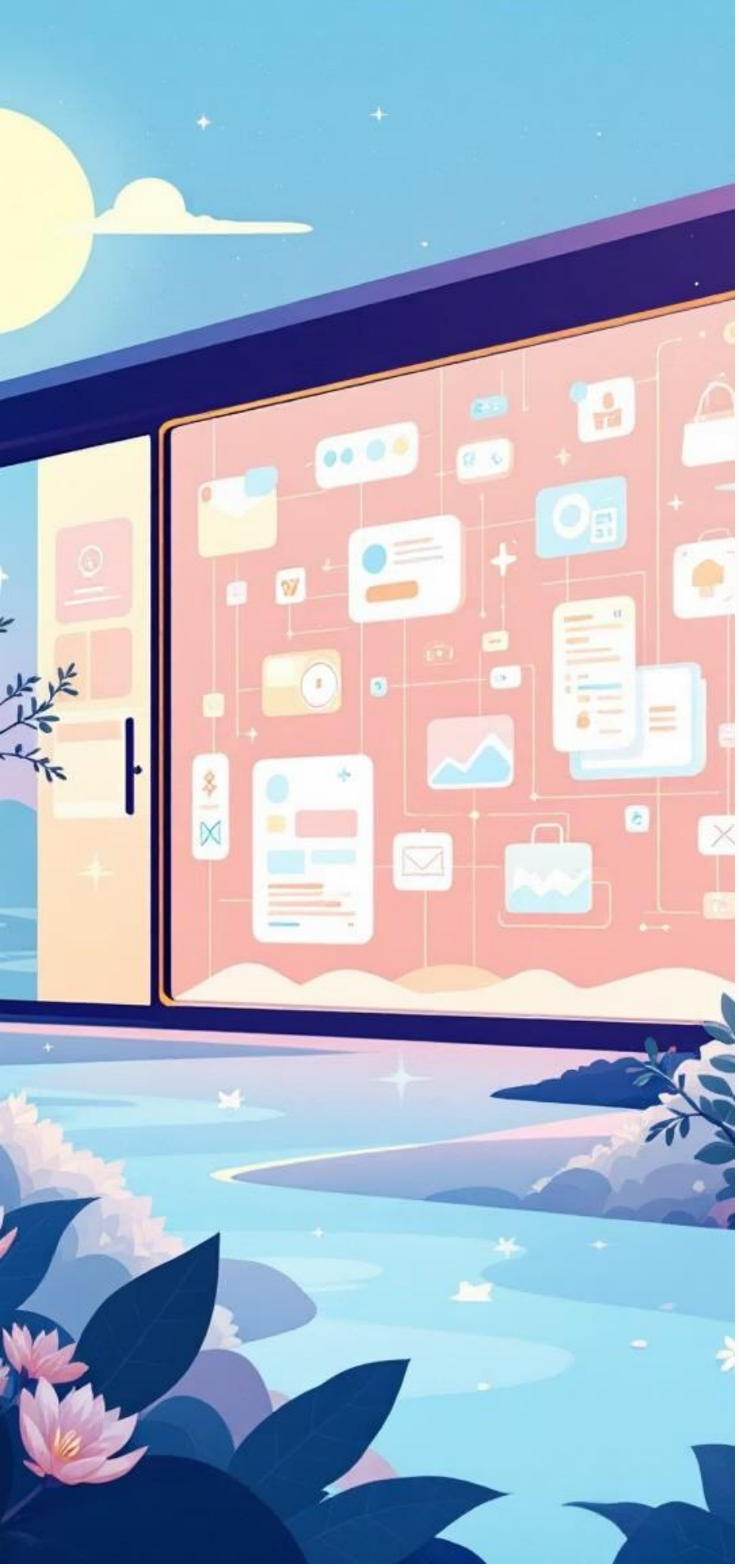
Universal Commerce Protocol (UCP) – Floral Boutique

AI-Native E-Commerce with MCP Integration



Project Overview

What We Built

Demonstrating AI-native commerce with UCP and MCP integration for AI shopping agents. Features complete checkout session management with production-grade security patterns.

Demo Store

Interactive flower shop showcasing Red Roses ($2.99), Pink Tulips ($1.99), and White Lilies ($3.99). Users can browse and purchase through natural conversation with AI agents.

Technology Stack

Built with Node.js, Express, and MCP SDK v1.0.4.

Implements modern rate limiting, idempotency patterns, and comprehensive input validation for secure operations.

