WAREHOUSE

Project Description with Modules Flow

A Warehouse is storage location of Items which are purchased from Mfg. Companies and sold to End user via E-Commerce Application. It has Roles Admin and AppUser. Admin Controls all aps users. Like Activate/Deactivate App user, Lock their Account etc.

On Registration of App User his details will be stored in DB also official email will be sent to him with welcome letter and App Credentials. Admin can also assign extra role "ACTURATOR" to access all Spring Boot actuator rest end points. Here we can see all the status like env, beans, profiles, log files.

Our warehouse applications has multiple operations those are divided into 3 levels.

1. Basic(Base) Operations

2. Inbound Operations

3. Outbound Operations

1. Basic Operations are: These are also called master data of warehouse. These are involved in both inbound and outbound operations.

Few example Basic operations are: UOM(Unit of Measure), Location, Item, Vendor and Customer, Shipment Type, Order Method etc...

These records can be created from UI , Excel and on the fly using RestWebServices. Even this data can be exported from Warehouse Database to Excel format.

2. Inbound Operations: To get an Item into warehouse multiple operations will be involved. These all are called as inbound operations. Inbound operations must be performed in given order only. PO->VI->GRN

Purchase Order (PO), Vendor Invoice (VI), Goods Receive Note (GRN).

3. Outbound operations: warehouse items can be sold to end users or wh customers. To sale an item, application should follow outbound operations. These outbound operations also must follow given order: SO->CI->SHIP

Sale Order (SO), Customer Invoice (CI), Shipment (SHIP).

UOM(UNIT OF MEASUREMENT):

To indentify one Item, it should be measured using one UOM Possible UOMs are: peice, box, pallet, rack. Every UOM will be one of Type PACKED ITEM, NON PACKED ITEM, PACKING Not APPLICABLE,

Ex:

TV = PACKED ITEM

COMPUTER ITEMS (to be assembled) = NA

Plastic Boxes and Raw Items = NON PACKED ITEMS

Here measurements are taken for every UOM in multiple levels. Example

BOX = SMALL,MEDIUM,LARGE,EXTRA LARGE are indicated using
BOXA,BOXB,BOXC,BOXD

in some ways PALLETA,PALLETB,PALLETC and PALLETD.

Racking process is given to LEVEL1,2,3,4 so they are called as RACKA =(1), RACKB=(2),RACKC=(3),RACKD=(4).

** Pallet is a wooden holder which contains Items on top of it. ** Pallets(Items) are moved using fork lifters.

SHIPMENT TYPE : It is used in outbound operation(Sending an Item from warehouse to end customer/E-Commerce Application).

Possible Shipping Types are Air (Cargo Flight), Ship (Ship Containers), Train (Goods Train) Truck (Tempo, DCM...). Every Shipment Type is identified using one unique code based on contractor ex: SG-558, VV-586 (Ship)...

Shipment must be enabled (YES must be checked) for usage. If contract is broken then Shipment will be disabled (not deleted). Every Shipment Type must be graded based on Security/Insurance/Faster Delivery ex: Grade A = Secured and Faster, Grade B = Faster
Grade C = Secured , Grade D = Normal

WAREHOUSE USER TYPE: Here application supports two types of Warehouse Users i warehouse concept.

1) Vendor : One who gets an item into warehouse. Person details used in Inbound operation

2)Customer: One who gets an item out of warehouse. Person details used in outbound operation.

It is like complete information of a User (User Registration process Name, Email, Mobile, Address, Id Details, unique code, Type(Vendor/Customer). One Item will be connected to multiple vendors and customers (at least one vendor and one customer)

To perform Inbound operation , must choose one Vendor. Then only those vendor connected items will be used in order process. In same way for outboudn Operation must choose one customer, only that customer Items will be used to create Sale Order.

User For (input) is a auto fill input(use JQuery or Java Script) on click "vendor" type It should have value "Purchase" also for "Customer" type value is "Sale". In same way if ID type is "Other" then . If other (input) must be enabled else it should be read only input.

