Assignment

ITS1013 – Database Programming

BSc (Hons.) in Computer Science via GDSE

Take-home assignment and VIVA

Total Marks: 100

Objectives

- Make the student able to develop a standalone application with database connectivity.
- Testing the student's ability to learn about software architectures and design patterns on their own.
- Make the student familiar with system documentation and System Manual creation.

Coursework Requirements and Instructions

- You are asked to analyze, design and implement a standalone application for a wholesale distributor using the knowledge on JDBC.
- You are required to create the application in this coursework as a Java Swing or JavaFX, whichever you are familiar with most.
- This coursework consists of a case study and the coursework guidelines section.
- You are required to refer to the Coursework Guidelines to understand the specific guidelines to be followed when developing the project required.
- You should be able to explain the functionalities and the concepts used in this project at the viva voce.

Submission

- You should submit the coursework on or before the due date specified.
- Your assignment is to be made into a .zip file with the file name format [GDSEBatch_ITS1013_Name], and submit your deliverables and application code to Google Classroom on or before the deadline.

Eg: If your GDSE batch is 55 and your name is Nimal Perera,

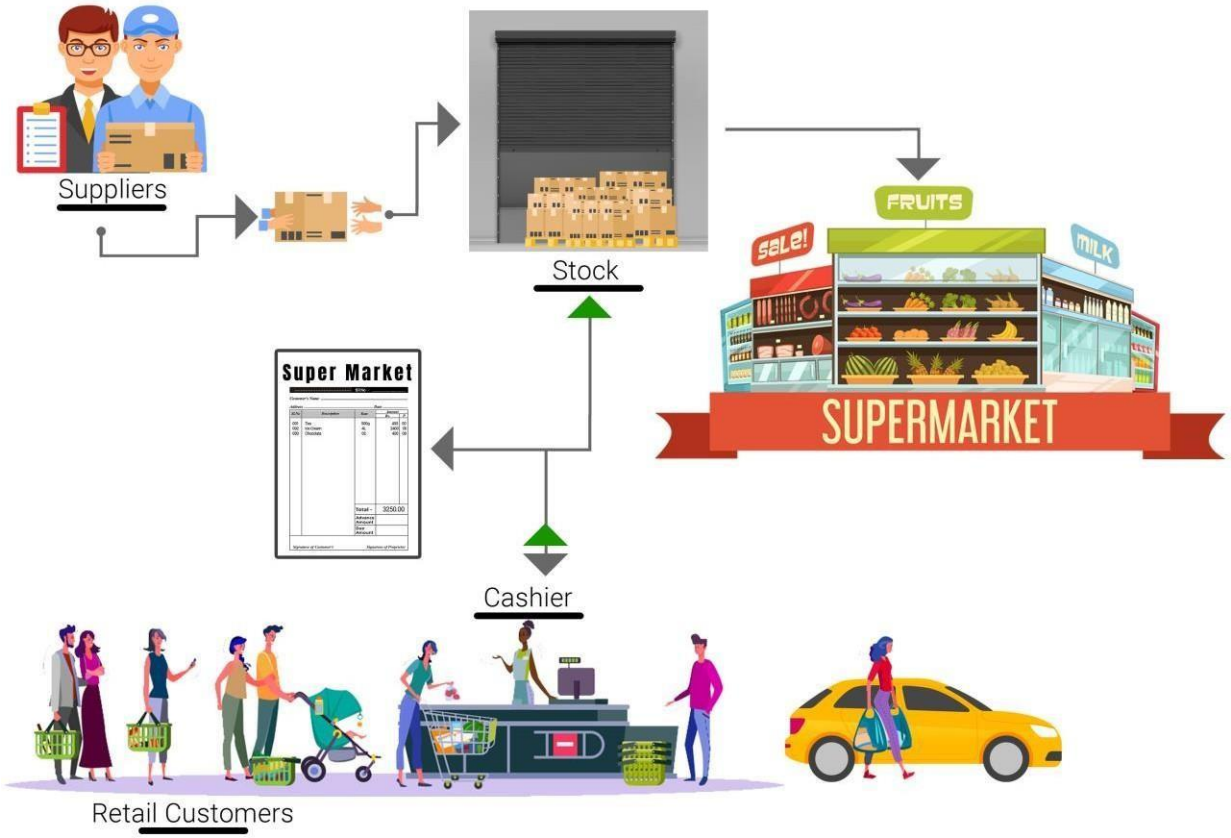
- The .zip file name: GDSE55_ITS1013_NimalPerera.zip

