

DESIGN AND DEVELOPMENT OF ARCHITECTURE OF POS

BY

FARHANA HAQUE

ID: 091-15-725

AND

TAMANNA RAHMAN

ID: 091-15-739

This Report Presented in Partial Fulfillment of the Requirements for the Degree of
Bachelor of Science in Computer Science and Engineering

Supervised By

Dr. Syed Akhter Hossain

Professor and Head
Department of CSE, CIS, CS
Daffodil International University



DAFFODIL INTERNATIONAL UNIVERSITY

DHAKA, BANGLADESH

OCTOBER 2012

APPROVAL

This Project titled **Design and Development of Architecture of POS**, submitted by Farhana Haque and Tamanna Rahman to the Department of Computer Science and Engineering, Daffodil International University, has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of B.Sc. in Computer Science and Engineering and approved as to its style and contents. The presentation has been held on *date*. [Font-12]

BOARD OF EXAMINERS

Dr Syed Akhter Hossain Chairman
Professor and Head
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Dr Yousuf Mahbubul Islam Internal Examiner
Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Dr Md Kabirul Islam Internal Examiner
Associate Professor
Department of Computer Science and Engineering
Faculty of Science & Information Technology
Daffodil International University

Dr Muhammad Shorif Uddin **External Examiner**
Professor
Department of Computer Science and Engineering
Jahangirnagar University

DECLARATION

We hereby declare that, this project has been done by us under the supervision of **Dr.Syed Akhter Hossain, Professor and Head, Department of CSE, CIS, CS Daffodil International University**. We also declare that neither this project nor any part of this project has been submitted elsewhere for award of any degree or diploma.

Supervised by:

Dr Syed Akhter Hossain
Professor and Head
Department of CSE, CIS, CS
Daffodil International University

Submitted by:

Farhana Haque
ID: 091-15-725
Department of CSE
Daffodil International University

Tamanna Rahman

ID: 091-15-739

Department of CSE

Daffodil International University

ACKNOWLEDGEMENT

First we express our heartiest thanks and gratefulness to almighty Allah for His divine blessing makes us possible to complete this project successfully.

We fell grateful to and wish our profound our indebtedness to **Dr.Syed Akhter Hossain, Professor and Head**, Department of CSE, CIS & CS Daffodil International University, Dhaka. Deep Knowledge & keen interest of our supervisor in the field of wireless network influenced us to carry out this project. His endless patience ,scholarly guidance ,continual encouragement , constant and energetic supervision, constructive criticism, valuable advice ,reading many inferior draft and correcting them at all stage have made it possible to complete this project.

We would like to express our heartiest gratitude to all the faculty members and the staff of CSE, CIS & CS department of Daffodil International University. We would also like to thank our entire course mate in Daffodil International University, who took part in this discuss while completing the course work.

Finally, we must acknowledge with due respect the constant support and patience of our parents.

ABSTRACT

POS (Point of Sale) system is a very popular information system widely used in most of the shopping malls worldwide. Despite its popularity, POS still needs architectural development from Software Engineering perspectives due to insufficient modeling of such systems in real life.

This project is on “**Design and Development of Architecture of POS**”. In this work we developed conceptual and logical model of POS from the perspectives of Software Engineering to propose an Object Oriented framework for the systems. The proposed model has also been implemented using JAVA with backend in MySQL and for the report generation we used ireport, an open source API for java report generation facilities.

After implementation the developed POS along with its model is tested for necessary transactions and found satisfactory. The future of this POS could be further improved through different extensions of the proposed model.

TABLE OF CONTENTS

CONTENS PAGE

Board of examiners.....	I
Declaration.....	II
Acknowledgements.....	III
Abstract.....	IV

CHAPTER

CHAPTER 1: Introduction

1.1 Overview.....	1
1.2 Problems in existing system.....	1
1.3 Motivation.....	1
1.4 Objective.....	2
1.5 Methodology.....	2
1.6 Benefits of the System.....	2
1.7 Summary.....	3

CHAPTER 2: Literature Review

2.1 Introduction.....	4
-----------------------	---

2.2 Research Paper Review.....	5
2.3 Software Development Process Model.....	6
2.3.1 The Spiral model.....	6
2.4 N-tier Architecture.....	8
2.4.1 The One and Two Tier.....	8
2.4.2 Three - Tier and more.....	9

CHAPTER 3: Requirement Analysis

3.1 Overview.....	12
3.2 Use Case Model of POS.....	12
3.3 Use Case Scenario.....	13
3.3.1 Sale Goods.....	13
3.3.2 Purchase Goods.....	14
3.3.3 Create Supplier.....	15
3.3.4 Create Category.....	16
3.3.5 Create Product.....	17
3.3.6 Create Customer.....	18
3.3.7 Create User Account.....	19
3.4 System Requirements.....	20

CHAPTER 4: Design and Model of POS

4.1 Object Model.....	21
4.2 Database Model (ERD).....	22
4.3 GUI Model.....	23
4.3.1 Login UI.....	25
4.3.2 User Account UI.....	26
4.3.3 Category UI.....	27
4.3.4 Product UI.....	28
4.3.5 Supplier UI.....	29
4.3.6 Purchase UI.....	30
4.3.7 Customer UI.....	31
4.3.8 Sales UI.....	32

CHAPTER 5: Implementation and Testing

5.1 Design Implementation of Models.....	35
5.2 Implementation of ERD.....	36
5.2.1 Customer Table Schema.....	36
5.2.2 Sales Table Schema.....	37
5.2.3 Sales Detail Table Schema.....	37
5.2.4 Product Table Schema.....	38
5.2.5 Category Table Schema.....	39
5.2.6 Stock Table Schema.....	39

5.2.7 Supplier Table Schema.....	40
5.2.8 Purchase Table Schema.....	40
5.2.9 Purchase Detail Table Schema.....	41
5.3 Implementation of GUI and Report Generation.....	41
5.4 Integration and Testing.....	43
5.4.1 Unit Test	43
5.4.2 Integration Test.....	44
 CHAPTER 6: Future Scope & Conclusion	
6.1 Future Scope.....	46
6.2 Conclusion.....	46
 APPENDIX	 47
REFERENCES	51

LIST OF FIGURES	PAGE
Figure 2.3.1: The Spiral Model.....	7
Figure 2.4.2(a): Three-Tier Architecture.....	9
Figure 2.4.2(b): More than three tier architecture components.....	10
Figure 3.2: Use Case Model for POS.....	13
Figure 4.1: Object Model.....	21
Figure 4.2: Database Model.....	22
Figure 4.3(a): Front page of system interface.....	23
Figure 4.3(b): System Login.....	24
Figure 4.3(c): Manager Home Page.....	24
Figure 4.3(d): Cashier Home Page.....	25
Figure 4.3.1: Login UI.....	25
Figure 4.3.2: User Account UI.....	26
Figure 4.3.3: Category UI.....	27
Figure 4.3.4: Product UI.....	28
Figure 4.3.5: Supplier UI.....	29
Figure 4.3.6: Purchase UI.....	31
Figure 4.3.7: Customer UI.....	32
Figure 4.3.7: Sales UI.....	33
Figure 5.1: High Level Architecture.....	35

Figure 5.3(a): Sales Voucher report.....	42
Figure 5.3(b): Sales Pie Chart of a month.....	42

LIST OF TABLES	PAGE
Table 5.2.1 Customer Table Schema.....	37
Table 5.2.2 Sales Table Schema.....	37
Table 5.2.3 Sales Detail Table Schema.....	38
Table 5.2.4 Product Table Schema.....	38
Table 5.2.5 Category Table Schema.....	39
Table 5.2.6 Stock Table Schema.....	39
Table 5.2.7 Supplier Table Schema.....	40
Table 5.2.8 Purchase Table Schema.....	40
Table 5.2.9 Purchase Detail Table Schema.....	41

Chapter 1

INTRODUCTION

1.1 Overview

A POS system is a computer software and hardware network that records sales. It solves a variety of operational and record-keeping headaches. The retail industries are now adopting new technologies. They want to make their business easier and more reliable to their customer. For this purpose in 1973 IBM first start Electronic Cash Register (ECR) after that graphical point of sale came in 1986. In 1992 the modern POS software starts its journey. There are some key requirement outcomes for modern POS they are high and consistent operating speed, reliability, easy to use, remote supportability, low cost, and rich functionality. Due to new features the POS system need to up-to-date with requirement and hardware. The POS system needs an architectural view to make its feature more feasible.

We tried to design our POS system from software engineering perspective. We proposed an Object Oriented Frame work where logical and conceptual model has been developed. We tried to enrich our system from the current needs of retail industries. In future we will try to cover more rich functionalities to make our system up to date with real world needs.

1.2 Problems in Existing System

The POS systems can be customized for every type of business. In the existing POS systems were typically installed on business computers as software packages on Windows computers. But the Operating system has been changed day by day and the hardware also changes the requirement of operating system. Due to upgrade of hardware and operating system the existing POS cannot work with a great speed which retailers actually need. Also because of power failure some systems fail to records the data for long time process. The existing system cannot meet with a speed which retailers actually need to increase their customer satisfaction. Some system do have the feature with reporting about daily or weekly or monthly over view of that store which very important information the retailers actually need to increase their business.

1.3 Motivation

We studied several papers and research about existing POS system. We found several limitations in existing system also most of them do not follow the actual software engineering framework. These very facts motivate us to develop a POS system which follows the conceptual and logical model from the perspective of software engineering. The architectural view implements object oriented framework using JAVA.

1.4 Objective

Our objective is to promote an enrich POS system where the transaction will be done very fast and the time for per customer service will be reduce than the existing POS system. Monitoring about the stock and the revenue of the store can easily be done. After each transaction the database will be updated very fast. Reporting system can give an overview about the store and there transaction. Fast transaction, monitoring and overview the store by reporting will grow the revenue and the customer service will be increase. Therefore the retailers will be benefited by using our system and a positive approach will come in their business.

1.5 Methodology

Under Revenue Principle of Accounting Point of Sale is a well know method for recording the transaction. This is a common method used in grocery stores or supper shop. The amount of revenue can be measured objectively and observe the company's further benefits. Here we are presenting a Point of Sale system which can store the data about sales and purchase for a supper shop and it also generate a report about that stock.

Depending on platform we chose java for our software development language so that under new operating system this software can run successfully. Hardware industry has been controlled by some common big company but still the software industry of POS is competitive [3]. The PCCD users are now moving on POS system. And existing system has been replaced by new POS system [1].

Under this situation our software can give an optimum solution for improved POS system. Our system architecture and development make our software different from other POS system. We used JAVA one of the widely used computer language which can support all operating system and our database system MySql is very user friendly. UIs have been developed using one of the popular IDE NetBeans which is very user friendly. To improve and control the stock we added reporting system where the system can give a report about the shop and the actual work to do.

1.6 Benefits of the System

By using our system the retailers will be benefitted. The main benefit of our system that it can handle transaction of store efficiently and the customer service time will be reducing. There are some other benefits the user can get they are :

- System designed to immediately record sales
- Maintain control of store even in absence of the owner

- Automated inventory control helps keep proper balance of stock
- Management is easier due to the reporting facility of the system

The architecture can be reused for any other POS software development. We attempt to develop and architecture of POS software where the general issue will be fulfill and the retails industry can get some new issue to develop their existing POS system.

1.7 Summary

We attempt to develop architecture of POS. We mainly focus on developing the architecture where the software engineering perspectives can be used. We tried to view our system from the retailer's side also where they are the main user of this system. Over all we tried to shows the real perspective of POS in retail industries.

Chapter 2

LITERATURE REVIEW

2.1 Introduction

Point of sale is systems which allow business transaction to be monitored and give the opportunity to monitor those tasks (such as monitoring stock, inventory managing) if done manually and time consuming. Point of sale also refers to collect the data and the customer payment information when the service brought or sold by the retailers. Using POS the retailers can keep eye on the stock to make sure is the inventory product is satisfying for the shop or not. A variety of devices are used for this system to keep track of the product. Computers, cash registers, optical and bar code scanners etc device are used in POS transaction.

All retailers know that well planed and merchandised counter areas will increase business and will give them more profit. On the other hand the consumers are becoming more demanding and the retailers are continuing to bring new shop experience for their consumers. Is therefore important to consider what new point of sale is best for their business. Point of sale can be viewed as an essential part of supporting or increase the sale of retail business.

Point of sale system keeps a more accurate accounting of the inventory. It identifies discrepancies between the computer record and the inventory on the shelves. These systems also record all of the information about the sales so that analyze can be done with present data and use it to make wise business decisions in retail business. Therefore retailers are the end users of POS systems, but the industry involves many other players.

What can POS System do?

Track all elements of each order, from payment to design details to delivery date, months. The system automatically records customer names, address if the customer registered or use credit card to purchase any products also what they brought and when. This information can be searched and selected to make targeted and timed mailings not only practical but easy.

