A-1 Launchpad: Challenge for Innovative Solutions

A1 Fence Products Company Private Limited

ProjectTitle: Electronics Lab Inventory Management System (LIMS)

Objective: Design, develop, and demonstrate a web-based inventory management system tailored for electronics R&D and manufacturing laboratories.

Team Name: KURO

College: BMS Institute of Technology and Management

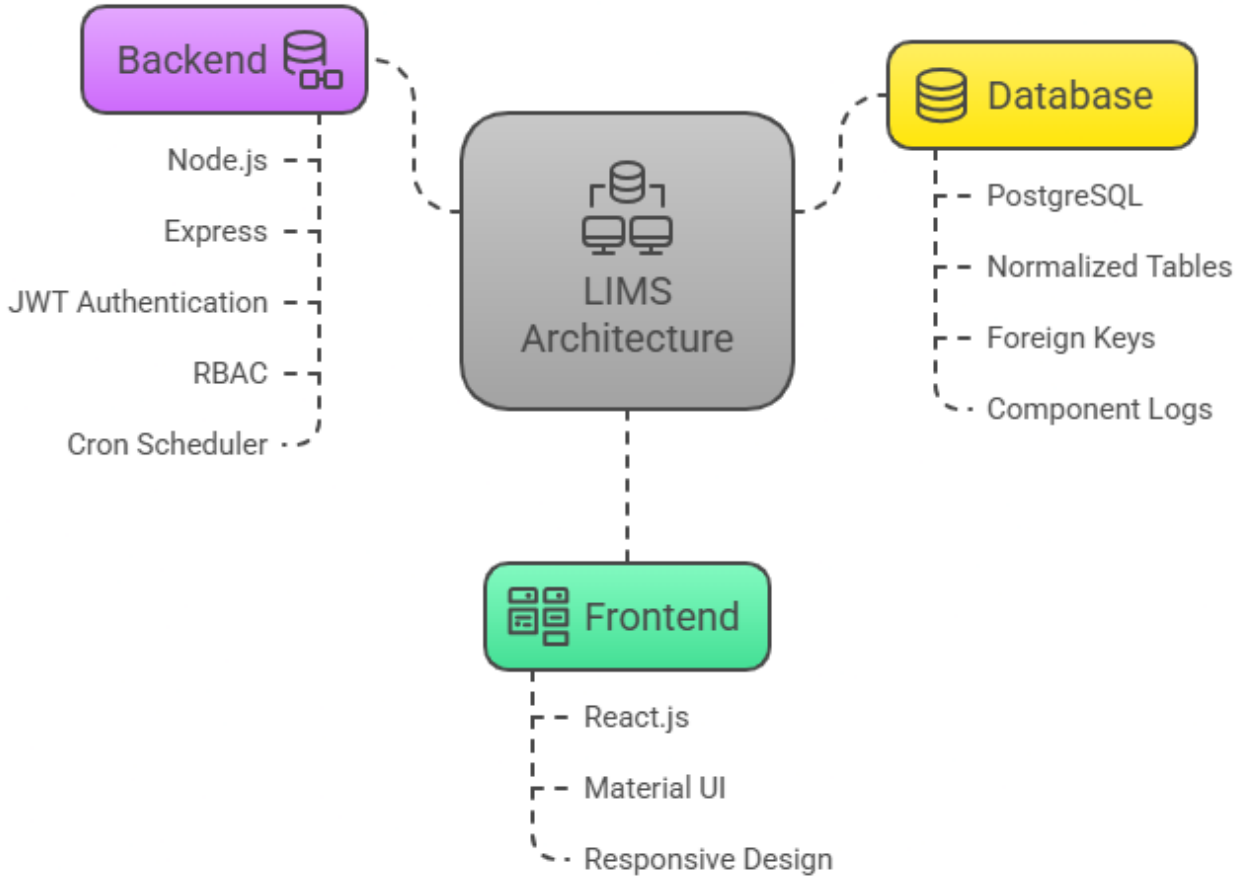
Team Members:

Sl.no	Name	Contact
1	Varsha S Kundur	8618438523
2	P Allen Thiagaraj	6361196329

Link to Github Repository:

https://github.com/AllenPrabu/LIMS-Inventory-Manager/

High-Level Architecture Diagram



The Electronics Lab Inventory Management System (LIMS) is built on a streamlined three-tier architecture, optimized for performance, security, and future extensibility.