Case Study

1.0 Nature of the Business / Business Overview

A wholesale distributor distributes goods for their retail clients (customer). The company has a huge stock and it is maintained by a storekeeper. Retail customers make their orders from the cashier and the cashier sends customer orders for the stores. (The system can have multiple cashiers.) Stores can prepare orders within a few hours. The cashier of the company is responsible for creating the final bill for the customer and the customer needs to pay the amount in full (all other payment options are disabled). Discounts are available for items. For management purposes, few reports have been introduced such as daily, monthly and annually sales statics and popular goods in different seasons. Seasonal trends are most important for the company when the new supplier orders made.

Figure 1.0.1 Business Overview



Step 01: Customer place their orders to the cashier (list of items)
Step 02: Cashier send the orders to the stores and store will prepare a list of goods and send them back to the cashier.
Step 03: Cashier creates the final bill of the order according to the items received from the stores.
Step 04: Customer pays the amount in full.

1.1 System Overview

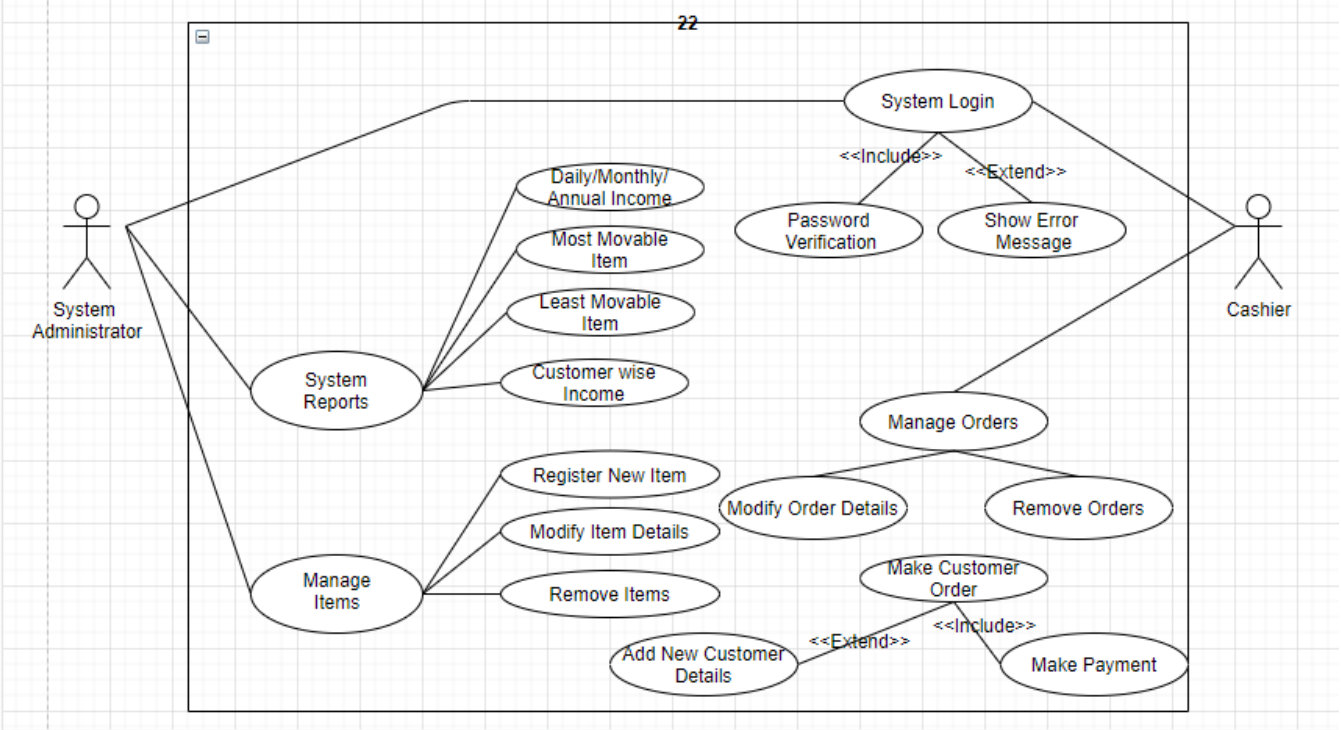


Figure 1.1.0 Use-case Diagram

Mainly system features can be divided into two sections, which are administrative tasks and user operations. Under administrative tasks, add, delete and modify items will be specified, and under user operations, add, delete and modify orders and generate system reports will be specified. In figure 1.1, the main use case of the system is to **"Place a Customer Order"**

Use-case description of the main use case.

