

Sample: fictional app, not a real client.

CODE AND PRODUCT AUDIT REPORT

bookings.example.com

<https://bookings.example.com/>

Prepared for	Example Bookings (a fictional company: this whole report is a sample)
Scope	Repository example-bookings/app at commit 4b7e21c, and https://bookings.example.com/ , reviewed 12 to 21 October 2026 with read access
Date	22 October 2026
Version	1
Status	Reviewed by Aaron

Executive summary

Not yet ready for the 30-studio chain the business wants to sign, for reasons a scanner cannot see. One two-week sprint closes the security gaps (F-01 to F-03), and a second fits the data model to the enterprise plan (F-04).

Any signed-in customer can open any other customer's booking, with their phone number and visit notes, by changing a number in the address (F-01). I reported it to the founder on 13 October, the day I found it, and a stopgap check went live the next day. Staff at one studio can list another studio's customers (F-02), and the browser, not the server, decides how much deposit to charge (F-03). Beyond security, the data model has no idea of a company with several studios (F-04), so the enterprise plan cannot be sold the way the chain needs it.

What is in good shape: the two urgent findings from the free check were fixed within three days, card details never touch the app (F-13), and the database schema is clean, with constraints and migrations in version control (F-14). That makes F-04 a change of weeks, not a rebuild, and there is no case for one.

I recommend the two-week production-readiness sprint in the sprint quote below, covering F-01 to F-03 and F-05 to F-10: it closes every gap the chain's security review is likely to find. F-04 ranks fourth, ahead of lesser security findings, because it decides whether the enterprise plan can be sold at all. It follows as a second sprint at the same price.

Findings by severity

Critical	High	Medium	Low	Info	Total
1	3	5	3	2	14

Business goals

Taken from the interview with the founder on 12 October 2026:

- Sign the first multi-location customer, a fitness chain with 30 studios, on an enterprise plan by March 2027.
- Pass that chain's security questionnaire and data-protection review before it signs.
- Make deposits reliable, so that no-shows fall.
- Raise a seed round in the first half of 2027. Two investors have already asked how the app was reviewed.

The flows that matter most to these goals are booking with a deposit, the studio staff calendar, and refunds.

One constraint shaped the review: some studios are physiotherapy clinics, so the visit notes the app keeps can include health details.

Scope and method

The repository at commit 4b7e21c and the deployed app, reviewed from 12 to 21 October 2026 with read access, through a dedicated Anyfront account.

- An interview with the founder on 12 October 2026, and a walkthrough of the staff screens with one studio's manager.
- The public-surface scan (engine 1.2.0), run again on 12 October, with each finding confirmed by hand.
- A review of sign-in and sessions, access to data, deposits and refunds, the data model, error handling, logging, backups and deployment, following the audit method.
- Two test customers and two test studios on the staging copy of the app, to check who can see what.
- Dependency audits, and a secret scan of the repository's history.
- Database queries run by the founder from scripts supplied for the purpose: settings and counts only, never rows.

Not done: no penetration testing, no load testing, and no access to production data or production secrets.

From the free check to the audit

The free check on 6 October looked at what bookings.example.com shows anyone on the internet. Its two urgent findings, a live Stripe secret key in the JavaScript and public source maps, were fixed by 9 October, and this audit confirmed both fixes. Its five smaller findings are still open, and are carried into F-07, F-10 and F-11.

The audit looked behind the surface, at the code, the database, the hosting and the founder's plans, and found what a scanner cannot:

- Whether one signed-in user can reach another's data (F-01, F-02). A scanner sees the sign-in page; it cannot sign in as two customers and compare what each one sees.
- Whether the server trusts what the browser sends it (F-03).
- Whether the data model fits where the business is going, which takes the founder's plans as well as the code (F-04).
- Whether a bad day can be recovered from: backups, alerts and tests (F-05, F-06, F-08).

Findings ranked by business impact

In the order they matter to the business. Each is set out in full in the next section.

