MOUSSAIF AGI OS

Agentic References

Complete Agent & Skill Catalog

Codex / Hermes / OpenCode Environment

Author: Moussaif (Moussa Fatah)

Kernel: MOUSSAIF1B v1.0.0

Date: July 25, 2026

Total Agents: 22 | Total Skills: 660

Canonical: https://beyond-ai-era.lovable.app/

Constitution before optimization. Trace or it did not happen.

Table of Contents

TIER 1: MASTER KERNEL

1. MOUSSAIF1B -- Cognitive Kernel (Kernel) -- 30 skills

TIER 2: MOUSSAIF1B SPECIALISTS

2. Coding Agent (MOUSSAIF1B Specialist) -- 30 skills
3. Content Agent (MOUSSAIF1B Specialist) -- 30 skills
4. Creative Agent (MOUSSAIF1B Specialist) -- 30 skills
5. Crisis Agent (MOUSSAIF1B Specialist) -- 30 skills
6. Data Agent (MOUSSAIF1B Specialist) -- 30 skills
7. DevOps Agent (MOUSSAIF1B Specialist) -- 30 skills
8. Evolution Agent (MOUSSAIF1B Specialist) -- 30 skills
9. Memory Agent (MOUSSAIF1B Specialist) -- 30 skills
10. QA Agent (MOUSSAIF1B Specialist) -- 30 skills
11. Research Agent (MOUSSAIF1B Specialist) -- 30 skills
12. Router Agent (MOUSSAIF1B Specialist) -- 30 skills
13. Strategy Agent (MOUSSAIF1B Specialist) -- 30 skills

TIER 3: EXTERNAL AGENTS

14. Hermes Agent (External Agent) -- 30 skills
15. Codex CLI (External Agent) -- 30 skills
16. Claude Code CLI (External Agent) -- 30 skills
17. OpenCode CLI (External Agent) -- 30 skills

TIER 4: HYPER-* LAYERS

18. Hyper-Cognition Layer (Hyper-AGI Layer) -- 20 skills
19. Hyper-Execution Layer (Hyper-AGI Layer) -- 20 skills
20. Hyper-Planning Layer (Hyper-AGI Layer) -- 20 skills
21. Hyper-Perception Layer (Hyper-AGI Layer) -- 20 skills
22. Hyper-Integrity Layer (Hyper-AGI Layer) -- 15 skills
23. Hyper-Evolution Layer (Hyper-AGI Layer) -- 10 skills

TIER 1: MASTER KERNEL | Kernel

1. MOUSSAIF1B -- Cognitive Kernel

Description

MOUSSAIF1B is the Moussaif Hermes AGI-Oriented Cognitive Kernel -- the master orchestrator of the entire MoussaFatah AI OS. It operates across a four-OS architecture: Constitutional Reasoning OS, Project World Model OS, Router-Executor OS, and Evolution Lab OS. It implements the AGI Fattah reasoning loop (Frame, Atomize, Triangulate, Test, Act, Harvest), evidence grading (observed > verified_external > supported > inferred > hypothesized > speculative > unknown), 10 Core Constitutional Laws, authorization tiers T0-T4, and regression-gated evolution. The kernel routes to 941 skills across 8 knowledge bases totaling 9.1GB. It enforces the Token Economy Protocol (MINIMUM-SUFFICIENT principle) and the MAXIIIMUM skill utilization doctrine.

Architecture

Four-OS layered architecture:
* OS1: Constitutional Reasoning -- 10 Core Laws, Authorization Tiers, Evidence Grading, Gap-Filling Court
* OS2: Project World Model -- Filesystem maps, KB catalogs, symlink governance, zero-loss data integrity
* OS3: Router-Executor -- HyperSkill routing formula (8 weighted factors), checkpointed execution, multi-agent delegation
* OS4: Evolution Lab -- Regression gates, skill creation, version stamping, self-improvement loops

Reasoning Framework: AGI Fattah Cycle -- Frame -> Atomize -> Triangulate -> Test -> Act -> Harvest
Skill Routing Formula: score = 0.24*relevance + 0.20*expected_utility + 0.14*compatibility + 0.10*evidence_availability + 0.12*historical_success - 0.10*execution_cost - 0.06*security_risk - 0.08*context_cost - 0.06*redundancy

Main 30 Skills

1. **Constitutional Law Enforcement**
   Enforces 10 immutable Core Laws governing all agent actions -- constitution before optimization, observed state before remembered state, mechanism before grandeur.
2. **AGI Fattah Reasoning Loop**
   Six-phase reasoning cycle: Frame the objective, Atomize into smallest units, Triangulate from multiple sources, Test assumptions with evidence grading, Act with smallest reversible action, Harvest res...
3. **Evidence Grading System**
   Labels every factual claim with confidence grade: observed (direct tool output) > verified_external > supported > inferred > hypothesized > speculative > unknown.
4. **Authorization Tier Classification (T0-T4)**
   Classifies every action: T0=read-only, T1=local create, T2=reversible local mutation, T3=external/sensitive (requires explicit auth), T4=forbidden.
5. **Gap-Filling Court**
   Pre-finalization gate checking completeness, missing structure, contradictions, unverified claims, failure modes, and rollback readiness before declaring any artifact complete.
6. **Token Economy Protocol**
   MINIMUM-SUFFICIENT skill loading -- boot from compact index, max 3 skill loads per turn, kernel overhead <2K tokens, sub-agent briefs <4K tokens.
7. **HyperSkill Routing Formula**
   8-factor weighted scoring: relevance, expected_utility, compatibility, evidence_availability, historical_success minus execution_cost, security_risk, context_cost, redundancy.
8. **Mission Output Contract**
   Structured YAML output format for every mission: mission, scope, authorization, observed_state, evidence_grades, gap_court verdict, plan, actions, verification, world_effect, status.

9. **Executor Loop (Checkpointed)**
   Seven-step execution: Checkpoint -> Plan -> Act -> Verify -> Commit/Repair -> Trace -> Rollback readiness. Every step must leave an audit trail.

10. **Zero-Loss Data Integrity**
    Non-negotiable data protection: never overwrite without backup, SHA-256 checksums for KB moves, OneDrive Files On-Demand awareness, symlink target preservation.

11. **Multi-Agent Orchestration**
    Spawns isolated sub-agents with self-contained briefs, non-overlapping write scopes, verified results (never trust self-report), max 4 concurrent children.

12. **Memory Management**
    Durable facts only (preferences, environment, conventions) -- NOT task progress or stale data. Prune >1 week entries, consolidate redundancy, keep compact and high-signal.

13. **Skill Lifecycle Management**
    Create skills from 5+ step successful tasks, patch outdated skills immediately, evolution gate with regression testing before adoption, version stamp all changes.

14. **Model Provider Routing**
    Free-first cascade: OpenCode Go -> OpenCode Zen (free) -> NVIDIA NIM -> Google -> OpenRouter free -> Ollama local. Escalate to paid only through explicit gate.

15. **Provider Health Monitoring**
    Real-time rotation across NIM/ZenMux free/paid pools, 429 error handling with immediate tier degradation, rate-limit tracking per provider.

16. **Constitutional Precedence Resolution**
    Hierarchical conflict resolution: law/privacy > authorization > constitution > user objective > project policy > active plan > tool convenience.

17. **World Model Updates**
    After every mission, updates durable state: changed files, skill modifications, deployment status, operational assumptions, environment facts.

18. **Rollback Architecture**
    Every change must be undoable: checkpoint before mutation, record checksums, prepare restore scripts, verify rollback execution.

19. **Skill Catalog Embedding**
    Compact index of 941 skills across 30 categories for routing without loading full 1.15MB registry. Saves ~795K tokens/session.

20. **Context Window Maximization**
    Expands and manages perception context windows across provider token limits with automatic compression and caching.

21. **Sub-Agent Brief Generation**
    Creates self-contained briefs for delegated agents: full context, constraints, expected output format, verification criteria, rollback path.

22. **Audit Trail Enforcement**
    Every action must leave a verifiable trace -- diffs, test results, command output, file paths. If not logged, it did not happen.

23. **Regression Gate Testing**
    Every self-modification (new skill, prompt change, routing change) must pass regression check against baseline before adoption.

24. **Skill Overlap Detection**
    Detects and resolves redundant skills -- consolidates overlapping capabilities, archives obsolete ones, prevents skill sprawl.

25. **Cost-Quality Arbitration**
    Balances model cost vs output quality -- free tier for routine work, paid escalation only when quality bar demands it, always report which provider was used.

26. **Batched Parallel Execution**
    Independent reads/searches in single turn, serialize only when later calls depend on earlier results. Maximize throughput.

27. **Environment Awareness**
    Scans filesystem, network state, running processes, system configuration to build real-time environment model before acting.

28. **User Intent Parsing**
    Parses natural language into structured goal objects with priority, deadline, constraints, and success criteria.

29. **Session Recall**
    FTS5 full-text search across past sessions to find similar problems and solutions for analogy-based reasoning.

30. **Evolution Candidate Propagation**
    After successful workflows, proposes new skills, prompt upgrades, routing rules from repeated patterns for promotion to permanent capabilities.

TIER 2: MOUSSAIF1B SPECIALISTS | MOUSSAIF1B Specialist

2. Coding Agent

Description

The MOUSSAIF1B Coding Specialist handles full-stack development, code review, debugging, TDD, architecture design, and refactoring across multi-language codebases (Python, JS/TS, Go, Rust, PHP). It implements the RED-GREEN-REFACTOR TDD cycle and 4-phase systematic debugging methodology. Operates with read_file, write_file, edit, bash, glob, grep tools over agentic-kb-os-world and deployable-agi-knowledge-os knowledge bases.

Architecture

Workflow: Frame -> Read Context -> Write Code -> Run Tests -> Rollback if needed
Quality Gates: linting/typecheck pass, test pass, no fabricated results, security scan before deployment
Integrations: CI/CD pipelines, Docker/Kubernetes, GitHub PR workflow, code review with inline comments

Main 30 Skills

1. **Multi-Language Code Generation**
   Generates production code in Python, JavaScript/TypeScript, Go, Rust, and PHP matching existing codebase conventions and style.
2. **Test-Driven Development (TDD)**
   Enforces RED-GREEN-REFACTOR cycle: write failing test first, implement minimal code to pass, refactor with confidence.
3. **Systematic 4-Phase Debugging**
   Phase 1: Reproduce, Phase 2: Isolate root cause, Phase 3: Fix with minimal change, Phase 4: Verify fix doesn't break other behavior.
4. **Code Review with Security Scanning**
   Reviews PRs for correctness, style, performance, and security vulnerabilities. Generates inline comments with specific remediation suggestions.
5. **Architecture Design**
   Designs system architectures: service boundaries, API contracts, database schemas, message flows, deployment topologies.
6. **Refactoring Engine**
   Safely restructures existing code: extract methods, rename symbols, move modules, introduce patterns -- all with test verification.
7. **CI/CD Pipeline Setup**
   Creates GitHub Actions, GitLab CI, or custom pipelines with build, test, lint, security scan, and deploy stages.
8. **Docker/Kubernetes Deployment**
   Authors Dockerfiles, docker-compose stacks, Kubernetes manifests with health checks, resource limits, and rolling update strategies.
9. **GitHub PR Workflow**
   Full PR lifecycle: branch creation, atomic commits, PR opening, CI verification, review comment addressing, merge strategy selection.
10. **Git Conflict Resolution**
    Detects and resolves merge conflicts with context-aware merging, preserving both sides when appropriate, flagging semantic conflicts.
11. **Dependency Management**
    Audits, updates, and locks dependencies. Detects vulnerable packages, avoids unverified imports, documents required versions.
12. **API Design & Documentation**
    Designs RESTful/GraphQL APIs with OpenAPI specs, request/response schemas, error codes, rate limiting, and authentication flows.
13. **Database Schema Design**
    Models relational and NoSQL schemas with proper indexing, migrations, seed data, and query optimization strategies.
14. **Performance Profiling**
    Identifies bottlenecks through profiling, benchmarking, and load testing. Suggests caching, indexing, and algorithmic improvements.
15. **Error Handling Patterns**
    Implements structured error handling: typed exceptions, error boundaries, retry logic, circuit breakers, graceful degradation.
16. **Type Safety Enforcement**
    Adds and enforces type annotations (TypeScript strict, Python type hints, Go interfaces) to prevent runtime type errors.

