UCP Agent for Magento

User Guide — Universal Commerce Protocol endpoints for Magento Open Source & Adobe Commerce

Version 0.1.3 · September 2026 · Package `ooasis/module-magento-agent`

Documentation & source: github.com/ooasis/module-magento-agent

1. Overview

UCP Agent makes your Magento store buyable by AI shopping agents. Once enabled, your store publishes a *Universal Commerce Protocol* (UCP) business profile at `/.well-known/ucp` and serves the UCP shopping API: agents can price products, build checkout sessions, apply your existing coupon codes, select shipping, complete payment, and receive signed order-status webhooks.

The extension is entirely self-hosted. No data is routed through the developer or any third-party service; the only outbound calls are payment processing to your own Stripe account (if configured) and the profile/webhook exchanges with the agent platform that is making a purchase, as defined by the UCP specification.

Completed agent checkouts become **real Magento orders** in Sales → Orders, placed under a dedicated "UCP Agent Payment" method, so your existing fulfillment, invoicing, and reporting workflows are unchanged.

2. Requirements

- Magento Open Source 2.4.6 or later (tested on 2.4.7). Adobe Commerce should work but is not yet certified.
- PHP 8.1, 8.2, or 8.3 with the `openssl` and `sodium` extensions (both standard).
- HTTPS on your storefront (agents require it in practice).
- Optional, for unattended agent payments: a Stripe account.

3. Installation

```
composer require ooasis/module-magento-agent
bin/magento module:enable Ucp_Agent
bin/magento setup:upgrade
bin/magento cache:flush
```

In production mode also run `bin/magento setup:di:compile` and redeploy static content as part of your usual release procedure. The module adds two database tables (`ucp_agent_session`, `ucp_agent_idempotency`) and no cron jobs.

4. Configuration

Go to **Stores** → **Configuration** → **Services** → **UCP Agent**. All settings are store-view scoped.

General

Setting	Default	Description
Enable UCP Endpoints	No	Master switch. While disabled, every UCP endpoint (including `/.well-known/ucp`) returns 404. On first enablement the module generates an ES256 signing key pair automatically and stores the private key encrypted with your Magento crypt key.
Require Signed Requests	No	Strict mode. When enabled, requests without an RFC 9421 signature are rejected with 401. When disabled, signed requests are still fully verified — unsigned ones are merely allowed through, as the UCP specification permits.
Simulation Secret	empty	**Test environments only.** Setting a value activates test mode: the mock payment handler is advertised and accepted, and the `/testing/simulate-shipping` hook used by the UCP conformance suite is enabled (gated by this secret). Leave empty in production.

Stripe (agent payments)

Setting	Description
Secret Key	Your Stripe secret key (`sk_live_…` or `sk_test_…`). When present, the Google Pay (via Stripe) payment handler is advertised to agents and charges are processed as Stripe PaymentIntents on your account. Stored encrypted.
Publishable Key	Your Stripe publishable key; included in the handler configuration agents use to tokenize Google Pay instruments for your gateway.

> Without Stripe keys the store is still fully agent-visible: agents can build and price checkouts, and payment simply is not offered unattended. Payment credentials supplied by agents are never stored and never echoed back in API responses.

5. Verifying your setup

After enabling, fetch your profile from any terminal:

```
curl https://your-store.example/.well-known/ucp
```

You should see a JSON document with `ucp.version`, a `dev.ucp.shopping` service entry pointing at `https://your-store.example/ucp`, five capabilities (checkout, order, discount, fulfillment, buyer_consent), your payment handlers, and your published signing keys. That document is exactly what agent platforms read to discover your store.

6. How an agent purchase works

1. The agent reads `/.well-known/ucp` and opens a checkout session (`POST /ucp/checkout-sessions`) with items identified by **SKU**.
2. The module prices the session from your live catalog (price, stock), applies any coupon codes against your cart price rules, and computes shipping options from your enabled carriers once the agent supplies a destination.
3. The agent selects a shipping option; the session becomes `ready_for_complete`.
4. On completion the module charges the payment instrument (e.g. Google Pay via your Stripe account), places a real Magento order as a guest checkout, and returns the order reference to the agent.
5. Order status changes are pushed to the agent platform as cryptographically signed webhooks containing the full order entity.

In the admin, agent orders look like any other order. Each carries a status-history comment identifying the UCP checkout and, for Stripe payments, the transaction id.

Product coverage

Simple and virtual products are supported directly; configurable products are purchasable through their child SKUs; grouped products through their component simples. Bundle and downloadable products are not exposed to agents in this release.

Discounts

Agents may submit coupon codes from your existing cart price rules (percent-of-cart and fixed-amount actions). Codes are matched case-insensitively; unknown codes are ignored rather than treated as errors, per the specification.

7. Security model

- **Disabled by default.** Nothing is exposed until you enable it per store view.
- **RFC 9421 HTTP Message Signatures.** Signed requests are verified against the signing keys published in the requesting platform's own UCP profile (ES256 baseline, ES384 and Ed25519 also supported; Content-Digest per RFC 9530). Enable “Require Signed Requests” to reject unsigned traffic outright.
- **Merchant signing keys** are generated locally on first enablement (never transmitted anywhere), stored encrypted, and published as public JWKs with RFC 7638 thumbprint key ids.
- **Idempotency.** Mutating endpoints require an `Idempotency-Key` header; retries replay the stored response and conflicting reuse returns 409.
- **Outbound protections.** Profile fetches and webhook deliveries refuse private and internal network addresses.
- **Payment hygiene.** Credentials are used once for the charge and never persisted; the mock handler cannot be used unless test mode is explicitly configured.

8. Multi-store behavior

The profile, catalog prices, currency, and checkout math follow whichever store view the request's URL resolves to — standard Magento store resolution. Enable the module only on the store views you want agent-visible. One signing key pair serves the installation.

9. Troubleshooting

Symptom	Cause / fix
`/.well-known/ucp` returns 404	Module not enabled for that store view, or config cache stale — check the toggle and run `bin/magento cache:flush`.
Checkout completion fails with "Product … not available"	Stock indexes out of date — run `bin/magento indexer:reindex`.
Agents see no shipping options	The flat-rate and/or free-shipping carriers must be enabled (Stores → Configuration → Sales → Delivery Methods). This release computes agent-facing options from those two carriers.
Payment declined with "Stripe is not configured"	An agent attempted unattended payment without Stripe keys present — add keys under UCP Agent → Stripe, or expect agents to fall back to their platform's escalation flow.
401 `signature_missing`	Strict mode is on and the caller sent an unsigned request. Disable strict mode if you intend to accept unsigned traffic.
409 `IDEMPOTENCY_CONFLICT`	Expected behavior: an `Idempotency-Key` was reused with a different request body.

10. Uninstalling

```
bin/magento module:disable Ucp_Agent
composer remove ooasis/module-magento-agent
bin/magento setup:upgrade
```

Orders created by agents are ordinary Magento orders and remain intact. The `ucp_agent_*` tables may be dropped manually if you want a complete cleanup.

11. Support & privacy

Issues and questions: github.com/ooasis/module-magento-agent/issues
Privacy policy: github.com/ooasis/module-magento-agent/blob/main/PRIVACY.md
License: MIT