#	ID	Finding	Severity	Effort
1	F-01	Any signed-in customer can open any booking Security exposure	Critical	Days
2	F-02	Studio staff can list another studio's customers Security exposure	High	Days
3	F-03	The browser decides how much deposit to charge Revenue	High	Hours
4	F-04	The data model has no place for a company with several studios Fundraising and enterprise sales	High	Weeks
5	F-05	Backups have never been restored, and recovery goes back to last night Revenue	Medium	Hours
6	F-06	Failed deposits and server errors reach nobody Revenue	Medium	Hours
7	F-07	Customers' contact details and visit notes are written to the logs Security exposure	Medium	Hours
8	F-08	No automated tests on bookings or deposits, and deploys skip CI Cost to change	Medium	Days
9	F-09	Sessions last a year and survive a password change Security exposure	Medium	Hours
10	F-10	Three of the free check's header and error-page findings are still open Fundraising and enterprise sales	Low	Hours
11	F-11	The booking page loads the staff screens it never shows Revenue	Low	Days
12	F-12	Two dependencies have published security advisories Fundraising and enterprise sales	Low	Hours
13	F-13	Card details never touch the app Fundraising and enterprise sales	Info	Hours
14	F-14	The database schema is a sound base Cost to change	Info	Hours

Detailed findings

Critical F-01 Any signed-in customer can open any booking

Business impact · Security exposure

Anyone with a customer account can read other customers' names, phone numbers and visit notes by changing a number in the address, which would end the chain's security review.

Recommended fix

Load every booking through the signed-in account, answer 404 for anything else, and add a test for each of the 23 booking routes that a second customer gets nothing. A stopgap check on the single-booking route went live on 14 October.

Effort

Days

Evidence

- `server/routes/bookings.js:41: Booking.findById(req.params.id)`, with no check of who owns the booking
- `staging`: with two test customers, customer B opened customer A's booking: 200, the whole record returned
- `public.bookings`: ids are sequential, about 18,400 issued, so every booking is one guess away

High F-02 Studio staff can list another studio's customers

Business impact · Security exposure

A staff account at one studio can download every other studio's customer list, which no studio, and no chain, will accept from a booking system.

Recommended fix

Take the studio from the signed-in staff member's membership, never from the request, scope every staff query by it in one place, and add tests that staff at a second studio get nothing.

Effort

Days

Evidence

- `server/routes/staff/customers.js:18: studio_id` read from the query string
- `staging`: with two test studios, staff at the second listed the first studio's customers by changing `studio_id`

High**F-03 The browser decides how much deposit to charge**

Business impact · Revenue

A customer can pay a one-cent deposit for any booking, so the deposits meant to stop no-shows can be skipped at will.

Recommended fix

Work out the deposit on the server from the studio's own settings, create the payment for that amount only, and add a test that a changed amount is refused.

Effort

Hours

Evidence

- `server/routes/payments.js:27`: the payment is created with `amount: req.body.amount`
- `src/pages/Book.tsx:212`: the deposit is worked out in the browser and posted with the booking

High**F-04 The data model has no place for a company with several studios**

Business impact · Fundraising and enterprise sales

The 30-studio chain would need 30 separate accounts, with no shared staff, reporting or billing, which blocks the enterprise plan the business means to sell next.

Recommended fix

Add an organisation above studios, and memberships that give each person a role at one or more of its studios; move each existing studio into an organisation of its own, and scope queries by organisation. About two weeks of work, best done once the sprint's access tests exist.

Effort

Weeks

Evidence

- `db/migrations/0003_users.sql`: `users.studio_id`: each person belongs to exactly one studio
- `db/migrations/0007_billing.sql`: subscriptions are keyed by studio, so a chain would get 30 invoices
- `server/lib/auth.js:12`: the session carries one studio, with no way to switch

Medium F-05 Backups have never been restored, and recovery goes back to last night

Business impact · Revenue

A bad migration or a mistaken delete would lose up to a day of bookings and deposits, and nobody yet knows how long getting the rest back would take.

