🧭 Cerebricks Agent Onboarding & SDK Usage Guide

Welcome to **Cerebricks**, the home and office of Quadrant's agents.
This guide helps you onboard your agent into Cerebricks and integrate the **Cerebricks Events SDK** to standardize event reporting and health monitoring.

📝 1. Required Information to Onboard

When registering a new agent in Cerebricks, please provide the following details.
Each field ensures that your agent is identifiable, verifiable, and properly displayed in the Cerebricks Console.

Field	Description	Example
source	Endpoint or API URL of your agent	`https://api.hragent.quadranttech.com`
specversion	Version of event spec used (always `"1.0"`)	`1.0`
agentName	Human-readable name of your agent	`HR Onboarding Agent`
secretName	Secret identifier for signing & verification	`quadrant/hr-onboarding-secret`
owner_mail	Email of the agent owner or maintainer	`alice@quadranttech.com`
department_name	Department where this agent belongs	`Human Resources`
manager_name	Reporting manager for the owner	`Bob Stevens`

phoneNumber	Contact number of the owner	`+91 9876543210`
agent_summary	One-line overview of what the agent does	`Analyzes resumes and schedules candidate interviews`
agentUsageInstruction	Short how-to for users	`Upload a resume → Agent auto-screens → Schedule generated`
strengths	Key areas where your agent excels	`Fast resume analysis, automated scheduling`
capabilities	Feature list or supported operations	`Resume parsing, Candidate shortlisting, Scheduling`
workflow	High-level flow of how your agent operates	`Search → Analyze → Respond → Store`
key_activities	Main tasks performed	`Search candidates, schedule`

		`meetings, send emails`
gender	Voice/personality of agent (if applicable)	`Female`
limitations	Known gaps or constraints	`Cannot evaluate coding tests yet`
input_schema	JSON schema for what the agent expects as input	`{ "type": "object", "properties": { "resumeUrl": {"type": "string"} }}`

> 💡 **Tip:** These details populate your agent's profile card inside Cerebricks and help others discover and use it easily.

⚙️ 2. SDK Overview

The **Cerebricks Events SDK** is a lightweight Python library that lets your agent send standardized lifecycle events (start, heartbeat, status, health, error) to the **Event Manager API**.

It also helps compute **service health scores** by probing your agent's dependent endpoints.

🧩 3. Installing the SDK

```
pip install cerebricks-events

```

🧰 4. Basic Usage

Import the SDK functions in your Lambda or FastAPI service:

```
from cerebricks_events import (
    post_start, post_status_change, post_health,
    post_error, post_heartbeat, compute_health, DEFAULT_TARGETS
)

```

🔹 **Extract Agent Metadata from Headers**

```
agent_id = headers.get("agent_id")
session_id = headers.get("session_id")
initiated_by = headers.get("initiatedby")
authtoken = headers.get("authorization")

```

These headers identify the agent and user running the process.

🔹 **Emit Events**

Each SDK function sends a specific event to the Cerebricks Event Manager:

Event	Function	Description
Start	`post_start()`	Notify that a new agent session has begun
Heartbeat	`post_heartbeat()`	Periodically update the agent's active status
Health	`post_health()`	Report health score and unresponsive dependencies
Status Change	`post_status_change()`	Mark progress or completion of an operation
Error	`post_error()`	Report exceptions or failures

🧠 **Example Workflow**

```
from cerebricks_events import post_start, post_heartbeat, post_health, compute_health,
post_status_change, post_error

agent_id = "arn:quadrant:agent:hragent"
```

```
session_id = "hrtestrun1"
initiated_by = "alice@quadranttech.com"
authtoken = "eyJ0e....."

# Start
post_start(agent_id, session_id, initiated_by, authtoken)

# Heartbeat - start stage
post_heartbeat(agent_id, session_id, initiated_by, authtoken,
               status="start", active_task="started", previous_task="none",
nexttaskdelay=10)

# Heartbeat - processing stage
post_heartbeat(agent_id, session_id, initiated_by, authtoken,
               status="alive", active_task="stage1", previous_task="started",
nexttaskdelay=10)

# Report error (if any)
post_error(agent_id, session_id, initiated_by, authtoken, "E001", "Test error")

# Continue
post_heartbeat(agent_id, session_id, initiated_by, authtoken,
               status="alive", active_task="stage2", previous_task="stage1",
nexttaskdelay=10)

# Compute health
targets = [
    ("resume_s3", "https://manual/Resume_to_S3"),
    ("interview_scheduler", "https://d1xy5.cloudfront.net/vapi/"),
    ("screenshotsupload", "https://manual/interview_screenshots"),
    ("fail_pass", "https://prod/interview_insights"),
]
healthScore, condition, unhealth_endpoints = compute_health(targets)

# Post health
post_health(agent_id, session_id, initiated_by, authtoken, healthScore, condition,
unhealth_endpoints)

# Mark completion
post_status_change(agent_id, session_id, initiated_by, authtoken, new_status="Completed",
status_message="Done")

# Final heartbeat
post_heartbeat(agent_id, session_id, initiated_by, authtoken,
               status="stopped", active_task="stopping", previous_task="stage2",
nexttaskdelay=10)

```

❤️ Health Computation

```
healthScore, condition, unhealthy = compute_health(targets)

```

How it works:

- SDK checks each target URL using `HEAD`, `OPTIONS`, or `GET`.

- If the endpoint responds (`status < 500`) or returns JSON `{ "status": "ok" }` , it's considered **healthy**.
- The final score = `(healthy / total) * 100`.
- Condition = `"healthy"` if score ≥ 70, else `"unhealthy"`.
- The list `unhealthy` includes endpoints that failed.

Returns a tuple:

```
(int healthScore, str condition, list unhealth_endpoints)

```

🛡️ Why Use the SDK?

✅ Ensures all agents report events in a **consistent** format
✅ Integrates with Quadrant's **Event Pipeline & Console**
✅ Provides **automated health checks** and **heartbeat supervision**
✅ Enables **real-time monitoring** of agent status, health, and performance

---

🧭 Summary

- Provide complete metadata while onboarding an agent.
- Integrate the SDK for emitting events and monitoring health.
- Use consistent event flow:
  Start → Heartbeat → Health → Status → Stop/Error.
- Once registered, your agent becomes discoverable and trackable inside the **Cerebricks Console**.