Table 1.1.0 use case description of "make a customer order"

Name of the Usecase	Make a customer order
Actor Involved	cashier
Pre-condition	Add a new customer is enabled. If adding a new customer is needed, Confirmation of order function/button should be disabled. Load descriptions of item to the relevant display component.
Flow of event	Add a new customer if it is needed. Auto-generate order ID. Select Item and display corresponding values(discount amount and quality on hand). Enter order quantity and activate "add to list" button Add items to the list (JTable). Calculate the current total price, discount and individual discount of each item. After adding item to the item list (JTable), confirmation of the order button should enable the user to press. When it is needed, the list of items can be edited (change the quantity or remove the item from the list). If there is no item in the item list JTable, confirmation of the order button should be disabled. Cancel order button is enabled all the time.

Post-condition	List of items that customer ordered should save in Orders and Order_detail tables and order quantity of each item should be deducted from the item tables' quantity on hand. A confirmation message should be displayed for every successful order.
Non-Functional requirements	Focus user-friendliness of the UI for the user with a light colour theme. Remove distracting colours from the UI. The colour of the "Confirmation of Order" button should be green or any kind of light colour except red, orange or pink (similar colour variations to red). User should be able to look at the interface for a long time without any vision issue.

Table 1.1.1 use case description of "manage customer order"

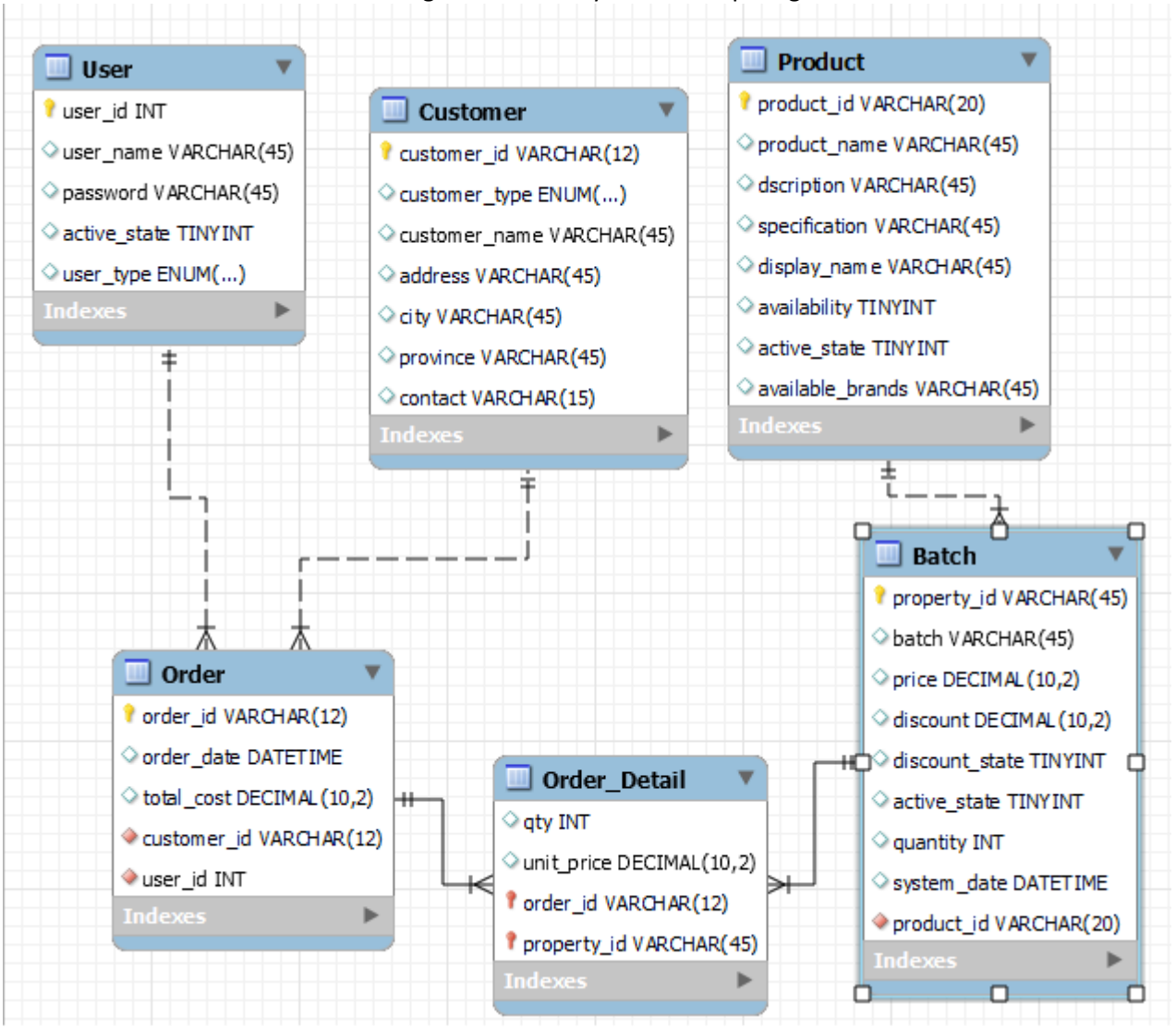
Name of the use case	Manage customer order
Actor involved	Cashier
Pre-Condition	Enter customer number / ID should enable the casher. Load descriptions of item to the relevant component. Orders list should be empty
Flow of event	Enter customer ID [edit order] Display list of orders of the particular customer which can be selected separately. When clicked on a particular order ID, all items of the order should be loaded to a JTable (Itemcode, description, orderQty, unitprice). Now cashier should able to make changes for the order (change order quantity or remove the item from the item list). Enable confirmation of order edit button. If there is no item in the item list JTable, confirmation order edits button should be disabled. Cancel order button is enabled all the time. [remove order] If remove is confirmed quantity of hand at item table should be updated simultaneously.
Post-condition	[edit order] If there are changes that have been done, order and order_details should update corresponding to the changes. Particular count of order items should update with the item table respectively. For order confirmations, a prompt for confirmation with a warning message should display. [remove order] A confirmation of remove order should prompt to the cashier.
Non-Functional requirement	Maintain user-friendliness UI for the user with a light colour them. Remove distracting colours from the UI. The colour of the "Confirmation of Order Edits" button should be greenor any kind of light colour except red, orange or pink (similar colours to red).

