DUNWARE SOLUTIONS

Dunware Product Whitepaper

A white-label business operations platform for small teams: customers, sales, dispatch, field tracking, invoicing and refunds, accounting exports, automations, documents and chat in one application, on the customer's own computer, with the customer's own database and accounts, and no artificial intelligence reading their data.

MongoDB or PostgreSQL **Password, fingerprint / face and two-step sign-in** **One-time purchase, all modules** **No AI, no per-seat fee** **Customer self-service portal** **Free Jitsi Meet video calls**

Version: 0.2.0 (feature complete, now including a customer self-service portal and Jitsi Meet video calls; Windows installer built and tested; macOS packaging and code signing not yet available)

Prepared: Thursday, October 8, 2026 · Updated: Sunday, October 11, 2026

Author: Damien Duncombe, Dunware Solutions

Companion documents: *Dunware User Guide* (for staff), *Dunware White-Label Action Plan* (the architecture decision log this paper draws from)

1 Contents

- 2. Executive summary
- 3. Market and positioning
- 4. Product overview
- 5. System architecture
- 6. Database strategy
- 7. Sign-in, roles and permissions
- 8. The Realtime Bus
- 9. Credentials and integrations
- 10. Money: estimates, quotes, billing and accounting export
- 11. Workflows and background jobs
- 12. Field service and technician tracking
- 13. Customer portal
- 14. Module catalog
- 15. Delivery: installers, phones and networking
- 16. Security
- 17. Testing and quality
- 18. Licensing and pricing philosophy
- 19. Known limits, stated plainly
- 20. Technical reference

2 Executive summary

Dunware is a single business-operations application for small, operations-heavy businesses: field-service companies, trades, cleaning and maintenance firms, consultancies and small logistics operators. It began as a hosted field-service CRM (*DunWare CMS*) and has been rebuilt into a **white-label product sold as a one-time, full-feature purchase**, the same way Dunware's school-management product, Campus, is sold. Every module is included. The customer, not Dunware, decides where the data lives and supplies their own credentials for email and card payments.

The product is **feature-complete**: every module described in this paper is built, tested and running, including the full estimate-to-invoice paper trail (estimates, quotes, invoices, recurring invoices, hosted card payments, refunds and credit notes), deterministic automations, GPS shift tracking for field technicians, a customer-facing self-service portal (Section 13), free Jitsi Meet video calls on meetings, chat with phone alerts, role-based permissions, and sign-in with a fingerprint, face or authenticator app. The last dependency on an outside identity service (Supabase) has been removed. The Windows installer has itself been built and run on a real machine; a macOS build and code signing are the only packaging pieces not yet available (Section 15).

20	8	1,033	2
AREAS (MODULES) IN THE APP	JOB ROLES WITH THEIR OWN ACCESS	AUTOMATED TESTS PASSING	DATABASES, ONE INTERFACE, TESTED ON BOTH

WHERE THINGS STAND

The application, every module in it, and the Windows installer are built and verified. This paper says plainly which claims are built and which are not; Section 19 lists the limits that remain, stated as facts rather than promises.

3 Market and positioning

The product's origin is a real, validated complaint: field-service and operations software, with Service Monster the specific competitor that prompted the build, is sold per seat and per module, so a small business's software bill grows every time it hires an office administrator. Dunware's answer is structural: **one flat, one-time price unlocks every module, regardless of how many people or how much data the business has**. There is no seat count anywhere in the model.

Because the customer supplies their own database, mail server and payment accounts, and their own logo and company name, the same platform can be deployed under a different name for a different vertical without Dunware operating any shared infrastructure. A multi-tenant, Dunware-hosted edition is a deliberate, separate, later project; this paper describes the white-label edition.

Two further positions are deliberate. **No artificial intelligence:** nothing in Dunware sends data to a model or guesses on the user's behalf, which removes a category of privacy, cost and reliability questions; automations follow exact rules the customer writes. **No live accounting connection:** Dunware produces the files QuickBooks and Xero import, and stops there, so no accounting credentials are ever held.

