CRM Documentation

Generated Tue, Sep 22, 2026 11:33 AM · Full documentation

Contents

Algos CRM — Demo Environment

Polished fictional tenant for the public live demo, screenshots, and marketing.

Seed command

`DEMO_SEED_PASSWORD` must be set in the environment. If it is missing, seeding and reset abort.

```
php artisan db:seed --class=DemoDataSeeder
```

Safe to re-run. Upserts the Northstar Solutions tenant only; does not wipe other companies.

If a demo persona email already belongs to a different company, seeding **aborts** and does not move or overwrite that user.

Reset command

```
php artisan demo:reset
```

Resets **only** the verified Northstar tenant (`northstar-solutions`), then re-seeds it. Refuses to run if:

- the company cannot be found
- the slug is not exactly `northstar-solutions`
- the resolved company is the default company

Do **not** run `demo:reset` against production unless it is explicitly approved. Use local or staging.

The command is registered on the existing `schedule:run` cron. It uses `withoutOverlapping()` and runs only when `DEMO_RESET_ENABLED=true`. Leave that flag false until production reset is explicitly approved.

Identity

Field	Value
Company	Northstar Solutions
Slug	`northstar-solutions`

Passwords exist only as `DEMO_SEED_PASSWORD` (environment secret) and as hashes in the database. They must not appear in source, git, HTML, JavaScript, or public docs.

What gets seeded

- 1 demo company (Enterprise/Professional plan when available)
- 3 users: Admin, Sales Manager (custom role), Sales Representative
- ~18 leads across pipeline stages
- 8 customers (from won leads)
- ~19 tasks (overdue / upcoming / completed)

- Lead activity timeline entries

Not included in production `DatabaseSeeder` auto-run.

docs/README.md

CRM Documentation

Professional documentation for the multi-tenant Laravel CRM platform.

> **Audience:** developers, operators, Super Admins, and tenant end users.
>
> **Stack:** PHP 8.2+, Laravel 12, AdminLTE 3, database queues, custom RBAC (no Spatie).
>
> **Diagrams:** Mermaid (rendered by GitHub, many IDEs, and most Markdown viewers).

Table of contents

Getting started

Architecture

Product modules

Channels (omnichannel)

Doc	Description
WhatsApp Cloud API	Inbound messages + Inbox reply (fully supported)
Website Forms	Existing website lead webhook + planned builder
Facebook Messenger	**Planned**
Instagram Lead Forms	**Planned**
Instagram DM	**Planned**
Public API channel	**Planned**

Super Admin

Doc	Description
Super Admin guide	Platform console: companies, plans, settings, inquiries

Operations

Doc	Description
Queues	Database queue, `channels` queue, workers
Scheduler	Cron / `schedule:run`, reminders, pruning
CI/CD	**Recommended** pipeline (not yet in repo)

Development

Doc	Description
Coding standards	Pint, Laravel conventions, tenancy/RBAC rules
Contributing	Public policy: who can contribute, PR-only, no production access, IP

End users

Doc	Description
End-user manual	Day-to-day CRM usage for admins & sales

Reference

Doc	Description
API	Public webhooks today; **Recommended** REST API later
Troubleshooting	Common failures & fixes
Roadmap	Delivered milestones & planned work

Doc	Description
Changelog	High-level product history

Quick start (local)

```
composer setup
php artisan db:seed
php artisan permissions:sync
php artisan serve
# other terminal:
php artisan queue:work --queue=channels,default
```

Open `APP_URL` (default `http://localhost:8000`).

In-app docs viewer

While logged in (tenant CRM or Super Admin), open:

- `/docs` — documentation home
- `/docs/getting-started/installation` — example page

Sidebar: **Account → Documentation** (CRM) or **Documentation** (Super Admin).

On any docs page you can:

- **PDF** — download the current page
- **Download all** — download the full documentation as one PDF (`/docs/pdf`)

Support notes

- Tenant CRM UI: AdminLTE (`crm-app` layout).
- Super Admin UI: separate `sa-app` layout under `/superadmin`.
- There is **no** `routes/api.php` yet; inbound integrations use webhooks on `routes/web.php` (CSRF-exempt).
- Channel providers listed in `config/channels.php` may appear in the UI before an adapter is registered — see Channels overview.

Changelog

High-level product history for operators and developers.

For git-level detail, use the repository commit / PR history.

Format inspired by Keep a Changelog.

Unreleased

Added

- Public contribution policy: PR-only workflow, no production access, and intellectual-property terms (`CONTRIBUTING.md`, `docs/development/contributing.md`).
- Pull request template and `CODEOWNERS` so outside contributions stay reviewed.
- Professional documentation tree under `/docs` (installation, architecture, channels, Super Admin, operations, user manual, roadmap).

2026-08 — Inbox & WhatsApp reply

Added

- Tenant **Inbox** (`/inbox`) with conversation list/thread UI.
- WhatsApp outbound replies via Graph API.
- Conversation assign + open/pending/closed status.
- Permissions: `view.inbox`, `reply.inbox`, `assign.inbox`.

2026-07 — Channels & Meta

Added

- Channels engine (connections, webhook events, contacts, conversations, lead channel meta).
- Channels UI for tenants (connect, test, sync, retry, disconnect, regenerate secret).
- Public webhook `GET|POST /webhooks/channels/{uuid}`.
- **Generic Webhook** adapter (signed JSON leads/messages).
- **Facebook Lead Ads** adapter (leadgen + Graph fetch).
- **WhatsApp Cloud API** inbound adapter + Meta verification.
- Config: `META_APP_SECRET`, `META_GRAPH_VERSION`, `CHANNELS_WEBHOOK_QUEUE`.
- MySQL-safe short index names + recovery for partial channel migrations.

Fixed

- Channel route binding company-scope leak (`orWhere` grouping).
- Company Profile vs Company Settings controller (`edit()` restore after bad merge).

2026-07 — Platform & tenancy UX

Added

- Company Profile read-only page + Company Settings editor.
- Super Admin contact inquiries for marketing form submissions.
- Soft-delete company identifier cleanup patterns.

Changed

- Create-user flow enhancements (preset password email, create-role modal) in earlier PRs.
- Password eye icon overlap fixes on user forms.

Earlier — Core CRM

Added

- Multi-company tenancy with `CurrentCompany` / `CompanyScope`.
- Custom RBAC (`permissions:sync`, admin/sales defaults).
- Leads, customers, tasks, reports, CSV import/export.
- Lead/task reminder scheduler commands.
- Super Admin: companies, plans/limits, analytics, email templates, impersonation.
- Marketing site + website lead webhook.
- AdminLTE tenant shell.

Notes

- Channel providers may appear in the UI before adapters ship; see Channels overview status matrix.
- CI/CD workflows are **Recommended** and documented under CI/CD.

Roadmap

Delivered and planned work for the omnichannel multi-tenant CRM.

Delivered

Milestone	Summary
Core CRM	Customers, leads, tasks, reports, users/roles, CSV, reminders
Multi-tenancy	Company scope, plans/limits, Super Admin companies
Super Admin platform	Plans, analytics, email templates, contact inquiries, impersonation
M1 Channels engine	Tables, ingestion, jobs, lead matching, generic adapter
M2 Channels UI	Tenant connect/test/sync/disconnect + public webhook URL
M3 Facebook Lead Ads	Meta verify, signatures, Graph lead fetch → CRM leads
WhatsApp Cloud	Inbound messages, verify, connection test
Inbox + WhatsApp reply	Conversation UI, assign/status, Graph send
Docs suite	This `/docs` tree

In progress / hardening

- Per-tenant Meta App Secret storage (today `META_APP_SECRET` is largely platform `.env`, with connection secret fallback)
- OAuth “Connect with Meta” instead of pasting tokens
- CI workflows in-repo (**Recommended**)

Planned channels

Provider	Priority notes
Instagram Lead Forms	Reuse Lead Ads patterns
Facebook Messenger	Inbox parity with WhatsApp
Instagram DM	Messaging + Inbox
Website Forms builder	First-class forms vs legacy website webhook
Public API	Versioned REST + API keys

Planned product

Item	Notes
Rich media in Inbox	Images/docs beyond text placeholders
WhatsApp template messages	Business-initiated outside session window
Delivery receipt updates	Apply status webhooks to stored messages
Realtime Inbox	Echo/Reverb or polling
OpenAPI docs	When REST API lands

Suggested build order

1. Stabilize WhatsApp production (tokens, ngrok/prod HTTPS, monitoring)
2. Instagram Lead Forms
3. Messenger / Instagram DM
4. Website Forms builder
5. Public REST API

Related

- Channels overview
- Changelog
- CI/CD

Troubleshooting

Common issues and fixes.