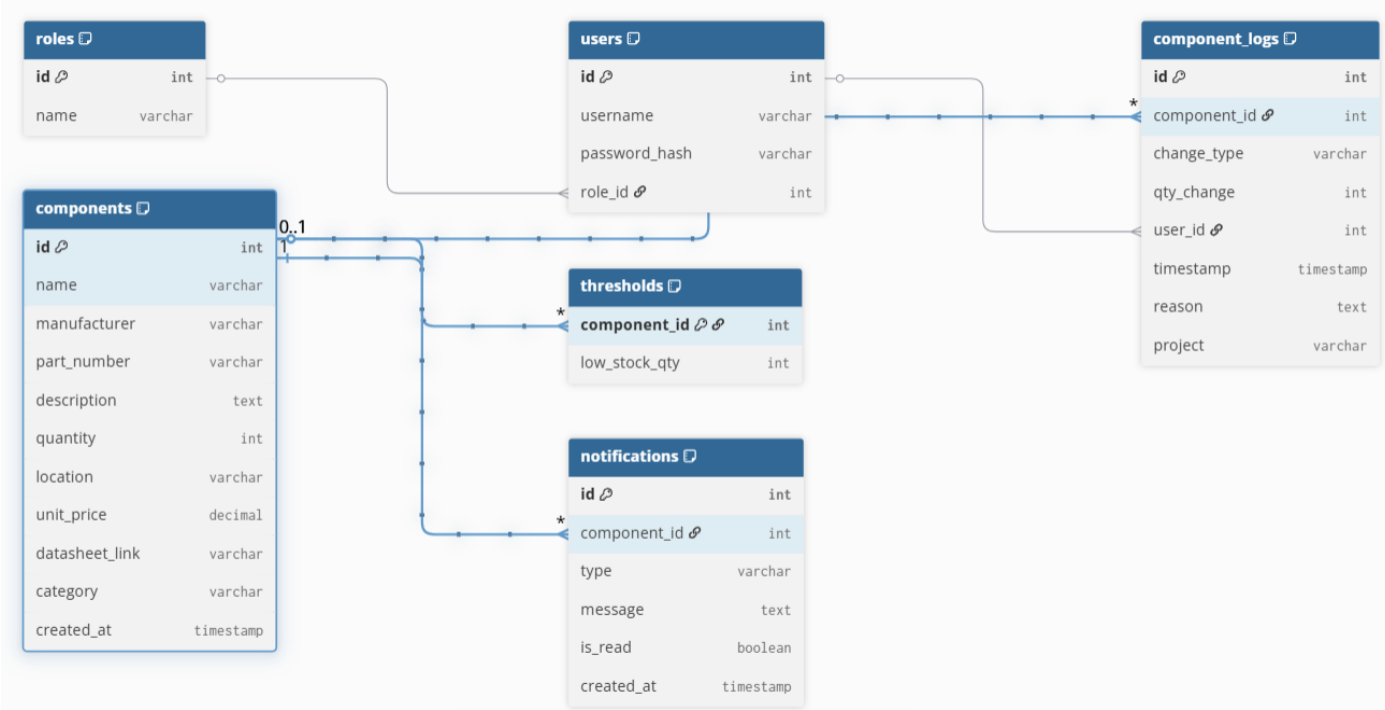
- Frontend: Built using React.js and Material UI, the responsive web app adapts to all screen sizes and communicates with the backend over HTTPS via Axios. It offers user-specific views based on roles, with secure route handling and dynamic rendering.
- Backend: Powered by Node.js and Express, the backend exposes RESTful APIs for authentication, inventory operations, logging, and alert generation. It uses JWT-based login, with role-based access control (RBAC) applied through middleware. Background tasks like stock validation are handled using a cron scheduler.
- Database: A PostgreSQL database stores normalized records for users, components, logs, thresholds, and notifications. Inventory updates are permanently recorded in the component_logs table for traceability.

The entire system is Dockerized for cross-platform deployment and supports automated workflows via GitHub Actions. It's stateless at the API layer, allowing horizontal scaling and future upgrades like mobile integration, QR scanning, or predictive analytics.

Technology Stack Justification

Layer	Technology	Why?
Frontend	React + Material UI	Fast, responsive, component-based design
Backend	Node.js + Express	Asynchronous, scalable, easy REST APIs
Database	PostgreSQL	Relational, ACID-compliant, supports schema constraints
DevOps	Docker + GitHub Actions	Containerized deploys, CI/CD ready
Auth	JWT + RBAC	Stateless and secure authentication
Charts & UI	Recharts + MUI	Smooth, readable data visualizations
Notification	WebSocket-ready design, fallback to polling	Real-time readiness

Database Schema Diagram



- components: Stores part details like quantity, location, and metadata.
- users & roles: Manage authentication and role-based access.
- component_logs: Tracks all inventory changes with user, quantity, and reason.
- thresholds: Defines low-stock limits per component.
- notifications: Stores system alerts for stock issues and updates.

Foreign keys maintain data integrity and enable traceable, scalable inventory management.

Major API Endpoints

Endpoint	Method	Description
/api/auth/login	POST	Login and get JWT
/api/auth/register	POST	Create new user (Admin only)
/api/components	GET	Search/filter components
/api/components	POST	Add new component
/api/components/:id/logs	GET	View logs for a component
/api/dashboard/low-stock	GET	List components below threshold
/api/dashboard/old-stock	GET	List components unused for 3+ months
/api/notifications	GET	Get unread notifications

Scalability & Maintainability

- **Modular Codebase**
  Responsibilities are cleanly separated across services, controllers, routes, and utilities—making the system easier to maintain and extend.
- **Containerized Deployment**
  Docker containers ensure consistent environments and enable smooth deployment to cloud platforms via Compose or Kubernetes.
- **Horizontal Scalability**
  The stateless backend scales via load balancing. PostgreSQL read replicas improve performance during high load.
- **Secure Auth & RBAC**
  JWT tokens and role-based middleware restrict access, allowing only authorized users to perform sensitive actions.
- **Future-Ready Enhancements**
  Designed for extensibility, planned features include:
  - Barcode/QR scanning via camera
  - React Native mobile app for real-time inventory updates
  - AI-driven restocking suggestions
  - Immutable, cryptographically secure audit trails

This architecture ensures LIMS is robust, secure, and scalable—ready to support growing academic, industrial, and research lab needs.