Manual de integración — Bitrix24 y Botmaker

Middleware Laravel · Generado: 2026-04-06 22:48 · URL base de referencia: http://localhost

1. ¿Qué es esta aplicación?

Es un **middleware** (puente) entre **Botmaker** (WhatsApp / bots conversacionales) y **Bitrix24** (CRM y leads). Recibe eventos por **webhooks**, los registra, los procesa en **cola** y llama a las APIs necesarias en ambos sentidos.

- **Botmaker → Bitrix24:** crea o actualiza **leads** en el CRM con los datos que llegan del bot.
- **Bitrix24 → Botmaker:** cuando Bitrix avisa un evento, la app puede enviar un **mensaje de WhatsApp** al contacto mediante la API de Botmaker.

2. Bitrix24 — qué debes configurar (explícito)

2.1 Webhook de entrada hacia esta aplicación

Bitrix debe poder enviar peticiones **POST** a:

`http://localhost/webhook/bitrix24`

Configura en Bitrix24 un **webhook saliente** (suscripción a eventos del CRM) que apunte a esa URL. La ruta exacta depende de tu `APP_URL` en `.env`.

2.2 Seguridad: token compartido (obligatorio)

En el cuerpo JSON del webhook, Bitrix envía normalmente un bloque `auth`. Esta aplicación exige que el campo:

`auth.application_token`

sea **exactamente igual** al valor que guardes en tu servidor como:

`BITRIX24_WEBHOOK_SECRET` (en `.env` o en Configuración del panel).

Si no coinciden, la respuesta será **401 Invalid signature** y no se procesará el evento.

2.3 Formato mínimo del JSON que valida la aplicación

Para que el webhook sea aceptado (antes de filtros y cola), el payload debe incluir:

- `event`: cadena no vacía (nombre del evento de Bitrix).
- `data.FIELDS.ID`: identificador del lead (u objeto relacionado según el evento).
- `auth.application_token`: presente y coincidente con el secreto configurado.

Nota: el parseo de teléfono y mensajes para Bitrix→Botmaker asume estructuras típicas de lead (`PHONE`, `COMMENTS`, `NAME`, etc.). Si tu portal usa campos distintos, deberás ajustar **mapeos dinámicos** en el panel (*Mapeo de campos* para plataforma Bitrix24) o el código.

2.4 Webhook de salida (REST) hacia Bitrix24

Para crear/actualizar leads desde el flujo Botmaker→Bitrix, necesitas una URL de webhook de Bitrix24 con permisos de CRM, por ejemplo:

`https://TU_PORTAL.bitrix24.com/rest/USUARIO/CODIGO_SECRETO/`

Esa URL completa va en `BITRIX24_WEBHOOK_URL`. La aplicación llamará métodos como `crm.lead.add`, `crm.lead.update` y `crm.contact.list` mediante POST a `{WEBHOOK_URL}/{metodo}`.

Asegúrate de que el webhook tenga permisos para: **leer contactos**, **crear y actualizar leads**.

3. Botmaker — qué debes configurar (explícito)

3.1 Webhook de entrada hacia esta aplicación

Botmaker debe enviar eventos **POST** a:

`http://localhost/webhook/botmaker`

3.2 Seguridad: cabecera de firma (obligatorio)

Cada petición debe incluir la cabecera HTTP:

`X-Botmaker-Signature: <valor_secreto>`

Ese valor debe ser **idéntico** (comparación segura) a `BOTMAKER_WEBHOOK_SECRET` en tu servidor. Si falta o no coincide → **401 Invalid signature**.

3.3 Formato mínimo del JSON que valida la aplicación

- `event` o `type`: cadena no vacía que identifique el evento.
- Teléfono o contacto: al menos uno de `whatsappNumber`, `contact.phone`, `phone`, `contactId` o `customerId` (como cadena identificable).

La aplicación fusiona datos de `messages`, `clientPayload`, `context`, `attributes` y `variables` para obtener nombre, apellidos, fechas, etc. Conviene que el flujo del bot rellene esos datos de forma coherente con tus **mapeos**.

