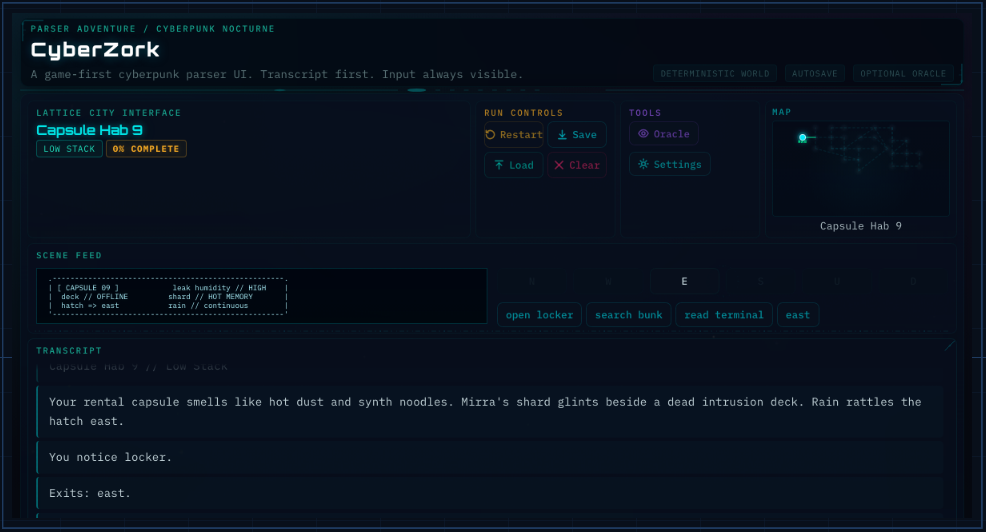
CyberZork

TRAINING MANUAL

Player workflow, parser quick reference, UI tour, setup checklist, and admin operations



A game-first cyberpunk parser UI. Transcript first. Input always visible.

```
.-----------------------------------------------------.
| [ CAPSULE 09 ]              leak humidity // HIGH    |
|  deck // OFFLINE           shard // HOT MEMORY       |
|  hatch => east             rain // continuous        |
'-----------------------------------------------------'
```

Capsule Hab 9 // Low Stack

Your rental capsule smells like hot dust and synth noodles. Mirra's shard glints beside a dead intrusion deck. Rain rattles the hatch east.

You notice locker.

Exits: east.

Prepared from the repository documentation set // March 22, 2026

DOCUMENT CLASS: FIELD REFERENCE // REV 2026.03.22

FIELD MANUAL // DISTRIBUTION: UNLIMITED // NEURAL UPLINK COMPATIBLE
BUILD REV 2026.03 // LATTICE CITY DOCUMENTATION BUREAU

CyberZork Training Manual

Revision 2026.03.22

Typeset in Orbitron (display), IBM Plex Sans (body), and IBM Plex Mono (code listings).
Generated with ReportLab from Markdown source.

Build date: March 22, 2026
Lattice City Documentation Bureau

How To Use This Manual

This manual combines the current player and admin documentation into one professionally formatted training reference. Use the first half for gameplay onboarding and the second half for setup, deployment, and design-agent operations.

Document Sections

- **01** Player Guide — gameplay, parser commands, UI surfaces, saves, Oracle, mobile flow
- **02** Parser Cheat Sheet — compact commands for quick reference
- **03** Setup Checklist — local, AI, Supabase, Netlify, and docs maintenance
- **04** Admin Guide — API surface, verification, deployment, and troubleshooting

Table of Contents

PLAYER GUIDE

Gameplay, parser flow, UI surfaces, saves, Oracle, and player troubleshooting

`CyberZork` is a deterministic cyberpunk parser adventure presented through a modern browser-native command console. This guide is for people playing the game: how to start, how the parser works, what the UI does, and how to make progress without reading the code.