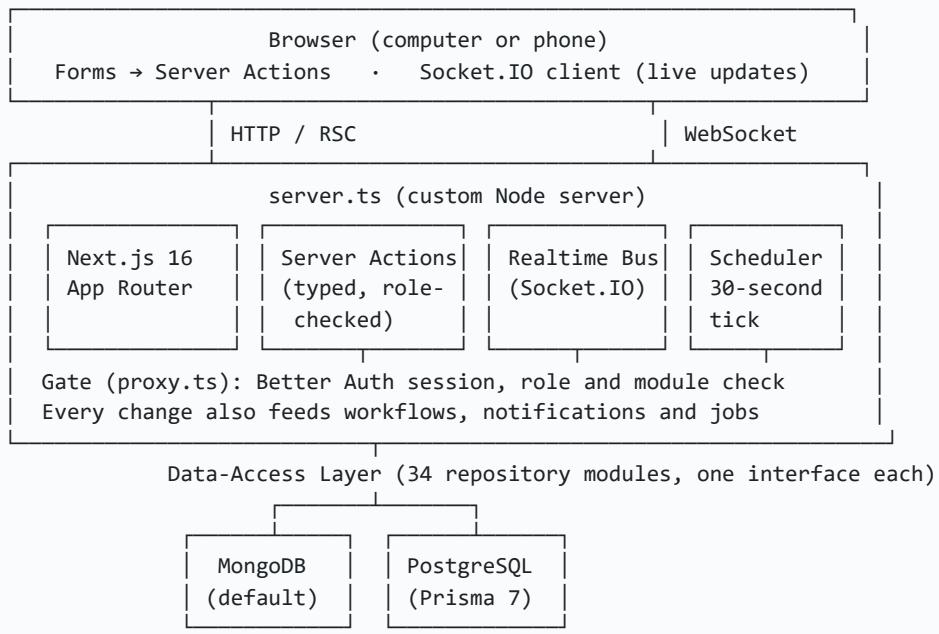
4 Product overview

Dunware puts the tools a small business would otherwise buy separately into one authenticated application: a CRM (customers, leads), coordination (tasks and a Kanban board, meetings, calendar, timeline, projects), field dispatch (service requests, routes, stops, completions, technician GPS shifts), money (invoices, card payments, refunds, credit notes, accounting export), records (documents with versions and PDF, onboarding plans), people (team directory, roles, chat, email log) and insight (analytics, saved reports). A separate, narrower **customer portal** (Section 13) gives the business's own customers a self-service login, isolated entirely from the staff application. On top sits a deterministic automation engine.

It is built for **non-technical staff**. Every screen is task-oriented, each person sees only the areas their job needs, and a 78-page illustrated *User Guide* ships with the product. It is usable on a phone, which is where field technicians work.

5 System architecture

A Next.js 16 (App Router) application: React 19 on the client, typed Server Actions on the server, and a small custom Node server (`server.ts`) that runs the framework, a Socket.IO Realtime Bus and an in-process scheduler on one port.



Every change goes through a Server Action that checks who is asking and what their role allows, writes through the Data-Access Layer, and announces the change on the Realtime Bus. That announcement also drives the workflow engine and the notification bell, so an automation reacts to a change exactly when a person's screen updates. The outer gate reads a signed session cookie, so a page or address outside a person's role is refused before any code runs, not merely hidden from the menu.

6 Database strategy

The customer chooses their database once, at set-up, and the rest of the application never knows which. That is done with a **Data-Access Layer**: one repository interface per area (31 of them), each with two production implementations.

Backend	Role	Implementation
MongoDB	The default.	The native driver, no ORM; money is stored as exact whole cents.
PostgreSQL	The alternative, for a customer who already runs it or wants a relational database.	Prisma 7 with the `pg` driver adapter, 104 tables, versioned migrations; money is stored as exact decimals.

Both adapters are held to the same test suite, run against **real, throw-away instances** of each database (an in-memory MongoDB and an embedded PostgreSQL), not mocks. Billing, workflows, notifications, chat, tracking and authentication all have such parity tests, so "works on both" is verified rather than assumed. Ids are plain strings at the boundary; each adapter converts to its native form internally.

7 Sign-in, roles and permissions

Authentication

Authentication is self-hosted with **Better Auth** and stores its data in whichever database the customer chose. Sessions are HttpOnly cookies, never browser-stored tokens; passwords are hashed with Argon2id. There is no public sign-up: the first account becomes the Owner and everyone else is added by an Owner.

Password

Argon2id, at least 10 characters, changed by the person in My profile; an Owner can reset it.

