



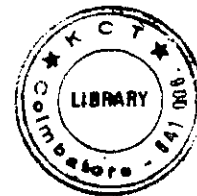
**SEMI INTELLIGENT ANALYSIS SYSTEM FOR
CO-OPERATIVE STORE SYSTEM**

P. 2260

By

N.Divya

Registration Number:71205621011



Of

**KUMARAGURU COLLEGE OF TECHNOLOGY
COIMBATORE**

A PROJECT REPORT

Submitted to the

FACULTY OF INFORMATION AND COMMUNICATION ENGINEERING

*In partial fulfillment of the requirements
for the award of the degree*

of

MASTER OF COMPUTER APPLICATION

**ANNA UNIVERSITY
CHENNAI 600 025**


June 2008

BONAFIDE CERTIFICATE

Certified that this project report titled Semi Intelligent Analysis System for Co-operative Store System is the bonafide work of Ms. N.Divya. (Registration Number :71205621011) who carried out the research under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form part of any other project report or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

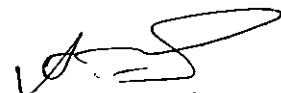


Supervisor

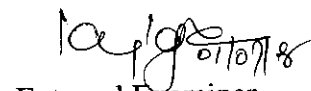


Head of the Department

Submitted to Project and Viva Examination held on 01-07-2008



Internal Examiner



External Examiner

June 2nd, 2008

To whomsoever it may concern

This is to certify that **N.Divya** (71205621011) student of Kumaraguru College of technology, doing her final year MCA (Computer Application) has successfully completed her project entitled '**Semi Intelligent Analysis System for Co-operative Store System**' under the guidance of **V.SathishKumar**-Production Manager from 4-1-2008 to 2-6-2008.

During her project duration her conduct and contribution has been outstanding.

We wish her all the best for her future endeavor.

For **ERBATECH-Machinery Pvt Ltd.,**



V.Sathish Kumar
(Production Manager)

ERBATECH MACHINERY (P) LTD.

Regd.Office and Works: S.F.No.41, Vadaputhur Village, Yelur (P.O.), Othakalmandapam, Coimbatore-641032, Tamil Nadu, India

Telephone: 00 91 (0) 4259 241055 · Web: <http://www.erbatech.de> · E-Mail: ebm.cbe@erbatech.com

TNGST: (119) 2283035 dt.4-7-2006 CST: 667094

TIN No. 3397 2283035

UTI Bank, Avanashi Road, Coimbatore-641037, India. A/C No: 090010200015905 SWIFT: UTIBINBB006

ABSTRACT

The Semi Intelligent Analysis System for Co-operative Store automates the legacy cooperative stores. This system allows user to analyze past sales trend and enable automated purchase if the quantity of the products drops below the threshold.

Initially the threshold is given as input by the user, later the system will automatically adjust its threshold based on the student strength and past trend accumulated from previous three years data. The system also allows the managers to analyze the data manually depending on requirements. The proposed system has facility to create number of users and provide privilege to various users.

This system has two major modules one is to automate the ledger based cooperative store system and other is to analyze the data and aid decision making such as purchase schedule of the products. The second module can be used with any other cooperative store automation system to perform analysis.

The analysis module will extract the data from various sources of the existing automated system and perform analysis. To initiate analysis and to make decision always user is involved in this system. In future machine learning technology can be used to develop fully automated intelligent system for cooperative stores.

ACKNOWLEDGEMENT

I wish to express my sincere thanks to **Dr.JOSEPH V.THANIKAL Ph.D.,** Principal, Kumaraguru College of Technology, Coimbatore, for giving me the consent to undertake this project.

My deepest acknowledgement to **Dr.M.GURURAJAN Ph.D.,** Head of the Department, Computer Applications, Kumaraguru College of Technology, Coimbatore, for his timely help and guidance throughout this project.

I am greatly indebted to **Mr.A.MUTHUKUMAR MCA.,MPhil.,** Project Coordinator, my Guide, Assistant Professor, Department of computer Applications and my sincere thanks to him for guiding and encouraging me at every stage of this project.

I express my sincere thanks to **Mr.V.Sathish Kumar,** Production Manager, ERBATECH Machinery Private Ltd, Coimbatore for his support and assistance at various levels of my project work.

Finally, I owe my great deal of gratitude to my parents for helping me to overwhelm in all my proceedings. I bow my heart and head with heartfelt thanks to my department staffs and all those who taught me their warm service to succeed and achieve my work.

TABLE OF CONTENTS

CHAPTER NO.	TITLE	PAGE NO.
	ABSTRACT	iv
	ACKNOWLEDGEMENT	v
	LIST OF FIGURES	ix
	LIST OF TABLES	x
	CHAPTERS	
I	INTRODUCTION	
	1.1 Organization Profile	1
	1.2 Problem Definition	2
II	SYSTEM ANALYSIS	
	2.1 Existing System	3
	2.2 Proposed System	3
	2.3 User Interface Requirements	4
	2.3.1 User Creation Module	4
	2.3.2 Customer Entry Module	4
	2.3.3 Purchase Module	4
	2.3.4 Sales Module	4
	2.3.5 Deposit Module	4
	2.3.6 Analysis Module	5