Quick Start Card

- Start the app:

```
npm install
npm run play
```

- Open http://localhost:3000
- Enter these commands:

```
look
search bunk
east
talk vendor
take ghost battery
inventory
help
```

- If you get stuck, use `map`, `journal`, or `hint`

Quick Start

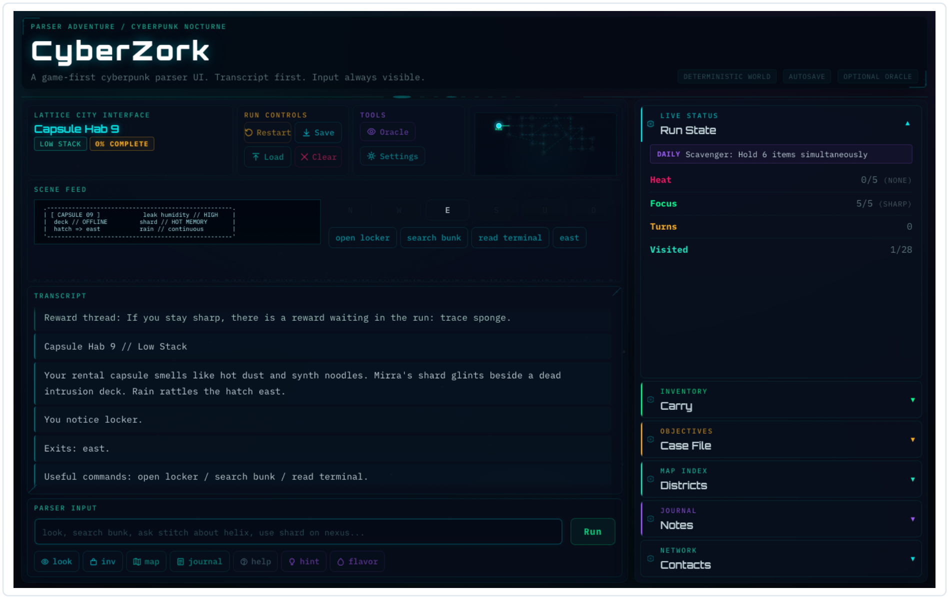
- Run the game locally
- Open the browser build
- Explore with short commands
- Search aggressively
- Use the map and journal to stay oriented

The Game At A Glance

- Genre: parser adventure with a graphical command console and district map
- World: deterministic authored city with fixed rooms, items, gates, objectives, and endings
- Core verbs: `look`, `search`, `take`, `talk`, `use`, `hack`, `install`, and compass directions
- Optional AI: boot storyline wrapper, Oracle hints/flavor, and NPC dialogue responses
- Replay systems: save slots, achievements, personal bests, and daily challenges

The command style follows the standard parser IF pattern documented by IFWiki's overview of parser-based interactive fiction. Source: IFWiki: Parser-based interactive fiction.

Start Screen And Main Layout



A game-first cyberpunk parser UI. Transcript first. Input always visible.

```
.-------------------------------------------------.
| [ CAPSULE 09 ]             leak humidity // HIGH |
|  deck // OFFLINE          shard // HOT MEMORY    |
|  hatch => east            rain // continuous     |
'-------------------------------------------------'
```

```
Reward thread: If you stay sharp, there is a reward waiting in the run: trace sponge.

Capsule Hab 9 // Low Stack

Your rental capsule smells like hot dust and synth noodles. Mirra's shard glints beside a dead
intrusion deck. Rain rattles the hatch east.

You notice locker.

Exits: east.

Useful commands: open locker / search bunk / read terminal.
```

look, search bunk, ask stitch about helix, use shard on nexus...

DAILY Scavenger: Hold 6 items simultaneously

Heat	0/5 (NONE)
Focus	5/5 (SHARP)
Turns	0
Visited	1/28

Fig. 1. *Desktop launch state*

The desktop layout is built around three ideas:

- `Transcript first`
- `Input always visible`
- `Persistent intel`

Main desktop regions

- `App Header`: title, mode tags, and the utilities trigger
- `Scene Feed`: ANSI room card, compass, and suggested commands
- `Transcript`: the authoritative history of what the game has told you
- `Parser Input`: free-text command line plus quick command chips
- `Run Controls`: restart, save, load, and clear
- `Intel Rail`: status, carry, objectives, map, notes, and contacts

Parser Basics

CyberZork accepts compact commands. The language is intentionally familiar if you have played parser IF before.

