Software Design Document

for

HYBRID APP WAREHOUSE MANAGEMENT

Version 1.0 approved

Prepared by : Chittoori Meha

Chaitanya Reddy

Jallepalli Prerna

Challamala Krishna priya

Madarapu Srikar

NIIT Univeristy

Revision History

Date	Version	Description	Author
04/18/07	<1.0>	Initial Version of Document	Team

1 Introduction

This software design documents helps the software developers to develop the product in a sequential way. This document gives an overview about developing of a hybrid application (in our perspective).software design document also includes narrative and graphical documentation of software design for the product including Use case models, sequence diagrams, Collaboration models, Objects behavior models, and supporting required information.

1.1 Purpose

The main purpose of the software design document is to provide a brief description of the design of a system so that to allow for software development to proceed with an idea of what is to be built next and how it is exactly supposed (expected) to be built.

1.2 Scope

Since our product is completely for corporate, we plan according to their requirements. Our product focuses on warehouse management part where in the software products are stored and updated information is sent across. This application is made up of relational database as a hybrid application which make the application work in any platform without creating a hiccup for the users. This product helps in managing all the products that are existing in any warehouse like, sales reports, purchase reports and others

The scope of the mobile application is as follows:

1. To build ERP Mobile Application for the warehouse management.
2. The mobile application needs to be developed using Hybrid Mobile Development Approach.
3. The following features are to be implemented as part of the mobile application. The features are to be thoroughly tested to ensure functionality works without any deviations
   i) **Purchase Order Creation from Mobile Application.**
   ii) **View List of Purchase Orders**
   iii) **Delete and Update the Purchase Orders**
   iv) Sales Order Reports
   v) Warehouse Management Reports

4. **Develop Graphical reports**
5. Use Responsive UI
6. The application needs to support 150 concurrent users and the design needs to be implemented to enhance the concurrent user's usage up to 500 users.
7. The application needs to provide the security features as defined by the warehouse management team policies

1.3 Definitions, Acronyms, and abbreviation

Acronyms/Abbreviations	Full form
SDD	Software Design document
FS	Functional Specification
DB	Database
WH	Warehouse
OS	Operating system
Android	Mobile phone OS from google
iOS	Mobile phone OS from apple
PO	Purchase Order
UI	User Interface
SQL	Structured Query Language
SE	Software Engineering

1.4 References:

The following are some of the references leveraged to understand the warehouse domain, mobile application development, software engineering concepts, design document. Based on some of these reference sources, the good practices are proposed to develop the mobile application which can be platform independent, scalable and secure.

Weblinks:
http://dlca.logcluster.org/display/LOG/Warehousing+and+Inventory+Management
http://community.mis.temple.edu/mis3504digitaldesignsections12/files/2013/09/WarehouseStakeholder-Case.pdf
https://en.wikipedia.org/wiki/Warehouse_management_system
http://searchmanufacturingerp.techtarget.com/definition/warehouse-management-system-WMS
http://www.infoworld.com/article/2615122/mobile-development/native--web--or-

hybrid--how-tochoose-your-mobile-development-path.html
http://www.3pillarglobal.com/insights/when-to-take-a-hybrid-approach-for-mobile-appdevelopment

https://en.wikipedia.org/wiki/Software_design_description

http://www-inst.eecs.berkeley.edu/~cs162/sp12/design.html

http://blog.slickedit.com/2007/05/how-to-write-an-effective-design-document/

https://www.toptal.com/freelance/why-design-documents-matter

http://csciwww.etsu.edu/nielsen/4417/DesignDocument.htm
Books: Software Engineering by Roger S Pressman 5th edition, Software Engineering by Pankaj Jalote 4th edition.

1.5 Overview

The Software Design Document is divided into 11 sections with various subsections. The sections of the Software Design Document are:

1 Introduction

1.1 Purpose

1.2 Scope

1.3 Definitions, Acronyms, Abbreviations.

1.4 References

1.5.1 Use case

1.5.2 Use cases descriptions

1.5.3 Dynamic Model

2. System Architecture

3. Detailed Description of components

4.Reuse and relationship to product

5. Design Decision and tradeoff

6. Pseudocode

1.5.1 Use case:

1.5.1.1 Actors:

1.5.1.1.1 Document manager:

This is an abstraction of specific users who perform same actions but for different reason. The specific actors who fall in this category are

- 1.5.3.1.1.1.1 field engineer
- 1.5.3.1.1.1.2 manager

Manager includes the user who can login into the application and can view the purchase order report, Sales order reports and warehouse details and can update and delete the purchase order and sales order. He can also view graphical representation of above mentioned reports. He can also view the warehouse details, Shipping details, return shipping, and so on which are mentioned in the figure**.** Field engineers manage the warehouse reports over it.

1.5.1.1.1.2 Additional Information:

The document user is the user who is seen in the use cases those are considered essential to the system under design. Of the essential use cases, view purchase order graphical reports, sales order graphical reports, warehouse management details have the highest priority . Following diagrams in section 3.3 contain current and future implemented use cases for illustrative purposes of future directions of the system under design.

1.5.1.1.2 System under design:

1.5.1.1.2.1

The system under design is the hybrid application for warehouse management system that is being created. The actors here represents the system and the action that place in the system.

1.5.1.1.3 Administrative User

1.5.1.1.3.1 Information:

The administrative user is who administers system by overseeing accounts creation and administration.

1.5.1.1.4: Public user

1.5.1.1.4.1 Information:

The Public user is the user who login and access the application to create a purchase order and view the purchase order status and purchase order with different payment .

1.5.2.2 List of Use Cases

1. Login

2. Purchase order reports

3. Purchase order status

4. Purchase order with different payments

5. Purchase order creation

6. Update purchase order

7. Delete purchase order

8. Sales order reports

9. Sales order status

10. Sales order with different payment

11. Warehouse management

12. Warehouse with sales order

13. Warehouse with purchase order

14. Return shipping

15. Warehouse by date

16. Inventory item

17. Delete sales order

18. Update sales order

.

1.5.2.2.1 Manager User Use Cases

1.5.2.2.1.1 Purchase order report (detail)

1.5.2.2.1.2. Update purchase order (overview)

1.5.2.2.1.3 Delete purchase order (overview)

1.5.2.2.1.4 Sales order report (Detail)

1.5.2.2.1.5 Delete sales order (overview)

1.5.2.2.1.6 Update sales order (overview)

1.5.2.2.1.7 Warehouse management (detail)

1.5.2.2.1.8 Warehouse with sales order

1.5.2.2.1.9 Warehouse with purchase order

1.5.2.2.1.10 Return shipping

1.5.2.2.1.11 inventory item

1.5.2.2.2 Customer Use case:

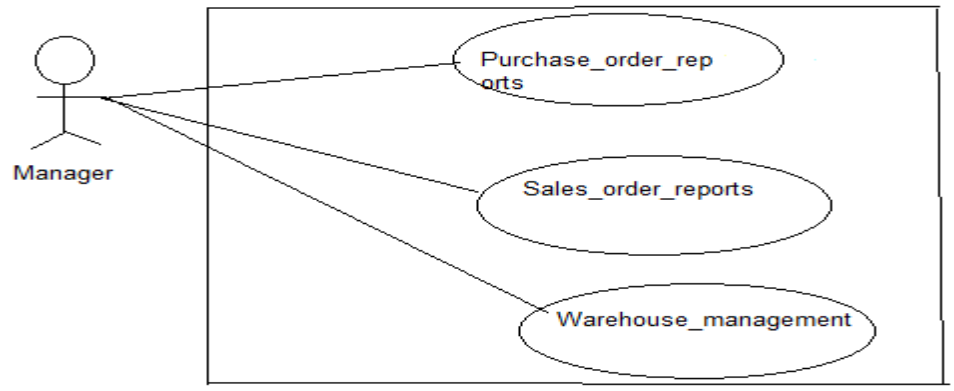
1.5.2.2.2.1 Purchase order creation (detail)

1.5.2.2.2.2 Purchase order status (overview)

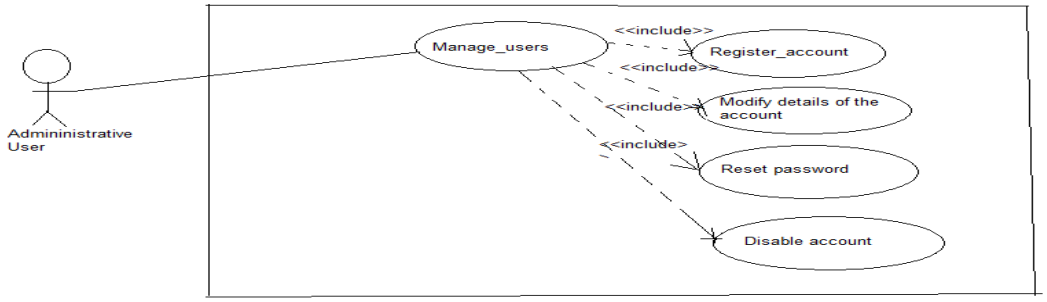
1.5.2.2.2.3 Purchase order with different payments (overview)