	2.3.7 Extraction Module	5
	2.3.8 Mapping Module	5
III	DEVELOPMENT ENVIRONMENT	
	3.1 Hardware Environment	6
	3.2 Software Environment	6
	3.3 Programming Environment	6
	3.3.1 Dot Net Framework	6
	3.3.2 VB.NET	8
	3.3.3 SQL (Structured Query Language)	12
IV	SYSTEM DESIGN	
	4.1 Data Model	16
	4.1.1 Table Relationship	16
	4.1.2 Data Relationship Diagram	22
	4.2 Process Model	23
	4.2.1 Data Flow Diagram	23
	4.2.2 Use case Diagram	25
	4.2.3 Sequence Diagram	27
	4.2.4 Collaboration Diagram	29
V	SYSTEM IMPLEMENTATION	32
VI	TESTING	33
	6.1 Test Case Reports	37

VII	CONCLUSION AND FUTURE ENHANCEMENT	39
	7.1 Conclusion	39
	7.2 Future Enhancement	39
VIII	APPENDICES	
	8.1 Sample Screens	40
IX	REFERENCES	52

LIST OF FIGURES

FIGURE NO.	NAME	PAGE NO.
4.2.1.1	Automated Cooperative Store System	24
4.2.1.2	Legacy Cooperative Store System	24
4.2.2.1	User Operation	25
4.2.2.2	Administrator Operation	26
4.2.3.1	Administrator Login	27
4.2.3.2	Creates New User	28
4.2.3.3	Creates New Customer	28
4.2.3.4	Viewing Stock Details	29
4.2.4.1	Administrator Login	30
4.2.4.2	Creates New User	30
4.2.4.3	Creates New Customer	31
4.2.4.4	Viewing Stock Details	31

LIST OF TABLES

TABLE NO.	NAME	PAGE NO.
4.1.1.1	User Table	18
4.1.1.2	Customer Table	18
4.1.1.3	Stock Table	19
4.1.1.4	Deposit Table	19
4.1.1.5	Sales Table	20
4.1.1.6	Sub Transaction Table	20

CHAPTER 1

INTRODUCTION

1.1 ORGANIZATION PROFILE

Erbatech- Machinery Private Ltd is a leading monopoly technical organization in Coimbatore. At its facilities in Germany, the company produces machinery for wet finishing processes of textiles and carpets. Kurt Brückner founded the company in 1963 to act as design and development centre for the Brückner-Group of companies and to be a technology partner for the textile industry.

In the late 1970s HASPELFLOW[®], a piece-dyeing machine for tubular knits and wovens was developed to replace the prevalent dye beck technology. The machine provided an economic and innovative alternative that was sold in large numbers worldwide and strengthened BRÜCKNER's name in the industry. In order to satisfy the increasing demand for open-width processing ranges the SCOUT[®] product line was developed. The first machines went into operation in 1995. Since then this modular and economic machine concept has developed into the most important product of ERBATECH, with total sales of more than 120 machines up to date.

In 2002 the company was split off the Bruckner-Group and renamed ERBATECH GmbH. Mr. Ulrich von Christen took over the MD position and is leading the operations. Under new ownership the company was restructured and focused only on wet finishing machines. All product lines were refurbished and further upgraded.

In 2004 the new dye padder SCOUT-COLOR[®] was developed together with partners from the textile industry. This dye padder is applicable for woven as well as for knitted fabrics and completes our product portfolio in order to provide the full package of machines for integrated finishing plants. In 2006 OBERMAIER[®] the well-known producer of yarn dyeing machines was acquired and integrated into ERBATECH.

1.2 PROBLEM DEFINITION

The Existing system takes much more in order to analyze the previous records and place the purchase order for the stock as the quantity of the stock seems to be varying at the different quarter of the year. So the administrator finds this as a tedious job.

The Administrator needs identification and analysis with regard to the setting of the threshold values so as to automate the existing legacy system with what the user needs rather than what they want. Not until the problem has been identified, defined, and evaluated which reaches the appropriate solution.

Once the analysis phase is done, the extraction and the mapping phase play an important role. These two modules helps to extract as well map the table attribute values from the existing legacy system to the current proposed system. And the values that are mapped is the initial value set for the proposed system. In case if the existing system is a manually recorded system the extraction and the mapping phase is not necessary as all the initial input is given by the user itself.

Determining the information each user needs is particularly a difficult task. In fact, it is recognized as one of the most difficult tasks in system development. The usual approach is to ask the user what information is currently available and what other information is required, with the needs and requirements of the user. The system has to be defined clearly previously.

CHAPTER 2

SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

The number of increasing tasks has completely led to the automation of the existing cooperative stores operation. The existing legacy system could also be automated, as it will not support the decision making.

The existing system will not allow the automated purchase from the supplier if the quantity of the items goes below the set threshold quantity value. Existing systems automates the cooperative stores operation. This will not support decision making.

The existing system does not allow setting the threshold value by analyzing the previous records. The major bottleneck of the existing system is the administrator cannot make decisions based on the past trends.

2.2 PROPOSED SYSTEM

The proposed system automates the legacy cooperative stores. This system allows the administrator to create number of users and provide privilege to various users using the system.

The newly proposed system allows user to analyze the past trend and enables automated purchase if the quantity drops below the threshold value. Initially the threshold in given as input by the user, later the system will automatically adjust its threshold based on the student strength and past trend accumulated from previous three years data.

This system also allows the administrator to analyze the data manually depending on requirements.

2.3 USER INTERFACE REQUIREMENTS

The proposed system is designed in such a way so as to meet the requirements. The user interface is key to application usability. The application should include content presentation, application navigation, and user assistance.

The modules which are used in the system are

2.3.1 User Creation Module

This module allows the administrator to create users and allow administrator to provide privilege to user. This module use encryption techniques to protect password of the individual users.