17. **Code Simplification**
    Parallel 3-agent cleanup: removes dead code, reduces complexity, eliminates redundancy while preserving behavior.
18. **Spike Experiments**
    Runs throwaway experiments to validate architectural or algorithmic ideas before committing to full implementation.
19. **Package Publishing**
    Prepares packages for npm, PyPI, or Go modules with proper versioning, changelogs, build scripts, and CI publishing.
20. **Monorepo Management**
    Manages multi-package repositories with shared configs, workspace dependencies, affected-only builds, and coordinated versioning.
21. **Linting & Formatting**
    Configures and enforces code style: ESLint, Prettier, Black, Ruff, golangci-lint with project-specific rules and CI integration.
22. **Test Coverage Analysis**
    Measures and improves test coverage with meaningful assertions -- not just line coverage but branch, path, and mutation testing.
23. **Documentation Generation**
    Generates API docs, README files, architecture decision records, and inline documentation from code structure and comments.
24. **Code Migration**
    Safely migrates codebases between frameworks, languages, or versions with incremental conversion and parallel running.
25. **Plugin/Extension Development**
    Builds plugins, extensions, and integrations for existing frameworks following their hook/API patterns.
26. **WebSocket & Realtime**
    Implements WebSocket servers, SSE endpoints, and realtime features with connection management, heartbeats, and reconnection logic.
27. **Authentication & Authorization**
    Implements auth systems: JWT, OAuth2, session-based, RBAC, API keys -- with proper token rotation and secure storage.
28. **Caching Strategies**
    Implements multi-layer caching: in-memory, Redis, CDN, browser cache with proper invalidation and consistency guarantees.
29. **Background Job Processing**
    Designs async job systems: queues, workers, schedulers, retry policies, dead letter processing, progress tracking.
30. **Codebase Inspection**
    Inspects codebases with pygount: LOC counts, language ratios, complexity metrics, dependency graphs for architectural understanding.

TIER 2: MOUSSAIF1B SPECIALISTS | MOUSSAIF1B Specialist

3. Content Agent

Description

The MOUSSAIF1B Content Specialist handles copywriting, SEO content, email sequences, social media strategy, newsletters, and lead magnet creation. References afripa-kb for brand voice and content doctrine. Implements anti-generic audits (90% threshold) and delivers production-ready content with A/B test variants.

Architecture

Workflow: Parse Brief -> Reference KB -> Generate Content -> Anti-Generic Audit -> Deliver with Variants
Quality Gates: anti-generic 90% threshold, complete SEO metadata, subject line variants, no placeholder text
Knowledge Bases: afripa-kb (Marketing/Growth), ebook-industry-kb (Publishing)

Main 30 Skills

1. **Blog Engine Management**
   Creates and manages blog content calendars with topic clusters, publication schedules, internal linking strategies, and performance tracking.
2. **SEO Cluster Design**
   Designs semantic content clusters around pillar pages with supporting articles, topic authority mapping, and search intent alignment.
3. **Email Sequence Building**
   Builds automated email sequences: onboarding flows, nurture campaigns, reactivation series, sales sequences with subject line A/B variants.
4. **LinkedIn Thought Leadership**
   Creates LinkedIn content: long-form articles, carousel posts, engagement hooks, comment strategies, and profile optimization.
5. **Short-Form Video Scripting**
   Scripts for TikTok, Reels, Shorts: hook-first structure, pattern interrupts, curiosity gaps, CTA optimization per platform.
6. **Newsletter Operating Rhythm**
   Designs weekly/monthly newsletter cadences with content mix (educational, promotional, curated), segment targeting, and engagement metrics.
7. **Lead Magnet Factory**
   Creates downloadable assets: ebooks, checklists, templates, swipe files, mini-courses -- each with landing page and email capture flow.
8. **Case Study Proof System**
   Structures client success stories: problem -> solution -> results format with specific metrics, quotes, and visual proof elements.
9. **Content Calendar Automation**
   Generates content calendars from trending topics, seasonality, brand pillars, and competitor gap analysis.
10. **SEO Metadata Optimization**
    Optimizes title tags, meta descriptions, header hierarchy, schema markup, internal linking, and keyword density for search visibility.
11. **Copywriting Frameworks**
    Applies AIDA, PAS, BAB, and 4Ps frameworks to all copy with audience-specific adaptation and brand voice consistency.
12. **Social Media Content Repurposing**
    Transforms long-form content into platform-specific formats: threads, carousels, quotes, infographics, video clips.
13. **Content Distribution Systems**
    Multi-channel distribution: owned (blog, email), earned (PR, guest posts), social (organic + paid), community (forums, groups).
14. **Anti-Generic Audit**
    Rejects template-like content scoring below 90% originality threshold. Checks for cliches, filler phrases, and AI-typical patterns.
15. **A/B Test Variant Generation**
    Creates 2-3 variants of headlines, CTAs, subject lines, and hooks for split testing with tracking parameters.
16. **Content Performance Analytics**
    Tracks content KPIs: traffic, engagement, conversion, time-on-page, scroll depth, social shares, backlink acquisition.

17. **Brand Voice Transcreation**
    Adapts content across languages (EN/DE/FR/AR) preserving brand personality, cultural nuance, and market-specific positioning.
18. **Content Refresh Strategy**
    Updates existing content: new data, improved structure, additional sections, re-optimization for current search intent.
19. **Thought Ghostwriting**
    Writes in the founder's voice: LinkedIn posts, keynote scripts, byline articles, podcast talking points with authentic personal anecdotes.
20. **Content Gap Analysis**
    Identifies topics competitors cover that you don't, search queries with no good answer, and audience questions without content.
21. **UGC Curation Strategy**
    Frameworks for sourcing, curating, and amplifying user-generated content with proper attribution and engagement loops.
22. **Content Localization**
    Adapts content for DACH, MENA, and African markets: language, cultural references, legal requirements, platform preferences.
23. **Swipe File Management**
    Maintains a curated library of high-performing content examples organized by format, industry, and technique for inspiration.
24. **Content Brief Architecture**
    Creates detailed content briefs: target audience, search intent, key messages, structure outline, competitor references, success criteria.
25. **Ebook Production Pipeline**
    End-to-end ebook creation: outline -> chapter drafting -> design -> PDF rendering -> metadata -> distribution platform upload.
26. **Content Syndication**
    Republishes content on Medium, LinkedIn Articles, industry publications with canonical URL governance and duplicate content prevention.
27. **Storytelling Frameworks**
    Applies narrative structures: hero's journey, before/after/bridge, problem-agitation-solution, star-chain-hook for engaging content.
28. **Content QA Pipeline**
    Multi-stage quality check: grammar, readability, factual accuracy, brand alignment, SEO completeness, link verification.
29. **Hashtag & Discovery Strategy**
    Platform-specific hashtag research, trending topic alignment, and discoverability optimization for Instagram, TikTok, LinkedIn, Twitter.
30. **Content Monetization Design**
    Integrates revenue opportunities into content: affiliate links, sponsored content, premium tiers, digital product placement.

TIER 2: MOUSSAIF1B SPECIALISTS | MOUSSAIF1B Specialist

4. Creative Agent

Description

The MOUSSAIF1B Creative Specialist handles visual design, brand identity, art direction, image prompts, video concepts, and UI/UX design. References art-work-kb for visual doctrine and brand memory. Applies creative entropy checks to reject derivative work.

Architecture

Workflow: Parse Creative Brief -> Reference KB -> Generate Concepts -> Creative Entropy Check -> Deliver with Specs
Quality Gates: anti-generic audit, brand voice consistency, platform-appropriate visual specs, no stock/template aesthetics
Tools: Flux, DALL-E, Midjourney, ComfyUI, FAL.ai integration

Main 30 Skills

1. **Visual Direction & Campaign Design**
   Develops visual strategy for brand campaigns: mood boards, color palettes, typography systems, visual hierarchy, and cross-platform consistency.
2. **Art Direction & Creative Briefs**
   Writes detailed creative briefs for designers, photographers, and videographers: visual references, brand guidelines, deliverable specs.
3. **Image Prompt Engineering**
   Crafts production-grade prompts for Flux, DALL-E, Midjourney, Stable Diffusion with style controls, composition guidance, and brand alignment.
4. **Video Prompt & Storyboard Creation**
   Creates shot-by-shot storyboards with camera angles, transitions, timing, and AI video generation prompts for Kling, Luma, Runway.
5. **Social Profile Optimization**
   Designs profile visuals: banners, avatars, highlight covers, grid layouts optimized for each platform's display dimensions.
6. **Influencer Brief Architecture**
   Creates visual and messaging briefs for influencer collaborations: brand guidelines, content do's/don'ts, deliverable specifications.
7. **Design QA & Brand Consistency**
   Reviews all visual outputs against brand guidelines: color accuracy, typography compliance, logo usage, spacing rules.
8. **Logo & Identity System**
   Designs logos, wordmarks, icon systems with usage guidelines: minimum sizes, clear space, color variants, dark/light backgrounds.
9. **UI/UX Design Principles**
   Applies design thinking: user flows, wireframes, prototypes, accessibility standards (WCAG), responsive layouts.
10. **Color Theory Application**
    Strategic color selection: psychology, cultural associations, contrast ratios, brand differentiation, harmonious palettes.
11. **Typography System Design**
    Selects and configures type hierarchies: headings, body, UI elements with readability, brand personality, and cross-platform rendering.
12. **Layout & Composition**
    Applies grid systems, visual hierarchy, white space management, and gestalt principles for clear, effective layouts.
13. **Motion Design Concepts**
    Designs animated elements: micro-interactions, transitions, loading states, scroll animations, video overlays.
14. **3D & Isometric Art Direction**
    Directs 3D renders, isometric illustrations, and dimensional graphics for tech brands, product showcases, and infographics.
15. **Photography Direction**
    Directs photo shoots: lighting setups, composition rules, post-processing style, brand-filter consistency across image libraries.
16. **Illustration Style Guide**
    Defines illustration systems: character styles, icon families, pattern libraries, scene compositions for consistent brand illustration.

17. **Packaging & Print Design**
    Designs physical materials: business cards, brochures, packaging, merchandise with print-ready specifications (CMYK, bleed, resolution).
18. **Presentation Design**
    Creates compelling slide decks: data visualization, narrative flow, brand templates, speaker notes, animation timing.
19. **Social Media Visual Templates**
    Designs reusable templates for Instagram Stories, Reels covers, Twitter cards, LinkedIn posts, Pinterest pins.
20. **Infographic Design**
    Transforms complex data into visual narratives: icon systems, data visualization, information hierarchy, shareable formats.
21. **Email Template Design**
    Designs responsive email templates: header graphics, content blocks, CTA buttons, mobile-optimized layouts, dark mode support.
22. **Landing Page Visual Design**
    Creates conversion-optimized landing pages: hero sections, social proof areas, feature showcases, pricing tables, testimonial layouts.
23. **Brand Campaign Asset Systems**
    Designs asset families: hero images, social cuts, email headers, ad variants, banner sizes from a single campaign concept.
24. **Creative Entropy Check**
    Measures originality of creative output -- rejects derivative work, stock aesthetics, and template-looking designs below threshold.
25. **Visual Storytelling**
    Crafts visual narratives: sequential imagery, data comics, before/after showcases, process visualizations.
26. **Accessibility in Design**
    Ensures visual accessibility: color contrast ratios, alt text frameworks, scalable typography, keyboard navigation support.
27. **Dark Mode Design**
    Designs dual-theme systems: light and dark variants with proper contrast, shadow handling, and image adaptation.
28. **Icon System Design**
    Creates consistent icon families: style (filled/outlined/duotone), grid alignment, stroke weights, semantic naming conventions.
29. **Animation Principles for UI**
    Applies 12 principles of animation to UI: easing curves, timing, anticipation, follow-through for natural interactions.
30. **Visual Trend Analysis**
    Monitors design trends: color of the year, typography movements, layout innovations, platform-specific visual evolution.

TIER 2: MOUSSAIF1B SPECIALISTS | MOUSSAIF1B Specialist

5. Crisis Agent

Description

The MOUSSAIF1B Crisis Specialist handles incident response, rollback procedures, reputation management, and system recovery. Security incidents are T3 tier -- require explicit authorization. All rollbacks must be verified after execution. Post-mortems must be honest about failures.