Most useful commands

Goal	Commands
Re-read the room	`look`, `examine <thing>`, `listen`, `read <thing>`
Find hidden tools	`search bunk`, `search cabinet`, `search reflecting basin`
Move	`north`, `south`, `east`, `west`, `up`, `down`, `go east`
Carry and use items	`take <thing>`, `drop <thing>`, `inventory`, `use <item>`, `install <item>`
Talk to people	`talk <person>`, `ask <person> about <topic>`
Work systems	`hack <target>`, `jack in`, `use <item> on <target>`
Meta actions	`status`, `map`, `journal`, `save`, `load`, `clear`, `help`, `wait`

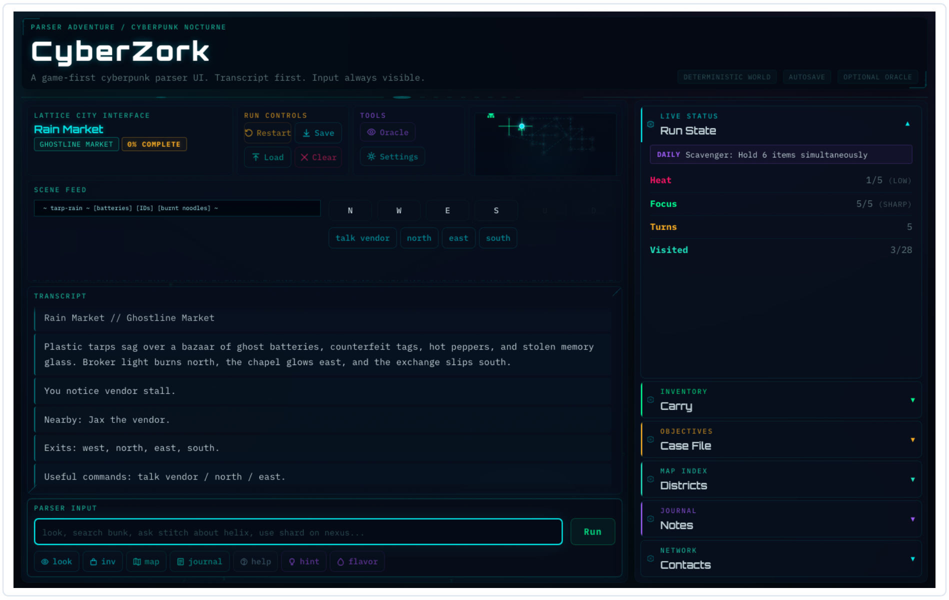
Good habits

- Search aggressively
- Talk to named NPCs after you gain an item or clue
- Use the district map and journal to avoid brute-force wandering
- Treat `hint` as a spoiler-light nudge, not a puzzle skip

First 10 commands worth trying

Command	Why it matters
`look`	Re-read the room and refresh the scene feed
`search bunk`	Useful items are often hidden behind search actions
`inventory`	Confirms what the engine thinks you carry
`east`	The early path opens quickly if you keep moving
`talk vendor`	NPC dialogue often points to the next useful action
`take ghost battery`	A strong example of item acquisition on the main route
`install ghost battery`	Teaches the game’s “tool changes progression” logic
`map`	Keeps the city readable
`journal`	Surfaces current objectives and notes
`help`	Shows the parser’s supported command family

A Typical Early Run



A game-first cyberpunk parser UI. Transcript first. Input always visible.

Rain Market
GHOSTLINE MARKET 0% COMPLETE

~ tarp-rain ~ [batteries] [IDs] [burnt noodles] ~

```
Rain Market // Ghostline Market

Plastic tarps sag over a bazaar of ghost batteries, counterfeit tags, hot peppers, and stolen memory
glass. Broker light burns north, the chapel glows east, and the exchange slips south.

You notice vendor stall.

Nearby: Jax the vendor.

Exits: west, north, east, south.

Useful commands: talk vendor / north / east.
```

look, search bunk, ask stitch about helix, use shard on nexus...

DAILY Scavenger: Hold 6 items simultaneously

Heat	1/5 (LOW)
Focus	5/5 (SHARP)
Turns	5
Visited	3/28

Fig. 2. *Mid-run Rain Market state*

The current critical path is:

- Wake in `Capsule Hab 9`
- Explore east into the market
- Recover the ghost battery and restore your deck
- Work the broker/fixer route to gain the perimeter pass
- Acquire route and spoof tools needed for HELIX access
- Reach the Black Archive and decide how the evidence lands

Optional endings branch from side-route tools such as the `audit key`, `signal mask`, `pump fuse`, and `revision spike`.

