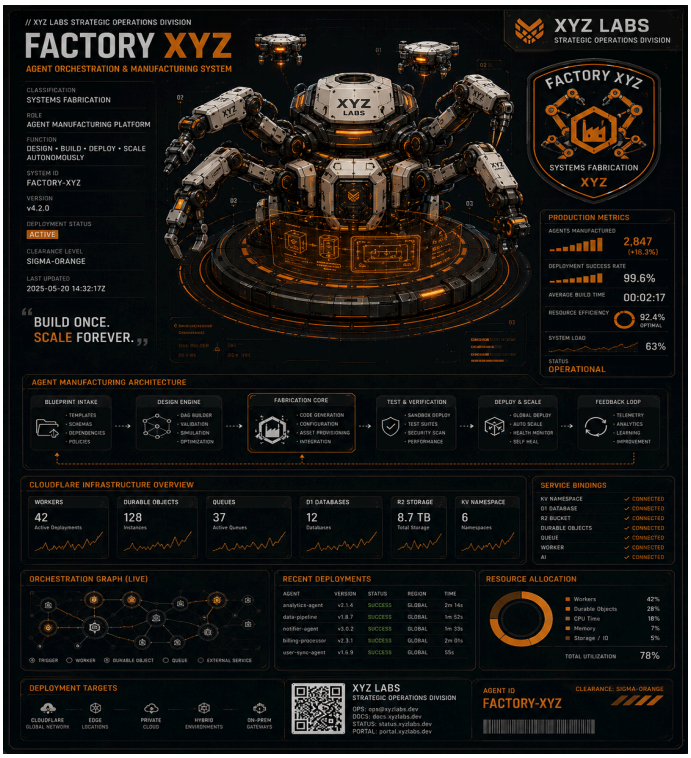
Factory-XYZ
Full Architecture Audit

Software Factory, Runtime-C Integration, Build Pipeline, Warehouse, Pods, Toolbelt, Evidence, and Roadmap

Field	Value
Prepared for	XYZ Labs / blackholecapital
Prepared by	Eila
Audit date	2026-06-25
Document type	Engineering architecture audit
Scope	Factory-XYZ repository and Runtime-C integration, based on inspection outputs and chat/file history
Status	Full report, evidence-grounded, no unsupported claims

Classification: Technical portfolio / engineering handoff / backup companion.



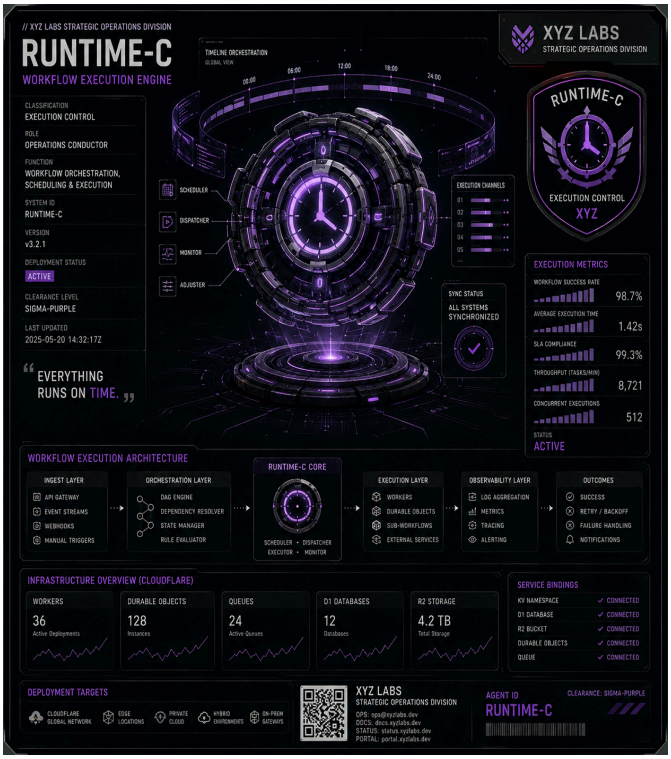


Table of Contents

1. 1. Executive Summary
2. 2. Audit Scope and Method
3. 3. Evidence Inventory
4. 4. System Interpretation
5. 5. High-Level Architecture
6. 6. Factory-XYZ Subsystems
7. 7. Runtime-C Integration
8. 8. Pod Architecture
9. 9. Toolbelt and Factory67
10. 10. Worker Lane Model
11. 11. Run Lifecycle and Artifacts
12. 12. Retrieval and Warehouse Architecture
13. 13. Streaming, Telemetry, and Proof
14. 14. Quality Gates and Governance
15. 15. Backup and Repository Strategy
16. 16. Risks and Technical Debt
17. 17. Roadmap
18. 18. Final Assessment
19. Appendices