1.5.2.3 Use case Diagram

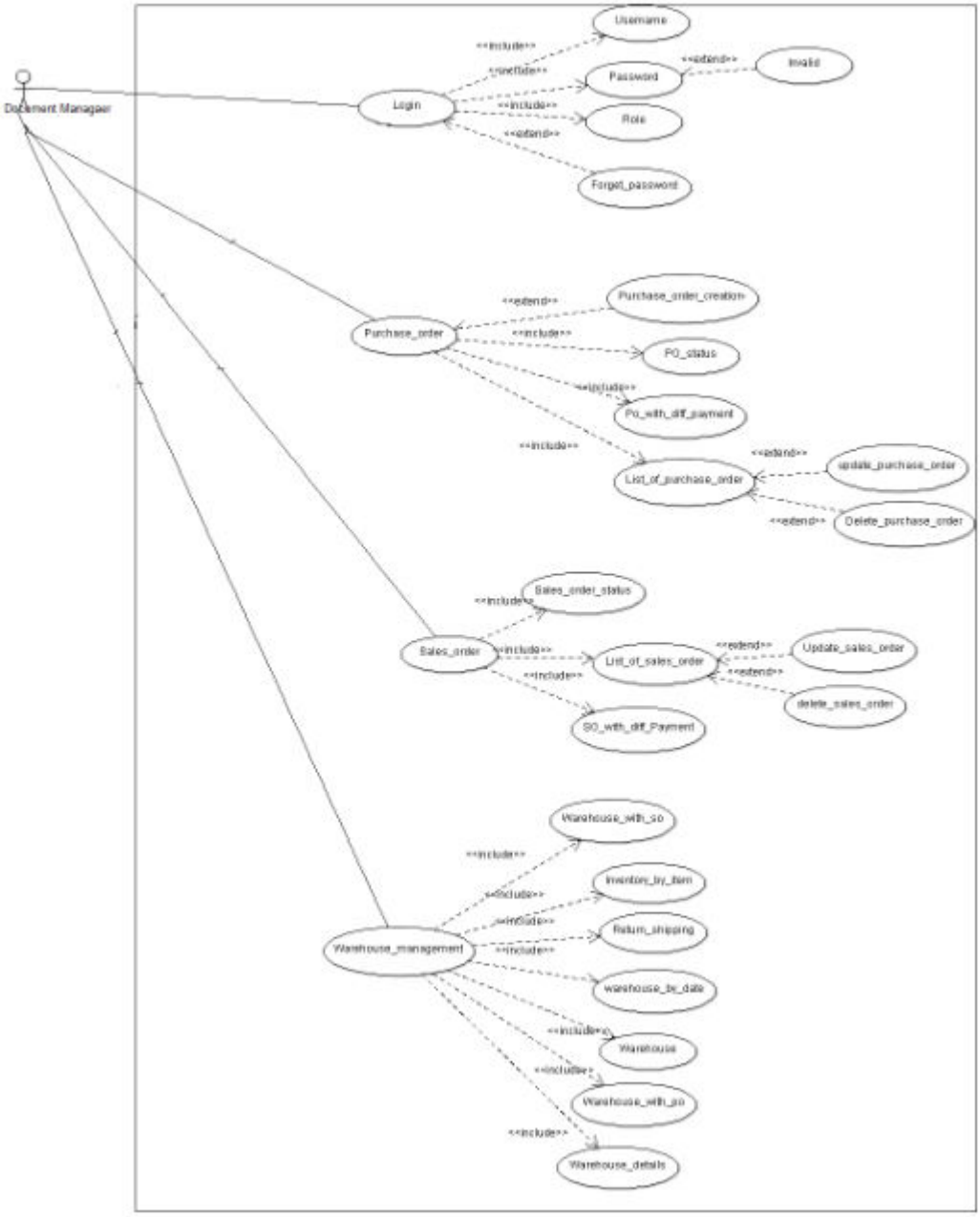
1.5.2.3.1 manager - Essential Use cases



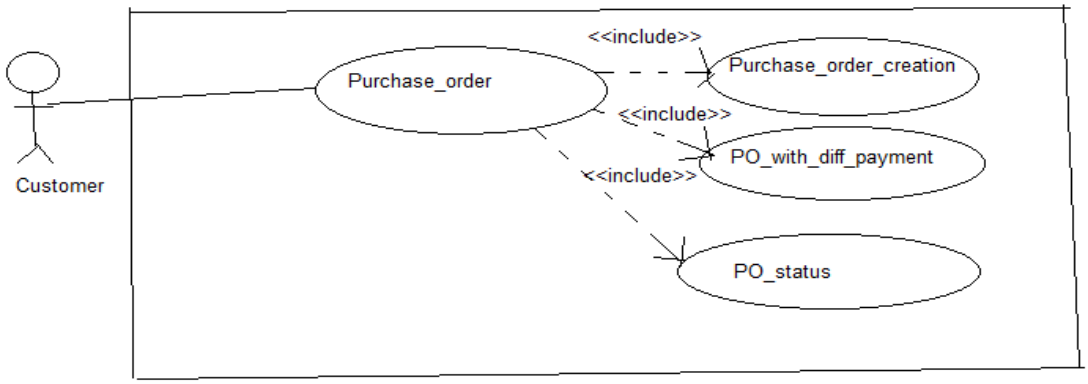
1.5.2.3.2 Administrative User



1.5.2.3.3 manager –Use cases


Document Manager
Login
Username
Password
Invalid
Role
Forget_password
<<include>>
<<extend>>
Purchase_order
Purchase_order_creation
PO_status
Po_with_diff_payment
List_of_purchase_order
update_purchase_order
Delete_purchase_order
Sales_order
Sales_order_status
List_of_sales_order
Update_sales_order
delete_sales_order
SO_with_diff_Payment
Warehouse_management
Warehouse_with_so
Inventory_by_item
Return_shipping
warehouse_by_date
Warehouse
Warehouse_with_po
Warehouse_details

1.5.2.3.4 Public User – Use Cases



1.5.3 USE CASE DESCRIPTION

1.5.3.1

Use case name: Purchase order Report(overview)	ID: PORO	Priority: High	
Primary Actor: Manager	Sources: Technical managers, Field engineer	Use case type: Business	Level: Overvie w
Interested stake holders: Manager, Field engineer			
Brief description: This use case describes the viewing of graphical representation of purchase order reports. In this use case actors goal is to view the graphical Representation of purchase order reports			
Goal: To view graphical reports of purchase order			
Success Measurement: The graphical reports of purchase order and sales order are generated and viewed by the manager			

Precondition: Manager must be able to successfully pass through authentication and authorization. Sufficient data (purchase and sales order reports) must be entered the system to show them in a graphical way
Trigger: Manager has reached a point in their workflow in which graphical reports are to generated and shown to him.
Typical Flow of events: 1. Manager login to the application 2. He selects a product for which he wants to view the purchase order reports 3. He selects which kind of graphical representation he wants to see (i.e. pie chart, bar graph…) 4. Graphical representation of purchase order report is shown Assumptions: 1. It is assumed that work flow will be carried out internally or with close partrened agencies that can be interact with n similar manner as with internal system. 2. it is assumed that this system hold an appropriate and valid data to construct graphical reports

1.5.3.2

User case name: Purchase order report (detail)	ID: PORD	Priority: High	
Primary Actor: Manager	Sources: Technical managers,	Use case type: Business	Level: Detail

	Field engineer		

Interested stake holders:
Technical Manager, Field engineer

Success Management:
The graphical reports of purchase order and sales order are generated and viewed by the manager

Precondition:
Manager must be able to successfully pass through authentication and authorization.
Sufficient data (purchase and sales order reports) must be entered into the system in
order to show them in a graphical way

Trigger:
Manager has reached a point in their workflow in which graphical reports are to generated and shown to him.

Typical flow of
events:

1. Manager needs to login the
application
2. Manager should select the product for which he wants to see the purchase order
reports
3. Data must be entered regularly as per purchase orders into the
database
4. When manager asks for a specific product purchase order report that data is retrieved from a database (here it is local storage)
5. when the date is retrieved it is then transferred to the representation form which manager wants
that is bar graph, pie chart
etc..
6. Graphical representation of the purchase order reports is shown to manager.

