**PDL IMS Outline (Inventory Management System)**

**Part 1 : Description of All Stores**

**1. Yarn Store:**

1. Yarn receives as per work order

2. Order wise Yarn allocation from advance store.

3. Yarn issue according to Buyer, style, PO for knitting. (Inside & outside factory)

4. Yarn return to supplier if required.

5. Sample program Yarn issue

6. Yarn additional Booking if required.

7. Yarn gate pass

8. Stock Yarn sale

**Report:**

1. Supplier, Buyer, Style, PO, yarn type wise yarn stock analysis report.

2. Yarn stock register. (L c/brand/count/lot wise)

3. Daily weekly, monthly yarn issue/Gray fabric received report

4. Order & count wise yarn closure report.

5. After completing knitting, planned VS actual yarn consumption report.

6. Yarn stock report with valuation

7. Yarn Return Report

8. Pipe line Yarn report

**2. Gray Fabric Store:**

1. Gray fabric receives according to Buyer, style, Po as well as yarn issue.

2. Gray fabric issue according to Buyer, style, Po to dyeing Factory

3. Gray fabric Gate pass

4. Gray fabric returns/reject from dyeing if required.

5. Gray fabric inspection report

6. Create Knitting bill as per Gray receive

**Report:**

1. Gray fabric stock according to Buyer, style, Po

2. Gray received and dyeing delivery status

3. Outside factory gray fabric status report.

4. Reject/Return gray fabric status report according to buyer, style, PO.

5. Party wise knitting bill statement as per gray received

**3. Finish Fabric Store:**

1. Finish fabric receive from dyeing according to byer, style, PO, color as well as batch and lot number.

2. Fabric issue to cutting as per requisition

3. Finish fabric return to Dyeing if required

4. Fabric send to leftover stock after shipment

5. Dyeing Bill create as per fabric received

**Report:**

1. Buyer, style, PO, color wise finish fabric receives and delivery status.

2. Finishing fabrics stock register

3. Dyeing fabrics due balance

4. Fabrics re dyeing send & received report

5. Party wise dyeing bill statement

6. Daily finish fabric receive status

7. Wastage finishing fabric status report

**4. Accessories Store:**

1. Accessories receive as per work order. Buyer, style, Po, color, size wise.

2. Accessories issue as per store requisition from concern dept.

3. Accessories allocation PO style wise from stock

4. Accessories return to supplier if required.

**Report:**

1. Buyer, style, PO, color wise accessories receives and issue status.

2. Buyer, style, PO, color wise accessories short/Excess report.

3. Challan wise accessories receive stratus

**5. General Store:**

1. Indirect material received against PO

2. Store requisition for general material

3. Store requisition approval

4. Issue material as per approved requisition

5. Acknowledge service as per service PO to generate the bill.

**Report:**

1. Stock flow Report and Stock Analysis Report

2. Current Stock Valuation Report

3. Item Movement Register

4. Statistical Report against Stock Flow.

5. Automatic report for short item according to lead time & daily consumption.

**6. IT Store:**

**Report:**

**Part 2:🏭 Inventory Management System Specification (Cut-to-Finish RMG Factory)**

**Prepared by:** Mahmudur Rahman  
**Purpose:** To design and develop a transparent, secure, and scalable multi-store inventory management system to eliminate corruption, mismanagement, and theft within a Cut-to-Finish RMG factory in Bangladesh.

**1️ System Overview**

This system manages **all inventory operations** across different stores in a Cut-to-Finish RMG factory — from purchase requisitions and goods receiving to internal issues, stock valuation, and audit tracking.  
It will maintain **traceability**, **multi-user authorization**, **role-based permissions**, and **multi-level approvals**, ensuring full visibility and control over inventory flow.