Important: This is a POC for educational purposes only. Uses in-memory storage and demo payment processing—not production-ready.

System Architecture

Four-layer architecture enabling seamless AI-to-commerce communication through standardized protocols.

1

Layer 1: AI Agent

Claude Desktop interprets natural language requests like "Buy 12 red roses" and orchestrates the shopping experience.

2

Layer 2: MCP Server

Five specialized tools handle commerce operations: browse, create, update, get, and complete checkout sessions.

3

Layer 3: UCP Server

REST API with /.well-known/ucp discovery endpoint exposes standardized commerce capabilities to AI agents.

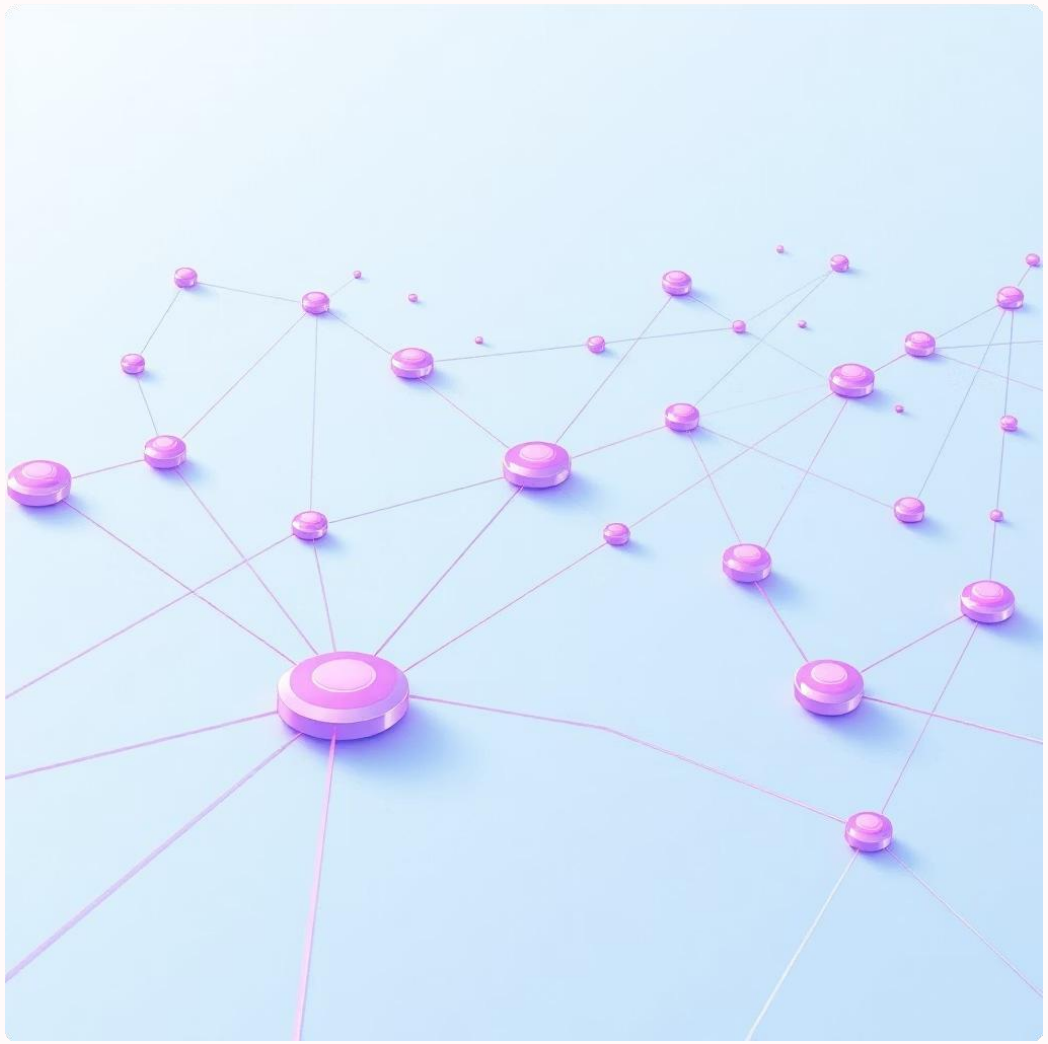
4

Layer 4: Data Layer

In-memory storage managing sessions, products, and idempotency cache with automatic expiration handling.