Everything about the customer's purchase can be immediately called up from the system. This makes handling payments, returns and other transactions easy even if the customer has misplace the paper work.

The computer keeps the store's inventory, so the POS terminals attached to it automatically update inventory with every sale or purchase. The central office knows at all times how much stock has or reduces.

The system knows at what rate each item is selling at all times, making it easy to decide what quantity to reorder and which item is more demandable and which one put out of interest for the customer. There is a report can always be helpful to reorder the product when the retailers know which product actually need to purchase from supplier.

2.2 Research Paper Review

Retail industry now using information technology in wide range to improve their business also makes new experience for their customer. To provide a best output for this retailer need to take a proper decision when adopting new technology. There are some arguments to adopt new technology when dealer has to choose the best technology. At initial stage the software, hardware, peripheral device and staffing cost of PCCD (PC cash drawer) are lower than POS system. On other hand PCCD can handle more product, backend technology is stronger than POS and offer low cost for the basic system where POS have some problem to handle with large amount of product and the backup technology still under developing process also need to make sure the hardware and software up to date to manage the data. Due to lower cost need to serve per customer and fast transaction also improving the customer experience POS is father better than PCCD system. Initial cost of POS is higher than PCCD but speedy customer services and efficient operating system POS work 70% faster than PCCD. PCCD life time half of the POS system therefore the total utilization of POS system is higher than the PCCD. Therefore the POS facing some challenges if it can overcome the POS system can dominate the retail industry [1].

Retail is fast changing industry where POS now made a new part of profit for retail business. Under the changing environment information system is playing a crucial role in effective customer interaction and service. 90% of sales challenge now can be handled by effective POS system. According to survey 63% of the retailers plan to move on POS system and 40-42% of their IT budgets typically now spending on POS technology. Average life of POS around 7 years and 38% retailers want to move on POS within next 2 years. POS hardware market dominated by some major company but the software is highly competitive and windows and Linux based operating system getting preferences for POS. Although there are some challenges also need to overcome in this industry. Handling the functionality and the lack of information flow make this system complicated and the high training cost also integration with new application become harder. Understanding of POS life cycle can be a way to make new changes on POS. Implementation could be vital issue for POS system. Proper approaches and analysis need to implement this system. Methodology should be understandable and flexible. Periodic audits and re-engineering can be done following standardized approaches. The main purpose should be how to make efficient and improved POS system for retailers [2].

Depending on automation POS testing it is different from other system. POS system needs interaction of human also depending on application the UI can be Non standard or dynamic.

Multiple Interface and complex configuration of POS application and some third party software and hardware involvement all together make this system automation harder than other system automation. Understanding Technical architecture, analysis of the automation tool and the high level Understanding of the POS application is crucial. Keyword Driven framework, data parameterization, Early automation and scope of automation can make some significant change of this. Tools automation, Operating system analysis is required to upgrade the test automation process of POS. POS testing automation challenge can be overcome by adopting strong strategy and make this Retailing business field faster and safe [3].

2.3 Software Development Process Model

A software development process is a structure imposed on the development of a software product. To ensure the successful execution of a project, it is necessary to break down the project into multiple manageable tasks. Each task is performed in a series using processes. A process or a combination of multiple processes is required to complete each task. To ensure smooth progress of a software project, relevant processes are arranged and executed in sequence. Every software project is discrete with respect to its complexity, size and goals. Therefore, different process models are designed for different software projects. These process models provide approaches that decide the path of software development from the conceptualization of a project to its formal termination. Our system development follows the Spiral Model. The Spiral model is described below:

2.3.1 The Spiral model

Spiral Model is an evolutionary software development life cycle that combines the linear nature of the Waterfall model and the iterative nature of the Prototyping model. In 1988, Barry Boehm published a formal software system development "spiral model," which combines some key aspect of the waterfall model and rapid prototyping methodologies. According to Boehm, "the major distinguishing feature of the Spiral Model is that it creates a risk-driven approach to the software process rather than a primarily document-driven or code-driven process. It incorporates many of the strengths of other models and resolves many of their difficulties" [4].

The project life cycle is divided into phases, and each phase is executed in all of the iteration of the Spiral Model. The basic goal of using the Spiral Model is to define ways of eliminating risks in the design phase of our system. At the beginning the project is small. Gradually it gets bigger and richer as it passes the different processes and hence the phases iteratively. Consequently the risks are minimized and gets manageable. The spiral model is favored for large, expensive, and complicated projects.

Starting at the center, each turn around the spiral goes through several task regions:

- Determine the objectives, alternatives, and constraints on the new iteration.
- Evaluate alternatives and identify and resolve risk issues.
- Develop and verify the product for this iteration. ▪ Plan the next iteration.

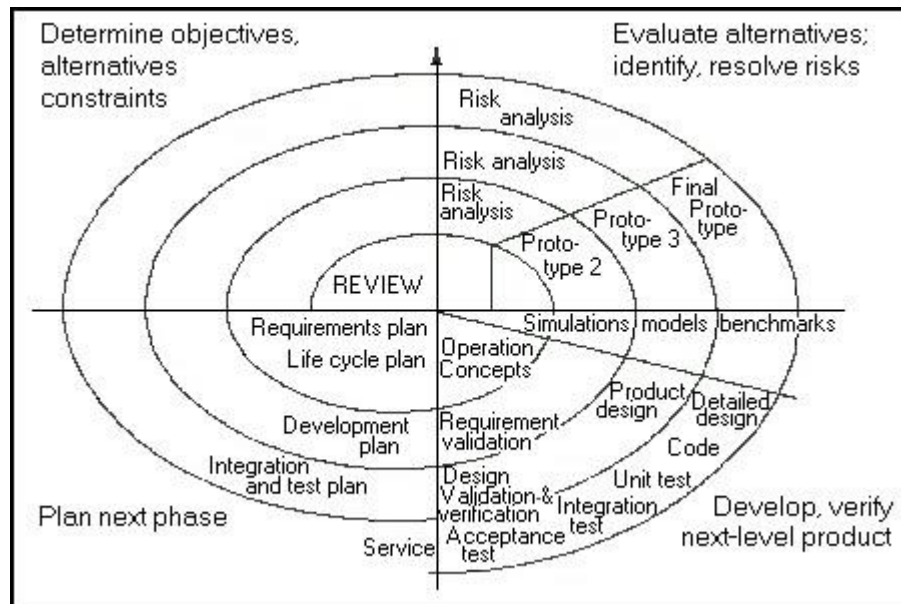


Figure 2.3.1: The Spiral Model

The steps in the spiral model can be generalized as follows:

1. The new system requirements are defined in as much detail as possible. This usually involves interviewing a number of users representing all the external or internal users and other aspects of the existing system.
2. A preliminary design is created for the new system.
3. A first prototype of the new system is constructed from the preliminary design. This is usually a scaled-down system, and represents an approximation of the characteristics of the final product.
4. A second prototype is evolved by a fourfold procedure: (1) evaluating the first prototype in terms of its strengths, weaknesses, and risks; (2) defining the requirements of the second prototype; (3) planning and designing the second prototype; (4) constructing and testing the second prototype.
5. At the customer's option, the entire project can be aborted if the risk is deemed too great. Risk factors might involve development cost overruns, operating-cost miscalculation, or any other factor that could, in the customer's judgment, result in a less-than-satisfactory final product.

6. The existing prototype is evaluated in the same manner as was the previous prototype, and, if necessary, another prototype is developed from it according to the fourfold procedure outlined above.
7. The preceding steps are iterated until the customer is satisfied that the refined prototype represents the final product desired.
8. The final system is constructed, based on the refined prototype.
9. The final system is thoroughly evaluated and tested. Routine maintenance is carried out on a continuing basis to prevent large-scale failures and to minimize downtime.

2.4 N-tier Architecture

In software engineering, multi-tier architecture (often referred to as n-tier architecture) is a client–server architecture in which presentation, application processing, and data management functions are logically separated. N-tier architecture is implemented in applications where a middleware is required to service data requests between a user and a database.

N-tier application architecture provides a model for developers to create a flexible and reusable application. By breaking up an application into tiers, developers only have to modify or add a specific layer, rather than have to rewrite the entire application, if they decide to change technologies or scale up. In the term "N-tier," "N" implies any number -- like 2-tier, or 3-tier, or 4-tier; any number of distinct tiers used in architecture.

2.4.1 The One and Two Tier

According to N-Tier Architecture and Tips By W.HU, 29 Aug 2012 [5]

One – Tier Architecture

The presentation, application and data storage layers can only run in one computer. In order to achieve 1-Tier, we need to use the embedded database system, which cannot run in an individual process. Otherwise, there will be at least 2-Tier because non-embedded databases usually can run in an individual computer (tier).

Two – Tier Architecture

Either presentation layer or application layer can only run in one computer, or application layer and data layer can only run in one computer. The whole application cannot run in more than 2 computers. *Advantages*

1-Tier and 2-Tier architecture is suitable for applications with lower number of user. It works fast and simple due deployment of lesser hardware and network bandwidth. Thus it reduces the cost of hardware, network and maintenance.

Disadvantage

This architecture is not suitable for applications with bigger number of user. It has limitation to solve issues like security, scalability, fault tolerance and etc because it can be deployed in only 1 or 2 computes.

The most widespread use of multi-tier architecture is the **three-tier architecture**. Most of the Pos architecture models follow the three-tier architecture.

2.4.2 Three - Tier and more

Three tiers might seem like separation of different layers:

- Three different physical layers, such as client, server and backend or mainframe. □
- Three logical layers, such as presentation, core application and data storage.

In most case these different layers may have sub-layers, which may overlap but don't match with each other.

Presentation	User Interface Layer (Client Side Presentation)	Client
	(Server Side) Presentation Layer	Server
Core Application	Business Layer (Session Layer)	
Data Storage	Entity Layer	
	Connectivity Layer	Backend
	Data Layer	

Figure 2.4.2(a): Three-Tier Architecture

Thus, both the presentation layer and the data storage layer are part of two physical systems: The presentation layer has a part on the client and a part on the server side. The data storage layer has a part on the server and a part on the backend side.

This architecture matches with modern architecture models such as the J2EE platform:

The server side of the presentation layer is the web server or web engine that combines the interface to the business logic with the elements of the (graphical) user interface. The server side of the data storage layer is typically a wrapper that hides the technical details of the backend (or enterprise information system) [6].

More than three layers of architecture

Some layers in 3-Tier can be broken further into more layers. These broken layers may be able to run in more tiers. For example, application layer can be broken into business layer, persistence layer or more. Presentation layer can be broken into client layer and client presenter layer. The layers are shown in the figure below:

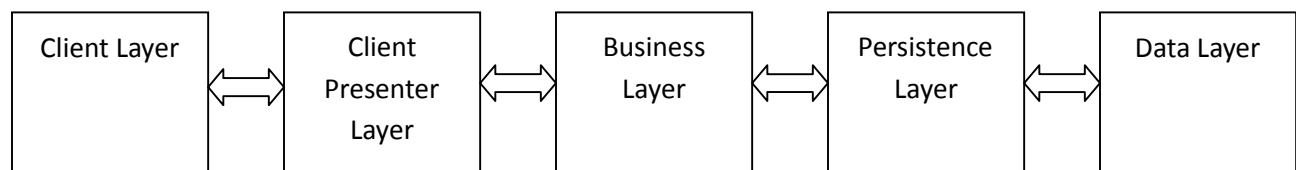


Figure 2.4.2(b): More than three tier architecture components

Advantages

By introducing the middle layer, the user is only handling presentation logic. This means that only little communication is needed between the client and the middle tier making the client “thin” or “thinner”.

Dividing the architecture into three layers has many advantages.

- For any changes in requirements or technology the three-tier architecture allow any of the three tiers to be upgraded or replaced independently. For example, a change of operating system in the presentation tier would only affect the user interface code.
- The application logic can be reused in other applications which results to quick development. Suppose a module handles adding, updating, deleting and finding customers in the system. As this component is developed and tested, we can use it in any other project that might involve maintaining customers.

- Changing a layer in the system is easy. Since the application logic is separate from the data access layer and the GUI layer, changing the data access layer won't affect the application logic module much.
- If there is a minor change in the application logic, we don't have to install the entire system in individual user's PCs.
- Security is ensured in the three-tier architecture because the middle layer protects the database tier.
- Having separate functionality servers allows for parallel development of individual tiers by application specialists.

Disadvantages

- More cost for hardware, network, maintenance and installation because more hardware and better network bandwidth are needed.
- The performance of the whole application may be slow if the hardware and network bandwidth aren't good enough because more networks, computers and processes are involved.

