**📦 Inventory Module – Warehouse Management**

**1. Overview**

The Warehouse Management feature allows users to create and maintain multiple warehouse records, assign one as the primary warehouse, and manage their status (active/inactive). Warehouses are essential for tracking stock movements, purchase orders, sales orders, and reporting inventory balances per location.

**2. Functional Requirements**

**2.1 Add Warehouse**

* Users can add new warehouses via a form.
* Fields and rules are:
  + **Name** (required, unique)
  + **Shipping Address** (optional)
  + **Status** (Active/Inactive, default = Active)
  + **Primary** (checkbox, only one warehouse allowed as Primary)

**2.2 Edit Warehouse**

* Users can update warehouse details.
* Business rules:
  + Cannot change status to **Inactive** if it is the **Primary warehouse**.
  + Changing to Primary must auto-update the previously set Primary to non-primary.
  + Only one primary warehouse at a time

**2.3 Delete Warehouse**

* Users can delete a warehouse only if it has **no linked transactions**.
* If transactions exist, the warehouse should instead be set to **Inactive**.

**2.4 List Warehouses**

* System must display all warehouses in a grid view with search & filter options.
* **Columns:**
  + Warehouse Name
  + Shipping Address
  + Status
  + Primary (Yes/No)
  + Actions (Edit/Delete)
* **Filters/Sorting:**
  + By Status (Active/Inactive)
  + By Primary (Primary/Secondary)

**3. UI/UX**

**3.1 Add/Edit Form**

Fields: Name, Shipping Address, Status (dropdown), Primary (checkbox).  
Buttons: Save, Cancel.

**3.2 List View**

* Table/Grid format with pagination.
* Each row has action buttons: Edit, Delete.
* "Add New Warehouse" button at top.

**4. Database Design**

**Table: Warehouses**

| **Column** | **Type** | **Notes** |
| --- | --- | --- |
| WarehouseID | INT (PK, Auto Increment) | Unique identifier. |
| Name | VARCHAR(100) | Unique constraint. |
| ShippingAddress | VARCHAR(255) | Nullable. |
| Status | ENUM(‘Active’, ‘Inactive’) | Default = ‘Active’. |
| IsPrimary | BOOLEAN | Only one record can be TRUE. |
| CreatedAt | DATETIME | Auto-generated. |
| UpdatedAt | DATETIME | Auto-updated. |

**5. Business Rules**

1. Warehouse **Name must be unique**.
2. Only **one warehouse can be Primary**.
3. A warehouse **cannot be deleted** if it has linked inventory transactions.
4. If a user tries to set a warehouse with transactions to Inactive, a warning message should appear.

**6. Integration & Dependencies**

* **Inventory Transactions:** All stock in/out must reference a warehouse.
* **Sales Orders:** Warehouse determines from where stock is allocated.
* **Purchase Orders:** Warehouse determines where stock is received.
* **Reports:** Stock balances, valuations, and movements must support filtering by warehouse.

**📑 BRSD – Inventory Module**

**Section: Products & Services**

**1. Overview**

The **Products & Services** section manages all items the business buys, sells, or tracks. It supports:

* **Products** (physical goods, can be tracked in inventory).
* **Services** (non-inventory, sold without stock tracking).
* **Item Groups** (variants/bundles).
* **Taxes** applied through **multi-tax profiles**.

**2. Functional Requirements**

**2.1 Products**

Users can create and manage products with the following fields:

**Item Details**

* **Name** (Required, unique per category).
* **Item SKU** (Auto/manual, unique).
* **Description** (Optional).
* **Photo** (Optional, upload).
* **Category** (Dropdown, required).
* **Brand** (Dropdown, optional).
* **Supplier** (Dropdown, optional).
* **Barcode** (Optional, unique).

**Pricing & Discounts**

* **Purchase Price** (numeric).
* **Selling Price** (numeric).
* **Minimum Price** (numeric, lowest allowed sale price).
* **Discount** (numeric).
* **Discount Type** (% or Fixed).
* **Profit Margin** (auto-calculated if purchase & selling price set).

**Taxes**

* Attach **one or multiple tax profiles**.
* Tax profile(s) are defined in the **Tax Settings** screen.
* Each profile may contain one or more tax components (e.g., VAT 14%, Eco Fee 2%).
* Support for **Exclusive** (added on top) or **Inclusive** (built into price).
* Exclusive ( will be added over the price)
* Inclusive ( will be included in the price)

**Inventory Management**

* **Track Stock** (checkbox).
* **Initial Stock Level** (numeric, opening balance).
* **Low Stock Threshold** (numeric, for alerts).

**Other**

* Internal Notes (optional).
* Tags (multi-select, optional).
* Status (Active/Inactive).

**2.2 Services**

Users can create and manage services with the following fields:

* **Name** (Required).
* **Service Code** (Unique).
* **Description** (Optional).
* **Photo** (Optional).
* **Category** (Dropdown).
* **Supplier** (Optional).
* **Pricing**: Purchase Price, Unit Price, Minimum Price, Discount, Profit Margin.
* **Taxes**: One or multiple tax profiles (same as products).
* **Other**: Notes, Tags, Status.

📌 **Services do not include inventory tracking.**

**2.3 Item Groups**

* **Group multiple related products** (e.g., T-shirt in different sizes/colors).
* **Fields**: Name, Category, Brand, Description.
* **Items inside group**: SKU, Purchase Price, Selling Price, Barcode.
* **Attributes** (e.g., size, color) handled by **system-wide multiple attributes feature**.