1. Executive Summary

Factory-XYZ is best understood as an AI software manufacturing platform rather than a single application repository. The inspected filesystem shows a layered system with manuals, build sheets, indexes, maps, prompts, workspace seeds, Runtime-C execution, pod-based authority, a large external asset warehouse, proof artifacts, sandboxed run records, streaming events, quality gates, and publication outputs.

The strongest architectural signal is the repeated separation of responsibilities: Factory-XYZ defines the manufacturing doctrine, Runtime-C executes, pods own isolated authority and LLM resources, worker lanes produce fragments and proof, and the merge/publish layer assembles final deliverables. This pattern is visible across factory rules, workspace maps, Runtime-C contracts, pod configs, tool manifests, and completed sandbox runs.

Conclusion	Confidence	Reason
Factory-XYZ is a software factory	High	Manuals, build sheets, toolbelt, factory rules, worker lanes, merge and publish artifacts.
Runtime-C is the execution engine	High	README, architecture JSON, runner, submit server, stage matrix, sandbox runs.
Pods are execution authorities	High	Pod A/B/C configs define model, handoff, rootfs, toolbelt, rules, workspace and artifact ownership.
Retrieval is integrated	High	Runtime-C retrieval config points to LanceDB, indexes, resource maps and runtime-c-assets.
Runs are auditable	High	Run folders preserve prompts, LLM receipts, streams, telemetry, metrics, proof and TPS reports.
Factory archive needs trimming	High	Full archive downloaded around 7.8GB, indicating sandbox/golden/quarantine bulk.

2. Audit Scope and Method

This audit consolidates the inspection work performed during the session. It uses terminal outputs, pasted file inspections, prior chat context, and observed directory trees. It does not invent internals that were not inspected. When a component is inferred, the report labels the inference as an architectural interpretation.

Included	Details
Factory-XYZ root	Inspected as the main platform filesystem.
Runtime-C	Inspected as the execution subsystem under Factory-XYZ.
Factory manuals	Factory, foreman, worker, orchestrator, filesystem, proof and stage manuals.
Factory indexes/maps	Runtime retrieval indexes, filesystem maps, resource maps, workspace maps and output contracts.
Pods	Pod A/B/C configs, rootfs, handoff scripts, golden directories and toolbelt.
Toolbelt	Build groups, Factory67, runtime-native tools, validators and merge tools.
Sandbox runs	Completed run layouts including prompts, receipts, stream logs, output, proof and telemetry.
Backups	Runtime-C tar.gz produced at 54MB and pushed to GitHub. Factory full archive downloaded around 7.8GB.

Excluded / Not Fully Verified	Reason
Podman overlay internals	Explicitly excluded due permission errors and low audit value.
node_modules	Excluded from backup and not part of architecture analysis.
Every legacy quarantine copy	Recognized as history/noise, not treated as current authoritative runtime.
Live model output quality	The audit documents architecture, not a current benchmark run.
Complete Factory source code call graph	Enough architecture evidence was collected; exhaustive code graph remains optional.

3. Evidence Inventory

Evidence Item	Observed Role	Audit Value
README_RUNTIME_C.md	Defines Runtime-C contract and active path expectations.	High
RUNTIME_C_ARCHITECTURE.json	Composable architecture source of truth.	High
FORENSIC_AUDIT_LEDGER_20260530.md	Documents proven execution route and historical findings.	High
RUNTIME_C_RETRIEVAL.json	LanceDB and source configuration for retrieval.	High
FILESYSTEM_MAP.md	Runtime root and required directories.	High
RESOURCE_MAP.md	External asset warehouse inventory and indexes.	High
WORKSPACE_MAP.md	Canonical run layout and workspace seed.	High
OUTPUT_CONTRACT.md	Persistent artifacts and delete-later policy.	High
RUNTIME_C_STAGE_MATRIX.md	Stage model and terminal states.	High
RUNTIME_C_TOOL_MANIFEST.json	77 registered tools with hashes.	High
Pod A/B/C POD_CONFIG.json	Execution authority, model routing, rootfs, workspace and artifact ownership.	High
Completed sandbox runs	Concrete evidence of run outputs and traceability.	High