2.3.2 Customer Entry Module

This module allows user to enter, modify and update Customer details.

2.3.3 Purchase Module

This module allows the user to place purchase order to the suppliers. This module enable online quotation request from various suppliers and allow the user to cancel or confirm the order.

2.3.4 Sales Module

This module automates the sales process of the cooperative stores. This module will get the customer ID as input and display the balance amount. If the balance is positive then the customer is allowed to purchase items. If the purchase amount exceeds the balance amount the transaction will be aborted.

2.3.5 Deposit Module

This module will allow the user to deposit amount in the customers account.

2.3.6 Analysis Module

This module allows the manager to manually analyze the data by passing queries. This module has some sub modules that changes the threshold of different items based on various factors whose initial values are given as input. Later these thresholds are adjusted by the system based on deciding factors.

2.3.7 Extraction Module

This module will extract the values from the legacy database and load it in the analysis database. This module is only for the system that uses existing automated cooperative store management system.

2.3.8 Mapping Module

This module is sub module of extraction module. This will provide user interface to map the existing automated system database attribute with the analysis system database attribute.

CHAPTER 3

DEVELOPMENT ENVIRONMENT

3.1 HARDWARE ENVIRONMENT

Monitors	:	800x600 minimum resolutions at 256 colors minimum
Memory	:	Approximately 64 MB of on board memory
I/O	:	Two or three button mouse and standard 101-key keyboard
MHZ	:	At least 166 MHZ processor

3.2 SOFTWARE ENVIRONMENT

Operating System	:	Windows XP, Windows Server 2003
Web Server	:	Microsoft IIS
Database	:	SQL Server 2000
Software for Development	:	VB.NET

3.3 PROGRAMMING ENVIRONMENT

3.3.1 DOTNET Framework

The Microsoft .NET Framework is a platform for building, deploying and running XML web services and applications. It provides a highly productive, standards-based, multilanguage environment for integrating existing investments with next-generation applications and services as well as the agility to solve the challenges of deployment and operation of Internet-scale applications.

Microsoft .NET Framework is a software component that is a part of several Microsoft Windows operating systems.

It has a large library of pre-coded solutions to common programming problems and manages the execution of programs written specifically for the *framework*. The .NET Framework is a key Microsoft offering and is intended to be used by most new applications created for the windows platform.

The pre-coded solutions that form the framework's Base Class Library cover a large range of programming needs in a number of areas, including user interface, data access, database connectivity, cryptography, web application development, numeric algorithms, and network communications.

The Common Language Runtime (CLR) is an important part of the .NET Framework environment. The CLR provides the appearance of an application virtual machine so that programmers need not consider the capabilities of the specific CPU that will execute the program. The CLR also provides other important services such as security, memory management, and exception handling. The class library and the CLR together compose the .NET Framework.

CLI (Common Language Infrastructure)

The core aspects of the .NET framework lie within the Common Language Infrastructure, or CLI. The purpose of the CLI is to provide a language-agnostic platform for application development and execution, including functions for exception handling, garbage collection, security, and interoperability. Microsoft's implementation of the CLI is called the Common Language Runtime or CLR.

The CLR is composed of four primary parts:

- Common Type System (CTS)
- Common Language Specification (CLS)
- Metadata
- Virtual Execution System (VES)

The .NET Framework is included with Windows Server 2008 and Windows Vista. The current version of the framework can also be installed on Windows XP and the Windows Server 2003 family of operating systems.

In short .NET Framework is:

- A platform designed from the start for writing Internet-aware and Internet-enabled applications that embrace and adopt open standards such as XML, HTTP, and SOAP.
- A platform that provides a number of very rich and powerful application development technologies, such as Windows Forms, used to build classic GUI applications, and of course ASP.NET, used to build web applications.
- A platform with an extensive classic library that provides extensive support for data-access (relational and XML), a director services, message queuing, and much more.
- A platform that has a base class library that contains hundreds of classes for performing common tasks such as file manipulations, registry access, security, threading, and searching of text using regular expressions.
- A platform that doesn't forget its origins, and has great interoperability support for existing components that you or third parties have written, using COM or standard DLL's.
- A platform with an independent code execution and management environment called the Common Language Runtime(CLR), which ensures code is safe to run, and provides an abstract layer on top of the operating system, meaning that elements of the .NET Framework can run on many systems and devices.

3.3.2 VB.NET

Visual Basic.NET has many new and improved language features — such as inheritance, interfaces, and overloading that make it a powerful object-oriented programming language. Using Visual Basic developer, we can now create multithreaded, scalable applications using explicit multithreading. Other new language features in Visual

Basic .NET include structured exception handling, custom attributes, and common language specification (CLS) compliance.

The CLS is a set of rules that standardizes such things as data types and how objects are exposed and interoperate. Visual Basic .NET adds several features that take advantage of the CLS. Any CLS-compliant language can use the classes, objects, and components you create in Visual Basic .NET. And you, as a Visual Basic user, can access classes, components, and objects from other CLS-compliant programming languages without worrying about language-specific differences such as data types. CLS features used by Visual Basic .NET programs include assemblies, namespaces, and attributes. These are the new features to be stated briefly: Inheritance, Exception Handling, Overloading, Constructors and Destructors, Data Types, Interfaces, Delegates, Shared Members, References, Assemblies and Attributes.

Visual Basic .NET provides the easiest, most productive language and tool for rapidly building Windows and Web applications.