**3. Business Rules**

1. SKU and Barcode must be unique.
2. Selling Price cannot be lower than Minimum Price.
3. If Track Stock = Yes → product must belong to a warehouse.
4. Services cannot have stock fields.
5. Products/Services may have **multiple tax profiles**.
6. At least one tax profile must be marked **Primary** when multiple assigned.
7. Tax profiles can be **Exclusive** (added to net price) or **Inclusive** (already in selling price).
8. Item Groups must allow **bulk generation of SKUs** for variants.

**4. Database Design (Simplified)**

**Products**

* ProductID (PK)
* Name, SKU, Description, CategoryID, BrandID, SupplierID, Barcode
* PurchasePrice, SellingPrice, MinPrice, Discount, DiscountType, ProfitMargin
* TrackStock, InitialStock, LowStockThreshold
* Status, CreatedAt, UpdatedAt

**Services**

* ServiceID (PK)
* Name, Code, Description, CategoryID, SupplierID
* PurchasePrice, UnitPrice, MinPrice, Discount, DiscountType, ProfitMargin
* Status, CreatedAt, UpdatedAt

**ItemGroups**

* GroupID (PK), Name, CategoryID, BrandID, Description

**ItemGroupItems**

* GroupItemID (PK), GroupID, ProductID, SKU, PurchasePrice, SellingPrice, Barcode

**ProductTaxProfiles**

* ProductID, TaxProfileID, IsPrimary

**5. Integrations**

* **Warehouses**: Products with stock tracking linked to warehouse.
* **Purchasing**: Supplier + purchase price.
* **Sales**: Selling price + tax profiles.
* **Accounting**: Tax components mapped to GL accounts.
* **Reports**: Product sales, stock levels, tax summaries.

**6. Reports**

* Product list with stock availability.
* Sales & purchases per product/service.
* Profit margin report.
* Tax summary by product/service.
* Low stock / out of stock alerts.

**Section: Products & Services (Updated with Product Details page)**

**1. Overview**

The **Products & Services** module defines all sellable items and services.  
Each product or service can be linked with tax profiles, suppliers, pricing, and stock management rules.  
Additionally, the **Product Detail Page** provides real-time insights into stock, transactions, timelines, and group relationships.

**2. Functional Requirements**

**2.1 Product Details Page**

When a user opens a product record, the system must show:

**Header Information**

* Product Name & Status (In Stock / Out of Stock).
* Quick Actions: Edit, Delete, Add Transaction, ~~Generate Voucher~~, Clone.

**Summary Panel**

* **On Hand Stock** (by warehouse).
* **Total Sold Items** (lifetime).
* **Sales – Last 28 Days / Last 7 Days**.
* **Average Unit Cost** (weighted average cost).
* Action: **Add Stock Transaction** button.

**Details Section**

* SKU, Purchase Price, Selling Price, Barcode.
* Linked Item Group (if applicable).
* Product photo placeholder.

**2.2 Stock Transactions Tab**

* Shows all stock movements for the product.
* Columns: Transaction ID, Date, Type (Manual Adjustment, Purchase, Sale, Transfer), Quantity Move (+/-), Unit Price, Stock Level After transaction.
* Filters: Date Range, Transaction Type, Warehouse.
* Business Rule: Each transaction updates **On Hand Stock** and recalculates **Average Unit Cost**.

**2.3 Timeline Tab**

* Chronological record of stock changes for this product.
* Each entry includes: Action type, Item reference, User, Timestamp, Stock balance, Average Price.
* Useful for quick investigation of stock changes.

**2.4 Activity Log Tab**

* Detailed audit log of user actions: Create, Edit, Delete, Transactions, Price Updates.
* Captures: User ID, Action Type, Before/After values (if applicable), Timestamp.
* Business Rule: Must be **read-only** and cannot be deleted.

**2.5 Item Group Tab**

* Displays group relationships of the product.
* Shows Item Group Name, Item Name, SKU, Quantity, Purchase Price, Selling Price, Barcode.
* Business Rule: A product can belong to **zero or one Item Group** at a time.

**3. Business Rules (Additions)**

1. **Average Unit Cost Calculation**
   * Weighted Average Method:
   * New Avg Cost = (Old Stock × Old Avg Cost + New Stock × New Unit Cost) ÷ (Old Stock + New Stock)
   * Updates automatically on each stock transaction.
2. **Stock Transactions** must always reconcile with warehouse balances.
3. **Timeline vs Activity Log**:
   * Timeline = operational stock movements.
   * Activity Log = user actions (audit trail).
4. **Item Groups** link products by variants; quantities update independently.

**4. Database Enhancements**

**StockTransactions**

* TransactionID (PK)
* ProductID (FK)
* WarehouseID (FK)
* TransactionType (ENUM: Adjustment, Purchase, Sale, Transfer, Return, etc.)
* Quantity (Positive/Negative)
* UnitCost
* StockLevelAfter
* CreatedBy (UserID)
* CreatedAt

**ProductStats** *(aggregated values for fast UI loading)*

* ProductID
* TotalSold (lifetime)
* SoldLast28Days
* SoldLast7Days
* OnHandStock (current)
* AvgUnitCost

**ActivityLog**

* LogID (PK)
* ProductID (FK)
* ActionType (Create, Update, Delete, Transaction, Price Change)
* UserID
* Timestamp
* BeforeValues (JSON)
* AfterValues (JSON)

**5. Integrations**

* **Warehouses**: All stock transactions tied to a warehouse.
* **Purchasing**: Incoming stock (Purchase Orders) adds to transactions.
* **Sales**: Outgoing stock (Invoices, Deliveries) reduces transactions.
* **Accounting**: Average Unit Cost used in COGS (Cost of Goods Sold).
* **Item Groups**: Keeps group link visible at product level.