Toolbelt tree	Compiler, lane, validation, merge and Factory67 components.	High

4. System Interpretation

Factory-XYZ is the umbrella platform. Runtime-C sits inside it as the operational runtime. Factory67 and the toolbelt supply the manufacturing logic, buildsheet compilation, lane materialization, validation, registry, provenance and promotion logic. Runtime-C executes builds through pods and six worker lanes. The external asset warehouse provides indexed materials and retrieval context.

```
Factory-XYZ
│
├── Factory doctrine
│   ├── manuals
│   ├── maps
│   ├── prompts
│   └── indexes
│
├── Runtime-C
│   ├── submit server
│   ├── runner
│   ├── pod resolution
│   ├── execution lanes
│   ├── streaming
│   ├── quality gates
│   ├── proof
│   └── publish/index
│
├── Pods A/B/C
│   ├── handoff
│   ├── rootfs
│   ├── golden rules
│   ├── toolbelt
│   └── LLM resource
│
├── Toolbelt / Factory67
│   ├── build groups
│   ├── lane runners
│   ├── validators
│   ├── merge writers
│   ├── provenance
│   └── promotion
│
└── Warehouse
    ├── runtime job queue
    └── external asset warehouse / retrieval indexes
```

5. High-Level Architecture

Layer	Primary Responsibility	Important Artifacts
Factory-XYZ	Manufacturing platform and system container.	factory rules, indexes, maps, prompts, products.
Factory doctrine	Defines how work is decomposed and evaluated.	00_FACTORY_MANUAL, FOREMAN_MANUAL, WORKER_MANUAL.

Runtime-C	Runs builds through pods, lanes, streaming and validation.	submit server, runner, lane-real, streaming orchestrator.
Pods	Own execution authority and LLM routing.	POD_CONFIG.json, rootfs, handoff script, golden toolbelt.
Worker lanes	Divide product generation into specialized outputs.	WA-WF prompts, receipts, streams, worker packets.
Toolbelt	Compiles, runs, validates and merges build groups.	compile-build-sheet, materialize-lanes, run-stage9, write-merge.
Factory67	Coordination/governance brain for build flows.	agent-brain, lane-vote, registry, workspace manager, promotion.
Retrieval	Injects indexed external context into generation.	LanceDB config, asset warehouse, Qwen indexes.
Sandbox	Captures reproducible run state.	prompts, receipts, EVENT_LOG, TPS, telemetry, output, proof.
Publishing	Turns build output into final artifacts and indexes.	merged_product, output, publish scripts, UI artifact index.

6. Factory-XYZ Subsystems

6.1 Factory Manuals

Manual / Rule	Purpose
00_FACTORY_MANUAL_V6_7.md	Defines factory operating doctrine and overall production expectations.
01_FOREMAN_MANUAL_V6_7.md	Defines foreman/operator behavior and coordination expectations.
02_WORKER_MANUAL_V6_7.md	Defines worker behavior and lane obligations.
03_ORCHESTRATOR_GPT_OVERLAY_V6_7.md	Defines orchestration overlay for model-assisted build logic.
04_FILESYSTEM_BASE_CONTRACT_V6_7.md	Defines base filesystem contract and expected layout.
05_RUNTIME_AND_PROOF_CARD_V6_7.md	Defines runtime proof expectations and evidence cards.
06_STAGE_MAP_44_TO_47_V6_7.md	Defines stage lineage and transitions.
WA_HTML_WORKER.md	HTML lane worker guidance.
WB_CSS_WORKER.md	CSS lane worker guidance.
WC_JS_WORKER.md	JavaScript lane worker guidance.
WD_PRODUCT_MANIFEST_WORKER.md	Product manifest worker guidance.
WE_REGRESSION_PROOF_WORKER.md	Regression proof worker guidance.
WF_OPERATOR_REVIEW_WORKER.md	Operator review worker guidance.

