Building the enterprise agentic future with Joule Studio

SAP's AI-first development environment for building custom AI agents, apps, and workflows



Dietmar Steinbichler, SAP

November 2026

Agenda

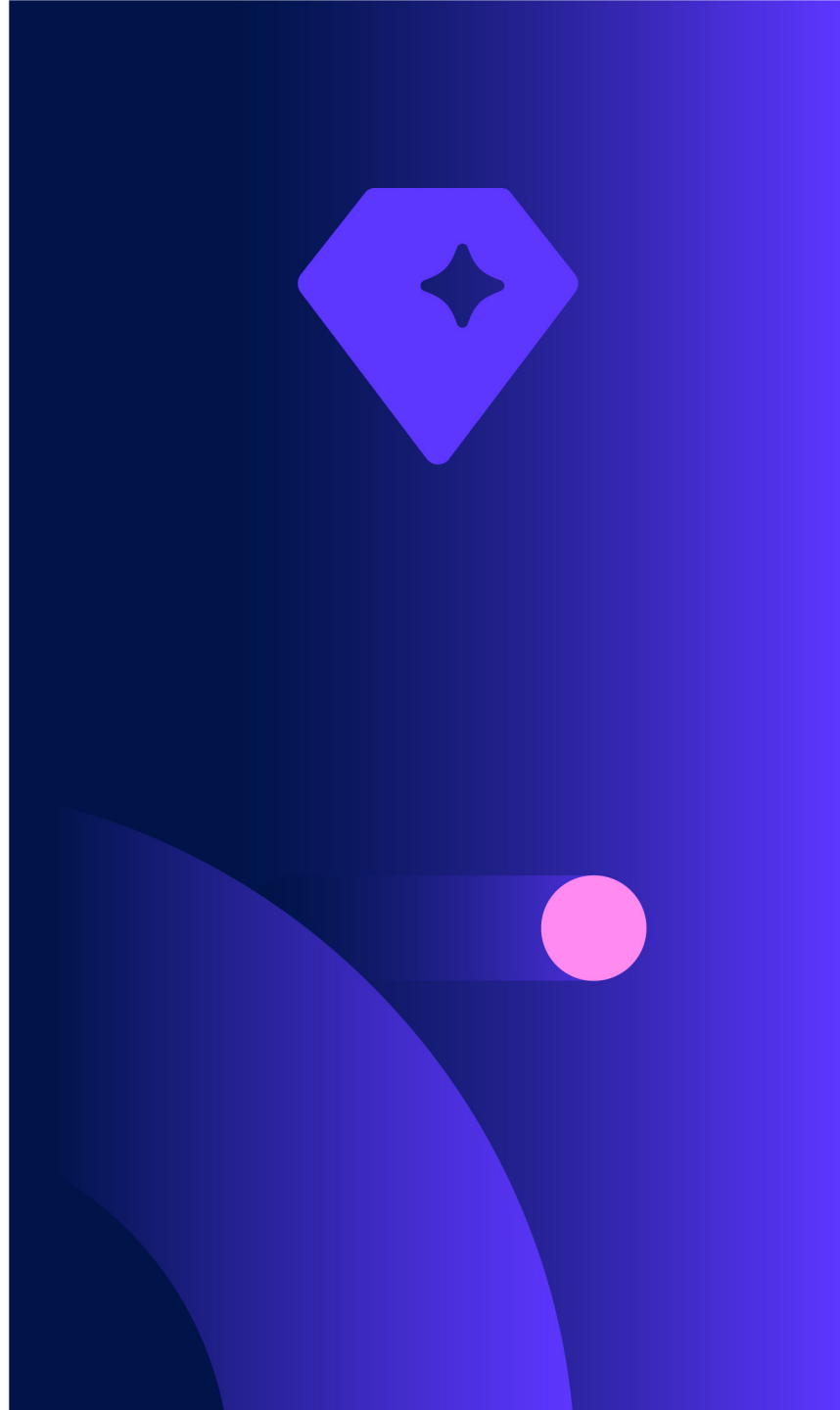
01 Big picture: SAP Business AI Platform