Visual Basic .NET comes with enhanced visual designers, increased application performance, and a powerful integrated development environment (IDE). It also supports creation of applications for wireless, Internet-enabled hand-held devices. The following are the features of Visual Basic .NET with .NET Framework 1.0 and Visual Basic .NET 2003 with .NET Framework 1.1.

Powerful Windows-based Applications

Visual Basic .NET comes with features such as a powerful new forms designer, an in-place menu editor, and automatic control anchoring and docking.

Visual Basic .NET delivers new productivity features for building more robust applications easily and quickly. With an improved integrated development environment (IDE) and a significantly reduced startup time.

Visual Basic .NET offers fast, automatic formatting of code as you type, improved IntelliSense, an enhanced object browser and XML designer, and much more.

Building Web-based Applications

With Visual Basic .NET we can create Web applications using the shared Web Forms Designer and the familiar "drag and drop" feature. You can double-click and write code to respond to events. Visual Basic .NET 2003 comes with an enhanced HTML Editor for working with complex Web pages. We can also use IntelliSense technology and tag completion, or choose the WYSIWYG editor for visual authoring of interactive Web applications.

Simplified Deployment

With Visual Basic .NET we can build applications more rapidly and deploy and maintain them with efficiency. Visual Basic .NET 2003 and .NET Framework 1.1 makes "DLL Hell" a thing of the past. Side-by-side versioning enables multiple versions of the same component to live safely on the same machine so that applications can use a specific version of a component. XCOPY-deployment and Web auto-download of Windows-based applications combine the simplicity of Web page deployment and maintenance with the power of rich, responsive Windows-based applications.

Powerful, Flexible, Simplified Data Access

You can tackle any data access scenario easily with ADO.NET and ADO data access. The flexibility of ADO.NET enables data binding to any database, as well as classes, collections, and arrays, and provides true XML representation of data. Seamless access to ADO enables simple data access for connected data binding scenarios.

Using ADO.NET, Visual Basic .NET can gain high-speed access to MS SQL Server, Oracle, DB2, Microsoft Access, and more.



P-2260

Improved Coding

You can code faster and more effectively. A multitude of enhancements to the code editor, including enhanced IntelliSense, smart listing of code for greater readability and a background compiler for real-time notification of syntax errors transforms into a rapid application development (RAD) coding machine.

Direct Access to the Platform

Visual Basic developers can have full access to the capabilities available in .NET Framework 1.1. Developers can easily program system services including the event log, performance counters and file system. The new Windows Service project template enables to build real Microsoft Windows NT Services. Programming against Windows Services and creating new Windows Services is not available in Visual Basic .NET Standard, it requires Visual Studio 2003 Professional, or higher.

Full Object-Oriented Constructs

You can create reusable, enterprise-class code using full object-oriented constructs. Language features include full implementation inheritance, encapsulation, and polymorphism. Structured exception handling provides a global error handler and eliminates spaghetti code.

XML Web Services

XML Web services enable you to call components running on any platform using open Internet protocols. Working with XML Web services is easier where enhancements simplify the discovery and consumption of XML Web services that are located within any firewall. XML Web services can be built as easily as you would build any class in Visual Basic 6.0. The XML Web service project template builds all underlying Web service infrastructure.

Mobile Applications

Visual Basic .NET 2003 and the .NET Framework 1.1 offer integrated support for developing mobile Web applications for more than 200 Internet-enabled mobile devices. These new features give developers a single, mobile Web interface and programming model to support a broad range of Web devices, including WML 1.1 for WAP—enabled cellular phones, compact HTML (cHTML) for i-Mode phones, and HTML for Pocket PC, handheld devices, and pagers. Please note, Pocket PC programming is not available in Visual Basic .NET Standard, it requires Visual Studio 2003 Professional, or higher.

COM Interoperability

You can maintain your existing code without the need to recompile. COM interoperability enables you to leverage your existing code assets and offers seamless bi-directional communication between Visual Basic 6.0 and Visual Basic .NET applications.

Reuse Existing Investments

You can reuse all your existing ActiveX Controls. Windows Forms in Visual Basic .NET 2003 provide a robust container for existing ActiveX controls. In addition, full support for existing ADO code and data binding enable a smooth transition to Visual Basic .NET 2003.

3.3.3 SQL (Structured Query Language)

SQL is a standard computer language for accessing and manipulating database. SQL is a standard interactive and programming language for querying and modifying data and managing databases.

What is SQL?

- SQL stands for Structured Query Language
- SQL allows to access a database
- SQL is an ANSI standard computer language
- SQL can execute queries against a database
- SQL can retrieve data from a database
- SQL can insert new records in a database
- SQL can delete records from a database
- SQL can update records in a database
- SQL is easy to learn

SQL as a Standard

SQL is an ANSI (American National Standard Institute) standard computer language for accessing and manipulating database systems. SQL statements are used to retrieve and update data in a database.

SQL works with database programs like MS Access, DB2, Informix, MS SQL Server, Oracle, Sybase, etc. Unfortunately, there are many different versions of the SQL language, but to be in compliance with the ANSI standard, they must support the same major keywords in a similar manner (such as SELECT, UPDATE, DELETE, INSERT, WHERE, and others).

SQL SERVER 2000

Microsoft SQL Server 2000, the data management and analysis backbone for Microsoft's .NET enterprise applications and servers, improves on the performance, reliability, scalability, quality, and ease of use of its predecessors.

SQL Server 7.0 and SQL Server 6.5, SQL Server 2000 offers rich Extensible Markup Language (XML) support, comprehensive analysis services, and simplified database administration - features that combine to produce a solution able to rapidly deliver reliable, scalable e-commerce, data warehousing and line of business upgrade from SQL Server 6.5 and SQL Server 7.0.