6.2 Factory Indexes

Index	Interpreted Function
ASSETS_INDEX.json	Catalogs available assets.
CSS_INDEX.json	CSS-related source/context index.
FACTORY_INDEX.json	Factory resource index.
HTML_INDEX.json	HTML-related source/context index.
JS_INDEX.json	JavaScript-related source/context index.
MANUALS_INDEX.json	Manuals and worker doctrine index.
MAP_INDEX.json	Filesystem/resource/workspace maps index.
OUTPUT_CONTRACT_INDEX.json	Output contract index.
PODS_INDEX.json	Pod file and capability index.
PROMPTS_INDEX.json	Prompt template index.
QWEN_BUNDLES_INDEX.json	Qwen-related bundle index.
QWEN_INDEX.json	Qwen retrieval index.
RESOURCES_INDEX.json	General resource index.
RETENTION_INDEX.json	Retention policy index.
RETRIEVAL_CORPUS_INDEX.json	Retrieval corpus index.
RUN_LAYOUT_INDEX.json	Run directory layout index.

TEMPLATES_INDEX.json	Template index.
TOOLS_INDEX.json	Tool inventory index.
WAREHOUSE_INDEX.json	Warehouse index.
WORKSPACE_SEED_INDEX.json	Workspace seed index.

6.3 Factory Maps and Contracts

Map / Contract	What It Establishes
FILESYSTEM_MAP.md	Required runtime roots, factory locations, warehouse references and asset map links.
RESOURCE_MAP.md	External runtime-c-assets warehouse summary and master asset database.
WORKSPACE_MAP.md	Canonical run root, run layout, keep/delete policy and seed location.
OUTPUT_CONTRACT.md	Persistent output artifacts, proof artifacts, prompts, receipts, TPS reports and temporary cleanup policy.
RETENTION_POLICY.md	Expected retention behavior for generated or temporary assets.

7. Runtime-C Integration

Runtime-C is not separate from Factory-XYZ in the inspected system. It lives under Factory-XYZ and provides the execution layer. The Factory layer supplies manuals, prompts, indexes and rules. Runtime-C converts build sheets into run directories, resolves a pod, executes lanes, streams state, validates outputs, promotes family state and writes publication indexes.

Runtime-C Component	Role in Factory-XYZ
runtime-c-submit-server.cjs	Ingress server and queue entrypoint.
runtime-c-submit-buildsheet.cjs	Buildsheet submission utility.
runtime-c-runner.cjs	Main scheduler, checkpoint and collector.
runtime-c-resolve-pod.cjs	Writes POD_RESOLUTION.json.
runtime-c-exec-pod-authority.cjs	Prepares or verifies pod authority.
runtime-c-lane-real.cjs	Executes worker lanes, retrieves context, calls LLM and writes artifacts.
runtime-c-streaming-orchestrator.cjs	Streaming run orchestration.
runtime-c-finalize-merge.cjs	Final merge step.
runtime-c-quality-gate.cjs	Quality validation.
runtime-c-regression-gate.cjs	Regression validation.
runtime-c-core7-review.cjs	Core7 review layer.
runtime-c-promote-family.cjs	Family promotion and lineage.
runtime-c-write-index.cjs	Public/runtime index writer.
runtime-c-write-tps.cjs	TPS report writer.
runtime-c-write-metrics.cjs	Metrics writer.
runtime-c-write-telemetry.cjs	Telemetry writer.

8. Pod Architecture

Pods are not merely endpoints. Each pod is an execution authority with a root, handoff script, factory rules, toolbelt, rootfs, workspace root, artifact root and LLM resource. Pod A, B and C share the same role shape but point to different model resources.

Pod	Resource	Endpoint	Model	Role
A	home-server-i7-50gb	http://100.105.163.20:11435	qwen3:8b	Primary pod authority.
B	warehouse-rx6800	http://100.105.163.20:11434	qwen2.5-coder:32b	Large model worker / warehouse GPU route.
C	blackhole-local	http://127.0.0.1:11434	qwen2.5-coder:14b	Local execution route.

Pod-Owned Asset	Function
factory_rules	Rules and buildsheet templates.
toolbelt	Executable build, lane, validation and merge tools.
workspace	Working directory for active run execution.
llm_resource	Model endpoint and generation configuration.
artifact_generation	Output production.
proof_generation	Evidence and proof production.
rootfs	Podman/basic root filesystem for execution isolation.
handoff script	Entry point that transfers control from Runtime-C to pod authority.