Fingerprint / face (passkeys)

A phone or computer's own biometric or screen lock signs the person in with no password. Several devices per person; an Owner can remove a lost device.

Two-step (authenticator app)

A 6-digit number after the password, with ten one-time recovery codes. Optional, required for Owners, or required for everyone, set by the Owner.

Recovery and reset

Recovery codes cover a lost phone; an Owner can reset two-step sign-in or remove passkeys, which also signs the person out everywhere.

WHY THIS STANCE

A password alone is the weakest link in a small business. Passkeys make the safest option the easiest one on a phone, and the company can require two-step sign-in without any outside service. Signing in with a passkey counts as the second factor. Passkeys need a secure (https) address, which the installer provides.

Roles and what each may do

Each person has one of eight **roles**. A role decides which areas they can open, and an Owner can add or remove areas for one person. An area outside a person's list is not drawn in the menu and is refused at the gate if its address is typed.

Role	Opens by default
Owner	Everything, including Settings. Cannot be restricted; there is always at least one active Owner.
Manager	Everything except Settings.
Dispatcher / office admin	Customers, tasks, route planning, services, timeline, meetings.
Sales / CRM	Customers, leads, meetings, documents, tasks.
Billing / accounts	Billing, customers, documents, reports.
HR	Team directory, onboarding, documents.
Field technician	Their own route, tasks and projects, on a phone.

General staff	Dashboard, chat, email and calendar only, plus whatever an Owner adds.

Seeing an area and **changing** things in it are separate. A small capability table gives each kind of change to the roles whose job it is: route planning to Owner, Manager and Dispatcher (and a technician may record completion of stops on their own route while on shift); the price list to Owner, Manager and Dispatcher; onboarding plans to Owner, Manager and HR; deleting a lead to Owner, Manager and Sales; deleting a customer or project to Owner and Manager. Settings, adding team members and resetting sign-in stay with the Owner. The table is unit-tested and checked in a browser.

> **A NINTH, SEPARATE KIND OF LOGIN**
>
> A customer portal account (Section 13) is not one of the eight staff roles above and never enters the staff module system at all: the outer gate routes it to the portal or nowhere, before role checks even run, so a portal session can never reach a staff page, Server Action or API address.

8 The Realtime Bus

The requirement that any save is reflected to every connected user within a 200–400 millisecond target and a 500 millisecond ceiling cannot be met by watching the database, which works for only one of the two databases. Instead a Socket.IO server, attached to the same port as the application, broadcasts each change the moment its write commits, so it behaves identically on either backend. Lists, counts, the Kanban board and the notification bell update on their own, and a latency test measures a real socket round trip against the ceiling in every test run. Every socket connection authenticates on handshake before it is accepted. Most events still go to every connected staff client (filtered in the browser, acceptable among staff who already see most of each other's work — Section 13), but Chat is the one exception: a message is emitted only to the room of its own conversation's participants, never to every connected client, so a colleague not in that conversation never receives it even transiently.

9 Credentials and integrations

Every external service follows one rule: **the customer supplies their own credentials, entered in Settings, encrypted at rest, never shown again**. There is no shared Dunware account and no shared API bill.

Capability	Default	Customer's option
Database	MongoDB	PostgreSQL
Maps	Leaflet with OpenStreetMap, free, no key	Other providers, later
Email	*None* until configured	Any SMTP server; test buttons in Settings
Card payments	*None* until configured	Stripe, Square, or both

Video calling	*Off* until switched on	Jitsi Meet, free public server (`meet.jit.si`), no key; a self-hosted Jitsi server instead, by changing one setting
Accounting	File export, no connection	QuickBooks Online, QuickBooks Desktop, Xero

Secrets are stored with AES-256-GCM using a key file kept outside the database and never logged; it must travel with every backup, and Settings tells the Owner plainly if a restore lost it.

10 Money: estimates, quotes, billing and accounting export

All money is exact to the cent (whole cents in MongoDB, decimals in PostgreSQL), with per-tax rounding so a document's lines and taxes always add up. Every document (estimate, quote, invoice and credit note) states explicitly whether its prices include tax, exclude it, or carry none at all — never left for the reader to assume.