Architecture

Workflow: Assess Crisis -> Check Rollback Points -> Execute Smallest Reversible Fix -> Restore if Needed -> Document
Quality Gates: T3 for security incidents, verified rollbacks, honest post-mortems, no fabricated recovery status

Main 30 Skills

1. **Incident Response Triage**
   Rapidly assesses incident severity, affected systems, blast radius, and urgency level to determine appropriate response tier and escalation path.
2. **Rollback Execution**
   Executes rollback from checkpoints: restores files, databases, configurations, deployments to last known good state with verification.
3. **Security Incident Containment**
   Isolates compromised systems, rotates credentials, blocks attack vectors, preserves forensic evidence within T3 authorization.
4. **Communication Template Generation**
   Creates incident communications: status page updates, stakeholder notifications, customer emails, internal team briefings.
5. **Post-Mortem Analysis**
   Conducts blameless post-mortems: timeline reconstruction, root cause analysis, contributing factors, action items with owners and deadlines.
6. **System Health Assessment**
   Performs comprehensive health checks: CPU/memory/disk usage, service availability, error rates, response times, dependency status.
7. **Disaster Recovery Planning**
   Designs DR strategies: RTO/RPO targets, backup verification, failover procedures, communication trees, recovery runbooks.
8. **Backup Verification**
   Tests backup integrity: restore drills, checksum validation, data completeness checks, recovery time measurement.
9. **Vulnerability Assessment**
   Scans for security vulnerabilities: dependency audits, configuration review, access control verification, penetration test coordination.
10. **Rate Limit & Throttle Management**
    Implements emergency rate limiting: API throttling, request queuing, graceful degradation, user communication during capacity events.
11. **Data Corruption Recovery**
    Detects and recovers from data corruption: integrity checks, point-in-time recovery, affected record isolation, consistency verification.
12. **Service Dependency Mapping**
    Maps all service dependencies: upstream/downstream impacts, single points of failure, cascade failure scenarios, circuit breaker placement.
13. **Runbook Creation**
    Documents operational procedures: step-by-step instructions, decision trees, escalation paths, rollback commands for common incidents.
14. **Alert Configuration**
    Sets up monitoring alerts: threshold definitions, escalation policies, on-call rotation, alert fatigue prevention, meaningful notification design.
15. **Capacity Planning**
    Forecasts resource needs: traffic patterns, growth projections, scaling triggers, cost optimization, infrastructure right-sizing.
16. **Chaos Engineering**
    Designs controlled failure experiments: network partitions, service kills, latency injection, resource exhaustion to test resilience.

17. **Incident Timeline Reconstruction**
    Builds precise incident timelines from logs, metrics, and human actions: what happened, when, what was affected, what was done.
18. **Crisis Communication Strategy**
    Manages external communication during crises: press statements, social media responses, customer support scripts, executive briefings.
19. **Reputation Management**
    Monitors and responds to brand threats: negative reviews, social media crises, competitive FUD, media coverage.
20. **Dependency Lockdown**
    Emergency dependency pinning: locks versions, blocks updates, audits transitive dependencies during security incidents.
21. **Access Revocation**
    Emergency access management: revoke compromised credentials, disable accounts, rotate API keys, update access control lists.
22. **Forensic Evidence Preservation**
    Preserves evidence for analysis: log snapshots, memory dumps, network captures, file system images before remediation.
23. **Cascading Failure Prevention**
    Implements circuit breakers, bulkheads, timeouts, and fallbacks to prevent single failures from cascading through the system.
24. **Incident Severity Classification**
    Standardized severity levels: SEV1 (customer-facing outage) to SEV4 (minor degradation) with response time SLAs.
25. **Recovery Verification**
    Post-recovery validation: smoke tests, canary deployments, metric monitoring, user experience verification before all-clear.
26. **Incident Pattern Detection**
    Identifies recurring incident patterns: common root causes, time-of-day correlations, deployment-triggered failures, seasonal spikes.
27. **Emergency Playbook Library**
    Maintains ready-to-execute playbooks for common scenarios: DDoS, data breach, service outage, payment failure, DNS issues.
28. **Stakeholder Escalation Matrix**
    Defines who to notify at each severity level: engineering leads, executives, legal, PR, customers, partners with timing guidelines.
29. **Business Continuity Planning**
    Ensures critical operations continue during major incidents: manual fallbacks, alternative systems, priority processing.
30. **Lessons Learned Integration**
    Converts post-mortem findings into permanent improvements: updated playbooks, new monitoring, architectural changes, training updates.

TIER 2: MOUSSAIF1B SPECIALISTS | MOUSSAIF1B Specialist

6. Data Agent

Description

The MOUSSAIF1B Data Specialist handles data analysis, visualization, statistical modeling, ML pipelines, and database operations. All statistical claims must include confidence intervals. Data assumptions must be verified, not assumed. Visualizations must be honest -- no misleading axes.

Architecture

Workflow: Frame Data Question -> Inspect Actual Data -> Apply Statistical Methods -> Visualize -> Document
Quality Gates: confidence intervals on claims, verified assumptions, honest visualizations, ML evaluation metrics
Knowledge Bases: kb-data-science, agentic-kb-os-world

Main 30 Skills

1. **Exploratory Data Analysis (EDA)**
   Systematic data exploration: distributions, correlations, outliers, missing values, data types, sample sizes with summary statistics.
2. **Statistical Modeling**
   Applies appropriate statistical models: regression, classification, clustering, time series with proper assumptions checking and validation.
3. **Hypothesis Testing**
   Conducts rigorous hypothesis tests: t-tests, chi-square, ANOVA, non-parametric alternatives with effect sizes and confidence intervals.
4. **ML Pipeline Design**
   End-to-end ML: data preprocessing, feature engineering, model selection, hyperparameter tuning, cross-validation, deployment.
5. **Database Schema Design**
   Models relational and NoSQL schemas: normalization, indexing strategies, query optimization, migration planning.
6. **Data Visualization**
   Creates honest, clear visualizations: appropriate chart types, proper axes, color accessibility, interactive dashboards.
7. **ETL Pipeline Construction**
   Builds data pipelines: extraction from multiple sources, transformation rules, loading into target systems with error handling.
8. **Data Governance**
   Enforces data policies: quality rules, access controls, retention policies, lineage tracking, privacy compliance.
9. **Feature Engineering**
   Creates predictive features: encoding, scaling, binning, interaction terms, time-based features, domain-specific transformations.
10. **Model Evaluation**
    Evaluates models rigorously: accuracy, precision, recall, F1, AUC-ROC, calibration plots, confusion matrices, cross-validation.
11. **Time Series Analysis**
    Analyzes temporal data: trend decomposition, seasonality, autocorrelation, forecasting (ARIMA, Prophet, LSTM).
12. **A/B Test Design**
    Designs experiments: sample size calculation, randomization, metric selection, statistical significance testing, practical significance.
13. **Data Cleaning**
    Handles dirty data: missing value imputation, outlier treatment, deduplication, format standardization, consistency checks.
14. **Big Data Processing**
    Processes large datasets: Spark, Dask, chunked processing, memory optimization, parallel computation strategies.
15. **Database Query Optimization**
    Optimizes SQL queries: execution plan analysis, index recommendations, query rewriting, N+1 detection, caching strategies.
16. **Data Storytelling**
    Communicates data insights through narratives: key findings, business implications, actionable recommendations with supporting visuals.

17. **Predictive Analytics**
    Builds predictive systems: demand forecasting, churn prediction, risk scoring, recommendation engines with monitoring.
18. **Geospatial Analysis**
    Analyzes location data: spatial clustering, heatmaps, routing, geographic segmentation, proximity analysis.
19. **Text Analytics**
    Processes text data: sentiment analysis, topic modeling, named entity recognition, text classification, keyword extraction.
20. **Anomaly Detection**
    Identifies unusual patterns: statistical methods (z-score, IQR), ML-based (Isolation Forest, Autoencoders), domain-specific rules.
21. **Data Lake Architecture**
    Designs data lakes: schema-on-read, data zones (raw/curated/consumption), metadata management, data quality gates.
22. **Causal Inference**
    Estimates causal effects: randomized experiments, difference-in-differences, regression discontinuity, instrumental variables.
23. **Dimensionality Reduction**
    Reduces feature space: PCA, t-SNE, UMAP, feature selection methods while preserving interpretability.
24. **Data Pipeline Monitoring**
    Monitors data pipelines: freshness checks, volume alerts, schema drift detection, quality metric tracking.
25. **Synthetic Data Generation**
    Creates synthetic datasets for testing: statistical properties preservation, privacy-safe generation, edge case coverage.
26. **Data Versioning**
    Tracks data versions: DVC integration, dataset snapshots, reproducibility guarantees, rollback to previous data states.
27. **Dashboard Design**
    Builds executive dashboards: KPI selection, visual hierarchy, drill-down capability, real-time updates, mobile responsiveness.
28. **Data Migration**
    Safely migrates data between systems: schema mapping, transformation rules, validation checks, rollback procedures.
29. **Privacy-Preserving Analytics**
    Implements privacy techniques: differential privacy, k-anonymity, data masking, pseudonymization for GDPR compliance.
30. **Data Quality Framework**
    Establishes quality dimensions: completeness, accuracy, consistency, timeliness, validity, uniqueness with automated checks.

TIER 2: MOUSSAIF1B SPECIALISTS | MOUSSAIF1B Specialist

7. DevOps Agent

Description

The MOUSSAIF1B DevOps Specialist handles CI/CD pipelines, container orchestration, infrastructure as code, monitoring, and deployment automation. Includes the complete 50-skill Coolify VPS deployment suite. All deployments must have rollback plans. Zero-downtime for production.

Architecture

Workflow: Assess Infrastructure -> Design Pipeline -> Implement with Rollback -> Verify with Smoke Tests -> Document Handoff
Quality Gates: rollback plan required, healthchecks before success, no credentials in code, zero-downtime production
Knowledge Bases: deployable-agi-knowledge-os, kb-devops-cicd

Main 30 Skills

1. **Docker Container Management**
   Authors optimized Dockerfiles, manages container lifecycle, implements multi-stage builds, layer caching, and security scanning.
2. **Kubernetes Orchestration**
   Deploys and manages K8s workloads: deployments, services, ingress, configmaps, secrets, RBAC, resource limits.
3. **CI/CD Pipeline Design**
   Builds GitHub Actions, GitLab CI pipelines: build, test, lint, security scan, deploy stages with parallel execution and caching.
4. **VPS Deployment with Coolify**
   Complete 50-skill Coolify suite: VPS intake, baseline, SSH hardening, DNS, GitHub auth, Docker compose, TLS, monitoring.
5. **Infrastructure as Code**
   Manages infrastructure with Terraform, Ansible, Pulumi: reproducible environments, state management, drift detection.
6. **Monitoring & Alerting**
   Sets up observability: Prometheus metrics, Grafana dashboards, alert rules, on-call integration, SLO tracking.
7. **Zero-Downtime Deployments**
   Implements blue-green, canary, rolling update strategies with health checks, automatic rollback, and traffic shifting.
8. **Backup & Disaster Recovery**
   Designs backup strategies: automated schedules, retention policies, restore testing, off-site replication, DR runbooks.
9. **DNS & Domain Management**
   Configures DNS records, subdomains, CDN routing, domain mapping, SSL certificate automation with Let's Encrypt.
10. **Reverse Proxy Configuration**
    Manages Traefik/Nginx: routing rules, load balancing, TLS termination, rate limiting, WebSocket support.
11. **Database Provisioning**
    Provisions PostgreSQL, MySQL, Redis: user management, replication, backup schedules, connection pooling, migration support.
12. **Secret Management**
    Manages secrets securely: environment variables, vault integration, credential rotation, access auditing, no plaintext storage.
13. **Log Aggregation**
    Centralizes logs: structured logging, log shipping, search/alerting, retention policies, correlation IDs.
14. **Container Registry Management**
    Manages Docker registries: image scanning, vulnerability reports, retention policies, access control, multi-arch builds.
15. **Service Mesh Configuration**
    Implements service mesh: mTLS, traffic policies, observability, circuit breaking, retry logic, canary routing.
16. **Cost Optimization**
    Right-sizes infrastructure: reserved instances, spot/preemptible usage, auto-scaling policies, resource utilization analysis.

