VELAR PLATFORM

TECHNICAL DUE DILIGENCE

CTO / Technical Buyer Review Document
August 2026

Review principle

This document intentionally distinguishes documented/current architecture from production requirements and future development. It is designed to support technical review without overstating implementation maturity.

01 Due Diligence Scope

VELAR Platform is an Electron + React desktop CRM/ERP application with a modular AI Core. The technical review should focus on source-code integrity, runtime behavior, architecture boundaries, build reproducibility, dependency posture and the gap between current application functionality and a future production SaaS deployment.

AREA	REVIEW FOCUS	STATUS POSITION
Source code	Application modules, components, services and architecture.	Existing / Reviewable
Frontend	React, Vite, JSX/JavaScript, routing, charts, icons and CSS.	Existing
Desktop	Electron main process, preload, IPC, windows and packaging.	Existing
AI Core	Context, events, memory, intelligence, knowledge, actions, automation and workflows.	Existing architecture / mixed maturity
Data layer	clients, contacts, deals, tasks, activity and related application state.	Existing application layer
Build	Vite production build and Electron Builder packaging.	Verified current status
Cloud / SaaS	Multi-tenant infrastructure, cloud database and hosted services.	Future Development
AI model infrastructure	Production model serving / proprietary trained LLM.	Future Development / not claimed

02 Project Structure

A technical buyer should review the repository as a modular application rather than as a single monolithic feature. The exact folder/file inventory should be verified against the transferred source package at handover.

LAYER	EXPECTED / DOCUMENTED RESPONSIBILITY
Frontend	React application, routes, views, UI components and state presentation.
AI Core	Context, event, memory, decision, learning, reasoning, recommendation, knowledge and action modules.
Automation	AutomationEngine, AutomationRules and AutomationExecutor.
Workflow	WorkflowEngine, Runner, Registry, Context, Condition, Action, History, Scheduler, Loader and ClientFollowUpWorkflow.
Electron	Main process, preload, IPC, window management and desktop integration.
Build	Vite build pipeline and Electron Builder packaging.
Data	Business entities including clients, contacts, deals, tasks and activity.

03 Frontend Technology

TECHNOLOGY	ROLE	DUE DILIGENCE NOTE
React	UI framework.	Review component boundaries, state flow and reusable UI patterns.
Vite	Development/build tooling.	Production build is documented as successful.
JavaScript / JSX	Application language.	Review typing strategy, runtime validation and maintainability.
React Router	Client-side routing.	Review protected routes and navigation state.
Chart.js	Analytics/chart rendering.	Review data preparation and chart lifecycle.
Lucide React	Icon system.	UI dependency; verify version/licensing in package manifest.
CSS	Visual styling.	Review global scope, maintainability and responsive behavior.

04 Electron Layer

The desktop shell is based on Electron and separates the browser-facing application from privileged desktop capabilities through preload and IPC boundaries.

COMPONENT	ROLE	TECHNICAL REVIEW
Main process	Application lifecycle and privileged desktop control.	Inspect window lifecycle, permissions and error handling.
Preload	Bridge between renderer and controlled Electron APIs.	Inspect exposed API surface and isolation posture.
IPC	Communication between renderer and main process.	Inspect channel naming, validation and authorization.
Window Management	Creation and lifecycle of application windows.	Inspect lifecycle, cleanup and edge cases.
AI Memory Integration	Desktop bridge for AI-memory-related operations.	Verify actual handlers and persistence behavior.
AI Dashboard IPC	IPC boundary for AI dashboard operations.	Verify payload validation and failure handling.
Learning IPC	IPC surface for learning-oriented operations.	Verify data flow and outcome semantics.
User IPC	IPC surface for user operations.	Verify validation and access boundaries.
Build & Packaging	Electron Builder packaging for Windows.	Verify reproducibility and signing strategy.

05 AI Core

The AI Core is modular by responsibility. Its presence in the architecture should not be interpreted as evidence of a proprietary model, autonomous agent or production-grade model serving stack.

MODULE	CURRENT ROLE	EXTENSION / REVIEW
ContextEngine / ContextBuilder	Context assembly and orchestration.	Review context freshness, size and source-of-truth rules.