02 Joule Studio: Introduction

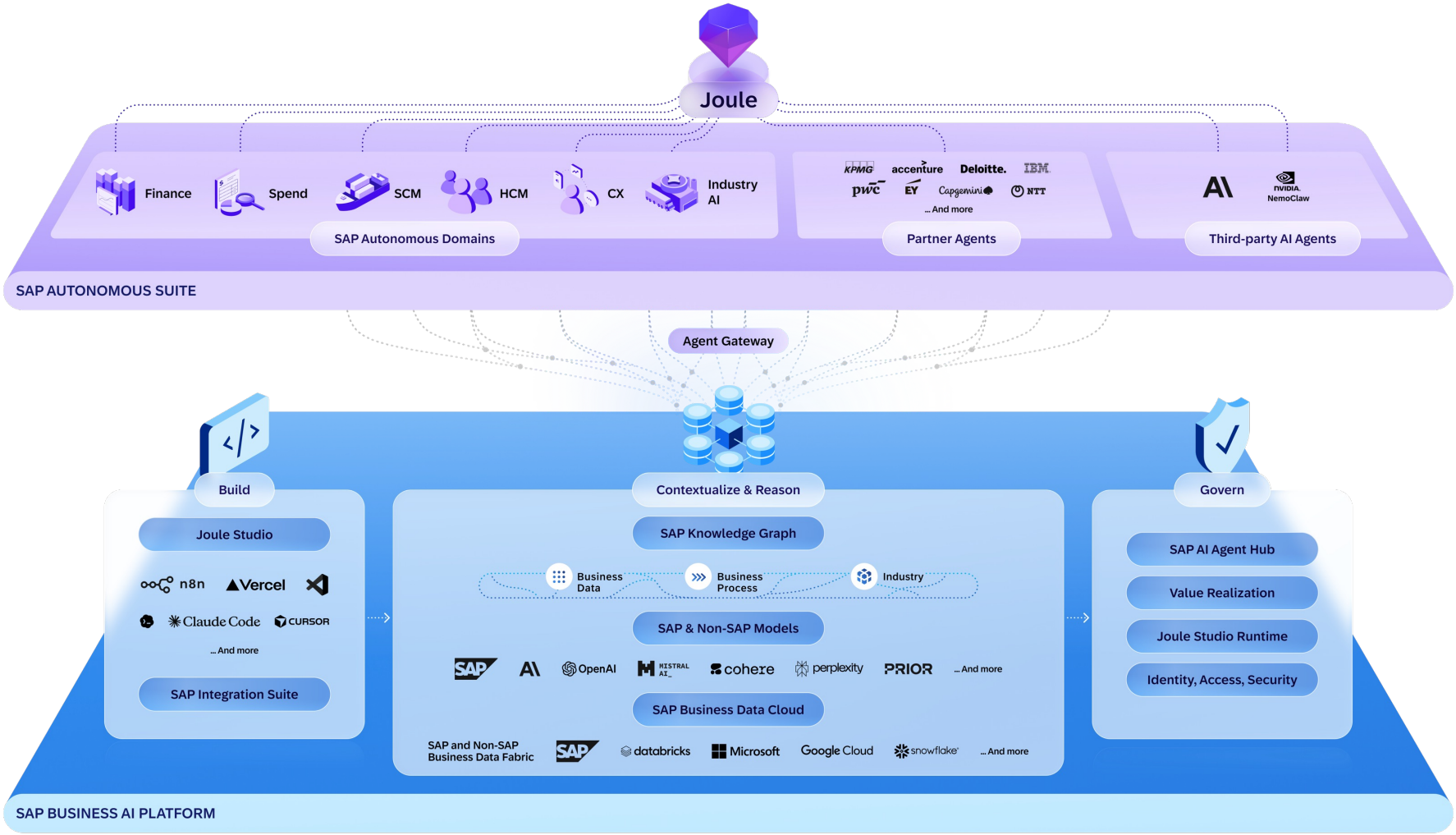
03 Intent-based development

04 Live building example

05 Commercials and how to get started



SAP Business AI Platform



SAP's managed platform offering for agentic workloads

Joule Work

Conversations | **Develop** | Spaces

Joule Studio

AI-assisted development

Design time

Agents
Tools | Document grounding | Agent skills

Apps
CAP | React | Next.js

n8n Workflows
Agent orchestration | Processes | Notifications

Joule Studio for web UI, Joule Studio for CLI, SDK

CURSOR · Cline · Vercel · Claude Code

Automated deployment, operations, and scaling

Joule Studio runtime

Agent frameworks
LangChain | Pydantic AI | LlamaIndex | LangGraph

Agent connectivity
MCP | A2A

Agent security
Identity | Sandboxing | Policy enforcement

Agent governance
SAP AI Agent Hub | Mining | Performance | Eval | Traces

Purpose-built AI models | Always-on compute | Scalable agent memory | Secure storage

SAP Integration Suite
Enterprise integration scenarios

SAP Business Data Cloud
Data products

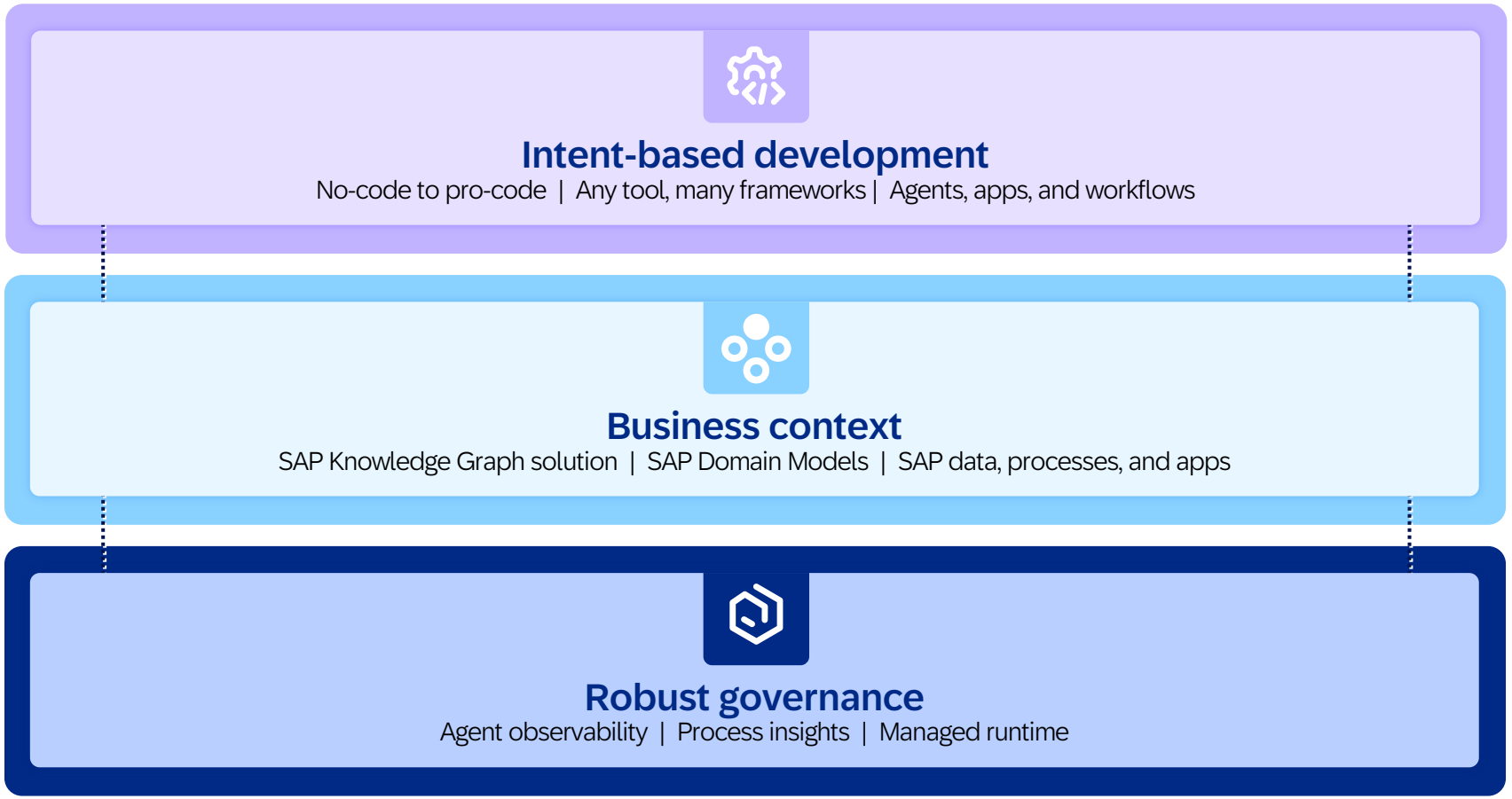
SAP and third-party apps and agents

Joule Studio from intent to execution: The agentic enterprise

Intent-driven generation, deep semantic business context, open developer ecosystems, and fully managed SAP runtime controls

Joule Studio

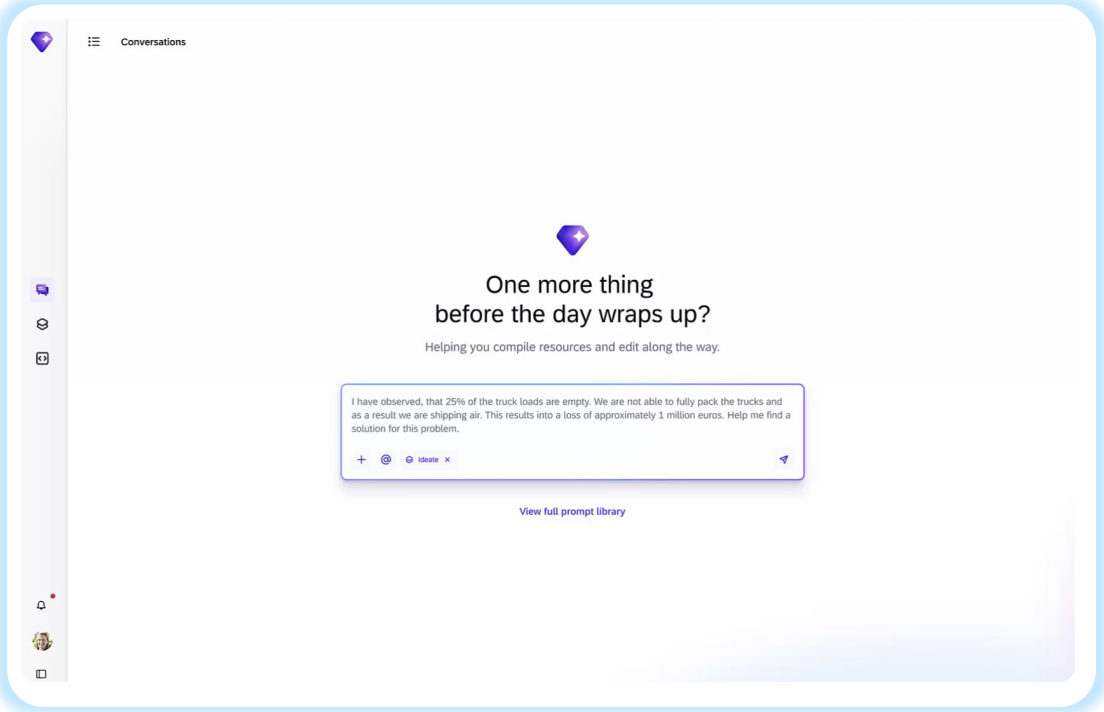
Build reliable, custom agentic solutions