Installation / boot

Symptom	Likely cause	Fix
No application encryption key	Missing `APP_KEY`	`php artisan key:generate`
SQLite cannot open DB	File missing	Create `database/database.sqlite`
Mixed content / wrong links	`APP_URL` mismatch	Set `APP_URL` including port
Assets 404	Frontend not built	`npm run build`

Permissions / menus

Symptom	Fix
Channels / Inbox missing in sidebar	`php artisan permissions:sync`
403 on expected page	Role lacks permission; check Roles UI / seeder
Sales can view but not manage channels	Expected — needs `manage.channels`

Tenancy

Symptom	Fix
Empty lists in production artisan/tinker	Fail-closed without CurrentCompany — set company context
Cross-company data visible in tests	Create distinct companies explicitly on users
Soft-deleted company blocks email reuse	Run identifier release command / restore flow

Queues / channels

Symptom	Fix
Webhook `202` but no lead	Start `php artisan queue:work --queue=channels,default`
401 Invalid signature	Wrong secret / `META_APP_SECRET`; regenerate; `config:clear`
Meta verify fails	Verify token mismatch; CRM down; using localhost without ngrok
MySQL migration “index name too long”	Use current short index migration; drop partial tables + migration row if stuck

Symptom	Fix
`channel_connections` already exists	Partial migration — drop channel tables carefully, delete migration record, re-run

WhatsApp / Facebook

Symptom	Fix
Test connection fails	Token / Phone Number ID / Page ID wrong
No Inbox thread	Worker; adapter; wrong phone number id filter
Reply fails	Conversation closed; token expired; Meta 24h window rules
Secret leaked in chat	Rotate App Secret + access tokens immediately

Mail

Symptom	Fix
No emails arriving	`MAIL_MAILER=log` only writes logs — switch to SMTP/Mailpit
Gmail blocked	App Password + correct from address

Scheduler

Symptom	Fix
No reminders	Cron missing `schedule:run`; reminder flags disabled in `.env`
Heartbeat stale	Cron / app path wrong in crontab

Company profile vs settings

Symptom	Fix
`/company/settings` 500 `edit()` missing	Ensure latest code with `CompanySettingsController::edit` restored

Getting more diagnostics

```
php artisan about
php artisan route:list --path=webhooks
php artisan queue:failed
tail -f storage/logs/laravel.log
```

For Meta local deliveries, open ngrok inspector: http://127.0.0.1:4040

Related

- Installation
- Queues

- Channels overview

API overview

Today — webhooks (public HTTP)

There is **no** versioned REST API (`routes/api.php` is absent). Integrations use webhook endpoints on the web stack (CSRF exempt).

Website leads

```
POST /webhooks/leads/website
```

Authenticated by shared secret middleware (`WEBSITE_LEAD_WEBHOOK_SECRET`).
See Website Forms.

Channels

```
GET|POST /webhooks/channels/{uuid}
```

Method	Purpose
GET	Meta subscription verification (`hub.challenge`) for Facebook Lead Ads / WhatsApp
POST	Inbound events (generic HMAC or Meta `X-Hub-Signature-256`)

Response on accept: `202` JSON `{ message, event_id, status }`.

Per-tenant isolation via connection UUID.
See Channels overview and provider docs.

Generic Webhook contract

Documented payload shapes: Generic Webhook.

Planned / Recommended — public REST API

Tracked as channel key `public_api` and future platform API.

Recommended direction:

Area	Proposal
Transport	`routes/api.php` + Sanctum/API tokens per company
Resources	Leads, Customers, Tasks (CRUD subsets)
Authz	Same RBAC permission slugs
Versioning	`/api/v1`

Area	Proposal
Docs	OpenAPI / Scramble later

Until built, prefer Generic Webhook for partner ingestion.

Related

- Public API channel (Planned)
- Roadmap

Multi-tenancy

Each paying customer is a **Company** (tenant). Users, leads, channels, conversations, and most records belong to exactly one company.

Concepts

Concept	Meaning
Company	Tenant workspace (status, plan, subscription)
CurrentCompany	Request-scoped active `company_id`
CompanyScope	Global Eloquent scope filtering by CurrentCompany
BelongsToCompany	Trait applying scope + auto-filling `company_id` on create
Super Admin	Platform operator — **no** company; uses `/superadmin`

How context is set

Middleware `company` (`EnsureCompanyContext`):

1. Clears any previous context.
2. If the user is a Super Admin → redirect to Super Admin dashboard (tenant CRM is not their home).
3. If the user has no company / company missing / suspended / subscription expired → deny (unless impersonating).
4. Otherwise `CurrentCompany::set($company)` and touch `last_active_at`.
5. On `terminate()`, clear context.

`EnsureCompanyContext` is prepended before route-model binding so `{id}` resolution respects the company scope.

Data isolation

[Diagram omitted in PDF — open this page in the online docs to view it.]

- Queries on scoped models automatically add `where company_id = ?`.
- Cross-tenant access must use `Model::withoutCompanyScope()` **intentionally** (e.g. webhook UUID lookup, then set company context).

Fail-closed behavior

`CompanyScope` (`app/Models/Scopes/CompanyScope.php`):

- If CurrentCompany is set → filter to that company.
- If **not** set:
    - **Production** (or when `TENANCY_FAIL_CLOSED_WITHOUT_CONTEXT=true`) → `whereRaw('0 = 1')` (zero rows).

    - Local/dev with config unset → scope is a no-op (helps artisan/tinker); prefer setting context explicitly.

Webhooks and tenancy

Public channel webhook: `GET|POST /webhooks/channels/{uuid}`

1. Resolve `ChannelConnection` **without** company scope by UUID.
2. Load that connection's company.
3. Ingest event with that `company_id`.
4. Job processing sets `CurrentCompany` from the event before running the adapter.

So each business's WhatsApp / Facebook / webhook URL is isolated by UUID + `company_id`.

Plan limits

`PlanLimitService` enforces `users`, `leads`, and `customers` against the company's plan (`max_*` columns and/or `plan_limits` rows). Channel-created leads call `assertCanAddLead`.

Impersonation

Super Admins may impersonate a company admin (logged). While impersonating, company context is the target tenant. Leave via `impersonation.leave`.

Developer rules

1. Prefer scoped queries; don't call `withoutCompanyScope()` unless necessary.
2. After unscoped lookup, set CurrentCompany before writes that should be tenant-owned.
3. Never trust a client-supplied `company_id` for authorization — derive from auth + scope/policy.
4. Policies should use `ChecksSameCompany` for model actions.

Related

- Architecture overview
- RBAC
- Channels overview

Architecture overview

High-level design of the multi-tenant CRM.

Stack

Layer	Choice
Framework	Laravel 12 / PHP 8.2+
Auth UI	Laravel Breeze (session auth)
Tenant UI	jeroennoten/laravel-AdminLTE 3
Super Admin UI	Custom `sa-app` layout
RBAC	Custom (`config/permissions.php` + `PermissionRegistry`) — **not** Spatie
Queues	Database driver (default)
PDF	barryvdh/laravel-dompdf (Super Admin exports)
Frontend build	Vite

Logical layers

[Diagram omitted in PDF — open this page in the online docs to view it.]

Route groups

Surface	Entry	Middleware highlights
Marketing	`routes/marketing.php`	Public
Tenant CRM	`routes/web.php`	`auth`, `verified.when_required`, `active`, `company`
Super Admin	`routes/superadmin.php`	`auth`, `active`, `superadmin`
Webhooks	top of `routes/web.php`	Throttle; CSRF exempt; no tenant session

There is **no** `routes/api.php` today. Integrations use webhooks. See API.