Chapter 3

REQUIREMENT ANALYSIS

3.1 Overview

The purpose of Requirements Analysis is to obtain a thorough and detailed understanding of the business or client's need and to break it down into discrete requirements. It is necessary to understand the purpose of the system. Our POS system is a computer system typically used at the front desk in retail stores to manage the sales and balance the stock. The users of a POS system are the employees of the store including cashiers and the administrator or the Manager. The Manager can access all the system management functions of the POS system but the access of few these functions is limited for the Cashier. The Cashier using the system does the sale transaction and may print receipt for customers when they buy product. The receipt is actually the sales report. The system can generate useful reports, which are mostly produced by the Manager. This system calculates stock balance by updating the decrease or increase of stock automatically during good purchase or sale transaction.

On the basis of the above purpose of the system, the requirements are analyzed. The use case analysis technique is used to identify the requirements of the POS system. This chapter will discuss the Use Case model of POS through the use case diagram. Later the use case scenario is given to give an elaborate description of the model. At last the system requirements are identified.

3.2 Use Case Model of POS

Use case is referred to the key functions of the system fulfilled by the user. The Use case model focuses on the use case (key functions), the users and interaction between them. The model can be shown graphically through the use case diagram. Use case diagram is the graphical modeling that shows the communication between actor and use case. The figure 3.2(a) shows the Use Case Diagram of our POS system. It shows the two active users of the system, namely the Manager and the Cashier. The Manager can access all the system management functions of the POS system, such as, create category, create product, create supplier, purchase goods, and create customer and sale goods. But the Cashier has only the access to create customer and sale goods.

Both the users can access to their respective functions through the Login.

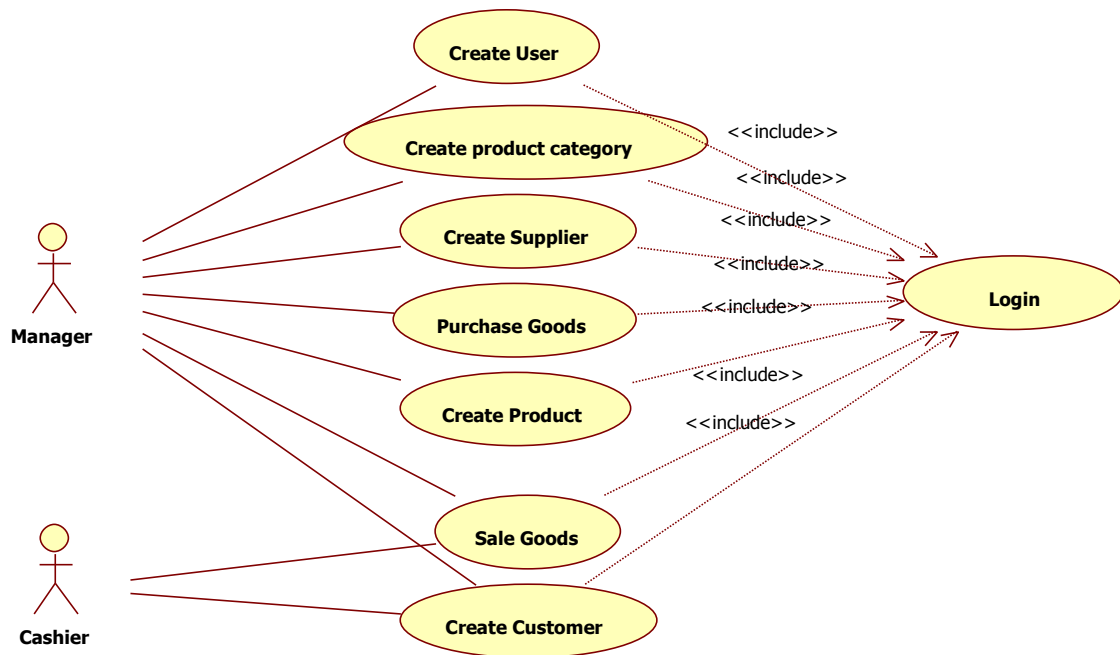


Figure 3.2: Use Case Model for POS

3.3 Use Case Scenario

Each use case provides a set of scenarios that convey how the system should interact with a human user or another system, to achieve a specific goal. Given below the step of actions of each use case.

3.3.1 Sale Goods

Actor: System

Pre-condition: Login

SAVE sale record Steps:

1. Generate and display sales ID
2. Generate and display Date and display
3. Input customer ID (Optional) and generate and display customer name.
4. Select Category
5. Select Product
6. Generate and display Price
7. Input Quantity of product
8. Click „Add Product“ button
9. Display added product in sales table
10. Generate and display Total, Total VAT and Grand Total
11. Input Paid amount
12. Auto calculate and display Change
13. Click „Save“ button

Exception

- 1.1 : Unable to „Add Product“ if Product not selected. Product must be selected
- 1.2 : Unable to „Add Product“, Quantity cannot be blank
- 10.1 : Paid input amount smaller than grand total, input amount greater than or equal to grand total

FIND sale record Steps:

1. Generate and display sales ID
2. Generate and display Date
3. Click „Find“ button
4. Input sales ID
5. Generate and display sales record table of the given sales ID
6. Enables “Delete” button

Exception

- 4.1: Incorrect input of sales ID, input correct and existing sales ID

Delete sale record

Pre-condition: Complete the FIND procedure Steps:

1. Click the „Delete“ button
2. Click „Yes“ button on the confirmation dialog to delete the sales record
3. Disables “Delete” button

Exception:

- 1.1 : System error cannot delete sales record

3.3.2 Purchase Goods

Actor: System

Pre-condition: Login

SAVE purchase record Steps:

1. Generate and display purchase ID
2. Generate and display Date and display
3. Input supplier ID
4. Generate and display supplier name.
5. Select Category
6. Select Product
7. Generate and display Price
8. Input Quantity of product
9. Click „Add Product“ button
10. Generate and display Grand Total
11. Input Paid amount
12. Auto calculate and display Change
13. Click „Save“ button

Exception

- 9.1: Unable to „Add Product“ if Product not selected. Product must be selected
- 9.2: Unable to „Add Product“, Quantity cannot be blank

- 11.1: Paid input amount smaller than grand total, input amount greater than or equal to grand total
13.1: Unable to save, supplier ID cannot be blank

FIND purchase record Steps:

1. Generate and display purchase ID
2. Generate and display Date
3. Click „Find“ button
4. Input purchase ID
5. Generate and display purchase record table of the given purchase ID
6. Enables “Delete” button

Exception

- 5.1: Incorrect input of purchase ID, input correct and existing purchase ID

Delete purchase record

Pre-condition: Complete the FIND procedure Steps:

1. Click the „Delete“ button
2. Click „Yes“ button on the confirmation dialog to delete the purchase record
3. Disables “Delete” button

Exception:

- 1.2 : System error cannot delete purchase record

3.3.3 Create Supplier

Actor: System

Pre-condition: Login

ADD supplier record Steps:

1. Input Name
2. Input „Address“, „Phone“, „Contact Person“ (Optional)
3. Click „Add“ button

Exception:

- 3.1: Cannot add supplier record. Name field cannot be blank

Find supplier record Steps:

1. Click „Find“ button
2. Input Supplier ID
3. Click „OK“
4. Generate and display supplier record of the given ID
5. Enables „Update“ and “Delete” button

Exception:

- 3.1: Wrong supplier ID, input a correct and existing supplier ID

condition: Complete the FIND procedure

Update supplier record Pre-

Steps:

1. Make required changes to the supplier record
2. Click the „Update“ button
3. Disables „Update“ and “Delete” button

Exception:

3.1: System error cannot update supplier record

Delete supplier record

Pre-condition: Complete the FIND procedure Steps:

4. Click the „Delete“ button
5. Click „Yes“ button on the confirmation dialog to delete the supplier record
6. Disables „Update“ and “Delete” button

Exception:

1.3 : System error cannot delete supplier record

3.3.4 Create Category

Actor: System

Pre-condition: Login

Save Category Steps:

1. Input Product „ID“
2. Input product „Name“
3. Click „Save“ button

Exception:

3.1: Unable to save category, fields cannot be blank

3.3: System Error, cannot save category

Find Category Steps:

1. Click „Find“ button
2. Input Category ID
3. Click „OK“
4. Generate and display product record of the given code
5. Enables „Update“ and “Delete” button

Exception:

3.1: Wrong Category ID, input a correct and existing Category ID

Update Category

Pre-condition: Complete the FIND procedure Steps:

1. Make required changes to the product record
2. Click the „Update“ button
3. Disables „Update“ and “Delete” button

Exception:

2.1: System error cannot update category record

Delete Category

Pre-condition: Complete the FIND procedure Steps:

1. Click the „Delete“ button
2. Click „Yes“ button on the confirmation dialog to delete the category record
3. Disables „Update“ and “Delete“ button

Exception:

1.4 : System error cannot delete category record

3.3.5 Create Product

Actor: System

Pre-condition: Login

Add product Steps:

1. Input Product „Code“
2. Input product „Name“
4. Input product „Price“
5. Select product „Category“
6. Click „Add“ button

Exception:

5.1: Unable to add product, fields cannot be blank

5.2: System Error, cannot add product

Find product Steps:

1. Click „Find“ button
2. Input Product code
3. Click „OK“
4. Generate and display product record of the given code
5. Enables „Update“ and “Delete“ button

Exception:

3.1: Wrong product code, input a correct and existing product code

Update product

Pre-condition: Complete the FIND procedure Steps:

1. Make required changes to the product record
2. Click the „Update“ button
3. Disables „Update“ and “Delete“ button

Exception:

2.1: System error cannot update product record

condition: Complete the FIND procedure

Delete product Pre-

Steps:

1. Click the „Delete“ button
2. Click „Yes“ button on the confirmation dialog to delete the product record
3. Disables „Update“ and “Delete” button

Exception:

1.5 : System error cannot delete product record

3.3.6 Create Customer

Actor: System

Pre-condition: Login

Add customer record Steps:

1. Input „Name“
2. Input „Address“
3. Input „Mobile“
4. Input „email“
5. Select product „Gender“
6. Click „Add“ button

Exception:

6.1: Unable to add customer record, „Name“ cannot be blank

6.2: System Error, cannot add Customer

Find customer record Steps:

1. Click „Find“ button
2. Input customer ID
3. Click „OK“
4. Generate and display customer record of the given ID
5. Enables „Update“ and “Delete” button

Exception:

3.1: Wrong customer ID, input a correct and existing customer ID

Update customer record

Pre-condition: Complete the FIND procedure Steps:

1. Make required changes to the customer record
2. Click the „Update“ button
3. Disables „Update“ and “Delete” button

Exception:

2.1 : System error cannot update customer record

condition: Complete the FIND procedure

Delete customer record Pre-

Steps:

1. Click the „Delete“ button
2. Click „Yes“ button on the confirmation dialog to delete the customer record
3. Disables „Update“ and “Delete” button

Exception:

1.1 : System error cannot delete customer record

3.3.7 Create User Account

Actor: System

Pre-condition: Login

Add User account Steps:

1. Input „User ID“
2. Input „Password“
3. Click „Create User“ button

Exception:

3.1: Unable to create User account, „ID“ and „Password“ cannot be blank

3.2: System Error, cannot add product

Find User account Steps:

1. Click „Find“ button
2. Input User ID
3. Click „OK“
4. Generate and display User account of the given ID
5. Enables „Update“ and “Delete” button

Exception:

3.1: Wrong User ID, input a correct and existing User ID

Update User account

Pre-condition: Complete the FIND procedure Steps:

4. Make required changes to the User account
5. Click the „Update“ button
6. Disables „Update“ and “Delete” button

Exception:

2.2 : System error cannot update User account

Delete User account Pre-

condition: Complete the FIND procedure

Steps:

4. Click the „Delete“ button
5. Click „Yes“ button on the confirmation dialog to delete the User account
6. Disables „Update“ and “Delete” button

Exception:

1.2 : System error cannot delete User account

3.4 System Requirements

The system requirement for our project is JAVA as front-end and MySql as back-end. The tools and application required to develop the system is given below:

Front-end: JAVA

- IDE: Netbeans 6.9.1 Back-end: MySQL
- Database Administration Tool: Navicat

8.0 Lite Report and Chart Generation

- Reporting Engine: JasperReport Library

□ Report Designer: iReport 4.1.1

Chapter 4

DESIGN AND MODEL OF POS

This chapter shows the design of different models and brief analysis of each model. Models include the Object Model, Database Model (ERD), and GUI Model.

4.1 Object Model

The POS system may require to be modified according to the needs of the client. By creating classes and defining their objects the data can be organized and hence can be easily modified. Our POS system uses Object Oriented Programming (OOP) in Java. Before implementing OOP, the object model was designed. It is the most important and classic model in OO analysis.

An **Object Model** also called a **domain model** is a representation of real-world conceptual class. It is used as a source of inspiration for designing software objects. An Object model is the most important artifact created during object-oriented analysis. It shows domain objects or conceptual classes, associations between conceptual classes, attributes of conceptual classes. It illustrates the attributes and methods of the classes.