Microsoft SQL Server 2000 introduce new feature that support XML functionality. The combination of these features makes SQL Server 2000 an XML enabled database server.

These features include:

- The ability to access SQL Server using HTTP.
- Support for XDR (XML – Data Reduced) schemes and the ability to specify XPATH queries against these schemas.
- The ability to retrieve and write XML data.
- Retrieve XML data using the SELECT statement and FOR XML clause.

IIS (Internet Information Services)

Microsoft Internet Information Services (IIS; formerly called Server) is a set of Internet-based services for servers using Microsoft Windows. It is the world's second most popular web server in terms of overall websites. As of May 2007 it served 31% of all websites according to Netcraft. The servers currently include FTP, SMTP, NNTP and

HTTP/HTTPS. Internet Information Services (IIS) is a powerful Web server that provides a highly reliable, manageable, and scalable Web application infrastructure for all versions of Windows Server 2003. IIS helps organizations increase Web site and application availability while lowering system administration costs. IIS supports the Microsoft Dynamic Systems Initiative (DSI) with automated health monitoring, process isolation, and improved management capabilities.

IIS has three new components, which allows IIS to segment Web server code from application handling code. These three new components are: a kernel-mode HTTP listener, called HTTP.sys; a user-mode configuration and process manager, called the WWW Service Administration and Monitoring component; and the application handler, which is loaded into each worker process. These worker processes, in turn, service requests for application pools in HTTP.sys.

In IIS, HTTP.sys listens for requests and routes them to the appropriate requests queue. Each request queue corresponds to one application pool. Because no application code runs in HTTP.sys, it cannot be affected by failures in user-mode code that normally affect the status of the Web service. If an application fails, HTTP.sys continues to accept and queue new requests on the appropriate queue until one of the following: the process has been restarted and begins to accept requests, there are no queues available, there is no space left on the queues, or the Web service itself has been shut down by the administrator.

CHAPTER 4

SYSTEM DESIGN

4.1 DATA MODEL

4.1.1 Table Relationship

Database Design

A database is a collection of interrelated data stored with minimum redundancy to serve many users quickly and efficiently. The general theme behind a database is to integrate all the information. The general objective of database design is to make the data access easy, inexpensive and flexible to the user.

The main objectives of designing a database are:

- Data integration
- Data integrity
- Data independence

Normalization

Normalization, sometimes referred to as canonical synthesis, is a technique for designing relational database tables to minimize duplication of information and, in so doing, to safeguard the database against certain types of logical or structural problems, namely data anomalies.

In normalization when multiple instances of a given piece of information occur in a table, the possibility exists that these instances will not be kept consistent when the data within the table is updated, leading to a loss of data integrity.

A table that is sufficiently normalized is less vulnerable to problems of this kind, because its structure reflects the basic assumptions for when multiple instances of the same information should be represented by a single instance only.

Accordingly, more highly normalized tables are typically used in database applications involving many isolated transactions, while less normalized tables tend to be used in database applications that need to map complex relationships between data entities and data attributes.

A table that is not sufficiently normalized can suffer from logical inconsistencies of various types, and from anomalies involving data operations. In such a table:

- The same information can be expressed on multiple records; therefore updates to the table may result in logical inconsistencies. This leads to the update anomaly.
- There are circumstances in which certain facts cannot be recorded at all which leads to the insertion anomaly.
- There are circumstances in which the deletion of data representing certain facts necessitates the deletion of data representing completely different facts which leads to the deletion anomaly.
- Ideally, a relational database table should be designed in such a way as to exclude the possibility of update, insertion, and deletion anomalies.

The normal forms of relational database theory provide guidelines for deciding whether a particular design will be vulnerable to such anomalies. It is possible to correct an unnormalized design so as to make it adhere to the demands of the normal forms: this is called normalization.

Normalization typically involves decomposing an unnormalized table into two or more tables that were they to be combined (joined), would convey exactly the same information as the original table.

Name : User Table

Description : Used to store user details with the user ID as the primary key

FIELD NAME	DATA TYPE	SIZE	KEY
UserID	Varchar	10	PK
Username	Varchar	20	
Password	Varchar	20	
Privilege	Varchar	20	

Table 4.1.1.1 User Table

Name : Customer Table

Description : Used to store the complete Customer information

FIELD NAME	DATA TYPE	SIZE	KEY
CustID	Varchar	10	PK
Custname	Varchar	20	
CustInfo	Varchar	20	

Table 4.1.1.2 Customer Table

Name : Stock Table
Description : Used to store the stock details and quantity

FIELD NAME	DATA TYPE	SIZE	KEY
Itemno	Varchar	10	PK
Itemname	Varchar	20	
TtlQuantity	Number	20	
Cost	Number	20	
Unit/Cost	Number	20	
Threshold	Number	20	

Table 4.1.1.3 Stock Table

Name : Deposit Table
Description : Used to store the Deposit details

FIELD NAME	DATA TYPE	SIZE	KEY
DepositID	Varchar	10	PK
TransID	Varchar	20	FK
CustID	Varchar	20	FK
CautionDeposit	Number	20	
Amount	Number	20	

Table 4.1.1.4 Deposit Table

Name : Sales Table
Description : Used to store the Sales details

FIELD NAME	DATA TYPE	SIZE	KEY
TransID	Varchar	20	PK
CustID	Varchar	20	FK
Cost	Number	20	

Table 4.1.1.5 Sales Table

Name : Sub Transaction Details
Description : Used to view the sub transaction details

FIELD NAME	DATA TYPE	SIZE	KEY
TransID	Varchar	10	PK
Itemno	Varchar	20	FK
Quantity	Number	20	
Date	Date	20	
Time	Time	20	

Table 4.1.1.6 Sub Transaction Table

Input Design

Input design is part of the overall system design which requires lot of attention. Computers require necessary data for their functioning. Hence, input design becomes an essential part of computer-oriented system.