Request lifecycle (tenant)

[Diagram omitted in PDF — open this page in the online docs to view it.]

Key directories

Path	Responsibility
`app/Support/CurrentCompany.php`	Request-scoped tenant id
`app/Models/Concerns/BelongsToCompany.php`	Global scope + auto `company_id`
`app/Services/PermissionRegistry.php`	Sync permissions from config
`app/Services/Channels/*`	Omnichannel engine
`app/Providers/ChannelServiceProvider.php`	Register channel adapters
`app/Http/Middleware/EnsureCompanyContext.php`	Set/clear tenant; block suspended companies

Domains

Tenant CRM

Customers, leads (kanban), tasks, inbox, channels, reports, users/roles, company profile/settings, notifications, activity logs, CSV import/export.

Platform (Super Admin)

Companies, plans/limits, analytics, email templates, contact inquiries, platform settings/announcements, impersonation, Super Admin users.

Channels engine

`channel_connections` → signed/verified webhooks → `channel_webhook_events` → queued processing → leads / conversations / `lead_channel_meta`.

Design principles

1. **Tenant isolation by default** — company scope on models; fail-closed in production without context.
2. **Authorize in policies** — permission slugs `{action}.{module}`.
3. **Services over fat controllers** — channel processing, plan limits, notifications.
4. **Queue inbound channel work** — keep webhook HTTP fast (`202 Accepted`).
5. **Config-driven permissions & channel catalog** — sync with Artisan; adapters register explicitly.

Related docs

- Multi-tenancy
- RBAC
- Channels overview
- Modules overview

RBAC (roles & permissions)

Custom role-based access control. Permissions are defined in config and synced to the database — **Spatie Permission is not used**.

Model

[Diagram omitted in PDF — open this page in the online docs to view it.]

- Permission slug format: `{action}.{module}`
  Examples: `view.leads`, `manage.channels`, `reply.inbox`
- Roles are **per company** (after migrations that added `roles.company_id`).
- System role `admin` receives all permissions (`*`).
- Default role `sales` receives an explicit allow-list.

Modules & actions

Source: `config/permissions.php`.

Module	Actions
customers	view, create, import, update, delete
leads	view (own), view_all, create, import, update, delete, assign, convert, log
tasks	view (own), view_all, create, change_status, update, delete, assign
users	view, create, import, update, delete
roles	view, create, update, delete
activity_logs	view, view_own
notifications	view
company_settings	update
reports	view, export
demo	website_lead
channels	view, manage
inbox	view, reply, assign

Default sales grants

Exact list from `database/seeders/RbacSeeder.php`:

- `view/create/import/update/delete.customers`

- `view/create/import/update/delete/convert/log.leads` (no `view_all.leads`, no `assign.leads`)
- `view/change_status/update/delete.tasks` (no `create.tasks`, `view_all.tasks`, `assign.tasks`)
- `view_own.activity_logs`, `view.notifications`
- `view/export.reports`
- `view.channels` (no `manage.channels`)
- `view.inbox`, `reply.inbox` (no `assign.inbox`)

Admin (`*`) receives every synced permission. After custom Role UI edits, the database is source of truth.

Syncing permissions

Whenever you change `config/permissions.php`:

```
php artisan permissions:sync
```

This will:

1. Upsert permission rows for every `{action}.{module}`.
2. Delete stale permissions (and detach them from roles).
3. Sync default roles via `RbacRoleSynchronizer`.
4. Refresh gates.

Authorization in code

Mechanism	Usage
`$user->hasPermission('view.inbox')`	Direct check
`$this->authorize('view', $model)`	Policy
`@can('manage', $channel)`	Blade
`middleware('permission:view.notifications')`	Route

Gates are registered from DB permissions by `PermissionRegistrar`.

AdminLTE menu

Sidebar entries use `'can' => 'view.channels'`-style gate abilities (`config/adminlte.php`). Missing menu after deploy usually means permissions were not synced.

Adding a permission

1. Add module/action in `config/permissions.php`.
2. Run `php artisan permissions:sync`.
3. Grant on roles (UI or update `RbacSeeder` defaults).
4. Enforce via policy / middleware / `@can`.
5. Add Feature tests for allow/deny.

Related

- Multi-tenancy
- Modules overview
- Contributing

Facebook Lead Ads

Fully supported Meta Lead Ads → CRM lead ingestion.

Prerequisites

- Meta app with Lead Ads / webhooks capability
- Facebook Page + lead form
- Page access token
- `META_APP_SECRET` in CRM `.env` (same Meta app)

Meta links:

- Apps: https://developers.facebook.com/apps
- App Secret: **App settings → Basic**
- Webhooks: app dashboard → Webhooks / Page subscriptions for `leadgen`

Connect in CRM

1. Set `META_APP_SECRET` and `php artisan config:clear`
2. **Channels → Connect channel → Facebook Lead Ads**
3. Enter:
    - Display name
    - **Facebook Page ID** (`external_page_id`)
    - **Page access token**
4. Connect, then copy **Webhook URL** + **Verify token**

Meta webhook setup

1. Callback URL: public HTTPS URL to `/webhooks/channels/{uuid}` (use ngrok for local)
2. Verify token: from CRM connection page
3. Subscribe to **leadgen** for the Page
4. CRM handles `GET` hub.challenge verification automatically for this provider

Processing behavior

1. Validate `X-Hub-Signature-256` with `META_APP_SECRET` (fallback: connection webhook secret)
2. Parse `leadgen` changes; skip other pages if Page ID is set on the connection
3. Fetch lead field data from Graph API with the page token
4. Create/update CRM lead (source `facebook`) + `lead_channel_meta` (campaign/ad/form/page ids)
5. Idempotency key: `facebook_leadgen_{leadgen_id}`

Test connection

Calls Graph `GET /{page-id}?fields=id,name` with the stored token.

Queue

Inbound POSTs are queued on `channels`. Keep a worker running.

Security

- Rotate App Secret / page tokens if exposed
- Prefer short-lived tokens only for demos; use long-lived / system user tokens in production

Related

- Channels overview
- WhatsApp Cloud API (shares Meta app secret pattern)

Facebook Messenger

> **Status: Planned** — listed in `config/channels.php` (`facebook_messenger`) but **no adapter** is registered in `ChannelServiceProvider` .

Intended behavior (planned)

- Meta Page Messenger webhooks
- Inbound messages → Inbox conversations
- Optional outbound reply (similar to WhatsApp)
- Per-tenant Page token + verify token

What you can do today

- Create a Channels UI row may be possible if the provider is enabled in config, but processing will not fully work (**Adapter coming soon**).
- For messaging now, use **WhatsApp Cloud API**.
- For custom bots, use **Generic Webhook** message payloads.

Related

- Channels overview
- WhatsApp Cloud API
- Roadmap

Generic Webhook

Fully supported channel for signed JSON inbound leads and messages.

When to use

- Website or middleware that can POST JSON
- Local end-to-end testing of the Channels engine
- Custom integrations without Meta

Connect in CRM

1. Admin → **Administration → Channels → Connect channel**
2. Provider: **Generic Webhook**
3. Display name (required)
4. Optional: external account id, access token, custom webhook secret (blank = auto-generate)
5. **Connect channel**
6. Copy the **webhook secret** immediately (shown once) and the **Webhook URL**

Endpoint

```
POST /webhooks/channels/{uuid}
Content-Type: application/json
X-Channel-Signature: sha256={hmac_sha256_hex_of_raw_body}
```

HMAC key = connection webhook secret.

Header may also be accepted as raw hex without prefix in some paths; prefer `sha256=` prefix.

CSRF exempt. Rate limited (`CHANNELS_WEBHOOK_RATE_LIMIT`).

Lead payload

```
{
  "type": "lead",
  "name": "Alex Morgan",
  "email": "alex@example.com",
  "phone": "+15550100000",
  "company": "Northline",
  "notes": "From website",
  "external_user_id": "site_123",
  "external_lead_id": "lead_456",
  "campaign": {
    "campaign_id": "camp_1",
    "form_id": "form_9"
  }
}
```

Requires `name` and at least one of `email` / `phone`.