1.2 System Reports

As reports are the main aspect of the business, few reports have been introduced to the system to support management decisions. For this system, the developer can show reports of data in JTables rather than adding third party software tools.

1.3 Database Design

Figure 1.3.0 Entity Relationship Diagram



Note: Orders are mentioned in plural because "Order" is a reserved word for SQL

- One customer can place at least one or many orders.
- One order must have a specific customer.
- One order can have at least one or multiple items.
- One item type may be sold in multiple orders or the particular item may not be sold at all.

The following assumptions must be concerned when drawing the ERD for the system. This ERD design should be limited to only 4 entities, namely - customer, order, item and order_detail. Because the basic functions of the POS system have to be developed. Handling GRN, GIN, Item categorisation and batch processing are not to be concerned when developing this system. The quantity on hand (current stock) is maintained in the item table itself.

Coursework Guidelines

2.1 Recommended Project Scope

1. Only four data tables should be created to keep track of the details of the system. (Refer to Figure 1.3.0)
2. “Add Item”, “Manage Customer Order” and “Place Customer Order” are the specified main functions of the system. Other functions can be ignored when marking the project.

2.2 Functional Behaviors

1. When an order is saved, automatically, corresponding details of goods should be saved. An order can be saved without a specific customer but adding customer details is optional for orders as the full payment is done by the customer. Because of the cash is handed in full, it is not compulsory to add customer details to the system.
2. When cashier creates the bill, the “quantity on hand” of each selected item should be displayed to the cashier.
3. A discount can be given for a specific or selected item when adding items to the item list. And able to edited on the item list.
4. The maximum discount of each item should be displayed to the cashier if a discount is available for the item.
5. Total price and discount price of each item and the collective discount should display in proper places.
6. Each data report should load to a JTable and you can specify the report to be daily or monthly or annually report at the beginning.
7. Proper validations for each interface are compulsory.
8. Maintain the data constancy using transactions.