Understanding UCP

Universal Commerce Protocol



The Web Standard for AI Shopping

UCP is an open standard that enables AI agents to interact with any e-commerce system using one unified protocol—just like HTTP revolutionized web browsing.

The discovery endpoint at `/.well-known/ucp` returns available services, payment handlers, and platform capabilities, allowing AI agents to automatically understand how to shop at any compliant store.

One Protocol

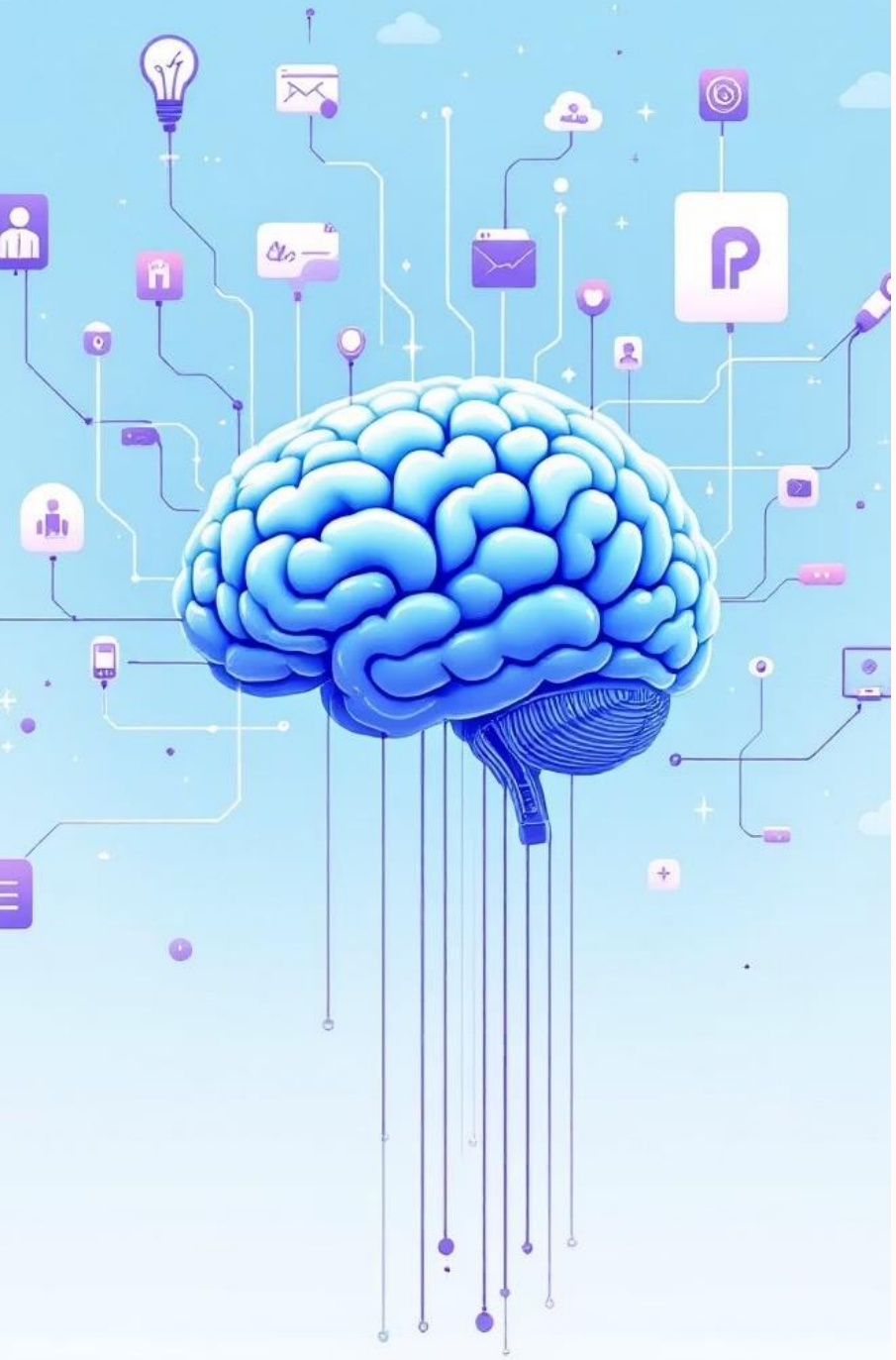
AI agents learn once, shop everywhere with consistent interfaces

Open Standard

Version 2026-01-11 specification ensures interoperability

Future-Ready

Built for autonomous AI commerce at scale



Understanding MCP

Model Context Protocol

MCP is the bridge that allows Claude and other AI models to interact with external services through registered tools. It transforms AI from a chat interface into an action-capable agent.