Assumptions:
It is assumed that work flow will be carried out internally or with close partened

agencies that can be interacted with n similar manner as with internal system It is assumed that this system holds an appropriate and valid data to generate the graphical reports

1.5.4.3

Use case name: Purchase order Report(overview)	ID: UPOO	Priority: Medium	
Primary Actor: Manager	Sources: Technical managers, Field engineer	Use case type: Business	Level: Overview
Interested stake holders: Manager, Field engineer			
Goal: successful updation of purchase order			
Success Measurement: Purchase orders are successfully updated			
Precondition: Manager must be able to successfully pass through authentication and authorization. valid data of purchase orders must be present to update the data.			
Trigger: Manager has reached a point in their workflow in which graphical reports are to generated and shown to him.			
Typical flow of events: 1. Manager needs to login the application			

2.He needs to select a product 3. after selecting a product he needs to go to the purchase order reports 4. He needs to select update option 5. updates

1.5.4.4

Use case name: Delete purchase order (overview)	ID: DPOO	Priority: Medium	
Primary Actor: Manager	Sources: Technical managers, Field engineer	Use case type: Business	Level: Overvie w
Interested stake holders: Technical Manager, Field engineer			
Goal: Deletion of purchase order			
Success Measurement: Purchase orders are successfully deleted			
Precondition: Manager must be able to successfully pass through authentication and authorization. valid data of purchase orders must be present to delete the data.			
Trigger: Manager has reached a point in their workflow in which purchase order reports are deleted			

Typical flow of events: 1. Manager needs to login the application 2.He needs to select a product 3. after selecting a product he needs to go to the purchase order reports 4. He needs to select delete option 5. deletes the purchase orders Assumptions: 1. It is assumed that work flow will be carried out internally or with close partened agencies that can be interacted with n similar manner as with internal system 2. It is assumed that this system holds an appropriate and valid data to delete the purchase orders.

1.5.4.5

Use case name: Sales order report (overview)	ID: SORO	Priority: High	
Primary Actor: Manager	Sources: Technical managers, Field engineer	Use case type: Business	Level: Overview
Interested stake holders: Technical Manager, Field engineer			
Brief description: This use case describes the viewing of graphical representation of sales order reports. In this use case actors goal is to view the graphical representation of sales order reports			

Goal: To view graphical reports of sales order
Success Measurement: The graphical reports of sales order are viewed by the manager
Precondition: Manager must be able to successfully pass through authentication and authorization. Sufficient data (sales order reports) must be entered the system to show them in a graphical way
Trigger: Manager has reached a point in their workflow in which graphical reports are to generated and shown to him.
Typical flow of events: 1. Manager login to the application 1. He selects a product for which he wants to view the sales order reports 3. He selects which kind of graphical representation he wants to see (i.e. pie chart, bar graph…) 4. Graphical representation of sales order report is shown

1.5.4.6

Use case name: Sales order report (Detail)	ID: SORD	Priority: High	
Primary Actor: Manager	Sources: Technical managers, Field engineer	Use case type: Business	Level: Detail
Interested stake holders: Technical			

Manager, Field engineer
Success Measurement: The graphical reports of sales order viewed by the manager
Precondition: Manager must be able to successfully pass through authentication and authorization. Sufficient data (sales order reports) must be entered the system to show them in a graphical way
Trigger: Manager has reached a point in their workflow in which Graphical reports are shown to him.
Typical flow of events: 1. Manager needs to login the application 2. Manager should select the product for which wants to see the sales order reports 3. Data must be entered regularly as per sales orders into the database 4. When manager asks for a specific product sales order report that data is retrieved from a database (here it is local storage) 5. when the date is retrieved, it is then transferred to the representation form which manager wants that is bar graph, pie chart etc.. 6. Graphical representation of the sales order reports is shown to manager. Assumptions: 1. It is assumed that work flow will be carried out internally or with close partened agencies that can be interacted with n similar manner as with internal system 2. It is assumed that this system holds an appropriate and valid data to generate the graphical reports

1.5.4.7

<table>
<tr><td>Use case name:
Delete Sales Order
(Overview)</td><td>ID:
DSOO</td><td>Priority:
Medium</td><td></td></tr>
<tr><td>Primary Actor:
Manager</td><td>Sources:
Technical managers, Field engineer</td><td>Use case type:
Business</td><td>Level:
Overview</td></tr>
<tr><td colspan="4">Goal:
Deletion of sales order</td></tr>
<tr><td colspan="4">Interested stake holders:
Technical Manager, Field engineer</td></tr>
<tr><td colspan="4">Success Measurement:
The graphical reports of sales order viewed by the manager</td></tr>
<tr><td colspan="4">Precondition:
Manager must be able to successfully pass through authentication and authorization.
valid data of sales orders must be present to delete the data.</td></tr>
<tr><td colspan="4">Trigger:
Manager has reached a point in their workflow in which Sales Order reports are deleted.</td></tr>
<tr><td colspan="4">Typical flow of events:
1. Manager needs to login the application
2. He needs to select a product
3. After selecting a product, he needs to go to the sales order reports
4. He needs to select delete option when the date is retrieved it is then transferred to the representation form which manager wants</td></tr>
</table>

that
is bar graph, pie chart etc.

5. deletes the sales orders.

Assumptions:

1. It is assumed that work flow will be carried out internally or with close partened agencies that can be interacted with n similar manner as with internal system
2. It is assumed that this system holds an appropriate and valid data to generate the graphical reports

1.5.4.8

Use case name: Update sales order (overview)	ID: USOO	Priority: Medium	
Primary Actor: Manager	Sources: Technical managers, Field engineer	Use case type: Business	Level: Overview
Interested stake holders: Technical Manager, Field engineer			
Goal: Successful updation of sales order			
Success Measurement: sales orders are successfully updated			
Precondition: Manager must be able to successfully pass through authentication and authorization. valid data of sales orders must be present to delete the data.			
Trigger:			

Manager has reached a point in their workflow in which sales order reports are updated.
Typical flow of events: 1. Manager needs to login the application 2. He needs to select a product 3. After selecting a product, he needs to go to the sales order reports 4. He needs to select Update Option. 5. updates the sales orders. Assumptions: 1. It is assumed that work flow will be carried out internally or with close partened agencies that can be interacted with n similar manner as with internal system 2. It is assumed that this system holds an appropriate and valid data to update the sales orders.

1.5.4.9

Use case name: Warehouse Management (overview)	ID: WMO	Priority: High	
Primary Actor: Manager	Sources: Technical managers, Field engineer	Use case type: Business	Level: Overview
Interested stake holders: Technical Manager, Field engineer			
Brief Description: This use case describes about the management of the warehouse which has warehouse details, return shipping details,			
Success Measurement: The graphical reports of sales order viewed by the manager			

Precondition: Manager must be able to successfully pass through authentication and authorization. valid data of sales orders must be present to delete the data.
Trigger: Manager has reached a point in their workflow in which Sales Order reports are deleted.
Typical flow of events: 1. Manager needs to login the application 2. He needs to select a product 3. After selecting a product, he needs to go to the sales order reports 4. He selects warehouse management 5. All necessary details are shown to the manager Assumptions: 1. It is assumed that work flow will be carried out internally or with close partened agencies that can be interacted with n similar manner as with internal system 2. It is assumed that this system holds an appropriate and valid data to show warehouse details to the user

1.5.4.10

Use case name: Warehouse management (detail)	ID: WMD	Priority: High	
Primary Actor: Manager	Sources: Technical managers, Field engineer	Use case type: Business	Level: Detail
Interested stake holders: Technical Manager, Field engineer			

Success Measurement: Necessary details of warehouse are viewed by manager
Precondition: Manager must be able to successfully pass through authentication and authorization. Sufficient data must be entered the system to show necessary warehouse details to the manager
Trigger: Manager has reached a point in their workflow in which necessary warehouse details are shown to the manager.
Typical flow of events: 1. Manager needs to login the application 2. Manager should select the product for which he wants to see warehouse details (so goes into warehouse management) 3. Data must be entered regularly as per warehouse management and details into the database 4. When manager asks for a specific warehouse detail (return shipping, inventory….) that data is retrieved from a database (here it is local storage) 5. Necessary details are shown to the manager Assumptions: 6. It is assumed that work flow will be carried out internally or with close partened agencies that can be interacted with n similar manner as with internal system 7. It is assumed that this system holds an appropriate and valid data to generate the graphical reports

1.5.4.11

Use case name: Purchase order creation(overview)	ID: POCO	Priority: High	
Primary Actor: Customer	Sources: Customer	Use case type: Business	Level: Detail
Interested stake holders: Customer			
Success Measurement: Success is measured by the successful creation of purchase order by the customer			
Precondition: Customer must be able to sucessfully login into the application			
Trigger: Customer has reached to a point in system in which he created the purchase order			
Typical Flow of events: 1. Customer login to the application. 2. He selects the product which he wants to purchase 3. He initiates purchase order creation 4. Order creation is done. Assumptions: It is assumed that work flow will be carried out internally or with close partened agencies that can be interacted with n similar manner as with internal system			

1.5.4.12

Use case name: Purchase order creation(detail)	ID: POCD	Priority: High	

Primary Actor: Customer	Sources: Customer	Use case type: Business	Level: Detail
Interested stake holders: Customer			
Success Measurement: Success is measured by the successful creation of purchase order by the customer			
Precondition: customer must be able to successfully login into the application			
Trigger: Customer has reached to a point in system in which he created the purchase order			
Typical flow of events: 1. Customer needs to login the application 2. He selects the product which he wants to purchase 3. He initiates purchase order creation 4. Purchase order that is created is stored in local storage for the manager to refer. 5. Order creation is done. Assumptions: It is asumed that work flow will be carried out internally or with close partened agencies that can be similar manner as with internal system.			