**6. Reports (New with Detail Page)**

* **Product Movement Report** (all transactions for a product).
* **Product Timeline Report** (chronological events).
* **Audit Report** (user activity logs).
* **Average Cost History Report** (changes in average cost over time).

**📑 BRSD – Inventory Module**

**Section: Taxes Settings**

**1. Overview**

The **Taxes Settings** section allows administrators to define and manage all tax rules that apply to products and services.

* Each tax rule is a **Tax Component** (e.g., VAT 14%).
* Multiple tax components can be combined into a **Tax Profile** (e.g., VAT 14% + Environmental Fee 2%).
* Profiles are reusable and can be attached to multiple products/services.

**2. Functional Requirements**

**2.1 Tax Component Management**

* Users can **add, edit, delete** tax components.
* Fields:
  + **Tax Name** (e.g., VAT, Eco Fee).
  + **Tax Value** (numeric, rate or fixed amount).
  + **Rate Type** (% or Fixed).
  + **Included/Exclusive**:
    - **Exclusive** → added on top of the item price.
    - **Inclusive** → included within the item price.

**2.2 Tax Profiles**

* Users can group multiple tax components into a single **profile**.
* Example:
  + Profile “Egypt VAT + Eco” = VAT 14% + Eco Fee 2%.
* Each product/service can be linked to one or more profiles.

**2.3 Application Rules**

* Taxes can apply to **Sales, Purchases, or Both**.
* A tax profile must have at least one component.
* A product/service may have multiple profiles, but one should be **Primary** (used by default).

**3. Business Rules**

1. Tax Name must be unique.
2. Tax Value must be a positive numeric value.
3. Inclusive vs Exclusive must be respected in calculations:
   * **Inclusive**: system backs out the tax portion from the price.
   * **Exclusive**: system adds the tax on top of base price.
4. Taxes must support compounding (e.g., one tax applied after another).
5. A tax profile cannot be deleted if currently linked to active products/services.
6. Tax breakdown per transaction must be stored for compliance reporting.

**4. Database Design (Simplified)**

**TaxComponents**

* TaxComponentID (PK)
* Name (e.g., VAT)
* RateType (ENUM: %, Fixed)
* RateValue (Decimal)
* IncludedType (ENUM: Inclusive, Exclusive)
* AppliesOn (ENUM: Sales, Purchases, Both)
* Active (Boolean)
* CreatedAt, UpdatedAt

**TaxProfiles**

* TaxProfileID (PK)
* Name, Description
* Active (Boolean)
* CreatedAt, UpdatedAt

**TaxProfileComponents**

* TaxProfileID (FK)
* TaxComponentID (FK)
* Priority (int – determines calculation order)

**5. UI/UX**

**Tax Settings Screen**

* Grid/List of defined taxes: Tax Name, Value, %, Inclusive/Exclusive, Delete option.
* “Add Tax” button → opens row with fields.
* “Save” → persists changes.

**Tax Profiles Screen** *(if separate)*

* Name, Description.
* Multi-select components.
* Option to set calculation order (Priority).

**6. Integration**

* **Products & Services**: Select tax profiles when defining products or services.
* **Purchasing/Sales**: Taxes auto-applied per line item.
* **Accounting**: Each tax component maps to a GL account.
* **Reports**: Tax breakdowns, summaries by component/profile.

**7. Reports**

* Tax Summary Report (by profile/component).
* Tax Breakdown Report (per invoice/transaction line).
* Zero-rated/Exempt report (if exemptions apply).

Ending for week #1

**📑 BRSD – Inventory Module**

**Section: Requisitions**

**1. Overview**

The **Requisition module** is used to formally request and record stock movements.  
It supports **Inbound Requisitions** (stock coming in) and **Outbound Requisitions** (stock issued/removed).  
Requisitions ensure controlled and auditable stock adjustments beyond sales and purchases.

**2. Functional Requirements**

**2.1 Requisition Information**

* **Requisition Type**: Dropdown (Inbound, Outbound, Transfer).
* **Date**: Auto-filled with current date, editable.
* **Number**: Auto-generated sequential reference, required.

**2.2 Other Information**

* **Journal Account**: Link requisition to a GL account for accounting purposes.
* **Supplier**: For inbound requisitions, optional link to supplier.
* **Notes**: Free text field.
* **Attachments**: Upload supporting documents (e.g., supplier invoice, approval sheet).

**2.3 Items Section**

* Add one or more items to requisition.
* Fields per item:
  + **Item** (dropdown search).
  + **Unit Price**.
  + **Quantity**.
  + **Stock On Hand** (current quantity in warehouse).
  + **New Stock On Hand** (calculated after requisition).
  + **Total** (Unit Price × Quantity).
* Bottom row: **Items Total** (sum of all lines).

**2.4 Actions**

* **Confirm** → validates requisition, updates stock, creates accounting entry.
* **Cancel/Delete** → allowed only before confirmation.
* **Edit** → only for draft requisitions.