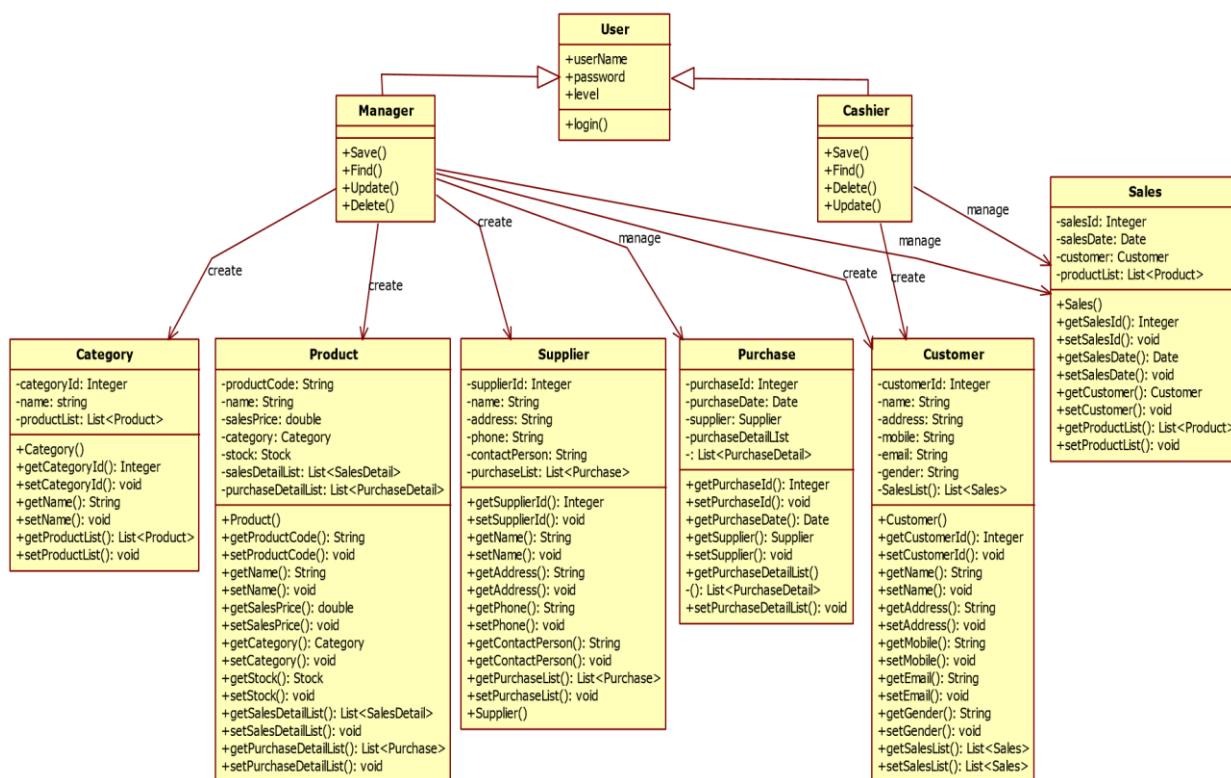


Figure 4.1: Object Model

Figure 4.1 shows the design of the Object Model of our POS system. Here, the Manager and Cashier class inherit the User class. The cashier is associated with „manage“ Sales class and „creates“ Customer class. For an example, the Sales class has attributes: salesId, salesDate etc and methods like, getSalesId(), setSalesDate() etc. the figure shows all the classes and their association with other classes.

4.2 Database Model (ERD)

We created a database named **stock**. We have nine tables where each table has relation with another table. The **constraints** enforces different rules the data has to pass. We normalized the sales table and made **sales** and **sales_deatil** table so that data will not redundant. We also normalized the purchase table and made **purchase** and **purchase_detail** so that the same data is not repeated. We made composite key to uniquely identify **sales_detail** and **purchase_detail** table. Our proposed database model for POS Figure 4.2(a) is given below:

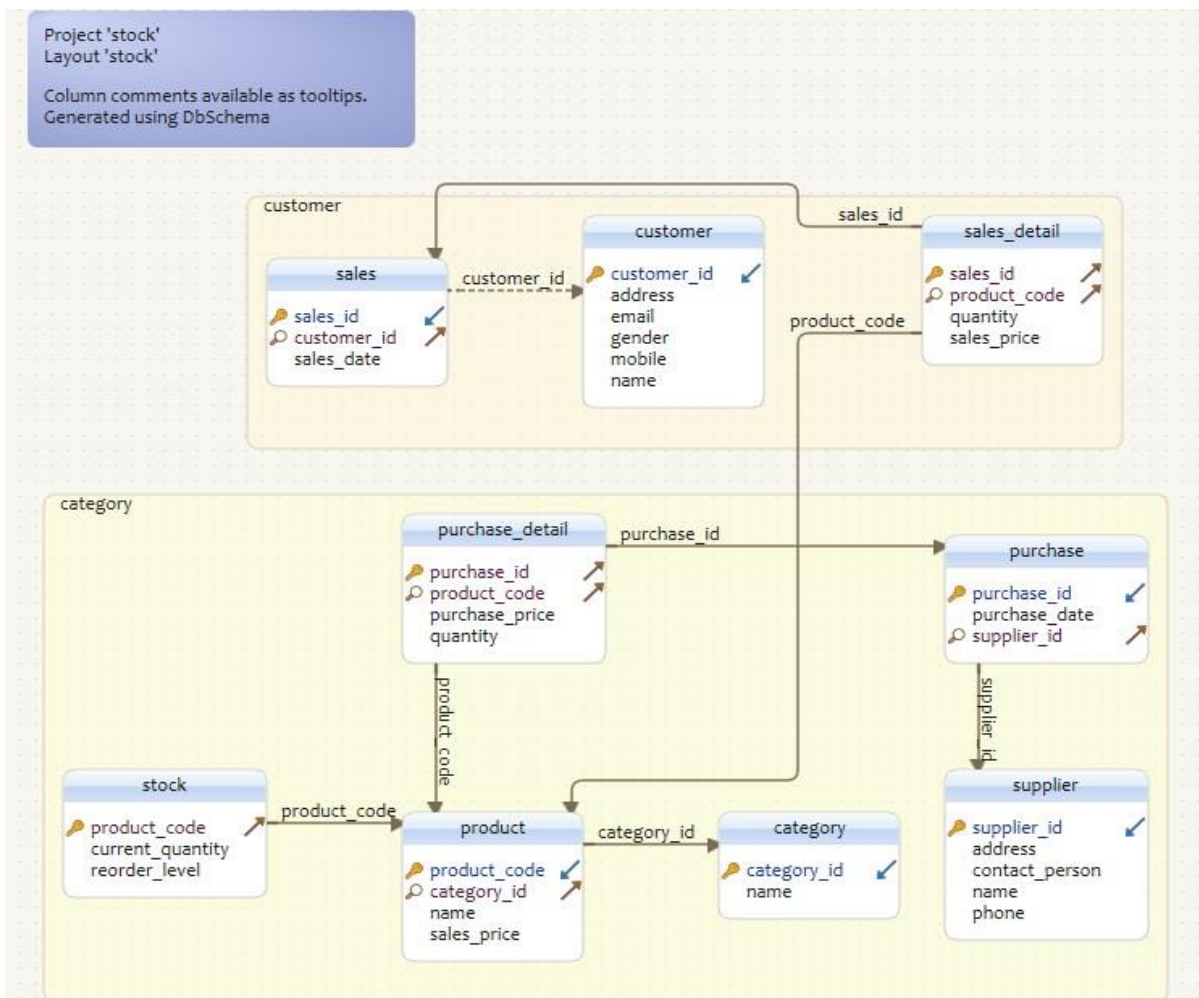


Figure 4.2: Database Model

4.3 GUI Model

Graphical User Interface (GUI) is an important part of a POS system. It allows the users (Manager or Cashier) to interact with the system using visual widgets rather than text commands. GUI is the logical design of the source codes of different functions in the system. The main aim of it is to enhance the efficiency and ease of use of these designs by the user and hence let the user achieve its goal of action. The GUIs represents the information and actions available to a user through graphical icons, typed command labels or text navigation. The actions are usually performed through direct manipulation of the graphical elements. The different GUI forms of our POS system are described below.

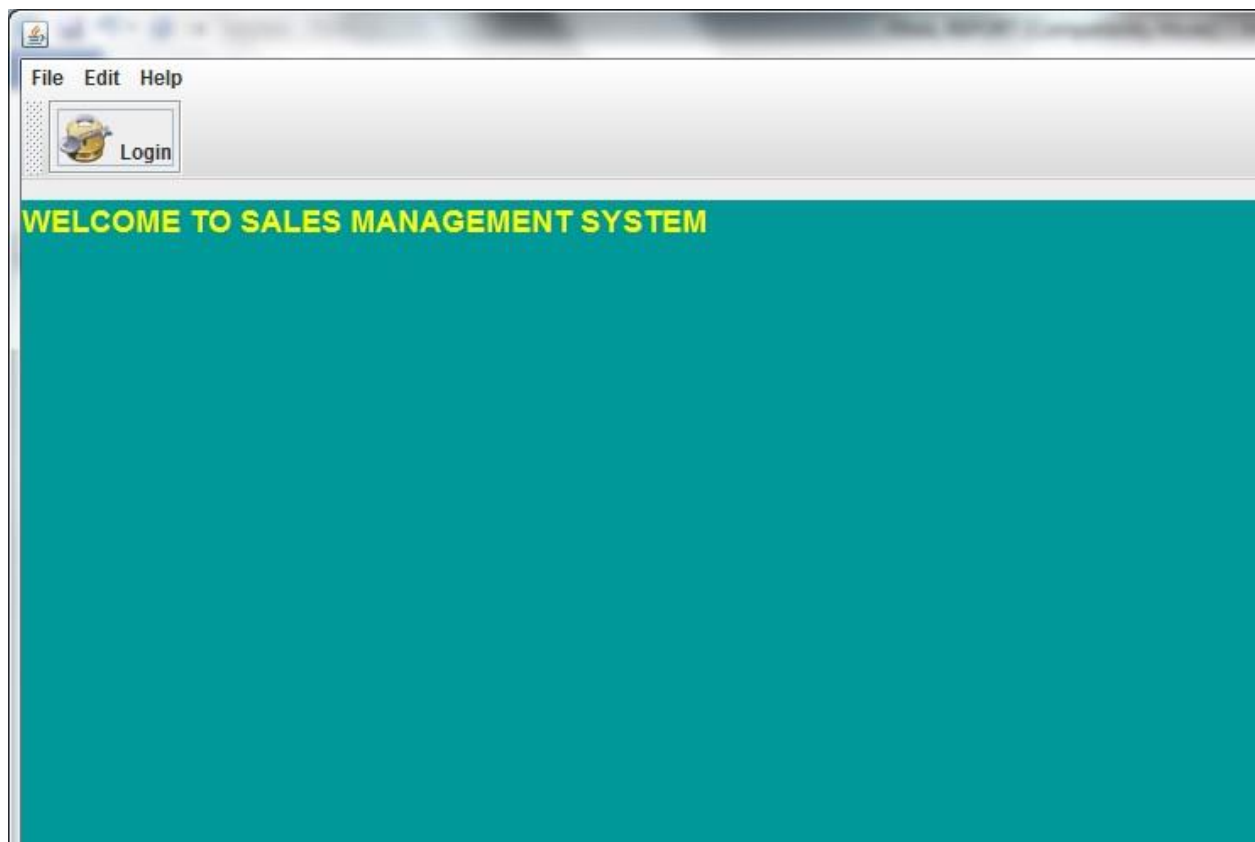


Figure 4.3(a): Front page of system interface

Anyone entering the system will see its front page first. The figure shows the front page of our system. The front page only consist of the „Login“ menu.

