

How I built a hotel PMS

Reservations to night audit, for one or many hotels. Built to be audited, not just demoed.

Guest → Reservation → Check-in → Folio → Night audit

Next.js

TypeScript

PostgreSQL

Prisma

Zod

Syed Asjad Abbas

AI/ML Engineer · Founder & Lead Engineer, Zeploy Tech

Swipe →

WHY IT HAS TO BE RIGHT

Hotel bugs cost **real money**

Guest billed twice refunds and complaints

Room sold twice overbooked night

Wrong property's data privacy breach

12 staff roles, one or more hotels, and every night an audit that closes the books.

WRITTEN DOWN ON DAY ONE

Rules before **the first line**

- 1 Thin routes, rules in services
- 2 Property scope from the server
- 3 Exact money, never floats
- 4 Ledgers are append-only
- 5 Slow work goes to a worker

Enforced by lint and tests, so they held across 127 database models.

Two paths, one database

REQUEST PATH

Staff browser

Next.js + RTK Query



Request boundary

session, CSRF, rate limits



Route handler

authorize, validate, scope



Domain services

rules → PostgreSQL

BACKGROUND PATH

Night audit queued

job row in PostgreSQL



Worker claims it

SKIP LOCKED, lease



Date rolls over

charges posted once



Boards refresh

database notifications

A hidden button is never the only protection.

Syed Asjad Abbas

AI/ML Engineer

Swipe →

MONEY

Retries can't double-charge

Payments and other financial actions carry an idempotency key. If the response is lost and the client retries, the server returns the first result.

```
# the same payment, sent twice
POST payment  Idempotency-Key: <uuid>
POST payment  Idempotency-Key: <uuid>
# → one payment recorded (acceptance test)
```

NIGHT AUDIT

The books close **in a worker**

Night audit never depends on a browser staying open. A worker claims the job, runs it under a lease, and retries up to three times.

Claim	FOR UPDATE SKIP LOCKED
-------	------------------------

While the date closes	postings held
-----------------------	---------------

Duplicate night charges	0 in testing
-------------------------	--------------

OFFLINE

Offline is **read-only**, on purpose

When the connection drops, the desk can still see arrivals and in-house guests. Nothing can be written.

See arrivals, in-house, departures	yes
Book, check in, take payment	no

That avoids the hardest problem in hotel software: merging conflicting offline bookings.

PERFORMANCE

Measure before you optimize

BOTTLENECK	BEFORE	AFTER
Global search	~~8.3 s~~	0.45 s
Guest statistics	~~168 ms~~	0.16 ms
Room board buffers	~~38,346~~	4,967
List page (ORM count)	<del>~1s</del>	fixed

One innocent-looking ORM count was scanning every reservation room on each request.

CONTENTION

One slow endpoint slows everything

Slow list queries held the connection pool, so simple auth checks went from **8 ms** to **550 ms**.

Before the fixes	saturated at 10–25 req/s
------------------	--------------------------

After four fixes	50 req/s, 0 errors
------------------	---------------------------

P95 at 50 req/s	192 ms
-----------------	---------------

Measured on one instance on a 4-core laptop. Load scripts are in the repo.

Syed Asjad Abbas

AI/ML Engineer

Swipe →

THE HONEST ONE

My own audit found **a release blocker**

A routine rate-plan change could silently re-bill nights already charged. Nothing crashed. Guests would just pay twice.

P0 Silent re-billing, fixed with tests

P1 No request size limit, now 1 MiB

P1 Offline page leaked guest lists

The PMS in 8 lines

- 01 Write the rules before the code
- 02 Authorize every request on the server
- 03 Store money exactly, never floats
- 04 Never edit a ledger, add a row
- 05 Idempotency keys on money routes
- 06 Close the books in a worker
- 07 Measure before you optimize
- 08 Audit your own work before release

Read the **code and docs**

Architecture, audit, load tests and test suite are all in the repository.

github.com/syedasjadabbas/serene-management

Follow for more on AI/ML and backend engineering.

Syed Asjad Abbas

AI/ML Engineer · Founder & Lead Engineer, Zeploy Tech