9. Toolbelt and Factory67

9.1 Build-Groups Toolbelt

Tool Category	Observed Examples	Purpose
Buildsheet compiler	compile-build-sheet-to-build-group.cjs	Converts high-level buildsheet into build group execution plan.
Lane materialization	materialize-lanes.cjs, write-lane-packets.cjs	Creates lane-specific packets and execution structure.
Lane execution	run-stage9-compiled-lane.cjs, run-lane-attempt-controller.cjs	Executes or controls worker lane attempts.
Production wrapper	run-stage9-production-wrapper.sh	Wraps production lane/build flow.
Validation	validate-build-group.cjs, verify-final-product.cjs	Checks build group and final output correctness.
Resolver readiness	write-resolver-readiness.cjs, write-resolver-readiness-and-graph.sh	Creates readiness reports and graph exports.
Merge	write-product-merge.cjs, write-stage9-compiled-merge.cjs	Combines worker fragments into product output.
Snapshot	write-base-snapshot.cjs, validate-base-snapshot.cjs	Maintains base filesystem snapshots.

9.2 Factory67 Coordination Layer

Factory67 Component	Purpose / Interpretation
factory67-agent-brain.cjs	Agentic planning/decision layer.
factory67-core7-lane-vote.cjs	Core7 lane voting or selection.
factory67-core7-overwatch-bridge.cjs	Overwatch/governance bridge.
factory67-overwatch.cjs	Supervisory process.
factory67-workspace-manager.cjs	Workspace preparation and state management.
factory67-worker-sandbox-init.cjs	Initializes worker sandbox.
factory67-worker-self-repair-shim.cjs	Supports worker repair flow.
factory67-artifact-cache-daemon.cjs	Artifact cache process.
factory67-artifacts-api.cjs	Artifact access API.
factory67-native-worker-provenance-registrar.cjs	Registers provenance for native worker artifacts.
factory67-ui-quality-gate.cjs	UI quality gate.
factory67-feature-coverage-enforcer.cjs	Feature coverage enforcement.
factory67-promote.cjs	Promotion logic.
factory67-registry-export.py / registry.py	Registry export / management.
factory67-variant-wave.cjs	Variant generation or wave execution.
product-spec-compiler.cjs	Compiles product specs.
prepare-job-filesystem.cjs	Prepares job filesystem.

10. Worker Lane Model

Lane	Worker	Primary Output
WA	HTML worker	index.html / structural UI.
WB	CSS worker	styles.css / visual presentation.
WC	JavaScript worker	app.js / behavior.
WD	Product manifest worker	PRODUCT_MANIFEST.md and structured product metadata.
WE	Regression proof worker	REGRESSION_PROOF.md and regression evidence.
WF	Operator review worker	OPERATOR_REVIEW_CARD.md and operator-facing review.

Completed run folders confirm that each lane retains PROMPT_*.json, LLM_RECEIPT_*.json, stream/*.jsonl and worker packet material. This makes each lane independently auditable.

11. Run Lifecycle and Artifacts

Observed completed runs include bettys-buns-bakery-clean1 and big-balls-bowling11. Their layouts confirm the run contract.

Run Artifact	Meaning
buildsheet.full.json	Full input specification for the run.
POD_RESOLUTION.json	Selected pod, model and resource resolution.
POD_AUTHORITY.json / IMPORT	Authority import and execution chain of custody.
PROMPT_WA-WF.json	Saved prompt for each worker lane.
LLM_RECEIPT_WA-WF.json	Saved receipt for each LLM call.
stream/WA-WF.jsonl	Streaming event log per lane.
worker_packets/WA-WF	Lane-specific worker input packets.
output/index.html	Generated HTML.
output/styles.css	Generated CSS.
output/app.js	Generated JavaScript.
merged_product/final_manifest.json	Final composed product manifest.
merged_product/merge_report.json	Merge outcome.
QUALITY_GATE.json	Quality gate output.
REGRESSION_GATE.json	Regression gate output.
CLASS_COVERAGE_GATE.json	Class/coverage gate output.
RUNTIME_METRICS.json	Run metrics.
LANE_TELEMETRY.json	Lane telemetry.
EVENT_LOG.jsonl	Global event log.
TEMPORAL_LINEAGE.json	Lineage and timing evidence.
TPS_REPORT.md/json/txt	TPS reporting outputs.
VERIFY_RUN.json	Verification result.

```
Observed run shape:
runtime-c/sandbox/runs/<run-id>/
├── buildsheet.full.json
├── PROMPT_WA.json ... PROMPT_WF.json
├── LLM_RECEIPT_WA.json ... LLM_RECEIPT_WF.json
├── stream/WA.jsonl ... WF.jsonl
├── worker_packets/WA ... WF
├── output/index.html styles.css app.js
├── merged_product/final_manifest.json merge_report.json
├── QUALITY_GATE.json REGRESSION_GATE.json CLASS_COVERAGE_GATE.json
├── EVENT_LOG.jsonl LANE_TELEMETRY.json RUNTIME_METRICS.json
└── TPS_REPORT.md TPS_REPORT.json VERIFY_RUN.json
```