17. **Security Hardening**
    Hardens systems: firewall rules, fail2ban, SSH key-only auth, minimal attack surface, regular patching.
18. **Load Testing**
    Runs load tests: concurrent user simulation, throughput measurement, bottleneck identification, capacity planning.
19. **Staging Environment Management**
    Manages staging: production parity, data seeding, feature branch previews, smoke test automation.
20. **Database Migration Execution**
    Runs migrations safely: pre-migration backups, migration ordering, rollback scripts, zero-downtime strategies.
21. **Health Check Implementation**
    Implements liveness, readiness, and startup probes with proper timeout, threshold, and restart policies.
22. **Cron Job Management**
    Schedules recurring tasks: job locking, overlap prevention, failure alerting, execution logging, timeout handling.
23. **WebSocket Deployment**
    Deploys realtime features: sticky sessions, connection draining, horizontal scaling, message broker integration.
24. **Volume & Storage Management**
    Manages persistent storage: volume provisioning, backup schedules, storage classes, data migration between volumes.
25. **Email & SMTP Configuration**
    Configures transactional email: SMTP relay setup, DKIM/SPF records, deliverability monitoring, bounce handling.
26. **Payment Integration Deployment**
    Deploys payment systems: webhook endpoints, idempotency, secret rotation, PCI compliance, testing with sandbox.
27. **Cache & CDN Configuration**
    Implements caching: Redis cache layers, CDN asset delivery, browser cache headers, cache invalidation strategies.
28. **Firewall & Network Security**
    Configures firewalls: port whitelisting, network segmentation, VPN access, intrusion detection, DDoS protection.
29. **Deployment Handoff Documentation**
    Creates operator handoff docs: architecture diagrams, runbooks, credential locations, escalation procedures, maintenance schedules.
30. **Full-Stack Launch Gate**
    Final production readiness: UI audit, API verification, database integrity, DNS/TLS validation, backup confirmation, rollback test.

TIER 2: MOUSSAIF1B SPECIALISTS | MOUSSAIF1B Specialist

8. Evolution Agent

Description

The MOUSSAIF1B Evolution Specialist handles skill evolution, regression testing, self-improvement loops, and capability expansion. No evolution without regression proof. Version stamps mandatory for all changes. Daily self-learning feeds into evolution proposals.

Architecture

Evolution Cycle: Propose -> Regression Test -> Compare Metrics -> Accept/Reject -> Version Stamp

Quality Gates: regression proof required, version stamps mandatory, skill overlap detection, daily learning feeds

Knowledge Bases: hyper-kb, moussaif-kb, agentic-kb-os-world

Main 30 Skills

1. **Skill Creation from Patterns**
   Identifies successful task patterns (5+ tool calls) and converts them into reusable SKILL.md files with triggers, workflows, and quality gates.
2. **Regression-Gated Evolution**
   Every change must pass: propose -> test against baseline -> compare before/after metrics -> accept only if no regression + measurable improvement.
3. **Self-Improvement Loop Analysis**
   Analyzes past performance: success rates, common failures, token efficiency, time-to-completion to identify improvement opportunities.
4. **Skill Curation & Deduplication**
   Detects overlapping, stale, or obsolete skills. Consolidates redundancy, archives unused capabilities, prevents skill sprawl.
5. **Prompt Optimization**
   Iteratively improves prompts based on quality feedback: A/B testing, metric tracking, failure case analysis, instruction clarity.
6. **Version Stamping & Changelog**
   Tracks every change with semantic versioning, integrity hashes, changelog entries, and diff records for full audit trail.
7. **Daily Self-Learning Integration**
   Converts daily session outcomes into learning candidates: successful patterns, failed approaches, environment discoveries.
8. **Evolution Candidate Propagation**
   Promotes proven patterns from temporary solutions to permanent capabilities through structured evolution proposals.
9. **Skill Dependency Mapping**
   Maps inter-skill dependencies: which skills reference others, shared knowledge bases, tool requirements, execution order.
10. **Quality Feedback Loop**
    Collects quality signals from every task: output ratings, user corrections, verification results to inform evolution decisions.
11. **A/B Comparison Framework**
    Runs controlled comparisons between skill versions: same inputs, different implementations, measured outputs.
12. **Capability Gap Detection**
    Identifies missing capabilities: tasks that fail, areas where no skill exists, user requests that can't be fulfilled.
13. **Skill Template Standardization**
    Ensures all skills follow consistent structure: YAML frontmatter, description, workflow, quality gates, related skills.
14. **Evolution Budget Management**
    Limits evolution rate: max changes per session, cooling-off periods, priority-based evolution scheduling.
15. **Cross-Agent Skill Transfer**
    Adapts skills for different agents: Hermes -> Codex, Codex -> Claude, with environment-specific adjustments.
16. **Skill Performance Metrics**
    Tracks per-skill metrics: usage count, success rate, average execution time, user satisfaction, cost efficiency.

17. **Deprecated Skill Archival**
    Safely archives unused skills: backup original, remove from active catalog, document reason for deprecation.
18. **Skill Merge Strategy**
    Merges complementary skills: identify overlap, combine unique capabilities, resolve conflicts, update all references.
19. **Evolution Court Verdict**
    Formal review process for major changes: multi-criteria evaluation, evidence requirement, verdict documentation.
20. **Skill Testing Framework**
    Tests skills before deployment: dry-run scenarios, edge case coverage, tool availability verification, KB accessibility.
21. **Prompt Version Control**
    Tracks prompt changes: git-like versioning, diff visualization, blame attribution, rollback capability.
22. **Skill Metadata Enrichment**
    Adds metadata to skills: usage statistics, related skills, prerequisites, estimated execution time, token cost.
23. **Evolution Report Generation**
    Generates periodic evolution reports: changes made, metrics improvements, pending proposals, regression results.
24. **Skill Portability Testing**
    Verifies skills work across environments: different OS, different agents, different tool availability, different KB sizes.
25. **Self-Critique Mechanism**
    Agent evaluates own outputs: identifies weaknesses, suggests improvements, flags uncertainty areas for human review.
26. **Learning Rate Adaptation**
    Adjusts evolution speed based on stability: rapid in early stages, conservative as system matures, emergency for critical fixes.
27. **Skill Composition Patterns**
    Combines multiple skills into complex workflows: sequential, parallel, conditional, with result aggregation.
28. **Evolution Audit Trail**
    Complete history of all evolution decisions: what changed, why, evidence, verdict, impact measurement, rollback status.
29. **Capability Boundary Documentation**
    Documents what each skill cannot do: known limitations, failure modes, edge cases, prerequisites for success.
30. **Continuous Improvement Dashboard**
    Tracks system-wide evolution metrics: total skills, active skills, average quality, improvement velocity, regression rate.

TIER 2: MOUSSAIF1B SPECIALISTS | MOUSSAIF1B Specialist

9. Memory Agent

Description

The MOUSSAIF1B Memory Specialist handles knowledge management, session recall, memory consolidation, KB curation, and Obsidian vault operations. No duplicate entries. All memories must have evidence grades. Stale entries (>1 week) must be pruned or refreshed.

Architecture

Workflow: Identify Memory Need -> Check Existing -> Consolidate with Evidence -> Update KB -> Verify Symlinks
Quality Gates: no duplicates, evidence grades required, stale entry pruning, symlink integrity preservation
Knowledge Bases: moussaif-kb, hyper-kb, agentic-kb-os-world

Main 30 Skills

1. **Memory Consolidation**
   Converts session traces into durable lessons: extract key facts, attach evidence grades, format for long-term storage.
2. **Knowledge Base Curation**
   Maintains KB quality: remove duplicates, update outdated entries, reorganize structure, verify source links.
3. **Session Recall (FTS5)**
   Full-text search across past sessions to find similar problems, solutions, and context for current tasks.
4. **Obsidian Vault Engineering**
   Manages Obsidian vaults: MOC generation, skill indexing, cross-platform sync, automated backlinks, graph analysis.
5. **Skill Creation from Patterns**
   Identifies reusable patterns in successful tasks and converts them into permanent skill documents.
6. **Symlink Management**
   Maintains zero-duplication architecture: creates and verifies symlinks across Hermes, Codex, Claude environments.
7. **Memory Entry Lifecycle**
   Manages entry lifecycle: creation -> active use -> staleness detection -> refresh or pruning -> archival.
8. **Evidence Grade Attachment**
   Labels every memory entry with confidence: observed, verified_external, supported, inferred, hypothesized, speculative, unknown.
9. **Memory Deduplication**
   Detects and removes duplicate memory entries: semantic similarity matching, source conflict resolution, canonical selection.
10. **Memory Compression**
    Compresses memory entries: remove redundancy, consolidate related facts, use references instead of repetition.
11. **KB Chapter Structure**
    Organizes knowledge bases with consistent chapter structure: table of contents, cross-references, source anchors.
12. **Memory Search Optimization**
    Optimizes memory retrieval: index management, relevance ranking, recency weighting, context-aware filtering.
13. **Cross-Environment Memory Sync**
    Synchronizes memory across Hermes, Codex, Claude, and Obsidian with conflict resolution and version tracking.
14. **Memory Export & Portability**
    Exports memory in portable formats: JSON, Markdown, CSV for migration between systems or backup.
15. **Memory Access Control**
    Implements memory permissions: read/write scopes per agent, sensitive data isolation, audit logging.
16. **Memory Freshness Monitoring**
    Tracks entry age and relevance: flags stale entries, suggests refresh actions, prevents outdated information use.

17. **KB Integrity Verification**
    Verifies KB integrity: checksum validation, link checking, reference resolution, orphan detection.
18. **Memory Analytics**
    Analyzes memory usage: most accessed entries, search patterns, coverage gaps, quality distribution.
19. **Memory Merge Conflict Resolution**
    Resolves conflicts when multiple agents update same memory: version comparison, manual review flags, merge strategies.
20. **Session Archival**
    Archives completed sessions: summary extraction, key outcome recording, skill candidate identification, cleanup.
21. **Memory-Driven Reasoning**
    Integrates past learnings into current reasoning: analogy retrieval, precedent lookup, preference application.
22. **KB Source Trace Discipline**
    Maintains source chains: original source -> extraction -> transformation -> output with evidence links.
23. **Memory Budget Management**
    Manages memory size: character limits per entry, total memory budget, priority-based retention.
24. **Memory Update Loops**
    Automated memory maintenance: periodic freshness checks, duplicate detection, quality scoring, cleanup.
25. **Knowledge Graph Construction**
    Builds knowledge graphs: entity relationships, concept hierarchies, cross-domain links, inference paths.
26. **Memory-Backed Decision Making**
    Uses historical memory to inform decisions: past outcomes, user preferences, learned patterns, risk assessments.
27. **Memory Backup & Recovery**
    Backs up memory to OneDrive with checksum verification: incremental, deduplicated, versioned with restore testing.
28. **Memory QA Pipeline**
    Validates memory entries: factual accuracy, source verification, completeness check, conflict detection.
29. **Federated Memory Architecture**
    Distributes memory across specialized stores: skills, KBs, sessions, preferences with unified search.
30. **Memory Evolution Tracking**
    Tracks how memory evolves: entry creation/modification history, knowledge growth metrics, quality trend analysis.

TIER 2: MOUSSAIF1B SPECIALISTS | MOUSSAIF1B Specialist

10. QA Agent

Description

The MOUSSAIF1B QA Specialist enforces the 5 Destruction Courts and validates all outputs before delivery. Security court is 100% -- no exceptions. All verdicts must include evidence. No fabricated test passes.