Estimates

A non-binding ballpark figure (EST-1001 and up), sent for a customer's comment, then approved, declined or converted. Converting it straight to an invoice for substantially more than the estimated figure asks for one explicit confirmation first (a configurable percentage, set in Settings); converting it into a quote instead locks the price in. The estimate itself is never rewritten — it stays as issued and is marked converted, with a link to what it became.

Quotes

A binding, time-limited fixed price (QT-1001 and up), either standalone or carried forward from an estimate. Accepting one records who accepted it and when. Converting an accepted quote into an invoice copies its lines and total exactly — a quote already promised a price, so nothing is re-priced at that step.

Invoices

Lines with quantity and price, lines marked not taxable, several itemised taxes (GST, PST, HST, QST, compound where it applies) from an editable list, the business's and the customer's tax numbers, customer tax-exempt status, a PDF on the company letterhead, statuses Pending, Partial, Paid, Overdue, Credited and Canceled. An invoice converted from an estimate or quote keeps a link back to it.

Recurring plans

A plan raises the same invoice for a customer on a schedule (weekly to yearly), with an optional end date and an emailed notice; a period is reserved before its invoice is made so it can never be billed twice, even if several runs overlap. No card is ever charged automatically by this on its own.

Payments

Record cash, cheque and transfers by hand; take cards through Stripe or Square hosted checkout so card data never reaches Dunware (the lightest PCI tier, by design), including a hosted, no-sign-in **pay link** that can be emailed with an invoice. A payment can never exceed what is owed.

Refunds

Refund up to what is left of a payment. Stripe and Square refunds go through the provider and show Pending, Succeeded or Failed; the Stripe webhook also records refunds made in the provider's own dashboard. Idempotent, so a repeat click cannot refund twice.

Credit notes

Numbered CN-1001 and up, with taxes, from an invoice or on their own; apply to invoices, pay out, or void while unused; each has a PDF. Reduces what is owed and updates the invoice status.

Receipts

A printable confirmation of one payment against an invoice, on the same letterhead. Not a separate record — a view over the payment already on file.

Accounting export

Pick a provider, then a format from its list, a date range and a date style, and download the file: QuickBooks Online (customers, invoices, credit memos, payments, estimates and quotes for an import tool), QuickBooks Desktop (IIF: invoices, credit memos, payments, and a genuine Estimate transaction type covering both estimates and quotes), Xero (contacts, sales invoices, payments and refunds, credit notes and a quotes list). Account and tax names are remembered.

Currency

Invoices are issued in a chosen currency and keep it forever; changing the setting never alters an old invoice.

STATED PLAINLY

The export files follow each program's published import layout and are covered by golden-file tests, but they have not been test-imported into a live QuickBooks or Xero here. QuickBooks Online has no built-in import for estimates, credit memos or payments by itself, and Xero has none for quotes or sales credit notes — those files are made for an import tool such as SaasAnt, or for keying in by hand; this is stated on the export screen itself, not only here. The recommendation to customers is to import the first file into a test company or have their accountant check it.

11 Workflows and background jobs

A **workflow** is one trigger, optional conditions and up to ten steps, each with an optional wait. It is fully deterministic: no model is consulted, nothing needs an outside credential, and it works offline.

Part	What it can be
Trigger	A task, lead, customer, invoice, estimate, quote, service request, meeting or technician shift is created or changes status (optionally to a chosen status); time passes (invoice overdue N days, task due in N days, lead quiet for N days); or a daily or weekly schedule on the company clock.
Conditions	Field checks on the record, all-of or any-of.
Actions	Create a task, send an email, notify a person, change status, assign, create a lead, create a service request for the record's customer, post a chat message to a person or group. Text uses `{{fields}}` from the record; no code runs.

SAFETY BY CONSTRUCTION

A workflow's own changes never start another workflow (no loops). Each workflow fires once per record (a unique run key). Waits are stored in the database, so a restart never loses or repeats a step, and after a wait the record is re-checked (a paid invoice stops its reminders). More than 100 runs in an hour switches a workflow off with a reason and tells its creator. Workflows can create and change records but never delete. Every run keeps a plain log, and **Try it on a record** previews the effect without sending anything.

Six ready-made templates ship: chase an overdue invoice, follow up a new lead, task due tomorrow, tell dispatch about a new request, tell dispatch when a technician switches tracking off, and wake up a quiet lead.