3.4 API de salida (envío de WhatsApp desde Bitrix→Botmaker)

Debes disponer de:

- `BOTMAKER_API_URL`: por defecto `https://go.botmaker.com/api/v1.0`
- `BOTMAKER_API_TOKEN`: token Bearer válido con permiso para enviar mensajes.

La aplicación llama al endpoint (POST):

`{BOTMAKER_API_URL}/chats-actions/send-messages`

con cuerpo JSON que incluye `chatPlatform: whatsapp`, `whatsappNumber` normalizado y el texto del mensaje. Si en el panel defines un **número/canal por defecto** (WhatsApp), se envía también `chatChannelId` cuando corresponde.

3.5 Correo electrónico (E-mail) hacia Bitrix

El flujo Botmaker→Bitrix puede enviar el **EMAIL** estándar del lead en Bitrix (mismo formato multivalor que `PHONE`: `VALUE` + `VALUE_TYPE: WORK`). Los alias de origen incluyen `email`, `correo`, `mail`, etc. (configurable en `config/integrations.php` o JSON en `.env`). El valor debe ser un correo válido; si no lo es, el campo no se envía.

Si en tu portal el correo obligatorio es un **campo personalizado** (`UF_CRM_*`), define en el mapeo `email` → ese código (y comprueba en Bitrix si el campo admite formato múltiple o solo texto plano).

4. Variables de entorno (.env) — referencia rápida

Variable	Uso
`APP_URL`	URL base pública del middleware (webhooks y enlaces).
`BOTMAKER_API_URL`	Base de la API REST de Botmaker.
`BOTMAKER_API_TOKEN`	Token para enviar mensajes y llamadas salientes.
`BOTMAKER_WEBHOOK_SECRET`	Debe coincidir con `X-Botmaker-Signature`.
`BITRIX24_WEBHOOK_URL`	URL REST de entrada (crm.lead.*, crm.contact.*).
`BITRIX24_WEBHOOK_SECRET`	Debe coincidir con `auth.application_token`.
`BOTMAKER_SALARY_CURRENCY`	Moneda para normalizar salario (ej. MXN).
`BOTMAKER_SOURCE_ALIASES_JSON`	Opcional: JSON de alias de campos (ver `config/integrations.php`).
`BOTMAKER_BITRIX_FIELDS_JSON`	Opcional: mapeo campo lógico → código de campo Bitrix.
`BOTMAKER_ENUM_MAPS_JSON`	Opcional: mapas de listas (estado, semanas cotizadas, etc.).

Valores sensibles también pueden gestionarse desde el panel en **Configuración** (tabla `settings`), con prioridad sobre el fichero de configuración cuando existan.

5. Flujos paso a paso

5.1 Botmaker → Bitrix24

1. Llega POST a `/webhook/botmaker` con firma válida.
2. Se crea un registro en *Registros de Webhooks* con `correlation_id`.
3. Si pasa *Filtros de eventos*, se encola el trabajo en la cola `webhooks`.
4. El worker parsea el payload, aplica mapeos (BD o integración por defecto).
5. Se busca contacto por teléfono en Bitrix; si hay lead asociado se **actualiza**, si no se **crea** lead.
6. Los campos estándar del lead incluyen título, comentarios, teléfono, **correo**, nombre y apellidos, fecha de nacimiento, listas (semanas cotizadas, estatus laboral, estado) y último salario, según el mapeo en configuración (clase `MapBotmakerCanonicalToBitrixLead` y/o mapeos dinámicos en BD).
7. El resultado (éxito o error) queda en el mismo registro de webhook.

5.2 Bitrix24 → Botmaker

1. Llega POST a `/webhook/bitrix24` con `auth.application_token` válido.
2. Registro en base de datos y, si aplica, filtros.
3. El worker obtiene teléfono y datos del evento, aplica reglas de **notificación** y plantillas.
4. Se llama a la API de Botmaker para enviar el mensaje al número indicado.