12. Retrieval and Warehouse Architecture

The audit distinguishes two warehouse concepts. The runtime warehouse under Runtime-C is a job lifecycle queue. The external asset warehouse, runtime-c-assets, is the large indexed source used for retrieval and context injection.

Warehouse	Path / Role	Contents
Runtime job warehouse	runtime-c/warehouse	bulk-builds, done/pass/fail/submitted, failed/expired, inbox, locks, queued, repair, trash.
External asset warehouse	/mnt/eila-hot-sidecar/runtime-c-assets	photos, indexes, workflow platforms, templates, CRM platforms, dashboard patterns, registries, design systems.
Master asset DB	runtime-c-assets/indexes/runtime_c_assets.sqlite	Reported 5,037,775 rows.
Qwen retrieval	runtime-c-assets/indexes/qwen	Qwen index system and bundles.
LanceDB	runtime-c/warehouse/retrieval/lancedb	Configured database for runtime_c_assets table.

Retrieval Config Field	Observed Value / Interpretation
enabled	true
engine	lancedb
table	runtime_c_assets
top_k	5
max_chars_per_hit	6000
max_total_chars	24000
sources	factory indexes, RESOURCE_MAP.md, qwen indexes, registries, ASSET_TREE_MAP.md.

13. Streaming, Telemetry, and Proof

Subsystem	Evidence	Purpose
Streaming	runtime-c-streaming-orchestrator, stream-append, stream-status, stream-handoff-watch	Incremental event capture during build execution.
Telemetry	LANE_TELEMETRY.json, runtime-c-write-telemetry.cjs	Runtime visibility per lane.
Metrics	RUNTIME_METRICS.json, runtime-c-write-metrics.cjs	Quantitative build/run metrics.
TPS	TPS_REPORT.md/json/txt, runtime-c-write-tps.cjs	Human-readable and machine-readable performance/status reporting.
Proof	pod_proof, REGRESSION_PROOF.md, proof directory	Run evidence and validation output.
Lineage	TEMPORAL_LINEAGE.json, FRAGMENT_LINEAGE.json	Temporal and fragment traceability.

14. Quality Gates and Governance

Gate / Governance Tool	Role
runtime-c-quality-gate.cjs	General product quality gate.
runtime-c-regression-gate.cjs	Regression gate.
runtime-c-mutation-check.cjs	Mutation or accidental change detection.
runtime-c-class-contract-check.cjs	Class/contract coverage gate.
runtime-c-css-parse-check.cjs	CSS parse validity.
runtime-c-verify-run.cjs	Run verification.
runtime-c-verify-pods.cjs	Pod verification.
runtime-c-verify-wa-artifacts.cjs	WA artifact-specific verification.

factory67-core7-lane-vote.cjs	Core7/lane decision layer.
factory67-ui-quality-gate.cjs	UI-specific quality gate.
factory67-feature-coverage-enforcer.cjs	Feature coverage enforcement.

The presence of independent gates supports a production-oriented interpretation: outputs are generated, checked, merged, proven and published rather than simply emitted by a single prompt.

15. Backup and Repository Strategy

Runtime-C was archived cleanly at approximately 54MB after excluding Podman overlays and node_modules. Factory-XYZ was archived at approximately 7.8GB, which is too large for ordinary GitHub repository storage and likely contains sandbox/golden/legacy/quarantine bulk.