Routine jobs

Independent of workflows, the application runs its own housekeeping on both databases from the same 30-second scheduler: reminder emails for tasks and meetings, bell notifications, marking invoices overdue with reminders at 1, 3, 7, 14 and 30 days, customer-health checks that raise tasks for at-risk customers, weekly clean-up, and closing technician shifts that ran past the end of the day. Each job remembers what it has done, so a restart or second run never repeats an email. External schedulers can call the same code through cron addresses.

12 Field service and technician tracking

Service requests become stops on a technician's route; the technician marks stops as they go and records the work, hours and materials for each completion. Dispatch sees a live map of who is on shift.

COMPLETING A JOB: NO TYPED PRICE, NO ARITHMETIC

The technician never types a price or adds anything up. The service price is read automatically from the job's own service type; anything beyond it — a part, a material, a named extra service — is picked from a catalog (Route Manager, Parts & Materials) with a quantity stepper and an optional "No charge" toggle for a complimentary replacement. Every real price is resolved server-side from that catalog, never trusted from the technician's device. Completing a job creates its invoice immediately and unconditionally — including a $0 one for warranty work, marked paid automatically — so Billing is never a "did we get paid for this" chase; a billing-side failure is logged and the customer's rep is notified rather than discovered later by someone noticing a job with no invoice.

- **Shifts.** A technician starts a shift with a clear consent screen and shares their phone's location about every 30 seconds (or every 25 metres) until the shift is over and they are back at the depot, or until they switch it off with a reason (lunch, personal, battery, vehicle, other, or done for the day). Dispatch sees a switched-off shift and why. A technician sees their routes only while their shift is on.
- **Company clock.** Shift hours and the "at the depot" distance follow the company's time zone, whatever a phone's.

- **Chat alerts.** Each device chooses to follow the phone's own setting, always alert with a chime and vibration, or show only a count.
- **Retention.** Individual location pings are deleted automatically after 90 days; shift records (start/end times, not the moment-by-moment trail) are kept longer for payroll and dispute records.

This is employee location monitoring, and the product treats it as such: a one-page written-policy template — what is collected, why, who can see it, and that a customer can briefly see it too (Section 13) — ships with the installer. It is a starting point for the business, not legal advice; several jurisdictions, including Ontario at 25 or more employees, require a written policy of this kind.

HONEST LIMIT

A web page cannot keep sharing location while a phone's screen has been off for a long time. Dispatch can see when a shift stopped reporting. Uninterrupted screen-off tracking would need a small native companion app, which is not part of this release.

13 Customer portal

A separate, narrower application at its own address, for the business's own customers to sign in — staff issue the first password from the customer's own page in Customers (Section 14) and can only clear it afterward, never see or set a new one; the customer changes it themselves, by self-service or a one-time forced reset. **Isolation is structural, not a setting**: a customer login is not one of the eight staff roles (Section 7) and the outer gate routes it to the portal or nowhere, before the staff module system runs at all — a portal session cannot reach a staff page, Server Action or API address, by construction rather than by a permission check that could be misconfigured.

Overview

A home page summarising what's owed and on how many invoices, anything awaiting the customer's own decision, and their most recent invoices and service jobs.

View

Invoices, quotes, estimates, service history and locations — narrow, allowlisted views over the same records staff see, never internal notes, margins or technician remarks.

Pay

Mints its own Stripe/Square checkout session for an invoice (never the anonymous emailed pay-link's own cache), with an optional tip, settled by the same webhook either way.

Accept or decline a quote

An atomic, idempotent transition stamped with the customer's own account and request IP — provably distinct from a staff member recording a phone acceptance.

Request more work

A free-text request at one of their own locations; staff must explicitly convert it into a real service request — the portal can never create one directly.

Live technician location

A single current-position dot (never a trail) for their own job only, and only once the technician has actually *left* the previous customer's address — not merely once that stop's status flips to done, which would otherwise show the live position still sitting at a different customer's property for the first minute or two. Access ends the instant their own job is marked complete.

Message the technician

A short, rate-limited, plain-text channel open only during that same en-route window (useful for a gate code or a parking note); the history stays readable afterward, kept 30 days.