Recommended fix

Turn on point-in-time recovery for the database, restore last night's backup into a scratch database once a month, and write down the steps and how long they took.

Effort

Hours

Evidence

- database host settings: from the founder's screenshot: daily snapshots kept for 7 days; point-in-time recovery off
- docs/: no restore procedure, and no record of a restore

Medium F-06 Failed deposits and server errors reach nobody

Business impact · Revenue

When a deposit fails or the booking page breaks, studios hear about it from their customers, in their busiest hours, instead of from an alert.

Recommended fix

Add error tracking to the server and the app, and send alerts for failed payments, server errors and a stalled job queue to the founder's phone.

Effort

Hours

Evidence

- server/index.js: errors are written to the console only, and the host keeps logs for 3 days
- Stripe dashboard: from the founder's export: 41 failed deposit payments in September, none followed up

Medium F-07 Customers' contact details and visit notes are written to the logs

Business impact · Security exposure

Names, phone numbers and visit notes, some of them health details, are copied to logs that more people and services can read than the database, which the chain's data-protection review will ask about.

Recommended fix

Log booking IDs instead of request bodies, take the email out of the confirmation page's address (the free check's finding), and delete the existing logs once the change is live.

Effort

Hours

Evidence

- `server/middleware/log.js:9`: every request body is logged in full
- `/booking/confirmed`: the customer's email is still in the address, as the free check found on 6 October

Medium F-08 No automated tests on bookings or deposits, and deploys skip CI

Business impact · Cost to change

Every change to booking or payment is checked by hand, so changes are slow, and nothing would stop the fixes in this report from coming undone.

Recommended fix

Add a CI workflow that runs on every pull request and blocks the deploy when it fails, with tests for booking, deposits, refunds and the access rules from F-01 and F-02.

Effort

Days

Evidence

- `tests/`: 4 unit tests, all on date formatting
- `hosting settings`: every push to main deploys, with no checks required

Medium F-09 Sessions last a year and survive a password change

Business impact · Security exposure

A lost phone, or a former staff member's laptop, keeps full access to a studio's calendar and customers for up to a year.

Recommended fix

Shorten sessions to 30 days, renewed while in use; end every session on a password change and when a staff member is removed; and let studio owners sign a staff member out everywhere.

Effort

Hours

Evidence

- `server/lib/auth.js:31`: cookie `maxAge` of 365 days, and no way to end a session early

Low F-10 Three of the free check's header and error-page findings are still open

Business impact · Fundraising and enterprise sales

Each is small, but together they are among the first things the chain's questionnaire and an investor's reviewer will check.

Recommended fix

Add Strict-Transport-Security, turn off debug error pages in production, and remove the X-Powered-By header, as the free check report of 6 October sets out.

Effort

Hours

Evidence

- `https://bookings.example.com/`: no Strict-Transport-Security header, and X-Powered-By: Express, checked again on 12 October
- `server/index.js:58`: the error handler sends the stack trace whatever the environment

Low F-11 The booking page loads the staff screens it never shows

Business impact · Revenue

Customers on a phone wait several seconds for the booking page, which costs bookings from the studios' paid traffic.

Recommended fix

Load the staff screens and the calendar library only on the routes that use them, so the booking page ships only what it draws.

Effort

Days

Evidence

- `/`: 5 scripts, 1,992,431 bytes decoded, as in the free check
- `src/main.tsx:7`: the staff routes are imported up front, and make up most of the bundle

Low

F-12 Two dependencies have published security advisories

Business impact · Fundraising and enterprise sales

Neither is reachable from the code reviewed, but both will show up in the chain's dependency scan and need an answer.

Recommended fix

Upgrade both, and turn on automated dependency updates, so a new advisory arrives as a pull request.