Backup	Status	Recommendation
runtime-c-FULL-20260625-0654.tar.gz	Created, downloaded and pushed to GitHub; 54MB warning but accepted.	Keep as emergency snapshot. Future archives should use GitHub Releases or LFS if repeated.
factory-xyz-FULL-20260625-0655.tar.gz	Created and downloaded; about 7.8GB.	Do not push directly to GitHub. Rebuild clean archive excluding sandbox/golden/legacy bulk.
factory-xyz-CLEAN	Recommended next archive.	Exclude Runtime-C sandboxes, golden copies, legacy-quarantine, Podman roots and node_modules.
Infrastructure audits repo	Common evidence anchor.	Use for PDFs/DOCX/MD audits and QR codes.
Runtime-C repo	Backup stored.	Add README with architecture summary and checksum.
Factory-XYZ repo	Pending clean package strategy.	Use source-only archive or release asset split.

```
Recommended clean Factory archive:
tar \
  --exclude='factory-xyz/runtime-c/.pdm-root' \
  --exclude='factory-xyz/runtime-c/.pdm-run' \
  --exclude='factory-xyz/**/node_modules' \
  --exclude='factory-xyz/runtime-c/sandbox' \
  --exclude='factory-xyz/runtime-c/golden' \
  --exclude='factory-xyz/runtime-c/legacy-quarantine' \
  --exclude='factory-xyz/runtime-c/pods/*/podman-root' \
  --exclude='factory-xyz/runtime-c/pods/*/rootfs' \
  -czf /home/blackhole/factory-xyz-CLEAN-$(date +%Y%m%d-%H%M).tar.gz \
  factory-xyz
```

16. Risks and Technical Debt

Risk / Debt	Impact	Mitigation
Legacy hardcoded roots	May cause active tools to read/write stale locations.	Normalize to RUNTIME_C_HOME and audit path dependencies.
Duplicate pod/golden copies	Archive bloat and confusion over source of truth.	Define canonical pod source and quarantine old copies.
Large Factory archive	Not suitable for GitHub push; difficult to move.	Create clean source archive and separate cold storage for run history.
Runtime vs asset warehouse ambiguity	Terminology confusion.	Document two warehouse meanings explicitly.
Stale UI index state	UI may report incorrect active/completed state.	Make index writer authoritative and remove fallbacks.

TPS fallback bugs	Reports may show wrong endpoints.	Fail closed or report UNRESOLVED instead of default fallback.
Missing master architecture diagram	Harder to onboard reviewers.	Publish Factory-XYZ + Runtime-C diagrams.
Token/credential exposure risk	Secrets may be leaked during troubleshooting.	Rotate exposed PAT and keep future tokens out of pasted logs.
No formal release packaging	Backups may mix source, run state and legacy data.	Create release profiles: source-only, full-runtime, evidence, cold-storage.

17. Roadmap

Priority	Action	Reason
P0	Create clean Factory-XYZ source archive.	Current full archive is too large and includes bulk runtime state.
P0	Generate checksums for Runtime-C and Factory archives.	Provides evidence and integrity checks.
P1	Create Runtime-C README in GitHub repo.	Makes backup understandable without unpacking.
P1	Create Factory-XYZ README and architecture diagram.	Explains system to reviewers/engineers.
P1	Move large archives to release assets or external storage.	Avoids repository bloat.
P2	Normalize hardcoded paths.	Reduces runtime drift.
P2	Separate active source from historical/quarantine data.	Improves auditability and backup size.
P2	Export execution graph from runner and lane tools.	Makes architecture mechanically verifiable.
P3	Build dashboard for run records and proof artifacts.	Turns sandbox evidence into a navigable operator view.
P3	Create cut sheets for Factory-XYZ, Runtime-C and Tracer.	Improves sales/investor communication.

18. Final Assessment

Factory-XYZ demonstrates a coherent and unusually deep software-factory architecture. The system is not merely an application scaffold. It includes doctrine, prompt specialization, deterministic workspace seeds, pod-level authority, multi-lane product generation, retrieval integration, streaming logs, proof artifacts, telemetry, staged gates, merge outputs, promotion logic and publication tools. The platform is real, but it needs packaging discipline: source, runtime, artifacts, legacy history and asset warehouse should be separated so the architecture is easier to preserve and present.

