

Technical Executive Summary

Seven Fortunas AI-Native Enterprise Infrastructure

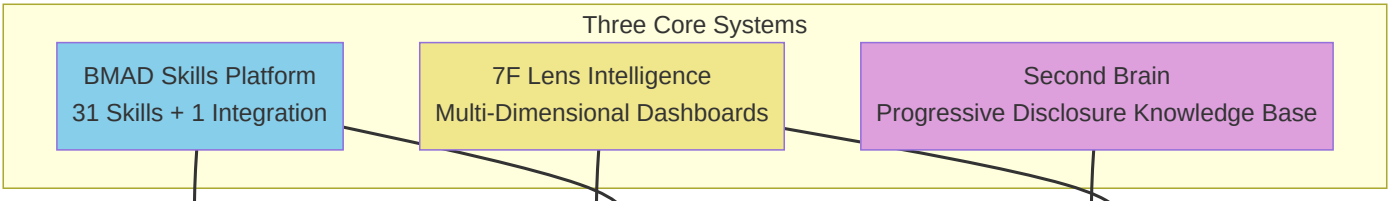
Project: GitHub Organization, Second Brain & 7F Lens Intelligence Platform **Timeline:** 5-7 Day MVP (Days 0-5 baseline + buffer) **Status:** Production-Ready (90.5/100 autonomous readiness score) **Budget:** 5 – 10/month MVP, 130-150/month Phase 2