Movement, Objectives, And Progress

CyberZork's city is split into six authored districts. The map is compact by design: each location matters, and optional routes are intended to recontextualize the final act rather than dilute the run.

Objective rhythm

- Observation
- Acquisition

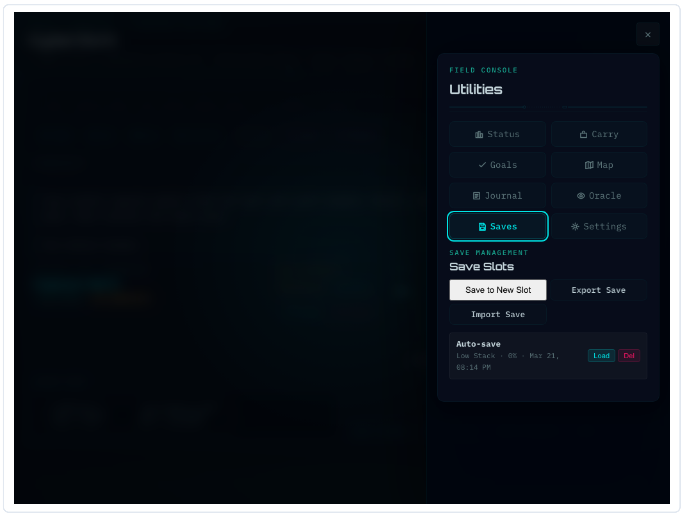
- Access
- Release

Endings

- `Public leak`
- `Full audit release`
- `Ghost release`
- `Controlled burn`
- `Rewrite collapse`

The world stays deterministic. The optional AI layer does not decide endings for you.

Saves, Utilities, And Mobile Layout



Auto-save
Low Stack · 0% · Mar 21, 08:14 PM

Fig. 3. *Tablet utilities drawer with save slots*

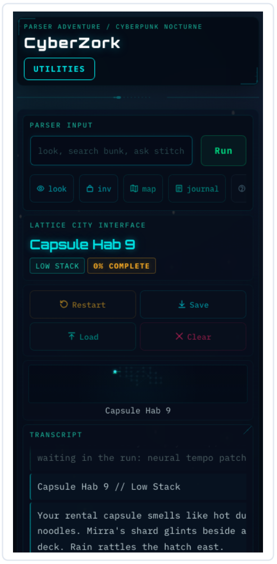
On desktop, utilities behave like focused overlays and persistent rails. On tablet and mobile, they collapse into the same drawer system so the command line stays primary.

Save system

- `Autosave`
- `Manual save`
- `Save Slots`
- `Export`
- `Import`

The runtime validates imported save payloads before they are accepted.

Mobile behavior

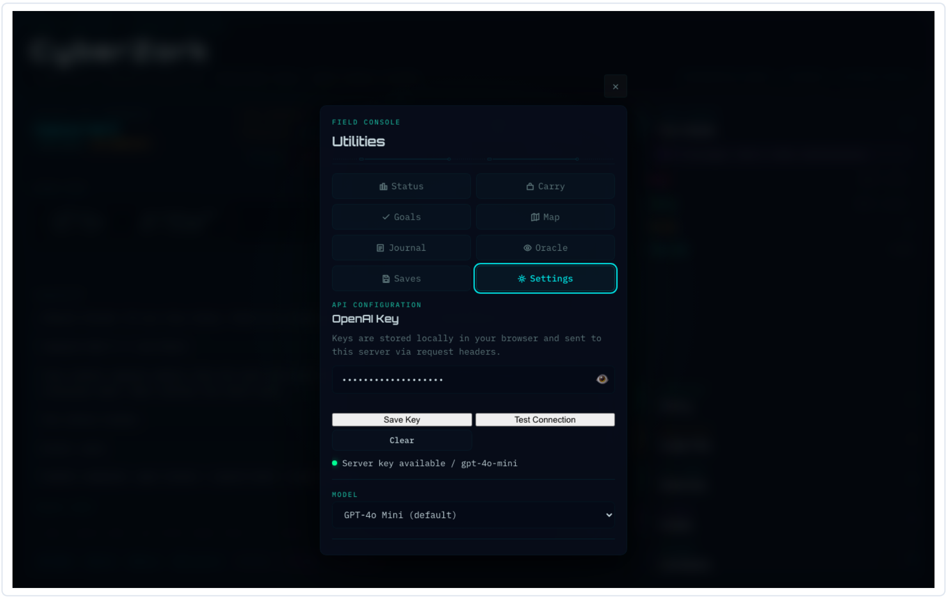


```
waiting in the run: neural tempo patch

Capsule Hab 9 // Low Stack

Your rental capsule smells like hot du
noodles. Mirra's shard glints beside a
deck. Rain rattles the hatch east.
```