Dimension	Assessment
Architecture maturity	High for a solo-built prototype; strong subsystem separation.
Auditability	High; prompts, receipts, streams, telemetry and proof are preserved.
Operational clarity	Medium; architecture exists but needs cleaned public documentation.
Backup hygiene	Medium; Runtime-C is clean, Factory-XYZ full archive is too large.
Presentation readiness	Medium-high after reports/cut sheets are published.
Commercial value	Strong if presented as Factory-XYZ + Runtime-C + Tracer stack.
Most important next step	Create clean Factory archive and publish architecture docs.

Appendix A: Key Directory Trees

```
runtime-c/factory
├── config/RUNTIME_C_RETRIEVAL.json
├── indexes/
│   ├── ASSETS_INDEX.json
│   ├── FACTORY_INDEX.json
│   ├── OUTPUT_CONTRACT_INDEX.json
│   ├── PODS_INDEX.json
│   ├── PROMPTS_INDEX.json
│   ├── RETRIEVAL_CORPUS_INDEX.json
│   ├── RUN_LAYOUT_INDEX.json
│   ├── TOOLS_INDEX.json
│   └── WAREHOUSE_INDEX.json
├── manuals/
├── maps/
└── prompts/

runtime-c/pods
├── pod-a/
│   ├── bin/
│   ├── golden/
│   ├── POD_CONFIG.json
│   ├── podman-root/
│   └── rootfs/
├── pod-b/
└── pod-c/

runtime-c/pods/pod-a/golden/factory_rules
├── 00_FACTORY_MANUAL_V6_7.md
├── 01_FOREMAN_MANUAL_V6_7.md
├── 02_WORKER_MANUAL_V6_7.md
├── 03_ORCHESTRATOR_GPT_OVERLAY_V6_7.md
├── 04_FILESYSTEM_BASE_CONTRACT_V6_7.md
├── 05_RUNTIME_AND_PROOF_CARD_V6_7.md
├── 06_STAGE_MAP_44_TO_47_V6_7.md
└── buildsheets/

runtime-c/workspace-seed/factory
├── indexes/
├── manuals/
│   ├── WA_HTML_WORKER.md
│   ├── WB_CSS_WORKER.md
│   ├── WC_JS_WORKER.md
│   ├── WD_PRODUCT_MANIFEST_WORKER.md
│   ├── WE_REGRESSION_PROOF_WORKER.md
│   └── WF_OPERATOR_REVIEW_WORKER.md
├── maps/
└── prompts/
    ├── POD_A_PROMPT_WA.md
    ├── POD_A_PROMPT_WB.md
    ├── POD_A_PROMPT_WC.md
    ├── POD_A_PROMPT_WD.md
    ├── POD_A_PROMPT_WE.md
    └── POD_A_PROMPT_WF.md
```

Appendix B: Command Log Highlights

```
Runtime-C archive created:
tar --exclude='runtime-c/.pdm-root' --exclude='runtime-c/.pdm-run' \
    --exclude='runtime-c/node_modules' --exclude='runtime-c/**/node_modules' \
    -czf /home/blackhole/runtime-c-FULL-20260625-0654.tar.gz runtime-c
```

```
Runtime-C archive size:
54M

Factory archive created:
tar --exclude='factory-xyz/runtime-c/.pdm-root' --exclude='factory-xyz/runtime-c/.pdm-run' \
    --exclude='factory-xyz/**/node_modules' \
    -czf /home/blackhole/factory-xyz-FULL-20260625-0655.tar.gz factory-xyz

Factory archive downloaded:
approximately 7.8GB
```

Appendix C: Glossary

Term	Definition
Factory-XYZ	Umbrella software factory platform.
Runtime-C	Execution engine inside Factory-XYZ.
Factory67	Coordination/tooling layer with agent brain, build tools, validators and promotion.
Pod	Execution authority with model endpoint, rootfs, toolbelt and factory rules.
Buildsheet	Input specification for a generated product/build.
Worker lane	Specialized generation lane WA-WF.
LLM receipt	Saved evidence for a model generation event.
TPS report	Runtime status/performance report.
Proof card/proof artifact	Evidence generated to validate product/run output.
Warehouse	Either runtime job queue or external asset/retrieval warehouse depending on context.
Golden	Canonical seed/tooling copy used by pods or runtime.
Quarantine	Historical or deprecated material retained but not treated as current source of truth.

End of Factory-XYZ Full Architecture Audit