INDEPENDENT TECHNICAL REVIEW

Meridian Bookings Ltd.

Booking platform codebase review

Prepared for the founders of Meridian Bookings Ltd. All names, findings, and details in this document are fictional. This sample shows the exact format of a real deliverable.

REVIEWED	STACK	ACCESS	CLOCK
Web app, API, infra	**Next.js · Node · Postgres**	**Repo + staging + Stripe sandbox**	**2 working days**

Prepared by Andrej Dragojevic · Invocation Technologies LLC · Stripe Certified Professional Billing Architect

The general read

You own more than you feared and less than you paid for. The product works: bookings flow, reminders send, and the calendar sync your agency demoed is real. The data model underneath it is sound, and that matters more than anything else in this report. Postgres schema design is the one thing that is expensive to redo, and yours does not need redoing.

The problems sit one layer up. The codebase shows two distinct authors: a careful one who built the booking engine and schema in the first three months, and a rushed one who bolted on payments, notifications, and the admin panel after. The second author copy-pasted instead of reusing, trusted the browser instead of the server, and stopped writing tests entirely. Commit history says this coincides with your March deadline. That deadline is now technical debt with your name on it.

Nothing here says rebuild. Two findings say stop-and-fix-now, both in the payment path, and both are days of work, not weeks. The rest is a sequencing problem: the fixes below are ordered so each one de-risks the next. Hand this table to your agency and ask for a line-item response before you approve the next invoice.

> **Verdict: continue.** Fix the two critical payment findings before any new feature work, then hold the agency to the sequence on page 2. A rebuild would throw away a schema and booking engine that are genuinely good.

What was verified, and what was not

Verified: full repository with commit history, staging environment with admin and customer roles, Stripe sandbox, deployment pipeline. Not verified: production data volumes and the SMS provider account (no access provided; findings that depend on scale are marked accordingly in the table).

This is a synthetic sample. It is not based on any real client, engagement, or codebase. Real reports are confidential and are never used as samples.

FINDINGS, FEATURE BY FEATURE

The findings table

One row per area reviewed. Severity is impact if left alone. Classification is the verdict: **salvage** (keep, minor cleanup), **fix** (correct in place), **rebuild** (replace the piece), **defer** (real, but not now).

AREA REVIEWED	FINDING	SEVERITY	CLASSIFICATION	CHALLENGE FOR YOUR EXISTING TEAM	RECOMMENDED NEXT ACTION
Checkout & order flow	Order totals are recalculated in the browser and trusted by the server. A modified request can check out at any price.	CRITICAL	FIX	Ask why price validation was left client-side, and what else follows that pattern.	Recompute totals server-side from the database price before payment capture. One focused day of work.
Stripe webhooks	Webhook handler has no idempotency guard. Stripe retries create duplicate bookings and double-send confirmation emails.	CRITICAL	FIX	Ask how they tested payment failure and retry paths before launch.	Upsert on Stripe event ID. Two days including backfill of the duplicate rows already in staging.
Booking engine	Availability logic is correct, well-factored, and covered by the only meaningful test suite in the repo.	LOW	SALVAGE	None. Credit where due: this is the part worth keeping.	Protect it. Require tests for any change touching this module.
Admin panel	Role checks exist only in the UI. Any logged-in customer can call admin API endpoints directly and export all customer data.	HIGH	FIX	Ask for the list of every endpoint with server-side authorization, in writing.	Add middleware-level role enforcement on all /admin routes. Three days including an endpoint audit.
Notifications	Email and SMS sending is copy-pasted across seven call sites with no queue. A provider outage blocks the booking request itself.	MEDIUM	REBUILD	Ask why sending happens inline in request handlers instead of a background job.	Replace the seven call sites with one queued worker. A week, and it removes the timeout spikes in your staging logs.
Test coverage	Tests stop entirely after March. New payment and admin code has zero coverage; CI runs but asserts nothing new.	MEDIUM	FIX	Ask what changed in March, and what their definition of done includes today.	Make tests a contractual deliverable for the payment and admin fixes above, not a separate line item.
Deployment & infra	Single Heroku dyno, no staging-to-production promotion, secrets committed in an old migration script (rotation status unverified).	HIGH	FIX	Ask when secrets were last rotated and who currently holds production access.	Rotate all secrets now, purge them from git history, add a promotion pipeline. Two days.
Dependency health	Framework and ORM are one major version behind with clean upgrade paths. No abandoned packages in the critical path.	LOW	DEFER	None worth spending a meeting on.	Schedule the upgrades after the fixes above land. Half a day each.

Prioritized next steps

1. Fix server-side price validation and webhook idempotency before any other work is approved.
2. Rotate secrets and close the admin authorization gap in the same sprint.
3. Move notifications to a queued worker; make tests contractual from here on.
4. Then, and only then, resume feature work.