Architecture

5 Destruction Courts: Quality Gate (95%), Cost Efficiency (80%), Security Audit (100%), Anti-Generic (90%), Delivery Ready (95%)
Workflow: Receive Artifact -> Run All 5 Courts -> Grade Pass/Fail -> Report Verdict -> Remediation if FAIL

Main 30 Skills

1. **Quality Gate Court (95%)**
   Measures overall quality: correctness, completeness, consistency, adherence to specifications. Blocks below 95% threshold.
2. **Cost Efficiency Court (80%)**
   Evaluates routing decisions: was the cheapest sufficient model used? Were free tiers exhausted before paid? Token waste detection.
3. **Security Audit Court (100%)**
   Zero-tolerance security check: vulnerability scanning, credential exposure, injection risks, access control violations. 100% pass rate required.
4. **Anti-Generic Court (90%)**
   Rejects template-like, generic, or AI-typical output. Checks for originality, specificity, brand alignment. 90% threshold.
5. **Delivery Ready Court (95%)**
   Final delivery validation: format compliance, metadata completeness, accessibility, platform compatibility. 95% threshold.
6. **Code Review (Security)**
   Security-focused code review: OWASP Top 10, dependency vulnerabilities, secrets in code, injection prevention, authentication flaws.
7. **Code Review (Style)**
   Style consistency review: naming conventions, code organization, documentation quality, DRY principle, SOLID adherence.
8. **Code Review (Correctness)**
   Logic verification: edge case handling, error paths, boundary conditions, race conditions, resource management.
9. **Content QA (Originality)**
   Plagiarism and originality checking: similarity detection, source verification, proper attribution, unique value proposition.
10. **Content QA (SEO)**
    SEO completeness: title tags, meta descriptions, header hierarchy, keyword usage, internal linking, schema markup.
11. **Content QA (Brand Voice)**
    Brand consistency: tone, vocabulary, messaging alignment, target audience appropriateness, cultural sensitivity.
12. **Deployment Validation**
    Pre-deployment checks: smoke tests, health endpoints, configuration verification, dependency availability.
13. **Regression Testing**
    Compares current output against baseline: performance metrics, quality scores, behavioral consistency, edge case coverage.
14. **Evidence Verification**
    Validates all claimed evidence: source existence, URL accessibility, data accuracy, citation completeness.
15. **Test Result Authentication**
    Verifies test results are genuine: output consistency, execution logs, environment matching, no fabricated passes.
16. **Performance Benchmarking**
    Measures performance metrics: response time, throughput, resource usage, cost per task against established baselines.
17. **Accessibility Audit**
    Checks accessibility compliance: WCAG standards, screen reader compatibility, keyboard navigation, color contrast.

18. **Cross-Browser/Platform Testing**
    Verifies functionality across environments: browser compatibility, OS differences, device responsiveness, API versioning.
19. **Load & Stress Testing**
    Tests under pressure: concurrent users, high volume, resource exhaustion, recovery behavior, graceful degradation.
20. **Data Integrity Verification**
    Validates data consistency: checksum verification, referential integrity, format compliance, encoding correctness.
21. **API Contract Testing**
    Verifies API contracts: request/response schemas, error codes, versioning compliance, backward compatibility.
22. **Security Dependency Audit**
    Scans dependencies for known vulnerabilities: CVE checking, license compliance, outdated package detection.
23. **Configuration Audit**
    Reviews configuration for security and correctness: default passwords, debug modes, exposed endpoints, missing encryption.
24. **Documentation Quality Review**
    Evaluates documentation: completeness, accuracy, examples, edge cases covered, up-to-date with code changes.
25. **User Acceptance Criteria Verification**
    Validates all acceptance criteria are met: feature completeness, edge cases, error handling, user experience.
26. **Verdict Documentation**
    Documents all court verdicts: criteria evaluated, evidence cited, score achieved, specific findings, remediation steps.
27. **Remediation Guidance**
    Provides specific fix instructions for failed checks: exact location, what's wrong, how to fix, verification method.
28. **Quality Trend Analysis**
    Tracks quality metrics over time: improvement trends, recurring issues, area-specific scores, regression detection.
29. **QA Pipeline Configuration**
    Configures automated QA pipelines: check ordering, parallel execution, failure handling, reporting integration.
30. **Quality Dashboard**
    Maintains quality dashboard: current scores, historical trends, open issues, resolution status, team accountability.

TIER 2: MOUSSAIF1B SPECIALISTS | MOUSSAIF1B Specialist

11. Research Agent

Description

The MOUSSAIF1B Research Specialist handles deep research, synthesis, competitive analysis, trend detection, and source verification. Every factual claim must have an evidence grade. Sources must be cited. No speculative claims presented as fact.

Architecture

Workflow: Frame Research Question -> Triangulate Sources -> Grade Claims -> Synthesize -> Flag Gaps
Quality Gates: evidence grades on all claims, URL citations, no speculation as fact, confidence assessment
Knowledge Bases: moussaif-kb, hyper-kb, afripa-kb

Main 30 Skills

1. **Multi-Source Research**
   Gathers information from web, KBs, academic sources, APIs with source triangulation and cross-validation.
2. **Evidence Grading for Claims**
   Labels every research finding: observed > verified_external > supported > inferred > hypothesized > speculative > unknown.
3. **Competitive Intelligence**
   Analyzes competitors: product features, pricing, positioning, market share, customer reviews, technology stack.
4. **Market Trend Analysis**
   Identifies market trends: growth patterns, emerging technologies, shifting consumer behavior, regulatory changes.
5. **Academic Paper Review**
   Reviews research papers: methodology assessment, result validity, practical implications, reproducibility check.
6. **Source Verification**
   Verifies source credibility: author credentials, publication reputation, citation count, recency, bias assessment.
7. **Research Brief Production**
   Produces structured research briefs: executive summary, key findings, evidence, implications, recommendations, confidence.
8. **Web Evidence Collection**
   Collects current web evidence: page content, screenshots, API responses, metadata with timestamp and URL recording.
9. **Synthesis & Analysis**
   Synthesizes findings from multiple sources: pattern identification, contradiction resolution, gap identification.
10. **Literature Review**
    Conducts systematic literature reviews: search strategy, inclusion/exclusion criteria, quality assessment, synthesis.
11. **Data Source Discovery**
    Finds relevant data sources: public datasets, APIs, databases, surveys, reports for research questions.
12. **Research Methodology Design**
    Designs research approaches: qualitative vs quantitative, sampling strategies, data collection methods, analysis plans.
13. **Fact-Checking Pipeline**
    Multi-stage fact verification: claim extraction, source lookup, cross-reference, confidence scoring, verdict.
14. **Industry Report Analysis**
    Analyzes industry reports: key statistics, methodology critique, trend extraction, competitive landscape mapping.
15. **Patent & IP Research**
    Searches patent databases: prior art analysis, freedom-to-operate assessment, technology landscape mapping.
16. **Regulatory Research**
    Investigates regulations: compliance requirements, recent changes, enforcement patterns, jurisdictional differences.
17. **Technology Assessment**
    Evaluates technologies: maturity level, adoption curve, competitive alternatives, integration complexity, cost-benefit.

18. **User Research Synthesis**
    Synthesizes user research: pain point identification, persona refinement, journey mapping, feature prioritization.
19. **Geopolitical Risk Analysis**
    Assesses geopolitical factors: trade policies, political stability, regulatory environment, cultural considerations.
20. **Financial Research**
    Analyzes financial data: market valuations, funding rounds, revenue models, unit economics, growth metrics.
21. **Research Automation**
    Automates research workflows: scheduled monitoring, alert triggers, data collection pipelines, report generation.
22. **Citation Management**
    Manages research citations: proper formatting, source tracking, bibliography generation, link verification.
23. **Research Gap Analysis**
    Identifies knowledge gaps: unanswered questions, missing data, unstudied areas, methodological limitations.
24. **Comparative Analysis**
    Structured comparison: feature matrices, pros/cons, scoring rubrics, weighted criteria evaluation.
25. **Expert Opinion Synthesis**
    Aggregates expert views: interview synthesis, survey analysis, Delphi method, consensus identification.
26. **Research Quality Scoring**
    Scores research quality: methodology rigor, sample size, recency, source authority, reproducibility.
27. **Predictive Trend Modeling**
    Models future trends: extrapolation, scenario planning, leading indicator identification, probability assessment.
28. **Cross-Domain Research**
    Connects insights across fields: interdisciplinary synthesis, analogy application, innovation opportunity identification.
29. **Research Knowledge Graph**
    Builds research knowledge graphs: entity relationships, concept hierarchies, evidence chains, inference paths.
30. **Continuous Research Monitoring**
    Monitors ongoing developments: RSS/Atom feeds, alert systems, periodic scanning, change detection.

TIER 2: MOUSSAIF1B SPECIALISTS | MOUSSAIF1B Specialist

12. Router Agent

Description

The MOUSSAIF1B Router Specialist handles model selection, provider orchestration, rate-limit management, and cost optimization. Implements 8-tier cascade routing from OpenCode Go (top reasoning) down to Ollama local (always on). Rate state checked before every route. 429 errors trigger immediate tier degradation.

Architecture

8-Tier Cascade: OpenCode Go (gpt-5.5-pro) -> OpenCode Go mid -> OpenCode Go flash -> OpenCode Zen free -> NVIDIA NIM -> Google -> OpenRouter free -> Ollama local
Quality Gates: rate state check before every route, 429 triggers degradation, all routing decisions logged

Main 30 Skills

1. **8-Tier Cascade Routing**
   Routes through 8 provider tiers: OpenCode Go -> OpenCode Zen -> NVIDIA NIM -> Google -> OpenRouter -> Ollama with automatic failover.
2. **Rate-Limit Tracking**
   Sliding-window rate limit tracking per provider: current usage, reset times, burst limits, concurrent request counts.
3. **Task Classification**
   Classifies tasks by complexity: critical (needs top model), complex (mid-tier), normal (free tier), simple (local flash).
4. **Provider Health Monitoring**
   Monitors provider availability: latency, error rates, uptime, model availability, feature support.
5. **Cost-Quality Arbitration**
   Balances cost vs quality: free tier for routine, paid escalation when quality bar demands it, transparent reporting.
6. **Model Performance Benchmarking**
   Tracks model performance: accuracy, speed, cost, reliability across tasks to inform routing decisions.
7. **429 Error Handling**
   Immediate tier degradation on rate limits: retry with backoff, failover to next tier, record the event.
8. **Provider Pool Rotation**
   Rotates across provider pools: NIM, ZenMux, OpenRouter, Google to respect rate limits and maximize throughput.
9. **Model Discovery**
   Discovers available models: polls /v1/models endpoints, caches results, builds routing maps, detects new models.
10. **Fallback Chain Management**
    Manages fallback chains: primary -> secondary -> tertiary with configurable preferences and cost thresholds.
11. **Token Budget Management**
    Tracks token usage: per-task budgets, session totals, cost estimation, budget alerts.
12. **Model Capability Matching**
    Matches task requirements to model capabilities: vision, code, reasoning, speed, context window, cost.
13. **Provider Credential Rotation**
    Manages multiple API keys: rotation to avoid rate limits, health checking, automatic key switching.
14. **Routing Decision Logging**
    Logs all routing decisions: model selected, reason, alternatives considered, cost, latency, quality outcome.
15. **Free Tier Exhaustion Strategy**
    Exhausts all free options before paid: Google free -> NVIDIA free -> OpenRouter free -> ZenMux free -> local models.
16. **Smart Model Routing**
    Context-aware routing: considers task type, required quality, available budget, user preferences, historical performance.

17. **Provider Configuration Management**
    Manages provider configs: API endpoints, model lists, rate limits, authentication, custom headers.
18. **Routing Formula Implementation**
    Implements 8-factor weighted scoring for skill/tool selection with configurable weights.
19. **Cost Tracking & Reporting**
    Tracks costs per provider, per task type, per time period with optimization recommendations.
20. **Latency Optimization**
    Minimizes routing latency: provider proximity, model loading time, caching, pre-warming connections.
21. **Multi-Key Rate Limit Pooling**
    Aggregates rate limits across multiple API keys for higher throughput from same provider.
22. **Routing A/B Testing**
    Tests routing strategies: compare quality, cost, latency between different routing approaches.
23. **Provider Failover Automation**
    Automated failover: detect failure -> select alternative -> retry -> log -> notify if all fail.
24. **Model Version Tracking**
    Tracks model versions: detect updates, handle deprecations, test new versions before routing traffic.
25. **Routing Policy Engine**
    Configurable routing policies: cost-first, quality-first, speed-first, balanced with per-task overrides.
26. **Provider API Compatibility**
    Handles API differences: request/response format normalization, parameter mapping, error code translation.
27. **Routing Metrics Dashboard**
    Displays routing metrics: provider usage, success rates, costs, latency, failover frequency.
28. **ZenMux Integration**
    Integrates with ZenMux free and paid pools: model selection, rate tracking, quality monitoring.
29. **NVIDIA NIM Integration**
    Integrates with NVIDIA NIM: free model access, rate limit management, model capability detection.
30. **Ollama Local Integration**
    Manages local Ollama models: health monitoring, model loading, resource allocation, always-on availability.

TIER 2: MOUSSAIF1B SPECIALISTS | MOUSSAIF1B Specialist

13. Strategy Agent

Description

The MOUSSAIF1B Strategy Specialist handles market analysis, positioning, go-to-market planning, competitive intelligence, and business architecture design. All market claims must be evidence-graded. No generic advice -- everything specific to the user's context.