**2️ Types of Stores Covered**

| **Store Type** | **Common Items** |
| --- | --- |
| General Store | Stationery, cleaning, tools, PPE, safety items |
| Trims & Accessories Store | Buttons, zippers, threads, tags, labels, elastic, etc. |
| Fabric Store | Main, contrast, lining, interlining fabrics |
| Spare Parts Store | Machine parts, needles, belts, spares |
| IT Store | IT equipment, cables, routers, peripherals, software licenses |
| Maintenance Store | Electrical, plumbing, light fixtures, motors, hardware |
| Finished Goods Store | Ready garments for shipment |
| Leftover Store | Excess/returned production items |

**3️ Unique Identification & Traceability**

* **Primary Key (System Level):** UUID for all entities (items, users, transactions, etc.)
* **Items:** Unique *Item Code (SKU)* manually defined, human-readable (e.g., FAB-0012, ZIP-0054).
* **Fabrics:** Track via **Lot Number** (mandatory).
* **Serialized Assets:**
  + Machines → Serial Number
  + IT Equipment → Serial + MAC + IMEI (if applicable)
  + IT Fixed Assets → Asset Tag
* **Batch Number** for any batch-controlled or chemical items.
* **No QR/Barcode at initial stage**, but structure should allow future integration.

**4️ Location Hierarchy**

A consistent **location pathing system** is required for all physical storage:

**Building → Floor → Room → Rack → Bin**

Each location will be stored hierarchically using a parent-child model with a full path for query optimization (e.g., B1/F2/RM05/RK03/BN12).

**5️ Units of Measurement (UoM)**

* **Default Setup (Initial Stage):**
  + **Fabrics:** Kilogram (KG)
  + **Trims, Accessories, General Items:** Piece (PCS)
* **Future Scope:**  
  Add flexible **UoM conversion** (e.g., Roll → Meter → Yard) via item\_uom\_conversion table for items needing multi-UoM tracking.

**6️ Stock Control Parameters**

| **Parameter** | **Description** |
| --- | --- |
| **Reorder Level** | Minimum stock before reorder alert |
| **Safety Stock** | Buffer stock to prevent shortages |
| **Max Stock** | Maximum allowed quantity |
| **EOQ / Lead Time** | For optimized purchase planning |

**7️ Valuation & Accounting**

* **Valuation Method:** FIFO
* **Default Currency:** BDT (scope for USD, EUR, etc.)
* **Each item carries its unit purchase price** (unit\_rate).
* **When stock is received:**  
  → Add its total value to inventory\_value (BDT)
* **When issued:**  
  → Deduct corresponding FIFO-based value from inventory.
* **Maintain historical rate and valuation logs** for financial accuracy.
* **Tax Info:** store VAT/TAX rate at PO or GRN level for reporting.

**8️ Procurement & Receiving Workflow**

**a. Requisitions**

* **External Requisition (Purchase Request):** Request to buy new items not in stock.
* **Internal Requisition:** Request to issue available items from store.

**b. Documents**

| **Document** | **Purpose** | **Approval Required** |
| --- | --- | --- |
| **Requisition (PR)** | Item requested by a department | Dept. Head + Inventory Manager |
| **Purchase Order (PO)** | Official order to supplier | Procurement + Management |
| **Goods Received Note (GRN)** | Acknowledges items received | QC + Storekeeper |
| **Invoice** | Billing record linked to PO/GRN | Accounts verification |
| **Delivery Challan** | Record of goods delivered | Supplier/Receiver sign-off |
| **QC Report** | Inspection outcome | QC approval |

All documents are **versioned and traceable** via audit logs.

**9️ QC & Acceptance**

| **Field** | **Description** |
| --- | --- |
| **Inspection Status** | Pending / Accepted / Rejected / Rework |
| **QC Remarks** | Inspector’s notes |
| **Accepted Qty** | Quantity passed QC |
| **Rejected Qty** | Quantity failed QC |

**10 Stock Movement Metadata**

| **Field** | **Description** |
| --- | --- |
| **Transaction Type** | Purchase, Issue, Transfer, Adjustment, Return, Scrap |
| **Reference ID** | Links to document or source record |
| **Requested By / Approved By** | Traceability of personnel |
| **Reason Code** | e.g., Production use, Maintenance, Damage |
| **Timestamp, IP, User Details** | Full user tracking |

**11️ Audit Trail & Tamper Evidence**All critical transactions (create, update, approve, delete) are logged with:

* User ID & Name
* Role & Department
* Timestamp & IP Address
* Before/After Values (JSON diff)
* Approval chain with signatures (digital/electronic)

Logs are immutable and accessible only to SuperAdmin and Auditors.