Lead source slug: `website`.

Message payload

```
{
  "type": "message",
  "external_user_id": "user_1",
  "body": "Hello",
  "phone": "+15550100000",
  "email": "alex@example.com",
  "provider_message_id": "msg_1"
}
```

Creates/updates conversation + lead matching.

Test with curl (Git Bash)

```
URL='http://127.0.0.1:8000/webhooks/channels/YOUR-UUID'
SECRET='YOUR-SECRET'
BODY='{"type":"lead","name":"Alex Morgan","email":"alex@example.com","phone":"+15550100000"}'
SIG=$(printf '%s' "$BODY" | openssl dgst -sha256 -hmac "$SECRET" | awk '{print $2}')

curl -i -X POST "$URL" \
  -H "Content-Type: application/json" \
  -H "X-Channel-Signature: sha256=$SIG" \
  -d "$BODY"
```

Expect `202` + `"accepted"`. Run:

```
php artisan queue:work --queue=channels,default
```

Then refresh channel **Recent events** and **Leads**.

UI Test connection

Validates HMAC signing health; does **not** create a lead.

Troubleshooting

Issue	Fix
401 Invalid signature	Wrong secret; regenerate and update client
202 but no lead	Queue worker not running
404 Unknown connection	Wrong UUID / deleted connection

Instagram DM

> **Status: Planned** — `instagram_dm` is listed in config; **no adapter** registered.

Intended behavior (planned)

- Instagram Messaging API webhooks
- Inbox threads + replies
- Per-tenant credentials

What you can do today

Use **WhatsApp** / **Generic Webhook** for messaging experiments.

Related

- Channels overview
- Roadmap

Instagram Lead Forms

> **Status: Planned** — `instagram_lead_forms` is enabled in config but has **no registered adapter**.

Intended behavior (planned)

- Similar pipeline to Facebook Lead Ads
- Meta Instagram lead form webhooks / Graph fetch
- CRM leads with source mapping + `lead_channel_meta`

What you can do today

Use **Facebook Lead Ads** for Meta lead forms on Pages, or ingest via **Generic Webhook** if you control a middleware.

Related

- Facebook Lead Ads
- Roadmap

Channels overview

Omnichannel **Channels Engine** lets each tenant connect inbound sources. Connections are per-company; webhooks are per-connection UUID.

Status matrix

Provider key	Label	Adapter registered	Notes
`generic_webhook`	Generic Webhook	**Yes**	Best for local E2E tests
`facebook_lead_ads`	Facebook Lead Ads	**Yes**	Meta leadgen + Graph fetch
`whatsapp_cloud`	WhatsApp Cloud API	**Yes**	Inbound + Inbox reply
`website_form`	Website Forms	**No** (UI listed)	Use dedicated website webhook today; builder **Planned**
`facebook_messenger`	Facebook Messenger	**No**	**Planned**
`instagram_lead_forms`	Instagram Lead Forms	**No**	**Planned**
`instagram_dm`	Instagram DM	**No**	**Planned**
`public_api`	Public API	**No**	**Planned**

Catalog: `config/channels.php`.
Registration: `app/Providers/ChannelServiceProvider.php`.

Multi-tenant model

[Diagram omitted in PDF — open this page in the online docs to view it.]

Each connection stores (as applicable): encrypted access token, webhook secret, verify token, external page/phone id, status/health.

Pipeline

[Diagram omitted in PDF — open this page in the online docs to view it.]

Permissions

Permission	Who
`view.channels`	List/show connections

Permission	Who
`manage.channels`	Create, test, sync, disconnect, secrets, delete
`view.inbox` / `reply.inbox` / `assign.inbox`	Messaging UI

UI entry

Administration → Channels

Actions on a connection: Test, Sync, Retry (on error), Disconnect, Regenerate webhook secret, Delete.

Environment

```
CHANNELS_WEBHOOK_QUEUE=channels
CHANNELS_WEBHOOK_RATE_LIMIT=60
META_APP_SECRET=
META_GRAPH_VERSION=v21.0
```

Workers must listen to `channels` (see Queues).

Local Meta testing

Meta cannot call `127.0.0.1`. Use ngrok (or deploy) and set the Callback URL to:

`https://{ngrok-host}/webhooks/channels/{uuid}`

Setup guides

- Generic Webhook
- Facebook Lead Ads
- WhatsApp Cloud API
- Website Forms
- Facebook Messenger (Planned)
- Instagram Lead Forms (Planned)
- Instagram DM (Planned)
- Public API (Planned)

Public API channel

> **Status: Planned** — `public_api` appears in the channel catalog; there is **no** REST API surface (`routes/api.php` does not exist) and **no** adapter.

Intended behavior (planned)

- Authenticated public HTTP API for creating leads / events per tenant
- API keys or OAuth per company
- Rate limiting & audit logs

What you can do today

- **Generic Webhook** with HMAC signatures per connection
- **Website lead webhook** for a shared server-to-server form endpoint

See also API overview.

Related

- Generic Webhook
- Roadmap

Website Forms

Current (supported) path — website lead webhook

Independent of the Channels UI “Website Forms” provider. Server-to-server POST creates a tenant lead.

Configure

```
WEBSITE_LEAD_WEBHOOK_SECRET=long-random-secret
WEBSITE_LEAD_CREATED_BY_EMAIL=admin@example.com
WEBSITE_LEAD_RATE_LIMIT=10
```

Endpoint

```
POST /webhooks/leads/website
```

Middleware: `website-lead-webhook` + throttle `website-leads`. CSRF exempt.

Never put the webhook secret in browser JavaScript. Call from your backend only.

Demo UI

Users with `website_lead.demo` can open **Administration → Website Lead Demo** to exercise the flow.

vs Channels Generic Webhook

	Website lead webhook	Generic Webhook channel
Config	`.env` secret	Per-connection secret
Multi-tenant routing	Implementation-specific / shared endpoint	Per-company UUID URL
UI connection row	No	Yes under Channels

For new multi-tenant form backends, prefer **Generic Webhook** per company unless you standardize on the dedicated website endpoint.

Planned — Website Forms channel builder

Listed in `config/channels.php` as `website_form` but **no channel adapter is registered** yet.

Planned capabilities:

- Form builder / embed snippets per tenant
- First-class Channels connection + health
- Unified event log with other providers

Until then, the provider may appear in the connect dropdown as **Adapter coming soon**.

Related

- Generic Webhook
- Channels overview
- Roadmap

docs/channels/whatsapp-cloud.md

WhatsApp Cloud API

Fully supported inbound WhatsApp messaging, lead/conversation creation, and Inbox replies.

Prerequisites

- Meta Developer app with WhatsApp product
- Phone Number ID + Cloud API access token
- `META_APP_SECRET` in CRM `.env`
- Public HTTPS webhook URL (ngrok for local)
- Queue worker on `channels`

Meta links:

- https://developers.facebook.com/apps
- App → **WhatsApp → Try it out / API Setup** (token + Phone Number ID)
- App → **WhatsApp → Production setup / Configuration** (webhooks)
- App Secret: **App settings → Basic**

Connect in CRM

1. `META_APP_SECRET=...` then `php artisan config:clear`
2. **Channels → Connect channel → WhatsApp Cloud API**
3. Fields:
    - Display name
    - **Phone Number ID** (stored as `external_page_id`)
    - Optional **WABA ID** (`external_account_id`)
    - **Access token**
4. Copy **Webhook URL** + **Verify token** from the connection page

Meta webhook

1. Callback URL = `https://YOUR_HOST/webhooks/channels/{uuid}`
2. Verify token = CRM value
3. **Verify and save**
4. Subscribe to **messages**

CRM answers Meta `GET` verification (`hub.challenge`) for WhatsApp connections.

Inbound behavior

- Validates `X-Hub-Signature-256`
- Parses WhatsApp `messages` webhooks

- Creates/updates lead (source `whatsapp`) + conversation + inbound message
- Status-only webhooks (delivered/read) are acknowledged and ignored
- Idempotency: `whatsapp_message_{wamid}`

Inbox reply

1. Open **CRM → Inbox**
2. Open the thread
3. Send reply (permission `reply.inbox`)
4. CRM calls Graph `POST /{phone-number-id}/messages` and stores an outbound message

Assign requires `assign.inbox` (admin by default).

