Mozrest Integration

What the AI can do

Summary

Mozrest Integration connects **Mozrest** (restaurant reservation platform) with **Newo Platform**.

It enables:

- Checking real-time table availability for a given date, time, and party size
- Booking restaurant reservations with full guest details
- Cancelling existing reservations
- Restaurant area/location management with AI-powered area matching
- Party size and booking window validation

This integration is designed for **restaurants and hospitality businesses** who need **automated reservation management through a conversational AI agent** (voice or chat).

Common use cases

- When a guest asks about availability → Query Mozrest for open slots with party size and preferred date/time
- When a guest wants to book a table → Validate details, match area via LLM, create reservation in Mozrest
- When a guest wants to cancel a reservation → Delete the booking from Mozrest
- When a guest mentions a preferred area → AI matches the description to available restaurant locations

Features at a glance

Feature	Supported	Notes
Check Availability	✅	By date, time, party size, and timezone
Book Reservation	✅	Full guest details with area selection
Cancel Reservation	✅	By booking ID
Area/Location Management	✅	Cached per session from availability response
AI Area Matching	✅	LLM matches guest description to restaurant areas
Party Size Handling	✅	Passed to availability and booking APIs
Large Group Detection	✅	Configurable minimum party size threshold
Booking Window Validation	✅	Max days in advance configurable
SMS Booking Confirmation	✅	Schema template for confirmation messages
Reschedule Reservation	❌	Cancel + rebook as workaround
Waitlist Management	❌	Not implemented
Multi-Restaurant Support	❌	Single restaurant per integration instance

Before You Start

Before installation:

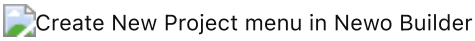
- Active **Mozrest** account with API access
- Restaurant configured in Mozrest with areas/locations
- **Mozrest connector** credentials (configured at platform level)

Setup

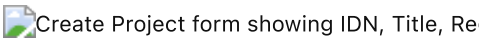
Create the Newo project

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 — Configure Mozrest Connector

1. Ensure your Mozrest account is active and API-enabled
2. Configure the Mozrest connector in the Newo platform with your account credentials
3. Verify the restaurant ID is available in project attributes

3.2 Step 2 — Connect in Platform

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

Attribute	Required	Description
`project_business_time_zone`	✅	Restaurant timezone (e.g., `America/New_York`)
`project_business_name`	✅	Restaurant name
`project_attributes_private_dynamic_restaurant_google_restaurant_id`	✅	Restaurant identifier for Mozrest

`mozrest_enable_slot_check`	❌	Enable availability checking (default: `True`)
`mozrest_enable_booking`	❌	Enable booking capability (default: `True`)
`mozrest_enable_cancellation`	❌	Enable cancellation capability (default: `True`)

3. Click **Save + Publish All**
4. On publish, the SetupFlow automatically:
    - Creates Mozrest and connections
    - Injects booking, availability, and validation payload schemas for the agent
    - Enables booking, availability, and cancellation features

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 Mozrest's reservation API for table management. Availability returns a list of areas with open slots, which are cached per session. During booking, an LLM matches the guest's area preference to available locations before creating the reservation.

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

Where to test

Testing can be done through the **Newo conversation interface** (chat or voice channel). Each flow (Availability, Booking, Cancellation) has a dedicated `integration_test` webhook endpoint for connectivity validation.

How to test that everything works

To test the integration:

1. **Setup**: Publish the project and verify connectors are created and schemas injected
2. **Availability**: Ask "Do you have a table for 4 tomorrow at 7 PM?" — verify available slots returned with area information
3. **Booking**: Complete a reservation — verify booking created in Mozrest with correct guest details and area
4. **Cancellation**: Cancel a reservation — verify booking removed from Mozrest

If no action occurs:

- Ensure Mozrest connector is properly configured with valid credentials
- Verify `project_business_time_zone` is set correctly

- Verify the restaurant ID attribute is populated
- Check feature flags are enabled (`mozrest_enable_slot_check` , `mozrest_enable_booking` , `mozrest_enable_cancellation`)
- Confirm the integration was re-published after changes
- Test individual flows via `integration_test` webhook endpoints

> *Note: This integration performs real operations in Mozrest. Reservations created or cancelled through the agent are reflected in the live restaurant system.*

FAQ

Q: How does the AI select the restaurant area? A: During availability checking, Mozrest returns a list of areas (e.g., patio, main dining, bar). These are cached per session. When the guest books, an LLM (Gemini 2.5 Flash) matches the guest's area description (e.g., "outside" or "by the window") to the closest available area. The matching uses low temperature (0.2) for deterministic results.

Q: What guest information is required for booking? A: All of the following: first name, last name, email, phone number, date/time, party size, and area preference. Optional: notes for special requests.

Q: Can I connect multiple restaurants? A: Each integration instance supports one restaurant. For multiple restaurants, create separate integration instances.

Q: How are phone numbers handled? A: Phone numbers are expected in E.164 format (e.g., `+15551234567`). The integration strips the leading `+` before sending to the Mozrest API.

Q: What happens if the guest doesn't specify an area? A: The area field is required in the booking payload. The agent should ask the guest for their preference, or suggest available areas returned from the availability check.

Q: Does the integration support large group reservations? A: Yes. The availability payload schema includes a `regularGroup` validation field and a configurable minimum party size threshold (`project_attributes_restaurant_large_group_reservation_minimum_party_size`) for special handling of large parties.

Q: How far in advance can reservations be made? A: Configurable via `project_attributes_setting_booking_check_availability_max_days_in_advance` . The availability validation schema enforces this booking window.

Q: Does the integration handle waitlists? A: No. If no slots are available for the requested date/time/party size, the agent can only suggest alternative times from the availability results.