Student Name: SELVAHAREESH K

Project ID: 34

Project title: BIT inventory management system

Technical Components

Component	Tech Stack
Backend	Django
Frontend	HTML , CSS
Database	PostgreSQL
API	Django Rest framework

Implementation Timeline

Phase	Deadline	Status	Notes
Stage 1		**In Progress**	**Planning and RequirementGathering**
Stage 2		**Not Started**	**Design and Prototyping**
Stage 3		**Not Started**	**DB Designing**
Stage 4		**Not Started**	**Backend Implementation**
Stage 5		**Not Started**	**Testing & Implementation**
Stage 6		**Not Started**	**Deployment**

PROBLEM STATEMENT:

The organization currently uses an inventory management system that requires manual tracking and updating of hardware components across several labs.

Inefficiency: Manually looking through several spreadsheets takes a lot of time and is prone to mistakes. Finding available components is delayed as a result, particularly when upgrades or replacements are required immediately.

Lack of Real-time Updates: Inventory changes (such as new purchases, transfers, or disposals) do not always appear instantly in all records, which can lead to disparities and misunderstandings regarding the true availability of specific parts.

Data Inconsistency: Inconsistent data may result from different labs using different spreadsheet formats and detail levels. It is challenging to compile data and keep an integrated picture of the inventory because of this inconsistency.

PROJECT-FLOW:

Purpose:

In order to facilitate effective searching and allocation, the BIT inventory management system attempts to consolidate the tracking and availability of hardware components across all labs. This technology ensures best resource usage and cost savings by giving visibility into current stock in other labs, hence reducing needless purchases.

Scope:

The system consists of a database for storing and retrieving inventory data, a web-based interface for lab incharges to enter and update stock details, and a search feature for finding available components across labs. It meets the requirements of several labs inside the company by offering role-based access, real-time updates, and thorough reporting.

Business Context:

With the current system, labs frequently buy new components without being aware of the stock levels in other labs, which wastes resources and results in needless expenses. In order to solve this problem, the inventory management system streamlines procurement procedures, improves resource sharing, and eventually lowers operating expenses, all of which improve the productivity and financial performance of the company.

Functional Requirements:

User Management:

- **Authentication:** Secure login and logout for users.
- **Authorization:** Assign roles (admin, faculty, staff) and permissions.
- **User Registration:** New users can register with appropriate details.
- **Profile Management:** Users can view and update their profiles.

Inventory Management:

- **Item Tracking:** Manage inventory items like books, lab equipment, etc.
- **Categorization:** Organize items into categories.
- **Item Details:** Store information such as name, category, quantity, location, condition, and acquisition date.

Inventory Operations:

- **Add/Update/Delete Items:** Manage inventory entries.
- **Search and Filter:** Locate items based on various criteria.

Check-in/Check-out System:

- **Issue and Return Items:** Users can borrow and return items.
- **Due Dates and Reminders:** Set return dates and send reminders.

Reporting and Analytics:

- **Inventory Reports:** Current status of items.
- **Usage Reports:** Insights into item usage.
- **Audit Trails:** Log of inventory transactions for auditing.

Notification System:

- **Alerts:** For low stock, overdue items, and maintenance.
- **Reminders:** Notify users of upcoming due dates and actions needed.

Non-Functional Requirements:

Performance:

- The system should respond to user requests within 2 seconds under normal conditions.

Usability:

- A clean, intuitive user interface accessible with minimal training.
- Comprehensive documentation for both users and technical aspects.

Reliability:

- The system should handle failures gracefully with minimal service disruption.

Security:

- Sensitive data should be encrypted both at rest and in transit.
- Implement robust authentication and authorization mechanisms.

Scalability:

- The system should support both horizontal and vertical scaling to manage increasing loads.

Maintainability:

- A modular architecture for easy updates and maintenance.
- Automated testing to ensure code integrity.