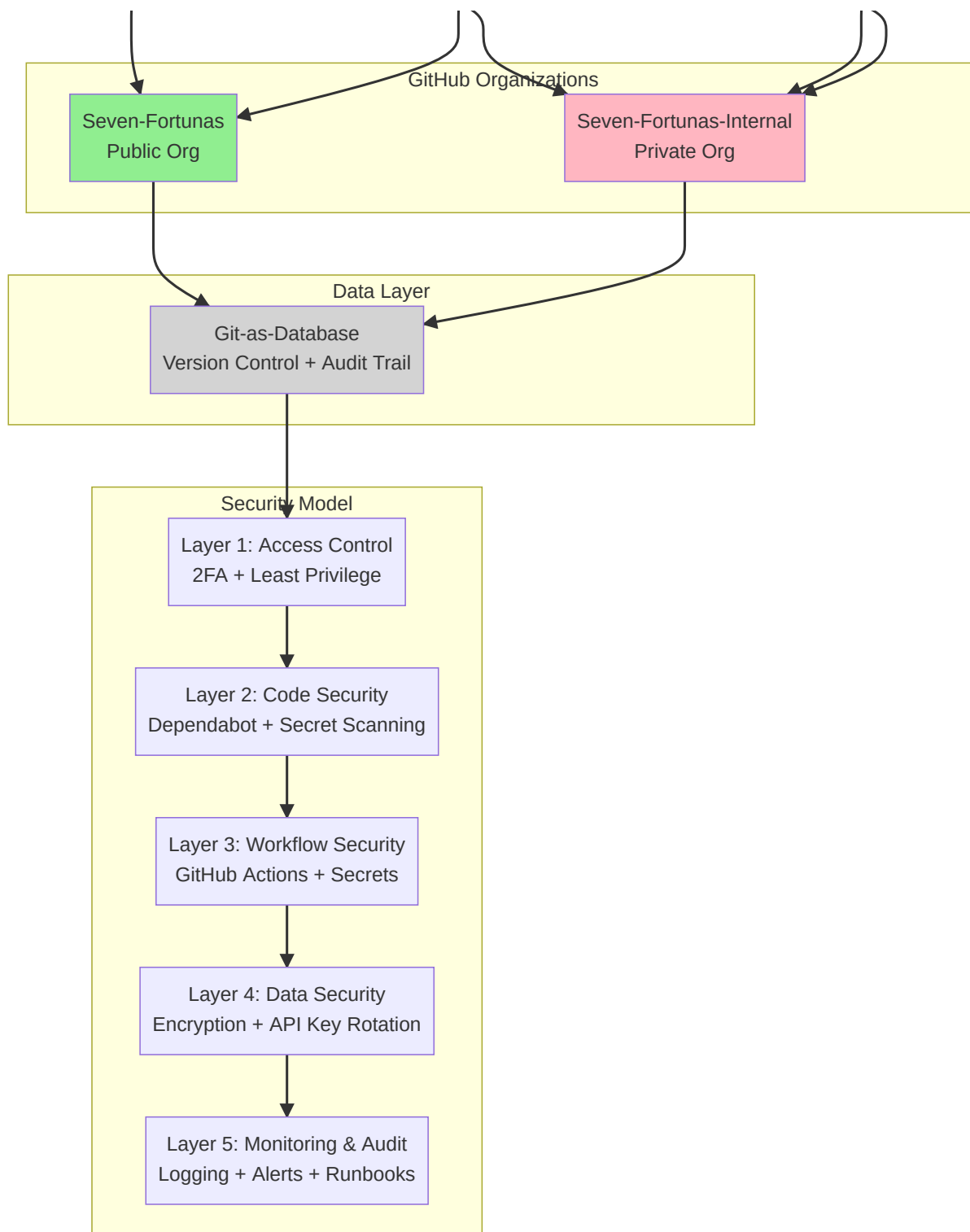
Project Overview

Seven Fortunas is building an **AI-native enterprise infrastructure**—a living, intelligent system where company knowledge, culture, and operations exist as structured, AI-accessible data. Unlike traditional organizations that retrofit AI onto static documents, Seven Fortunas designs for AI collaboration from inception: every document, every repository, every standard is optimized for both human understanding and AI ingestion.

The infrastructure establishes professional, scalable foundations rapidly using autonomous AI agents to build 60-70% of features (18-25 of 28 features). This reduces traditional 9-week setup to 1.75 weeks—demonstrating measurable productivity gains when infrastructure is purpose-built for AI collaboration.

System Architecture





Architecture Style: Layered monolithic architecture (Presentation → Business Logic → Data), Git-as-database **Deployment Model:** GitHub-hosted (Free → Team → Enterprise tiers) **Security Model:** 5-layer defense in depth

Key Architecture Decisions

ADR-001: Two-Organization Model

Decision: Separate public (`Seven-Fortunas`) and private (`Seven-Fortunas-Internal`) GitHub organizations

Rationale: Security isolation (proprietary IP protection), brand separation (showcase vs operations), compliance readiness (SOC 2 prep Phase 1.5)

Consequences:

- Upfront cost: 8 hours org setup (Day 1)
 - GitHub Private Mirrors App development: 16-24 hours (Phase 2) for selective publishing
 - Ongoing: Maintain two org configurations, manage cross-org access
-

ADR-002: Git-as-Database Pattern

Decision: Store all operational data in Git repositories (dashboards, Second Brain, configuration)

Rationale: Built-in version control, audit trail, disaster recovery via Git history, no separate database infrastructure

Consequences:

- Commits = transactions, branches = isolation, rollback via `git revert`
 - Disaster Recovery: RTO <1 hour, RPO <6 hours
 - Limitation: Not suitable for high-frequency writes (>1 commit/minute)
 - Benefit: Zero database operational overhead
-

ADR-003: BMAD-First Approach

Decision: Leverage 70+ existing BMAD (Business Method and Development) workflows instead of building custom skills from scratch

Rationale:

- 81% cost reduction (7,000 *vs* 36,000)
- 4.5x faster delivery (1.75 weeks vs 9 weeks)
- 290 hours effort saved
- Proven workflow patterns

Coverage:

- 20 BMAD skills adopted as-is
- 5 adapted with Seven Fortunas customization (using `bmad-bmb-create-workflow` process)
- 6 custom skills built from scratch
- **Total:** 31 skills + 1 integration (MVP: 25 skills)

Custom Workflow Process: Custom skills use BMAD's `bmad-bmb-create-workflow` meta-workflow to generate skill structure:

1. Build first 3 skills manually (establish patterns)
2. Adapt `bmad-bmb-create-workflow` → `7f-skill-creator` (Phase 1.5)
3. Use `7f-skill-creator` for all future skills (Phase 2+), reducing creation time from 5 hours to <2 hours

ADR-004: Progressive Disclosure Architecture (Second Brain)

Decision: Three-level knowledge hierarchy (index.md → domain directories → specific documents) with YAML frontmatter for AI filtering

Rationale:

- Reduces AI token usage (load only relevant context)
- Faster context loading for AI agents
- Scalable to 1000+ documents
- Obsidian-compatible for human browsing

Implementation:

- All docs ≤ 3 levels deep
 - YAML frontmatter tags (`context-level` , `relevant-for` , `last-updated`) enable AI-driven navigation
 - Index-first loading pattern (AI agents load index.md → specific sections as needed)
-

ADR-005: Autonomous Agent Implementation

Decision: Use Claude Code SDK with two-agent pattern (initializer + coding agent) to build 60-70% of infrastructure features autonomously

Rationale: Autonomous implementation is a powerful tool for sophisticated applications, but it critically depends on the quality, clarity, and thoroughness of product requirements. This approach requires expert knowledge in:

- Requirements engineering (atomic, testable, specific acceptance criteria)
- System architecture (clear dependencies, bounded scope)
- Software engineering practices (testing strategies, validation procedures)

Implementation Context: This specific project serves as a validation case study for autonomous implementation workflows. Success metrics and lessons learned will be documented upon completion to inform future autonomous implementations.