AIPageTracker / Provider / Watcher	Application context tracking and propagation.	Review event frequency and lifecycle behavior.
AIEvent / AIEventBus	Event-oriented AI communication boundary.	Review event contracts, ordering and failure handling.
MemoryStore / DecisionMemory	Memory and decision-oriented storage abstractions.	Review persistence and retrieval semantics.
PersistentMemory	Persistence-oriented abstraction.	Verify actual backing store; do not infer production DB.
MemoryListener / Optimizer	Observation and memory organization.	Review consistency and optimization rules.
DecisionEngine	Decision-oriented orchestration.	Review inputs, outputs and deterministic behavior.
LearningEngine	Feedback-oriented learning / outcome tracking.	Not a claim of model training or proprietary LLM learning.
ReasoningEngine	Reasoning-oriented processing boundary.	Verify actual implementation and model dependency.
RecommendationEngine	Recommendation orchestration.	Review explainability and fallback behavior.
KnowledgeGraph / entities	Structured knowledge representation.	Review schema, persistence and graph operations.
ActionEngine	Action execution boundary.	Review permissions, idempotency and side effects.

06 Automation & Workflow

COMPONENT	ROLE	KEY REVIEW ITEMS
AutomationEngine	Coordinates automation.	Triggers, concurrency, failure handling.
AutomationRules	Rule/condition definitions.	Rule validation and conflicting rules.
AutomationExecutor	Executes automation actions.	Side effects, retries and idempotency.
WorkflowEngine	Workflow orchestration.	State model and error recovery.
WorkflowRunner	Workflow execution.	Concurrency and cancellation.
WorkflowRegistry	Workflow definitions registry.	Versioning and loading.
WorkflowContext	Execution context.	Data isolation and lifecycle.
WorkflowCondition	Conditional logic.	Validation and deterministic evaluation.
WorkflowAction	Workflow step/action.	Permission and side-effect controls.
WorkflowHistory	Execution records.	Persistence and retention.
WorkflowScheduler	Scheduling boundary.	Production scheduler requirements remain open.
WorkflowLoader	Loads workflow definitions.	Validation and compatibility.
ClientFollowUpWorkflow	Concrete follow-up workflow.	Verify triggers, actions and persistence.

07 Data Layer

The documented application data layer includes business concepts such as clients, contacts, deals, tasks and activity. A technical buyer should distinguish application data structures from a production database architecture.

DATA DOMAIN	REVIEW
clients	Schema, identifiers, relationships, validation and persistence.
contacts	Client/contact relationships and uniqueness rules.
deals	Deal state, value, lifecycle and links to clients.
tasks	Ownership, status, deadlines and completion history.
activity	Event/activity history, ordering and retention.

Production database boundary

No production database should be assumed from the presence of data models or persistence abstractions. A buyer should verify the actual storage implementation, migrations, backup strategy, concurrency behavior and recovery procedures.

08 Build & Packaging

CHECK	CURRENT STATUS	DUE DILIGENCE ACTION
Vite production build	Successful; Vite v8.1.4, 1830 modules transformed.	Reproduce from clean environment.
Production assets	Created successfully.	Verify hashes/output consistency.
Electron Builder	Packaging successfully completed.	Rebuild using documented environment.
Windows installer	VELAR Platform Setup 1.0.0.exe.	Verify installer behavior and signing.
Target	Windows x64.	Confirm runtime compatibility and minimum OS assumptions.

09 Dependencies

Dependency review should be performed directly against the transferred package manifest and lockfile. The following categories are documented technology choices; exact versions, licenses and vulnerabilities must be verified at diligence time.

CATEGORY	DEPENDENCY / TOOLING	REVIEW
UI	React	Version, license, security advisories.
Build	Vite	Version, plugin compatibility.
Routing	React Router	Version and route behavior.
Charts	Chart.js	Version and licensing.
Icons	Lucide React	Version and licensing.

Desktop	Electron	Version, Chromium security posture, update strategy.
Packaging	Electron Builder	Version and reproducibility.
Package management	Project manifest + lockfile	Pinned versions and integrity.

10 Security Considerations

Security maturity should be assessed separately from feature breadth. Key areas for a buyer include:

- Electron isolation, preload exposure and IPC validation.
- Secrets handling and absence of hard-coded credentials.
- Input validation at renderer, IPC and service boundaries.
- Permission boundaries for user and action operations.
- Safe execution of automation and workflow actions.
- Dependency vulnerability monitoring and update policy.
- Secure storage of any persistent business or AI memory data.
- Logging that avoids accidental exposure of sensitive client information.