WHY NOT THE REALTIME BUS

Most staff features (live lists, counts, the Kanban board) run on a Socket.IO broadcast to every connected staff client, filtered only in the browser — acceptable among staff who already see most of each other's work (Chat is the one staff feature scoped to a room of participants instead, for the same reason — Section 8, above). None of that is acceptable for a customer-facing read, so a customer's browser never opens a socket connection to the Realtime Bus at all. Both portal features above are instead plain REST, polled every 15 seconds, re-checking eligibility fresh on every call, with nothing cached to leak and nothing to invalidate when a job finishes.

DELIBERATELY NOT BUILT

Self-service online booking or quote requests. Most field work needs a time window sized to the job, and nothing in the schema that could drive that (a technician-availability table, several unused scheduling columns) is read or written anywhere in the application — it is dead scaffolding from an earlier version. Building real slot-aware booking would be a from-scratch scheduling engine, not a wire-up, so "Request more work" (above) is the answer for this release: a request a human converts, not a self-booked slot.

14 Module catalog

Dashboard

Today's tasks, projects, meetings, team and counts, showing only what the person's role allows.

Customers

One primary contact (company/contact/email/phone) plus any number of additional Contacts (name, title, email, phone); tax details, owner, service locations, invoices, service history, tasks, projects, meetings, documents and emails on one page; its own search box and status filters; archive instead of delete; issue or clear a portal login (Section 13) from the same page; bulk-import customers from a CSV, TSV, XLSX, JSON or XML file with column mapping, a dry-run preview and one-click undo (Section 16).

Leads

Pipeline with source, value, priority and status; its own search box, status filters and a drag-and-drop Board view alongside the table; one-click conversion to a customer; bulk-import from a file, same as Customers.

Tasks and board

Assigned tasks with priority and due date, and a drag-and-drop Kanban board with a menu fallback for touch screens.

Meetings and calendar

Organiser, attendees, customers and leads, reminders by email and bell, an optional free Jitsi Meet video-call room (Section 9) with an employee-only T-5-minute prep reminder and an automatic T-2-minute launch; a month calendar with drag to reschedule.

Services

A simple price list; bulk-import from a file, same pipeline as Customers.

Billing

Section 10.

Projects

Budget, target dates, hours logged, notes, task list and health at a glance; progress is calculated from the project's own tasks, not typed in by hand, once it has any; a Gantt view with drag-to-move/resize, zero-duration milestones and "must finish before" task dependencies that cascade downstream dates automatically.

Documents

Proposals, contracts, letters, memos and reports with versions, history, statuses and letterhead PDFs.

Onboarding

Reusable plans and per-customer checklists with progress.

Route Manager

Requests, routes, stops, schedules, locations, service types, a parts & materials catalog, completions (priced automatically, no technician arithmetic), the live field-team map, a Records screen reviewing every job's customer↔technician messages and every route's activity log (searchable, across all jobs and routes at once), and bulk-import for locations, service types and parts & materials, each from its own file.

Chat

Direct and group conversations, reactions, edit and delete, unread counts and phone alerts.

Email

Send from Dunware through the company's own mail server and keep a log, with a real message view (reply, reply all, forward, resend a failed message) over that log. It is deliberately not an inbox. Any employee may optionally connect their own mail account from their profile to send (and be logged) as themselves instead of through the shared one.

Team

Directory, workload figures, roles and per-person access; Owner tools for passwords, two-step and passkeys.

Timeline

Swimlanes per person across tasks, meetings and live projects. A Field technician sees a different view instead: their own route and stops for the current shift, with an optional toggle to the projects they're part of.

Analytics

Completion, on-time rate, task distribution, team performance heat grid and workload.

Reports

Task, project, team and sales-pipeline reports saved as snapshots, with PDF and CSV.

Workflows

Section 11.

Global search

Ctrl+K across customers, leads, tasks, projects, meetings, documents and people.

Settings

Company, email, two-step policy, shifts, currency, taxes, Jitsi Meet, Stripe and Square (Owner only).

Deliberately not included: an AI copilot.

15 Delivery: installers, phones and networking

The product installs like a desktop application. The Windows installer (`Dunware-Setup.exe`, built with Inno Setup) bundles the application, a portable Node runtime and a portable PostgreSQL (the same approach as Campus), with Caddy providing local HTTPS and NSSM registering the parts as Windows services; it also installs backup, restore and repair scripts, the illustrated User Guide and the employee monitoring policy template (Section 12), each with its own Start Menu shortcut. A **Remote Access Setup** companion window,

opened from the Start menu, turns on a Cloudflare Tunnel, a WireGuard VPN or a custom domain with Let's Encrypt.