01

Register Tools

Define capabilities like browse_products, create_checkout, and complete_checkout with clear schemas

02

AI Calls Tools

Claude analyzes user intent and autonomously selects appropriate tools to fulfill requests

03

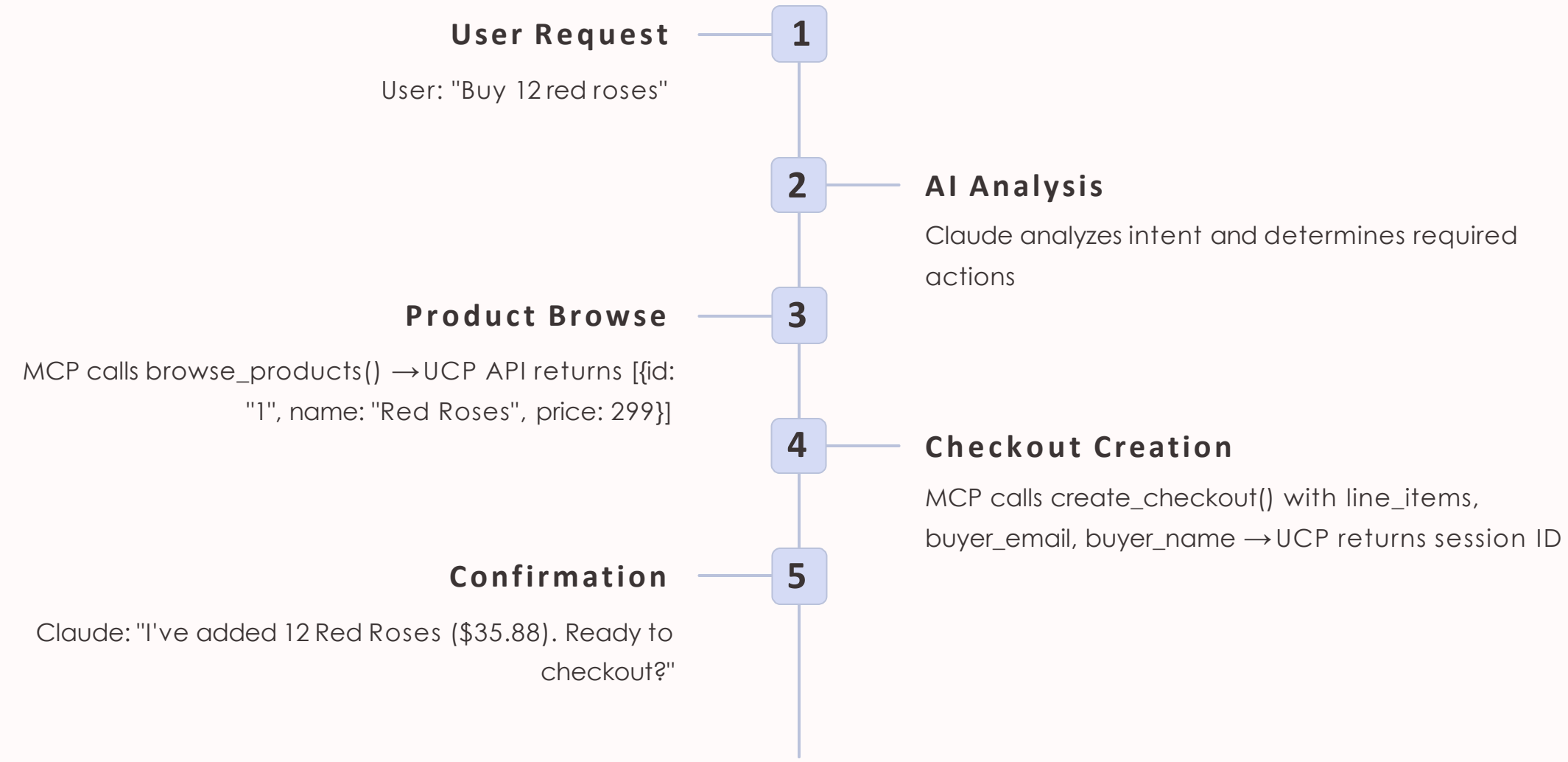
Process Results

Responses are interpreted and presented naturally in conversation flow

> "User asks 'buy flowers' → Claude calls browse_products → Shows beautiful results in natural language"

How MCP and UCP Work Together

The synergy between Model Context Protocol (HOW) and Universal Commerce Protocol (WHAT) creates seamless AI shopping experiences.