2.3 Architectural Design of the system

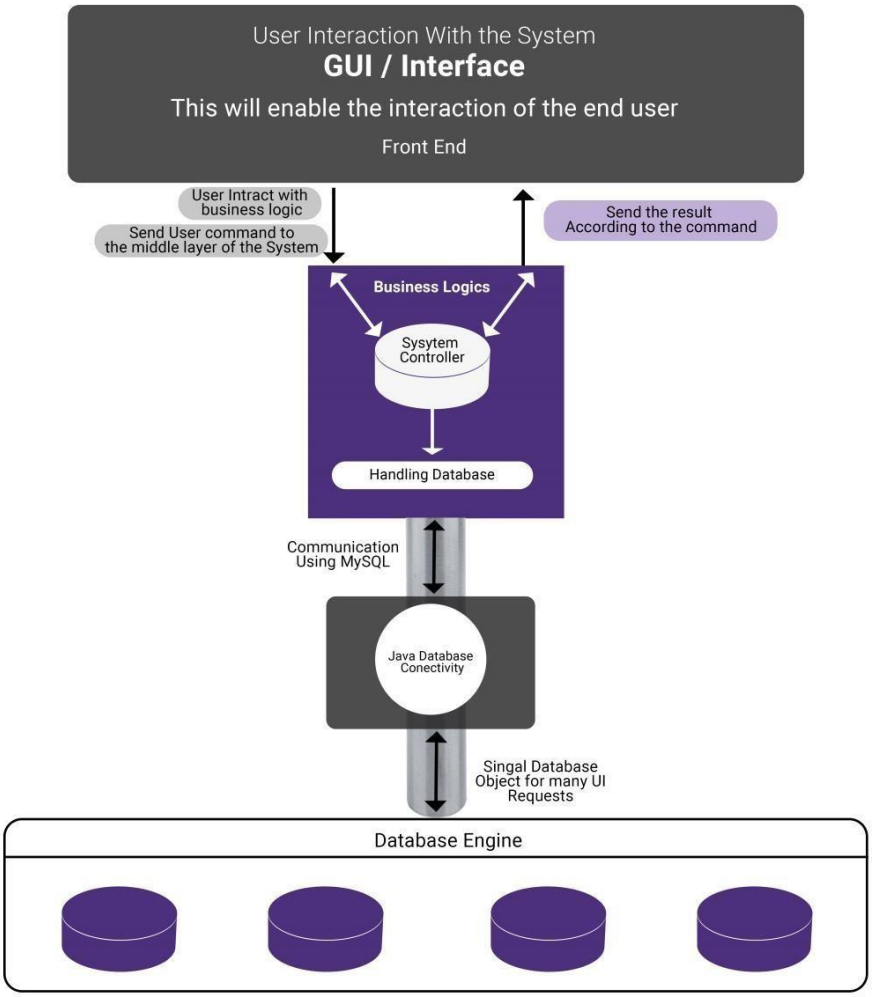


Figure 2.3.0 design of the system

2.4 Wireframes

