**Axomate – Technical Requirements**

v1.0

*Author: Harpreet Chahal*

Date: August 22, 2025

# Index

# 1. Architecture Overview

* Layered architecture: UI (WPF/MVVM) → Application Layer (services, validators) → Domain (POCO models) → Infrastructure (EF Core, SQLite).
* MVVM: ViewModels expose observable state (ObservableObject), IRelayCommand commands, and mode flags (NewCustomerMode, NewVehicleMode, NewInvoiceMode).
* Persistence via EF Core DbContext; service interfaces (ICustomerService, IVehicleService, IInvoiceService, etc.) hide repositories.
* PDF: IInvoicePdfService abstraction keeps UI decoupled from rendering engine.

# 2. Technology Stack & Versions

* .NET 6+ (WPF).
* CommunityToolkit.Mvvm for MVVM primitives.
* Entity Framework Core (SQLite provider).
* SQLite local database.
* Pluggable PDF engine via IInvoicePdfService.
* Secure admin credentials via IAuthService with salted hashes.

# 3. Layer Responsibilities

* UI: Pure bindings; no business logic beyond input filters.
* ViewModel: Orchestrates flows; validates inputs; toggles flags; calls services; handles IsBusy; logs errors.
* Application Services: CRUD operations and use cases (customer, vehicle, service items, invoices, mileage, PDF).
* Infrastructure: EF Core mappings, migrations, constraints, value converters (decimal↔double for SQLite).

# 4. Data Model & Persistence Rules

* Entities: Customer, Vehicle, ServiceItem, Invoice, InvoiceLineItem, MileageHistory, Company, AdminCredential.
* Audit fields: CreatedDate (UTC) and CreatedBy (string) required on inserts across entities.
* Relationships: Customer 1..\* Vehicles; Invoice 1..\* InvoiceLineItems; Invoice → Customer/Vehicle required.
* Constraints: Price ≥ 0, Quantity ≥ 1, Description ≤ 200; Mileage 0..2,000,000; Vehicle LicensePlate unique (case-insensitive in SQLite); VIN length ≤ 17 (optional unique).
* Indexes: Invoice.ServiceDate; (CustomerId, VehicleId, ServiceDate).
* SQLite decimal handling via ValueConverter<decimal,double> and nullable variant.

# 5. EF Core Configuration & Migrations

* Fluent config for property lengths, delete behaviors, and check constraints.
* Maintain migrations; apply on startup or via tooling.
* All core tables are keyed; no keyless entities for core flows.

# 6. Services & Contracts

* ICustomerService: GetAllAsync(), AddOrUpdateAsync(Customer), UpdateAsync(Customer) …
* IVehicleService: GetByCustomerIdAsync(int), AddOrUpdateAsync(Vehicle) …
* IServiceItemService: GetAllAsync(), SearchByNameAsync(string), AddAsync(ServiceItem).
* IInvoiceService: AddAsync(Invoice), GetByVehicleAsync(int, includeDetails=true).
* IMileageHistoryService: RecordAsync(vehicleId, mileage, when, source, notes), GetLatestForDayAsync(vehicleId, refDate), GetLatestOnOrBeforeAsync(vehicleId, refDate), GetLastRecordTimeAsync(vehicleId, refDate).
* IInvoicePdfService: GenerateAsync(Invoice, CancellationToken).
* IAuthService: Admin password set/verify (salted hash). Services set CreatedDate/CreatedBy on inserts.

# 7. UI Binding & Commanding

* Mileage TextBox binds to MileageText (string, TwoWay, UpdateSourceTrigger=PropertyChanged); VM mirrors to Mileage (int?).
* Programmatic prefill guarded by \_suppressMileageRecording flag to avoid false records.
* Command gating: SaveNewCustomer only when NewCustomerMode; SaveNewVehicle only when NewVehicleMode && form valid; SaveAndPrint only when NewInvoiceMode && !IsBusy.
* IsBusy wraps long ops in try/finally to re-enable UI.

# 8. Mileage Recording – Technical Rules

* 5‑hour lock: ignore/deny new record if last record < 5 hours prior to new timestamp.
* Regression guard: ignore values < latest persisted.
* Prefill on vehicle/date change: fetch latest for day; set under suppression.
* Save‑time insert: add record only if last record ≥ 5 hours at ServiceMomentNow.

# 9. Transactions & Concurrency

* Prefer transaction wrapping for Save & Print (create entities, mileage, invoice) to avoid partial saves.
* Use async EF APIs; update UI-bound properties on UI thread.

# 10. Security

* IAuthService uses salted+hashed passwords; unique username enforced.
* No plaintext secrets in repo; externalize config.
* CreatedBy set from current authenticated user or environment.

# 11. Error Handling & Logging

* Catch at command boundaries; log to error.log; show concise user message.
* Logging failures must be swallowed (no crash).

# 12. Performance & Reliability

* Use ObservableCollection for bound lists; minimize refresh scope (e.g., reload VehiclesByCustomer only when needed).
* Keep PDF generation off UI thread; indicate progress via IsBusy.

# 13. Testing

* Unit tests: mileage lock/regression; command gating; mapping custom items to catalog.
* Service tests: EF Core with SQLite in-memory to verify constraints/mappings.
* Integration: Save & Print atomicity and PDF call verification.

# 14. Deployment & Data Upgrade

* Ship initial EF migration; apply subsequent migrations at startup or via admin tool.
* Backup before breaking schema changes.

# 15. Configuration

* Externalize PDF path, logo assets, company settings (Company table or config).
* Support currency/decimal formatting via company settings where applicable.

Note: In Word, press F9 or use References → Update Table to refresh the Index (TOC).