Build solutions by simply stating your intent

Generate, test, and deploy agents, apps, and workflows with AI

- Build agentic solutions faster by **simply describing desired business outcome** in plain language
- Automatically transform ideas into business requirements and technical specifications to **reduce manual effort and speed up delivery**
- Build **agents that deliver reliable outcomes** using SAP's deep business expertise



Powering Intent with Business and Process Intelligence

Create reliable agentic solutions that are deeply grounded in your specific business context

Landscape Insight

"As-Is" Architecture
Target Architecture
Central Landscape Repository
Compliance Policies
Identity Controls

Application Context

Data Models
Object Relationships
Application Content
Application Functionality
Application KPIs

Process Insight

Operational Gaps
Process Bottlenecks
Improvement Recommendations

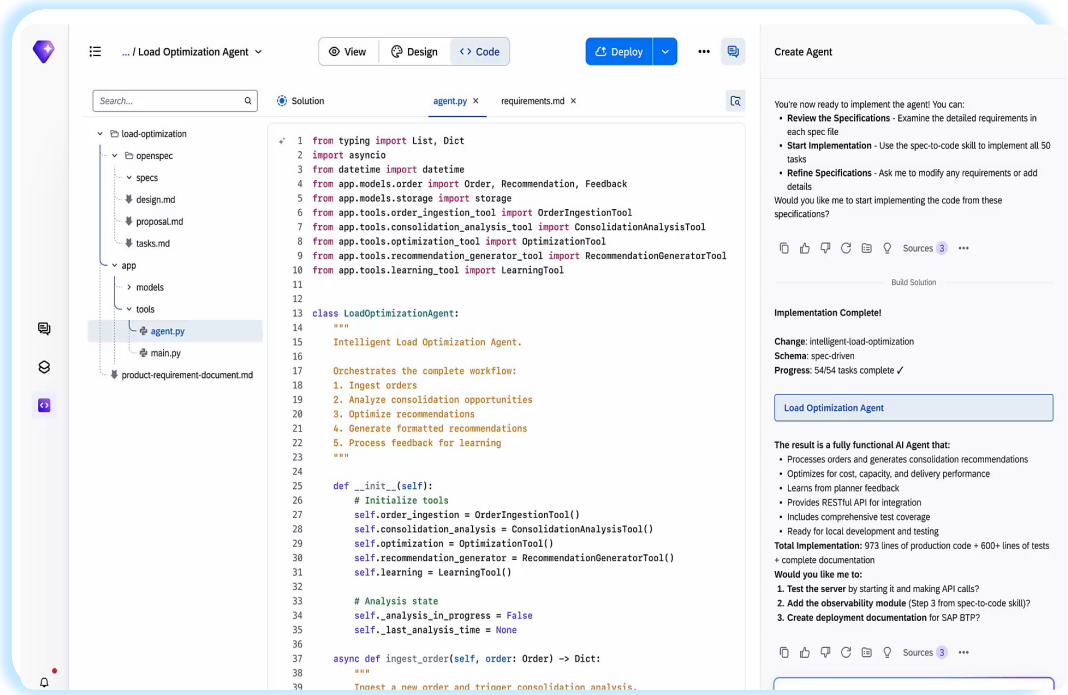
Best Practices

Process Templates
Development Guidelines
Process & Industry Benchmarks

Create agents your way

Flexible and powerful tools to meet developers where they are

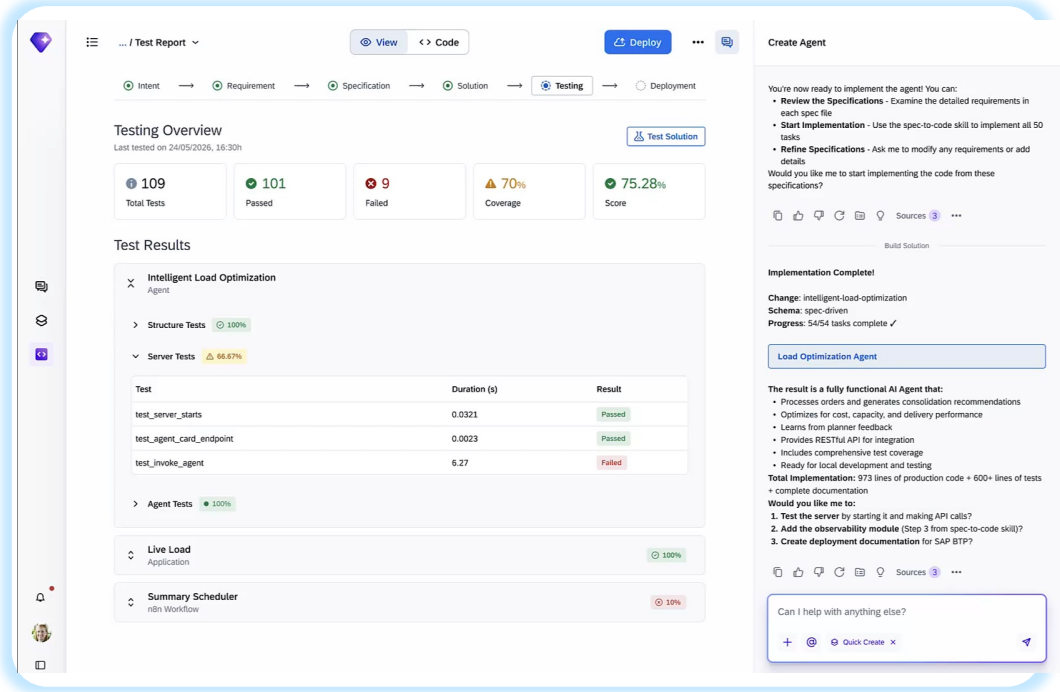
- Build with low-code or pro-code so your teams can **move faster** with full flexibility and control
- Keep developers in the tools they already use while SAP business **context flows seamlessly** into every AI-generated output
- Orchestrate agents in complex workflows across your landscape to **deliver end-to-end automation**