Interoperability:

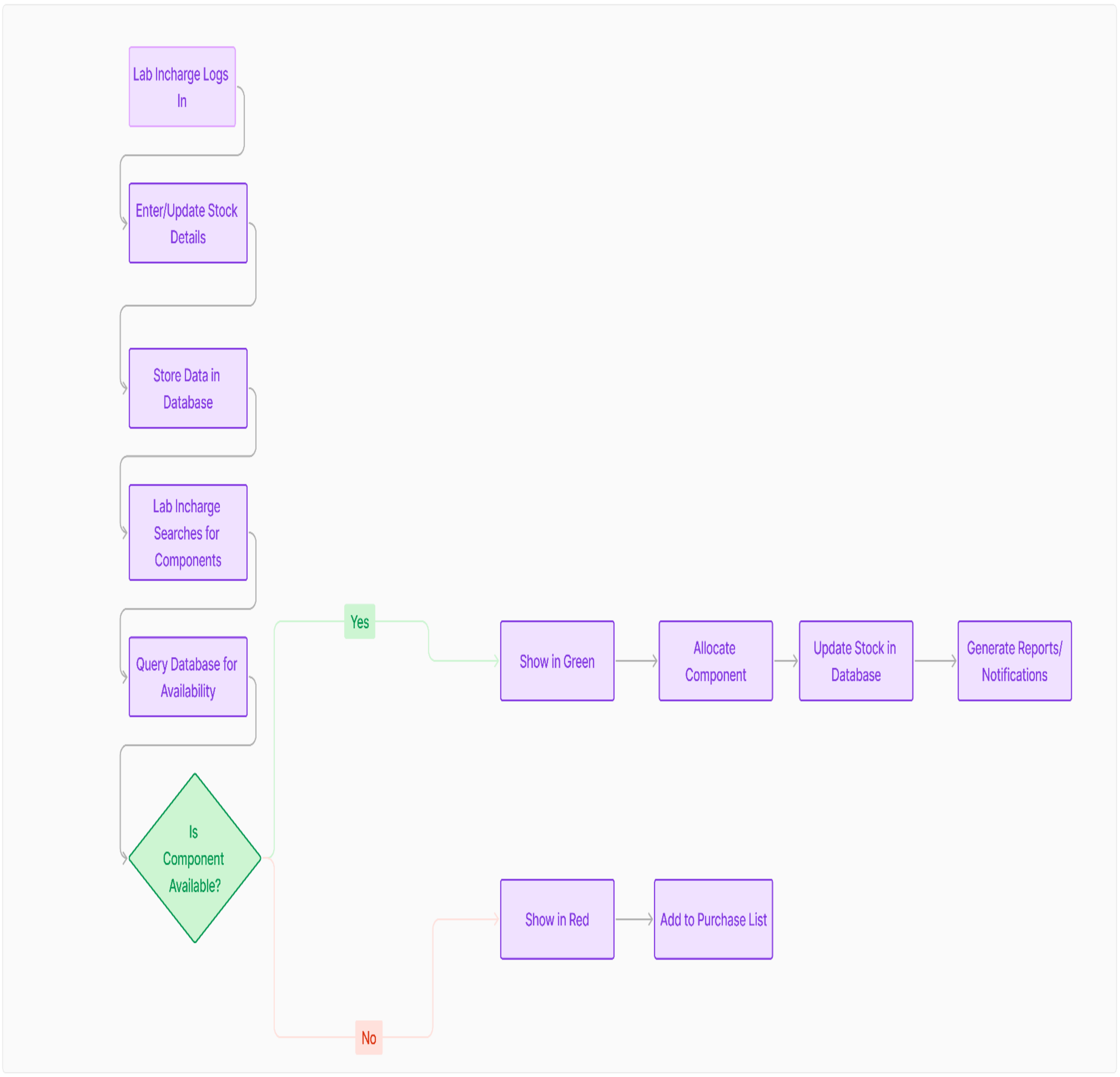
- Provide APIs adhering to RESTful principles and support JSON and XML.
- Seamless integration with other college systems like student information and financial management systems.