**3. Business Rules**

1. **Requisition Number** must be unique and sequential.
2. **Requisition Type** determines stock effect:
   * **Inbound** → increases stock.
   * **Outbound** → decreases stock.
   * **Transfer** (if supported) → decreases stock in one warehouse and increases in another.
3. **Stock On Hand** and **New Stock On Hand** must be automatically calculated.
4. Negative stock not allowed unless system permits **backorder/negative inventory**.
5. Confirmed requisitions cannot be deleted; they can only be reversed with an opposite requisition.
6. Journal Account must be mapped for accounting integration (e.g., Stock Adjustment, Raw Materials, Expense).
7. Supplier is optional for inbound requisitions but mandatory if tied to purchases.

**4. Database Design (Simplified)**

**Requisitions**

* RequisitionID (PK)
* Type (Inbound, Outbound, Transfer)
* Date
* Number (unique)
* JournalAccountID (FK)
* SupplierID (FK, optional)
* Notes, Attachments
* Status (Draft, Confirmed, Cancelled)
* CreatedBy, CreatedAt, UpdatedAt

**RequisitionItems**

* ReqItemID (PK)
* RequisitionID (FK)
* ProductID (FK)
* UnitPrice
* Quantity
* StockOnHand (before)
* NewStockOnHand (after)
* LineTotal

**5. Integrations**

* **Warehouses**: Adjusts stock levels.
* **Products**: Updates product balances and average unit cost.
* **Accounting**: Journal entries posted against selected account.
* **Suppliers**: For inbound requisitions linked with vendors.
* **Reports**: Stock adjustment analysis, requisition logs.

**6. Reports**

* **Requisition Register** (all requisitions with filters by type/date/user).
* **Stock Adjustment Report** (total impact of requisitions).
* **Inbound/Outbound Movement Report**.
* **Audit Log Report** (who created/confirmed requisitions).

**📑 BRSD – Inventory Module**

**Section: Requisitions (with Workflow States)**

**1. Overview**

The **Requisition module** manages stock movements outside of normal purchase/sales cycles.  
It supports **Inbound**, **Outbound**, and optionally **Transfer requisitions**.  
Each requisition passes through workflow states to ensure **control, approval, and auditability**.

**2. Workflow States**

| **State** | **Description** | **Actions Available** |
| --- | --- | --- |
| **Draft** | Requisition created but not yet submitted. Editable by creator. | Edit, Delete, Submit |
| **Submitted** | Sent for approval. Locked for editing except by approvers. | Approve, Reject, Cancel |
| **Approved** | Reviewed and authorized by designated approver. | Confirm (posts to stock & accounting) |
| **Confirmed** | Final state. Stock updated, accounting entry posted. Immutable record. | Reverse (creates opposite requisition if needed) |
| **Rejected** | Denied by approver. Remains in system for audit trail. | None (read-only) |
| **Cancelled** | Voided before approval/confirmation. No effect on stock or accounts. | None (read-only) |

**3. Functional Requirements (Extended)**

**3.1 Requisition Information**

* Requisition Type, Date, Number (auto), Journal Account, Supplier, Notes, Attachments.
* System auto-assigns **Status = Draft** at creation.

**3.2 Approval Workflow**

* When user **Submits** requisition:
  + Status → Submitted.
  + Notification sent to approvers (role-based).
* Approvers can:
  + **Approve** → Status → Approved.
  + **Reject** → Status → Rejected.
* Once Approved, only authorized users can **Confirm** → posts stock & accounting.

**3.3 Stock Update**

* Confirmed requisition updates:
  + **On Hand Stock** (by warehouse).
  + **Average Unit Cost** (if inbound).
  + **Stock Movements Table** (audit).

**3.4 Reversal**

* To undo a confirmed requisition, system creates an **opposite transaction** (reverse requisition).
* Example: Outbound 10 units reversed by Inbound 10 units with reference to original.

**4. Business Rules (Extended)**

1. Only **Draft** requisitions can be **edited or deleted**.
2. Submitted requisitions cannot be edited; only approvers can approve/reject.
3. Approved requisitions require **confirmation** to affect stock & accounting.
4. Rejected requisitions remain in history but never affect stock/accounts.
5. Cancelled requisitions are ignored for stock/accounts but remain auditable.
6. Each requisition must keep **full audit log**: creator, approver, confirmer, timestamps.
7. Only authorized roles can perform **Approve / Confirm / Reverse** actions.

**5. Database Design (Additions)**

**Requisitions** *(new columns)*

* Status (ENUM: Draft, Submitted, Approved, Confirmed, Rejected, Cancelled)
* SubmittedBy, SubmittedAt
* ApprovedBy, ApprovedAt
* ConfirmedBy, ConfirmedAt
* ReversedBy, ReversedAt
* ParentRequisitionID (for reversal link)

**RequisitionApprovalLog**

* LogID
* RequisitionID
* Action (Submit, Approve, Reject, Confirm, Cancel, Reverse)
* UserID
* Timestamp
* Comments

**6. Integration**

* **Warehouses**: Updates stock at confirmation.
* **Accounting**: Journal entries only posted once requisition is Confirmed.
* **Workflow Engine**: Approval rules configurable by role/department.
* **Notifications**: Approvers notified on requisition submission.

**7. Reports (Extended)**

* Requisition Workflow Report (with states & timestamps).
* Pending Approval Requisitions (per user/role).
* Rejected/Cancelled Requisitions Log.
* Requisition Reversals Report.