PURCHASE ORDER :

To place one order, end user should Provide Main Details and Items Details. Main Details are called as PurchseOrderHeader (PoHdr), it is given as Screen#1. It contains information like : Order Code, Shipment Mode, Vendor Code, Reference Number, Quality check, default status(OPEN).

If PO(Purchase Order) Main details is created, then user can select Items. One line is called as One PoDtl (PurchaseOrderDetail). Every PoDtl contains slno, ItemCode, BaseCost, Qty

Purchase Order Status Codes:

OPEN - It is default status, on creating POHdr it will be assigned.

PICKING - If at least one Item is added to Po using PoDtl Screen then status will be PICKING. If all Items are removed from PODtl then Status will be OPEN. In PoHdr, "vendor" can be editable if PoHdr status is "OPEN". Else read only

ORDERED: No more picking items to PO. It says Order is completed.

INVOICED : After Generating Bill for Vendor as a PDF then PO Status should be INVOICED. Only ORDERED Status PO can be used in Vendor Invoice process

RECEIVED: If GRN is done for PO having status INVOICED then it will be changed to RECEIVED.

VENDOR INVOICE : It is a final bill generated for Vendor for purchsed Item using Warehouse PO. Base cost it considered from Item, Quantity will be taken from PO child Screen. Final cost of PO = Sum of Final Values of all Line Items are taken. Final Currency conversion will be done based conversion factors exist in application.

At last one PDF will be generated which holds details like Vendor Invoice code, vendor code, Shipment Details, Final Cost for all Items. Purchase Order code. For PO if Invoice is generated then PO will be ready for GRN (next level)

GRN (GOODS RECEIVE NOTE): On performing this operation Stock will be added to Wh Application. Here we can do quality check for item. If valid Items are accepted else we can reject Item. Accepted Items quantity will be increased. At last QOH(Quanityt On Hand) table will be updated with all vendors, Items , PO and Invoice details.

** Here we provide fast accept process for GRN by clicking accept all or return all buttons.

**It also contains main details and child details. In Main details we should enter GNR ID,CODE, Shipment Details, Vendor Details. Selected Vendor PO will having status Invoiced will be displayed in Dropdown. On Performing GRN , PO status will be changed to Received.

SALE ORDER : Here we can selected items which are bonded to One customer. Customer will provide all his item details, price retrieved from Item, he can also apply discounts. Final bill will be calculated after discounts and all. It is like baseAmt-discountMoney * itemsCount

Only items having QOH(quantity >0) will be displayed in dropdown. Those are only involved in Sale Order. Else that will be converted to back order. Back Order can again converted to Re-Sale Order when quantity is available.

On creating Sale order it's status will be SALE-READY. Once if it is used in Customer invoice then status will be changed to SALE-READY-INVOICED. Only INVOICED status Sales are used in Shipments.

CUSTOMER INVOICE : It is a final bill generated for Customer for selected sale-Order. It will give information like Item details, base cost, UOM used , discount value, final cost, line level cost etc. Once this Bill is generated then status will be changed to SALE-INVOICED.

SHIPPING : Here we can confirm the shipment process and also update the status info. We can do full shipping ie sending all items as single shipment or partial shipping ie sending few items in shipmenent. At last shipment is confirmed then process will be started.

Once shipment is done fully then Sale order status will be SALE-SHIPPED.

WH BILLING : We can select one vendor or customer to find all their PO or SO's Bill. It is also called as WHBilling.

In our application we applied concepts like Spring Boot Security for Login process, Swaggers For Rest Operations, Spring Data JPA for DB operations, Spring Boot AOP for adding Log4J using Stopwatch (here performance analysis is done). Spring boot Embedded Servers and DB's which works in any environments. Also supports Spring boot cloud deployment.

Spring Boot Actuator also added to see all rest end points given by Spring boot actuator, to access all these AppUser must have Role "ACTURATOR".