6. Panel web (/monitor)

Tras iniciar sesión y verificar correo, según tu rol verás secciones como:

- **Tablero** — resumen del día y últimos eventos.
- **Registros de Webhooks** — auditoría y detalle por `correlation_id`.
- **Webhooks fallidos** — reintentos y diagnóstico.
- **Configuración** — credenciales y parámetros persistidos.
- **Mapeo de campos** — rutas JSON → campos destino por plataforma.
- **Reglas de notificación** — mensajes salientes Bitrix→Botmaker por evento.
- **Plantillas** — textos reutilizables con variables.
- **Números WhatsApp** — canal por defecto para envíos.
- **Filtros de eventos** — ignorar o aceptar eventos según reglas.
- **Pruebas de integración** — (permiso `settings.manage`, misma categoría que Configuración) permite ejecutar desde el navegador: creación de un **lead de prueba completo** en Bitrix24 y comprobación de conectividad (API Botmaker, API Bitrix vía `crm.lead.list`, y revisión de registros de webhook atorados en cola). Muestra resumen de webhooks del día. Ruta: `/monitor/integration-tests`.

En la pantalla de **inicio de sesión** hay un enlace al manual público `/manual`.

Esta misma guía está en **Manual de integración** del menú lateral (usuarios con acceso al tablero). La copia pública sin sesión es la ruta `/manual`; dentro de la app autenticada, `/monitor/manual`.

7. Límites y buenas prácticas

- **Rate limit:** hasta 120 solicitudes por minuto por IP en rutas `/webhook/*`.
- **Reintentos:** los jobs usan backoff configurable (`retry.*` en settings o config).
- No publiques tokens en repositorios; usa `.env` o el panel solo en servidores controlados.
- Prueba primero con eventos de prueba y revisa *Registros de Webhooks* antes de producción.

8. Comandos útiles

- `php artisan webhook:status` — resumen de actividad del día, últimos eventos y comprobación de Botmaker API, Bitrix API y cola (usa `IntegrationProbeService`).
- `php artisan bitrix:test-lead` — crea un **lead real** en Bitrix24 con todos los campos del mapeo estándar (incluido EMAIL). Opción `--show-fields` imprime el JSON enviado a `crm.lead.add`. Misma lógica que el botón del panel «Pruebas de integración».
- `php artisan manual:pdf` — regenera el PDF estático en `public/docs/Manual-Integracion-Bitrix24-Botmaker.pdf` (mismo contenido que la descarga en línea desde `/manual/descargar-pdf`).

9. Pruebas desde el webservice (panel y JSON)

Además de Artisan, puedes probar integraciones con la **misma sesión** del panel (cookies de Laravel).

9.1 Panel web

Ruta: `http://localhost/monitor/integration-tests`. Requiere usuario autenticado, correo verificado y permiso **gestionar configuración** (`settings.manage`).

- **Crear lead de prueba:** llama a Bitrix `crm.lead.add` con datos de ejemplo (nombre, teléfono, email, campos personalizados del mapeo por defecto, listas, etc.). Úsalo solo en entornos controlados.
- **Probar conectividad:** petición ligera a Botmaker, listado mínimo de leads en Bitrix y comprobación de registros «atorados» en estados recibido/procesando.

Las llamadas a Bitrix pueden tardar varios segundos (timeout HTTP configurable). Si algo falla, revisa la consola del navegador y `storage/logs/laravel.log`.

9.2 Endpoints JSON (HTTP)

Para Postman, scripts internos o orquestadores que compartan la sesión del navegador:

- `POST http://localhost/monitor/integration-probes/bitrix-sample` — crea el lead de prueba; respuesta JSON con `success`, `lead_id`, `fields`, `body`, etc. Requiere cabecera `X-XSRF-TOKEN` (valor de la cookie `XSRF-TOKEN`) y cookie de sesión. Límite aproximado: 15 peticiones por minuto (throttle).
- `GET http://localhost/monitor/integration-probes/connectivity` — estado de Botmaker, Bitrix, cola y resumen de webhooks del día. Límite: 60 peticiones por minuto.

No están pensados para integración pública sin autenticación; no sustituyen a los webhooks oficiales `/webhook/botmaker` y `/webhook/bitrix24`.

9.3 Servicio interno

La lógica compartida vive en `App\Services\IntegrationProbeService`, usada por los comandos, el panel Livewire y el controlador JSON.