✅ Now requisitions follow a **controlled approval process**:  
**Draft → Submitted → Approved → Confirmed → (Reversed if needed)** with clear audit trails.

**📑 BRSD – Inventory Module**

**Section: Price List (Full Business Specification)**

**1. Overview**

The **Price List Module** manages product and service pricing for sales and purchases.  
It supports:

* Multiple price lists (Retail, Wholesale, Export, Supplier Pricing).
* Multi-currency support.
* Dynamic pricing rules (markup %, margin-based).
* Bulk/volume-based discounts.
* Seasonal/contract-based promotional pricing.
* Customer/supplier-specific assignments.

This ensures pricing is flexible, competitive, and automatically enforced during transactions.

**2. Functional Requirements**

**2.1 Price List Management**

* Create/Edit/Delete price lists.
* Fields: Name, Type (Sales or Purchase), Currency, Validity Period (From/To), Default Flag, Status (Active/Inactive).
* Copy/Clone existing price lists.
* Import/Export price list data.

**2.2 Price List Items**

For each product/service in a price list:

* Product/Service reference.
* Base Price (from master item).
* List Price (manually set or rule-based).
* Discount (Fixed or %).
* Final Price (calculated).
* Tax Profile (default from product but can be overridden).
* Effective dates (per item).

**2.3 Dynamic Pricing Rules**

* Rules can be applied to **entire price list** or **specific items**.
* Supported rules:
  1. **Markup %** → Final Price = Purchase Price × (1 + Markup%).
  2. **Margin %** → Final Price = Purchase Price ÷ (1 – Margin%).
  3. **Fixed Adjustment** → Add/subtract fixed value.
  4. **Currency Conversion** → Auto-update based on exchange rates.
  5. **Promotional Discounts** → Valid only for defined period (e.g., “Black Friday –10%”).

**2.4 Bulk/Volume Discounts**

* Define tiered discounts per item:
  + Example:
    - Qty 1–10 → 0% discount
    - Qty 11–50 → 5% discount
    - Qty 51+ → 10% discount
* Automatically applied when quantity meets threshold in sales order/invoice.

**2.5 Customer & Supplier Assignments**

* Assign price lists to:
  + Customers (VIP, Resellers).
  + Customer Groups (Retail, Wholesale, Export).
  + Suppliers (special negotiated purchase pricing).
* System auto-selects price list when transaction is created for assigned entity.

**2.6 Validity & Priority Rules**

* Price lists and individual items may have **validity periods** (from/to).
* Priority:
  1. Customer-specific Price List.
  2. Customer Group Price List.
  3. Default Sales Price List.
* If multiple lists valid, system applies **highest priority + lowest price** rule (configurable).

**2.7 Transaction Integration**

* When creating sales or purchase orders:
  + System auto-applies correct price list.
  + Bulk discounts applied if quantity threshold met.
  + Promotions applied if within validity period.
  + User overrides allowed only with special permission.

**3. Business Rules**

1. Each active product/service must have a price entry in at least one active price list.
2. Customer/Supplier must be linked to exactly one default price list.
3. Expired price lists cannot be applied.
4. Overlapping price lists → priority rules must resolve conflict.
5. Promotions always override standard pricing during their validity period.
6. Discounts cannot result in negative prices.
7. Price overrides at transaction level must be logged in the **Activity Log**.
8. Price List currency must be consistent; conversion uses system exchange rates.
9. Price List cannot be deleted if it has historical transactions; only deactivated.

**4. Database Design (Simplified)**

**PriceLists**

* PriceListID (PK)
* Name
* Type (Sales/Purchase)
* CurrencyID (FK)
* ValidFrom, ValidTo
* IsDefault (Boolean)
* Status (Active/Inactive)
* CreatedAt, UpdatedAt

**PriceListItems**

* PriceListItemID (PK)
* PriceListID (FK)
* ProductID/ServiceID (FK)
* BasePrice
* ListPrice
* DiscountValue
* DiscountType (% / Fixed)
* FinalPrice (calculated)
* TaxProfileID (FK, optional override)
* ValidFrom, ValidTo

**PriceListRules**

* RuleID (PK)
* PriceListID (FK)
* RuleType (Markup, Margin, Fixed Adjustment, Currency Conversion, Promotion)
* Value
* StartDate, EndDate
* Priority

**BulkDiscounts**

* BulkDiscountID (PK)
* ProductID (FK)
* PriceListID (FK)
* MinQty
* MaxQty
* DiscountType (% / Fixed)
* DiscountValue

**Assignments**

* AssignmentID (PK)
* EntityType (Customer/Supplier/Group)
* EntityID (FK)
* PriceListID (FK)

**5. UI/UX Features**

* **Price List Grid**: List of price lists with filters (Active, Validity, Currency).
* **Price List Detail Page**: Header info + Item table (with base price, list price, discount, final price).
* **Rules Tab**: Manage dynamic pricing rules.
* **Bulk Discounts Tab**: Define volume discounts per item.
* **Assignments Tab**: Manage linked customers/suppliers.
* **Preview Mode**: Show simulated transaction prices for testing.

**6. Integration**

* **Products & Services** → Source of base prices.
* **Taxes** → Integrated via tax profiles.
* **Customers/Suppliers** → Assignment of price lists.
* **Sales Orders/Invoices** → Price list applied automatically.
* **Purchases** → Supplier price list used for cost control.
* **Accounting** → Final prices flow to GL entries.
* **Reports** → Profit margin analysis using price list vs. cost.