Compliance:

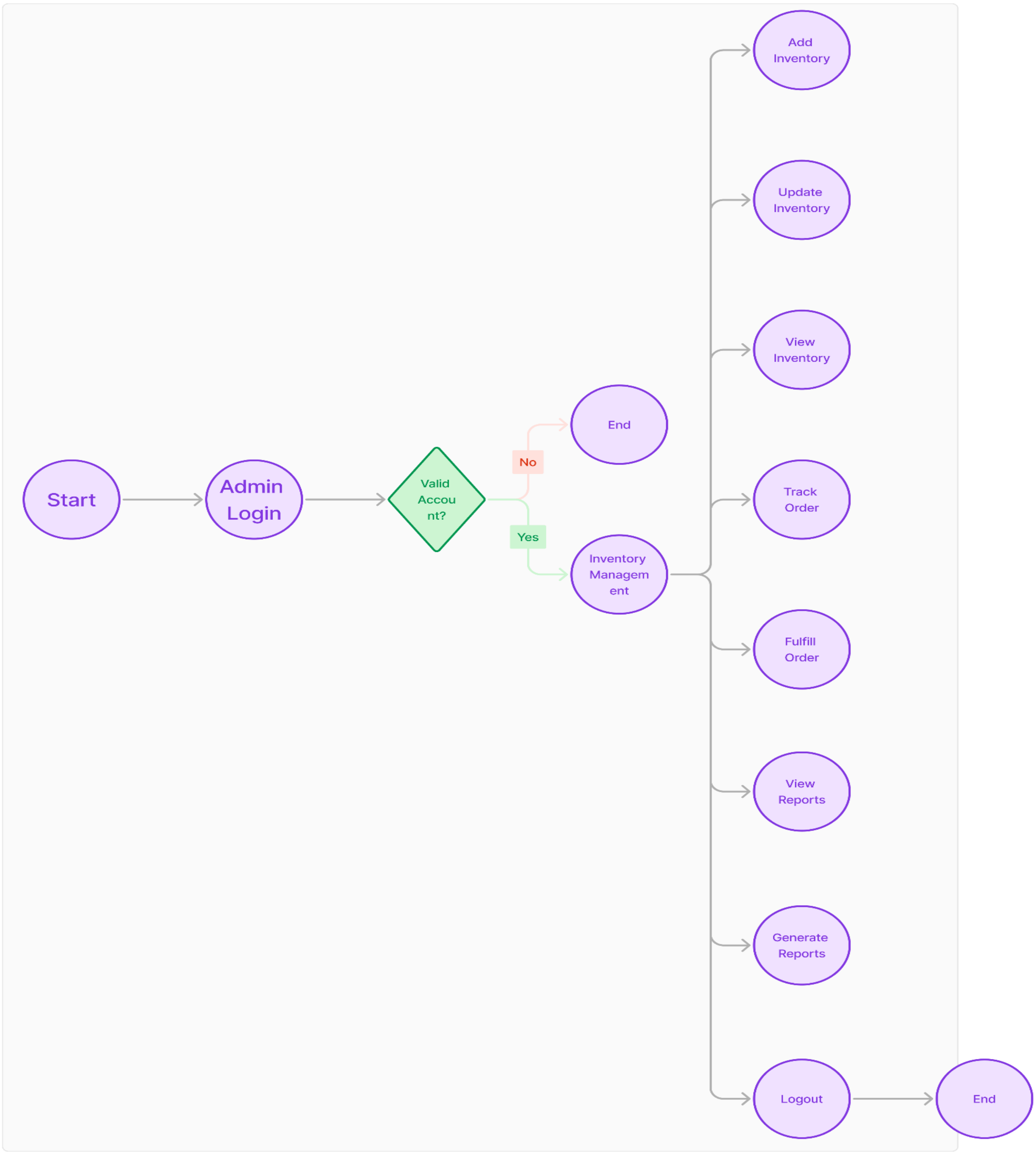
- The system should comply with relevant legal and regulatory requirements, such as GDPR and FERPA.

FLOWCHART:

STAFF FLOWCHART:



ADMIN FLOWCHART:


Start
Admin Login
Valid Account?
No
End
Yes
Inventory Management
Add Inventory
Update Inventory
View Inventory
Track Order
Fulfill Order
View Reports
Generate Reports
Logout
End