

NORTHWIND STUDIO CLIENT CABINET

SCOPE	TOOK	REVIEWER	FINDINGS
Access • payments • the AI • data • infra	5 working days	Independent audit	12 ranked

LAUNCH: NOT YET

SIX THINGS FIRST: THREE CRITICAL, THREE HIGH.

01 • Critical

The database key is in the
browser

02 • Critical

Change the number, read
someone else's invoice

03 • Critical

A forged message marks an
invoice paid

04 • High

Double-tap Pay, charged twice

05 • High

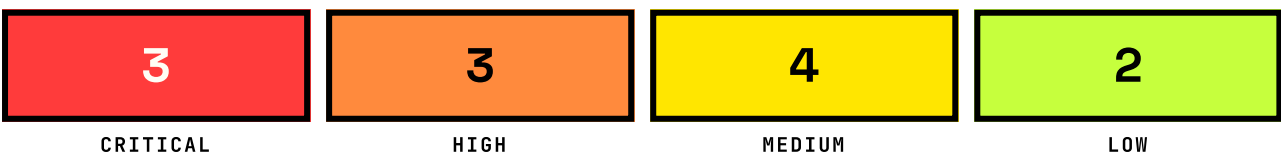
The cabinet bot reads every
client's files

06 • High

Money is stored as a decimal
that drifts

Fix these six — the three critical and three high — and the cabinet is safe to launch. The rest are schedule, not blockers.

SUMMARY



Automatic tools found 0 of these 12. The dependency scanner flagged 17 out-of-date-library advisories and the secret scanner found 0 stray keys — but none of

these findings, which are about how money, access and files move. Every one came from reading the code by hand. 8 of 12 are the same class of bug this studio has written up before.

FINDINGS

CRITICAL · 3

CRITICAL C1 The database key is in the browser

The master key to the whole database is sitting in the web page, where any visitor can copy it.

FOR NORTHWIND: With that key, anyone reads and changes every client's invoices and files directly, not just their own. The login in front of it stops nothing.

FIX: Keep the full-rights key on the server; give the browser a key scoped to the logged-in client, or no key at all.

REFERENCE: 4.1 SOMEONE ELSE'S FILES

CRITICAL C2 Change the number, read someone else's invoice

To see an invoice the cabinet only checks that you're logged in. Change one number in the address and you're looking at another client's invoice.

FOR NORTHWIND: Logged in as Brightlane, change the number and another client's invoice opens: amount, plan, payment status. Nothing was hacked.

FIX: Check the invoice belongs to the logged-in client, and let the database refuse the rest even if a page forgets.

REFERENCE: 4.1 SOMEONE ELSE'S FILES

CRITICAL C3 A forged message marks an invoice paid

When the payment company says an invoice is paid, the cabinet doesn't check the message is really from them. A faked message works just as well.

FOR NORTHWIND: A €5,800 invoice flips to paid with no money behind it. The studio ships the work, then finds the money never arrived.

FIX: Verify the provider's signature over the raw body before acting, and ignore anything that fails.

REFERENCE: 2.3 WEBHOOKS

HIGH**H1 Double-tap Pay, charged twice**

If you tap Pay twice, or your phone retries, you're charged twice for the same invoice.

FOR NORTHWIND: The €5,800 Studio retainer comes out of a client's account twice. They notice before the studio does.

FIX: Carry one idempotency key per payment all the way to a unique constraint in the database.

REFERENCE: 2.2 IDEMPOTENCY

HIGH**H2 The cabinet bot reads every client's files**

The help bot reads every client's files to answer anyone. It just hides where the answer came from.

FOR NORTHWIND: A client asks about their own project and the answer quotes Tidepool's confidential file. Tidepool signed a confidentiality clause.

FIX: Search only the files that client may open, so the bot never reads the rest.

REFERENCE: 4.2 THE BOT KNOWS TOO MUCH

HIGH**H3 Money is stored as a decimal that drifts**

Money is kept as the kind of number that can't hold exact cents. Split a bill in two and a cent goes missing.

FOR NORTHWIND: Split invoices and partial payments stop adding up. By month end the books and the bank disagree, and nobody can say why.

FIX: Store money as whole cents in integers; split with a method that assigns the leftover cent on purpose.

REFERENCE: 2.6 MONEY MATH

MEDIUM

M1 **Personal data in briefs goes to the AI**

Anything a client puts in a brief is sent to the AI as-is, even personal details they shouldn't have shared.

FOR NORTHWIND: Halden's brief has patient records in it. They leave the building for a third-party model with the rest of the text.

FIX: Swap personal data for placeholders before sending, and put it back only for the person reading the answer.

REFERENCE: 4.4 WHAT LEAVES FOR THE AI

MEDIUM

M2 **File links never expire**

A link to a file works forever, for anyone who has it, long after the project ends.

FOR NORTHWIND: The freelancer who rebuilt the cabinet still has links to a finished project's files. They open fine today.

FIX: Sign links, expire them in minutes, bind them to one person, and kill them when access is revoked.

REFERENCE: 4.6 THE LINK THAT NEVER DIES

MEDIUM

M3 **No limit on logins or bot questions**

Nothing stops someone trying thousands of passwords, or asking the bot so many questions it runs up a bill.

FOR NORTHWIND: One client account is one guessed password away from the invoices behind it, and the AI bill has no ceiling.

FIX: Count attempts per account and per address, slow them down after a few, and cap the bot's spend.

MEDIUM

M4 **Backups run, restore was never tested**

The cabinet is backed up every night, but nobody has ever checked that a backup can actually be put back.

FOR NORTHWIND: The one morning the database is lost, the studio finds out whether the backups work. A backup you can't restore is a file, not a backup.

FIX: Restore a backup into a scratch database on a schedule and check row counts; a backup is only real once it's been put back.

LOW

L1 Secrets printed to the logs

The cabinet writes its passwords and keys into its logs, where more people can see them than should.

FOR NORTHWIND: The secrets you'd rotate after a breach are already sitting in the log history, waiting to be read.

FIX: Keep secrets out of logs; redact known key shapes before anything is written.

LOW

L2 No staging, deploys from a laptop

There's only the live site. New versions go straight onto it from someone's laptop, with nowhere to test first.

FOR NORTHWIND: Every change is tried for the first time on real clients. A bad release is live until someone's laptop fixes it.

FIX: Add a staging copy, deploy from a pipeline, and keep the last good release to roll back to.

This is a sample. Yours would be about your app.

The cabinet above is real and deliberately broken; every finding reproduces live in the lab. A report on your own code reads the same way: ranked, with the fix and a way to reproduce each one.