1.5.4.13

Use case name: Purchase order status(overview)	ID: POSO	Priority: Medium	
Primary Actor: Customer	Sources: Customer	Use case type: Business	Level: Overview
Interested stake holders: Customer			
Success Measurement: Success is measured by the view of status of the product to the customer			
Precondition: customer must be able to successfully login into the application			
Trigger: Customer has reached to a point in system in which he views the status of purchase order			
Typical Flow of events: 1. Customer login to the application. 2. He selects the product which he created purchase order 3. He then views the status of the purchase order Assumptions: It is asumed that work flow will be carried out internally or with close partened agencies that can be interacted with n similar manner as with internal system			

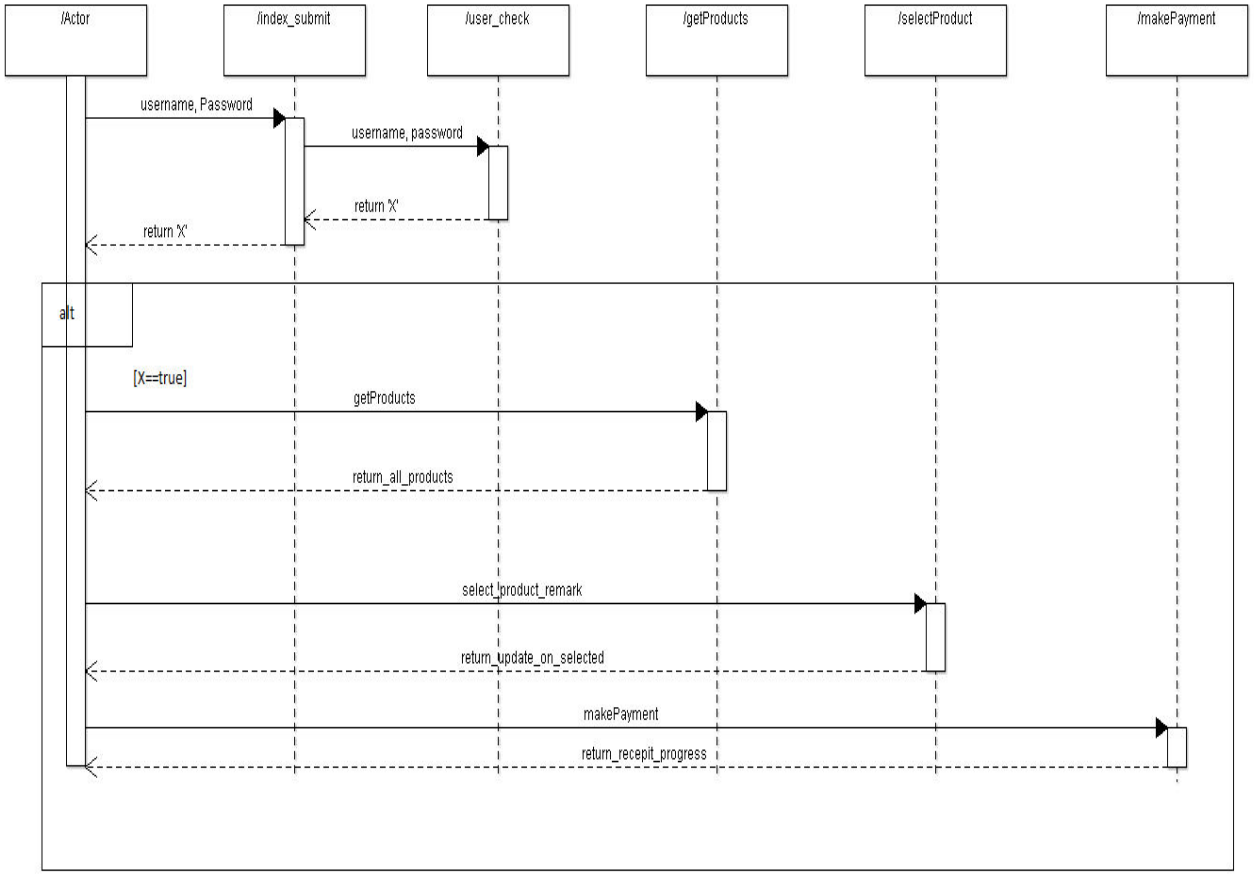
1.5.4.14

Use case name: Purchase order with different payments	ID: PODP	Priority: Medium	
Primary Actor: Customer	Sources: Customer	Use case type: Business	Level: Overview
Interested stake holders: Customer			
Success Measurement: Success is measured by the view of different options of payment for customer			
Precondition: customer must be able to successfully login into the application			
Trigger: customer has reached to a point in system in which he views different options of payment			
Typical Flow of events: 1. Customer login to the application. 2. He creates purchase order 3. He then goes to payment option and select the different payment option Assumptions: It is assumed that work flow will be carried out internally or with close partened agencies that can be interacted with n similar manner as with internal system			

1.5.3 Dynamic Model

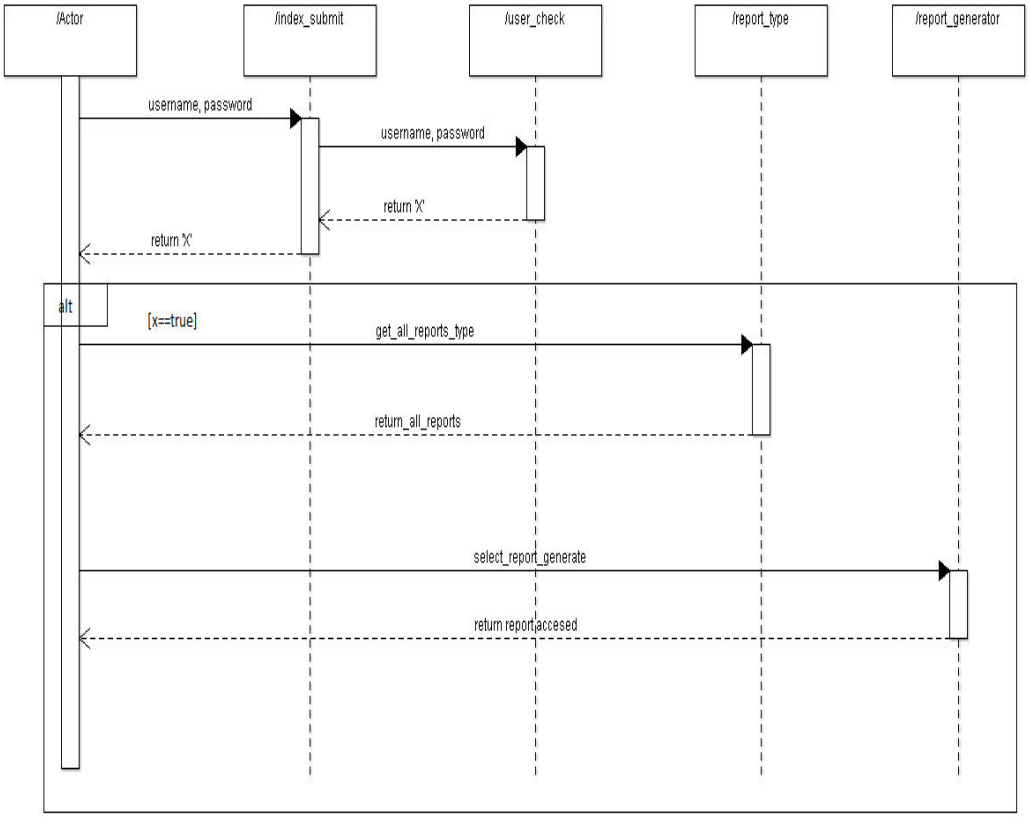
1.5.3.1 Sequence Diagrams

Purchase Order Creation Sequence diagram:



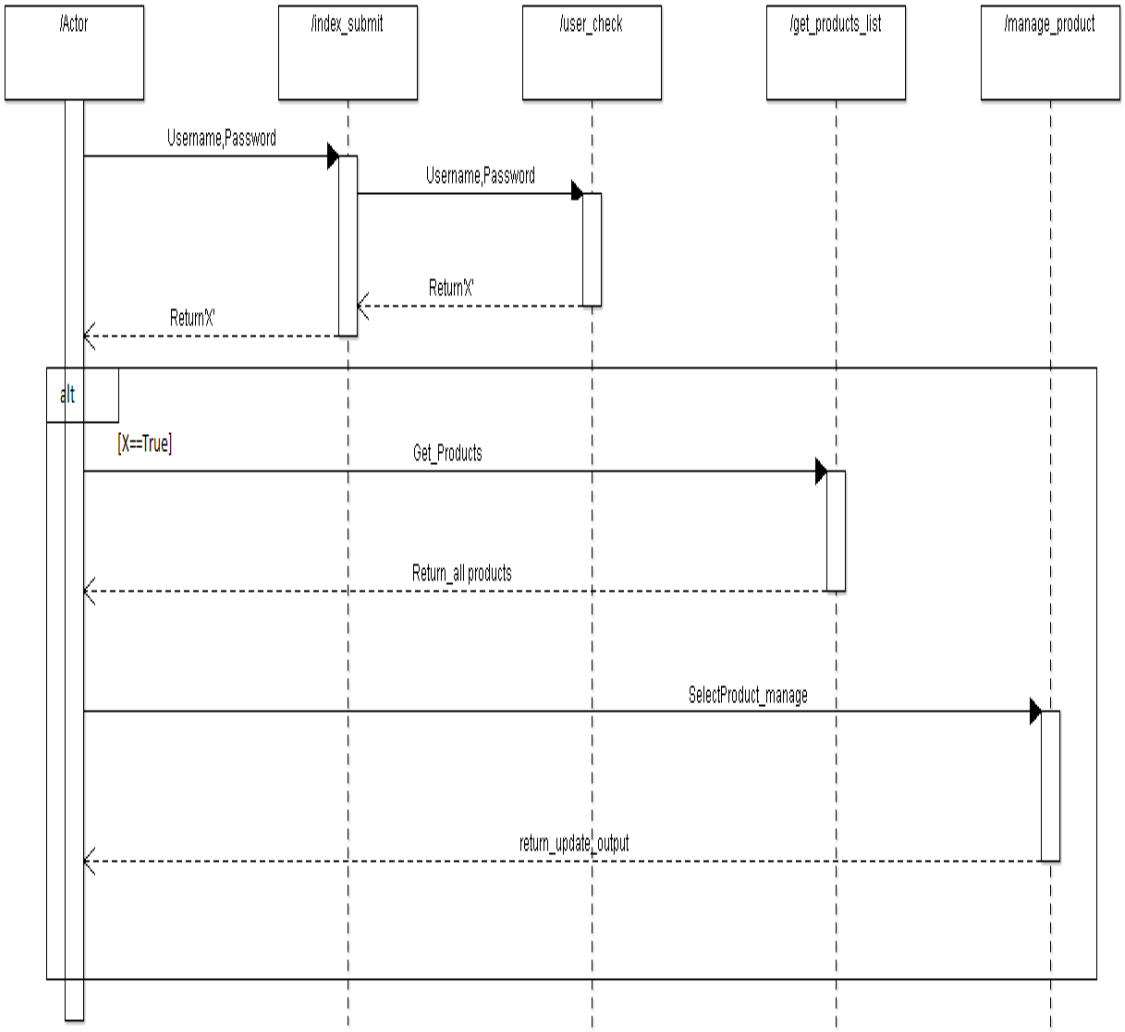
Above diagram shows the purchase order creation by the Customer(actor) in this context.

1.5.3.2 Sequence diagram for viewing report



Above diagram represents that Manager can view the reports of purchase order and sales order.

1.5.3.3 Sequence diagram for Selecting and managing the product



Above diagram describes that customer and manager can select the product among the product list.

2. System architecture Description

2.1 overview of Components:

1. Authentication

2. Product Management

3. Purchase Order

4. Purchase order report

5. Sales order

6. Sales Order report

7. Warehouse management

8. Warehouse report

9. Graphical report

1. **Authentication:** User has to login to the application using user name, password, role , and if these all things match then authentication is done by the user.

2. **Product Management:** After proper authentication, in this module User has to select the product among the product list provided for the further access.

3. **Purchase Order:** In this module customer is supposed to initiate the product creation and he can also view purchase order status.

4. **Purchase Order report:** In this module manager can view purchase order reports.

5. **Sales order:** In this module it provides details about sales order.

6. **Sales order report:** In this module manager can also view sales order reports when he wants to view them.

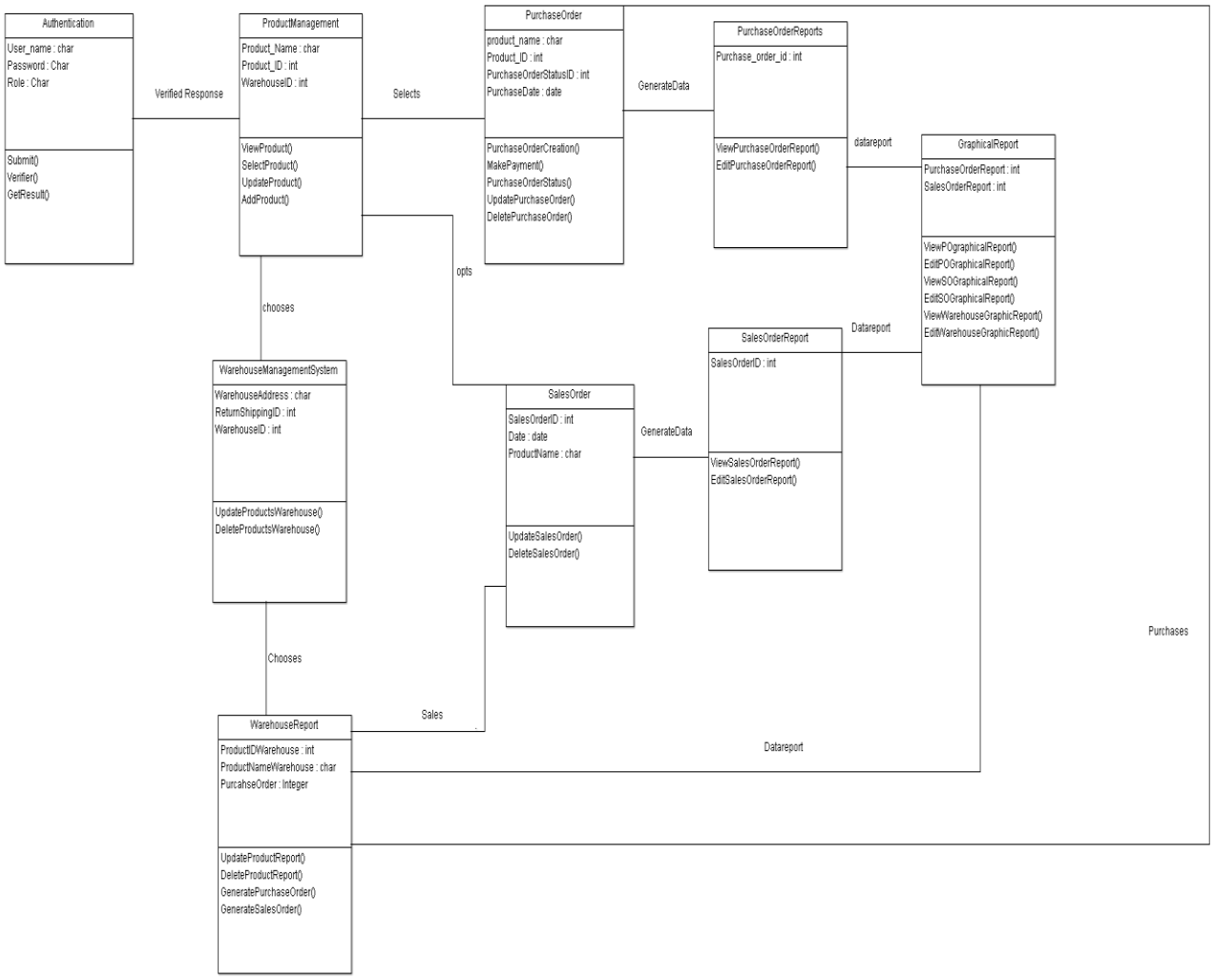
7. **Warehouse Management:** In this module, it contains details about warehouse address, and purchase and sales order in warehouse and return shipping details.

8. **Warehouse Report:** In this module it contains reports about details provided in above module warehouse management

9. **Graphical report:** In this module all the graphical reports of purchase order reports, sales order reports, and warehouse reports are showed in graphical form.

NOTE: As discussed earlier we are drawing component diagram instead of Class diagram.

2.2 Structure and relationships



2.3 User Interface Issues:

The user interface consists of a set of menu through which the user can interact with data on our application server. These sections include “login menu”, “product selection menu”, “report choosing menu” “product creation menu”, “graphical reports displaying menu”.