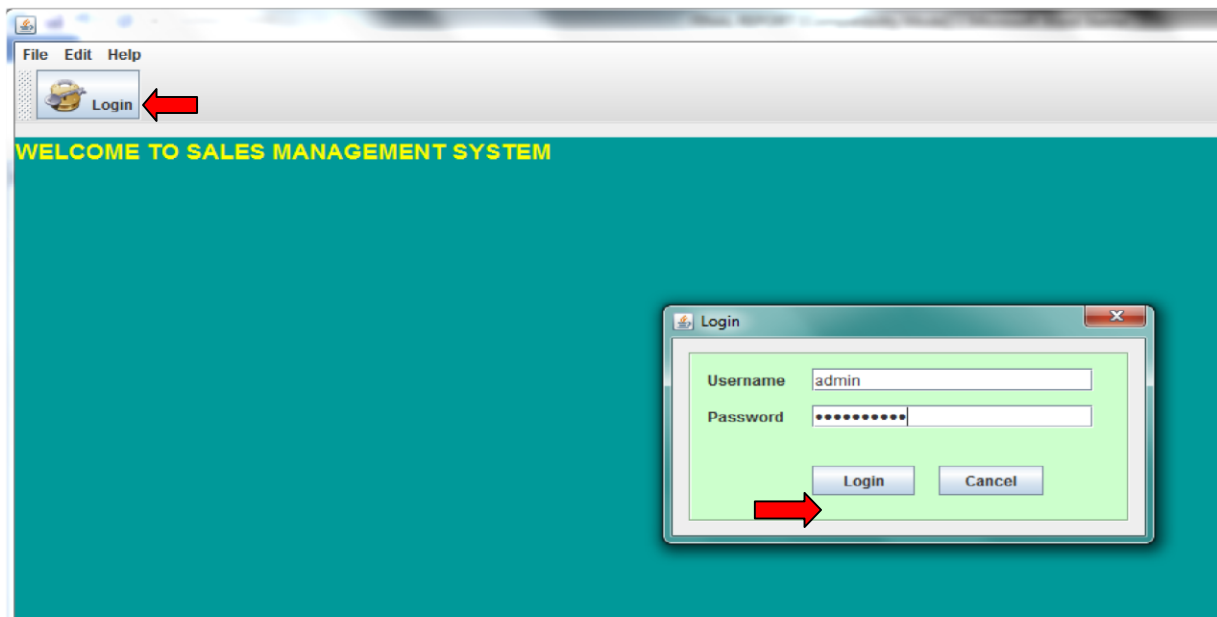


Figure 4.3(b): System Login

To enter the system, user has to login by clicking on the „Login“ menu. A login form appears on the screen. User has to input username and password and click the „Login“ button.

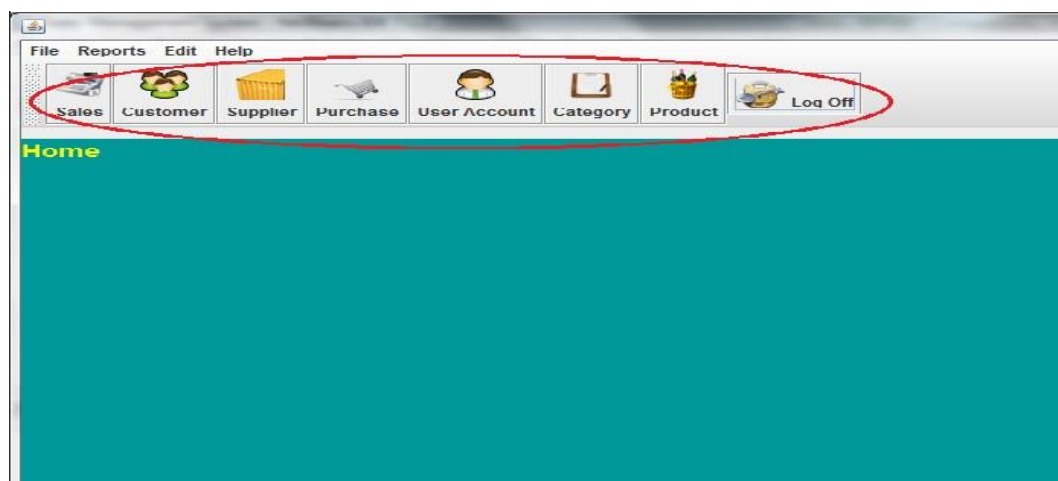


Figure 4.3(c): Manager Home Page

The figure shows the Home page for the Manager Login. Manager can access all the functions of the system from the menu items.

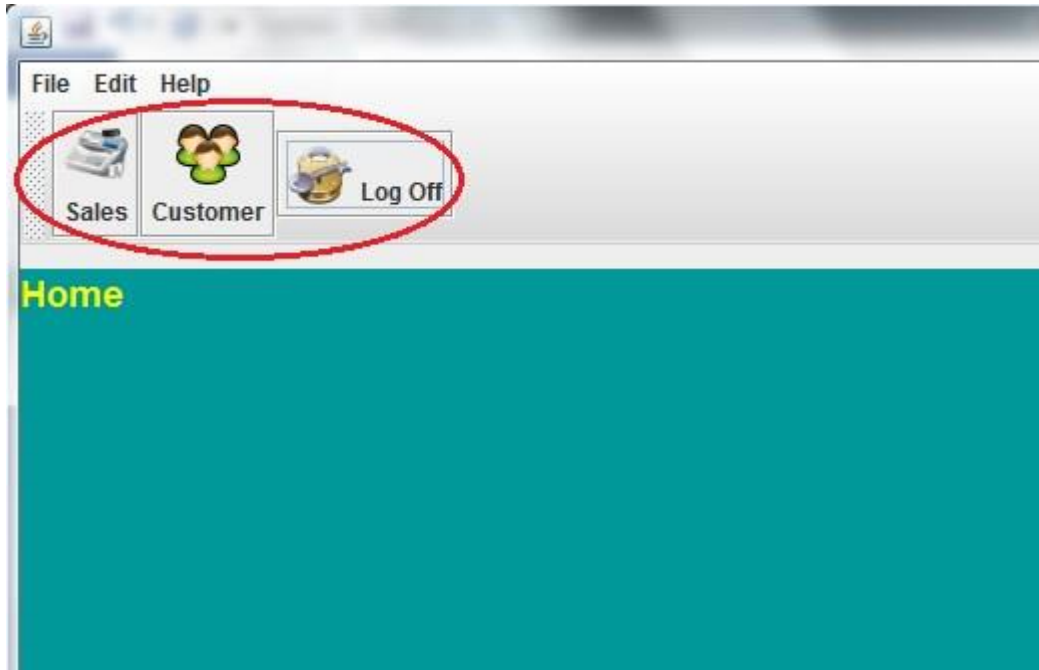


Figure 4.3(d): Cashier Home Page

The figure shows the Home page for the Cashier Login. Cashier can access only the „Sales“ and „Customer“ UI of the system from the menu items.

4.3.1 Login UI:

A user must log in to use the POS. At the beginning of entering the system, authorized users (Manager or Cashier) must first login through the Login UI. User must enter his/her user name in the „Username“ field and enter the password in the „Password“ field. Then click the „Login“ button. If the username and password is correct then the home page of the system is displayed.

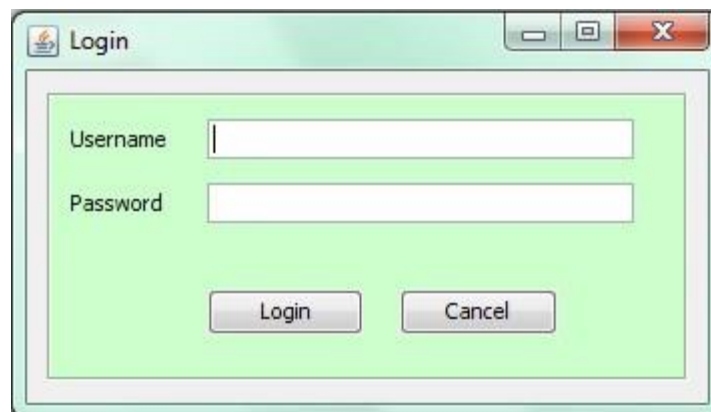


Figure 4.3.1: Login UI

4.3.2 User Account UI

The User Account UI creates a User. It is used by the Manager. It can **CREATE ACCOUNT**, **FIND**, update and **DELETE** user records. To create a user input User ID and password in the „User ID“ and „Password“ fields respectively. These two fields cannot be blank. Click the „**CREATE ACCOUNT**“ button. A message will appear showing “user account has been created successfully” if saving is successful otherwise unsuccessful due to system error.

A user account can be searched by the user ID. To find a user account, first click the „**FIND**“ button. Then input the user ID in the input dialog box. If the given ID is correct then it displays the account in the UI otherwise shows a message for wrong input of user ID.



Figure 4.3.2: User Account UI

An existing account can be modified using the „**Update**“ button. But before that the desired account must be displayed in the UI using the „Find“ procedure. Making the required changes, click the „Update“ button. A message will appear showing “user account has been updated successfully” if update is successful otherwise unsuccessful due to system error.

A user account can be deleted. First the „Find“ procedure is done and then click the „**DELETE**“ button. A message will appear showing “user account has been deleted successfully” if delete is successful otherwise unsuccessful due to system error.

4.3.3 Category UI

The Category UI creates a category. It is used by Manager. It can save, find, update and delete category records.

To save a category input category ID and name in their respective fields. Then click the **'SAVE'** button. A message will appear showing “category record has been saved successfully” if save is successful otherwise unsuccessful message due to system error.

A category record can be searched by the category ID. To find a category record, first click the **'FIND'** button. Then input the category ID in the input dialog box. If the given ID is correct then it displays the record in the UI otherwise shows a message for wrong input of category ID.

An existing record can be modified using the **'UPDATE'** button. But before that the desired record must be displayed in the UI using the **'Find'** procedure. Making the required changes, click the **'Update'** button. A message will appear showing “category record has been updated successfully” if update is successful otherwise unsuccessful due to system error.

A category record can be deleted. First the **'Find'** procedure is done and then click the **'DELETE'** button. A message will appear showing “category record has been deleted successfully” if delete is successful otherwise unsuccessful due to system error.

The image shows a software window titled "Category". Inside the window, there is a green rectangular area containing three input fields. The first field is labeled "Category ID:" and contains the number "12". The second field is labeled "Category name:" and is empty. The third field is labeled "VAT %" and is empty. Below these fields, there is a horizontal bar containing five buttons: "Add", "Find", "Update", "Delete", and "Clear". The window has a standard Windows-style title bar with minimize, maximize, and close buttons.

Figure 4.3.3: Category UI

4.3.4 Product UI

The Product UI creates a product. It is used by Manager. It can add, find, update and delete product records.

To add a product input the product code in the „code“ field. Code field cannot be blank. Input name, price, select category and input reorder Level in their respective fields. Then click the **‘ADD’** button. A message will appear showing “product record has been added successfully” if add is successful otherwise unsuccessful message due to system error.

A product record can be searched by the product CODE. To find a product record, first click the **‘FIND’** button. Then input the product CODE in the input dialog box. If the given CODE is correct then it displays the record in the UI otherwise shows a message for wrong input of product CODE.

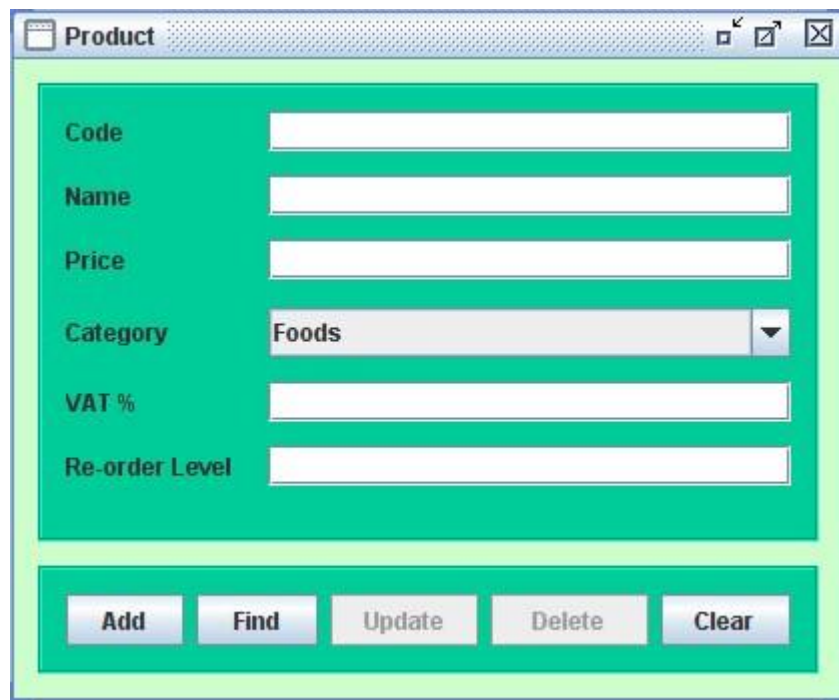
The image shows a software window titled "Product". Inside the window, there is a green rectangular area containing several input fields. The fields are labeled "Code", "Name", "Price", "Category", "VAT %", and "Re-order Level". The "Category" field is a dropdown menu with "Foods" selected. Below the green area, there is a row of five buttons: "Add", "Find", "Update", "Delete", and "Clear". The window has a standard title bar with minimize, maximize, and close buttons.

Figure 4.3.4: Product UI

An existing record can be modified using the **‘UPDATE’** button. But before that the desired record must be displayed in the UI using the „Find“ procedure. Making the required changes, click the **‘Update’** button. A message will appear showing “product record has been updated successfully” if update is successful otherwise unsuccessful due to system error.

A product record can be deleted. First the **‘Find’** procedure is done and then click the **‘DELETE’** button. A message will appear showing “product record has been deleted successfully” if delete is successful otherwise unsuccessful due to system error.