Test connection

Graph lookup on Phone Number ID (`display_phone_number`, `verified_name`).

Local checklist

```
php artisan serve
php artisan queue:work --queue=channels,default
ngrok http 8000
```

Update Meta callback when the ngrok host changes.

Troubleshooting

Issue	Fix
Verify fails	Wrong verify token; CRM/ngrok down; using http://127.0.0.1 in Meta
Test connection fails	Bad token / Phone Number ID / Graph version
Message received in Meta logs but not CRM	Worker not running; signature secret mismatch
Reply errors	Closed conversation; missing token; outside WhatsApp messaging window (Meta policy)

Related

- Channels overview
- End-user manual — Inbox
- Queues

Coding standards

Conventions for PHP/Laravel work in this CRM.

Style

- Use **Laravel Pint** (default Laravel preset; no custom `pint.json` in repo).
- Run before commit:

```
vendor/bin/pint
```

CI should use `vendor/bin/pint --test` (**Recommended** — see CI/CD).

PHP / Laravel

- PHP 8.2+ features are fine (enums, readonly DTOs already used in Channels).
- Prefer constructor property promotion and explicit return types on new code.
- Keep controllers thin; put domain logic in `app/Services`.
- Use Form Requests for non-trivial validation.
- Authorize via policies / `$this->authorize` / `@can` / permission middleware.

Tenancy rules

1. Models with tenant data should use `BelongsToCompany`.
2. Do not bypass `CompanyScope` unless required (webhooks, Super Admin tools).
3. After `withoutCompanyScope()` lookups, set `CurrentCompany` before tenant writes.
4. Never trust client-provided `company_id` for authz.

RBAC rules

1. New abilities → `config/permissions.php` then `php artisan permissions:sync`.
2. Slugs are `{action}.{module}`.
3. Update `RbacSeeder` defaults when a permission should ship to `sales`/`admin`.
4. Cover allow + deny in Feature tests.

Channels rules

1. New providers need an adapter implementing `ChannelAdapter` **and** registration in `ChannelServiceProvider`.
2. Listing a key in `config/channels.php` alone only exposes UI — mark docs as Planned until registered.
3. Webhook HTTP handlers must stay fast; enqueue work on `channels`.
4. Store secrets encrypted on `channel_connections`.

Testing

- PHPUnit 11 (`tests/Unit`, `tests/Feature`).
- Prefer Feature tests for HTTP + tenancy + permissions.
- Fake HTTP for Meta Graph (`Http::fake`).
- `Queue::fake()` when asserting dispatch without processing.

```
php artisan test
# or
composer test
```

Comments

- Do not narrate obvious code.
- Comment only non-obvious constraints (Meta quirks, tenancy escape hatches, migration MySQL limits).

Related

- Contributing
- Architecture

Contributing

Public contribution policy for this multi-tenant CRM.

Short answer: outside contributors are welcome **via pull request**. They are not granted production access, Super Admin, or direct push to `main`.

By opening a pull request you agree to this policy, including the intellectual property terms.

Who can contribute

Who	What they may do
Anyone	Open an issue or a pull request with a **specific** bug fix or feature
Trusted teammates	Same as above; write access to a feature branch is optional and still PR-only into `main`
Maintainer (repo owner)	Reviews, merges, deploys, and holds production credentials

Contributions are accepted when they are scoped, reviewed, and safe for a multi-tenant product. Vague "give me access and I'll help with everything" requests are declined.

You do **not** need to be a collaborator on GitHub to contribute. Fork the repository, make a branch, and open a PR.

How we accept work (PR-only)

1. Fork (external) or create a feature branch from up-to-date `main`.
   Example: `cursor/short-description-a8c9`
2. Implement with tests.
3. Run Pint + tests locally (see Before you push).
4. Open a **pull request against `main`**.
5. Address review. The maintainer merges when ready.
6. Ensure CI passes once workflows exist (CI/CD).

Not allowed

- Direct pushes to `main`
- Force-pushes to shared branches
- Merging your own PR unless the maintainer has explicitly said so
- Asking for admin, maintain, or production deploy rights as a condition of helping

Write access to this repository, if ever granted, is only for pushing **feature branches**. It is not permission to change production or skip review.

What contributors never get

This CRM stores tenant customer data, conversations, and channel secrets (WhatsApp, Meta, webhooks). Contributors work on **code**, not the live system.

Do **not** request or accept:

- Railway / hosting / SSH / database credentials
- Production or staging `.env` files
- Meta App Secret, WhatsApp tokens, webhook signing secrets, mail or payment keys
- Super Admin access on the live product
- Tenant logins, customer exports, or database dumps
- Impersonation of real companies

Use local setup from Installation and `.env.example`. Seeded / demo data is for local development only.

If you accidentally receive a secret, **do not** put it in a PR, issue, or chat log. Tell the maintainer privately so it can be rotated.

Intellectual property

The repository owner retains all rights to this CRM. There is no inbound license that makes your patch automatically open source on your terms.

By submitting a contribution (pull request, patch, or issue that includes code or docs) you confirm that:

1. The work is yours, or you have permission to submit it, and it does not include secrets or someone else's proprietary code.
2. You grant the repository owner a **perpetual, worldwide, irrevocable, royalty-free** license to use, copy, modify, merge, publish, distribute, sublicense, and **relicense** the contribution as part of this CRM, in source or compiled form, for any purpose.
3. Git history may keep your name as author; the owner is not required to credit you in the product UI or marketing.
4. The contribution does **not** entitle you to payment, equity, revenue share, or production access unless a **separate written agreement** says otherwise.
5. You can be asked to change or remove a contribution during review; the owner may decline any PR for any reason.

If you cannot agree to these terms, do not open a pull request.

Security and tenancy

This is a multi-tenant app. A bad change can leak Company A's data to Company B.

- Follow Coding standards (CompanyScope, policies, permissions).
- Cover happy path **and** cross-tenant 403/404 in feature tests.
- Never commit `.env`, tokens, or `docs/demo-credentials.local.md`.
- Keep Super Admin routes in `routes/superadmin.php`.
- Do not use `withoutCompanyScope()` unless the code truly needs it (webhooks, platform tools), and set company context before tenant writes.

See Multi-tenancy and RBAC.

Before you push

```
vendor/bin/pint
php artisan test
# if permissions config changed:
php artisan permissions:sync
```

PR checklist

- [ ] Tenant isolation considered (CompanyScope / policies)
- [ ] Permissions updated + synced if needed
- [ ] Feature tests for happy path + 403/404 cross-tenant
- [ ] No secrets committed (`.env`, tokens)
- [ ] Docs updated under `/docs` when behavior changes
- [ ] Migrations are MySQL-safe (index name lengths)
- [ ] You agree to this contribution policy (access + IP)

Channels / Meta changes

- Document setup steps in `docs/channels/`
- Fake Graph HTTP in tests
- Note queue worker requirements

Super Admin vs tenant

- Keep Super Admin routes in `routes/superadmin.php`
- Do not leak platform queries into tenant controllers without `withoutCompanyScope` intent

Commit messages

Prefer clear imperative subjects:

- `Add Inbox UI with WhatsApp reply support`
- `Fix company settings edit method`

Related

- Coding standards
- Roadmap
- Installation

Configuration

Environment and config reference for the CRM.

Primary source: `.env` / `.env.example`. Most values map into Laravel `config/*.php` files.

