The Wrong Chosen

One shared world, no server left running. Between keystrokes, nothing is awake.

A browser MUD running on Cloudflare Workers: one Durable Object per zone, hibernating WebSockets, and a world written as TypeScript modules rather than rows in a CMS. The interesting engineering is not the game loop. It is the set of guards standing in front of a deploy that cannot be taken back.

TOPOLOGY

One Durable Object per zone. WebSocket hibernation.

CONTENT MODEL

10 zones, 135 rooms, 108 NPCs, all TypeScript. No CMS.

BUILD GATE

Eight validators plus two typecheckers, in front of every deploy.

PERSISTENCE

Neon Postgres, four tables, mutable state only.

Solo build: architecture, game systems, content, and delivery · Live · August 2026

PROBLEM

A MUD wants a process that never stops. Workers do not have one.

The classic shape is an always-on server with a world tick, which is the worst possible fit for a platform that bills for the time your code is awake. The usual escape is a long-lived box somewhere, at which point most of the reason to be on the edge has gone. We wanted the shared persistent world without the heartbeat, and we wanted a returning player to reconnect into a room that still had other people standing in it.

DECISION

Turn-based resolution, so the server is allowed to sleep.

Command, wake, resolve, broadcast, sleep. Nothing runs in between. Combat still reads as live because the server resolves a whole exchange atomically and stamps effect metadata per line, then the client staggers those lines about 550 ms apart. The one timer in the system is a 45-second respawn alarm, and it rehydrates before it acts.

01

One object per zone, not per room

Walking around inside a zone is an in-memory function call. Only a zone transition crosses an object boundary: the socket closes with 4001, the client reconnects, and the Worker routes it to the destination zone. Room granularity would have made every step a network hop.

02

Hibernation is the cost model, not an optimisation

Sockets are accepted through the hibernation API, so an idle connection costs close to nothing while the object is evicted. Per-socket identity survives in the socket attachment. Character bags outgrew the 2 KB attachment cap, so they live in a lazily rehydrated cache that is force-persisted on every mutation.

03

The database holds mutable state only

Four Postgres tables on Neon, written behind meaningful events like a kill, a move, or a death, never per keystroke. No D1, no KV, no content tables. Sessions are stateless signed tokens, so the hot path verifies a signature instead of doing a lookup.

PRODUCT SURFACE

Screens from the shipped work.

One screen: log stream on the left as append-only DOM, Svelte panels on the right.

The server resolves the exchange atomically. The client staggers the lines so it feels live.

The pattern library. Every border in the game is drawn from this set.

SYSTEM

Content is code, so content gets a code review.

Zones, rooms, residents, monsters, quests, and items are TypeScript modules in the repo. A joke arrives as a diff. That only works if something checks the world before it ships, so eight validators run ahead of the bundler and the build stops on any of them. Auth gets the same treatment from two directions: a sliding-window limiter inside the Worker, keyed per bucket, and a coarser Cloudflare rule at the edge as a backstop.

Author

`Rooms, NPCs, quests, and items as typed modules. Weapon dice are parsed out of the joke text rather than duplicated into a stat table that would drift.`

Validate

`World graph, gatekeeper reachability, art fonts, host interface, audio, sprites, model budget, and asset inventory. Every allow-list also fails when an entry goes stale.`

Typecheck

`svelte-check across 723 files and a separate strict tsc pass for the Worker. Zero errors, zero warnings.`

Verify

`After the deploy, a script asks production whether a bogus hashed asset returns a 404 and whether the Draco decoder is still served without an immutable header.`

OUTCOME

A deploy that cannot be taken back

Assets are cached immutable for a year. If a request for a missing hashed chunk is answered with the HTML shell, that browser caches a document under a script URL for a year and executes it as JavaScript on every visit. A redeploy does not fix it and a purge does not fix it, because the bad copy is in the browser, not at the edge. This has already cost a sibling project a year of poisoned caches, so the 404 path is now asserted against production on every deploy.

A model pool sized for the panel it renders in

A hundred generated GLBs arrived at 391 MB, mean 3.9 MB, none under 2 MB, for a viewer 128 CSS pixels wide. Two passes fixed it: textures resampled to 512 px in their original container, then geometry re-encoded with Draco, ending at 121 files and 25.4 MB. The decoder is self-hosted rather than pulled from Google, because a sibling repo had already been bitten by that dependency.

A layout that stops moving

Two font preloads and one card were most of the shift on the play route. Removing them took Lighthouse mobile CLS from 0.173 to 0.0000, with first contentful paint down from 2,407 ms to 2,259 ms. SEO and best practices both score 100 across the five pages measured. The typing path is free: 56 synthetic keystrokes produced no event over 16 ms.

`Svelte 5` `Vite 8` `TypeScript` `Cloudflare Workers` `Durable Objects` `Neon` `Drizzle ORM` `Sentry`

What the eight validators are actually for.

None of them catch a bug in the game. They catch the class of mistake that ships quietly and stays shipped: a room with no way in, an item that nothing in the world can drop, a 4 MB model behind a 128 pixel window, a hashed chunk that answers with HTML. The world graph is walked from the starting zone on every build. If a zone has no door, the build stops.

threesided.com/projects/the-wrong-chosen

wrongchosen.com

hello@threesided.com