**7. Reports**

* Price List Register (all active/inactive lists).
* Customer/Supplier Price Assignment Report.
* Product Price Comparison (across lists).
* Bulk Discount Utilization Report.
* Promotion Impact Report (sales uplift during promotional period).
* Margin Analysis Report (price list vs cost).

**📑 BRSD – Inventory Module**

**Section: Price List – Example Pricing Calculation Flow**

**1. Calculation Flow**

When a **Sales Order / Invoice** is created, the system calculates the **Final Invoice Price** in the following order:

**Step 1 – Select Price List**

* System checks in order of priority:
  1. Customer-specific price list.
  2. Customer group price list.
  3. Default Sales price list.
* If multiple valid → applies **highest priority**.

**Step 2 – Get Base Price**

* Pull base price from **Product Master**.
  + Example: Product = "Laptop X" → Base Price = **1000 L.E.**

**Step 3 – Apply Price List Rule**

* If dynamic rules exist in price list → apply rule to base price.

Examples:

* **Markup Rule** (20%): 1000 × 1.20 = **1200 L.E.**
* **Margin Rule** (25%): 1000 ÷ (1 – 0.25) = **1333.33 L.E.**
* **Fixed Adjustment** (+100): 1000 + 100 = **1100 L.E.**
* **Promotional Discount** (–10%, valid date): 1200 × 0.90 = **1080 L.E.**

**Step 4 – Apply Price List Discount (if any)**

* Check discount defined at **item level in price list**.
  + Example: Discount = 5% → 1080 × 0.95 = **1026 L.E.**

**Step 5 – Apply Bulk/Volume Discounts**

* If order quantity matches discount tiers → apply volume discount.

Example:

* Ordered Qty = 20 units
* Bulk Rule: 11–50 units = 5% discount
* 1026 × 0.95 = **974.70 L.E. per unit**

**Step 6 – Taxes (from Tax Profile)**

* Add taxes defined in item’s **tax profile** (multiple allowed).

Example:

* VAT 14% (Exclusive).
* 974.70 × 1.14 = **1111.16 L.E. per unit (Final Invoice Price)**

**Step 7 – Round & Validate**

* Apply rounding rules (e.g., 2 decimals).
* Ensure final price is not negative.
* Record breakdown in transaction log.

**2. Worked Example (Full Calculation)**

| **Step** | **Calculation** | **Result (L.E.)** |
| --- | --- | --- |
| Base Price | Product Master | 1000.00 |
| Markup Rule (20%) | 1000 × 1.20 | 1200.00 |
| Promotional Discount (10%) | 1200 × 0.90 | 1080.00 |
| Price List Discount (5%) | 1080 × 0.95 | 1026.00 |
| Bulk Discount (5% for Qty > 10) | 1026 × 0.95 | 974.70 |
| VAT 14% (Exclusive) | 974.70 × 1.14 | 1111.16 |
| **Final Invoice Price per unit** |  | **1111.16** |

If Qty = 20 units →  
**Total Invoice Value = 1111.16 × 20 = 22,223.20 L.E.**

**3. Business Rules for Calculation**

1. Rules are applied in **fixed sequence**:  
   **Base Price → Price List Rule → Price List Discount → Bulk Discount → Taxes**
2. Promotions override regular price list rules during their validity period.
3. If multiple taxes apply, system calculates sequentially (VAT + Service Tax, etc.).
4. All applied rules must be logged in the **Price Calculation Log** for auditability.
5. If user overrides final price manually → system stores **override reason** and **user ID** in Activity Log.

**4. Database Enhancements**

**PriceCalculationLog**

* LogID
* TransactionID
* ProductID
* AppliedRule (Markup/Discount/Tax/etc.)
* ValueBefore
* ValueAfter
* UserID (if override)
* Timestamp

**📑 BRSD – Inventory Module**

**Section: Stocktaking**

**1) Overview**

Stocktaking lets users count physical inventory for a **warehouse** and reconcile differences with the **system count**. The process produces per-item **shortage/overage** adjustments and (optionally) posts them to inventory & accounting.

**2) Key Screens & Actions (from your UI)**

* **Create Stocktaking**
  + Fields: **Warehouse**\*, **Number**\* (auto), **Date & Time**, **Notes**.
  + Action: **Do Stocktaking** (starts a session).
* **Do Stocktaking (Session)**
  + Top bar: progress (e.g., “0 / 1 Recorded Products”), **Done**, **Add Item**.
  + Toolbar: **Add Item**, **Import**, **Export**, **Add All Products From Warehouse**, menu for more.
  + Grid columns: Product, Unit, **Physical Count** (editable), **System Count** (read-only), **Shortage/Overage**, **Image**, **Note**, **Remove**.
  + Per-line “camera” for proof photo.
  + Mode switch: **Update System Count** ▼ (behavior, see Business Rules).
* **Add Item modal**
  + Fields: **Select Product**\*, **Physical Count**\*, **System Count** (shown), **Note**.
  + Save & Continue / Cancel.

**3) Functional Requirements**

**3.1 Create & Manage Sessions**

* Start a **Stocktaking Session** for a single **Warehouse** (required).
* Auto-generate **Number** (sequential; editable if permission allows).
* Capture **Date & Time** and **Notes**.
* Session **Status**: Draft → Counting → Review → Posted (see Workflow).

