Project Requirement Document (PRD)

Lab Inventory & Attendance System

Version: 1.0
Date: 2026-02-17
Status: Draft / Ready for Development

---

1. Executive Summary

The **Lab Inventory & Attendance System** is a web-based application designed to streamline the management of laboratory materials and track user attendance within a college or corporate lab environment. It provides a centralized platform for Admins to manage stock, generate QR codes for assets, and monitor user activity. For Users, it offers a simplified interface to view profile details, track attendance hours, and access relevant lab resources. The system leverages **Node.js** and **PostgreSQL** for a robust backend and is architected for seamless deployment on **Render.com**.

2. Project Goals & Objectives

- **Digitize Inventory:** Eliminate manual stockkeeping by providing a digital CRUD interface for materials.
- **Automate Tracking:** Simplify asset tracking using QR code generation for materials.
- **Streamline Attendance:** Automatically calculate and display user attendance hours based on login/logout or scan activities.
- **Enhance Security:** Implement role-based access control (RBAC) to separate Admin and User privileges.
- **Ensure Scalability:** Build a cloud-ready application deployable on modern PaaS providers like Render.

3. Problem Statement

Current manual methods of tracking lab inventory (spreadsheets, paper logs) are prone to errors, data loss, and inefficiency. Furthermore, tracking student/employee time in the lab is often disconnected from the inventory system, leading to fragmented data. There is a need for a unified solution that handles both material management and user time tracking in a secure, accessible web application.

4. Scope

In Scope

- **User Authentication:** Secure login for Admins and Standard Users.
- **Dashboard:** Role-specific dashboards (Admin vs. User).
- **Material Management:** Add, Update, Delete, View materials. Bulk upload via Excel.
- **QR Code Generation:** Generate printable QR codes for inventory items.
- **User Management:** Admin can register, edit, and delete users.
- **Time Tracking:** View total hours logged by users.
- **Search & Filtering:** Real-time search for materials and users.

Out of Scope

- **Mobile App:** Native mobile application (web app is responsive but not native).
- **External Hardware Integration:** Direct integration with biometric scanners (manual/web-based entry only).
- **Payment Gateway:** No monetary transactions involved.

5. Stakeholders

- **Lab Administrators:** Manage inventory, users, and overall system configuration.
- **Lab Users (Students/Employees):** Access lab resources, view profile, and track their own time.
- **Development Team:** Responsible for building, maintaining, and deploying the system.

6. User Personas

- **Admin Alice:** Needs a high-level view of all stock. Wants to quickly bulk-upload new equipment lists and see who is currently active in the lab.
- **Student Sam:** Needs to check if a specific component (e.g., "Resistor 10k") is in stock and see how many hours he has spent in the lab for his credit requirements.

7. User Stories

- *As an Admin*, I want to upload an Excel sheet of materials so that I don't have to enter them one by one.
- *As an Admin*, I want to edit user details if they change departments or lose access.
- *As a User*, I want to see my total logged hours upon logging in.
- *As an Admin*, I want to generate QR codes for selected items to label the physical inventory.
- *As a User*, I want to switch the login page to dark mode to reduce eye strain.

8. Functional Requirements

8.1. User Authentication Module

- **Login:** Validate credentials (email/username + password).
- **Role Check:** Redirect to `adminhome.html` or `userhome.html` based on role.
- **Logout:** Secure session termination.

8.2. Admin Dashboard Module

- **Overview Stats:** Display total materials and total users count.
- **Navigation:** Easy access to User Management, Material Management, and Reports.

8.3. Material Management Module

- **CRUD Operations:** Create, Read, Update, Delete materials.
- **Excel Upload:** Parse `.xlsx` files to bulk insert materials (Required columns: `material_code`, `material_name`).
- **Stock Levels:** Track Opening Balance, Issued Qty, and Available Qty.

8.4. User Management Module

- **Registration:** Form to add new users with Role, Department, Roll No, etc.
- **List View:** Tabular view of all registered users with search/filter.
- **Edit/Delete:** Modify user details or remove accounts.

8.5. QR Code Module

- **Selection:** Select multiple materials from the list.
- **Generation:** dynamic QR code generation containing `material_code`.
- **Print View:** Grid layout optimized for printing labels.

9. Non-Functional Requirements

- **Performance:** API response time < 200ms for standard queries.

- **Security:** Passwords hashed using `bcrypt` (if applicable) or secure storage. Input validation to prevent SQL injection.
- **Scalability:** Stateless backend design suitable for horizontal scaling on Render.
- **Reliability:** 99.9% uptime target via Render's managed services.
- **Accessibility:** High-contrast modes (Dark Mode) and responsive design for various screen sizes.

