Syrve Integration

What the AI can do

Summary

Syrve Integration connects **Syrve** (POS/CRM platform for restaurants and delivery) with **Newo Platform**.

It enables:

- Creating food delivery orders programmatically through conversational AI
- Checking existing order and delivery status in real-time
- Caching and synchronizing restaurant menus (products, sizes, modifiers, pricing)
- Looking up customers by phone via Syrve loyalty system
- Multi-location order routing via terminal groups
- Fuzzy product matching from cached menu using AKB (Agent Knowledge Base)

This integration is designed for **restaurants and food delivery businesses** who need **automated order taking and status tracking through a conversational AI agent** (voice or chat).

Common use cases

- When a customer wants to place a delivery order → AI extracts order details from conversation, matches menu items, creates order in Syrve
- When a customer asks about order status → AI looks up customer by phone, retrieves orders and delivery info
- When a conversation starts → Menu is automatically refreshed if cache is outdated (24h default)
- When a customer mentions a menu item → Fuzzy search matches the item from cached menu with product ID and price

Features at a glance

Feature	Supported	Notes
Create Delivery Order	✅	LLM payload extraction + fuzzy menu matching
Check Order Status	✅	By customer phone → order ID → delivery status
Menu Synchronization	✅	Auto-cache with 24h TTL, products + sizes + modifiers
Customer Lookup	✅	Via Syrve loyalty by phone number
Terminal Group Routing	✅	Auto-selects first available terminal group
Fuzzy Product Matching	✅	AKB-based search, 0.6 threshold
Modifier Support	✅	Menu cached with modifier groups and prices
Order Context Injection	✅	Order info injected into prompt after creation
Table Reservation	❌	Not implemented
Payment Processing	❌	Not implemented
Order Modification	❌	Create-only, no edits after submission

Multi-Item Orders	✅	Supports array of order items

Before You Start

Before installation:

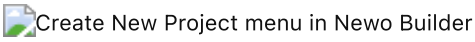
- Active **Syrve** account with API access
- **Syrve API key** (apiLogin credential)
- At least one organization and terminal group configured in Syrve
- Menu/nomenclature populated in Syrve

Setup

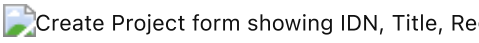
Create the Newo project

If this project already exists in **Newo Builder**, skip this subsection and continue with the integration-specific settings below.

1. In **Newo Builder**, open the projects list and click **Create Project** (or **Create New Project** from the top-right menu).

 Create New Project menu in Newo Builder

2. Fill **IDN** and **Title**. The exact names do not matter; use any clear names your team will recognize.
3. In **Registry**, choose the release channel:
   - **staging** — the newest module fixes appear here first. Use it when you need the latest fix, but expect possible unfinished changes.
   - **production** — the final stable version for live projects.
4. In **Module**, select the module for this integration.

 Create Project form showing IDN, Title, Registry, and Module fields 5. Leave **Module version** on **Latest version** unless support tells you to pin a specific version, then click **Create**.

3.1 Step 1 — Get API Key

1. Log in to your Syrve management console
2. Navigate to API settings and obtain your API key (apiLogin)
3. Note the API region (default: EU — `https://api-eu.syrve.live`)

3.2 Step 2 — Connect in Platform

1. Open **Newo → Projects**
2. Set the following attributes in **Builder / Attributes**:

Attribute	Required	Description
`syrve_api_key`	✅	Syrve API key (apiLogin)
`syrve_base_url`	❌	API base URL (default: `https://api-eu.syrve.live`)
`syrve_delivery_menu_update_interval`	❌	Menu cache TTL in seconds (default: `86400` = 24 hours)

`syrve_override_agent_attributes`	❌	Override SuperAgent attribute schemas (default: `True`)

3. Click **Save + Publish All**
4. On publish, the SetupFlow automatically:
    - Exchanges the API key for an access token
    - Discovers the organization ID
    - Prepares the connections for API and token operations
    - Registers order creation and status check tools with the agent

How to use the integration

This section explains how the integration works, how to configure automation, and how to test it.

4.1 How the Integration Works

The integration uses Syrve's REST API for order management, customer loyalty, and menu data. Orders are created through LLM-powered payload extraction from conversation context, with fuzzy matching against the cached menu. The menu is auto-refreshed on session start when the cache expires.

- A **Trigger** is a system event that starts a flow
- An **Action** is the API operation performed in Syrve

Where to test

Testing can be done through the **Newo conversation interface** (voice or chat). Each flow (CreateOrder, CheckOrders, CollectMenu) has a dedicated webhook test endpoint for connectivity validation.

How to test that everything works

To test the integration:

1. **Setup**: Publish the project and verify token exchange + organization discovery succeeds
2. **Menu**: Start a conversation — verify menu is synced (check `syrve_delivery_menu` attribute is populated)
3. **Create Order**: Say "I want to order a [menu item] to [address]" — verify order created in Syrve with correct items and pricing
4. **Check Status**: Ask "What's the status of my order?" — verify delivery status returned

If no action occurs:

- Ensure `syrve_api_key` is set correctly
- Verify `syrve_access_token` was obtained (check it's not empty after setup)
- Confirm `syrve_organization_id` was discovered
- Check that menu cache is populated (`syrve_delivery_menu` is not `[]`)
- Verify the menu was synced to AKB with label `syrve_menu`
- Re-publish the project if credentials were changed

> *Note: This integration creates real orders in Syrve. Orders submitted through the agent are sent to the live POS system with* `sourceKey: "voice-agent"` .

FAQ

Q: How does the agent match menu items from conversation? A: The integration uses a two-step process: (1) LLM extracts the item name from conversation memory, (2) fuzzy AKB search with `syrve_menu` label matches the name to cached products (threshold: 0.6). The matched product's ID and price are used for the order.

Q: How often is the menu refreshed? A: By default every 24 hours (`syrve_delivery_menu_update_interval = 86400` seconds). The cache is checked on every `session_started` event. If the timestamp exceeds the TTL, the menu is re-fetched from Syrve's nomenclature API.

Q: What happens if a menu item isn't found? A: The agent receives a `product_not_found` error, which it can communicate to the customer. Only items present in the Syrve nomenclature and cached in AKB can be matched.

Q: Does the integration support modifiers (e.g., extra cheese)? A: The menu cache includes full modifier group data (group names, required/optional flags, min/max counts, individual modifiers with prices). However, the current order creation flow sends an empty `modifiers: []` array — modifier selection is not yet implemented in the ordering pipeline.

Q: Which LLM models are used? A: The primary model is `gemini25_flash` (Google Gemini 2.5 Flash) for most operations. The CheckOrdersFlow's `_getContactSuccessSkill` uses `gpt4` (OpenAI) — this is the only skill in the integration that uses a different model.

Q: How does multi-location routing work? A: During order creation, the integration fetches terminal groups for the organization and automatically selects the first available terminal group. The `syrve_terminal_group_id` is stored and used for order routing.

Q: What customer information is required for ordering? A: First name, phone number (E.164 format), and delivery address are required. Last name is optional — if missing, the first name is copied to the last name field. The phone is used to look up the customer in Syrve's loyalty system.

Q: What is injected into the prompt after order creation? A: The full `orderInfo` object from the Syrve API response is injected as a custom prompt section (`ExistingOrderInfo role="context"`), allowing the agent to reference order details (ID, status, items) in subsequent conversation turns.