VULCANEX

Production Intelligence System for O-Ring Vulcanization

A single-file micro-MES (Manufacturing Execution System) built for the shop floor. VULCANEX manages dual hydraulic heat press operations for custom O-ring fabrication — tracking jobs, cure cycles, scrap, redos, inventory, OEE metrics, and shift handoffs in one offline-capable web application.

Built to run on a kiosk tablet bolted to a press station. No server. No internet required. No dependencies. One HTML file.

What It Does

VULCANEX replaces the clipboard-and-spreadsheet workflow for vulcanized rubber cord processing. It handles the full production loop:

1. **Job Queue Management** — Create jobs by vessel number or sales order. Assign material (Viton, Buna-N, EPDM), cross-section diameter (4mm, 5mm, 6mm), mean diameter, quantity, and priority. Jobs flow through `pending → in-progress → done/delayed` states.

2. **Dual Press Control** — Load any queued job onto Press 1 or Press 2 (or both simultaneously for the same job). Each press runs an independent cure timer calculated from material/size lookup tables. Presses operate as deterministic finite state machines: `idle → running → paused → complete → error`.

3. **Cycle Tracking** — Every cure cycle is logged with timestamp, press number, job reference, material, size, duration, and result. Multi-piece jobs track cycle progress (e.g., “Cycle 4 of 12 — 33% done”) with automatic next-cycle workflows.

4. **Redo vs Scrap Separation** — These are two completely different things on the floor and the system treats them that way:

   - **Redo**: The splice failed or the piece needs to go back through the press. The ring is not thrown out — it’s re-cured. Cycle count does NOT change. A redo reason is logged to a separate redo ledger. The press restarts automatically for the same piece.

   - **Scrap**: The piece is garbage and gets thrown out. Cycle count REVERTS by one since that piece is gone and needs to be remade. Scrap reason is mandatory. Logged to the scrap ledger and affects OEE quality score and scrap rate.

Both are tracked independently in the Scrap / Redo tab with separate tables, counts, and clear buttons.

5. **Scrap Enforcement** — Scrap events require a mandatory reason. You cannot bypass the modal. Scrap entries are atomically logged to both the scrap ledger and the immutable audit log.
6. **OEE + Metrics Engine** — Real-time computation of Availability, Performance, Quality, and composite OEE score. Cycle time statistics (average, standard deviation, fastest, slowest) and shift output projection update continuously throughout the day. Quality calculation uses only actual scrap (discarded pieces), not redos.
7. **Transactional Inventory** — Cord stock tracked per material/size combination. Inventory auto-deducts on job close. Low stock warnings trigger at configurable thresholds. No negative stock allowed. All changes recorded in a delta history.
8. **Cut Length Calculator** — Enter a mean diameter, get the cut length (MD × π) and press time instantly. Save calculations for reference.
9. **Shift Handoff Reports** — Auto-generated end-of-shift summaries with production stats, OEE, scrap/redo counts, pending/delayed jobs, and quality data. Add technician notes. Print or export PDF.
10. **Immutable Audit Ledger** — Every state change, job operation, inventory adjustment, scrap event, redo event, and system recovery is logged with a UUID, ISO timestamp, operator tag, job ID, event type, and structured payload. The audit log can never be deleted. Export as JSON or CSV.
11. **PDF Export** — One-click end-of-day production report with KPIs, OEE metrics, material output breakdown, press log, and shift notes. Formatted for print.

Architecture

Centralized State Engine

All application state lives in a single `AppState` object. Every mutation flows through a `dispatch(action, payload)` function that:

- Validates the action
- Updates state

- Persists to `localStorage`
- Logs to the audit ledger
- Triggers UI re-render

No direct state mutation from UI code.

Press Finite State Machine

Press logic is implemented as a strict FSM with five states and a whitelist of valid transitions:

```
idle → running
running → paused / complete / error
paused → running (resume) / idle / error
complete → idle
error → idle
```

Illegal transitions throw controlled errors and log to the audit ledger. A debounce guard prevents rapid toggle exploits. There is no path to double-start a press.

Cycle Result Flow

When a press timer completes, the system checks an internal `isRedo` flag:

- **Normal cycle**: `cyclesDone` increments, stats update, output credited.
- **Redo cycle**: Count stays the same. Logged as `redo-complete`. No output credited.

After a cycle completes, five actions are available:

Button	What It Does
Next	Start the next piece. Cycle count already incremented.
Redo	Piece goes back through the press. Count unchanged. Redo reason logged.
Scrap	Piece is discarded. Count reverts by 1. Scrap reason mandatory.
Close Job	Finalize the job. Credits output, deducts inventory.
Delay	Return job to queue with delayed status and reason.

Crash Recovery

On page load, the system:

1. Validates the localStorage schema version
2. Attempts to restore state safely
3. If corruption is detected, performs a soft reset and logs the event
4. Recovers mid-cycle timing using wall-clock timestamps

If a press was running when the page crashed or refreshed, VULCANEX calculates elapsed time from the stored `startedAt` timestamp minus any accumulated pause duration. If the cycle would have completed during absence, it auto-credits (respecting the `isRedo` flag — redo cycles that completed during absence don't increment count). If time remains, it resumes the timer from the correct position.

Old data from previous schema versions auto-migrates.

Schema Versioning

The localStorage key includes the schema version number. When the schema changes, old data is migrated forward automatically. If migration fails, state is soft-reset and the event is logged.

Offline / PWA

VULCANEX registers an inline service worker with versioned cache invalidation. The entire application works without internet after first load. Cache is automatically purged when the version number changes.

A base64-encoded web manifest is embedded in the HTML for Add to Home Screen support on mobile devices.

Kiosk Hardening

Designed for shop floor touchscreen use:

- All interactive elements have a minimum 44×44px touch target
- Viewport zoom is disabled to prevent accidental pinch-zoom during operation
- Touch-action set to `manipulation` to eliminate 300ms tap delay

- Audio feedback via Web Audio API: distinct tones for cycle complete, error, and scrap events
- Notification toasts auto-dismiss with fade animation

Data Safety

- **Auto-backup**: System checks every 30 minutes and triggers an automatic JSON backup download every 4 hours of active use
- **Manual backup**: One-click full state export from the dashboard
- **Audit trail**: Immutable, append-only event log that survives data clears
- **No negative inventory**: Enforced at the transaction level
- **Mandatory scrap documentation**: Scrap events cannot proceed without a logged reason
- **Redo tracking**: Separate ledger ensures redos never inflate production counts
- **Schema validation**: Every load validates data structure integrity

---

Technical Details

- **Single file**: `index.html` — no build step, no bundler, no framework
- **Zero dependencies**: Vanilla JavaScript, no npm, no CDN runtime deps
- **Typography**: Bebas Neue (display), JetBrains Mono (data/monospace), Inter (body) via Google Fonts
- **Persistence**: `localStorage` with JSON serialization
- **Audio**: Web Audio API oscillator synthesis (no audio files)
- **Print**: CSS `@media print` rules generate a formatted PDF-ready layout
- **Compatibility**: Modern browsers (Chrome 80+, Safari 14+, Firefox 78+, Edge 80+)

Cure Time Reference

Material	4mm	5mm	6mm
Viton	240s	300s	360s
Buna-N	240s	300s	360s
EPDM	180s	240s	300s

Cut length formula: `MD × π`

License