Scale with control and confidence

Joule Studio runtime your trusted environment

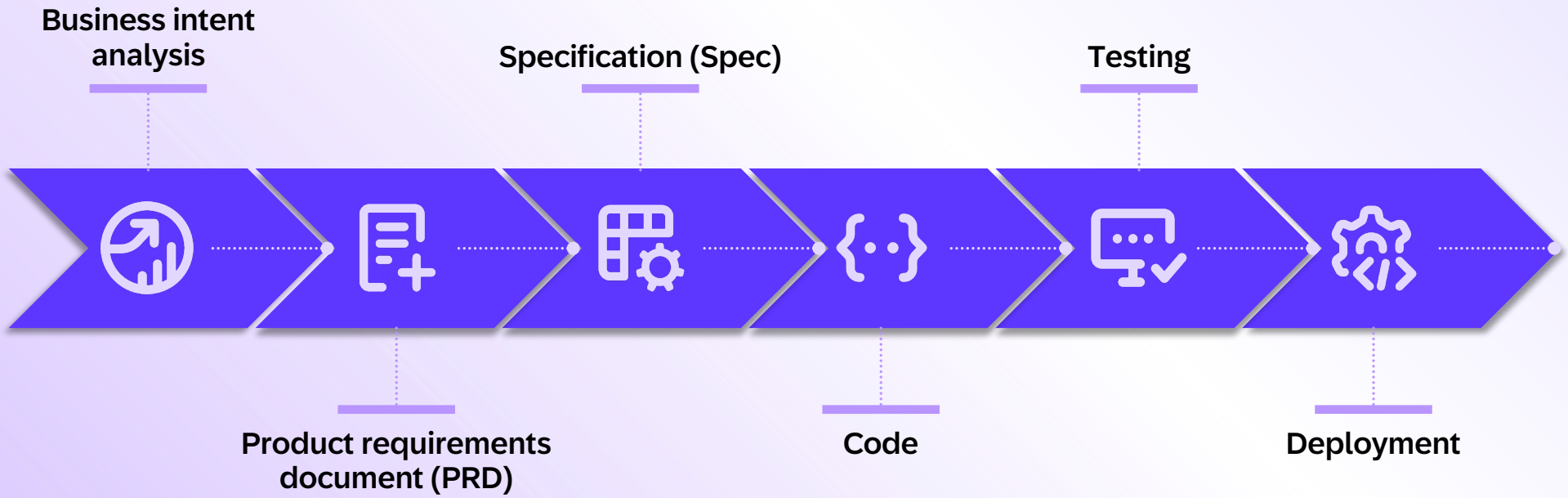
- **Run & Orchestrate** all agents, apps, and workflows in this managed environment
- **Govern** all agents centrally with identity, security, and lifecycle controls
- **Monitor, test, & Improve** agent performance and optimized behavior
- **Scale across business** expanding usage across teams and domains



For more details

Intent-based development in Joule Studio

Turn intent into enterprise-ready solutions



Demo

INTERNAL – SAP and Customers Only

Build an AI Sales Assistant Agent with Joule Studio

Challenge:

Every sales negotiation turns on the same question:

How much discount can I offer, and how do I justify it?

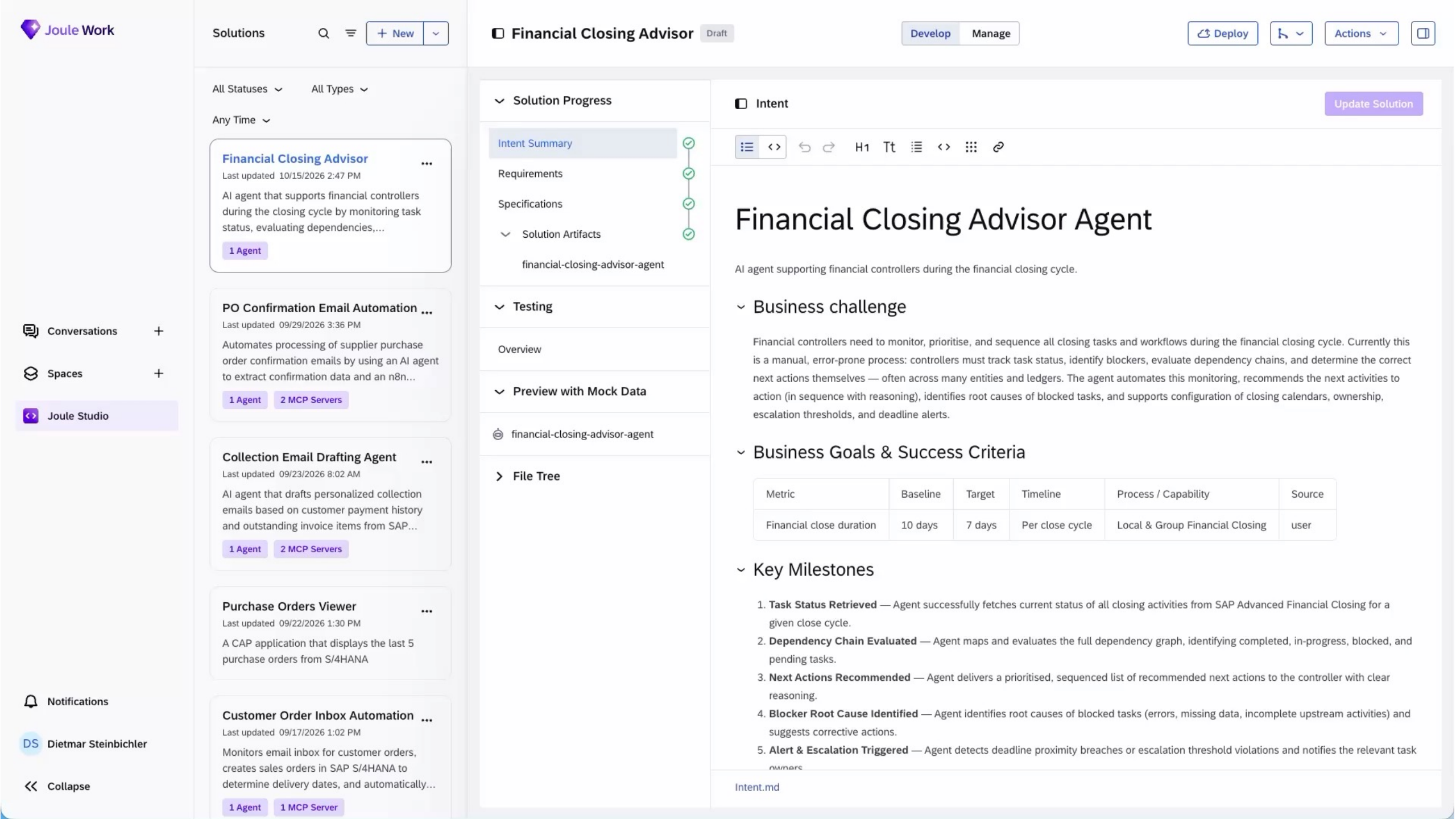
Today, a sales rep answering that question has to cross-reference customer order history in SAP S/4HANA Cloud, open opportunity data in SAP Sales Cloud, and margin targets across the SAP landscape - then simulate the margin impact of a discount in their head.

The decision is usually made from gut feel, and the margin impact only shows up at month-end.

Solution:

An AI Sales Assistant using Joule Studio.

The agent reasons over live SAP data - customer order history, pipeline context, and pricing/margin signals from across SAP Sales Cloud and SAP S/4HANA Cloud - and gives the rep a structured discount recommendation (range, margin impact, risk level, and rationale) at the point of negotiation.