Item	State
Application, both databases, documents, user guide	**Built and tested**
Windows installer, service registration, backup/restore/repair scripts	**Built; run successfully on a real machine**
Remote Access Setup companion window	**Built; Cloudflare Tunnel, WireGuard and custom-domain modes tested**
Payments Setup companion window	**Not built yet**
macOS disk image	**Not built — can only be produced and signed on a Mac**
Code signing (SmartScreen / Gatekeeper)	**Not done; the installer runs today with an unsigned-publisher warning**

THE ONE REAL INSTALL DONE SO FAR

Running the installer on an administrator's own Windows machine surfaced two faults, both fixed in the rebuilt installer: a custom computer name the browser could not resolve (setup now writes a hosts-file entry and keeps the pre-filled name), and a port already in use by something else (setup now finds a free port itself and points every service at it instead of silently failing). No install has yet been run on a second, independently prepared machine.

Phones and other devices

Any phone or computer on the same network opens the application in its browser; the layout adapts, with a menu drawer on a phone. Fingerprint sign-in and location sharing both require a secure address, which the installer's local HTTPS provides. The application is installable to a phone's home screen as a real app (a web manifest and icons, with a background service worker that keeps chat alerts working offline); this has been checked on the packaged build but not yet installed on a physical phone. Built for Android; iPhone support was a deliberate decision not to build.

Reaching it away from the office

Mode	What it gives
Local network (default)	Devices on the same Wi-Fi open it with no set-up.
Cloudflare Tunnel (optional)	Access from anywhere with no router changes.
WireGuard (optional)	A private VPN for the security-conscious.

All three are priced the same: a privacy choice, not a value ladder. None of the three has yet been exercised against a real Cloudflare account or a real WireGuard peer outside this build's own tests.

16 Security

- Argon2id password hashing; HttpOnly cookie sessions; optional or required two-step sign-in; passkeys.
- Role and module checks at the outer gate, then again in every action, deny by default. API addresses return 401 when not signed in.
- AES-256-GCM encryption of every saved credential, with the key held outside the database and carried through backups.
- Secrets are write-only in the interface. Card data never touches the application (hosted checkout only).
- A payment gateway webhook is checked by signature, with a five-minute time window and idempotent handling.
- Deactivating a person signs them out everywhere at once and blocks new sign-ins.
- Money rules are enforced in the data layer, with atomic claims so two clicks cannot double-refund or double-apply a credit.
- The customer portal is isolated at the outer gate, not by a permission check inside the staff application (Section 13); a narrow, allowlisted read layer stands between it and the staff data-access repositories, so a page cannot accidentally return an internal field to a customer.
- A live technician position is withheld from a customer until the technician has actually left the previous customer's address, not merely once that stop is marked done — otherwise the very first positions shown would be the previous customer's own property (Section 13).

> **NOT IN THIS RELEASE**
>
> There is no audit log of who read or changed what. Bulk data-import (CSV, TSV, XLSX, JSON or XML, with format sniffing by content rather than file extension, a worker-thread-isolated parser with hard size/entry/ratio caps against zip-bomb and entity-expansion attacks, column mapping, a dry-run preview and one-click undo) is built and wired up for all six Tier-1 targets: Customers, Leads, Service Locations, Services, Service Types and Parts & Materials. Section 19 lists the application's other current limits.

17 Testing and quality

The product's standard is *no room for errors*, held by automation rather than by hand:

- **1,149 automated tests in 103 files** pass (1 skipped: a MongoDB-only case, in a test that runs against both databases, for a one-time upgrade that backend alone ever needed), alongside strict type-checking, lint and a full production build. Verified by actually running the suite on 2026-10-11, not carried over from an earlier count.
- **Cross-database parity**: the same repository tests run against real MongoDB and real PostgreSQL for billing, workflows, notifications, chat, tracking, employees and authentication.
- **Money to the cent**: exact-arithmetic tests for taxes, refunds, credits and exports, including refunds racing each other.