After changing `.env`:

```
php artisan config:clear
# production:
php artisan config:cache
```

Application

Variable	Default (example)	Purpose
`APP_NAME`	Laravel	Display name / mail from name
`APP_ENV`	local	Environment
`APP_KEY`	*(generated)*	Encryption (required)
`APP_DEBUG`	true	Detailed errors (disable in production)
`APP_URL`	http://localhost	Must match the URL you open (include port)
`APP_LOCALE`	en	Locale
`LOG_CHANNEL`	stack	Logging
`LOG_LEVEL`	debug	Log verbosity
`BCRYPT_ROUNDS`	12	Password hashing cost

Database & session / cache

Variable	Default	Purpose
`DB_CONNECTION`	sqlite	`sqlite` or `mysql`
`DB_HOST` / `DB_PORT` / `DB_DATABASE` / `DB_USERNAME` / `DB_PASSWORD`	*(commented)*	MySQL settings
`SESSION_DRIVER`	database	Session store
`SESSION_LIFETIME`	120	Minutes
`SESSION_SECURE_COOKIE`	false	Set `true` behind HTTPS in production
`CACHE_STORE`	database	Cache backend

Queue

Variable	Default	Purpose
`QUEUE_CONNECTION`	database	Use `database` in local/prod; tests force `sync`
`CHANNELS_WEBHOOK_QUEUE`	channels	Queue name for channel webhook jobs
`CHANNELS_WEBHOOK_RATE_LIMIT`	60	Per-minute throttle for `/webhooks/channels/{uuid}`

See Queues.

Mail

Variable	Default	Purpose
`MAIL_MAILER`	log	`log` writes to storage logs (not real inboxes)
`MAIL_HOST` / `MAIL_PORT`	127.0.0.1 / 2525	SMTP
`MAIL_USERNAME` / `MAIL_PASSWORD`	null	SMTP auth
`MAIL_FROM_ADDRESS` / `MAIL_FROM_NAME`	hello@example.com / `${APP_NAME}`	From header

Local inbox tip: Mailpit + `MAIL_MAILER=smtp`, `MAIL_HOST=127.0.0.1`, `MAIL_PORT=1025`.

Auth emails (verification / password reset) send immediately (no queue worker required).

Website lead webhook

Server-to-server form ingestion (not the Channels “Website Forms” adapter).

Variable	Purpose
`WEBSITE_LEAD_WEBHOOK_SECRET`	Shared secret header validation — **never** expose in browser JS
`WEBSITE_LEAD_CREATED_BY_EMAIL`	Fallback owner email for created leads
`WEBSITE_LEAD_RATE_LIMIT`	Requests per minute (default 10)

Endpoint: `POST /webhooks/leads/website`

Docs: Website Forms.

Channels / Meta

Variable	Default	Purpose
`META_APP_SECRET`	*(empty)*	Validates Meta `X-Hub-Signature-256` for Facebook Lead Ads & WhatsApp
`META_GRAPH_VERSION`	v21.0	Graph API version
`CHANNELS_WEBHOOK_QUEUE`	channels	Job queue

Per-tenant tokens (page token, WhatsApp token, verify token) are stored **encrypted on each `channel_connections` row**, not in `.env`.

Lead follow-up reminders

Variable	Default	Purpose
LEAD_FOLLOW_UP_REMINDERS_ENABLED	true	Master switch
LEAD_FOLLOW_UP_REMINDER_TIME	08:00	Daily schedule time
LEAD_FOLLOW_UP_DEFAULT_TIME	09:00	Default follow-up time of day
LEAD_FOLLOW_UP_REMINDER_1D_ENABLED	true	Day-before tier
LEAD_FOLLOW_UP_REMINDER_2H_ENABLED	true	Hours-before tier
LEAD_FOLLOW_UP_REMINDER_DUE_ENABLED	true	Due tier
LEAD_FOLLOW_UP_REMINDER_OVERDUE_ENABLED	true	Overdue tier
LEAD_FOLLOW_UP_REMINDER_OVERDUE_REPEAT_DAYS	1	Overdue repeat interval

Task reminders

Variable	Default	Purpose
TASK_REMINDERS_ENABLED	true	Master switch
TASK_REMINDER_TIME	08:15	Daily schedule time
TASK_REMINDER_DUE_ENABLED	true	Due tier
TASK_REMINDER_OVERDUE_ENABLED	true	Overdue tier
TASK_REMINDER_OVERDUE_REPEAT_DAYS	1	Overdue repeat

See Scheduler.

Tenancy

Variable	Purpose
TENANCY_FAIL_CLOSED_WITHOUT_CONTEXT	When set, forces fail-closed (or open) company scoping without context. If unset, production fails closed and local/dev does not.

See Multi-tenancy.

Related config files

File	Role

`config/permissions.php`	RBAC module/action registry
`config/channels.php`	Channel providers + Meta + webhook settings
`config/tenancy.php`	Tenant fail-closed behavior
`config/website_leads.php`	Website webhook settings
`config/lead_reminders.php` / `config/task_reminders.php`	Reminder schedules
`config/adminlte.php`	Tenant sidebar / branding
`config/queue.php`	Queue connections

Deployment

Production deployment guide for the multi-tenant CRM.

Architecture in production

[Diagram omitted in PDF — open this page in the online docs to view it.]

You need **three** long-running concerns:

1. **HTTP** — PHP-FPM / Octane / shared hosting entrypoint
2. **Queue worker** — `channels` + `default`
3. **Scheduler** — `php artisan schedule:run` every minute

Pre-deploy checklist

- [ ] `APP_ENV=production`, `APP_DEBUG=false`
- [ ] Strong `APP_KEY` (never commit)
- [ ] `APP_URL` is the public HTTPS URL
- [ ] `SESSION_SECURE_COOKIE=true` behind HTTPS
- [ ] Real mailer (`smtp` / SES / Postmark / Resend) — not `log`
- [ ] `QUEUE_CONNECTION=database` (or Redis if you configure it)
- [ ] `META_APP_SECRET` set if using Facebook Lead Ads / WhatsApp
- [ ] Database backups configured
- [ ] SSL certificate valid for webhook callbacks

Deploy steps (typical VPS)

```
cd /var/www/crm
git pull origin main

composer install --no-dev --optimize-autoloader
npm ci && npm run build

php artisan migrate --force
php artisan permissions:sync
php artisan config:cache
php artisan route:cache
php artisan view:cache
php artisan event:cache   # if applicable

# restart PHP-FPM / Octane / queue workers after deploy
```

Always run migrations **before** relying on new features. For MySQL, ensure prior channel migrations completed cleanly (partial tables from failed runs must be cleaned up — see Troubleshooting).

Queue worker (required)

Channel webhooks enqueue `ProcessChannelWebhookJob` on the `channels` queue.

```
php artisan queue:work --queue=channels,default --sleep=1 --tries=5 --timeout=90
```

Recommended: supervise with Supervisor / systemd so the worker restarts on failure and after deploys.

Example Supervisor program:

```
[program:crm-worker]
process_name=%(program_name)s_%(process_num)02d
command=php /var/www/crm/artisan queue:work --queue=channels,default --sleep=1 --tries=5 --timeout=90
autostart=true
autorestart=true
user=www-data
numprocs=1
redirect_stderr=true
stdout_logfile=/var/www/crm/storage/logs/worker.log
```

After code deploys: `php artisan queue:restart`.

Details: Queues.

Scheduler (required)

Add a system cron entry:

```
* * * * * cd /var/www/crm && php artisan schedule:run >> /dev/null 2>&1
```

This drives lead/task reminders, trial notices, pruning, and platform alert jobs.

Details: Scheduler.

Web server notes

- Document root: `public/`
- Allow `POST` (and `GET` for Meta verify) to:
  - `/webhooks/leads/website`
  - `/webhooks/channels/{uuid}`
- These paths are CSRF-exempt in `bootstrap/app.php`.
- Prefer HTTPS end-to-end; Meta requires a public HTTPS callback for WhatsApp / Lead Ads.

Zero-downtime tips

1. Put app in maintenance: `php artisan down` (optional).
2. Deploy code + `composer install --no-dev`.
3. Migrate.
4. Rebuild caches.
5. `php artisan queue:restart`.
6. `php artisan up`.

Health checks

Check	How
App up	`GET /up` (Laravel health)
Scheduler alive	Super Admin platform settings / `scheduler_last_run_at` heartbeat (every 5 minutes)
Queues draining	`jobs` table not growing unbounded; worker logs show processed jobs
Webhooks	Channel detail → Recent events; Meta webhook delivery logs

---

Security

- Never commit `.env` or secrets shared in chat — rotate Meta tokens/App Secret if exposed.
- Restrict Super Admin accounts tightly.
- Keep `WEBSITE_LEAD_WEBHOOK_SECRET` server-side only.
- Review failed webhook events and activity logs regularly.