Effort

Hours

Evidence

- `package-lock.json`: 2 high advisories, both in indirect dependencies, neither on a code path reviewed

Info

F-13 Card details never touch the app

Business impact · Fundraising and enterprise sales

Stripe's own checkout collects every card, which keeps the app out of most card-security requirements and the chain's payment questions short.

Recommended fix

Keep it that way: take deposits and refunds through Stripe's server-side API only, as the fix for F-03 does.

Effort

Hours

Evidence

- `src/pages/Book.tsx`: a redirect to Stripe's checkout; no card fields in the app

Info

F-14 The database schema is a sound base

Business impact · Cost to change

Constraints, foreign keys and migrations in version control make the change in F-04 a matter of weeks, not a rebuild.

Recommended fix

Keep every schema change in a reviewed migration, as now, and add the organisation tables from F-04 the same way.

Effort

Hours

Evidence

- `db/migrations/`: 14 migrations, applied in order on every environment

Remediation plan

Now (the founder, this week):

- Keep the stopgap check on the single-booking route in place, and search the access logs since July for bookings opened by an account that does not own them. If any were, take advice on whether to tell the customers concerned.
- Turn on point-in-time recovery for the database (F-05).

Effort F-05 · Hours

Sprint (two weeks):

- F-01, F-02 and F-03: data access and deposits, each with the tests that prove it.

Effort F-01 · Days F-02 · Days F-03 · Hours

- F-08: CI on every pull request, so those tests guard every later change.

Effort F-08 · Days

- F-05, F-06, F-07, F-09 and F-10: backups, alerts, logs, sessions, and the free check's open findings.

Effort F-05 · Hours F-06 · Hours F-07 · Hours F-09 · Hours F-10 · Hours

Next (a second sprint, then the retainer):

- F-04: organisations, memberships and roles, as a second two-week sprint at the same fixed price, scoped in the last days of the first.

Effort F-04 · Weeks

- F-11 and F-12 under the engineering retainer, alongside the chain's onboarding.

Effort F-11 · Days F-12 · Hours

Watch:

- Run the free check again after any release that changes the hosting or the headers, and before the chain's review.
- Restore a backup every month, and keep the record for the chain's questionnaire (F-05).

Effort F-05 · Hours

Not recommended: a rewrite, or a move to another platform. The stack fits the business. The gaps are in a few places in the code and in the data model, and each can be closed where it is.

Sprint quote

Example only: the business, the scope and the dates are invented. The price and the terms are the published ones.

- Scope, in rank order: F-01, F-02 and F-03 (bookings and customer lists that answer only their owner, and a deposit set on the server, each with tests that another customer or studio gets nothing and a changed amount is refused); F-08 (CI on every pull request, blocking a failing deploy); and F-05, F-06, F-07, F-09 and F-10 (a recorded restore, alerts, no personal data in the logs, 30-day sessions, and the free check's three open fixes).
- Price: US\$8,000, fixed. Prices exclude GST / consumption tax where applicable.
- Terms: two weeks from 4 November 2026, as pull requests that Example Bookings merges (arm's length access, the default). 50% at booking and 50% on delivery, net 7. The audit fee (US\$3,300) is credited if the sprint is booked by 22 November 2026, within 30 days of the readout.
- What does not fit: F-04, as a second two-week sprint at the same fixed price (US\$8,000), to start once this sprint's tests are in place; then an engineering retainer (US\$3,700 a month, up to 20 hours a month) for F-11, F-12 and the chain's onboarding.

Limitations

This report is a professional review of the scope described above, at the time it was done. It is not a security guarantee, a penetration test or a certification. It reflects the code at commit 4b7e21c and the app as it was on 21 October 2026; changes made since are not covered. It relies on the information and the query results provided by Example Bookings. Automated tools and manual review can both miss issues, so the absence of a finding does not mean the absence of a problem. No real payment was made: deposits and refunds were checked in the code and on staging.

Not assessed

- **Performance:** Load testing was out of scope for this audit.

A professional review of the stated scope at the stated time. Not a security guarantee, penetration test or certification.