11 Scalability Considerations

AREA	CURRENT CONSIDERATION	FUTURE REQUIREMENT
Desktop runtime	Local Electron application.	Evaluate service extraction and remote execution for scale.
Data persistence	Application-level data/persistence abstractions.	Production database, migrations, backups and HA strategy.
AI processing	Modular AI architecture.	Model gateway, queues, rate limits and observability.
Automation	Application-level automation components.	Distributed execution, retries, idempotency and monitoring.
Workflow	Workflow orchestration components.	Durable state, scheduling infrastructure and horizontal execution.
Memory	Memory abstractions.	Scalable storage, indexing, retention and evaluation.
Users	User-facing desktop surface.	Identity, RBAC, tenant isolation and auditability for SaaS.

12 Cloud Migration Considerations

A cloud/SaaS transition is a development program, not an assumed existing capability. A plausible path would separate business services, persistence, authentication, AI/model services, workflow execution and observability from the Electron shell while retaining Electron as a desktop client if commercially useful.

WORKSTREAM	TARGET CAPABILITY
API layer	Versioned backend API and service boundaries.

Database	Production relational/document storage with migrations and backups.
Identity	Authentication, authorization and role/tenant model.
AI gateway	Provider/model abstraction, quotas, logging and evaluation.
Workflow workers	Durable asynchronous execution.
Observability	Centralized logs, metrics, traces and alerting.
Deployment	CI/CD, environments, secrets and rollback.
Billing / tenancy	Subscription, tenant isolation and usage controls.

13 Known Limitations / Technical Debt Areas

Transparent assessment

The following are not presented as defects proven by a code audit. They are the principal diligence questions and likely hardening areas implied by the current product scope.

- Production-grade persistence and database strategy require explicit verification.
- Cloud SaaS infrastructure is not established by the desktop application architecture.
- AI model integration and model-serving maturity must be verified separately from AI Core module names.
- LearningEngine should be understood as feedback-oriented learning / decision outcome tracking, not autonomous model training.
- Security, permissions, auditability and secrets management require formal review before enterprise deployment.
- Workflow and automation components require production-scale reliability engineering for high-volume workloads.
- Automated tests, coverage, CI/CD and release governance should be assessed from the actual repository.
- Dependency versions, licenses and vulnerability posture must be checked against the final transfer package.

14 Recommended Technical Review Sequence

STEP	ACTION	OUTPUT
1	Repository inventory	Verified source tree and dependency map.
2	Build reproduction	Independent successful build/package.
3	Runtime walkthrough	Verified major application surfaces.
4	Architecture trace	Module-to-module dependency and data-flow map.
5	Data review	Verified persistence mechanism and schemas.
6	AI Core review	Verified actual implementations and model boundaries.
7	Security review	IPC, secrets, permissions and dependency findings.
8	Scalability review	Gap analysis for database/cloud/AI/workflow scale.
9	Transfer validation	Source, build instructions and artifacts

verified.

15 Buyer Verification Checklist

ITEM	VERIFY
Source code	Complete repository and history, where applicable.
Dependencies	Manifest, lockfile, versions and licenses.
Build	Clean-machine reproducibility.
Installer	Windows x64 installer launches and operates.
Frontend	Major CRM/ERP screens and routes.
AI Core	Actual module implementations and integration points.
Memory	Actual persistence behavior and storage location.
Automation	Rules, executor behavior and side effects.
Workflow	Execution, history and scheduling behavior.
IPC	Exposed APIs and validation.
Security	Secrets, permissions and Electron isolation.
Documentation	Architecture, setup, build and transfer notes.

16 Technical Conclusion

VELAR Platform presents a substantial modular desktop product foundation combining CRM/ERP functionality with an AI-oriented architecture, automation and workflow orchestration. The strongest technical diligence posture is transparent: demonstrate what is implemented, identify architectural abstractions, and explicitly reserve production database, cloud SaaS, advanced model integration and enterprise hardening for future development where they are not yet established.

Diligence position

The product should be evaluated as an extensible technology asset with a working desktop application and a modular AI architecture—not as a completed cloud SaaS or autonomous AI system.