4.3.5 Supplier UI

The Supplier UI creates a supplier. It is used by Manager. It can add, find, update and delete supplier records.

To add a supplier input name in the „Name“ field. Name field cannot be blank. Input address, phone and contact person in their respective fields. Then click the **‘ADD’** button. A message will appear showing “supplier record has been added successfully” if add is successful otherwise unsuccessful due to system error.

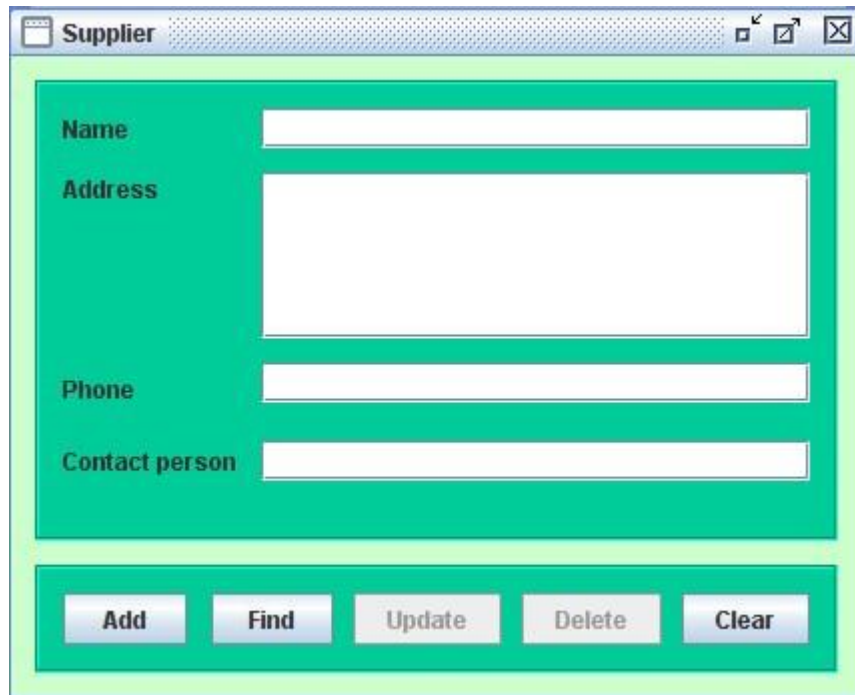
The image shows a graphical user interface window titled "Supplier". The window has a standard title bar with minimize, maximize, and close buttons. The main content area has a green background. It contains four text input fields stacked vertically, labeled "Name", "Address", "Phone", and "Contact person". Below these fields is a horizontal row of five buttons: "Add", "Find", "Update", "Delete", and "Clear". The buttons are light blue with black text.

Figure 4.3.5: Supplier UI

A supplier record can be searched by the supplier ID. To find a supplier record, first click the **‘FIND’** button. Then input the supplier ID in the input dialog box. If the given ID is correct then it displays the record in the UI otherwise shows a message for wrong input of supplier ID.

An existing record can be modified using the **‘UPDATE’** button. But before that the desired record must be displayed in the UI using the „Find“ procedure. Making the required changes, click the **‘UPDATE’** button. A message will appear showing “supplier record has been updated successfully” if update is successful otherwise unsuccessful due to system error.

A supplier record can be deleted. First the **‘Find’** procedure is done and then click the **‘DELETE’** button. A message will appear showing “supplier record has been deleted successfully” if delete is successful otherwise unsuccessful due to system error.

4.3.6 Purchase UI

The Purchase UI creates a purchase. It is used by Manager. It can add, find, update and delete purchase records. The purchase ID and Date is generated automatically.

To add a purchase, input supplier ID in the „Supplier ID“ field. Supplier ID field cannot be blank. Select Category, Product in their respective combo boxes. Input the price and quantity of the product in their respective fields. Click the **‘ADD PRODUCT’** button. Every time a product is added it is displayed in the purchase table of the UI. The table displays the product code, name, quantity and price and calculates the total and displays it. To edit any of the added products from the table, user can click on the product row. The selected product’s detail is transferred to the respective fields. Making the required changes **‘ADD BUTTON’** is clicked again. The refreshed table is displayed. The grand total price of all the products selected is calculated and displayed in the Grand Total field every time the user clicks the Add Product button. Input the paid amount which will show the change to be given. Then the **‘SAVE’** is enabled. Click the **‘SAVE’** button to save the purchase record. A message will appear showing “purchase record has been saved successfully” if save is successful otherwise unsuccessful due to system error.

A purchase record can be searched by the purchase ID. To find a purchase record, first click the **‘FIND’** button. Then input the purchase ID in the input dialog box. If the given ID is correct then it displays the record in the UI otherwise shows a message for wrong input of purchase ID.

For correct input, the UI’s fields are filled with the selected ID’s information.

A purchase record can be deleted. First the „Find“ procedure is done and then click the **‘DELETE’** button. A message will appear showing “purchase record has been deleted successfully” if delete is successful otherwise unsuccessful due to system error.

Figure 4.3.6: Purchase UI

4.3.7 Customer UI

The Customer UI creates a customer. It is used by the Manager and the Cashier. It can **ADD**, **FIND**, update and **DELETE** customer records.

To add a customer input name in the „Name“ field. Name field cannot be blank. Input address, mobile, email and gender in their respective fields. Then click the „**ADD**“ button. A message will appear showing “customer record has been added successfully” if add is successful otherwise unsuccessful due to system error.

A customer record can be searched by the customer ID. To find a customer record, first click the „**FIND**“ button. Then input the customer ID in the input dialog box. If the given ID is correct then it displays the record in the UI otherwise shows a message for wrong input of customer ID.

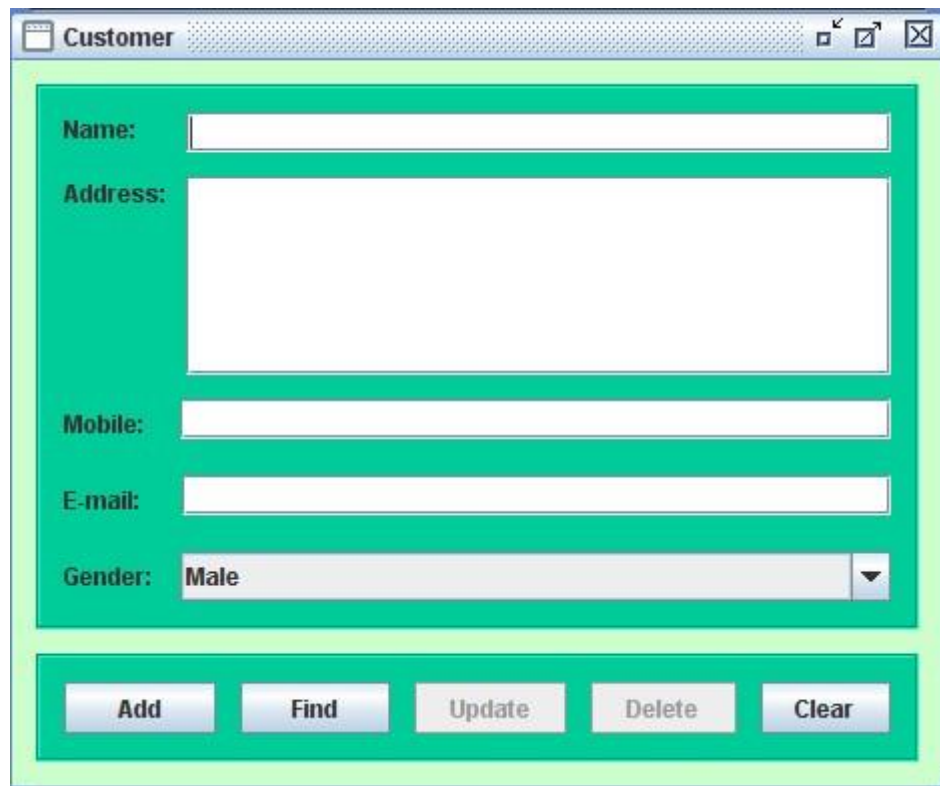
A screenshot of a software window titled "Customer". The window has a standard Windows-style title bar with minimize, maximize, and close buttons. The main content area has a light green background. It contains five input fields: "Name:" (a single-line text box), "Address:" (a multi-line text box), "Mobile:" (a single-line text box), "E-mail:" (a single-line text box), and "Gender:" (a dropdown menu currently showing "Male"). At the bottom of the form, there is a horizontal bar with five buttons: "Add", "Find", "Update", "Delete", and "Clear".

Figure 4.3.7: Customer UI

An existing record can be modified using the „**Update**’ button. But before that the desired record must be displayed in the UI using the „Find” procedure. Making the required changes, click the „Update” button. A message will appear showing “customer record has been updated successfully” if update is successful otherwise unsuccessful due to system error.

A customer record can be deleted. First the „Find” procedure is done and then click the “**DELETE**” button. A message will appear showing “customer record has been deleted successfully” if delete is successful otherwise unsuccessful due to system error.

4.3.7 Sales UI

The most basic function of a POS system is to handle sales. When a customer arrives at a POS counter with goods to purchase, the cashier will start a new sale transaction. The Sale UI creates a sale. It is used by Cashier, the Manager has also access to it. It can add, find, update and delete sale records. The sale ID and Date is generated automatically.

The Sales UI window is divided into several sections. The top section contains input fields for Sales ID (1210190002), Date (Oct 19, 2012), Customer ID, and Customer name. To the right, there are dropdown menus for Category and Products, and input fields for Price and Quantity. An 'Add Product' button is located below these fields. The middle section features a table with the following columns: Product Code, Name, Quantity, Price, and Total. A 'Remove' button is positioned to the right of the table. The bottom section contains input fields for Total, Total Vat, Grand Total, Paid, and Change. At the very bottom, there are buttons for Save, Find, Delete, Clear, and Close.

Figure 4.3.7: Sales UI

To add a sale, input customer ID in the „Customer ID“ field. The name of the customer will appear in the „Name“ field. Select Category, Product in their respective combo boxes. Selecting a product will automatically display the price. Input the quantity of the product in the „Quantity“ field. To add the product, quantity field cannot be blank. Click the **‘ADD PRODUCT’** button. Every time a product is added it is displayed in the sales table of the UI. The table displays the product code, name, quantity and price and calculates the total and displays it. To edit any of the added products from the table, user can click on the product row. The selected product’s detail is transferred to the respective fields. Making the required changes **‘ADD BUTTON’** is clicked again. The refreshed table is displayed. The total price (excluding VAT), total VAT amount, grand total price (including VAT) of all the products selected is calculated and displayed in their fields every time the user clicks the Add Product button. Input the paid amount which will show the change to be given. Then the **‘SAVE’** is enabled. Click the **‘SAVE’** button to save the purchase record. A

message will appear showing “purchase record has been saved successfully” if save is successful otherwise unsuccessful due to system error.

A sale record can be searched by the sale ID. To find a sale record, first click the ***FIND*** button. Then input the sale ID in the input dialog box. If the given ID is correct then it displays the record in the UI otherwise shows a message for wrong input of sale ID. For correct input, the UI’s fields are filled with the selected ID’s information.

A sale record can be deleted. First the „Find“ procedure is done and then click the ***DELETE*** button. A message will appear showing “sale record has been deleted successfully” if delete is successful otherwise unsuccessful due to system error.

Chapter 5

IMPLEMENTATION AND TESTING

5.1 Design and Implementation of Models

POS software usually implements the three-tier (layer) architecture in which the user interface, application logic and data storage and data access are developed and maintained as independent modules or most often on separate platforms. The three-tier architecture is intended to allow any of the three tiers to be upgraded or replaced independently in response to changes in requirements of technology. Basically, there are 3 layers, tier 1 (presentation tier, GUI tier), tier 2 (application logic tier) and tier 3 (data access tier). These tiers can be developed and tested separately.

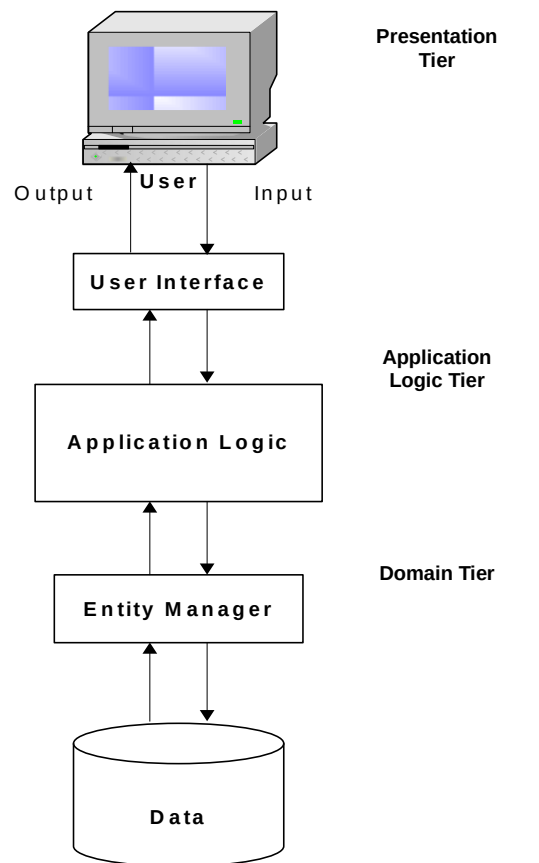


Figure 5.1: High Level Architecture

Presentation Tier

The top most level of the application is the user interface. The main task of this layer is to receive user input and display the output. The main function of the interface is to translate the tasks and results to something the user can understand.