Consequences:

- **Upfront Investment:** Requirements must achieve $\geq 75/100$ autonomous readiness score (this project: 90.5/100)
 - **Expert Required:** Product Owner/Architect must validate requirements quality before agent execution
 - **Risk Mitigation:** Bounded retry logic (3 attempts), test-before-pass requirement, human validation gates
 - **Benefit:** 60-70% autonomous completion demonstrates AI-driven productivity for leadership
 - **Validation:** Post-implementation report will document actual vs. target performance, challenges encountered, and refinements needed
-

Implementation Timeline

Phase	Duration	Key Activities	Success Criteria
Day 0: Foundation	10-11h	BMAD v6.0.0 installation, 18 skill stubs, GitHub CLI auth (jorge-at-sf), app_spec.txt generation, autonomous agent setup	Environment validated, agent executable, 28 features in app_spec.txt
Days 1-2: Autonomous Build	48h	Two-agent execution, 28 features implementation, automated testing, progress monitoring	18-25 features "pass" status, feature_list.json tracking
Day 3: Validation	6-8h	4 founder aha moments (Patrick 2h, Buck 2-3h, Jorge 2h, Henry 3h)	≥99.5% secret detection, infrastructure validated
Day 4: Bug Fixes	12-16h	Parallel debugging (Jorge 6-8h, Patrick 3-4h, Buck 2-3h, Henry 1h)	Critical bugs fixed, 20+ features operational
Day 5: Polish & Demo	5h	Documentation review (2h), polish (1h), leadership demo (2h)	Professional presentation ready

Buffer: 23% contingency (16 hours) built-in for debugging and availability constraints

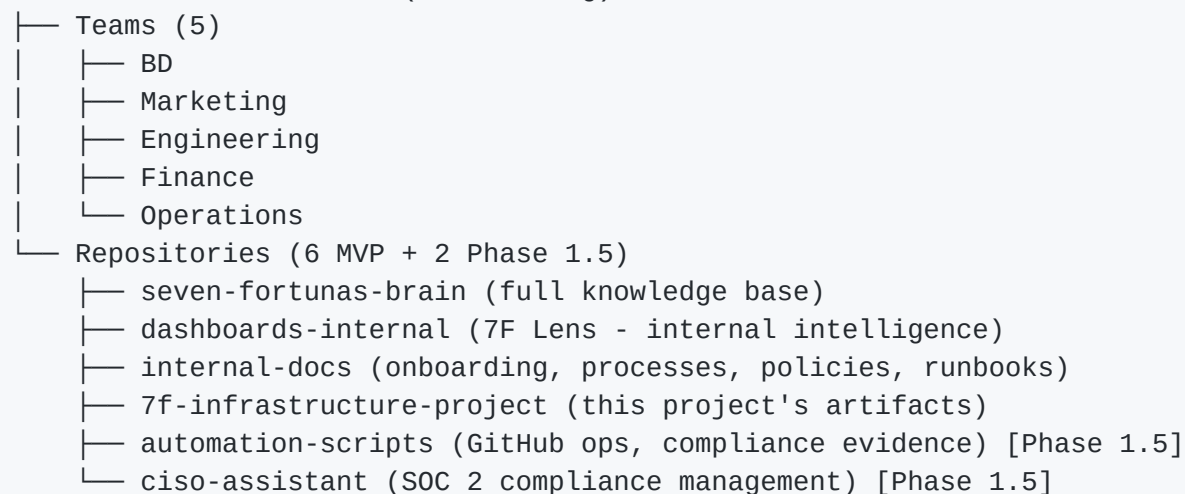
Key Technical Specifications

GitHub Organizations (Tree Structure)

Seven-Fortunas (Public Org)



Seven-Fortunas-Internal (Private Org)



Security Configuration:

- 2FA enforced (100% compliance)
- Default permission: none (explicit grants required)
- Branch protection on all main branches
- Secret scanning + push protection enabled

Second Brain: Progressive Disclosure Knowledge Base

What It Is: A structured, AI-accessible knowledge repository using progressive disclosure architecture to optimize both human browsing and AI context loading.

