# Dukapal System: Workflow and Database Schema Explanation

October 21, 2025

## 1 System Overview

Dukapal is a retail management system inspired by Shopify (products/inventory), QuickBooks (finance), Zoho/Ice CRM (CRM), and Oracle HR (HR). It manages products, procurement, inventory, point-of-sale (POS), finance, shipping, CRM, and HR, assuming an existing authentication module. The system uses a relational database (e.g., PostgreSQL) with tables linked via foreign keys to ensure data integrity. The workflow follows: Procurement → Inventory → Products/POS → Shipping → Finance, with CRM and HR integrations.

## 2 Modules and Workflow

The system is modular, with data flowing sequentially and bidirectionally where needed (e.g., sales update inventory and finance). Key scenarios include cash/loan purchases, sales, refunds, damages, and multi-currency transactions.

- **Procurement**: Create budgets, issue purchase orders (POs) to suppliers (cash or loan), receive goods, and record bills. Example: A PO for phones updates inventory and posts a bill to finance.
- **Inventory**: Track stock levels and adjustments (e.g., damages, theft). Example: Damaged goods trigger inventory deduction and financial loss entry.
- **Products & POS**: Manage product catalog and process sales. Sales deduct inventory and generate invoices/journals. Example: A POS sale of a phone posts revenue and cost of goods sold (COGS).
- **Shipping**: Handle deliveries, track shipments, and manage returns. Example: Damaged delivery triggers refund and inventory adjustment.
- **Finance & Accounting**: Record transactions via double-entry journals, generate reports (e.g., P&L, balance sheet). Example: Loan repayment updates liability accounts.
- **CRM**: Manage leads, contacts, and interactions, linking to sales. Example: A lead converts to a customer, triggering a POS sale.
- **HR**: Manage employees, payroll, and commissions. Example: A sales staff's commission is added to payroll after a sale.

### 2.1 Key Workflow Scenarios

1. **Cash Purchase & Sale**: Budget → PO → Pay cash → Receive goods → Update inventory → POS sale → Deduct stock → Post journals (Debit Cash, Credit Sales; Debit COGS, Credit Inventory) → Generate reports.
2. **Loan Purchase**: PO with loan → Record loan → Receive goods → Post journals (Debit Inventory, Credit Loans Payable) → Repay or default → Update ledgers.
3. **Damaged Goods**: Shipment marked damaged → Adjust inventory (deduction) → Issue refund → Reverse journals → Log CRM complaint.
4. **Theft & Multi-Currency**: Record theft (Debit Loss, Credit Inventory) or convert USD sale to KES for journals.
5. **CRM/HR Integration**: Lead converts to sale → Assign employee → Pay commission via payroll → Update finance.

## 3 Database Schema

The schema includes 22 tables across modules, with 5–8 fields each, using standard data types (e.g., INT, VARCHAR, DECIMAL) and constraints (e.g., NOT NULL, UNIQUE). All tables include implied fields: `id` (primary key, auto-increment), `created_at`, `updated_at`. Key tables are summarized below (full schema in separate DBML file).

### 3.1 Products Module

| Table | Description | Fields |
|---|---|---|
| products | Product details | title (VARCHAR NOT NULL), cost_price (DECIMAL), selling_price (DECIMAL), sku (VARCHAR UNIQUE), category_id (INT FK) |
| product_variants | Variants (e.g., size) | product_id (INT FK), option_name (VARCHAR), option_value (VARCHAR), price_adjustment (DECIMAL), sku (VARCHAR UNIQUE) |
| product_categories | Categories | name (VARCHAR NOT NULL), parent_id (INT FK) |

## 3.2 Procurement Module

| Table | Description | Fields |
|---|---|---|
| suppliers | Vendors | name (VARCHAR NOT NULL), email (VARCHAR), phone (VARCHAR), credit_terms (INT) |
| purchase_orders | POs | supplier_id (INT FK), order_date (DATE NOT NULL), total_amount (DECIMAL), status (ENUM: draft, sent, received), payment_method (ENUM: cash, loan, credit) |
| po_items | PO items | po_id (INT FK), product_id (INT FK), variant_id (INT FK), quantity (INT), unit_price (DECIMAL) |