Application Logic Tier

This layer co-ordinates the application, processes commands, makes logical decisions and evaluations and performs calculations. It also moves and processes data between the two surrounding layers.

Data Tier

This tier consists of database servers. Here information is stored and retrieved. The information is then passed back to the logic tier for processing and eventually back to the user. This tier keeps data neutral and independent from application servers or business logic. Giving data on its own tier also improves scalability and performance.

5.2 Implementation of ERD

A database schema is a collection of meta-data that describes the relations in a database. A schema can be simply described as the "layout" of a database or the blueprint that outlines the way data is organized into tables. Schema are normally described using Structured Query Language as a series of CREATE statements that may be used to replicate the schema in a new database.

Databases are software program to store program for storing data. In database data are stored in a **Table**. 'stock' named database has been created for this application. In 'stock' database data are organized into table.

The schema for this application has been described by the following process:

5.2.1 Customer Table Schema

customer table store the data about the customer. Customer buy products from shop and at the same time cashier sell product to the customer and save the customer data in data base. The **customer_id** is the number which uniquely identifies each customer in database **stock**. The **Table 5.2.1** shows the variable types with a specific precision of **customer** table. The **constraints** enforces different rules the data has to pass. Here primary key (pk) **customer_id** pass information about the customer in **sales** table.

Table 5.2.1: Schema of customer Table

customer	
customer_id	INT NOT NULL AUTOINCREMENT
address	VARCHAR(500)
email	VARCHAR(100)
gender	VARCHAR(1)
mobile	VARCHAR(15)
name	VARCHAR(100)
Indexes	
pk_customer primary key	ON customer_id

5.2.2 Sales Table Schema

sales table creates the data about sales. Here **sales_id** is the unique identification number of each sale. The **Table 5.2.2** column shows the data type with a specific precision. Here the foreign key (fk) **customer_id** refers the customer table data.

Table 5.2.2: Schema of sales Table

sales	
sales_id	INT NOT NULL
customer_id	INT
sales_date	DATE NOT NULL
Indexes	
pk_sales primary key	ON sales_id
fk_sales_customer_id	ON customer_id
Foreign Keys	
fk_sales_customer_id	(customer_id) ref customer (customer_id)

5.2.3 Sales Detail Table Schema **sales_detail** is the child table of **sales** table. **Table 5.2.3** column shows the data type with a specific precision. To eliminate data redundancy and „many to many“ relations this table has been created as a normalized form of sales table. **sales_detail** table contains composite primary key to uniquely identify the sales detail data. **sales_detail** Table has a relation with **product** Table via the **product_code** (fk).

Table 5.2.3: Schema of sales_detail Table

sales_detail	
sales_id	INT NOT NULL
product_code	VARCHAR(20) NOT NULL
quantity	INT
sales_price	DOUBLE
Indexes	
pk_sales_detail primary key	ON sales_id, product_code
fk_sales_detail_product_code	ON product_code
Foreign Keys	
fk_sales_detail_product_code	(product_code) ref product (product_code)
fk_sales_detail_sales_id	(sales_id) ref sales (sales_id)

5.2.4 Product Table Schema

product Table stores the information about the product. **Table 5.2.4** column shows the column shows the data type with a specific precision. **product_code** is the unique identification number of each product under category of that product. **product** Table has relation with **category** Table via **category_id** (fk).

Table 5.2.4: Schema of product Table

product	
product_code	VARCHAR(20) NOT NULL
catagory_id	INT NOT NULL
name	VARCHAR(100) NOT NULL
sales_price	DOUBLE NOT NULL
Indexes	
pk_product primary key	ON product_code
fk_product_catagory_id	ON catagory_id
Foreign Keys	
fk_product_catagory_id	(catagory_id) ref catagory (catagory_id)

5.2.5 Category Table Schema

category Table store the data about product category. **Table 5.2.5** the column shows the data type with a specific precision. **category_id** is the unique identification number of each category. Table 5.2.5: Schema of category Table

category	
category_id	INT NOT NULL AUTOINCREMENT
name	VARCHAR(100) NOT NULL
Indexes	
pk_category primary key	ON category_id

5.2.6 Stock Table Schema

stock Table store the data about stock. **sales** and **purchase** table are connected via **product** Table. **product_code** is the primary key for this table. Depending on sale and purchase the data of **stock** table is changed. . **Table 5.2.6** the column shows the data type with a specific precision. Table 5.2.6: Schema of stock

stock	
product_code	VARCHAR(20) NOT NULL
current_quantity	INT NOT NULL
reorder_level	INT NOT NULL
Indexes	
pk_stock primary key	ON product_code
Foreign Keys	
fk_stock_product_id	(product_code) ref product (product_code)

5.2.7 Supplier Table Schema

supplier Table create the information about supplier. **supplier_id** is the unique identification number of each supplier. **Table 5.2.7** the right column shows the data type with a specific precision.

Table 5.2.7: Schema of supplier

supplier	
supplier_id	INT NOT NULL AUTOINCREMENT
address	VARCHAR(500) NOT NULL
contact_person	VARCHAR(100)
name	VARCHAR(200) NOT NULL
phone	VARCHAR(15)
Indexes	
pk_supplier primary key	ON supplier_id

5.2.8 Purchase Table Schema

purchase table store the data about product which purchased from supplier. **purchase_id** is the unique identification number of each purchase. The **constraints** enforces different rules the data has to pass. **Table 5.2.8** the column shows the data type with a specific precision. Table 5.2.8: Schema of purchase Table

purchase	
purchase_id	INT NOT NULL
purchase_date	DATE NOT NULL
supplier_id	INT NOT NULL
Indexes	
pk_purchase primary key	ON purchase_id
fk_purchase_supplier_id	ON supplier_id
Foreign Keys	
fk_purchase_supplier_id	(supplier_id) ref supplier (supplier_id)

5.2.9 Purchase Detail Table Schema

purchase_detail table store the information detail about each purchase. **Table 5.2.9** the column shows the data type with a specific precision. The **constraints** enforces different rules the data has to pass.

Table 5.2.9: Schema of purchase_detail

purchase_detail	
purchase_id	INT NOT NULL
product_code	VARCHAR(20) NOT NULL
purchase_price	DOUBLE NOT NULL
quantity	INT NOT NULL
Indexes	
pk_purchase_detail primary key	ON purchase_id, product_code
fk_purchase_detail_product_code	ON product_code
Foreign Keys	
fk_purchase_detail_product_code	(product_code) ref product (product_code)
fk_purchase_detail_purchase_id	(purchase_id) ref purchase (purchase_id)

5.3 Implementation of GUI and Report Generation

The front-end of the system is JAVA and the GUI forms are implemented using the Netbeans IDE 6.9.1. In our system we have used Swing library to create Graphical user interfaces with Java. It is an official Java GUI toolkit released by Sun Microsystems. It has a rich set of widgets.

From basic widgets like buttons, labels, scrollbars to advanced widgets like trees and tables. Swing itself is written in Java. It is a collection of packages for creating full featured desktop applications. It consists of AWT, Swing, Accessibility, Java 2D, and Drag and Drop. Most of our GUI forms were created using the Swing Drag and Drop. Drag and drop functionality is one of the most visible aspects of the graphical user interface. Drag and drop operation enables users to do complex things intuitively.

Report generation is an essential part of POS system. But many POS softwares do not provide this feature. In our developed POS system we have integrated the Report generation feature. The report layouts were designed using iReport 4.1.1 report designer tool.

iReport is the most popular visual designer tool for JasperReports Library and JasperReports Server. The JasperReports Library is the world's most popular open source reporting engine. It is entirely written in Java and can use data coming from any datasource and generate report. In our

project to create reports we connected entities from our mySql database server. The entities are displayed as tables in the report. Reports and sub reports are created to generate important POS reports. E.g. for making a sales voucher we actually created particular sale report which also has the sales detail as sub report. The reports were embedded within our Java sales management application. Figure 5.3 shows sales voucher report created using ireport and generated from Netbeans IDE.

In our project we have also created charts to show the quantity of sale of products for „today“ and for a particular duration of time, whose dates are given by the user. Charts were also created by iReport. Figure 5.3(b) shows pie chart of sales summary of dates between Sept. 18 to Oct. 19.

Sales Voucher				
Sales ID	1210190001	Customer ID	1	
Sales Date	19/10/2012	Customer Name	Farhana	
Product Code	Product Name	Price	Quantity	Total
D103	Frutika	80.0	2	160.0
S101	Pen	10.0	5	50.0
S102	Copy 100 pages	85.0	2	170.0
T102	Close Up 250 g	100.0	1	100.0
Total				480.0
VAT				10.2
Grand Total				489.8
Paid				500.0
Change				10.20

Figure 5.3(a): Sales Voucher report

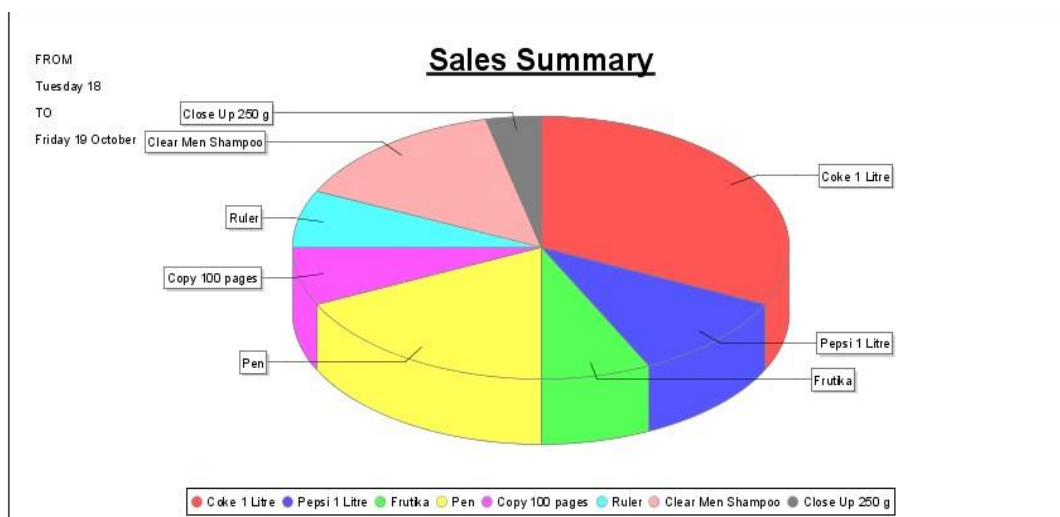


Figure 5.3(b): Sales Voucher report

5.4 Integration and Testing

Software testing is an activity that verifies whether the actual results match the expected results and to ensure that the software system is defect free i.e. detects errors in the software. It validates the software by checking that the software provides what the user actually wanted.

In our project we have performed the Unit Test and the Integrity Test. For the Unit Test we have applied the Black Box testing method. Black-box testing treats the software as a "black box", examining functionality without any knowledge of internal implementation. The tester is only aware of what the software is supposed to do, not how it does it.

5.4.1 Unit Test

Unit testing of software applications is done during the development (coding) of an application. The objective of unit testing is to isolate a section of code and verify its correctness. In procedural programming a unit may be an individual function or procedure. Functional correctness of GUIs can be ensured through unit testing. Unit testing done during the development stage saves both time and money in the end.

Table 5.4.1 shows unit tests performed in each of the functions in the „Sales“ UI. The rest of the unit testing of the other UIs is given in Appendix.

Table 5.4.1: Unit test of Sales UI functions

Test Case Objective	Method	Expected Result
Save a new Sales record using the „Sales“ UI	1. Input Customer ID (optional) 2. Select a Category 3. Select a Product 4. Input Quantity 5. Click „Add Product“ button 6. Input value in „Paid“ field 7. Click „Save“ button	• Display the Sales record in the sales table after the „Add Product“ button is pressed. After clicking the save button a message appears showing sales record saved successfully. •
Find a Sales record using the „Sales“ UI	1. Click „Find“ button 2. Input sales ID in the Input Dialog Box 3. Click „Ok“	☐ After clicking „Ok“ the sales record of the sales ID is displayed on the sales table of the UI ☐ Update and Delete button enabled

Delete a Sales record using the „Sales“ UI	1. Do the Find procedure for the sales to be deleted 2. Click „Delete“ button	☐	After clicking the „Find“ button a message appears showing sales record deleted successfully.
Clear the field values from the „Sales“ UI	1. Click the „Clear“ button	☐	Clears values of all the fields in the UI.
Exit from the UI	2. Click the „Exit“ button	☐	Exits from the „Sales“ UI to the Main page.

