Kolla Integration

What the AI can do

Summary

Kolla Integration connects **Kolla** (unified dental API built on OpenDental EHR) with **Newo Platform**.

It enables:

- **Dynamic provider selection** based on treatment type (e.g., cleaning → Hygienist, filling → General Dentist)
- Checking real-time appointment availability based on provider schedule
- Booking dental appointments for both new and existing patients
- Cancelling existing appointments
- Automatic existing patient detection and appointment history retrieval during conversations
- Provider and operatory resource management

This integration is designed for **dental clinic administrators and front-desk teams** who need **automated appointment scheduling through a conversational AI agent** (voice or chat).

Common use cases

- When a patient asks about available slots → Dynamically resolve provider by treatment type, check provider schedule, subtract booked appointments, return free slots
- When a patient wants to book an appointment → Resolve appropriate provider, detect new vs existing patient, create appointment
- When a patient wants to cancel an appointment → Cancel appointment in Kolla with provider reference
- When a patient's phone number is identified → Automatically look up existing contact and their upcoming appointments
- When a conversation ends → Clean up patient context from the agent's prompt

Features at a glance

Feature	Supported	Notes
Check Availability	✅	Provider schedule-based algorithm
Dynamic Provider Resolution	✅	LLM matches treatment type to provider taxonomy (feature-flagged)
Book Appointment (Existing)	✅	Uses existing contact record, marked "(EP)"
Book Appointment (New)	✅	Creates contact + appointment atomically, marked "(NP)"
Cancel Appointment	✅	Via provider-referenced cancellation
Existing Patient Lookup	✅	By phone number with auto-normalization
Appointment History	✅	Retrieves upcoming appointments for existing patients

Dynamic Provider Selection	✅	Feature-flagged. Based on service request + provider taxonomy or custom criteria (Gen())
Custom Provider Criteria	✅	Optional provider:service mapping for fine-tuned doctor selection
Default Provider Fallback	✅	Configurable fallback when resolution disabled or not possible
Operatory Selection	✅	Configurable default operatory/room
Configurable Slot Duration	✅	Default 30 min, configurable via attribute
Booking Window	✅	Configurable days ahead (default: 7)
Multi-Location Support	❌	Single operatory per instance
Reschedule Appointment	❌	Cancel + rebook as workaround

Before You Start

Before installation:

- Active **Kolla** account with API access
- **Kolla API Token** (Bearer token)
- **Connector ID** (e.g., `opendental`)
- **Consumer ID** (e.g., `kolla-opendental-sandbox`)
- Kolla API base URL (default: `https://unify.kolla.dev/dental/v1/`)

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 — Obtain Credentials from Kolla

1. Log in to your **Kolla** account
2. Navigate to the API / Integrations section
3. Obtain the following:
   - **API Token** (Bearer token for authentication)
   - **Connector ID** (identifies the dental system connector, e.g., `opendental`)
   - **Consumer ID** (identifies your consumer instance)
4. Store all credentials securely

3.2 Step 2 — Connect in Platform

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

Attribute	Required	Description
`kolla_api_token`	✅	Bearer token for Kolla API authentication
`kolla_connector_id`	✅	Connector identifier (e.g., `opendental`)
`kolla_consumer_id`	✅	Consumer identifier (e.g., `kolla-opendental-sandbox`)
`kolla_base_url`	❌	API base URL (default: `https://unify.kolla.dev/dental/v1/`)
`kolla_timezone`	❌	Timezone for slot display (default: `America/Chicago`)
`kolla_booking_window`	❌	Days ahead for availability search (default: `7`)
`kolla_appointment_length`	❌	Default appointment duration in minutes (default: `30`)
`kolla_availability_slot_offset`	❌	Days offset from current date (default: `0`)
`kolla_default_provider`	❌	Fallback provider when dynamic resolution unavailable (auto-populated)
`kolla_default_operatory`	❌	Selected operatory/room for bookings (auto-populated)
`kolla_providers_catalog`	❌	JSON catalog of providers with taxonomy (auto-populated)
`kolla_operatories_list`	❌	JSON list of operatory names for room availability check (auto-populated)
`kolla_slots_available`	❌	Enable availability checking (default: `True`)
`kolla_booking_available`	❌	Enable booking capability (default: `True`)
`kolla_cancelation_available`	❌	Enable cancellation capability (default: `True`)

`kolla_enable_provider_selection`	❌	Enable AI-based provider selection (default: `False`). When disabled, always uses default provider. When enabled, LLM matches service request to the best provider.
`kolla_selection_provider_criteria`	❌	Custom provider-to-service mapping for AI selection. Format: `ProviderName:service1; AnotherProvider:service2`. Example: `Jones, Tina:cleaning, hygiene; Albert, Brian:filling, root canal`. If empty, AI uses provider taxonomy from catalog.

3. Click **Save + Publish All**
4. On publish, the SetupFlow automatically:
   - Validates the API credentials
   - Fetches available resources (providers and operatories) from Kolla
   - Populates provider and operatory selection dropdowns
   - Builds `kolla_providers_catalog` with taxonomy data for dynamic provider resolution
   - Builds `kolla_operatories_list` for room availability checking

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 operates based on **Triggers and Actions** via the event-driven system.