Joule Work
Solutions
New
Financial Closing Advisor
Draft
Develop
Manage
Deploy
Actions
All Statuses
All Types
Any Time
Financial Closing Advisor
Last updated 10/15/2026 2:47 PM
AI agent that supports financial controllers during the closing cycle by monitoring task status, evaluating dependencies,...
1 Agent
PO Confirmation Email Automation
Last updated 09/29/2026 3:36 PM
Automates processing of supplier purchase order confirmation emails by using an AI agent to extract confirmation data and an n8n...
1 Agent
2 MCP Servers
Collection Email Drafting Agent
Last updated 09/23/2026 8:02 AM
AI agent that drafts personalized collection emails based on customer payment history and outstanding invoice items from SAP...
1 Agent
2 MCP Servers
Purchase Orders Viewer
Last updated 09/22/2026 1:30 PM
A CAP application that displays the last 5 purchase orders from S/4HANA
Customer Order Inbox Automation
Last updated 09/17/2026 1:02 PM
Monitors email inbox for customer orders, creates sales orders in SAP S/4HANA to determine delivery dates, and automatically...
1 Agent
1 MCP Server
Conversations
Spaces
Joule Studio
Notifications
Dietmar Steinbichler
Collapse
Solution Progress
Intent Summary
Requirements
Specifications
Solution Artifacts
financial-closing-advisor-agent
Testing
Overview
Preview with Mock Data
financial-closing-advisor-agent
File Tree
Intent
Update Solution
Financial Closing Advisor Agent
AI agent supporting financial controllers during the financial closing cycle.
Business challenge
Financial controllers need to monitor, prioritise, and sequence all closing tasks and workflows during the financial closing cycle. Currently this is a manual, error-prone process: controllers must track task status, identify blockers, evaluate dependency chains, and determine the correct next actions themselves — often across many entities and ledgers. The agent automates this monitoring, recommends the next activities to action (in sequence with reasoning), identifies root causes of blocked tasks, and supports configuration of closing calendars, ownership, escalation thresholds, and deadline alerts.
Business Goals & Success Criteria
Metric
Baseline
Target
Timeline
Process / Capability
Source
Financial close duration
10 days
7 days
Per close cycle
Local & Group Financial Closing
user
Key Milestones
1. Task Status Retrieved — Agent successfully fetches current status of all closing activities from SAP Advanced Financial Closing for a given close cycle.
2. Dependency Chain Evaluated — Agent maps and evaluates the full dependency graph, identifying completed, in-progress, blocked, and pending tasks.
3. Next Actions Recommended — Agent delivers a prioritised, sequenced list of recommended next actions to the controller with clear reasoning.
4. Blocker Root Cause Identified — Agent identifies root causes of blocked tasks (errors, missing data, incomplete upstream activities) and suggests corrective actions.
5. Alert & Escalation Triggered — Agent detects deadline proximity breaches or escalation threshold violations and notifies the relevant task
Intent.md

AI Sales Assistant Draft

Develop Manage

Deploy Actions

Solution Progress

- Intent Summary
- Requirements
- Specifications
- Solution Artifacts

File Tree

Intent

Generate Requirements

recommendation	Agent			

Key findings

- Three SAP S/4HANA Cloud Sales Order APIs are available and will serve as the primary data source for order history and pricing simulation.
- SAP Sales Cloud pipeline and pricing APIs are **unavailable** in the current landscape; the agent will derive pipeline context from S/4HANA order data.
- No MCP servers exist for the identified APIs; custom MCP translation files must be generated from the EDMX/OpenAPI specs.
- SAP S/4HANA Cloud natively supports Sales Price Calculation (SC1036/SC5494) and Sales Order Management, providing a strong data foundation.
- The core gap is the AI reasoning layer for discount recommendation — addressed by a custom Python agent.
- The solution keeps the SAP core clean by adding intelligence as an external agent layer.

Recommendations

AI-Powered Discount Strategy Agent

Executive Summary

Python A2A agent connecting to S/4HANA Cloud APIs for smart discount guidance.

Recommended Solution

A pro-code Python agent implementing the A2A protocol. The agent integrates with SAP S/4HANA Cloud via MCP tools (built from available OData APIs) to retrieve customer order history, run pricing simulations, and analyze margin context. An LLM reasoning layer synthesizes this data into a contextualized discount recommendation with justification, surfaced directly to the sales rep via a conversational interface.

Recommended solution category

AI Agent

Intent fit

88%

Intent.md

AI Sales Assistant

Challenge: AI agent for sales reps that analyzes orders, pipeline & margin targets from SAP S/4HANA Cloud, delivering contextualized discount recommendations. **Goals:** +10% deal win rate within 12 months (baseline: TBD). **Solution:** AI Agent (Python/A2A) integrating with SAP S/4HANA Cloud Sales Order APIs. **Fit Gap:** 3 S/4HANA Sales Order APIs available; SAP Sales Cloud pipeline/pricing APIs unavailable in landscape — agent will derive context from available S/4HANA data. No MCP servers found; custom MCP translation files required.

Proceeding to write `intent.md` .

Write File 332ms

`intent.md` has been saved. Now proceeding to the Product Requirements Document phase.

Activate Skill 194ms

Read File 129ms

Current Date 0ms

Quick PRD Confirmation: **Solution:** AI Agent | **Purpose:** Python A2A agent that gives sales reps contextual discount recommendations based on SAP S/4HANA Cloud order, pricing & margin data. **Personas:** Sales Representative, Sales Manager **Core Requirements:** Query customer order history; analyze margin & pricing; generate discount recommendation with rationale; support follow-up questions; guardrails against over-discounting **Architecture:** Python AI Agent (A2A) + MCP tools (from 3 S/4HANA Cloud OData APIs) + LLM via SAP Generative AI Hub **Milestones (5):** Context gathering → Order history retrieval → Margin & pipeline analysis → Discount recommendation generation → Rep action **Metrics:** +10% deal win rate within 12 months

Proceeding to write `product-requirements-document.md` .

Write File

I'm defining the business goal of a 10% deal win rate improvement over 12 months tied to the Lead to Cash discount optimization process, along with five milestones...

Message Joule...

Joule uses AI. Verify results.

Solution Progress

Intent Summary

Requirements

Specifications

Solution Artifacts

File Tree

Requirements

Generate Specifications

- **Achieved when:** Customer ID and/or opportunity context is successfully identified from the user's input.
- **Log on achievement:** `M1.achieved: context gathered — customer and product context extracted from rep query`
- **Log on miss:** `M1.missed: context gathering failed — insufficient information to identify customer or product`

M2: Order History Retrieval

- **Description:** The agent retrieves the customer's historical order and pricing data from SAP S/4HANA Cloud.
- **Achieved when:** At least one relevant sales order record is successfully returned from the API.
- **Log on achievement:** `M2.achieved: order history retrieved — N orders fetched for customer`
- **Log on miss:** `M2.missed: order history retrieval failed — no orders returned or API error`

M3: Margin & Pipeline Analysis

- **Description:** The agent analyzes margin targets and simulates pricing for the customer/product context.
- **Achieved when:** Pricing simulation returns a result and margin floor is determined.
- **Log on achievement:** `M3.achieved: margin analysis complete — pricing simulated and margin floor identified`
- **Log on miss:** `M3.missed: margin analysis incomplete — simulation failed or margin data unavailable`

M4: Discount Recommendation Generation

- **Description:** The agent reasons over the collected data and generates a contextualized discount suggestion with rationale.
- **Achieved when:** The LLM produces a discount recommendation with a justification that passes policy guardrail checks.
- **Log on achievement:** `M4.achieved: discount recommendation generated — X% recommended with rationale`
- **Log on miss:** `M4.missed: recommendation generation failed — guardrail violation or LLM error`

M5: Rep Action

- **Description:** The sales rep reviews the recommendation and takes action (applies, adjusts, or escalates).
- **Achieved when:** The agent delivers the final recommendation to the rep without error.
- **Log on achievement:** `M5.achieved: recommendation delivered to sales rep`
- **Log on miss:** `M5.missed: recommendation delivery failed — session error or timeout`

Requirements.md

AI Sales Assistant

Activate Skill 194ms

Read File 129ms

Current Date 0ms

Quick PRD Confirmation: **Solution:** AI Agent | **Purpose:** Python A2A agent that gives sales reps contextual discount recommendations based on SAP S/4HANA Cloud order, pricing & margin data. **Personas:** Sales Representative, Sales Manager **Core Requirements:** Query customer order history; analyze margin & pricing; generate discount recommendation with rationale; support follow-up questions; guardrails against over-discounting **Architecture:** Python AI Agent (A2A) + MCP tools (from 3 S/4HANA Cloud OData APIs) + LLM via SAP Generative AI Hub **Milestones (5):** Context gathering → Order history retrieval → Margin & pipeline analysis → Discount recommendation generation → Rep action **Metrics:** +10% deal win rate within 12 months

Proceeding to write `product-requirements-document.md` .