Related

- Installation
- Configuration
- CI/CD (recommended)

Installation

Local development setup for the multi-tenant CRM.

Requirements

Component	Version / notes
PHP	8.2+ with extensions typical for Laravel (`pdo`, `mbstring`, `openssl`, `tokenizer`, `xml`, `ctype`, `json`, `bcmath`, `fileinfo`)
Composer	2.x
Node.js + npm	For Vite frontend assets
Database	SQLite (default) or MySQL 8+
Optional	Mailpit (local mail), ngrok (Meta webhooks to localhost)

1. Clone and install PHP dependencies

```
git clone <repository-url> crm
cd crm
composer install
```

Or use the project setup script (installs deps, copies `.env`, generates key, migrates, builds assets):

```
composer setup
```

2. Environment file

```
cp .env.example .env
php artisan key:generate
```

Set at least:

```
APP_NAME="CRM"
APP_URL=http://127.0.0.1:8000
DB_CONNECTION=sqlite
QUEUE_CONNECTION=database
```

For SQLite, ensure the database file exists:

```
# Windows PowerShell
New-Item -ItemType File -Force database/database.sqlite

# macOS / Linux
touch database/database.sqlite
```

For MySQL, comment out SQLite and set:

```
DB_CONNECTION=mysql
DB_HOST=127.0.0.1
DB_PORT=3306
```

```
DB_DATABASE=crm
DB_USERNAME=root
DB_PASSWORD=
```

See Configuration for the full variable list.

3. Migrate and seed

```
php artisan migrate
php artisan db:seed
php artisan permissions:sync
```

`permissions:sync` upserts permissions from `config/permissions.php` and refreshes default role grants.

4. Frontend assets

```
npm install
npm run build
# or for hot reload during development:
npm run dev
```

5. Run the app

Option A — simple

```
php artisan serve
```

Option B — full local stack (server + queue + logs + Vite):

```
composer dev
```

Queues (required for channel webhooks)

In a separate terminal if not using `composer dev`:

```
php artisan queue:work --queue=channels,default
```

6. First login

After seeding you typically have a default company and users from the database seeders. Create or reset a user as needed:

```
php artisan tinker
```

Confirm you can open:

- Marketing site: `/`
- Tenant CRM: `/login` → `/dashboard`
- Super Admin: `/superadmin` (platform Super Admin users only)

7. Optional: Meta / Channels local testing

1. Set `META_APP_SECRET` in `.env` and run `php artisan config:clear`.
2. Expose localhost with ngrok: `ngrok http 8000`.
3. Connect a channel under **Administration → Channels**.

4. Point Meta webhooks at the ngrok HTTPS URL.

Details: Channels overview.

Verification checklist

- [ ] `php artisan about` runs without errors
- [ ] `php artisan test` passes (or at least Feature suite for your area)
- [ ] Dashboard loads for a tenant admin
- [ ] Queue worker processes a test job
- [ ] Mail: with `MAIL_MAILER=log`, check `storage/logs` — or use Mailpit on port 1025

Common install issues

Symptom	Fix
`No application encryption key`	`php artisan key:generate`
SQLite “unable to open database”	Create `database/database.sqlite` and check permissions
MySQL index name too long (channels migration)	Use current migrations (short custom index names); drop partial tables if a prior run failed mid-way
Channels menu missing	`php artisan permissions:sync`
CSS/JS missing	`npm run build`

More: Troubleshooting.

Modules overview

Tenant CRM feature map (company users). Super Admin is documented separately.

Sidebar map (typical)

Area	Menu	Permission (gate)
CRM	Dashboard	authenticated tenant
CRM	Customers	`view.customers`
CRM	Leads	`view.leads`
CRM	Inbox	`view.inbox`
CRM	Tasks	`view.tasks`
CRM	Reports	`view.reports`
Administration	Channels	`view.channels`
Administration	Website Lead Demo	`website_lead.demo`
Administration	Users	`view.users`
Administration	Roles	`view.roles`
Administration	Activity Log	activity log access
Administration	Company Profile / Settings	company settings access
Account	Profile	authenticated

Exact labels live in `config/adminlte.php`.

Module summaries

Dashboard

`DashboardController` — company KPIs and shortcuts.

Customers

Full resource CRUD, CSV import/export, plan limit on create.

Leads

Resource CRUD, board/kanban updates, convert to customer, activity log entries, CSV import/export, follow-up reminders (scheduler).

Inbox

List/show conversations from channels (notably WhatsApp). Reply (`reply.inbox`), assign (`assign.inbox`), status open/pending/closed. See WhatsApp Cloud.

Tasks

Resource CRUD, status updates, board updates, due/overdue reminders.

Channels

Connect providers, webhook URL, test/sync/retry/disconnect, regenerate secret. See Channels overview.

Reports

Aggregated reporting UI + export endpoints. See Reports.

Users & invitations

User CRUD, activate/deactivate, invite flow, CSV import/export. Subject to plan `max_users`.

Roles

Per-company roles and permission checkboxes (system admin role protected).

Company profile & settings

- **Profile** (`/company`) — read-only overview
- **Settings** (`/company/settings`) — edit identity, address, hours, logo

Activity logs

Tenant audit trail of important actions.

Notifications

In-app notifications (+ email depending on preferences / templates).

Global search

Search across tenant entities (`search.index`, `search.suggest`).

Website Lead Demo

Internal demo of the website webhook payload for permitted users.

CSV import / export

Type	Import	Export
leads	yes	yes
customers	yes	yes
users	yes	yes

Type	Import	Export
tasks	—	yes

Routes under `imports.*` and `exports.*`.

Related

- RBAC
- End-user manual
- Super Admin

Reports

Tenant reporting for CRM metrics.

Access

- View: permission `view.reports`
- Export: permission `export.reports`
- Routes: `reports.index`, `reports.export` (`GET /reports/export/{type}`)
- Controller: `App\Http\Controllers\ReportController`

Sales role includes view + export by default.

UI

Reports in the CRM sidebar opens the reports index. Filters and available charts/tables depend on the current `ReportController` implementation and Blade views under `resources/views/reports/`.

Exports

Export endpoints are throttled (see route middleware `throttle:30,1` on export). Use exports for offline analysis; prefer CSV imports/exports modules for bulk record movement.

Operational notes

- All report data is **company-scoped** via tenancy.
- Heavy exports should be run off-peak; consider queueing in a future iteration if files grow large (**Recommended** enhancement).

Related

- Modules overview
- End-user manual

CI/CD

> **Status: Recommended** — this repository does not currently ship a project-specific GitHub Actions workflow under `.github/` (beyond whatever may exist upstream). The following is the **recommended** pipeline to add.

Goals

- Run automated tests on every pull request
- Enforce code style (Pint)
- Block merge on failure
- Optional: deploy to staging/production on `main`

Recommended GitHub Actions (PR)

```
name: CI

on:
  pull_request:
  push:
    branches: [main]

jobs:
  tests:
    runs-on: ubuntu-latest
    steps:
      - uses: actions/checkout@v4
      - uses: shivammathur/setup-php@v2
        with:
          php-version: '8.2'
          extensions: mbstring, pdo_sqlite, gd
          coverage: none
      - uses: actions/setup-node@v4
        with:
          node-version: '20'
          cache: npm
      - run: composer install --no-interaction --prefer-dist
      - run: cp .env.example .env
      - run: php artisan key:generate
      - run: npm ci
      - run: npm run build
      - run: vendor/bin/pint --test
      - run: php artisan test
```

Recommended deploy job (outline)

On push to `main` (or tags):

1. SSH / container deploy
2. `composer install --no-dev`
3. `php artisan migrate --force`
4. `php artisan permissions:sync`
5. `php artisan config:cache && route:cache && view:cache`
6. Restart PHP-FPM + `queue:restart`

See Deployment.