Description of User interface

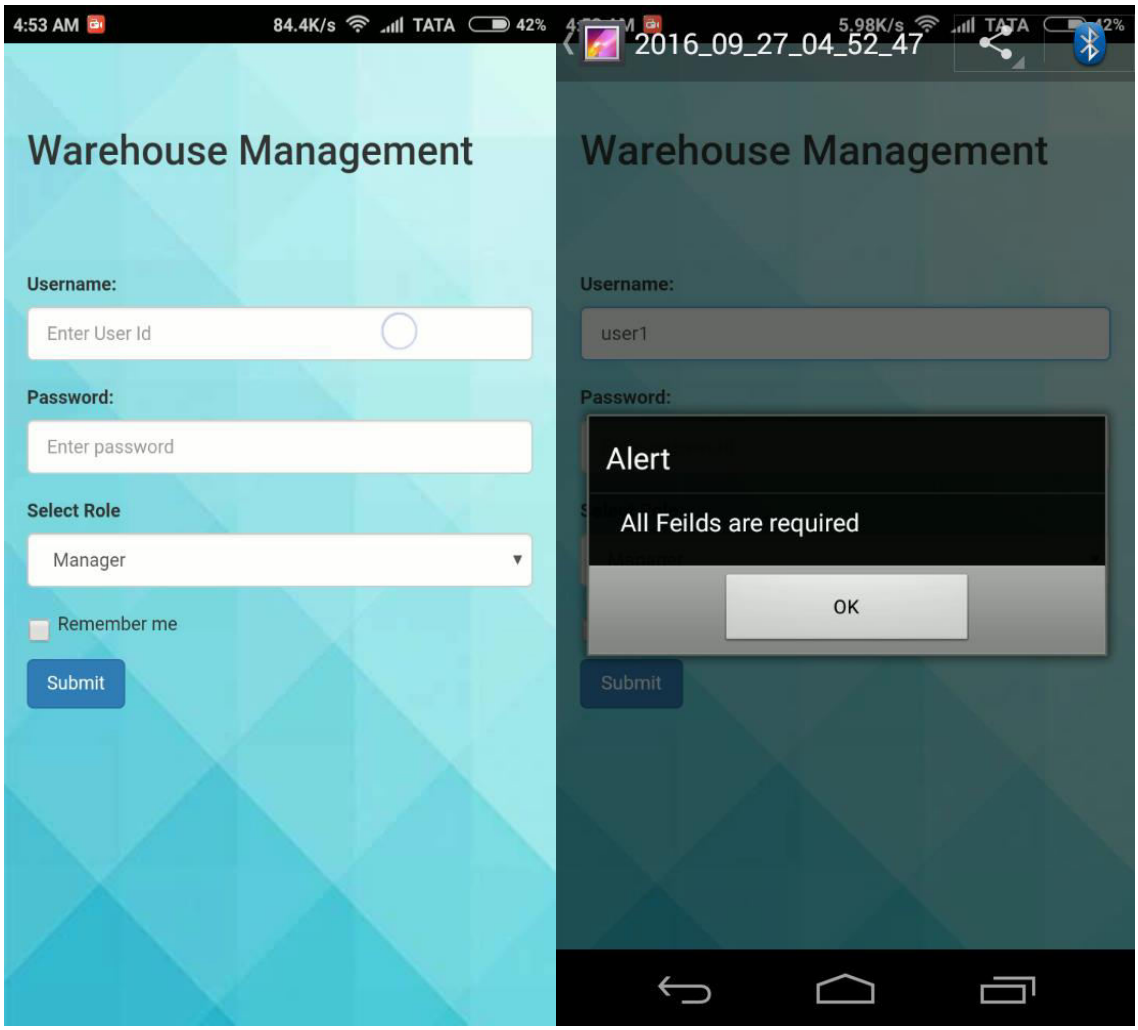
Each menu will consist of various GUI components, such as buttons, labels, text fields, and list objects. These components will be arranged in such a way that the user will be able to quickly grasp the purpose of each menu and perform whatever task it is designed for efficiently. A detailed description of these menus and their interactions with each other will be described in the below sections.

Objects and actions

The next several pages describe and illustrate the following application menus:

LOGIN

In this page it contains User name, password and role buttons. The user should fill the User name, password and the role. It is verified against the database where all the data is stored and if it matches the user successfully completes authentication. Else forgot password is provided for the user to reset the password, and f the user is new then sign up button is provided to sign up. If required fields are not filled then it generates a warning that is shown below.

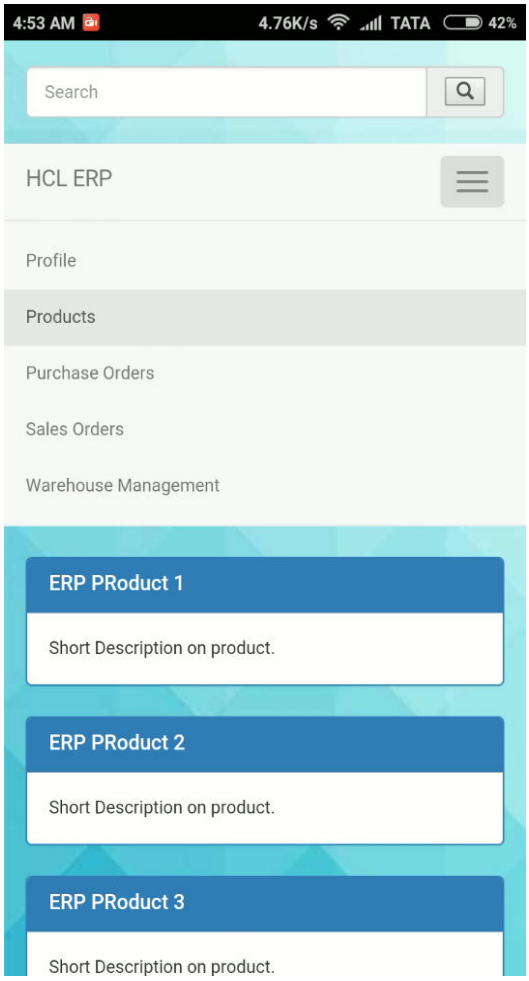

4:53 AM
84.4K/s
TATA
42%
Warehouse Management
Username:
Enter User Id
Password:
Enter password
Select Role
Manager
Remember me
Submit
2016_09_27_04_52_47
5.98K/s
TATA
Warehouse Management
Username:
user1
Password:
Alert
All Feilds are required
OK
Submit

Product Selection

User can select the Product among the products which are listed, after choosing a product he can go into that product details and then access the option which he want.

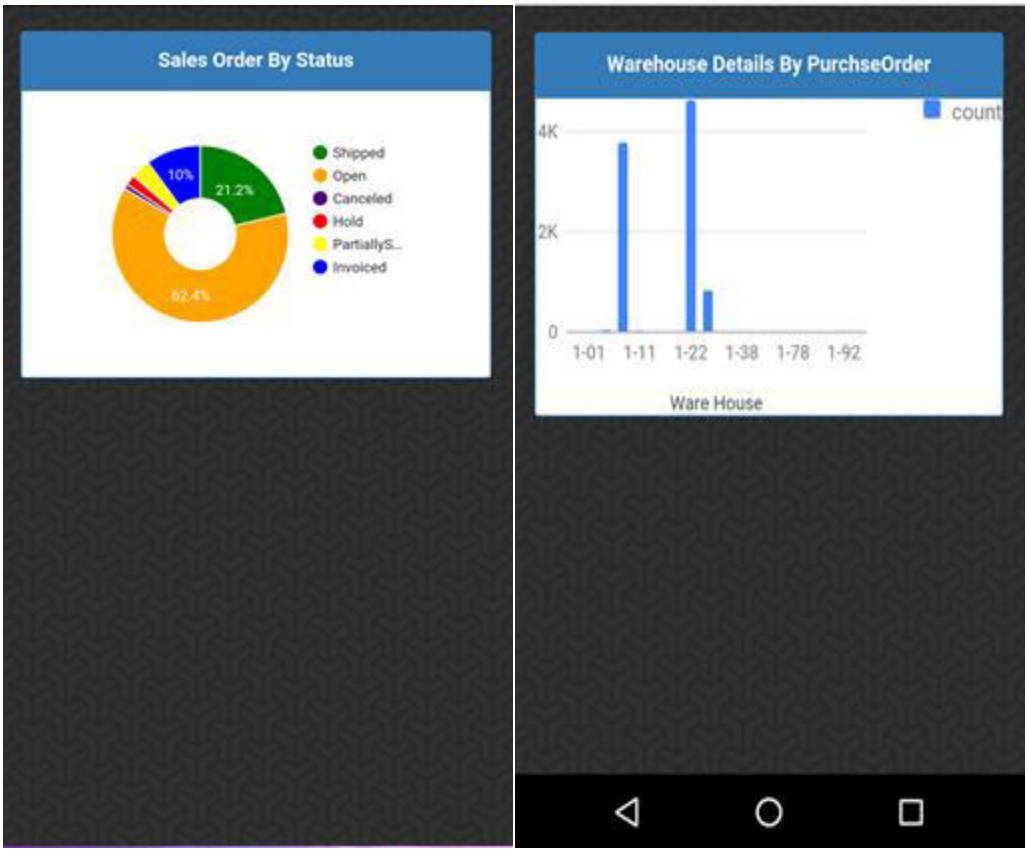
Report choosing

User can choose the purchase order report, sales order report, Warehouse management report and purchase order. After choosing he will be able to see the report



Graphical report

If user chooses purchase order reports then it displays the purchase order reports graphically in the form of bar graph or pie chart etc. and if user chooses sales order reports then it displays the sales order reports graphically in the form of bar graph, pie chart etc.. and same with



3.Detailed description of components

3.1 Authentication

Component Name: Authentication	
Brief Description: User has to login to the application using user name, password, role , and if these all things match then authentication is done by the user.	
Attributes	Attribute description
User name	This will be the unique identity of the users. Username will be helping for the users to login to the system. This will be the primary identity of the authentication.
Password	This will be secret identity of the user to logon to the system. Along with the username this secret (“Password”) will be needed for a successful authentication.