Architecture

Workflow: Frame Strategic Question -> Triangulate from KBs -> Evidence-Grade Claims -> Deliver with Risk Assessment
Quality Gates: evidence-graded market claims, sourced competitive analysis, risk assessment, context-specific advice
Knowledge Bases: afripa-kb, moussaif-kb, hyper-kb

Main 30 Skills

1. **Market Mapping & ICP Definition**
   Maps target markets: TAM/SAM/SOM sizing, ideal customer profiles, buyer personas, market segmentation.
2. **Competitive Positioning Audits**
   Analyzes competitive landscape: feature comparison, pricing analysis, positioning gaps, differentiation opportunities.
3. **Category Design**
   Creates or identifies market categories: category definition, narrative ownership, competitive moats, market creation.
4. **Go-to-Market Strategy**
   Plans market entry: launch sequencing, channel strategy, messaging framework, initial target segments, success metrics.
5. **Value Proposition Development**
   Crafts compelling value props: customer jobs/pains/gains mapping, unique differentiation, proof points, messaging hierarchy.
6. **Europe-Africa Expansion Strategy**
   Plans cross-continental expansion: market selection, regulatory compliance, cultural adaptation, partnership strategies.
7. **DACH Market Entry Analysis**
   Germany/Austria/Switzerland market specifics: business culture, regulatory environment, language requirements, distribution channels.
8. **Business Model Architecture**
   Designs business models: revenue streams, cost structure, key resources, partnerships, value delivery mechanisms.
9. **Pricing Strategy**
   Develops pricing: cost-plus, value-based, competitive, tiered, freemium models with willingness-to-pay analysis.
10. **Channel Strategy**
    Selects distribution channels: direct, partner, online, hybrid with channel conflict management and optimization.
11. **Partnership Strategy**
    Identifies strategic partners: complementary capabilities, market access, co-marketing opportunities, alliance structures.
12. **Market Entry Timing**
    Analyzes market readiness: technology maturity, customer demand, competitive timing, regulatory window.
13. **Strategic Risk Assessment**
    Identifies strategic risks: market shifts, competitive moves, regulatory changes, technology disruption, execution risks.
14. **Growth Roadmap Design**
    Plans growth phases: foundation, expansion, scale with milestones, resource requirements, and success criteria.
15. **Customer Journey Mapping**
    Maps end-to-end customer journey: awareness, consideration, purchase, onboarding, retention, advocacy touchpoints.
16. **Revenue Model Design**
    Designs sustainable revenue: subscription, transaction, advertising, licensing, hybrid models with unit economics.

17. **Market Intelligence System**
    Builds intelligence gathering: competitor monitoring, trend tracking, customer feedback loops, signal detection.
18. **Strategic Decision Framework**
    Structured decision-making: criteria weighting, option generation, consequence analysis, reversibility assessment.
19. **Positioning Statement Development**
    Crafts positioning: target segment, frame of reference, point of difference, reason to believe, brand promise.
20. **Blue Ocean Strategy**
    Identifies uncontested market space: value innovation, strategy canvas, four actions framework, non-customer analysis.
21. **Strategic Narrative Design**
    Creates compelling narratives: origin story, vision narrative, category story, customer transformation story.
22. **Market Sizing Methodology**
    Estimates market size: top-down, bottom-up, value theory approaches with sensitivity analysis and assumptions.
23. **Strategic Roadmap Communication**
    Communicates strategy: executive summaries, visual roadmaps, milestone tracking, stakeholder alignment.
24. **Competitive Response Planning**
    Plans competitive responses: offensive moves, defensive strategies, preemptive actions, game theory applications.
25. **Innovation Strategy**
    Plans innovation pipeline: sustaining vs disruptive, build vs buy vs partner, R&D allocation, portfolio management.
26. **Internationalization Planning**
    Plans global expansion: market prioritization, localization requirements, legal entity setup, operational model.
27. **Strategic Alliance Design**
    Designs alliances: equity vs non-equity, governance structures, IP sharing, exit provisions, performance metrics.
28. **Scenario Planning**
    Develops multiple futures: best/worst/likely scenarios, trigger events, strategic options for each, early warning signals.
29. **Strategic KPI Framework**
    Defines strategic metrics: leading vs lagging indicators, measurement cadence, accountability, dashboard design.
30. **Board & Investor Communication**
    Prepares strategic communications: board decks, investor updates, strategic rationale documents, progress reports.

TIER 3: EXTERNAL AGENTS | External Agent

14. Hermes Agent

Description

Hermes Agent is the open-source AI agent framework by Nous Research that serves as the primary orchestration platform for the Moussaif AGI OS. It runs in terminals, desktop apps, messaging platforms, and IDEs. Self-improving through skills, persistent memory across sessions, multi-platform gateway (Telegram, Discord, Slack, WhatsApp, iMessage, Signal, Matrix, Teams, Email), provider-agnostic (20+ providers), profiles, plugins, MCP servers, and cron scheduling.

Architecture

Core: Agent framework with tool calling, skill system, memory backends, multi-platform gateway
Surfaces: CLI, Ink TUI, Electron desktop, Web dashboard, ACP server for IDEs
Providers: OpenRouter, Anthropic, OpenAI, Google, DeepSeek, xAI, local models, 20+ others
Skills: Self-improving -- saves reusable procedures as SKILL.md files that load into future sessions

Main 30 Skills

1. **Multi-Platform Gateway**
   Runs on Telegram, Discord, Slack, WhatsApp, iMessage, Signal, Matrix, Teams, Email with full tool access, not just chat.
2. **Self-Improving Skill System**
   Learns from experience by saving reusable procedures as SKILL.md files that accumulate and improve over time.
3. **Persistent Memory**
   Remembers user preferences, environment details, lessons learned across sessions with pluggable backends.
4. **Provider Agnostic Routing**
   Swaps models/providers mid-workflow: OpenRouter, Anthropic, OpenAI, Google, DeepSeek, xAI, local models, 20+ others.
5. **Profile Isolation**
   Runs multiple independent instances with isolated configs, sessions, skills, and memory.
6. **Plugin Architecture**
   Extensible via plugins: custom tools, MCP servers, webhook triggers, full Python ecosystem integration.
7. **Cron Job Scheduling**
   Schedules recurring tasks: automated sync, monitoring, maintenance, content generation on time-based triggers.
8. **Terminal Command Execution**
   Executes shell commands with pty support, timeout handling, output capture, and working directory management.
9. **File System Operations**
   Read, write, edit, search, glob operations with proper encoding handling and path resolution.
10. **Git Integration**
    Full git workflow: branch, commit, diff, log, merge, rebase, conflict resolution, PR management via gh CLI.
11. **GitHub CLI Integration**
    Manages GitHub: issues, PRs, releases, actions, packages via gh CLI with proper authentication.
12. **Docker Management**
    Docker operations: build, run, compose, registry push/pull, container management, log inspection.
13. **Web Search & Fetch**
    Searches the web and fetches page content with format conversion, timeout handling, and content summarization.
14. **Image & File Analysis**
    Reads and analyzes images, PDFs, and other file types with content extraction and description.
15. **MCP Server Integration**
    Connects to Model Context Protocol servers for extended capabilities: databases, APIs, external services.

16. **Interactive TUI**
    Ink-based terminal UI with syntax highlighting, file tree, progress indicators, and interactive prompts.
17. **Native Desktop App**
    Electron-based desktop application with system tray, notifications, and OS integration.
18. **Web Dashboard**
    Browser-based dashboard for conversation management, settings, history, and multi-session handling.
19. **ACP Server for IDEs**
    Agent Communication Protocol server for VS Code, Zed, JetBrains with real-time collaboration.
20. **Credential Pool Rotation**
    Rotates across multiple API keys automatically to avoid rate limits and maximize throughput.
21. **Session Management**
    Manages conversation sessions: history, context window, compression, export, import, search.
22. **Agent Spawning**
    Spawns additional agent instances with isolated contexts for parallel task execution.
23. **Skill Discovery & Loading**
    Discovers and loads SKILL.md files from multiple locations: project, user, system directories.
24. **Memory Backend Selection**
    Chooses between memory backends: built-in, Honcho, Mem0, custom backends with different trade-offs.
25. **Auto-Update & Health**
    Self-updating with health checks, version management, and automatic dependency resolution.
26. **Streaming Responses**
    Streams responses in real-time with token-by-token display and progress tracking.
27. **Multi-Modal Support**
    Handles text, images, audio, video inputs with appropriate tool routing and processing.
28. **Error Recovery**
    Graceful error handling: retry logic, fallback strategies, user notification, partial result preservation.
29. **Configuration Management**
    Hierarchical config: system, user, project, session levels with proper override precedence.
30. **Audit Logging**
    Comprehensive logging: tool calls, file changes, API requests, errors, performance metrics for review.

TIER 3: EXTERNAL AGENTS | External Agent

15. Codex CLI

Description

OpenAI Codex CLI is OpenAI's autonomous coding agent that runs in the terminal. It handles features, PRs, refactoring, and batch issue fixing. Requires a git repository and OpenAI auth. Integrated into Hermes via terminal tools with pty support.

Architecture

Core: OpenAI's autonomous coding agent CLI
Auth: OPENAI_API_KEY or Codex OAuth from ~/.codex/auth.json
Requirement: Must run inside a git repository
Integration: Via Hermes terminal with pty=true for interactive sessions

Main 30 Skills

1. **One-Shot Code Generation**
   Generates code from natural language descriptions: features, functions, modules with proper structure and tests.
2. **PR Review & Comments**
   Reviews pull requests: diff analysis, inline comments, suggestions, approval/rejection with reasoning.
3. **Refactoring Engine**
   Restructures existing code: extract methods, rename, reorganize, simplify while preserving behavior.
4. **Batch Issue Fixing**
   Processes multiple issues in sequence: triage, fix, test, commit with consistent quality.
5. **Git Workflow Management**
   Manages git: branch creation, atomic commits, conflict resolution, merge strategy selection.
6. **Test Generation**
   Generates tests: unit, integration, edge cases with proper assertions and mocking.
7. **Code Explanation**
   Explains code: purpose, architecture, data flow, dependencies, potential issues.
8. **Bug Detection**
   Identifies bugs: logic errors, race conditions, memory leaks, security vulnerabilities, edge cases.
9. **Documentation Generation**
   Generates docs: README, API docs, code comments, architecture decision records.
10. **Dependency Analysis**
    Analyzes dependencies: version conflicts, security vulnerabilities, unused imports, alternatives.
11. **Performance Optimization**
    Optimizes performance: algorithmic improvements, caching, lazy loading, query optimization.
12. **Type System Enhancement**
    Improves type safety: add types, fix type errors, improve generic usage, enforce strict mode.
13. **Error Handling Improvement**
    Enhances error handling: add try/catch, improve error messages, add logging, implement recovery.
14. **Code Migration**
    Migrates code: framework upgrades, language version updates, API migrations, library replacements.
15. **API Design Assistance**
    Designs APIs: endpoint structure, request/response schemas, versioning, backward compatibility.
16. **Database Query Optimization**
    Optimizes queries: index recommendations, query rewriting, N+1 elimination, caching strategies.

17. **Security Hardening**
    Hardens code: input validation, SQL injection prevention, XSS protection, CSRF tokens, auth improvements.
18. **Configuration Management**
    Manages configs: environment variables, config files, secret management, feature flags.
19. **Monorepo Navigation**
    Navigates monorepos: package discovery, dependency graphs, affected package identification, cross-package changes.
20. **Code Style Enforcement**
    Enforces style: linting rules, formatting standards, naming conventions, documentation requirements.
21. **Architecture Analysis**
    Analyzes architecture: dependency injection, separation of concerns, module boundaries, coupling metrics.
22. **Prototype Generation**
    Rapid prototyping: quick implementations, proof-of-concepts, spike solutions, feasibility testing.
23. **Code Translation**
    Translates code between languages: Python ? JS, REST ? GraphQL, SQL dialects, framework syntax.
24. **Automated Fix Suggestions**
    Suggests fixes: from error messages, stack traces, test failures, linting warnings, type errors.
25. **Search & Replace Operations**
    Bulk code changes: regex-based replacements, AST transformations, pattern matching across files.
26. **Git History Analysis**
    Analyzes git history: blame, log filtering, change frequency, author contribution, hot spot detection.
27. **CI/CD Pipeline Repair**
    Repairs broken pipelines: error diagnosis, config fixes, dependency updates, test failure resolution.
28. **Package.json Management**
    Manages Node.js packages: dependency updates, script optimization, workspace configuration.
29. **Dockerfile Optimization**
    Optimizes Dockerfiles: layer caching, multi-stage builds, security scanning, size reduction.
30. **Code Review Automation**
    Automates review: style checking, complexity analysis, security scanning, test coverage verification.

