I. Project Overview

a. Project Description

The **Sheems Steel Construction Supply Sales and Inventory Management System (SIMS)** is a **custom-built, web-based full-stack application** designed to modernize and streamline the core operations of Sheems Steel Construction Supply, a local business specializing in construction materials. Currently, the business relies entirely on **manual, paper-based processes** for recording sales, calculating profits, and tracking inventory. This leads to operational inefficiencies, frequent human errors in financial records, reactive inventory management causing stock-outs or overstocking, and a lack of **real-time business insights** for decision-making.

The SIMS directly addresses these challenges by providing a **centralized digital platform** for the business owner-operator. It automates the sales process through a **Point-of-Sale (POS) interface**, ensures **real-time and accurate inventory updates** with each transaction, calculates profits automatically, and generates essential business reports on demand. By replacing ledgers and calculators with a secure, integrated system, SIMS aims to transform Sheems Steel from a manually-intensive operation into a **data-driven business**, enhancing accuracy, saving time, and supporting sustainable growth.

b. Project Objectives

The primary goal of this project is to develop a functional, reliable, and user-friendly full-stack system that automates the critical daily operations of Sheems Steel. Our specific, measurable objectives are:

1. A**utomate Core Business Operations:**
    - To reduce the time required to record a sale and update inventory from over 5 minutes (manual process) to under 30 seconds.

- To eliminate 100% of manual calculation errors in daily profit reporting by implementing automated profit logic based on cost and sale price.

2. **Establish Proactive Inventory Control:**
   - To provide a single source of truth for inventory, offering real-time stock level visibility across all product categories.
   - To implement an automated low-stock alert system that proactively notifies the owner before items run out, aiming for 100% coverage of stocked items.
3. **Deliver Actionable Business Intelligence:**
   - To enable the generation of key reports—such as Daily Sales Summaries and Inventory Status reports—with a single click, eliminating manual data compilation.
   - To achieve a post-deployment user satisfaction score of at least 4.0 out of 5 from the business owner regarding the system's ease of use and effectiveness.

c. Project Scope & Limitations

To ensure a focused and deliverable project within the academic timeline, the following scope and limitations have been defined:

- **In Scope (What the system WILL include):**
  - Features: Secure User Authentication for the business owner, Comprehensive Product & Category Management, a functional Point-of-Sale (POS) transaction module, Real-time Inventory Tracking & Adjustment, Automated Low-Stock Alerts, a Dashboard with key metrics, Supplier Management, and Reporting modules for Sales and Inventory.
  - Users: The system is designed for a single, primary user: the Business Owner-Operator, who manages all aspects of sales and inventory.
  - Platform: The system will be developed as a responsive web application accessible via modern browsers.
- **Out of Scope (What the system WILL NOT include in this version):**

- Features: Advanced accounting modules (Accounts Payable/Receivable, Payroll), an e-commerce platform for online sales, integration with hardware like barcode scanners, or a multi-user role-based access control system.
- Platforms: Development of native iOS or Android mobile applications.
- Data Integration: Automated data feeds from suppliers; all product and inventory data will be manually entered or imported via file.

- **Key Constraints & Assumptions:**
  - **Constraint: T**he system's accuracy is dependent on correct data entry by the user. The project does not include providing hardware (e.g., receipt printers, scanners).
  - **Assumption:** The primary user possesses basic computer literacy. The client will be responsible for securing and maintaining the hosting environment after the initial deployment.
  - **Limitation:** The initial version (V1.0) is designed as a foundational management system for a single operator. Features supporting multiple staff with different permissions are planned for future iterations.

II. Overall System Design & Development

a. System Architecture

The SIMS follows a modern client-server architecture with a clear separation between the Vue.js frontend (client) and the Laravel backend (server), communicating via a RESTful API. This decoupled design ensures maintainability, scalability, and a responsive user experience.

