledger?**

Money returned to a customer and money moved between accounts are different facts. Recording both as a negative payment makes it impossible to answer how much was actually refunded.

☐ **Are gateway-initiated reversals ingested, not just yours?**

A payer can reverse an M-Pesa transaction. If your system only knows about refunds it issued, a reversed payment stays credited and the balance is wrong until the bank statement disagrees.

☐ **Does a reversal restore the invoice it was allocated to?**

Reversing the money and leaving the invoice paid is the most common half-done correction, and the one that takes longest to find.

Money that never touched the gateway

☐ **Are cash payments recorded in the same ledger, not in a separate book?**

Two books never agree. If cash is in a notebook, a spreadsheet or somebody's phone, the system's total is a partial total and everybody knows it except the report.

☐ **Are bank transfers ingested or keyed?**

Keyed is fine, if it is keyed once, into the same place, against the same accounts. What breaks is bank money living in the accounting package and mobile money living in the operational system.

☐ **Does a manual entry record who made it and why?**

Manual entries are where corrections live, and corrections are what an auditor reads first. An amount with no author is an amount nobody will defend.

☐ **Is there a limit on who can create a manual payment?**

If everybody can, the ledger is a suggestion. This is the single control most often missing in a system that otherwise looks careful.

Suspense

☐ Does unmatched money have somewhere to sit that is not “nowhere”?

A suspense account is not an admission of failure. It is the honest place for money that has definitely arrived and cannot yet be assigned. Without one, the options are guess or drop, and both are worse.

☐ Is somebody named as responsible for clearing suspense?

Not a team. A person, and a cadence. Suspense with no owner grows quietly for a year and then has to be cleared by somebody who was not there for any of it.

☐ Can you see how old each item in suspense is?

Ageing is what turns a queue into a decision. A payment sitting unmatched for ninety days is a different problem from one that arrived this morning.

☐ Is writing an item off a recorded decision rather than a deletion?

Writing off is sometimes right. Writing off without a reason, an amount and a name attached is how the same money gets written off twice.

The audit trail

☐ Can you trace any allocation from the invoice back to the gateway reference?

In one screen, without a developer. If answering “why is this invoice marked paid” needs a database query, the answer will not be available on the day it is actually asked.

☐ Does the record say who changed an allocation, and when?

Every reallocation is somebody’s judgement call. The name is what makes it reviewable, and it is the first thing missing when a system was built quickly.

☐ Is history immutable, or can a row be edited in place?

If a payment’s amount can be edited after allocation, no report about last month is stable. Corrections should add rows, never rewrite them.

The close

☐ Is the cutoff for a month defined, and does everything obey it?

A payment received at 23:58 on the last day of the month and a callback processed at 00:04 the next morning belong to different months, depending on which timestamp you close on. Pick one, write it down, and apply it everywhere.

☐ Do you distinguish the date the money moved from the date you recorded it?

They differ, sometimes by days. Reporting on the recording date makes a slow week look like a bad one, and makes a caught-up backlog look like a spike.

☐ **Does the ledger tie back to the Paybill statement, line by line?**

Not the totals. The lines. Matching totals can hide two offsetting errors, and usually does.

☐ **Does it tie back to the bank statement as well?**

Mobile money settles to a bank account, and the settlement is where the transaction costs, the delays and the occasional missing batch become visible. A system that reconciles to the Paybill and never to the bank is reconciling to a copy of itself.

☐ **If every answer above were no, would anyone find out this month?**

This is the one question that matters. A reconciliation problem is not discovered by the system that has it. It is discovered by a customer, an auditor, or a number that will not balance at the worst possible time.

Where this came from

We build payment ingestion, allocation and reconciliation systems, and this list is what we run against an existing one before quoting the work. It is useful on its own, and you are welcome to use it without speaking to us.