**12️ Stock-taking / Physical Count**

**Example Flow:**

1. Storekeeper initiates a *Stocktake Session*.
2. Items counted physically by location.
3. Entered counts compared with system quantity.
4. Discrepancies recorded → sent to Inventory Manager for approval.
5. Approved variance automatically adjusts system balance (if authorized).

**Recorded Fields:**  
item\_id, location\_id, system\_qty, counted\_qty, variance, remarks, verified\_by, approved\_by

**13️ Adjustments & Exceptions**

* Record **damage, scrap, loss, or theft** incidents separately.
* Include reason, date, witness, approved\_by.
* Only authorized roles can approve adjustments.

**14️ IT Store – Specialized Data**

| **Field** | **Description** |
| --- | --- |
| **Asset Model, Manufacturer, Serial, MAC, IMEI** | Identification |
| **Warranty Start/End** | Warranty management |
| **Purchase Invoice No. & Value (BDT)** | Financial link |
| **Assigned To / Employee ID** | Asset ownership |
| **Assignment / Return Date** | Lifecycle tracking |
| **Status** | New / Active / Repair / Retired |
| **License Info** | Software name, expiry date |
| **Acceptance Sign-off** | Employee acknowledgment |
| **Value** | BDT valuation per item |

**15️ Notifications & Reports**

**In-system notifications** to alert:

* Low stock / Reorder level reached
* Aging or slow-moving stock
* High-value issue alerts
* Stock movement summary
* GRN pending QC
* Pending approvals

**Reports:**

* Stock valuation summary (by store, by category)
* Daily stock movement ledger
* Item history (in/out)
* Issue register (department-wise)
* Aging & consumption trends
* QC pass/fail reports
* Adjustment & variance logs

**16️ Security & Controls**

* **Two-person approval** required for high-value or sensitive items.
* **Role-Based Access Control (RBAC):**  
  Define permissions per module and restrict access by role.
* **Separation of Duties:**  
  Requester ≠ Approver ≠ Receiver for high-value items.
* **Soft Deletes** with timestamp for recoverability.
* **Encrypted Credentials & TLS** for all connections.
* **Immutable Audit Logs**.

**17️ User Hierarchy (Roles & Access)**

| **Role** | **Core Permissions** |
| --- | --- |
| **SuperAdmin** | Full access, configuration |
| **Factory Manager** | View/approve high-level data |
| **Inventory Manager** | Manage stores, approve issues/transfers |
| **Storekeeper** | Receive, issue, update stock |
| **Procurement Officer** | Create/approve PO, manage suppliers |
| **QC Inspector** | QC acceptance/rejection |
| **Maintenance/Engineering** | Request spare parts, maintenance approval |
| **IT Admin** | Manage IT store, assign/return assets |
| **Accounts/Finance** | View valuation, match invoices |
| **Department Head** | Approve requisitions and issues |
| **Requester/Employee** | Create requisitions only |
| **Auditor** | Read-only, view audit and reports |

**18️ Approval Workflow (Summarized)**

| **Flow** | **Steps** | **Notes** |
| --- | --- | --- |
| **External Requisition** | Requester → Dept Head → Inventory/Procurement | For purchase |
| **Internal Requisition** | Requester → Dept Head → Inventory Manager | For internal issue |
| **Purchase Flow** | PR → PO → GRN → QC → Accounts → Inventory | End-to-end |
| **IT Asset Assignment** | Requester → Dept Head → IT Admin | Sign-off required |
| **High-Value Issue** | Requester → Dept Head → Inventory Manager → Factory Manager | Two-person rule |

**19️ Database Schema Overview (Optimized for Django + PostgreSQL)**

* **Master Tables:** items, locations, suppliers, uoms, categories, users, roles
* **Transaction Tables:** purchase\_requests, purchase\_orders, grn, stock\_movements, stock\_balance, assets
* **Ledger Tables:** audit\_logs, approvals, notifications, stocktake\_sessions
* **Materialized View:** stock\_balance (auto-refreshed via trigger)

All PKs are UUIDs; soft deletes via deleted\_at.  
Foreign keys use cascading updates; integrity enforced at DB level.