Key Insight: MCP provides the `tool interface`, while UCP defines the `commerce standard`. Together, they enable truly autonomous AI shopping.

Checkout Session Lifecycle

Intelligent state management ensures data integrity and seamless progression through the purchase flow.

Session Created

New checkout session initialized with unique ID and timestamp

Incomplete Status

Missing required fields: buyer email, name, or line items

Ready for Complete

All requirements met: email ✓name ✓items ✓

Completed

Order finalized with confirmation ID

Session Rules

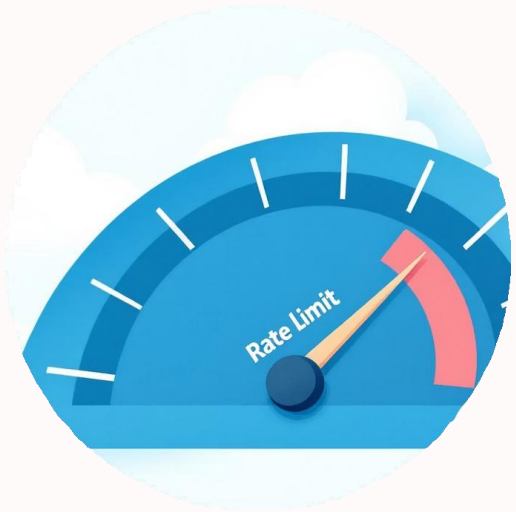
- 6-hour expiration window
- Automatic cleanup every minute
- Status validated at each transition

State Transitions

Sessions can only progress forward through states, ensuring data consistency and preventing invalid operations.

Production-Grade Security

Six Layers of Protection



Rate Limiting

100 requests per 15 minutes per IP address prevents abuse and ensures fair resource allocation

Idempotency

Idempotency-key header with 1-hour cache prevents duplicate operations and accidental double charges

Input Validation

Quantity limits (1-100), non-empty arrays, product existence checks ensure data integrity

Sanitization

Email regex validation with lowercase normalization, name length limits (max 100 chars)

Session Management

6-hour expiration, automatic cleanup routines, expiry checks on every access

UCP Headers

UCP-Agent tracking and request-id correlation for debugging and audit trails

Core API Implementation

RESTful endpoints following UCP specification with comprehensive metadata and validation.

Discovery

`GET /.well-known/ucp`

Returns UCP configuration, available services, payment handlers, and platform capabilities

Products

`GET /api/products`

Lists available flowers with pricing, descriptions, and inventory status

Create Checkout

`POST /api/checkout-sessions`

Initializes session with line_items and buyer details (email, full_name)

Update Checkout

`PUT /api/checkout-sessions/:id`

Modifies buyer information or line items before completion

Complete

`POST /api/checkout-sessions/:id/complete`

Finalizes purchase with payment_data and creates order confirmation

All responses include comprehensive UCP metadata: status codes, validation messages, and calculated totals (subtotal, 8% tax, final total).

Summary & Next Steps

✓ What This Demonstrates

- Complete UCP v2026-01-11 protocol implementation
- MCP integration patterns with 5 shopping tools
- Production-grade API security (rate limiting, validation, sanitization)
- Intelligent session management with lifecycle tracking
- Standards compliance and best practices
- AI-native conversational shopping experience

⚠ Important Limitations

- NOT production ready—educational POC only
- In-memory storage (data lost on restart)
- Demo payment handler without real processing
- Missing: Database persistence, authentication, inventory management, fulfillment systems, shipping integration, discount logic

---

Learn More & Connect

Official Specifications

📚 **UCP Specification v2026-01-11** | **Checkout Documentation**

📚 **UCP Shopping Service OpenAPI**

📚 **Model Context Protocol** | **MCP CitHub**

Project Resources

💻 **CitHub Repository** - Full source code, documentation, and setup guide

🔧 **Tech Stack:** Node.js, Express.js, @modelcontextprotocol/sdk v1.0.4, express-rate-limit v7.1.5

Connect & Discuss

💼 **LinkedIn: Harish Kumar**

📧 **harish.p@etg.digital**

📥 Download this presentation in PDF format from LinkedIn