Figure 2.4.0 wireframe of place customer order

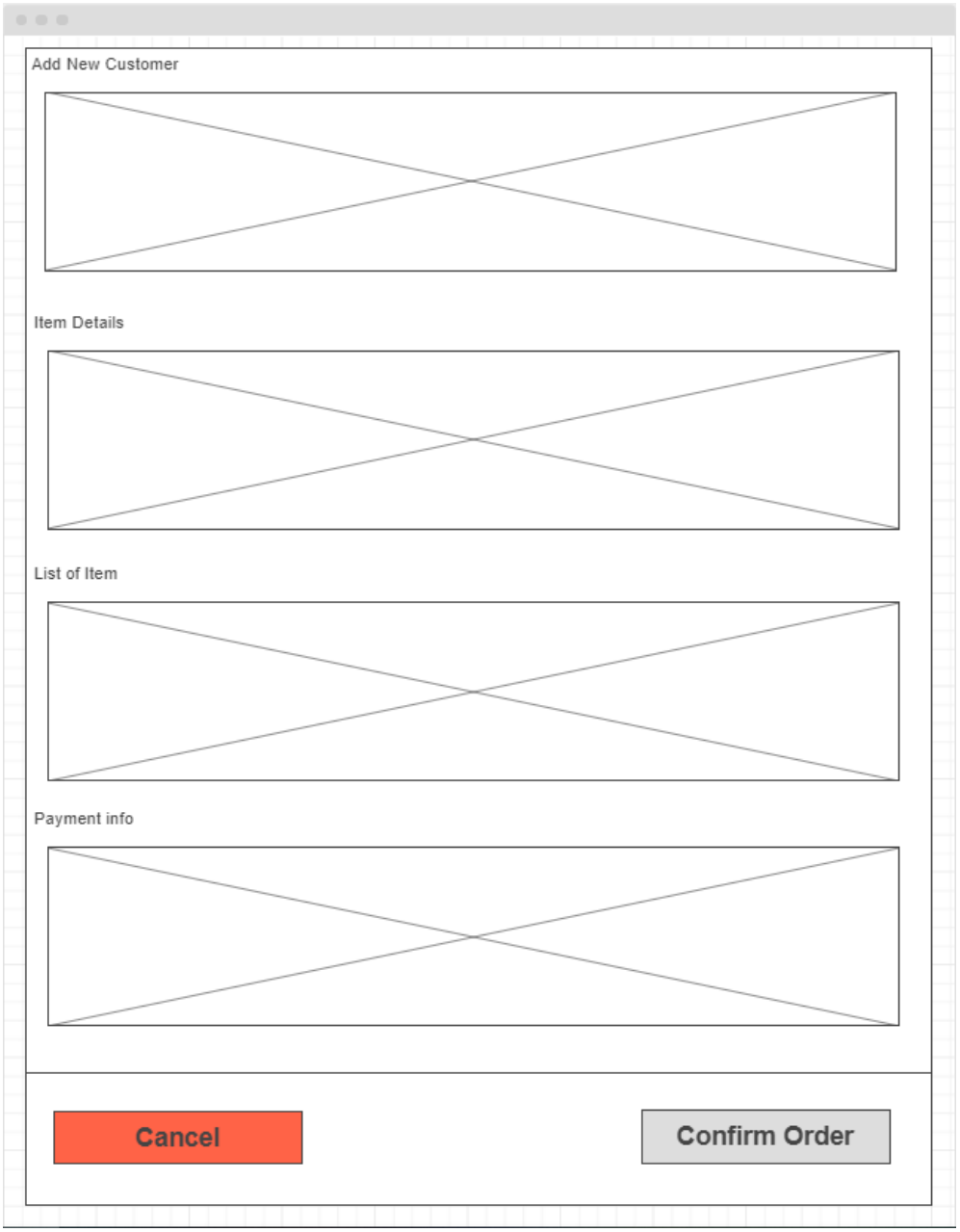
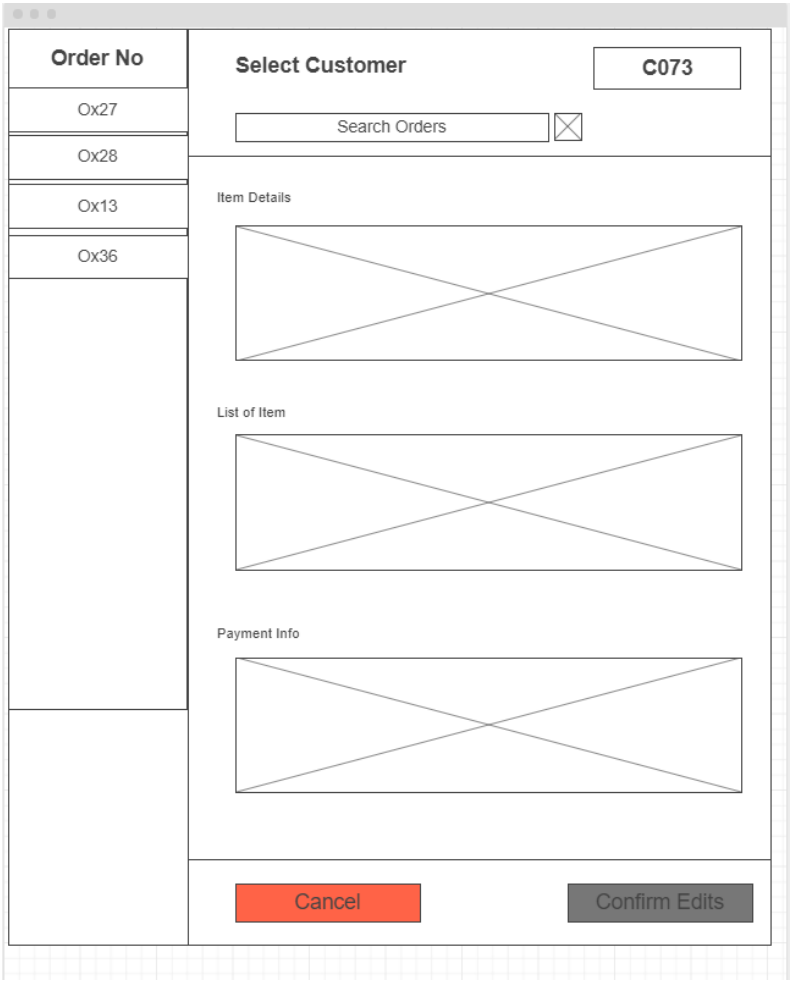