**20️ Technical Recommendations**

| **Aspect** | **Recommendation** |
| --- | --- |
| **Backend** | Python + Django (REST Framework) |
| **Database** | PostgreSQL (UUIDs, JSONB support, Materialized Views) |
| **API Architecture** | REST + JWT auth, optional WebSocket for alerts |
| **Frontend** | React / Vue (optional for later phase) |
| **Triggers / Signals** | Auto-update stock\_balance after stock\_movements |
| **Valuation** | FIFO logic implemented at stock issue level |
| **Backup & Audit** | Daily snapshots; immutable logs (only super admin can edit it) |
| **Future Add-ons** | Barcode/QR integration, Vendor Portal, Report Export (XLS/PDF) |

✅ **End of Specification (As of now)**

**Part 3: Inventory Management System Draft Database Schema**

## 1) Draft Database schema (recommended tables & key columns)

### Core tables (short names, PKs)

* **companies**
  + company\_id UUID PK, name, address, timezone, created\_at, updated\_at
* **factories**
  + factory\_id UUID PK, company\_id FK, name, address, created\_at
* **locations** (warehouse/room/rack/bin)
  + location\_id UUID PK, factory\_id FK, parent\_location\_id FK nullable, code, name, type (warehouse/room/rack/bin), path (text), created\_at
* **users**
  + user\_id UUID PK, username, full\_name, email, password\_hash, active, factory\_id FK, created\_at
* **roles**
  + role\_id UUID PK, name, description
* **user\_roles**
  + user\_id FK, role\_id FK, assigned\_by, assigned\_at
* **permissions**
  + permission\_id, key (e.g., approve\_po), description
* **role\_permissions**
  + role\_id FK, permission\_id FK
* **suppliers**
  + supplier\_id UUID, name, contact\_person, phone, email, address, tin\_vat, payment\_terms, created\_at
* **item\_categories**
  + category\_id, name, parent\_id
* **uoms**
  + uom\_id, code (mtr/pcs/kg), description
* **items**
  + item\_id UUID PK, sku, name, category\_id FK, brand, default\_uom\_id FK, reorder\_level numeric, safety\_stock numeric, valuation\_method enum, is\_serialized boolean, is\_asset boolean, is\_consumable boolean, created\_at
* **item\_uom\_conversion**
  + item\_id FK, from\_uom\_id, to\_uom\_id, factor
* **purchase\_requests (pr)**
  + pr\_id, requester\_id FK, department, factory\_id FK, status, total\_value numeric, created\_at
* **purchase\_request\_line**
  + pr\_line\_id, pr\_id FK, item\_id FK, qty, uom\_id, required\_by, justification
* **purchase\_orders (po)**
  + po\_id, po\_no, supplier\_id FK, created\_by, factory\_id, status, total\_value, currency, created\_at
* **po\_line**
  + po\_line\_id, po\_id FK, item\_id FK, qty, uom\_id, unit\_price, delivery\_date
* **goods\_received\_notes (grn)**
  + grn\_id, grn\_no, po\_id FK nullable, supplier\_id FK, received\_by, received\_at, factory\_id, qc\_status, total\_value
* **grn\_line**
  + grn\_line\_id, grn\_id FK, item\_id FK, qty\_received, uom\_id, batch\_no, expiry\_date nullable, serial\_no nullable, unit\_price, qc\_remark
* **qc\_inspections**
  + qc\_id, grn\_id FK, inspector\_id, date, status, remarks, attachments jsonb
* **stock\_batches** (for batch-controlled items)
  + batch\_id, item\_id FK, batch\_no, location\_id FK, qty\_on\_hand numeric, uom\_id, expiry\_date nullable, manufacture\_date, created\_at
* **stock\_movements** (ledger of all In/Out/Transfers/Adjustments)
  + movement\_id UUID PK, item\_id FK, from\_location\_id nullable, to\_location\_id nullable, qty numeric, uom\_id, movement\_type enum (PURCHASE\_IN, ISSUE\_OUT, TRANSFER, ADJUSTMENT, RETURN, SCRAP), reference\_id (po/grn/pr/issue no), batch\_id nullable, serial\_no nullable, created\_by, created\_at, approved\_by nullable, approved\_at nullable, remarks
* **stock\_balance** (materialized view or aggregated table)
  + item\_id, location\_id, batch\_id nullable, qty\_on\_hand numeric, last\_movement\_at
* **inventory\_adjustments**
  + adjustment\_id, performed\_by, approved\_by, date, reason, attachments
* **stocktake\_sessions**
  + stocktake\_id, factory\_id, created\_by, start\_date, end\_date, status
* **stocktake\_lines**
  + stocktake\_line\_id, stocktake\_id, item\_id, location\_id, counted\_qty, system\_qty, variance, remarks
* **transfers**
  + transfer\_id, from\_location\_id, to\_location\_id, requested\_by, approved\_by, status, created\_at
* **assets (for IT / serialized assets)**
  + asset\_id, item\_id FK, asset\_tag, serial\_no, model, manufacturer, purchase\_date, warranty\_start, warranty\_end, invoice\_id FK, current\_status enum (Stock, Assigned, Repair, Retired), assigned\_to\_user\_id FK nullable, assigned\_at nullable, location\_id, attachments jsonb
* **asset\_assignment\_history**
  + assignment\_id, asset\_id, from\_user\_id, to\_user\_id, assigned\_by, assigned\_at, returned\_at, remarks
* **invoices**
  + invoice\_id, po\_id, supplier\_id, invoice\_no, amount, due\_date, status
* **returns**
  + return\_id, grn\_id nullable, po\_id nullable, supplier\_id, qty, reason, processed\_by
* **approvals** (generic approval flow)
  + approval\_id, entity\_type (PR/PO/GRN/Adjustment/Transfer/Issue), entity\_id, requested\_by, status, current\_step, approver\_role, approved\_by, approved\_at, notes, metadata jsonb
* **audit\_logs**
  + log\_id, user\_id, entity\_type, entity\_id, action (CREATE/UPDATE/DELETE/APPROVE), before jsonb, after jsonb, ip, timestamp
* **notifications**
  + notification\_id, user\_id, type, payload jsonb, is\_read, created\_at