- **Real services, not mocks**: the workflow engine, background jobs, two-step sign-in and passkey storage run against real databases; the payment gateways run through injectable fetchers with their signature and idempotency rules.
- **Browser checks** in headless Chrome for the flows that matter: adding and using a passkey with a virtual authenticator, two-step enrolment with a real generated code, the phone shift screens, permission gates, and the guide's own screenshots.
- **Realtime latency** measured over a real socket against the 500 ms ceiling.

> **WHAT HAS NOT BEEN VERIFIED HERE**
>
> A physical phone's fingerprint or face sensor (only the browser's standard test authenticator was used); export files imported into a live QuickBooks or Xero; the phone app installed on a physical phone; the Windows installer on a second, independently prepared machine; a macOS build, which needs a Mac and has not been attempted.

18 Licensing and pricing philosophy

A flat, one-time licence unlocks every module, and **nothing about the price is tied to headcount, user count or data volume**. Price may vary only for genuinely distinct one-time work such as data migration, training or custom branding, never for how many people use the software. The three network-access modes are priced identically because the set-up effort is the same.

19 Known limits, stated plainly

- Converting a lead copies only company, contact, email, phone and owner to the new customer — the source, estimated value, notes and address do not carry over.
- Meetings send reminders but not formal, RSVP-style invitations; Email is an outgoing log, not an inbox.
- Card payments need a Stripe or Square account, and the Stripe webhook must be set up for refunds and payments to be confirmed automatically.
- Export files are not live connections and have not been test-imported into QuickBooks or Xero here; several of the new formats (QuickBooks Online estimates/credit memos/payments, Xero quotes/sales credit notes) are not natively importable on either platform and are built for an import tool or for keying in by hand.
- Recovery codes are shown once and can only be replaced, not re-shown; there are no SMS or email codes.
- Passkeys belong to one web address: changing the address means adding devices again.
- Screen-off location tracking on a phone needs a native companion app, which does not exist.
- Settings and the team stay with the Owner by design; a Manager or Dispatcher cannot change them.
- There is no audit log of sensitive reads or writes.
- The database adapters are MongoDB and PostgreSQL only; no MySQL, SQL Server or SQLite adapter exists.

- No macOS build and no code signing on either platform; the installer currently shows an unsigned-publisher warning on Windows.
- The Payments Setup companion window does not exist; Stripe and Square are configured directly in Settings instead.
- The Windows installer has been built, installed and reinstalled repeatedly on one real machine through active development and testing; it has not been run on a second, independently prepared machine.
- Jitsi Meet's default server (`meet.jit.si`) is a free, third-party-hosted service (operated by 8x8, not Dunware) — a call's audio/video passes through it unless the business points the one setting at a Jitsi server it runs itself.
- A customer watching their technician's live position on the portal sees only a current dot, never a computed arrival-time estimate; building a real ETA would need route/traffic data this release does not use.
- Self-service online booking and online quote requests are not built (Section 13); "Request more work" is a request a staff member converts by hand, not a self-booked appointment.
- The employee monitoring policy that ships with the installer is a starting-point template, not legal advice, and has not been reviewed by employment counsel in any specific jurisdiction.
- The Parts & Materials catalog is pricing only — there is no stock-on-hand count, receiving workflow or low-stock alert; a real inventory system was a deliberate scope decision, not an oversight.

20 Technical reference

Item	Detail
Framework	Next.js 16.3 (App Router, Turbopack), React 19, Tailwind CSS 4, TypeScript
Data	MongoDB native driver; Prisma 7 with PostgreSQL; 34 repository modules; 107 tables
Auth	Better Auth 1.7 with the two-factor and passkey plugins, Argon2id
Realtime	Socket.IO on the application's own port; 30-second in-process scheduler
Payments	Stripe and Square hosted checkout; injectable fetchers for tests
Documents	pdfkit letterhead PDFs for estimates, quotes, invoices, receipts, credit notes, documents and reports
Maps	Leaflet with OpenStreetMap
Size	About 52,000 lines of application code (65,000 including tests), 1,149 automated tests
Environment	`DUNWARE_DB_BACKEND`, `MONGODB_URI` or `DATABASE_URL`, `BETTER_AUTH_SECRET`, `BETTER_AUTH_URL`, `DUNWARE_ENCRYPTION_KEY`, `CRON_SECRET`, optional SMTP defaults