The activity of putting data into the computer for processing can be activated by instructing the computer to read data from a written printed document or it can occur by keying data directly into the system.

The goal of designing input data is to make data entry easy, logical and free from errors, avoid delays extra steps, and keeping the process simple. Input are raw data that are accepted by the system and are processed to produce the output will magnify these errors.

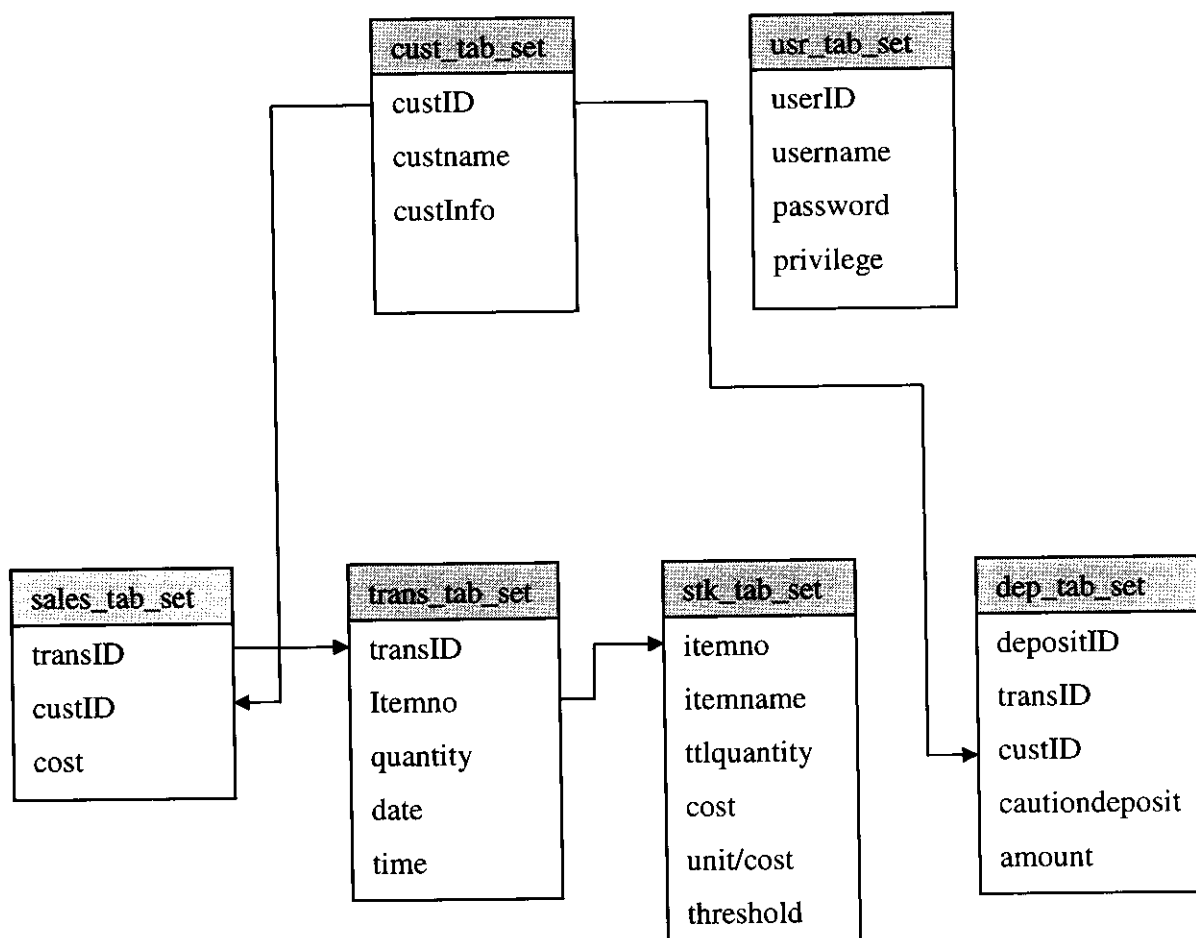
Output Design

A quality output is one, which meets the requirements of the end user and presents the information clearly. In any systems results of processing are communicated to the user and to other systems through outputs. In the output design it is determined how the information is to be displayed for immediate needs.

It is the most important and direct source information to the user. Efficient and intelligent output design improves the system's relationship with the user and helps in decision-making.

The objective of the output design is to convey the information of all the past activities, current status and to emphasize important events. The output generally refers to the results and information that is generated from the system. Outputs from computers are required primarily to communicate the results of processing to the users.

4.1.2 Data Relationship Diagram



4.2 PROCESS MODEL

4.2.1 Data Flow Diagram

Data Flow Diagram is directed graphs in which the nodes specify processing activities and the arcs that specify data items transmitted between processing nodes. Like flow charts, data flow diagram can be used at any desired level of abstraction.

A data flow diagram can be used to represent data flow between individual statements or block statements in a routine, data flow sequential routine between concurrent processes or data in a distributed computing system, where each node represents a geographically remote processing unit.