Role	Role will be differentiating among the users. This is based on the hierarchy levels of the users. Depending on the role the activates of the users may be different.
Methods	Method Description
Submit();	This method will helps us to submit the values entered by the user and hand it over to verifier.Values in this function will be sent as a data string.
Verifier();	This method will get the values from the submit function and verify them to the database, and return true if the user is valid.
Getresult();	This method will take the result from the verifier method and return it to the actor(user).Allow the user to do the further steps if valid or rejects all the other request.

3.2 Product Management

Component Name: Product Management	
Brief Description: After proper authentication, in this module User has to select the product among the product list provided for the further access.	
Attributes	Attribute description
Product_Name	This will be the name of the product that is existing on the database. This will be helpful for defining the particular product.
Product_ID	This attribute will be primarily defining the product. This will be unique for all the products and helps us to identify the product at any point of code
Warehouse_id	This will be Id of the warehouse to which the product will be belonging. This defines that particular will be belonging to warehouse for better monitoring.
Methods	Method Description

viewProduct();	This method will give complete details of the product that is existing in the database. This will be read only function. Updates cannot be done here.
SelectProduct();	This method helps in selecting particular product for that session. This can make us to use a same product of that session.
UpdateProduct();	Modifications of the product will be updated to the database using this method. Selecting of the product will be done using the selectproduct() method.
AddProduct();	Using this method we can add new products to the database. All the variability of the product i.e all the attributes of the product which are mandatory should be given here.

3.3 Purchase Order

Component name: Purchase_Order	
Brief Description: In this module customer is supposed to initiate the product creation and he can also view purchase order status.	
Attributes	Attribute Description
Product_name	This will be name of the product that was made as purchase. The name of the products will be from the products. This help user to identify the product.
Product_ID	This will be id of the product that was made as purchase. This helps the developer to identify the product at any point of the product. This will be the primary identification of the product.
PurchaseorderstatusID	This will be the primary identification of the purchase that has been made. This helps us to track the purchase order.
Purchase_date	This attribute will be storing the date of the purchase has been made. This helps us to track down the reports. This

	makes reports much easier and much accurate.
Method();	Method description
Productordercreation();	This method will be creating the orders based on the inputs of the actors. Purchase Ordes will be created this function and cannot be modified here.
Makepayment();	After PurchaseOrderCreationId() method creates order payment will be done uding this method. This only helps in payment using orderId
PurchaseOrderStatus();	This method will retrieve the status of the purchase that has been made. This will can be established using purchase order Id attribute.
UpdatePurchaseStatus();	After the purchase is made and its payment is made through respective methods. To modify that particular purchase order this method helps handful. This is done using PurchaseOrderId.

DeletePurchaseOrder();	After the purchase is made and its payment is made through respective methods. To delete the existing Purchase Order in the database, this method helps us to achieve this.

3.4 Purchase Order Report:

Component Name: Purchase Order report	
Brief Description: In this module manager can view purchase order reports.	
Attributes	Attribute description
Purchase order id	When the customer initiates order creation a purchase order id is developed that is included in purchase order report.
Methods	Method Description

ViewPurchaseOrderReport()	When the customer initiates order creation a purchase order id is developed that is included in purchase order report data to represent them in the form of report, and when the manager wants to view the purchase order report we use this function.
EditPurchaseOrderReport()	When manager wants to modify the purchase order data in the report then we use the function edit purchase order report.

3.5 Sales order

Component Name: Sales Order	
Brief Description: Request of the needed product gets granted, if required is present in warehouse. Ever order has its own unique ID which contain rest all the required products under it.	
Attributes	Attribute description
SalesOrderID	Since every delivery to the customer needs to be distinctively identified, hence each sales order is given a unique ID to do the needed. Under this will contain all the products

	ordered by the customer.
Date	Date of order by the customer
ProductName	This attribute will give us information regarding all the product_name under this sales order and with quantity.
Methods	Method Description
UpdateSalesOrder()	This method helps to create a sales order when requested by customer. This will eventually then create a unique ID i.e., SalesOrderID.
DeleteSalesOrder()	This method helps in deleting SalesOrder once the order is successfully delivered to the customer, i.e., eventually will delete SalesOrderID when done.

3.6 Sales order Report

Component Name: SaleOrderReport	
Brief Description: Sales order report deals with viewing the existing sales orders and other purchase order and their tracking details. The data is viewed in graphical format.	
Attributes	Attribute description
SalesOrderID	Since every delivery to the customer needs to be distinctively identified, hence each sales order is give a unique ID to do the needed. Under this will contain all the products ordered by the customer.
Methods	Method Description
ViewSalesOrderReport()	This method helps in viewing existing sales order by the authenticated users. It shows the updated version of sales order and their tracking details and an expected date of delivery.
EditSalesOrderReport()	This helps in manually edit the tracking details of order by the authorized persons.

3.7 Warehouse Management

Component Name: Warehouse Management	
Brief Description: Each warehouse has a unique Warehouse address and shipment to this address, any returned product will then get updated into the warehouse.	
Attributes	Attribute description
WarehouseAddress	There are various warehouses connected to the company, hence for every warehouse its address needs to be specified for further updation ad deletion of products and see the availability of products.
ReturnShippingID	Once a customer wants to return any of the product, then product gets a ShipmentID which will get updated to its nearest warehouse, and accordingly the quantity is updated.
WarehouseID	Since there are various warehouses, hence each of it is given a unique ID to identify distinctively. This will further help for shipment of products to customer and also have a common place which will keep track of the products in and out of warehouse.

Methods	Method Description
UpdateProductWarehouse()	This method helps update the quantity of the product in the warehouse, and similarly updates for any of return cases too depending on the warehouse it enters.
DeleteProductWarehouse()	This method helps to delete or decrease the quantity of the product once the product leaves warehouse for shipment to delivery to customer.

3.8 Warehouse Reports

Component Name: Warehouse Report	
Brief Description: Every warehouse has a unique ID and along with its name, report tells us about the product purchase order, hence will generate purchase order request as per the customer request. And similarly, sales order is generated which will further help in processing the product to its destination.	
Attributes	Attribute description
ProductIDWarehouse	This attribute gives a unique ID to product in the warehouse. As their will be various different products

	available in the warehouse, hence ID helps to distinctively identify the products with their quantity which further helps to update it whenever needed.
ProductNameWarehouse	This attribute provides name to the product. Or in other words a superior name to the product to which various other sub-products get their unique ID to distinctively get identified.
PurchaseOrder	This attribute helps to provide an information regarding order that needs to get placed when needed by the warehouse for completion of any intermediate processes.
Methods	Method Description
UpdateProductReport()	- This method updates the product name into the warehouse when comes in. It also updates the quantity of the product name in warehouse.
DeleteProductReport()	This method helps to decrease quantity of the product from warehouse once an order is placed and it leaves the warehouse to fulfil the customer request.

GeneratePurchaseOrder()	This method helps to generate a purchase order ones it finds that the quantity of the product required does not meet existing quantity, to give out the final product which needs to be sold.
GenerateSalesOrder()	This method helps to generate a sales order when a customer places its purchase request, it changes into sales order to the warehouse, for further processing of the request.

3.9 Graphical Report

Component Name: Graphical report	
Brief Description: In this module all the graphical reports of purchase order reports, sales order reports, and warehouse reports are showed in graphical form.	
Attributes	Attribute description
PurchaseOrderReport	When purchase order is done modification is done in purchase order report.
SalesOrderReport	When sales order is done modification is done in sales order report.

Methods	Method Description
ViewPurchaseOrderGraphicalReports()	When manager wants to view the purchase order reports in the form of pie charts or bar graphs etc.. Then View Purchase order Graphical reports is used.
Edit PurchaseOrderGraphicalReport()	When manager wants to edit the purchase order reports to modify them in the form of pie charts or bar graphs etc.. Then edit Purchase order Graphical reports is used.
ViewSalesOrderGraphicalReport()	when manager wants to view the sales order reports in the form of pie charts or bar graphs etc.. Then View sales order Graphical reports is used.
EditSalesOrderGraphicalReport()	When manager wants to edit the sales order reports to modify them in the form of pie charts or bar graphs etc.. Then edit sales order Graphical reports is used.
ViewWarehouseGraphicalReport()	When manager wants to view warehouse graphical reports of various sales and purchase orders we use the function view warehouse graphical reports.
EditWarehouseGraphicalReport()	When we to edit warehouse graphical reports of various sales

	and purchase order we use the function edit warehouse graphical report.