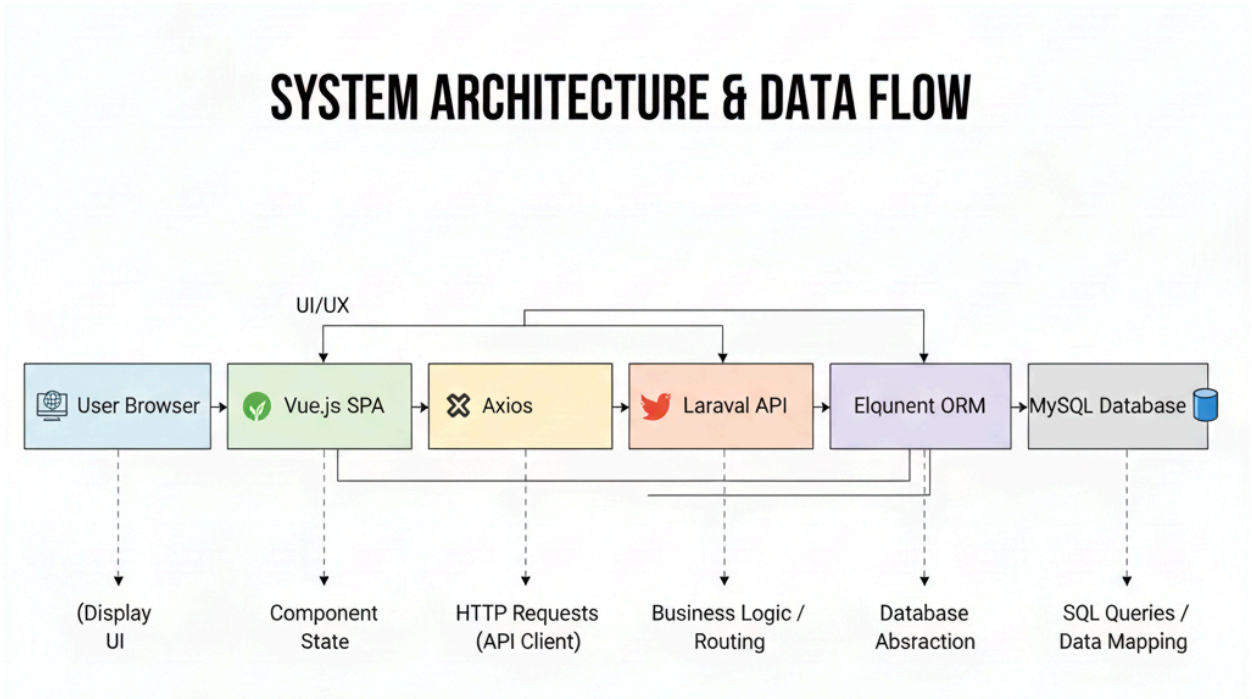
Complete System Flow (Frontend + Backend Integration)

1. **User Interaction Layer (Vue.js Frontend):**
   - The user interacts with a responsive Vue.js Single Page Application (SPA) hosted in a web browser.

- Components such as POS.vue, Inventory.vue, Sales.vue, and Dashboard.vue render dynamic interfaces using Vue Router for navigation.

2. **API Communication Layer (Axios HTTP Client):**
   - Vue.js components make asynchronous HTTP requests to the Laravel backend using Axios.
   - All requests include an authentication token (Laravel Sanctum) in the header for secure access.
3. **Application Logic Layer (Laravel Backend):**
   - Requests are routed through Laravel's API routes (routes/api.php) to dedicated controllers (e.g., SalesController, InventoryController).
   - Laravel handles business logic: validating input, calculating totals, updating inventory, and enforcing business rules.
4. **Data Persistence Layer (MySQL Database):**
   - Laravel Eloquent ORM interacts with the structured MySQL database.
   - Key tables include users, products, sales, sales_items, inventories, suppliers, and inventory_logs, which maintain relational integrity and audit trails.
5. **Response Flow:**
   - After processing, Laravel returns a JSON response (success/error) to the frontend.
   - Vue.js updates the UI state reactively, providing immediate feedback (e.g., updated stock levels, new sale confirmation).

High-Level Architecture Diagram Concept:



Tools and Technologies Used

- **Frontend:** Vue.js 3, Vue Router, Vuex/Pinia (for state management), Axios, Tailwind CSS/Bootstrap (for styling)
- **Backend:** Laravel 10.x, Laravel Sanctum (API authentication), Eloquent ORM
- **Database:** MySQL 8.0
- **Development Tools:** Composer, npm, VS Code

b. Backend Implementation

Database Structure

The database is normalized across 10 core tables to ensure data integrity, avoid redundancy, and support auditability:

- **users**: Stores administrator credentials and profile (linked via id to sales.ClerkID and inventory_logs.created_by).