Unlike flowcharts, data flow diagrams do not indicate decision logic or condition under which various processing nodes in the diagram being activated.

A data flow diagram (DFD) is a graphical representation of the "flow" of data through an information system. A data flow diagram can also be used for the visualization of data processing (structured design). It is common practice for a designer to draw a context-level DFD first which shows the interaction between the system and outside entities. This context-level DFD is then "exploded" to show more detail of the system being modeled.

Data flow diagrams are excellent mechanisms for communicating with customers during requirement analysis; also they are widely used for representation of external and top-level internal design specifications.

The Data flow diagrams may be used to represent a system or software at any level of abstraction. DFD's may be partitioned into levels that represent increasing information flow and functional details.

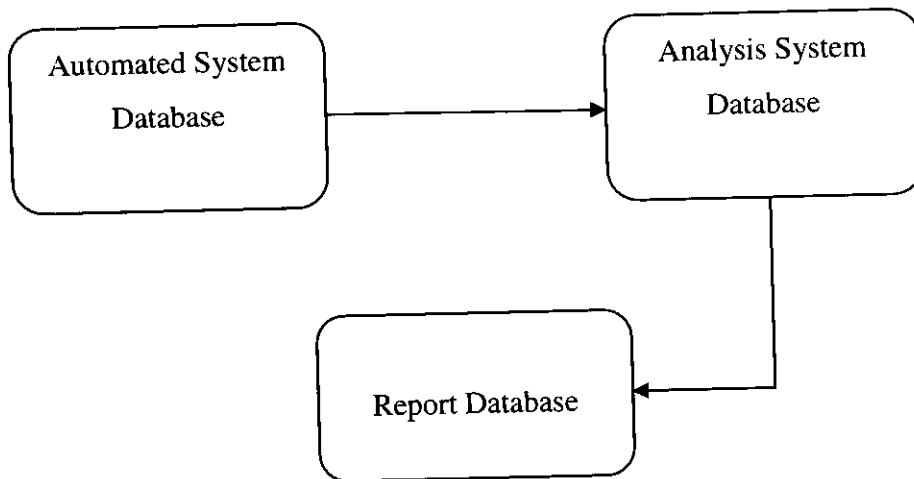


Figure 4.2.1.1 Automated Cooperative Store System

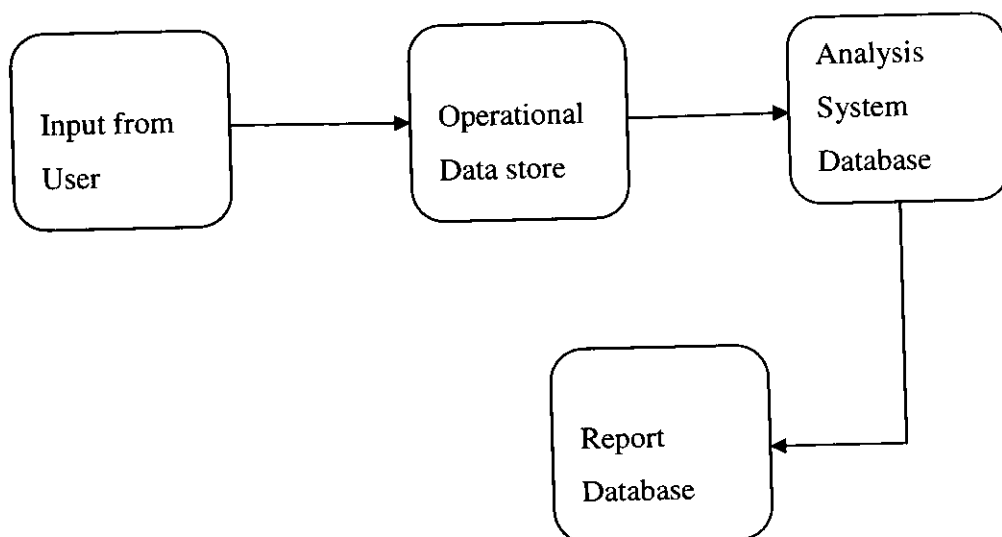


Figure 4.2.1.2 Legacy Cooperative Store System

4.2.2 Use case Diagram

A **use case diagram** is used to present a graphical overview of the functionality provided by a system in terms of actors, their goals, represented as use cases and any dependencies between those use cases.

Use case diagram is a type of behavioral diagram defined by the Unified Modelling Language (UML) created from a Use-case analysis. The main purpose of a use case diagram is to show what functions are performed by which actors.

The true value of a use case lies in two areas:

- The written description of system behavior regarding a business task or requirement. This description focuses on the value provided by the system to external entities such as human users or other systems.
- The position or context of the use case among other use cases. As an organizing mechanism, a set of consistent, coherent use cases promotes a useful picture of system behavior, a common understanding between the customer/owner/user and the development team.