Structure:

```

seven-fortunas-brain/second-brain-core/
├─ index.md                # Hub (AI agents load FIRST)
├─ brand/                  # Brand identity (colors, fonts, voice,
positioning)
├─ culture/                # Mission, vision, values, operating principles
├─ domain-expertise/       # Tokenization, EduPeru, Compliance, Security
├─ best-practices/         # Engineering, operations, runbooks, standards
└─ skills/                 # Custom BMAD skills documentation

```

Strategic Relevance:

- **Token Optimization:** AI agents load index.md first (~200 tokens), then specific sections as needed (not entire 50+ docs)
- **Scalability:** Supports 50+ team members without information overload, scales to 1000+ documents
- **Dual-Audience:** Markdown for humans (readable in GitHub/Obsidian), YAML frontmatter for AI filtering
- **Self-Documenting:** README.md at every directory level (FR-6.1 requirement), enables 1-2 day onboarding for new members
- **AI-Accessible:** Structured format allows AI agents to find relevant context without human intervention

Documentation System (FR-6.1): All repositories include comprehensive documentation:

- Root README: Project overview, quick start, navigation
- Directory READMEs: Purpose, contents, usage
- Code READMEs: Setup, dependencies, examples
- Architecture READMEs: Design decisions, patterns, ADRs

This self-documenting architecture enables new team members to onboard in 1-2 days (self-service) and supports the revolutionary productivity gains of AI-first infrastructure.

7F Lens Intelligence Platform: Multi-Dimensional Dashboards

Vision: Track strategic intelligence across internal and external domains using automated aggregation + AI-generated summaries.

External-Looking Dashboards (Public): Track industry trends, competitive landscape, emerging technologies:

- **MVP:** AI Advancements Tracker (auto-update every 6 hours)
 - Sources: arXiv, Reddit r/MachineLearning, YouTube, X API, GitHub releases
 - Weekly AI summaries (Claude API)
 - GitHub Pages hosting (zero infrastructure cost)
- **Phase 2:** Fintech Trends Dashboard, EduTech Dashboard (Peru market focus), Security Intelligence Dashboard

Internal-Looking Dashboards (Private - Phase 2): Track organizational progress, compliance posture, project status:

- **FR-8.2 (Functional Requirement):** Sprint Dashboard (GitHub Projects integration, sprint velocity + burndown + blockers)
- **FR-8.3:** Project Progress Dashboard (technical and business project tracking)
- **FR-5.4:** CISO Assistant Controls Dashboard (SOC 2 compliance posture, control drift alerts <15 min)

Why It Matters:

- Leadership stays informed with 5-minute weekly review (AI summaries reduce signal-to-noise)
- Public dashboards showcase Seven Fortunas' AI expertise (thought leadership)
- Internal dashboards provide real-time visibility into compliance, security, and project health

Dual Use of Sprints: Technical & Business Projects

FR-8.1: Unified Sprint Management The sprint tracking system is **project-type agnostic**—it tracks both:

Technical Projects:

- Engineering stories (feature implementation, bug fixes, refactoring)
- Code reviews (peer review, security audit, architectural validation)
- Architecture tasks (ADR creation, system design, dependency upgrades)

Business Projects:

- Campaign execution (marketing launches, content calendars, social media)
- Content creation (blog posts, white papers, investor decks)
- Compliance workflows (SOC 2 evidence collection, policy reviews, security audits)

Implementation:

- BMAD sprint workflows (`bmad-bmm-create-sprint` , `bmad-bmm-sprint-review`) support flexible terminology
- GitHub Projects provides Kanban board visualization (Backlog → In Progress → Review → Done → Blocked)
- Sprint dashboard (FR-8.2, Phase 2) shows velocity, burndown, blockers regardless of project type

Strategic Benefit: Unified sprint management enables leadership visibility across all company work (engineering + operations + marketing) using consistent metrics and workflows.

Business-Side Agent Workflows

BMAD Business Method (BMM) Skills: AI agents support business operations beyond code development:

Product & Project Management:

- `bmad-bmm-create-prd` - Generate Product Requirements Documents (used to create this project's requirements)
- `bmad-bmm-create-architecture` - Generate system architecture documents
- `bmad-bmm-create-epic` - Create epics from high-level requirements
- `bmad-bmm-create-story` - Create user stories with acceptance criteria
- `bmad-bmm-create-sprint` - Sprint planning (Phase 2)
- `bmad-bmm-sprint-review` - Sprint retrospectives (Phase 2)

Operations & Documentation:

- `bmad-bmm-create-sop` - Create standard operating procedures
- `bmad-bmm-transcribe-audio` - Transcribe audio files to text (meeting notes, interviews)
- `7f-sop-generator` - Generate SOPs with Seven Fortunas branding (adapted from `bmad-bmm-create-sop`)

Onboarding & Profile Management (Phase 2):

- `7f-manage-profile` - User profile management (deferred to Phase 2)
 - Creates user profiles with role information (engineer, operations, marketing, leadership)
 - AI agents use profile context to tailor responses and recommendations
 - Supports 1-2 day self-service onboarding for new team members

Strategic Benefit: Business-side agents demonstrate AI productivity gains beyond engineering, enabling operations, marketing, and compliance teams to work at "AI speed."