**3.2 Add Products to Count**

* **Add Item** (search by name/SKU/barcode).
* **Add All Products From Warehouse**: bulk-add every active product with on-hand > 0 (config toggle to include zero).
* **Import**: CSV/XLS template with Product, PhysicalCount, Note, ImageURL (optional).
* **Export**: template or results for offline counting.

**3.3 Enter Counts**

* For each product, user types **Physical Count**.
* System shows current **System Count** (snapshot logic below).
* System computes **Shortage/Overage = Physical − System** in real time.
* Optional **photo** and **note** per line.

**3.4 Update System Option**

* Session has a mode selector (top-right): **Update System Count** (Yes/No).
  + **Yes** → on Post, create inventory **adjustment transactions** to bring stock to Physical Count.
  + **No** → record variances only (no stock movement); useful for dry runs/audits.

**3.5 Post & Adjust**

* On **Done** → **Review** screen (totals, per-line variances, value impact).
* On **Post**:
  + Create adjustment movements per item:
    - **Shortage (<0)** → decrease stock (write-off).
    - **Overage (>0)** → increase stock (write-on).
  + Recompute **On-hand** and **Average Unit Cost** according to costing policy (see 4.4).
  + Write **Activity Log** and (if enabled) **Accounting Entries**.

**3.6 Search & History**

* List all sessions with filters by Warehouse, Date, Status, Created By.
* Open a past session to view lines, images, imports, and audit trail.

**4) Business Rules**

**4.1 Workflow & Permissions**

* **Statuses**
  + Draft: header created; editable; no items yet (or items editable).
  + Counting: items added; physical counts can be edited; system locks session header.
  + Review: counts completed; adjustments preview; requires approver sign-off.
  + Posted: adjustments (and accounting) recorded; read-only.
* **Permissions**
  + Creator: create/edit in Draft/Counting, attach photos, import/export.
  + Approver: move to Posted, override value method, reject back to Counting.
  + Auditor (read-only): view sessions, exports, and logs.

**4.2 Snapshot & Concurrency**

* At **session start**, take a **System Count Snapshot** for all included products (qty per warehouse).
* While in Counting, new inventory transactions may occur; when posting:
  + Option A (recommended default): **Rebase to Live** — compare Physical Count to **current** on-hand, not the original snapshot; show a warning if drift > threshold.
  + Option B (strict freeze): **Reserve/Frozen** — temporarily suspend stock movements for counted SKUs/warehouse during the session (requires operational buy-in).
* System flags items whose system count changed > threshold since snapshot.

**4.3 Validation**

* Physical Count must be **≥ 0**.
* A product can appear only **once** per session.
* Posting requires at least **one recorded product** or the session is rejected.
* If **Update System Count = No**, posting must **not** create movements.

**4.4 Valuation (Cost Impact)**

* **Shortage (write-off)**:
  + Quantity decrease at **current Average Unit Cost (AUC)** of the warehouse.
  + Debit: Inventory Shrinkage Expense; Credit: Inventory (stock).
* **Overage (write-on)**:
  + Quantity increase at **Valuation Basis** (configurable per company):
    1. **AUC** (current average), or
    2. **Last Purchase Cost**, or
    3. **Manual override** per line (approver only).
  + Debit: Inventory (stock); Credit: Inventory Gain (Other Income) or Adjustment Clearing.
* After posting, recompute AUC:
  + AUC\_new = (Qty\_old × AUC\_old + AddedQty × ValuationCost) ÷ (Qty\_old + AddedQty)
  + For write-offs, only quantities change; AUC stays unless policy allows recalculation.

**4.5 Accounting (if enabled)**

* Create one **Journal Entry per session** (summarized) or **per line** (detailed), company setting.
* Multi-warehouse companies map adjustments to warehouse-specific inventory accounts.

**4.6 Audit & Evidence**

* Store **who/when** entered each line, edits, approvals, and posting.
* Keep **images** (optional) per line as evidence.
* Persist **import file** and generated **export** copies.

**5) Data Model (simplified)**

**StocktakingHeader**

* StocktakingID (PK)
* WarehouseID (FK)
* Number (unique)
* DateTime
* Notes
* Status ENUM(Draft,Counting,Review,Posted)
* UpdateSystem BOOL
* CreatedBy,CreatedAt,UpdatedAt
* ApprovedBy,ApprovedAt,PostedBy,PostedAt

**StocktakingLine**

* LineID (PK)
* StocktakingID (FK)
* ProductID (FK)
* UnitID (FK, optional)
* PhysicalQty DECIMAL(18,3)
* SystemQtySnapshot DECIMAL(18,3)
* SystemQtyAtPost DECIMAL(18,3)
* VarianceQty (computed)
* ValuationCost DECIMAL(18,4) (basis used for write-on)
* VarianceValue DECIMAL(18,4)
* Note TEXT
* ImagePath VARCHAR
* CreatedBy,CreatedAt,UpdatedAt

**StocktakingAttachment** (optional)

* AttachmentID, StocktakingID, FilePath, UploadedBy, UploadedAt

**StockSnapshot** (optional table when not storing per line)

* StocktakingID, ProductID, QtyAtStart

**6) Algorithms**

**Variance**

VarianceQty = PhysicalQty - SystemQtyAtPost (or SystemQtySnapshot per policy)

VarianceValue =

if VarianceQty > 0: VarianceQty × ValuationCost (write-on)

if VarianceQty < 0: |VarianceQty| × AUC\_current (write-off)

**Posting (pseudo)**

1. For each line:
   * Read SystemQtyAtPost.
   * Compute VarianceQty. If 0 → skip.
   * Create **InventoryMovement** (Adjustment In/Out).
   * Build accounting entry per rule (5.1).
2. Update product warehouse on-hand and (if write-on) AUC.
3. Mark header **Posted**; write audit log.