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

Where to test

Testing can be done through the **Newo conversation interface** (chat or voice channel) by interacting with the agent. Each flow also has a dedicated test skill triggered via `test_*` webhook events with sample data.

How to test that everything works

To test the integration:

1. **Setup**: Publish the project and verify logs show successful resource fetch and `kolla_providers_catalog` populated
2. **Dynamic Provider**: Ask for "teeth cleaning" — verify `resolved_provider` is a Hygienist (e.g., `provider_2`). Ask for "filling" — verify General (e.g., `provider_1`)
3. **Existing Client**: Start a conversation from a known phone number — verify patient context injected into agent prompt
4. **Availability**: Ask the agent "I need a cleaning this week" — verify slots are based on the resolved provider's schedule
5. **Booking (existing)**: Book with a known patient phone — verify appointment marked "(EP)" with correct provider
6. **Booking (new)**: Book with an unknown phone — verify contact + appointment created, marked "(NP)"

7. **Cancellation**: Cancel an appointment — verify status changed in Kolla

If no action occurs:

- Ensure `kolla_api_token`, `kolla_connector_id`, and `kolla_consumer_id` are all set correctly
- Verify `kolla_base_url` points to the correct Kolla API endpoint
- Confirm the integration was re-published after configuration changes
- Check that feature flags are enabled (`kolla_slots_available`, `kolla_booking_available`, `kolla_cancelation_available`)
- If provider selection is not working: verify `kolla_enable_provider_selection` is `True` and `kolla_providers_catalog` is populated
- If criteria-based selection is not working: verify `kolla_selection_provider_criteria` format — use semicolons between providers, colons between name and services
- Verify `kolla_default_provider`, `kolla_default_operatory`, `kolla_providers_catalog`, and `kolla_operatories_list` were populated during setup

> *Note: This integration performs real operations in Kolla/OpenDental. Appointments created or cancelled through the agent are reflected in the live system.*

FAQ

Q: Does the integration import historical appointments? A: No. Only appointments created after activation are managed. However, existing patient appointments are retrieved for context when a known phone number is detected.

Q: Can I connect multiple Kolla accounts? A: Each integration instance supports one set of API credentials (token, connector ID, consumer ID). For multiple accounts, create separate integration instances.

Q: How does the availability algorithm work? A: The integration: (1) resolves the appropriate provider based on treatment type via `Gen()`; (2) fetches the provider's schedule blocks; (3) subtracts the provider's own booked appointments to get doctor-free slots; (4) slices into fixed-length candidate slots (default: 30 min); (5) for each candidate slot, checks if at least one operatory (room) is free by counting how many rooms have overlapping appointments — if busy rooms < total rooms, the slot is available. Operatory availability is determined from existing appointments, not from operatory schedules.

Q: How does dynamic provider selection work? A: Provider selection is controlled by the `kolla_enable_provider_selection` feature flag (default: off). When **disabled**, all appointments use `kolla_default_provider` — no AI matching. When **enabled**, the integration uses an inline LLM call (`Gen()`) to match the treatment description to a provider. There are two modes: (1) **Taxonomy mode** — if `kolla_selection_provider_criteria` is empty, the AI matches based on provider taxonomy from the catalog (e.g., "cleaning" → Hygienist, "filling" → General); (2) **Criteria mode** — if custom criteria are set (e.g., `Jones, Tina:cleaning, hygiene; Albert, Brian:filling`), the AI uses these explicit mappings for more precise control. In both cases, if resolution fails, it falls back to `kolla_default_provider`.

Q: How do I configure custom provider selection criteria? A: Set `kolla_selection_provider_criteria` with provider:service pairs separated by semicolons. Each entry maps a provider name to the services they handle. Examples: `Jones, Tina:cleaning, teeth whitening, hygiene; Albert, Brian:filling, extraction, root canal; Lexington, Sarah:exam, checkup, consultation`. Use natural language descriptions — the AI understands synonyms (e.g., "cleaning" matches "dental hygiene"). Make sure `kolla_enable_provider_selection` is set to `True` for criteria to take effect.

Q: What happens if a patient already exists in the system? A: When a phone number is identified during the conversation, the integration searches for existing contacts. If found, the patient's info is injected into the agent's prompt and the booking uses the existing contact record (marked "EP"). If not found, a new patient is created atomically with the appointment (marked "NP").

Q: What are the "(EP)" and "(NP)" markers? A: "(EP)" means Existing Patient — the appointment was created for a known contact. "(NP)" means New Patient — a new contact was created along with the appointment. These markers help clinic staff identify patient status.

Q: How is the phone number normalized? A: The leading `+` is removed, and if the number is 11 digits starting with `1` (US country code), the `1` is stripped. For example, `+12125551234` becomes `2125551234`.

Q: What are connector-id and consumer-id headers? A: These are Kolla-specific identifiers. `connector-id` identifies the dental system type (e.g., `opendental`), and `consumer-id` identifies your specific consumer instance within Kolla. Both are required for all API calls alongside the Bearer token.