Figure 2.4.1 wireframe of “manage customer order”



2.5 User Privileges

A user privilege is the right to execute a particular function, feature or a task from the system. In this system, two types of users can be identified, which are administrator and user.

Role of Administrator:
The administrator of the system has rights to add, delete and modify items and, able to manage system reports.

Role of User:
The user of the system has rights to place customer orders and manage customer orders as mentioned in the high-level overview.

2.6 Deliverables

1. Executable file using EXE4J or any other third party software.
2. User manual describing all features and functions of the system.

Note: No need to deliver the project files as your knowledge on the project developed will be tested at the viva.

2.7 Skeleton of the system

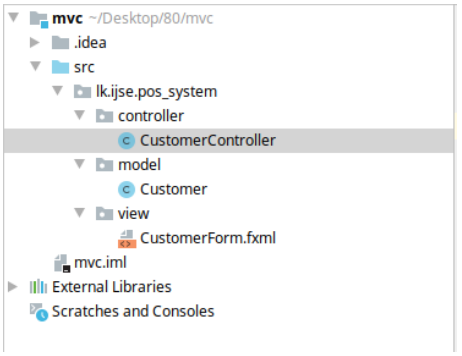


Figure 2.7.0 MVC

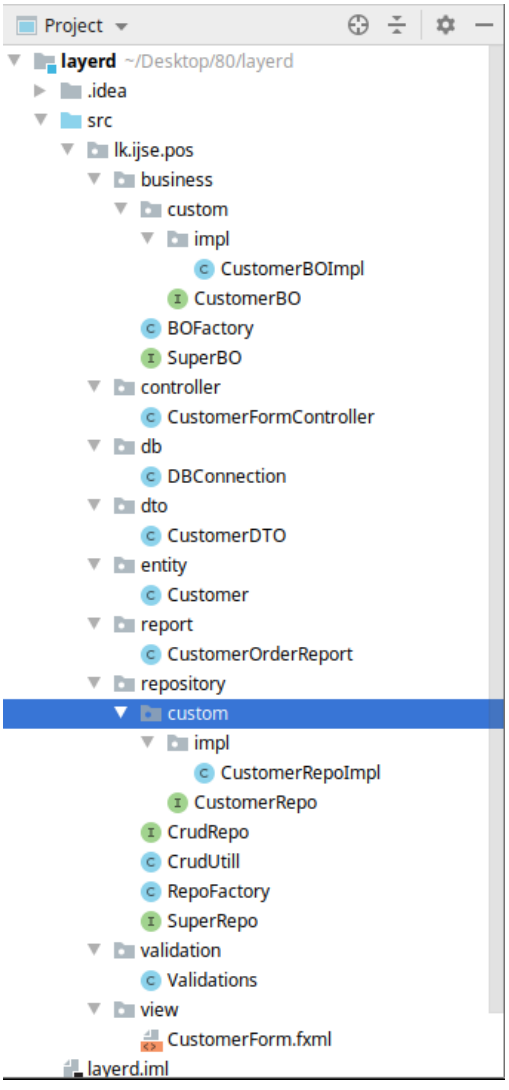


Figure 2.7.1 Extended MVC (Layered Architecture)

2.8 Development Environment

Table 2.8.0 development

IDE	InteliJ Idea
Language	Java
Version	JDK version(1.8)
DBMS	MySQL version(x)

x= any version that you are familiar with.

References:

MVC architecture: https://www.oracle.com/technical-resources/articles/java/java-se-app-design-with-mvc.html

Layered architecture: https://www.oreilly.com/library/view/software-architecture-patterns/9781491971437/ch01.html