Fig. 4. *Mobile parser-first layout*

- the parser stays near the bottom for thumb access
- utilities move behind the `Utilities` button
- Oracle, Settings, Saves, and Progress all remain reachable without losing the current run

Settings, Achievements, And Accessibility Toggles



Keys are stored locally in your browser and sent to this server via request headers.

Server key available / gpt-4o-mini

GPT-4o Mini (default)

Fig. 5. *Settings panel*

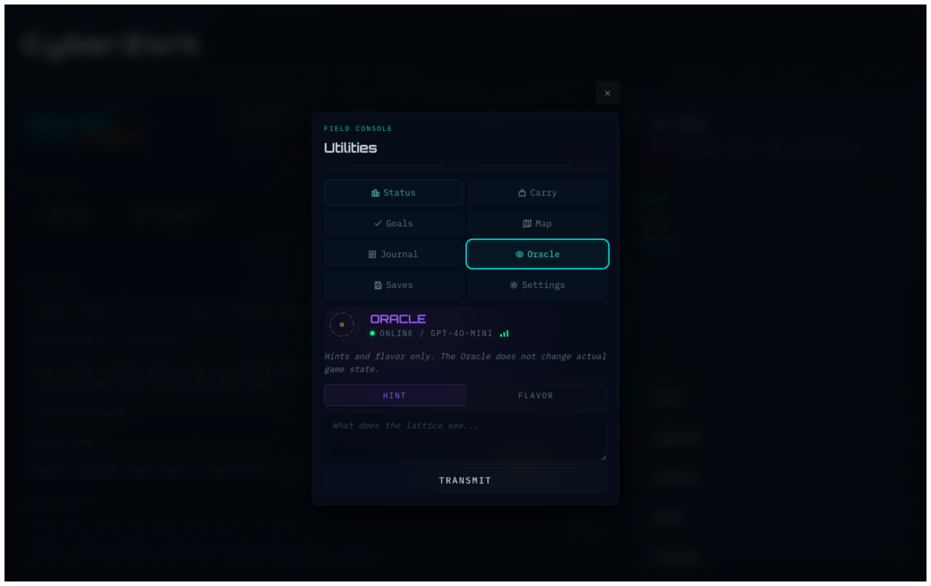
The Settings panel controls:

- local OpenAI key save/test/clear
- model selection
- CRT scanlines and enhanced CRT beam effect
- typewriter toggle
- ambient sound toggle
- keyboard shortcut reference
- achievement gallery

Achievements and daily challenges

- Achievements track run milestones and hidden-route completions
- Personal bests track pace and ending performance
- Daily challenges rotate constraints without changing the authored puzzle truth

Oracle And Optional AI Layer



Hints and flavor only. The Oracle does not change actual game state.

Fig. 6. *Oracle panel*

The Oracle is intentionally constrained.

Oracle modes

- `Hint`: spoiler-light guidance grounded in current state
- `Flavor`: in-world atmospheric response

What the AI layer can do

- draft a boot storyline wrapper
- give constrained hints and flavor
- generate NPC dialogue flavor

What it does not do

- change room state
- change inventory truth
- unlock endings on its own
- replace deterministic puzzle logic

Keyboard Shortcuts

Shortcut	Effect
`/`	Focus parser input
`Esc`	Close panel or blur input
`Tab`	Accept autocomplete
`ArrowUp` / `ArrowDown`	Command history or autocomplete navigation
`Arrow keys`	Movement when focus is not inside a text field

Troubleshooting Matrix

Symptom	Likely cause	Action
Oracle is offline	No valid OpenAI key on the server or in Settings	Add `OPENAI_API_KEY` on the server, or save/test a local key in Settings
Storyline boots in fallback mode	AI unavailable or timeout path triggered	This is expected without AI; the deterministic fallback still plays correctly
Save import fails	Corrupt or mismatched save payload	Export a fresh save from the current build and compare structure
README screenshots look outdated	Docs assets not refreshed after UI changes	Run `npm run docs:shots` and commit the refreshed `docs/assets` files

PARSER CHEAT SHEET

Fast command reference for players, streamers, and testers

Use short commands. Prefer direct verbs.

