

COMPLETE REFERENCE



# ACOS Reference

---

The definitive guide to every command, agent, skill, and workflow in the Agentic Creator OS.

FRANKX | v7.0 | 2026

# 01 SYSTEM

## System Architecture

ACOS v7.0 is built on a layered architecture: **CLAUDE.md** defines the system, **skill-rules.json** manages auto-activation, **commands/** contain slash command definitions, and **hooks/** automate lifecycle events.



*Every command routes through intelligent classification to the optimal agent*

## 02

## COMMANDS

## Command Reference: Core

COMMAND	CATEGORY	DESCRIPTION
<code>/acos</code>	Router	Universal auto-router — classifies intent and routes to optimal agent
<code>/frankx-ai-build</code>	Dev	Full build session with quality gates and testing
<code>/frankx-ai-deploy</code>	DevOps	Two-repo production deployment to frankx.ai
<code>/frankx-ai-blog</code>	Content	Create SEO-optimized blog post with schema markup
<code>/frankx-ai-seo</code>	SEO	Site-wide SEO analysis and optimization
<code>/frankx-ai-components</code>	UI	React component development with Tailwind
<code>/frankx-ai-content-pipeline</code>	Content	End-to-end content: research → write → SEO → publish
<code>/frankx-ai-products</code>	Product	Digital product creation and launch workflow
<code>/frankx-ai-prompts</code>	AI	Prompt library management and generation
<code>/frankx-ai-analytics</code>	Data	Site analytics and performance monitoring
<code>/frankx-ai-daily</code>	Ops	Daily operations and task management

# Command Reference: Agentic

COMMAND	CATEGORY	DESCRIPTION
<code>/ultraworld</code>	Creative	Arcanean swarm world-building — 7 creative agents
<code>/acos-swarm</code>	Multi-Agent	Swarm coordination for parallel agent execution
<code>/acos-flow</code>	Workflow	Workflow orchestration and pipeline management
<code>/acos-agents</code>	Agents	Agent lifecycle: spawn, monitor, coordinate
<code>/acos-memory</code>	Memory	Persistent memory and cross-session learning
<code>/acos-checkpoint</code>	State	Session state checkpointing and recovery
<code>/agentic-jujutsu</code>	Learning	Trajectory extraction and pattern learning

## Advanced Orchestration

COMMAND	CATEGORY	DESCRIPTION
<code>/claude-flow-swarm</code>	Internal	Low-level swarm spawning and control
<code>/claude-flow-memory</code>	Internal	Neural memory persistence layer
<code>/claude-flow-sparc</code>	Internal	SPARC methodology execution
<code>/claude-flow-agents</code>	Internal	Agent type registry and capabilities

*"83 internal claude-flow commands were absorbed into ACOS v7.0 — never surfaced to users, always working behind the scenes."*

# Agent Reference



## Technical Architect

**Skills:** architecture-patterns, ai-agents-architect, api-design, oracle-adk

**Triggers:** architecture, system design, backend, API



## Music Producer

**Skills:** suno-ai-mastery, suno-prompt-architect

**Triggers:** suno, music, song, audio



## Content Engine

**Skills:** frankx-brand, seo-content-writer, schema-markup

**Triggers:** article, blog, content, post



## SEO Intelligence

**Skills:** seo-fundamentals, keyword-strategist, schema-markup

**Triggers:** seo, keywords, rankings



## Frontend Designer

**Skills:** vercel-react, web-design, shadcn-ui, tailwind, accessibility

**Triggers:** component, design, ui, ux



## DevOps Engineer

**Skills:** vercel-deployment, docker, github-actions

**Triggers:** deploy, build, ci, pipeline

# Auto-Activation Rules

The `skill-rules.json` system uses 22 rules to automatically activate skills based on file patterns, keywords, and task context. 5 core skills are always active.

## Always Active (Core 5)

`frankx-brand` Brand voice & guidelines

`clean-code` Code quality standards

`next-best-practices` Next.js patterns

`typescript-docs` TypeScript standards

`vercel-react` React optimization

## Sample Activation Rules

TRIGGER	PATTERN	SKILLS ACTIVATED
File: <code>*.tsx</code>	Component files	vercel-react, tailwind-css, shadcn-ui
File: <code>*.mdx</code>	Blog content	seo-content-writer, schema-markup
Keyword: "deploy"	Deployment tasks	vercel-deployment, github-actions
Keyword: "suno"	Music creation	suno-ai-mastery, suno-prompt-architect
Keyword: "agent"	Agent design	ai-agents-architect, autonomous-agents
Context: Arcanea	Creative gates	Gate-specific skills (10 gates mapped)

# 06

AUTOMATION

## Hooks & Lifecycle Events

ACOS hooks into 7 lifecycle events with 20+ automation hooks, executing shell scripts and Node.js hooks automatically during session and task lifecycles.

### SessionStart

Loads context, initializes learning trajectories, activates skill profiles, sets output style.

### UserPromptSubmit

Classifies intent, suggests skills, activates brand guidelines, logs context.

### PreToolUse

Validates tool permissions, applies security policies, manages file access.

### PostToolUse

Logs tool usage, updates trajectory data, triggers quality checks.

#### Hook Execution Log

```
[SessionStart] Loading context...
→ resume hook: ✓ ACOS Session Context loaded
→ learning hook: ✓ 23 trajectories, 22 patterns
→ output-style hook: ✓ explanatory mode

[UserPromptSubmit] Classifying intent...
→ skill-suggest hook: ✓ 3 skills recommended
→ brand hook: ✓ frankx-brand activated
✓ All hooks executed in 0.8s
```

# Memory & Learning

ACOS maintains persistent memory across sessions and uses **Agentic Jujutsu** to extract successful patterns from past interactions, continuously improving performance.

## Memory Layers

### **MEMORY.md**

Auto-loaded system prompt.  
Architecture decisions, patterns, workarounds. Kept under 200 lines.

### **Topic Files**

Detailed notes organized by topic: debugging.md, patterns.md, blog-design-patterns.md.

### **MCP Memory**

Knowledge graph via MCP server — entities, relations, observations across sessions.

## Agentic Jujutsu

### **Trajectory Tracking**

Every session records tool sequences, outcomes, and success rates.

### **Pattern Extraction**

Successful tool sequences are extracted and stored for reuse.

### **Self-Improvement**

Patterns inform future routing decisions, improving over time.



# 08

CONFIG

## Configuration Reference

### File Structure

ACOS Directory Structure

```
.claude/
├─ commands/ # 46+ slash command definitions
├─ agents/ # Agent profile definitions
├─ skills/ # Local skill rules & profiles
├─ hooks/ # 4 shell + 4 JS hooks
├─ context/ # Decision framework, identity
├─ planning/ # Task plans and strategies
├─ settings.json # Permissions and config

CLAUDE.md # System definition — the brain
skill-rules.json # 22 auto-activation rules
```

### Key Configuration Files

FILE	PURPOSE	FORMAT
CLAUDE.md	Master system prompt — brand, agents, routing	Markdown
skill-rules.json	Auto-activation rules and skill profiles	JSON
settings.local.json	Tool permissions and allowed commands	JSON
.frankx/brand.md	Brand voice and design guidelines	Markdown
MEMORY.md	Persistent auto-memory (< 200 lines)	Markdown

REFERENCE V7.0

# Agentic Creator OS

---

630+ skills. 40+ agents. 130+ commands.  
One operating system for everything you create.

`frankx.ai` | `github.com/frankxai/agentic-creator-os`

Build what matters.