### Indexes & constraints

* Index on items.sku (unique), stock\_movements.item\_id, stock\_balance(item\_id, location\_id), grn.po\_id.
* Foreign key constraints for referential integrity.
* Unique constraint on asset.asset\_tag and asset.serial\_no.
* Use DB triggers or application logic to update stock\_balance on stock\_movements.

## Quick ER / relationship summary (text)

* factory has many locations.
* items belong to category; item\_uom\_conversion maps UoMs.
* po → po\_line → items. grn references po and creates grn\_line.
* Each grn\_line creates stock\_movements (PURCHASE\_IN) and updates stock\_batches/stock\_balance.
* stock\_movements is the ledger for all changes; stock\_balance is aggregated state.
* assets are special items with serials and assignment history.
* users assigned to roles; approvals reference requests and maintain approval chain.
* audit\_logs capture all user actions for traceability.

✅ **End of Draft Database Schema (As of now)**

All of these are just the draft plan I created. I don’t know best practices but I don’t want to complicate too much by inserting too much useless details for the inventory. That’s why I really need your help to validate what I have done so far, give me your recommendation because ultimately you will be the developing this for me using free tools only. Please read this whole document with utmost concentration and tell me if I have missed anything crucial.

In part 1, I have talked about the purpose of different stores, some I did not mention there but when you read part 2 and 3 you will automatically understand.

In part 2, I have talked about some system specification that I think I should have.

In part 3, I have talked created a draft database schema, which may have some missing data in comparison to part 2.

Therefore, I humbly request you to read all these and tell me if the framework I chose (Python + Django) is suitable for this or should i choose Laravel + Mysql. I will go with whatever you say as I don’t know any of them, I will solely reply on your codes, structures and plans. After deciding the framework, we will move on to creating the final database schema (that is not missing anything). When the database schema is final, then we will move on to setting up the development environment.  
Let me mention another crucial information, after developing this system locally on my device, I will be using a server and client based model. I have another normal pc which will act as the server and the user PCs will be clients. Keeping these in mind, please go ahead with the tasks I have given you or you can ask me anything you are still don’t understand or confused about.