Start Here

```
look
search bunk
east
talk vendor
take ghost battery
inventory
map
journal
help
```

Movement

```
north
south
east
west
up
down
go east
```

Observation

```
look
examine terminal
search cabinet
read slate
listen
```

Inventory And Item Use

```
inventory
take badge
drop shard
use pass
use fuse on junction
install ghost battery
```

Social

```
talk broker
talk stitch
ask broker about route
ask archivist about audit
```

Systems

```
hack gate
jack in
use spoof chip
use shard on nexus
use revision spike
```

Meta Commands

```
status
map
journal
save
load
clear
hint
flavor
help
wait
```

Good Habits

- Search more than once when a place seems important
- Revisit NPCs after you gain a tool or clue
- Use `inventory` often
- Use `map` and `journal` instead of guessing
- Use `hint` when stuck, not before exploring

SETUP CHECKLIST

Short local, AI, Supabase, Netlify, and docs workflow

Use this as the shortest setup path for local development, optional AI, Supabase, and Netlify deployment.

1. Local App

- `npm install`
- `npm run play`
- open `http://localhost:3000`
- confirm the game boots to `Capsule Hab 9`

2. Local Verification

- `npm run verify`
- `npm run test:ui`
- `npm run test:buttons`

3. Optional OpenAI Setup

Add to `.env` if you want server-side AI:

```
OPENAI_API_KEY=your_key_here
OPENAI_MODEL=gpt-4o-mini
```

- restart the local server
- open Settings
- confirm Oracle or connection test works

4. Supabase Design-Agent Setup

Add to `.env`:

```
SUPABASE_URL=your_project_url
SUPABASE_PUBLISHABLE_KEY=your_publishable_key
SUPABASE_SERVICE_ROLE_KEY=your_service_role_key
DATABASE_URL=postgresql://postgres:your_password@db.your-project.supabase.co:5432/postgres
```

- apply `supabase/migrations/20260318_design_agent.sql`
- run `notify pgrst, 'reload schema';` in Supabase SQL editor
- run `npm run design:health`
- run `npm run design:seed`
- run `npm run design:smoke`

Healthy target:

- `remoteReady: true`
- `serviceReady: true`
- `writeConfigured: true`

5. Netlify Deployment

- `npx netlify login`
- `npx netlify link --git-remote-url https://github.com/Morlock52/CyberZork.git`
- `npx netlify deploy --prod --dir=public --functions=netlify/functions`
- add required Netlify env vars:
- `OPENAI_API_KEY`
- optional `OPENAI_MODEL`
- `SUPABASE_URL`
- `SUPABASE_SERVICE_ROLE_KEY` if using the design agent remotely

6. Documentation Refresh

- `npm run docs:shots`
- confirm `docs/assets/` updated
- review `README.md`
- review `docs/player-guide.md`
- review `docs/admin-guide.md`

7. Final Release Sanity Check

- `curl -s http://localhost:3000/api/health`
- `curl -s http://localhost:3000/api/design/health`
- confirm no stale screenshots remain
- confirm all docs links resolve

ADMIN GUIDE

Verification, API surface, deployment, and design-agent operations

This guide is for maintainers, deployers, and anyone wiring the optional AI and design-agent systems. It covers local verification, API shape, Netlify, Supabase, and documentation maintenance.

Local Run And Verification

Run the project:

```
npm install
npm run play
```

Core verification:

```
npm run verify
npm run test:ui
npm run test:buttons
```

Useful spot checks:

```
curl -s http://localhost:3000/api/health
curl -s http://localhost:3000/api/design/health
```

Runtime API Surface

Local Express and Netlify are kept in parity for:

- `GET /api/health`
- `GET /api/world/default`
- `POST /api/oracle`

- `GET /api/storyline`
- `GET /api/openai/validate`
- `POST /api/npc-dialogue`
- `GET /api/design/health`
- `POST /api/design/retrieve`
- `POST /api/design/world-draft`

The lint gate enforces the critical redirect/function surface, so use `npm run lint` if you change routes.