5.4.2 Integration Test

In Integration Testing, individual software modules are integrated logically and tested as a group. Software may consist of multiple software modules; each module may be coded by different coders. Integration testing focuses mainly on the interfaces & flow of data/information between the modules. Here priority is to be given for the integrating links rather than the unit functions which are already tested. Since our software is linked with various modules in the frontend user so we have performed the Integrity test. Table 5.4.2 shows the entire Integrity test of different modules of the software.

Table 5.4.2: Integrity test of the entire GUI

Test Case ID	Test Case Objective	Test Case Description	Expected Result
1	Check the interface link between the Login and Home page	• Click Login Button • Enter login information • Click on the Login button	To be directed to the Home page
2	Check the interface link between the Cashier Login and File Menu Items	• Enter Cashier Login information • Click on the Login button • Click the File Menu	Sales and Customer Button only enabled. Other buttons are disabled for cashier user.
3	Check the interface link between the Manager Login and File Menu Items	• Click Login Button • Enter Manager Login information • Click on the Login button	Category, Product, Purchase, Supplier and User Account Buttons and report Menu item only Enabled.

4	Check the display of Category UI	□	Click the Category Button	Display Category UI
5	Check the display of Customer UI	□	Click the Customer Button	Display Customer UI
6	Check the display of Product UI	□	Click the Product Button	Display Product UI
7	Check the display of Sales UI	□	Click the Sales Button	Display Sales UI
8	Check the display of Purchase UI	□	Click the Purchase Button	Display Purchase UI
9	Check the display of Supplier UI	□	Click the Supplier Button	Display Supplier UI
10	Check the interface of stock reorder report	• • •	Click the Report menu Click the Stock menu item Click the reorder sub menu item	Display the Stock report for reorder
11	Check the interface of Product List report	•	Click the Report menu Click the Product List menu item	Display the Product List report.
12	Check the interface of Particular Sales report	• • • •	Click the Report menu Click the Sales menu item Click the Particular Sales sub menu item Enter a particular SalesId	To be directed to the Sales report of the particular SalesId.
13	Check the interface of Particular Sales report of Today's Sale	• • •	Click the Report menu Click the Sales menu item Click the Today's Sales sub menu item	To be directed to the Sales report of Today's Sale summary along with a Pie chart at the end.
14	Check the interface of Particular Sales report of Sales Pie Chart	• • •	Click the Report menu Click the Sales menu item Click the Sales Pie Chart sub menu item	To be directed to the Sales report of Sales Pie Chart.
15	Check the interface of Customer List report	• • •	Click the Report menu Click the Customer menu item Click the Customer List sub menu item	Display the Customer List report.

14	Check the interface of Particular Customer report	• Click the Report menu • Click the Customer menu item • Click the Particular Customer sub menu item • Enter a particular CustomerId	Click To be directed to the Customer report of the particular CustomerId.
15	Check the interface of Particular Gender Customer report	• Click the Report menu • Click the Customer menu item • Click the Particular Gender sub menu item • Enter a particular gender	Click To be directed to the Customer info report by gender, of the particular gender.

Chapter 6

FUTURE SCOPE AND CONCLUSION

6.1 Future Scope

POS has options for implementing different features in the future. Technology and business changes are taking place at faster pace, as a result the retail shop POS softwares demand features for better business decision and customer service. The POS system will always be needed wherever there is sale transaction in a smart environment. In future it will get more faster, smarter and intelligent.

In this project we have attempted to design a complete software engineering architecture for the POS system. Based on this architecture the system can be extended with more desirable features in future. Our system is designed for the front desk retail shop. In future it can be converted to cloud based system where the entire POS service is accessed through the internet. End users can access cloud-based applications through a web browser or mobile app while the business software, user data, including sales and inventory are not stored locally, but in a remote server. The advantages of a cloud-based POS are instant centralization of data, ability to access data from anywhere where there is internet connection, faster and yet lower costs.

Future POS systems will be intelligent. Using Biometric Scanner, any customer checking in or out is scanned to recognize face, gender, identity and display past purchases, suggest promotions based on gender etc. Customer may even walk out of the store without actually scanning purchases of the goods because all the products selected have RFID tags, so the retailer's inventory is deducted and tracked by those tags. It may automatically charge customer's credit card and send an email for confirmation.

The POS system may have voice operated product recognition feature. Products need not to be scanned, rather may have a voice operated system to recognize the products and take the product entry. This service may include calculating sales, saving sales record etc through voice commands. Likewise, there may be many more future scopes for the POS systems making the system more of a service than just software.

6.2 Conclusion

The system is developed after a thorough review of the existing POS systems. The system is implemented in the perspective of software engineering which comprises of different useful models. It also produces reports which are a major requirement for any POS system. The system tries to cover all the aspects of a POS system but does not claim to fulfill the ultimate requirements. The system could implement more software engineering models and also generate many other relevant reports.

APPENDIX

Table: Unit test of Purchase UI functions

Test Case Objective	Method	Expected Result
Save a new Purchase record using the „Purchase UI“	• Input Supplier ID • Select a Category • Select a Product • Input Price • Input Quantity • Click „Add Product“ button • Input value in „Paid“ field • Click „Save“ button	• Display the Purchase record in the Purchase table after the „Add Product“ button is pressed. • After clicking the save button a message appears showing Purchase record saved successfully.
Find a Purchase record using the „Purchase“ UI	• Click „Find“ button • Input Purchase ID in the Input Dialog Box • Click „Ok“	☐ After clicking „Ok“ the Purchase record of the Purchase ID is displayed on the Purchase table of the UI ☐ Update and Delete button enabled
Delete a Purchase record using the „Purchase“ UI	• Do the Find procedure for the Purchase to be deleted • Click „Delete“ button	☐ After clicking the „Find“ button a message appears showing Purchase record deleted successfully.
Clear the field values from the „Purchase“ UI	☐ Click the „Clear“ button	☐ Clears values of all the fields in the UI.

Exit from the UI	☐	Click the „Exit“ button	☐	Exits from the „Purchase“ UI to the Main page.
-------------------------	---	-------------------------	---	--

Table: Unit test of Category UI functions

Test Case Objective	Method	Expected Result
Save a new Category record using the „Category UI“	• Input Category name • Input VAT amount • Click „add“ button	☐ After clicking the add button a message appears showing Category record added successfully.
Find a Category record using the „Category UI“	• Click „Find“ button in the • Input Category ID Input Dialog Box • Click „Ok“	• After clicking „Ok“ the record of the Category ID is displayed on the UI • Update and Delete button enabled
Update a Category record using the „Category UI“	• Do the Find procedure for the Category to be updated Make necessary changes of the Category in the fields Click • „Update“ button	☐ After clicking the „Update“ button a message appears showing Category record updated successfully.
Delete a Category record using the „Category“ UI	• Do the Find procedure for the Category to be deleted • Click „Delete“ button	☐ After clicking the „Find“ button a message appears showing Category record deleted successfully.
Clear the field values from the „Category“ UI	☐ Click the „Clear“ button	☐ Clears values of all the fields in the UI.
Exit from the UI	☐ Click the „Exit“ button	☐ Exits from the „Category“ UI to the Main page.

Table: Unit test of Product UI functions

Test Case Objective	Method	Expected Result
Save a new Product record using the „Product UI“	• Input Product code • Input Product name • Input Product price • Select category • Input VAT amount • Input Product re-order level • Click „add“ button	☐ After clicking the Add button a message appears showing Product record added successfully.

Find a Product record using the „Product UI	- Click „Find“ button - Input Product Code in the Input Dialog Box - Click „Ok“	- After clicking „Ok“ the record of the Product Code is displayed on UI. Update and Delete button enabled
Update a Product record using the „Product UI	Do the Find procedure for the Product to be updated Make necessary changes of the Product in the fields Click „Update“ button	☐ After clicking the „Update“ button a message appears showing Product record updated successfully.
Delete a Product record using the „Product“ UI	Do the Find procedure for the Product to be deleted Click „Delete“ button	☐ After clicking the „Find“ button a message appears showing Product record deleted successfully.
Clear the field values from the „Product“ UI	☐ Click the „Clear“ button	☐ Clears values of all the fields in the UI.
Exit from the UI	☐ Click the „Exit“ button	☐ Exits from the „Product“ UI to the Main page.

Table: Unit test of User Account UI functions

Test Case Objective	Method	Expected Result
Create a new User account using the „User Account UI“	- Input User ID - Input User password - Select Level of the supplier - Click „add“ button	☐ After clicking the Add button a message appears showing User account added successfully.
Find a User account using the „User UI	- Click „Find“ button - Input User ID in the Input Dialog Box - Click „Ok“	- After clicking „Ok“ the account of the User ID is displayed on UI. Update and Delete button enabled
Update a User account using the „User UI	Do the Find procedure for the User to be updated Make necessary changes of the User in the fields Click „Update“ button	☐ After clicking the „Update“ button a message appears showing User account updated successfully.
Delete a User account using the „User“ UI	Do the Find procedure for the User to be deleted Click „Delete“ button	☐ After clicking the „Find“ button a message appears showing User account deleted successfully.

Clear the field values from the „User“ UI	□	Click the „Clear“ button	□	Clears values of all the fields in the UI.
Exit from the UI	□	Click the „Exit“ button	□	Exits from the „User“ UI to the Main page.

Table: Unit test of Supplier UI functions

Test Case Objective	Method	Expected Result
Save a new Supplier record using the „Supplier UI“	• Input Supplier name • Input Supplier address • Input Phone number • Input Contact person • Click „add“ button	☐ After clicking the Add button a message appears showing Supplier record added successfully.
Update a Supplier record using the „Supplier UI	• Do the Find procedure for the Supplier to be updated Make necessary changes of the Supplier in the fields Click • „Update“ button	☐ After clicking the „Update“ button a message appears showing Supplier record updated successfully.
Find a Supplier record using the „Supplier UI	• Click „Find“ button • Input Supplier ID in the Input Dialog Box • Click „Ok“	• After clicking „Ok“ the record of the Supplier ID is displayed on UI. • Update and Delete button
		enabled
Delete a Supplier record using the „Supplier“ UI	• Do the Find procedure for the Supplier to be deleted • Click „Delete“ button	☐ After clicking the „Find“ button a message appears showing Supplier record deleted successfully.
Clear the field values from the „Supplier“ UI	☐ Click the „Clear“ button	☐ Clears values of all the fields in the UI.
Exit from the UI	☐ Click the „Exit“ button	☐ Exits from the „Supplier“ UI to the Main page.

Table: Unit test of Customer UI functions

Test Case Objective	Method	Expected Result
---------------------	--------	-----------------

Save a new Customer record using the „Customer UI“	• • • • •	Input Customer name Input Customer address Input Mobile number Input e-mail address Select gender of the supplier Click „add“ button	□	After clicking the Add button a message appears showing Customer record added successfully.
Find a Customer record using the „Customer UI“	• • •	Click „Find“ button Input Customer ID in the Input Dialog Box Click „Ok“	• •	After clicking „Ok“ the record of the Customer ID is displayed on UI. Update and Delete button enabled
Update a Customer record using the „Customer UI“	• • •	Do the Find procedure for the Customer to be updated Make necessary changes of the Customer in the fields Click „Update“ button	□	After clicking the „Update“ button a message appears showing Customer record updated successfully.
Delete a Customer record using the „Customer“ UI	• •	Do the Find procedure for the Customer to be deleted Click „Delete“ button	□	After clicking the „Find“ button a message appears showing Customer record deleted successfully.
Clear the field values from the „Customer“ UI	□	Click the „Clear“ button	□	Clears values of all the fields in the UI.
Exit from the UI	□	Click the „Exit“ button	□	Exits from the „Customer“ UI to the Main page.

REFERENCES

- [1] Randy Perry Michael J. Witty, WHITE PAPER (Total Cost of Ownership for Point - of -Sale and PC Cash Drawer Solutions: A Comparative Analysis of Retail Checkout Environments, 2006 Update), November 2006
- [2] Abhishek Dasaper, A Thought Paper, Art of POS transformation
- [3] Sujith George, Developing a successful Point-of-Sale (POS) test automation strategy
- [4] Boehm, B. (1988), "A Spiral Model of Software Development and Enhancement," *IEEE Computer* 21, 5, 61-72.
- [5] W.HU, N-Tier Architecture and Tips, 29 Aug 2012
- [6] Nicolai Josuttis, Designing a 3-Tier-Architecture, A Position Paper for the Design Fest of the EuroPLOP 2001