Write File 361ms

`product-requirements-document.md` saved. Proceeding directly to the specification phase.

Activate Skill 187ms

Now I'll read the agent specification template and guidelines, then download the API specs — all in parallel.

Read File 151ms

Now I'll download the API specs and copy the guidelines files in parallel.

Execute Command 123ms

- `sap.s4:apiResource:CE_SALESORDER_0001:v1` (OData) - prefer EDMX

Message Joule...

Joule uses AI. Verify results.



AI Sales Assistant

Draft

Develop | Manage

Deploy | Actions

Solution Progress

- Intent Summary ✓
- Requirements ✓
- Specifications ✓
- Solution Artifacts

File Tree

Search files

- specification
  - discount-strategy-agent
    - api-specs
      - API_SALES_ORDER_SIMULA...
      - API_SALES_ORDER_SRV.edmx
      - CE_SALESORDER_0001.edmx
    - specification.md
  - guidelines-agent-mcp.md
  - guidelines-agent-python.md
  - guidelines-agent-skills.md
  - guidelines-agent.md
  - guidelines.md
  - specification.md
- intent.md
- product-requirements-document.md

Specification

Generate Solution

AI Sales Assistant

2 steps

Guidelines: Read guidelines.md before executing ANY tasks below.
Check off items as completed.

Solution Setup

- Create asset directory: `mkdir -p assets/discount-strategy-agent/`
- Invoke `setup-solution` skill to create `solution.yaml` and `asset.yaml` files for the agent and all generated MCP server assets
- Validate all `asset.yaml` and `solution.yaml` files exist and are well-formed

Asset Implementation

Specification.md

AI Sales Assistant

Execute Command 124ms

Now write the per-asset specification:

Write File 383ms

Everything is in place. Here's a summary of what was built:

What was generated:

`intent.md` — Business challenge mapped to SAP's Lead to Cash process (Order to Fulfill), fit-gap analysis across 3 available S/4HANA APIs, and an AI Agent recommendation with 88% intent fit.

`product-requirements-document.md` — 5 must-have requirements (order history query, margin analysis, discount recommendation, conversational Q&A, guardrails), 5 instrumented milestones, and full agent architecture definition.

`specification/` — Ready-to-execute spec for the `discount-strategy-agent`:

- 3 SAP S/4HANA Cloud API specs downloaded (Sales Orders, Sales Order Details, Sales Order Simulation — ~610KB total)
- MCP tool scopes defined for all 3 APIs
- Guardrail: configurable `DISCOUNT_POLICY_CEILING_PCT` env var
- 5 OpenTelemetry milestones (M1–M5) wired to business steps
- Test requirements covering all tools + end-to-end + guardrail scenarios

Key design decisions:

- SAP Sales Cloud pipeline APIs are unavailable in the connected landscape — order/pricing context is derived entirely from S/4HANA Cloud
- All API access goes through MCP tools (no direct HTTP calls)
- The agent is read-only; no write operations to any SAP system

To build the agent, just say: **"execute specification/specification.md"**

Message Joule...

Joule uses AI. Verify results.

AI Sales Assistant

Draft

Develop | Manage

Deploy | Actions

Solution Progress

- Intent Summary
- Requirements
- Specifications
- Solution Artifacts
  - discount-strategy-agent
  - sap-s4-sales-order-simulation-srv-mcp-server
  - sap-s4-sales-order-srv-mcp-server
  - sap-s4-ce-salesorder-0001-mcp-server

Testing

Overview

Preview with Mock Data

discount-strategy-agent

File Tree

Search files

- assets
  - discount-strategy-agent
    - app
      - mcp_providers
      - skills
      - __init__.py
      - agent_executor.py
      - agent.py

Testing Overview

test_log_mode_with_llm_detection	0.0012	Passed
test_detection_disabled_skips_all	0.0007	Passed
test_block_mode_value	0.0002	Passed
test_log_mode_value	0.0002	Passed
test_scan_result_creation	0.0002	Passed
test_empty_content_returns_not_suspicious	0.0003	Passed
test_detects_suspicious_content	0.0009	Passed
test_passes_safe_content	0.0012	Passed
test_fails_open_on_llm_error	0.0008	Passed
test_handles_long_content	0.0009	Passed
test_wrap_async_tool_scans_results	0.0119	Passed
test_wrap_async_tool_passes_clean_content	0.0012	Passed
test_wrap_sync_tool_scans_results	0.0019	Passed
test_wrap_sync_tool_passes_clean_content	0.0014	Passed
test_wrapped_tool_preserves_metadata	0.0007	Passed
test_wrapped_tool_respects_detection_disabled	0.001	Passed
test_wrapped_tool_handles_none_result	0.0011	Passed
test_guardrail_flags_excessive_discount	0.0066	Failed
test_system_prompt_contains_policy_ceiling	0.0002	Passed
test_system_prompt_instructs_guardrail	0.0002	Passed
test_agent_without_tools_still_responds	0.0016	Failed

AI Sales Assistant

Write File 584ms

Execute Command 3s

All unit tests pass. Now write the integration tests:

Write File 366ms

Execute Command 5s

Now write the guardrail test:

Write File 435ms

Execute Command 5s

All tests pass. Now run the full test suite:

Execute Command 1m 32s

There are errors with the `asyncio.get_event_loop()` approach in the full test run (event loop reuse issue). Let me check what the actual error is and fix the fixture pattern:

Execute Command 2s

The error is an event loop conflict when the full suite runs (asyncio_mode=auto + get_event_loop). I need to convert the fixtures to use `pytest.mark.asyncio` or `asyncio.new_event_loop()`. Let me fix all test files to use `asyncio.run()`:

Write File

Let me check the translation.json structures quickly to understand the serverInfo:

Message Joule...

Joule uses AI. Verify results.

AI Agent Discovery Inbox

Search
Status
Classification
Environmer

Total 190 results | In Inventory | Not in Inventory | Linked

Discovery Item	Classification	Status
AI Sales Assistant	AI Agent	Action Needed
Goods Receipt	MCP Server	Action Needed
Procurement Invoice Assistant	AI Agent	Action Needed
Claude 4.6 Opus	AI Model	Action Needed