4.Reuse and relationships to other products:

Reusing the existing or earlier code will be a better and enhanced way of making the product instead of replacing or writing the code from scratch. This is because writing each and every component and class will be taking a long time and development of that particular code will takes more time. Writing from scratch will kill all our time for developing the code to higher level. As the project that we are working is to be made from scratch, because there “nothing” made on this type ever before. So coding will be one of the most crucial part of the project as we need to make everything from scratch. Even though there was nothing made related to this project there are many frameworks which we can use for faster and enhanced development of the product. Framework such as Bootstrap, Google material design and other front-end frameworks are used in developing the user interface and UX. This made the code simpler and still more easier for developing. While developing the front-end of the product more time has been utilized in developing the module instead of writing it from the scratch or the basic model of it. Reusing the framework helps in developing the product to the higher level rather than adjusting its basic model. For back-end working also there are many frameworks available but we are not able to use them in developing the product. This is because we restricted to use all sever side scripts as hybrid application development platform. PhoneGap- Hybrid applications development , will not allow to use any type of server side scripts. So we cannot use any backend frameworks. Even in the project much of the code is reusable like, graphical reports generator methods, custom front end designs and others for upcoming projects.

5.Design Decision and Tradeoff

Software design document that has been prepared for this document is completely reader friendly. All the sections and sub sections are mentioned clearly. Reader must at least know about the idea of the product that is being developed and he/she can understand complete document. Even if the reader does not have any idea of the project he can understand this if he can understand basic software engineering terms. All the notations that are used/represented in the following document are in general notation. Even if the notations are not understandable, all the descriptions are given below that section. Most of the problematic situations are easily tackled using many type of graphical representation. Use cases, sequence diagrams, module representations and many others are used to make readers understand the situation/problem easily. All these graphical models of representing things helps to solve the problem easily also understanding them. Some of the user interface is compromised, even though those are helpful for the users they make them confusing in using them. Not only them some of the important functions they are abandoned, this is because those come in the mid-way of the users while using application. Although application should be developed with all the functions, finally this should be used by users if they are not comfortable in using them then removable is better than anything.

6.Pseudo Code

Module → Authentication	

Submit()	Var userid;
	Var Password;
	Var Role;
	//Send to verifier method using ajax and post
	$.ajax{
	Method: post;
	url: //define url
	}}

Verifier()	$userid , $password, $role; //get them from submit
	$result = “Select * from where …… “ // sql query to to check whether user exists and match all needed factors
	Return $result; }

Getreult()	$result ; //get the combined result
	$vaild = // validate the result for appropriate user
	$values = “select * from where….. “ //sql query to get user details
	Return $values;} // return final valid ouptput

Module → ProductManagement	

viewProduct()	Var array;
	$result = “select * from where….. “ //query to get the product from database
	$arrary = mysql_fetch_row($result); //store values in arrary
	Return $arrary;}

SelectProduct()	Var product_id; //select product from list
	Window.localstorage = (“id” : “product_id”); //making it global
	$result = “select * from where…. “ //get product
	Return $result;}

updateProduct()	$arrary_modifies; //modified values
	$q = “update set= …. Where …… “ //updating
	Return $q;} //return progress;

addProduct()	$array_values; //new product values
	$result = “ Insert into value….. “ //inserting
	Return $result;} //return progress

Module → PurchaseOrder	

purchaseOrderCreation()	var productId;
	var array_details;
	var purchaseOrder_id; //created Id
	$res = “Insert into values= …. “ //inserting new purchase
	Return $res;}

Makepayment()	Var purchaseOrderId;
	Var cost;
	$result = Thirdpartycall(); //make payment from third party apps with id and cost
	//verify $result;
	Return $result;}

purchaseOrderStatus()	var purchaseOrderId; $result = “select status from where….. “ //getting from database $value = mysql_fetch_row($value); Return($value); }

updatePurchaseOrder()	var pruchaseOrderId;
	var array_modifies; //updated values
	$update = “update set= ….. where… “//updating
	Return $update;} //return progress

deletePurchaseOrder()	var purchaseOrderId;
	var remark;
	//validity check, can delete or not
	$res = “Delete from where… “ //deleting
	Return $res;} //progress

Module → PurchaseOrderReport	

viewPurchaseOrderReport()	var array_variables; //all required variables like fromdate , to date and othere
	$result = “select *from where…. “ //getting all details from database using above variables
	$result = mysql_fetch_row($result);
	Return $reuslt;} //final values as rows

editPurchaseReports()	var array_values; //all editable values
	$result = “update set…. Where… “;
	Return $result;} //return progress

Module → GraphicalReport	

ViewPOgraphicalReport()	Var array_values; //get all purchase order values from the
	
	Var report = generateGraphicalReport(array_values); //this will be the third party application method which will generate graphical using the above values
	
	Var new_report = valid(report); //this will be managing the sizes and others of the report
	
	Return new_report;

editPOgraphicalReport()	var array_updates; //changeable values updates
	
	var update = generateGraphicalReport(array_ updates); //this will be the third party application method which will generate graphical using the above updated values
	
	Var new_report = valid(update); //this will be managing the sizes and others of the report
	
	Return new_report;}

ViewSOgraphicalReport()	Var array_values; //get all Sales order values from the
	
	Var report = generateGraphicalReport(array_values); //this will be the third party application method which will generate graphical using

	the above values Var new_report = valid(report); //this will be managing the sizes and others of the report Return new_report;

editSOgraphicalReport()	var array_updates; //changeable values updates
	
	var update = generateGraphicalReport(array_ updates); //this will be the third party application method which will generate graphical using the above updated values
	
	Var new_report = valid(update); //this will be managing the sizes and others of the report
	
	Return new_report;}

ViewWarehousegraphicalReport()	Var array_values; //get all Warehouse values from the
	
	Var report = generateGraphicalReport(array_values); //this will be the third party application method which will generate graphical using the above values
	
	Var new_report = valid(report); //this will be managing the sizes and others of the report
	
	Return new_report;}

editWareHousegraphicalReport()	var array_updates; //changeable values updates
	
	var update = generateGraphicalReport(array_ updates); //this will be the third party application method which will generate graphical using the above updated values
	
	Var new_report = valid(update); //this will be managing the sizes and others of the report
	
	Return new_report;}

Module → WarehouseManagementSystem	

UpdateProductswarehouse()	Var $array_update; //changeable values
	
	$products = “select * from where… " //sql query to get all products from ware house
	
	$array_pro = mysql_fetch_row($products);
	
	//select one product from above array
	
	$sql = “update set=…. Where…… “; //update on selected
	
	Return $sql;} // get progress

deleteProductswarehouse()	Var $delete; //deleteable values
	
	$products = “select * from where… " //sql query to get all products from ware house
	
	$array_pro = mysql_fetch_row($products);
	
	//select one product from above array
	
	$sql = “Delete from Where…… “; //delete the selected
	
	Return $sql;} // get progress

Module → WarehouseReport	

updateProductReport()	var product_variables;
	
	$products = “select * from where….. “ //selecting all the products with above variables filtering
	
	$reports = mysql_fetch_row($products);
	
	//show all the rows as areport from here
	
	$select_productreport = //select from above
	
	$query = “update set = …. from where…. “ //update if allowed
	
	Return $query;} //return progress

deleteProductReport()	var product_variables;
	
	$products = “select * from where….. “ //selecting all the products with above variables filtering
	
	$reports = mysql_fetch_row($products);
	
	//show all the rows as areport from here
	
	$select_productreport = //select from above
	
	$query = “Delete where…. “ //delete if allowed
	
	Return $query;} //return progress

generatePurchaseOrderReport()	var purchaseOrder_variables;
	
	$purchase = “select * from where….. “ //selecting all the purchase orders with above variables filtering
	
	$reports = mysql_fetch_row($purchases);
	
	//show all the rows as a report from here
	
	Return $reports;} //return progress

generateSalesOrderReport()	var salesOrder_variables;
	
	$sales = “select * from where….. “ //selecting all thesales orders with above variables filtering
	
	$reports = mysql_fetch_row($sales);
	
	//show all the rows as a report from here
	
	Return $reports;} //return progress

Module → SalesOrders	

UpdateSalesOrder()	var sales_variables;
	
	$sales = “select * from where….. “ //selecting all the sales with above variables filtering
	
	$reports = mysql_fetch_row($sales);
	
	//show all the rows as areport from here
	
	$select_productreport = //select from above
	
	$query = “update set… from where…. “ //delete if allowed
	
	Return $query;} //return progress

deleteSalesOrder()	var sales_variables;
	
	$sales = “select * from where….. “ //selecting all the sales with above variables filtering
	
	$reports = mysql_fetch_row($sales);
	
	//show all the rows as areport from here
	
	$select_productreport = //select from above
	
	$query = Delete from where…. “ //delete if allowed
	
	Return $query;} //return progress

Module → SalesOrderReport	

viewSalesOrderReport()	var sales_variables;
	$sales = “select * from where….. “
	//selecting all the sales with above variables filtering
	$reports = mysql_fetch_row($sales);
	//show all the rows as areport from here
	//show the report in all formats if needed
	Return $query;} //return progress

editSalesOrderReport()	var sales_variables;
	var update_variables;
	$sales = “select * from where….. “
	//selecting all the sales with above variables filtering
	$reports = mysql_fetch_row($sales);
	//show all the rows as areport from here
	//show the report in all formats if needed
	$query = //make possible query upon needed requiremnts
	Return $query;} //return progress