**7) Imports/Exports**

**Import template columns**

* SKU / Barcode / ProductCode
* PhysicalQty
* Note
* ImageURL (optional)

**Validation**: reject unknown SKUs; collect all issues and surface in an error report.

**Export options**

* Blank sheet (with SystemQty snapshot for reference).
* Results with Physical, System, Variance, Value, image links.

**8) UI/UX Details**

* **Add All Products From Warehouse**: confirm modal explaining potential volume; allow filter by category/brand.
* **Update System Count ▼** menu:
  + Update & Post Adjustments (default)
  + Record Only (No Adjustments)
* **Row coloring**: shortages in red, overages in green.
* **Progress pill**: “X / Y Recorded Products”.
* **Done** button:
  + If lines missing Physical Count → prompt to continue or mark “Not Counted”.
* **Line photo**: supports mobile capture.

**9) Roles & Permissions**

* **Stock Controller**: create sessions, count, import/export.
* **Inventory Manager**: approve, post, choose valuation basis.
* **Auditor**: view all, export, no edits.
* Optional **Segregation of Duties**: creator cannot post their own session.

**10) Reports**

* **Stocktaking Register** (filters: warehouse/date/status/posted by).
* **Variance Summary** (qty & value by product/category/warehouse).
* **Shrinkage/Overage Trends** (periodic).
* **Exception Report** (large variances, high-value items).
* **Uncounted Items** (in a session).

**📑 Business Requirements – Product Settings**

**1. Purpose**

Product Settings provide configuration options to standardize how products and services are defined, classified, and managed across the system. These settings ensure consistency in product identification, categorization, measurement, taxation, and branding.

**2. Functional Requirements**

**2.1 Barcode Settings**

* **Fields & Options:**
  + **Barcode Type** (Dropdown: Code 128, EAN-13, UPC, QR Code, etc.)
  + **Enable Weight Embedded Barcode** (Boolean)
  + **Embedded Barcode Format** (Text pattern, e.g., XXXXXXXXWWWWWWPPPPN)
  + **Weight Unit Divider** (Numeric, to convert embedded weight into system units)
  + **Currency Divider** (Numeric, to parse price from barcode)
* **Business Rules:**
  + Barcode must be unique per product.
  + If weight embedded is enabled, system should automatically read and parse weight/price at POS.
  + Validation on format pattern must match defined standard.

**2.2 Categories**

* **Fields & Options:**
  + **Category Name** (String, Required)
  + **Description** (Text, Optional)
  + **Attachments** (Optional, images/docs for category illustration)
  + **Parent Category** (Dropdown, allows multi-level hierarchy).
* **Business Rules:**
  + Supports unlimited parent-child nesting.
  + Deleting a category with child items must prompt reassignment or block deletion.
  + Categories are mandatory for reporting & filtering in sales, purchases, and inventory.

**2.3 Unit Templates**

* **Fields & Options:**
  + **Base Unit Name** (String, Required, e.g., Gram)
  + **Short Name** (String, e.g., gm)
  + **Additional Units**:
    - **Unit Name** (e.g., Kilogram)
    - **Factor** (e.g., 1000 for converting grams → kg)
    - **Short Name** (kg).
  + **Template Name** (String, e.g., Weight)
  + **Status** (Active/Inactive).
* **Business Rules:**
  + Each template supports multiple conversion units.
  + Factor must be >0 and unique per unit within a template.
  + Changing a template affects all products linked to it.

**2.4 Custom Fields**

* **Fields & Options:**
  + Define custom attributes for products (text, number, date, boolean, dropdown).
* **Business Rules:**
  + Admin defines field type, label, and whether it’s required.
  + Custom fields appear on product creation/edit forms.
  + Search and reporting must support filtering by custom fields.

**2.5 Default Taxes**

* **Fields & Options:**
  + **Tax Name** (e.g., VAT, GST).
  + **Tax Value** (% or Fixed).
  + **Included/Exclusive** (Dropdown: Included in price or Added to price).
* **Business Rules:**
  + Multiple taxes can be defined and later grouped into tax profiles.
  + Taxes apply globally unless overridden at product/service level.
  + If set as default, tax is pre-applied to new products.

**2.6 Brands**

* **Fields & Options:**
  + **Brand Name** (String, Required).
* **Business Rules:**
  + Each product can be linked to a single brand.
  + Deleting a brand with assigned products should prompt reassignment.
  + Used for reporting, filtering, and analytics.

**3. Non-Functional Requirements**

* **Data Integrity:** Validation on unique names for categories, brands, units, and taxes.
* **Performance:** Tax calculation and barcode parsing must occur in <1 sec.
* **Security:** Only Admin/Manager roles can configure Product Settings.

**4. Integration**

* **POS/Invoice:** Barcode parsing, taxes applied.
* **Inventory:** Categories, units, and brands affect reporting.
* **Accounting:** Tax mappings for compliance with Egyptian e-invoicing.