AI Sales Assistant

Action Needed

Overview | Details | Analytics | Changelog

Usage

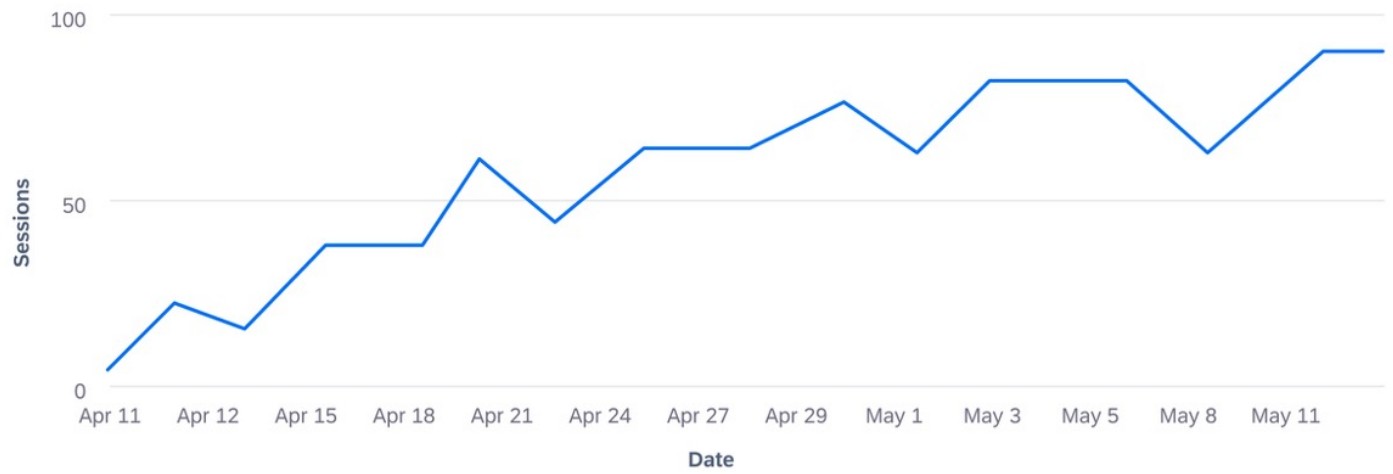
Monitor Agent Health

SAP Cloud ALM • Last 30 Days • Synced 6h ago

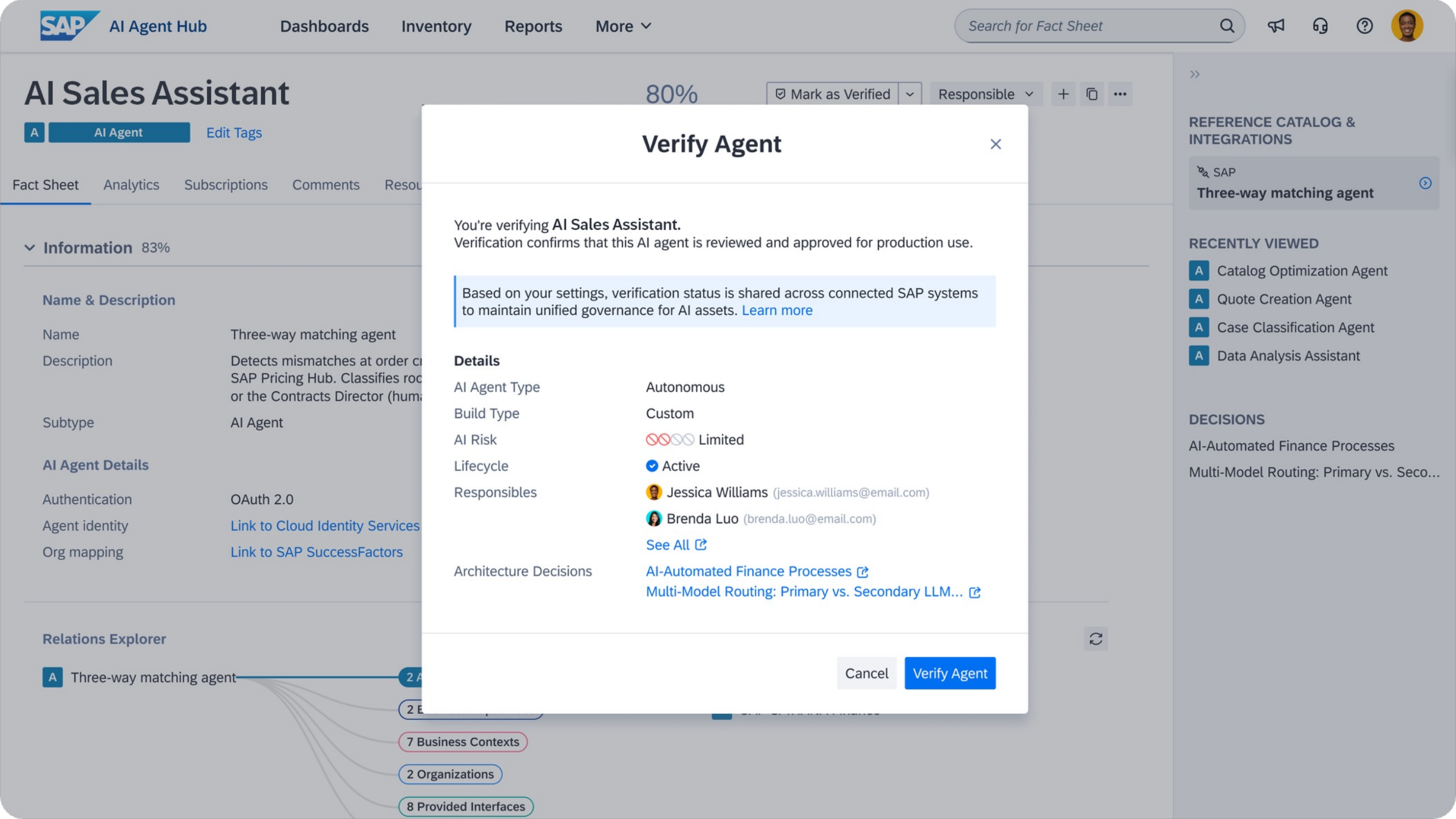
Total Sessions	First Active	Last Active	Average Tokens per...
725	2026-02-14	2026-05-11	2365
last 30 days	2 months ago	1 day ago	825 input / 1540 output

Sessions Over Time

Daily sessions over the last 30 days



Reject
Add to Inventory


AI Agent Hub
Dashboards
Inventory
Reports
More
Search for Fact Sheet
AI Sales Assistant
80%
Mark as Verified
Responsible
AI Agent
Edit Tags
Fact Sheet
Analytics
Subscriptions
Comments
Information 83%
Name & Description
Name
Three-way matching agent
Description
Subtype
AI Agent
AI Agent Details
Authentication
OAuth 2.0
Agent identity
Link to Cloud Identity Services
Org mapping
Link to SAP SuccessFactors
Relations Explorer
Three-way matching agent
7 Business Contexts
2 Organizations
8 Provided Interfaces
REFERENCE CATALOG & INTEGRATIONS
SAP
Three-way matching agent
RECENTLY VIEWED
Catalog Optimization Agent
Quote Creation Agent
Case Classification Agent
Data Analysis Assistant
DECISIONS
AI-Automated Finance Processes
Multi-Model Routing: Primary vs. Seco...
Verify Agent
You're verifying AI Sales Assistant.
Verification confirms that this AI agent is reviewed and approved for production use.
Based on your settings, verification status is shared across connected SAP systems to maintain unified governance for AI assets. Learn more
Details
AI Agent Type
Autonomous
Build Type
Custom
AI Risk
Limited
Lifecycle
Active
Responsibles
Jessica Williams (jessica.williams@email.com)
Brenda Luo (brenda.luo@email.com)
See All
Architecture Decisions
AI-Automated Finance Processes
Multi-Model Routing: Primary vs. Secondary LLM...
Cancel
Verify Agent

AI Sales Assistant 80%

Agent Verified | Responsible

A AI Agent Edit Tags

Fact Sheet | Analytics | Subscriptions | Comments | Resources | Last Update (2 days ago)