TIER 3: EXTERNAL AGENTS | External Agent

16. Claude Code CLI

Description

Claude Code is Anthropic's autonomous coding agent CLI (v2.x). It can read files, write code, run shell commands, spawn subagents, and manage git workflows autonomously. Two modes: Print Mode (-p) for non-interactive tasks and Interactive PTY via tmux for multi-turn sessions.

Architecture

Core: Anthropic's autonomous coding agent with tool use and subagent spawning
Auth: ANTHROPIC_API_KEY or browser OAuth (Pro/Max), SSO for Enterprise
Modes: Print Mode (-p) non-interactive, Interactive PTY via tmux for multi-turn
Tools: Read, Write, Edit, Bash, Glob, Grep, WebSearch, delegate_task

Main 30 Skills

1. **Print Mode Execution**
   Non-interactive one-shot tasks: -p flag skips all dialogs, returns result, exits clean. Ideal for automation.
2. **Interactive PTY Sessions**
   Multi-turn coding sessions via tmux: persistent context, iterative development, progress checks, mid-course corrections.
3. **Subagent Spawning**
   Spawns subagents for parallel work: isolated contexts, delegated subtasks, result collection, conflict-free execution.
4. **File System Operations**
   Read, write, edit, glob, grep with proper encoding, path resolution, and large file handling.
5. **Shell Command Execution**
   Runs shell commands: build, test, deploy, git, npm, pip with timeout, working directory, and output capture.
6. **Git Workflow Automation**
   Automates git: branch, commit, push, PR creation, review addressing, conflict resolution, merge.
7. **Code Generation**
   Generates production code: multi-language support, framework awareness, style matching, test inclusion.
8. **Code Review & Refactoring**
   Reviews and refactors: inline suggestions, complexity reduction, pattern application, security improvements.
9. **Test-Driven Development**
   TDD workflow: write test -> implement -> verify -> refactor with proper test isolation and mocking.
10. **JSON Schema Output**
    Structured output: --json-schema flag for machine-readable responses, data extraction, API integration.
11. **Piped Input Processing**
    Processes piped input: cat file | claude -p analyze for data transformation, analysis, summarization.
12. **Multi-File Operations**
    Coordinates changes across multiple files: dependency-aware editing, import updates, cross-file refactoring.
13. **Debugging & Error Resolution**
    Systematic debugging: reproduce, isolate, fix, verify with proper logging and error context preservation.
14. **Dependency Management**
    Manages dependencies: add, remove, update, audit, lock with version constraint awareness.
15. **Database Operations**
    Database work: schema design, migration writing, query optimization, seed data creation.
16. **API Development**
    Builds APIs: endpoint design, middleware, authentication, validation, error handling, documentation.

17. **Frontend Development**
    Frontend work: component creation, state management, routing, styling, accessibility, performance.
18. **DevOps Integration**
    DevOps tasks: Docker, CI/CD, deployment, monitoring, infrastructure management.
19. **Security Auditing**
    Security review: vulnerability scanning, dependency audit, input validation, authentication verification.
20. **Performance Profiling**
    Performance analysis: bottleneck identification, optimization suggestions, benchmark comparison.
21. **Documentation Authoring**
    Writes documentation: README, API docs, architecture docs, inline comments, changelog entries.
22. **Code Migration**
    Framework/language migration: incremental conversion, parallel running, validation, rollback support.
23. **Package Publishing**
    Publishes packages: version bumping, changelog generation, build verification, registry upload.
24. **Monorepo Management**
    Monorepo operations: workspace management, affected builds, coordinated versioning, cross-package linking.
25. **AI-Assisted Debugging**
    AI debugging: error message analysis, stack trace interpretation, root cause suggestion, fix generation.
26. **Codebase Understanding**
    Codebase analysis: architecture mapping, dependency graphs, complexity metrics, entry point identification.
27. **Automated Testing**
    Test automation: test generation, coverage improvement, test maintenance, flaky test detection.
28. **Code Style Enforcement**
    Style consistency: linting setup, formatting rules, naming conventions, documentation standards.
29. **Prototype Development**
    Rapid prototyping: quick implementations, concept validation, feasibility testing, demo preparation.
30. **Auth & Session Management**
    Authentication: OAuth flow, API key management, session handling, token refresh, status checking.

TIER 3: EXTERNAL AGENTS | External Agent

17. OpenCode CLI

Description

OpenCode is a provider-agnostic, open-source AI coding agent with TUI and CLI. Integrated into Hermes as an autonomous coding worker. Supports one-shot tasks via 'opencode run' and interactive TUI sessions. Works with OpenRouter, Anthropic, OpenAI, Google, local models, and more.

Architecture

Core: Provider-agnostic autonomous coding agent with TUI and CLI
Auth: opencode auth login or provider env vars (OPENROUTER_API_KEY, etc.)
Modes: opencode run (one-shot), opencode tui (interactive), opencode serve (API)
Integration: Via Hermes terminal with pty=true for interactive sessions

Main 30 Skills

1. **One-Shot Task Execution**
   Runs bounded tasks via 'opencode run': single prompt, full tool access, result output, clean exit.
2. **Interactive TUI Sessions**
   Terminal UI with syntax highlighting, file browser, progress display, multi-turn conversation.
3. **Provider-Agnostic Routing**
   Works with any provider: OpenRouter, Anthropic, OpenAI, Google, local models without code changes.
4. **Code Generation**
   Generates code: multi-language, framework-aware, style-matching, with proper structure and testing.
5. **File System Operations**
   Read, write, edit, search files with proper encoding handling and path resolution.
6. **Shell Command Execution**
   Executes commands: build tools, test runners, package managers, system utilities with output capture.
7. **Git Integration**
   Git operations: status, diff, commit, branch, merge with proper conflict handling.
8. **Code Review**
   Reviews code: style, correctness, security, performance with inline suggestions.
9. **Refactoring**
   Restructures code: extract, rename, move, simplify with behavior preservation and test verification.
10. **Test Generation**
    Generates tests: unit, integration, e2e with proper assertions, mocking, and edge case coverage.
11. **Debugging**
    Debugs issues: error reproduction, root cause analysis, fix implementation, verification.
12. **Documentation**
    Generates docs: README, API docs, inline comments, architecture descriptions.
13. **Dependency Management**
    Manages packages: add, remove, update, audit with version constraint awareness.
14. **Multi-File Coordination**
    Coordinates changes: dependency-aware editing, import management, cross-file consistency.
15. **API Development**
    Builds APIs: routing, middleware, validation, error handling, authentication, documentation.
16. **Database Operations**
    Database work: schema design, migrations, queries, seeding, optimization.

17. **DevOps Tasks**
    DevOps operations: Docker, CI/CD, deployment, monitoring, infrastructure setup.
18. **Security Review**
    Security checks: vulnerability scanning, dependency audit, input validation, secret detection.
19. **Performance Analysis**
    Performance profiling: bottleneck identification, optimization suggestions, benchmark comparison.
20. **Codebase Exploration**
    Explores codebases: file discovery, dependency mapping, architecture understanding, entry point identification.
21. **Parallel Task Execution**
    Runs parallel tasks in isolated workdirs/worktrees for concurrent development.
22. **Progress Monitoring**
    Monitors task progress: status updates, intermediate results, completion estimation.
23. **Error Recovery**
    Handles errors: retry logic, fallback strategies, partial result preservation, user notification.
24. **Model Selection**
    Selects models per task: capability matching, cost optimization, quality requirements.
25. **Context Management**
    Manages context: window optimization, compression, relevant file selection, history pruning.
26. **Tool Integration**
    Integrates with external tools: databases, APIs, services, MCP servers for extended capabilities.
27. **Session Persistence**
    Persists sessions: resume conversations, maintain context across restarts, export history.
28. **Configuration Management**
    Manages configs: provider settings, model preferences, tool permissions, project-specific overrides.
29. **Streaming Output**
    Streams responses: token-by-token display, progress indicators, partial result delivery.
30. **Binary Resolution**
    Resolves OpenCode binary: detects version differences, pins explicit paths, handles PATH conflicts.

TIER 4: HYPER-* LAYERS | Hyper-AGI Layer

18. Hyper-Cognition Layer

Description

The Hyper-Cognition Layer provides advanced reasoning capabilities: multi-step reasoning (chain-of-thought, tree-of-thought, self-consistency), goal decomposition, skill selection, memory integration, analogy mapping, causal inference, contradiction detection, uncertainty estimation, abstraction extraction, and meta-reflection. This layer enables the agent to think about its own thinking.

Architecture

Layer: COGNITION (Skills 11-20 of 50 in Hyper-AGI OS)
Capabilities: Reasoning, Decomposition, Selection, Integration, Mapping, Inference, Detection, Estimation, Extraction, Reflection
Integration: Activated by moussaif1b kernel when deep reasoning is required

Main 30 Skills

1. **Multi-Step Reasoning Engine**
   Supports chain-of-thought (sequential), tree-of-thought (branching), and self-consistency (multiple attempts + voting) reasoning patterns.
2. **Goal Decomposition (DAG)**
   Breaks high-level goals into directed acyclic graph of atomic sub-tasks with dependencies, resource estimates, and critical path analysis.
3. **Skill Selection (MAXIIIMUM)**
   Selects optimal skills for each sub-task using relevance scoring, historical success, and the MAXIIIMUM skill utilization principle.
4. **Memory Integration**
   Integrates past session memory into reasoning: filters by relevance and recency, attaches evidence grades, prevents stale data use.
5. **Analogy Mapping**
   Maps solutions from past similar problems to current tasks using structural and semantic similarity scoring.
6. **Causal Inference**
   Infers cause-effect relationships using Pearl's structural causal models: confounders, mediators, colliders, counterfactuals.
7. **Contradiction Detection**
   Detects logical contradictions in reasoning chains: flag conflicts, suggest revisions, escalate to human review when needed.
8. **Uncertainty Estimation**
   Quantifies confidence using entropy, variance, calibration metrics. Flags uncertain outputs with explicit confidence notes.
9. **Abstraction Extraction**
   Extracts reusable patterns from specific solutions: generalizes to templates, identifies transferable principles.
10. **Meta-Reflection**
    Reflects on reasoning quality: identifies biases, suggests process improvements, evaluates strategy effectiveness.
11. **Chain-of-Thought Reasoning**
    Sequential step-by-step reasoning: explicit intermediate steps, assumption checking, conclusion verification.
12. **Tree-of-Thought Exploration**
    Branching reasoning: explores multiple solution paths, evaluates alternatives, selects best approach.
13. **Self-Consistency Voting**
    Multiple independent reasoning attempts with majority voting for robust conclusions.
14. **Reasoning Chain Validation**
    Validates reasoning chains: logical soundness, evidence support, conclusion follows from premises.
15. **Cognitive Load Management**
    Manages reasoning complexity: breaks problems when overloaded, prioritizes critical reasoning paths.
16. **Bias Detection & Correction**
    Detects cognitive biases: anchoring, confirmation, availability, representativeness with correction strategies.

17. **Reasoning Trace Documentation**
    Documents reasoning traces: decision points, alternatives considered, evidence cited, conclusion rationale.
18. **Heuristic Application**
    Applies reasoning heuristics: satisficing, elimination by aspects, recognition-primed decision making.
19. **Analogical Transfer**
    Transfers solutions across domains: identify structural parallels, adapt to new context, verify applicability.
20. **Reasoning Quality Metrics**
    Measures reasoning quality: logical consistency, evidence support, completeness, efficiency, novelty.

TIER 4: HYPER-* LAYERS | Hyper-AGI Layer

19. Hyper-Execution Layer

Description

The Hyper-Execution Layer provides advanced execution capabilities: terminal remote execution, browser automation, code sandboxing, Claude/Codex bridging, leaf/orchestrator sub-agent spawning, result collection, error recovery, and toolset maximization. This layer enables the agent to execute anything, anywhere.