Security Model (5 Layers)

Layer 1: Access Control

- GitHub OAuth + 2FA enforcement (100% compliance)
- Least privilege (team-based grants, not individual)
- `jorge-at-sf` verification script (BLOCKING all GitHub operations if incorrect account is active)

Layer 2: Code Security

- Dependabot (weekly updates, auto-merge if tests pass)
- Secret scanning + push protection ($\geq 99.5\%$ detection rate)
- Branch protection (PR reviews, status checks, no force push)

Layer 3: Workflow Security

- Approved GitHub Actions allowlist (`actions/checkout@v4`, `actions/setup-python@v5`, `anthropics/claude-api@v1`)
- GitHub Secrets management (org-level, 90-180 day rotation)
- OIDC for short-lived tokens (Phase 2)

Layer 4: Data Security

- Encryption: AES-256 at rest, TLS 1.3 in transit
- API key rotation: 90 days (GitHub/Claude), 180 days (external)
- Data classification: Public / Internal / Confidential / Restricted

Layer 5: Monitoring & Audit

- Structured JSON logging (6 event types tracked)
 - Security alerts (Dependabot, secret scanning, failed auth, unusual API usage)
 - Incident runbooks (compromised keys, unauthorized access, data leak)
-

Technology Stack

Core Infrastructure:

- GitHub (hosting, storage, automation)
- GitHub Pages (website)
- GitHub Actions (aggregation, CI/CD)
- Claude API (AI processing, summaries)
- Markdown + Git (Second Brain)

Languages & Tools:

- Python 3.11+ (aggregation scripts, automation)
- Markdown + YAML (content, skills, configuration)
- HTML + CSS (website, dashboards)
- Bash (scripts, deployment)

Dependencies:

- feedparser, praw, anthropic, requests, pyyaml, python-dotenv

Cost Structure:

- **MVP:** 5 – 10/month (*ClaudeAPI* 5-7, Reddit/YouTube free tier)
 - **Phase 2:** 130 – 150/month (*ClaudeAPI* 10-15, Matrix 12 – 20/month, *GitHubTeam* 4/user/month × 4, X API \$100/month)
 - **Labor:** 7,000total (70hours × 100/h), vs \$36,000 build-from-scratch (81% savings)
-

Production Readiness

Comprehensive Validation Complete (6 Phases):

Phase	Result	Evidence
Requirement Count	✅ PASS	67 requirements (33 FRs + 34 NFRs), all present
Autonomous Readiness	✅ PASS (90.5/100)	Score 20% above threshold
Implementation Detail	✅ PASS	Day 0-5 executable, all scripts specified
Content Coverage	✅ PASS	14/14 source docs mapped, zero gaps
Fix Impact	✅ PASS	100 fixes, zero information loss
Cross-References	✅ PASS	100% accuracy, zero broken links

Overall Assessment: Production-ready for Day 0 implementation **Confidence Level:** HIGH (systematic validation across 6 dimensions) **Blocking Issues:** NONE

Conclusion

Seven Fortunas AI-Native Enterprise Infrastructure demonstrates a **paradigm shift in technical foundation building**: designing for AI collaboration from inception rather than retrofitting AI onto static systems. The comprehensive validation (90.5/100 autonomous readiness score, zero information loss across 6 validation phases) confirms production readiness.

The autonomous agent approach (60-70% automated infrastructure build) serves as a validation case study for sophisticated autonomous implementation workflows, with success metrics and lessons learned documented after implementation. The BMAD-first methodology delivers **81% cost reduction and 4.5x faster delivery**, establishing a replicable pattern for future development.

Next Action: Execute Day 0 Foundation (10-11 hours) using master-implementation.md

Document Version: 2.0 **Date:** 2026-02-16 **Author:** Mary (Business Analyst Agent) with Jorge (VP AI-SecOps) **Status:** Production-Ready **Changes:** Removed non-technical content, expanded technical specifications, added architecture diagram, clarified ADRs