Usage

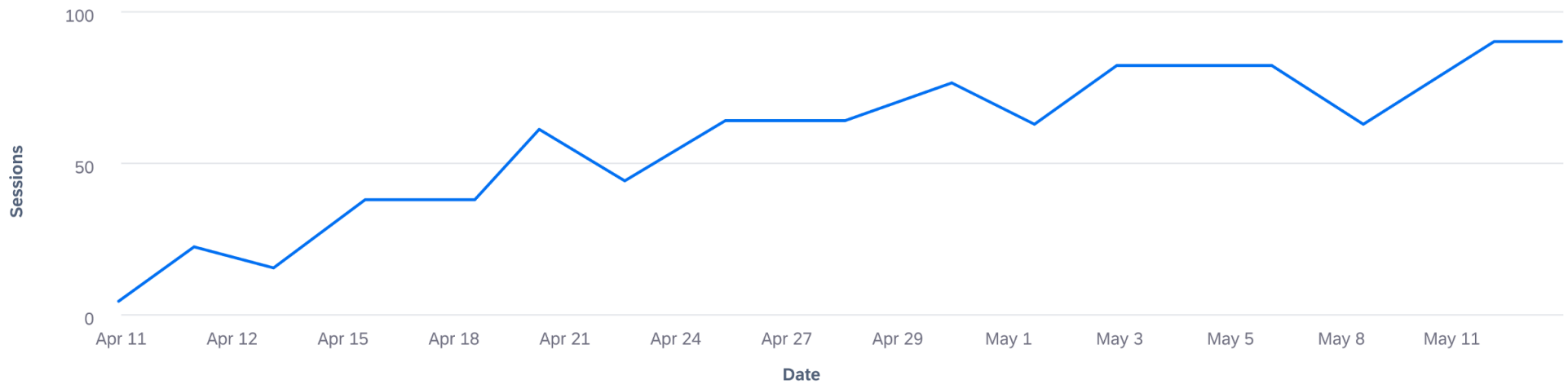
Monitor Agent Health

SAP Cloud ALM • Last 30 Days • Synced 1h ago

Total Sessions	First Active	Last Active	Average Tokens per Session
725	2026-02-14	2026-05-11	2365
last 30 days	10 days ago	Today	825 input / 1540 output

Sessions Over Time

Daily sessions over the last 30 days



REFERENCE CATALOG & INTEGRATIONS

SAP
AI Sales Assistant

RECENTLY VIEWED

- A Catalog Optimization Agent
- A Quote Creation Agent
- A Case Classification Agent
- A Data Analysis Assistant

DECISIONS

AI-Automated Finance Processes

Multi-Model Routing: Primary vs. Seco...

Joule Studio: Commercial model

Prerequisite

AI units or Joule Base ($0 SKU) for **access to Joule Work**

SAP BTPEA, CPEA, or pay as you go for **design-time tooling and runtime usage for Joule Studio**

Joule Work

Free

Access layer for shared experiences, approvals, dev spaces, SAP AI Agent Hub, MCP builder, and SAP BTP–hosted apps and extensions

- A unified development entry point
- Prebuilt content and guided experiences
- Built-in visibility and lifecycle support

Premium spaces and experiences planned for Joule Work could have a price point in the future

Design-time tooling

Free

Tooling for building and testing agents, apps, and workflows, with AI-assisted development under fair-use limits

- Build apps, agents, and workflows through intent-based development (IBD).
- Use low-code and pro-code tools .
- Test AI-enabled solutions.
- Use business context and enterprise knowledge.

Usage of AI capabilities for IBD is expected to have a price point beyond the fair use limits.

Joule Studio runtime

Paid

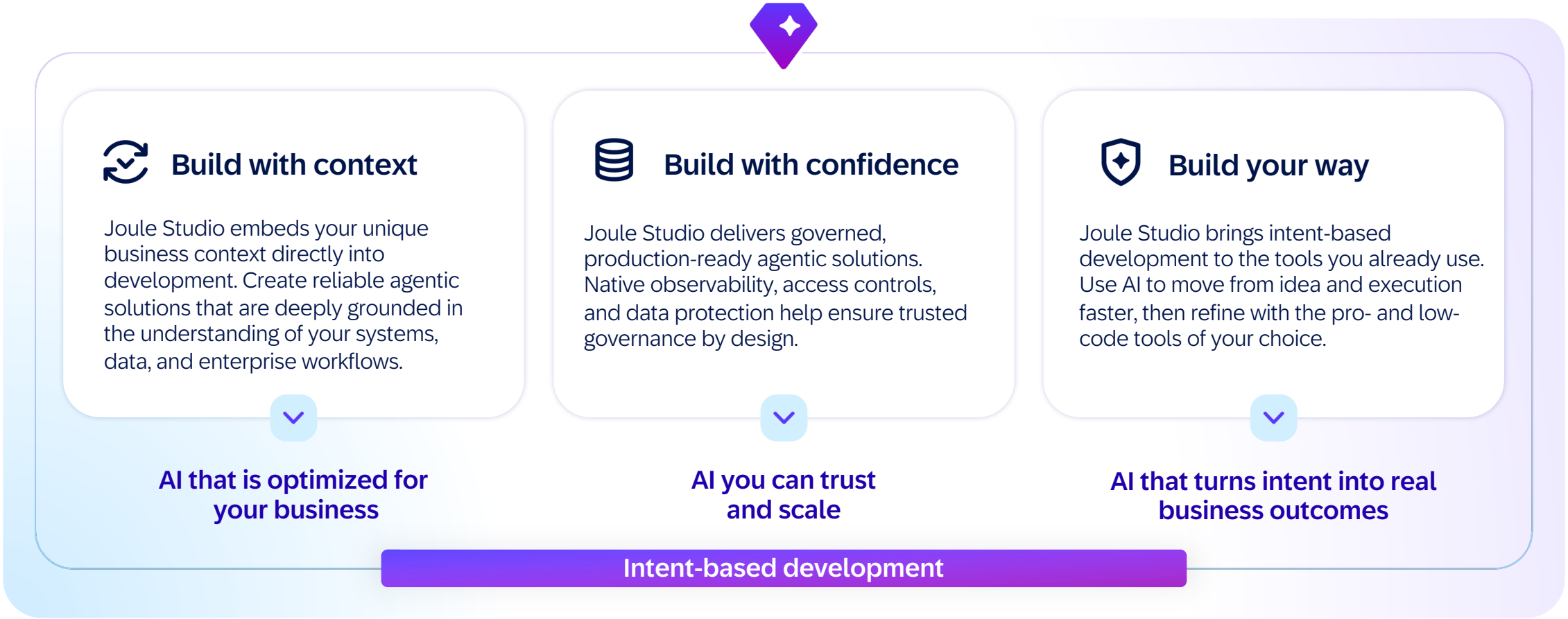
Productive runtime for customer-built agents, apps, workflows, and extensions, billed on resource consumption across compute, storage, transactions, and AI tokens

- Run apps, agents, and workflows.
- Provide logging, monitoring, and auditability.
- Govern runtime operations with built-in security controls.
- Provide runtime data storage and persistence.
- Connect data and models.
- Extend enterprise runtime capabilities.

Under a time-bound promotion, running custom agents will remain free through December 2026 across both BAF and Joule Studio runtime

Build trusted agentic solutions with Joule Studio

Joule Studio provides a unified, AI-first development environment for creating agents, apps, and workflows



INTERNAL – SAP and Customers Only

What do I do now?

Try Joule Studio

Join us at the SAP Booth for a hands-on experience!

Get certified for free

Start your Joule Studio journey or scan the code!

learning.sap.com/joule-studio-certification

Thank you.

© 2026 SAP SE or an SAP affiliate company. All rights reserved. See Legal Notice on www.sap.com/legal-notice for use terms, disclaimers, disclosures, or restrictions related to this material.