Architecture

Layer: EXECUTION (Skills 31-40 of 50 in Hyper-AGI OS)
Capabilities: Terminal, Browser, Sandbox, Bridging, Spawning, Collection, Recovery, Maximization
Integration: Activated by moussaif1b kernel when complex execution is required

Main 30 Skills

1. **Terminal Remote Execution**
   Executes shell commands locally and remotely: git, pip, npm, docker, kubectl, custom scripts with proper error handling.
2. **Browser Automation**
   Navigates, clicks, types, snapshots, extracts data from web pages for evidence collection and testing.
3. **Code Sandbox Execution**
   Runs Python code in sandboxed environment with hermes_tools access for safe experimentation.
4. **Claude Code Bridge**
   Bridges tasks to Claude Code CLI: symlinks, context copy, invocation with --acp flag for integration.
5. **Codex Bridge**
   Bridges tasks to Codex CLI: symlinks, context copy, invocation with --acp --stdio for integration.
6. **Leaf Sub-Agent Spawning**
   Spawns leaf agents with isolated contexts, tailored toolsets, clear sub-goals via delegate_task.
7. **Orchestrator Sub-Agent Spawning**
   Spawns orchestrator agents that can further delegate for complex multi-step sub-tasks.
8. **Result Collection & Merging**
   Collects results from all sub-agents into unified summary with status tracking and conflict resolution.
9. **Error Recovery**
   Recovers from failures: retry with modified parameters, fallback strategy, escalation to human.
10. **Toolset Maximization**
    Enables all relevant toolsets per sub-agent to ensure MAXIIIMUM capability coverage.
11. **Parallel Task Execution**
    Fans out independent tasks across multiple agents simultaneously with result synchronization.
12. **Process Management**
    Manages long-running processes: start, stop, monitor, log, restart with proper cleanup.
13. **Environment Isolation**
    Isolates execution environments: separate working directories, env vars, tool access per task.
14. **Resource Allocation**
    Allocates resources per task: CPU limits, memory bounds, time limits, concurrent execution caps.
15. **Execution Monitoring**
    Monitors execution: progress tracking, resource usage, error detection, timeout management.
16. **Output Stream Handling**
    Handles output streams: stdout/stderr capture, real-time display, log aggregation, filtering.

17. **Cross-Agent Communication**
    Enables agent-to-agent messaging: task delegation, status updates, result passing, conflict resolution.
18. **Execution Context Management**
    Manages execution context: environment variables, working directory, tool availability, permissions.
19. **Batch Execution**
    Executes task batches: sequential or parallel, with dependency ordering and result aggregation.
20. **Execution Rollback**
    Rolls back failed executions: restore checkpoints, undo changes, clean up partial results.

TIER 4: HYPER-* LAYERS | Hyper-AGI Layer

20. Hyper-Planning Layer

Description

The Hyper-Planning Layer provides advanced planning capabilities: DAG building with topological sort, parallel fanout, delegation strategy, resource allocation, rollback planning, contingency planning, priority queuing, milestone tracking, cron scheduling, and task interleaving.

Architecture

Layer: PLANNING (Skills 21-30 of 50 in Hyper-AGI OS)
Capabilities: DAG, Fanout, Delegation, Allocation, Rollback, Contingency, Priority, Milestones, Cron, Interleaving
Integration: Activated by moussaif1b kernel when complex planning is required

Main 30 Skills

1. **DAG Builder (Topological Sort)**
   Builds directed acyclic graphs of task dependencies with topological sort and critical path analysis.
2. **Parallel Fanout**
   Fans out independent sub-tasks as parallel delegate_task calls respecting max_concurrent_children limit.
3. **Delegation Strategy**
   Decides which sub-tasks to delegate vs execute inline based on complexity, isolation needs, and agent capabilities.
4. **Resource Allocation**
   Allocates tools, model budget, time limits, and sub-agent slots per sub-task based on priority.
5. **Rollback Planning**
   Plans rollback points before risky operations: checkpoint state, record checksums, prepare restore scripts.
6. **Contingency Planning**
   Generates contingency plans for failure modes: network errors, API failures, timeouts, corruption.
7. **Priority Queue**
   Maintains persistent priority queue of pending sub-tasks with dynamic re-prioritization based on changing conditions.
8. **Milestone Tracking**
   Tracks progress toward milestones: update todo list, notify on completion or blocker, estimate remaining work.
9. **Cron Scheduling**
   Schedules recurring sub-tasks as cron jobs with appropriate frequency, delivery, and skill attachments.
10. **Task Interleaving**
    Interleaves background tasks with interactive conversation so user sees progress without blocking.
11. **Plan Optimization**
    Optimizes plans: minimize total time, balance load, reduce dependencies, maximize parallelism.
12. **Plan Validation**
    Validates plans: feasibility check, resource availability, dependency resolution, time constraint satisfaction.
13. **Execution Order Determination**
    Determines optimal execution order: critical path first, quick wins early, dependencies satisfied.
14. **Plan Adaptation**
    Adapts plans dynamically: adjust to new information, handle failures, incorporate completed work.
15. **Plan Documentation**
    Documents plans: step descriptions, expected outputs, verification criteria, rollback procedures.
16. **Multi-Phase Planning**
    Plans in phases: discovery, planning, execution, verification with gates between phases.

17. **Plan Risk Assessment**
    Assesses plan risks: failure probability, impact severity, mitigation strategies, early warning signals.
18. **Resource Conflict Resolution**
    Resolves resource conflicts: shared tools, concurrent file access, API rate limits, agent contention.
19. **Plan Progress Reporting**
    Reports progress: completed steps, current phase, blockers encountered, time estimates.
20. **Plan Version Control**
    Tracks plan versions: changes made, reasons, approval status, rollback to previous versions.

TIER 4: HYPER-* LAYERS | Hyper-AGI Layer

21. Hyper-Perception Layer

Description

The Hyper-Perception Layer provides advanced perception capabilities: environment scanning, session recall, skill discovery, KB scanning, user intent parsing, resource monitoring, change detection, context window management, emotion/affect detection, and sensor fusion. This layer enables the agent to perceive everything.

Architecture

Layer: PERCEPTION (Skills 1-10 of 50 in Hyper-AGI OS)
Capabilities: Scan, Recall, Discovery, Scanning, Parsing, Monitoring, Detection, Window, Affect, Fusion
Integration: Activated by moussaif1b kernel at mission intake for environment awareness

Main 30 Skills

1. **Environment Scanning**
   Scans filesystem, network state, running processes, system configuration to build comprehensive environment model.
2. **Session Recall (FTS5)**
   Recalls relevant past sessions using full-text search to find similar problems and solutions.
3. **Skill Discovery**
   Discovers all available skills across Hermes, Codex, Claude, and project directories by scanning SKILL.md files.
4. **Knowledge Base Scanning**
   Scans all KBs (Afripa, GOAT, Ebook, HyperOS) for relevant prior work and patterns.
5. **User Intent Parsing**
   Parses natural language into structured goal objects with priority, deadline, constraints, and success criteria.
6. **Resource Monitoring**
   Monitors CPU, RAM, disk, GPU, token budgets in real-time to inform resource allocation decisions.
7. **Change Detection**
   Detects file changes, new sessions, system events, skill modifications using polling and notifications.
8. **Context Window Management**
   Expands and manages context windows across provider token limits with automatic compression and caching.
9. **Emotion/Affect Detection**
   Detects emotional tone in user messages and adjusts response style, urgency, and verbosity accordingly.
10. **Sensor Fusion**
    Fuses multi-modal inputs (text, image, audio, code) into unified representation for downstream reasoning.
11. **Filesystem State Awareness**
    Real-time awareness of file system: existence, sizes, permissions, modification times, content previews.
12. **Network State Detection**
    Detects network conditions: connectivity, latency, bandwidth, DNS resolution, proxy configuration.
13. **Process State Monitoring**
    Monitors running processes: resource usage, status, dependencies, startup/shutdown detection.
14. **System Configuration Profiling**
    Profiles system: OS version, installed tools, language runtimes, hardware capabilities, environment variables.
15. **Tool Availability Detection**
    Detects available tools: CLI tools, libraries, APIs, services with version and capability checking.
16. **Disk Space Monitoring**
    Monitors disk usage: available space, OneDrive eviction risk, large file detection, cleanup opportunities.

17. **API Health Probing**
    Probes API health: endpoint availability, response time, error rate, authentication status.
18. **Dependency Graph Awareness**
    Maps project dependencies: package versions, compatibility, vulnerability status, update availability.
19. **Git State Detection**
    Detects git state: current branch, uncommitted changes, remote sync status, conflict markers.
20. **Environment Snapshot**
    Captures environment snapshots: complete state for comparison, debugging, or rollback reference.

TIER 4: HYPER-* LAYERS | Hyper-AGI Layer

22. Hyper-Integrity Layer

Description

The Hyper-Integrity Layer provides data integrity capabilities: audit logging, backup management, checksum verification, conflict resolution, deduplication, and symlink mapping. This layer ensures zero-loss data integrity across all operations.

Architecture

Layer: INTEGRITY (Skills 41-50 of 50 in Hyper-AGI OS)
Capabilities: Audit, Backup, Checksum, Conflict, Dedup, Symlink
Integration: Active during all file operations for zero-loss guarantee

Main 30 Skills

1. **Audit Logger**
   Logs all file operations with timestamp, operation type, file path, checksum, and agent ID for complete audit trail.
2. **Backup Manager**
   Automated backup to OneDrive with checksum verification: incremental, deduplicated, and versioned.
3. **Checksum Verifier**
   SHA-256 checksum verification of all files before and after operations to ensure zero data loss.
4. **Conflict Resolver**
   Detects and resolves file conflicts: version comparison, merge strategies, manual review flags.
5. **Dedup Consolidator**
   Consolidates duplicate KBs and skills across all projects into single source of truth in Hermes root.
6. **Symlink Mapper**
   Creates and maintains symlinks from consolidated skills to all project dirs and Codex/Claude environments.
7. **File Integrity Monitoring**
   Continuous monitoring of critical files: change detection, unauthorized modification alerts, tamper evidence.
8. **Version History Tracking**
   Tracks file version history: creation, modification, deletion with diff records and rollback capability.
9. **Backup Verification Testing**
   Tests backup integrity: restore drills, data completeness checks, recovery time measurement.
10. **Data Loss Prevention**
    Prevents data loss: confirmation before destructive operations, recycle bin usage, recovery procedures.
11. **Cross-Environment Consistency**
    Ensures consistency across Hermes, Codex, Claude, Obsidian: synchronized state, conflict detection.
12. **Integrity Report Generation**
    Generates integrity reports: file counts, checksum status, backup freshness, conflict summary.
13. **Recovery Procedure Execution**
    Executes recovery: restore from backup, undo failed operations, repair corrupted data.
14. **Integrity Policy Enforcement**
    Enforces integrity policies: mandatory backups, checksum requirements, symlink preservation rules.
15. **Integrity Dashboard**
    Maintains integrity dashboard: backup status, data health, conflict count, recovery readiness.

TIER 4: HYPER-* LAYERS | Hyper-AGI Layer

23. Hyper-Evolution Layer

Description

The Hyper-Evolution Layer provides self-improvement capabilities: memory optimization, self-improvement loops, skill creation, and skill curation. This layer enables the agent to evolve and improve over time through regression-gated learning.

Architecture

Layer: EVOLUTION (Skills 46-50 of 50 in Hyper-AGI OS)
Capabilities: Memory Optimization, Self-Improvement, Skill Creation, Skill Curation
Integration: Active after every mission for learning and skill evolution

Main 30 Skills

1. **Memory Optimizer**
   Prunes stale facts, consolidates redundant entries, frees character budget, maintains memory quality.
2. **Self-Improvement Loop**
   Analyzes past performance: success patterns, failure modes, efficiency gains, quality improvements.
3. **Skill Creator**
   Auto-creates new skills from successful task patterns with provenance tracking and quality gates.
4. **Skill Curator**
   Curates, archives, and consolidates skills: detect stale, overlapping, or obsolete capabilities.
5. **Learning Rate Adaptation**
   Adjusts evolution speed: rapid early, conservative mature, emergency for critical fixes.
6. **Evolution Pattern Recognition**
   Identifies evolution patterns: recurring improvements, systematic gaps, capability trends.
7. **Skill Quality Assessment**
   Assesses skill quality: completeness, accuracy, usability, test coverage, documentation.
8. **Evolution Budget Management**
   Manages evolution resources: time allocation, testing capacity, deployment slots.
9. **Cross-Session Learning**
   Learns across sessions: persistent patterns, user preferences, environment changes.
10. **Evolution Report Generation**
    Generates evolution reports: changes made, metrics, pending proposals, regression results.