Local pre-push checklist

```
vendor/bin/pint
php artisan test
```

Related

- Coding standards
- Contributing

Queues

Background job processing for the CRM.

Defaults

Setting	Value
Connection	`database` (`.env QUEUE_CONNECTION`)
Channel webhooks queue	`channels` (`CHANNELS_WEBHOOK_QUEUE`)
Max attempts (channel config)	5
Tests	Forced `QUEUE_CONNECTION=sync` in `phpunit.xml`

Why workers matter

`WebhookIngestionService` dispatches `ProcessChannelWebhookJob` onto the `channels` queue. Without a worker:

- HTTP still returns `202 Accepted`
- Events stay `queued`
- Leads / Inbox messages do not appear

Run locally

```
php artisan queue:work --queue=channels,default
```

Or via:

```
composer dev
```

(`queue:listen` is included in the concurrent `composer dev` script.)

Production

Supervise a long-running worker (Supervisor / systemd). After deploys:

```
php artisan queue:restart
```

See Deployment.

Failed jobs

Inspect `failed_jobs` table / `php artisan queue:failed`.

Channel events also store status + error on `channel_webhook_events` and connection health fields.

Related queues / mail

- Auth emails send synchronously (no worker required).
- Other notifications may queue depending on implementation — keep `default` in the worker list.

Related

- Channels overview
- Troubleshooting

docs/operations/scheduler.md

Scheduler

Laravel scheduler entrypoint: `routes/console.php`.

Cron requirement

```
* * * * * cd /path/to/crm && php artisan schedule:run >> /dev/null 2>&1
```

Without cron, reminders and pruning will not run.

Scheduled tasks

Schedule	Command / job	Notes
Daily @ lead reminder time (default 08:00)	`leads:send-follow-up-reminders --tier=day_before|due|overdue`	Gated by `lead_reminders` config
Hourly	`leads:send-follow-up-reminders --tier=hours_before`	Gated
Daily @ task reminder time (default 08:15)	`tasks:send-reminders --tier=due|overdue`	Gated by `task_reminders`
Every 5 minutes	Platform scheduler heartbeat	Sets `scheduler_last_run_at`
Every 15 minutes	`platform:send-alert-notifications`	`withoutOverlapping`
Daily 09:00	`trials:send-ending-notifications --days=3`	Trial ending notices
Weekly	`activity-logs:prune --days=90`	Retention
Weekly	`notifications:prune`	Retention

Manual commands (not all scheduled)

Command	Purpose
`permissions:sync`	Sync RBAC registry
`platform:optimize-logo`	Logo optimization
`companies:release-deleted-identifiers`	Soft-delete identifier cleanup

Verifying the scheduler

1. Ensure cron is installed.
2. Check Super Admin / platform setting `scheduler_last_run_at` updates roughly every 5 minutes.
3. Temporarily run `php artisan schedule:run` and watch logs.

Related env

See Configuration reminder section.

Related

- Deployment
- Queues

Super Admin guide

Platform console for operators who manage all tenant companies.

Access

- Middleware: `auth`, `active`, `superadmin`
- URL prefix: `/superadmin`
- Route names: `superadmin.*`
- Layout: Super Admin (`sa-app`), not AdminLTE tenant chrome
- Super Admins typically have `company_id = null` and are redirected away from tenant CRM by `EnsureCompanyContext`

Capabilities

Area	Routes (examples)	Purpose
Dashboard	`superadmin.dashboard`	Platform KPIs
Search	`superadmin.search.*`	Cross-tenant search
Analytics	`superadmin.analytics.*`	Companies / leads / customers analytics
Companies	`superadmin.companies.*`	CRUD, status, restore soft-deletes, PDF, CSV import/export
Impersonation	`superadmin.companies.impersonate`, `impersonation.leave`	Act as tenant admin
Plans	`superadmin.plans.*`	Plans, limits, import/export, duplicate, bulk
Super Admins	`superadmin.super-admins.*`	Manage platform operators
Settings	`superadmin.settings.*`	Platform branding, announcement
Email templates	`superadmin.email-templates.*`	Template CRUD, preview, test send
Contact inquiries	`superadmin.contact-inquiries.*`	Marketing contact/demo form submissions
Notifications	`superadmin.notifications.*`	Platform notifications

Companies & soft delete

Soft-deleted companies can be viewed/restored from Super Admin. Identifier cleanup commands exist for releasing emails/slugs (`companies:release-deleted-identifiers`).

Plans & limits

Plans define `max_users` / `max_leads` / `max_customers` (nullable = unlimited) and optional `plan_limits` rows. Tenant creates are blocked by `PlanLimitService` when exceeded.

Contact inquiries

Marketing site contact submissions are persisted for Super Admins (not the same as tenant Channels). Manage under **Contact inquiries**.

Operational heartbeats

Scheduler writes `scheduler_last_run_at` platform setting every five minutes — use it to verify cron is alive.

Security practices

1. Minimize Super Admin accounts
2. Prefer impersonation over sharing tenant passwords
3. Audit impersonation / company changes via activity logs
4. Keep platform settings changes rare and documented

Related

- Multi-tenancy
- Deployment
- Scheduler

End-user manual

Guide for tenant **Administrators** and **Sales** users of the CRM.

Roles at a glance

Capability	Admin	Sales (default)
Customers / Leads / Tasks day-to-day	Yes	Yes (with some limits)
Manage users & roles	Yes	No
Connect channels	Yes (`manage.channels`)	View only
Inbox view + WhatsApp reply	Yes	Yes
Assign inbox conversations	Yes	No
Company settings	Yes	No

Exact rights can be customized under **Roles**.

Sign in

1. Open your company CRM URL.
2. Sign in with email/password.
3. You land on **Dashboard**.

Forgot password uses the email reset flow (requires mail to be configured by your operator).

Customers

Customers → add, edit, search, import/export CSV.

Customers are won/converted accounts (also created when converting leads).

Leads

Leads supports list and board (kanban) views.

Common actions:

- Create / edit lead
- Move status on the board
- Assign (if permitted)
- Log activities
- Convert to customer

- Import/export CSV

Follow-up dates drive reminder notifications when the scheduler is running.

Inbox

Inbox shows conversations from connected messaging channels (WhatsApp today).

1. Open a conversation (unread clears on open).
2. Read the thread; open linked lead if present.
3. **Reply** sends a WhatsApp message when the channel supports it.
4. Admins can **Assign** and change status (Open / Pending / Closed).

Closed conversations must be reopened before replying.

Tasks

Create and update tasks, change status, use the task board.

Due/overdue reminders appear when enabled by your operator.

Reports

Open **Reports** for company metrics. Export if you have `export.reports`.

Channels (admins)

Administration → Channels

1. **Connect channel** — choose provider (Generic Webhook, Facebook Lead Ads, WhatsApp, …).
2. Enter credentials shown on the form.
3. Copy webhook URL / secrets / verify token as instructed.
4. Use **Test connection**, **Sync**, **Disconnect**, or **Regenerate webhook secret** as needed.

Providers marked **Adapter coming soon** cannot fully process events yet.

Detailed setup:

- Generic Webhook
- Facebook Lead Ads
- WhatsApp

Each company connects **its own** WhatsApp/Facebook/forms — data never mixes with other companies.

Users & roles (admins)

- **Users** — invite or create users, activate/deactivate, import CSV.
- **Roles** — adjust permissions. Do not remove critical admin access without a backup admin.

Plan limits may block adding users.

Company profile & settings (admins)

- **Company Profile** — read-only overview (logo, contact, plan badges).
- **Company Settings** — edit profile fields, regional defaults, business hours, logo.
- Saving settings returns you to the profile.

Notifications & profile

- Bell / **Notifications** — in-app alerts.
- **Profile** — password, photo, notification preferences, sessions.

Activity log

Activity Log shows important changes (lead updates, channel events, settings saves) for auditing.

Tips

1. If a webhook test succeeds in the tool but CRM doesn’t update, ask your operator to start the **queue worker**.
2. Never paste access tokens into chat or email.
3. For Meta on localhost, your operator must use **ngrok** (or staging HTTPS).

Related

- Modules overview
- Troubleshooting