10. System Architecture Overview

10.1. Technology Stack

- **Frontend:** HTML5, CSS3 (Custom + Responsive), JavaScript (Vanilla ES6+).
- **Backend:** Node.js (Runtime), Express.js (Web Framework).
- **Database:** PostgreSQL (Relational Database).
- **Deployment:** Render.com (PaaS).

10.2. High-Level Diagram

```
graph TD
    Client[Web Browser] <-->|HTTP/JSON| Server[Node.js + Express]
    Server <-->|SQL| DB[(PostgreSQL)]
    Server -->|Logs| Logger[Console/Render Logs]
```

11. Deployment Requirements

- **Platform:** Render.com
- **Pipeline:** "Antigravity" (Automated deployment from Git repository).
- **Build Command:** `npm install`
- **Start Command:** `node server.js`
- **Environment Variables:**
  - `DATABASE_URL` : Connection string for PostgreSQL.
  - `PORT` : Port for the server to listen on (default 5000).
  - `NODE_ENV` : `production`

12. API Requirements

Method	Endpoint	Description
`POST`	`/login`	Authenticate user and return role.
`GET`	`/materials`	Fetch all inventory items.
`POST`	`/upload-materials`	Bulk upload materials from Excel data.
`PUT`	`/materials/:id`	Update a specific material.
`DELETE`	`/materials/:id`	Delete a specific material.
`GET`	`/users`	Fetch list of all users.
`POST`	`/admin/register`	Register a new user.

13. Database Schema

Table: `materials`

Column	Type	Constraints	Description
`id`	SERIAL	PRIMARY KEY	Unique ID
`material_code`	VARCHAR(50)	UNIQUE	Barcode/QR Code string
`material_name`	VARCHAR(200)	NOT NULL	Item Name
`material_type`	VARCHAR(100)	DEFAULT 'General'	Category
`balance`	INT	DEFAULT 0	Current Stock
`available_quantity`	INT	DEFAULT 0	Allocatable Stock

Table: `users`

Column	Type	Constraints	Description
`id`	SERIAL	PRIMARY KEY	Unique ID
`username`	VARCHAR(50)	UNIQUE	Login Username
`password`	VARCHAR(255)	NOT NULL	Hashed/Plain Password
`role`	VARCHAR(20)	CHECK ('admin', 'user')	Permission Level
`total_time`	VARCHAR(20)	DEFAULT '00:00:00'	Accumulated Lab Hours

14. UI/UX Requirements

- **Login Page:** Clean, focused interface with Dark Mode option.
- **Admin Dashboard:** Card-based layout for quick access to modules.
- **Tables:** Striped rows, hover effects, sticky headers for long lists.
- **Modals:** Use modal popups for forms (Edit/Add) to keep context.
- **Feedback:** Toast notifications (Success/Error) instead of native alerts.

15. Security Requirements

- **SQL Injection:** Use parameterized queries (`pg` library handles this).
- **XSS Protection:** Escape HTML content when rendering user input in the frontend.
- **CORS:** Restrict API access to trusted origins.
- **Environment Variables:** Never commit secrets to Version Control.

16. Logging & Monitoring

- **Application Logs:** Console logs for request handling (Errors, Upload summaries).
- **Database Monitoring:** Render functionality checking connection pool health.

17. Risks & Mitigation

- **Risk:** Data loss during Excel upload.

    - *Mitigation:* Transaction-based inserts or atomic operations; detailed error reporting per row.
- **Risk:** Unauthorized Admin access.
    - *Mitigation:* Strong password policies; role checks on every Admin API endpoint.

18. Milestones & Timeline

- **Phase 1:** Core Authentication & Database Setup (Day 1-2)
- **Phase 2:** Material Management & Excel Upload (Day 3-4)
- **Phase 3:** User Management & Time Tracking (Day 5-6)
- **Phase 4:** UI Polish, Dark Mode, & Deployment (Day 7)

19. Acceptance Criteria

- Admin can successfully upload an Excel file with 50+ items without error.
- System rejects login for invalid credentials.
- New users defaults to 0 hours logged.
- Application deploys to Render and connects to DB successfully.

20. Future Enhancements

- **Email Notifications:** Send alerts for low stock.
- **Check-in/Check-out Hardware:** Integration with RFID readers.
- **Audit Logs:** Detailed history of who modified what material and when.