## 3.3 Inventory Module

| Table | Description | Fields |
|---|---|---|
| inventory_locations | Storage sites | name (VARCHAR NOT NULL), type (ENUM: warehouse, store) |
| inventory_levels | Stock levels | product_id (INT FK), variant_id (INT FK), location_id (INT FK), quantity (INT NOT NULL) |
| inventory_adjustments | Stock changes | product_id (INT FK), variant_id (INT FK), location_id (INT FK), quantity (INT), adjustment_type (ENUM: addition, deduction, damage) |

## 3.4 POS Module

| Table | Description | Fields |
|---|---|---|
| sales_orders | Sales | customer_id (INT FK), sale_date (DATE NOT NULL), total_amount (DECIMAL), payment_method (ENUM: cash, card, mpesa), status (ENUM: completed, refunded) |
| sales_items | Sale items | sales_order_id (INT FK), product_id (INT FK), variant_id (INT FK), quantity (INT), unit_price (DECIMAL) |
| payments | Payments | sales_order_id (INT FK), amount (DECIMAL), method (VARCHAR), transaction_id (VARCHAR) |

## 3.5 Finance & Accounting Module

| Table | Description | Fields |
|---|---|---|
| accounts | Chart of accounts | name (VARCHAR NOT NULL), type (ENUM: asset, liability, equity, income, expense), balance (DECIMAL) |
| journal_entries | Transactions | date (DATE NOT NULL), description (TEXT), reference_id (VARCHAR), reference_type (ENUM: po, sale, bill) |
| journal_entry_lines | Debits/credits | journal_entry_id (INT FK), account_id (INT FK), debit (DECIMAL), credit (DECIMAL) |
| invoices | Customer bills | customer_id (INT FK), invoice_date (DATE), total (DECIMAL), status (ENUM: sent, paid) |
| bills | Supplier bills | supplier_id (INT FK), bill_date (DATE), total (DECIMAL), status (ENUM: received, paid) |
| budgets | Budgets | name (VARCHAR), total_amount (DECIMAL), actual_spent (DECIMAL) |
| loans | Loans | lender_name (VARCHAR), amount (DECIMAL), interest_rate (DECIMAL), status (ENUM: active, repaid) |

## 3.6 Shipping Module

| Table | Description | Fields |
|---|---|---|
| carriers | Shipping providers | name (VARCHAR NOT NULL), contact (VARCHAR), rates (JSON) |
| shipments | Deliveries | sales_order_id (INT FK), carrier_id (INT FK), tracking_number (VARCHAR), status (ENUM: shipped, delivered) |

### 3.7 CRM Module

| Table | Description | Fields |
|---|---|---|
| crm_leads | Potential customers | name (VARCHAR), email (VARCHAR), status (ENUM: new, qualified, lost), assigned_to (INT FK) |
| crm_contacts | Customers | lead_id (INT FK), name (VARCHAR NOT NULL), email (VARCHAR), phone (VARCHAR) |
| crm_interactions | Interactions | contact_id (INT FK), type (ENUM: call, email), date (DATE), notes (TEXT) |

### 3.8 HR Module

| Table | Description | Fields |
|---|---|---|
| employees | Staff | first_name (VARCHAR), last_name (VARCHAR NOT NULL), email (VARCHAR UNIQUE), job_title (VARCHAR), salary (DECIMAL) |
| departments | Units | name (VARCHAR NOT NULL), manager_id (INT FK) |
| payroll | Payroll | employee_id (INT FK), pay_date (DATE), net_amount (DECIMAL) |

## 4 Relationships and Implementation

Tables are linked via foreign keys (e.g., `sales_orders.customer_id` to `crm_contacts.id`). Key relationships:

- `products.category_id` → `product_categories.id`
- `purchase_orders.supplier_id` → `suppliers.id`
- `sales_items.sales_order_id` → `sales_orders.id`
- `journal_entry_lines.account_id` → `accounts.id`

Implement using a relational database with indexes on foreign keys for performance. Use triggers for automatic updates (e.g., deduct `inventory_levels.quantity` on sale). Generate reports via views (e.g., balance sheet from `accounts` sums).

# 5 Conclusion

Dukapal integrates retail operations efficiently, supporting cash/loan purchases, sales, refunds, damages, and multi-currency transactions. The schema is scalable, with concise fields and clear relationships, inspired by industry tools like Shopify and QuickBooks.