

SmartAssist Enterprise Agent Platform: MCP Tool and Skill Development

Building governed MCP tool servers and skills for SmartAssist's specialized agents, with Beast Assist as the working reference implementation.

PREPARED FOR

Cognizant · SmartAssist Platform Lead (via Long Finch Technologies)

PREPARED BY

Bryan Ruiz · Scarlet Beast Corporation

ENGAGEMENT

Contract, remote, long term · start: immediately

DOCUMENT

Version 1.0 · 21 October 2026

1 Purpose

This Scope of Work describes what the contractor will build and maintain for the SmartAssist Enterprise Agent Platform (Serenity), how the work will be delivered and accepted, and what the contractor needs from the platform team. It is a proposal for discussion with the SmartAssist Platform Lead. The phases, priorities and backend systems are expected to change once the contractor has access to the platform's own template, registry and backlog.

2 Background

SmartAssist assembles specialized AI agents from two building blocks: **MCP tool servers** that expose enterprise systems, and **skills** that package those tools with business instructions, an authorization scope and workflows. Well-designed tools plus well-scoped skills let the agent development team build reliable, least-privilege agents without changing application code.

Reference implementation. Beast Assist, built by the contractor against this role's job description, shows the approach working end to end: two FastAPI MCP servers, three SKILL.md skills, a PostgreSQL Skills Registry, OPA policy, Entra ID tokens through Azure API Management, a LangGraph coordinator agent on Amazon Bedrock, pgvector RAG, evaluation suites and Kubernetes manifests. It runs live at cognizant.scarletbeast.com, and the source is at github.com/christianscarlet2/beast-assist.

3 Objectives

- Grow SmartAssist's tool catalog with new single-platform MCP servers built from `mcp-template`, each production-ready, tested and documented.
- Author and maintain skills whose `allowed_tools` scope is minimal and correct, each passing `validate_skills.py` and `audit-skill-config` with no failures before production.
- Make every tool call governed end to end: Entra JWT at APIM, OPA scope from the Skills Registry, and least privilege by default.
- Give the agent team evaluation suites that prove tools work, and observability that explains agent runs.
- Leave everything maintainable by internal teams without the contractor.

4 In scope

4.1 MCP tool servers

- New servers from `mcp-template`, one backend per server, following auto-discovery in `src/tools/` and the single-responsibility file structure.
- List, get and create-style tools plus a bootstrap guide; LLM-friendly schemas, slim outputs, actionable errors.
- Unit tests and integration tests against each backend's test environment.

4.2 Skills

- SKILL.md manifests: `allowed_tools`, pre-flight inputs, aliases, tags, lifecycle stage.
- Workflows with execution steps, error-handling tables and gotchas.
- Validation, audit, promotion and publishing through the registry upload scripts.

4.3 Platform integration

- Azure APIM routing and policy; Entra JWT authentication and app roles.
- Skills Registry (PostgreSQL) and OPA enforcement with least-privilege scoping.

4.4 Deployment and operations

- Docker images, Kubernetes manifests, health and readiness endpoints, certificates, environment configuration.
- CI/CD in the platform's Git-based multi-repo workflow.

4.5 Quality, observability, knowledge transfer

- Evaluation suites and benchmark scenarios per server.
- Alignment with existing tracing, tool-call analysis and logging.
- Technical documentation, architecture diagrams and handover sessions.

5 Out of scope

- Changes to the SmartAssist agent runtime or the agent team's application code, beyond agreed contract changes.
- Building or replacing the platform's core services (APIM, the Skills Registry, OPA, Kubernetes clusters). The contractor integrates with them and proposes improvements.
- Administering the backend systems (ServiceNow, GitLab, Ansible, Jira or others) or changing their data models.
- Production on-call outside agreed hours, unless added by change request.

6 Deliverables and acceptance criteria

ID	Deliverable	Accepted when
D1	Onboarding notes and backlog proposal	The Platform Lead agrees on the first servers and skills, their priority and their backends
D2	An MCP tool server per agreed backend	Built from <code>mcp-template</code> ; unit and integration tests pass in CI; <code>/healthz</code> and <code>/readyz</code> respond; deployed behind APIM with Entra JWT
D3	Tools on each server, including a bootstrap guide	Each tool has a schema, a description, tests and documentation; errors follow the platform's error convention
D4	Skills for each agreed use case	<code>validate_skills.py</code> passes; <code>audit-skill-config</code> reports no failures; published through the registry; OPA allows exactly the declared <code>allowed_tools</code>

D5	Skill workflows	Steps, error-handling tables and gotchas are reviewed by the agent team and exercised by evaluation scenarios
D6	Evaluation suites and benchmark scenarios	Run in CI; cover the happy path, refusals (scope, role, lifecycle) and backend errors for each server
D7	Deployment configuration	Docker images and Kubernetes manifests merged; certificates and environment configuration documented and running
D8	Observability alignment	Tool calls show up in the platform's tracing and logging with skill, tool, caller and outcome
D9	Documentation and knowledge transfer	README, architecture diagram and runbook for each server; a handover session held; an internal engineer can add a tool without help

7 Approach and timeline

A proposed rhythm. After phase 2, new servers and skills follow the same cycle and run in parallel as the backlog allows.

Week 1	Onboard. Access, environments, a walkthrough of <code>mcp-template</code> , the registry, APIM and OPA. Review the catalog and its conventions. Deliver D1.
Weeks 2–3	First server. One server end to end on the agreed backend: tools, tests, deployment behind APIM, health endpoints, documentation (D2, D3, D7).
Weeks 3–5	First skills. Skills on the new server with workflows and error tables; validate, audit, publish; check OPA scoping (D4, D5).
Weeks 4–6	Quality loop. Evaluation suites and benchmark scenarios in CI; observability alignment (D6, D8).
Week 6 onward	Scale out. More servers and skills from the backlog in two-week cycles; catalog consistency reviews; documentation and handover for each component (D9).

8 Acceptance

Each deliverable is submitted as a merge request with passing CI, evaluation results and documentation. The Platform Lead (or someone they name) reviews it within five business days and either accepts it or lists specific changes. Accepted deliverables are promoted through the platform's lifecycle stages (draft → beta → production) under its existing rules.

9 Assumptions and dependencies

- Access to the SmartAssist repositories, `mcp-template`, the `create-skill` and `audit-skill-config` skills, the Skills Registry, and non-production APIM, Entra and Kubernetes environments.
- Test credentials and test tenants for each backend, plus a contact who knows its API.
- Entra app registrations and role assignments requested through the platform team's process.
- Backlog priorities come from the Platform Lead; the agent team is available to review tool and skill contracts.

10 Roles and responsibilities

Role	Responsibilities
Contractor (Bryan Ruiz)	Design, build, test, document and support the deliverables above; raise risks early; keep tool and skill contracts consistent across the catalog

SmartAssist Platform Lead	Priorities, reviews and acceptance; access and environments; decisions on conventions
AI agent development team	Agent requirements; review of tool and skill contracts; feedback from real agent runs
Backend system owners	API access, test data, and answers about backend behavior

11 Communication and change control

- A weekly written status: delivered, in progress, risks and blockers. Work is tracked on the platform's board.
- New backends, scope beyond section 4, or changed acceptance criteria are agreed in writing before work starts.

12 Commercial terms

Time and materials, under the agreement between Long Finch Technologies and Scarlet Beast Corporation. Rates, invoicing and term are set in that agreement and not in this document. Availability: immediate start, remote.

For Cognizant / SmartAssist: name, title, date

Bryan Ruiz, Scarlet Beast Corporation: date