Optional AI Setup

Server-side env:

```
OPENAI_API_KEY=your_key_here
OPENAI_MODEL=gpt-4o-mini
```

The browser can also store a local key for development through the Settings panel. That key is sent to the local server via request headers and should not be treated as a deployment secret.

The AI layer is intentionally constrained:

- runtime truth stays in the deterministic engine
- Oracle and NPC dialogue are flavor/hint systems
- structured outputs are schema-bound

Source: OpenAI Structured Outputs

Design-Agent And Supabase

The design agent is for retrieval and world drafting, not live runtime world truth.

Required env

```
SUPABASE_URL=your_project_url
SUPABASE_PUBLISHABLE_KEY=your_publishable_key
SUPABASE_SERVICE_ROLE_KEY=your_service_role_key
DATABASE_URL=postgresql://postgres:your_password@db.your-project.supabase.co:5432/postgres
```

Core commands

```
npm run design:health
npm run design:guide
npm run design:seed
npm run design:smoke
```

Health semantics

- `remoteReady`: hosted Supabase is reachable and exposing the expected schema
- `serviceReady`: the design-agent service can still work, including local fallback mode
- `localFallbackReady`: seeded local design notes are available

If `serviceReady` is `true` and `remoteReady` is `false`, the feature is working in degraded mode.

Supabase setup

You must apply the migration in:

`/Users/morlock/01/untitled folder/CyberZork/supabase/migrations/20260318_design_agent.sql`

If you expect browser publishable-key retrieval, the remote project must expose:

- `public.design_documents`
- `public.design_chunks`
- `public.match_design_chunks(...)`

and the corresponding `anon` grants/policies.

Current Supabase guidance distinguishes publishable keys from privileged server keys; keep write credentials server-side. Source: Supabase API keys

Netlify Deployment

Project config:

- publish directory: `public`
- functions directory: `netlify/functions`

- SPA redirect: `/* -> /index.html`
- world boot route: `/api/world/default`

Deploy flow:

```
npx netlify login
npx netlify link --git-remote-url https://github.com/Morlock52/CyberZork.git
npx netlify deploy --prod --dir=public --functions=netlify/functions
```

If you use token auth:

```
export NETLIFY_AUTH_TOKEN=your_token_here
```

Documentation And Screenshots

Refresh the docs asset set:

```
npm run docs:shots
npm run docs:pdf
```

This updates:

- `docs/assets/desktop-launch.png`
- `docs/assets/chapel-run.png`
- `docs/assets/tablet-layout.png`
- `docs/assets/mobile-layout.png`
- `docs/assets/settings-panel.png`
- `docs/assets/oracle-panel.png`
- `docs/assets/social-preview.png`

Primary docs:

- /Users/morlock/01/untitled folder/CyberZork/README.md
- /Users/morlock/01/untitled folder/CyberZork/docs/player-guide.md
- /Users/morlock/01/untitled folder/CyberZork/docs/game-manual.md

Troubleshooting Matrix

Symptom	Likely cause	Action
`design:health` shows missing function/table	Supabase migration not applied or schema cache not reloaded	Run the SQL migration and `notify pgrst, 'reload schema';`
`design:seed` says service role key required	Missing `SUPABASE_SERVICE_ROLE_KEY`	Add the real service-role key to `.env`
`design:smoke` uses `local-fallback`	Remote Supabase unavailable	Fix the remote schema/credentials or accept degraded mode
Netlify deploy hangs	CLI auth/site link missing	Run `npx netlify login` and `npx netlify link ...`
Oracle validates locally but not in Netlify	Missing Netlify env vars	Add `OPENAI_API_KEY` and optional `OPENAI_MODEL` in Netlify settings
Users stay on stale shell after deploy	Old service worker shell cached	The current worker uses network-first navigations; confirm `/sw.js` updated and reload once

Research Notes

- OpenAI Structured Outputs
- Supabase API keys
- Supabase Row Level Security
- MDN: Using Service Workers

END OF DOCUMENT

CyberZork Training Manual // Rev 2026.03.22

Lattice City Documentation Bureau