# Agent Forge – Comprehensive Documentation (Phase 0 – 5)

**Purpose**: This document provides a single reference containing the final project folder layout, detailed module & file descriptions, and the complete roadmap from Phase 0 through Phase 5 including final production-polish tasks.

**Final Repository Layout**

agent\_forge\_project/  
├── README.md  
├── requirements.txt  
├── config.yaml  
├── .gitignore  
├── config\_loader.py  
├── run\_forge.py  
├── forge\_ui.py  
│  
├── forge\_core/  
│ ├── \_\_init\_\_.py  
│ ├── schemas.py  
│ ├── forge\_logging.py  
│ ├── component\_registry.py  
│ ├── agent\_builder.py  
│ ├── behavior\_tree.py  
│ ├── evaluation.py  
│ ├── trace\_loader.py  
│ ├── llm\_judge.py  
│ └── safety\_guardrails.py  
│  
├── capabilities/  
│ ├── \_\_init\_\_.py  
│ ├── base\_capability.py  
│ ├── math\_skill.py  
│ └── web\_search\_tool.py  
│  
├── agents/  
│ ├── \_\_init\_\_.py  
│ ├── base\_agent.py  
│ ├── simple\_agent.py  
│ ├── coordinator\_agent.py  
│ └── bt\_agent.py  
│  
├── definitions/  
│ ├── agents/simple\_agent\_v5.yaml  
│ ├── skills/math\_skill\_v1.yaml  
│ └── tools/web\_search\_tool\_v1.yaml  
│  
├── strategies/rag\_search\_strategy.yaml  
└── test\_cases/basic\_math.yaml

**Module & File Descriptions**

* **forge\_core.schemas –** Pydantic data-models for validating YAML component definitions.
* **forge\_core.component\_registry –** Loads YAML definitions into an in-memory registry, reports validation errors.
* **forge\_core.agent\_builder –** Dynamic builder that wires Agents + Skills + Tools based on registry entries.
* **forge\_core.behavior\_tree –** Tiny Behaviour-Tree executor and YAML loader (Sequence, Selector, Action).
* **forge\_core.evaluation –** Batch testing harness, produces pass/fail & logs.
* **forge\_core.forge\_logging –** Structured JSONL log writers + helpers.
* **forge\_core.safety\_guardrails –** Keyword filter (placeholder for full moderation pipeline).
* **capabilities.math\_skill –** Arithmetic operations with step-level trace logging.
* **capabilities.web\_search\_tool –** DuckDuckGo text search wrapper.
* **agents.simple\_agent –** Echo agent skeleton; demonstrates Skill invocation.
* **agents.bt\_agent –** Behaviour-Tree-driven agent.
* **agents.coordinator\_agent –** Prototype for multi-agent orchestration.
* **run\_forge.py –** CLI: build an agent and run a prompt.
* **forge\_ui.py –** Streamlit UI: prompt box, agent picker, trace viewer stub.

**Roadmap & Phase Deliverables**

1. Phase 0 – Baseline

* Core Python package skeleton (forge\_core, agents, capabilities).
* Declarative YAML definitions & registry.
* Behaviour-Tree engine MVP.
* Streamlit demo UI (manual prompt).
* RAG stub & basic ChromaDB integration.
* Evaluation harness with keyword checks.

1. Phase 1 – Sandboxing & Plugin API

* System-call / subprocess isolation (seccomp / firejail on Linux).
* Declarative plugin manifest & signature validation.
* User-permission prompts for high-risk capabilities.
* Tool marketplace stub (index & discovery endpoints).
* Versioned capability registry + auto-update.

1. Phase 2 – Multi-Agent & Coordination

* CoordinatorAgent + worker pattern with YAML team definitions.
* Shared state schema & in-memory message bus.
* Built-in coordination protocols (Hierarchical, Round-Robin, Consensus).
* Trace visualisation of inter-agent messages.
* LLM-based dynamic task allocation experiment.

1. Phase 3 – Automated Evolution (Genetic / RL-HF)

* Population manager + YAML mutation operators.
* Fitness scoring: keyword pass rate, LLM-judge score, resource usage.
* Distributed evaluation orchestrator (Ray / Celery).
* Checkpointing & lineage tracking of top agents.
* Dashboard summarising generation metrics.

1. Phase 4 – Cloud UI & Sharing Hub

* Docker-compose / Kubernetes deployment manifests.
* OAuth2 login & multi-tenant workspaces.
* Agent gallery with ratings, comments, and one-click "Run".
* Vector-store management panel.
* User notebooks for iterative agent tuning.

1. Phase 5 – Final Polish & Production Hardening

* Black + Ruff formatting, full type-hints, mypy clean.
* mkdocs-material documentation published to GitHub Pages.
* CI/CD (GitHub Actions): tests → lint → build → PyPI release → Docker push.
* Security audit (OWASP dependency-check, secret scanning).
* Performance profiling & optimisation of hot paths.
* Interactive trace viewer (tree + timeline) with dark/light theme.
* Accessibility compliance (WCAG 2.1) & i18n framework.
* Installer script & `pip install agent-forge` distribution.
* Release notes, migration guides, and community code-of-conduct.

**Production Readiness Checklist**

* 100 % test coverage & all tests green.
* Static type analysis passes with zero errors.
* CI pipeline green on main & feature branches.
* Package publishes to TestPyPI & PyPI without issues.
* Docker image boots UI in <30 s on 2-CPU node.
* No critical vulnerabilities in dependency scans.
* Documentation site passes link & spell checker.
* User telemetry opt-in, GDPR compliance confirmed.