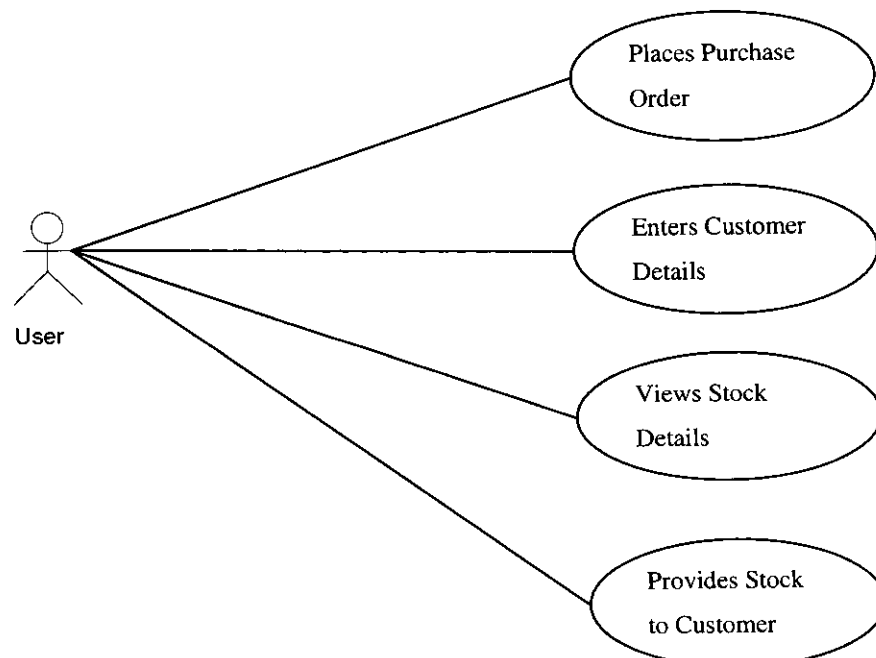


Figure 4.2.2.1 User Operation

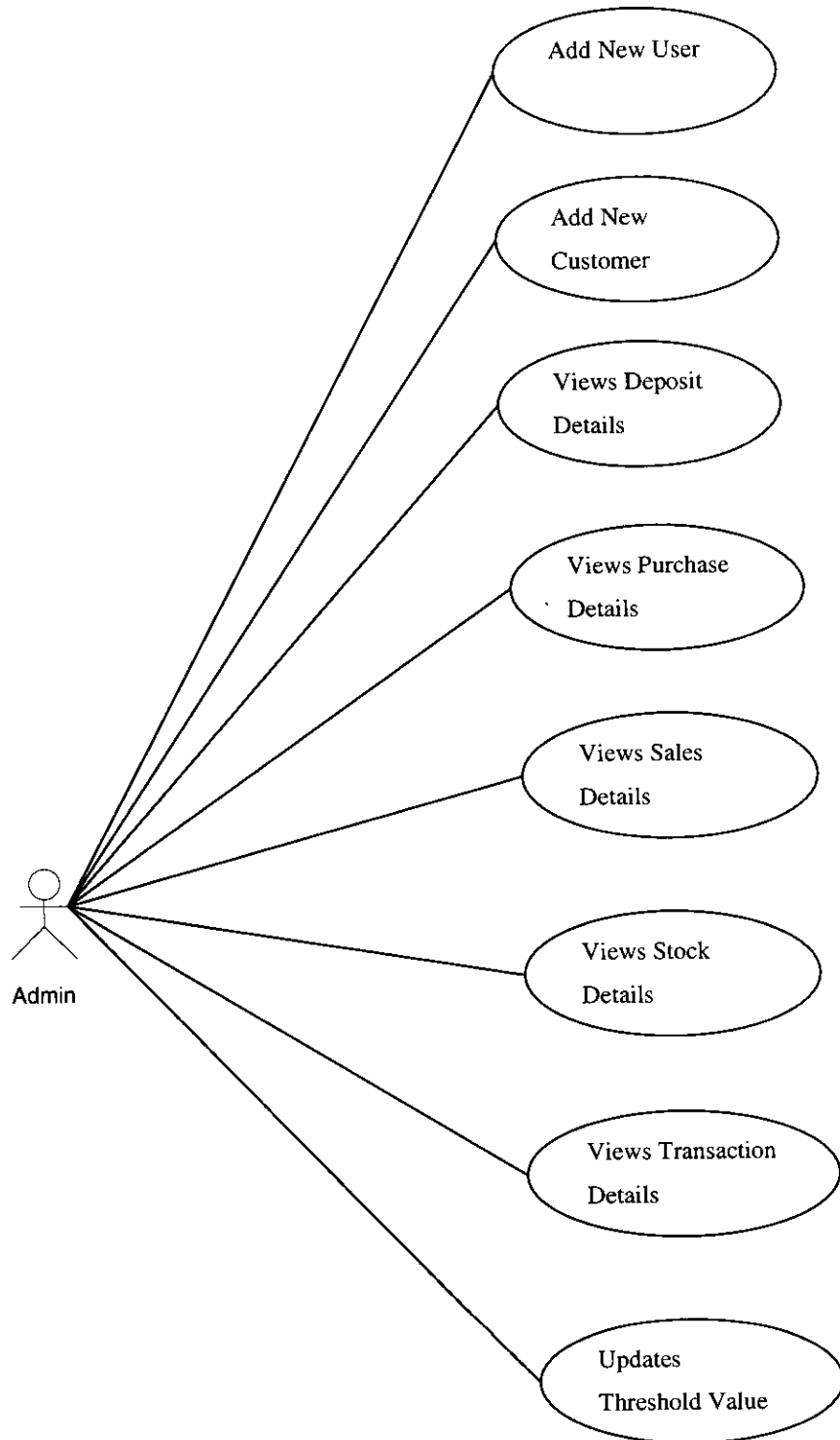


Figure 4.2.2.2 Administrator Operation

4.2.3 Sequence Diagram

A sequence diagram is a form of interaction diagram which shows objects as lifelines running down the page, with their interactions over time represented as messages drawn as arrows from the source lifeline to the target lifeline. Sequence diagrams are good at showing which objects communicate with which other objects; and what messages trigger those communications.

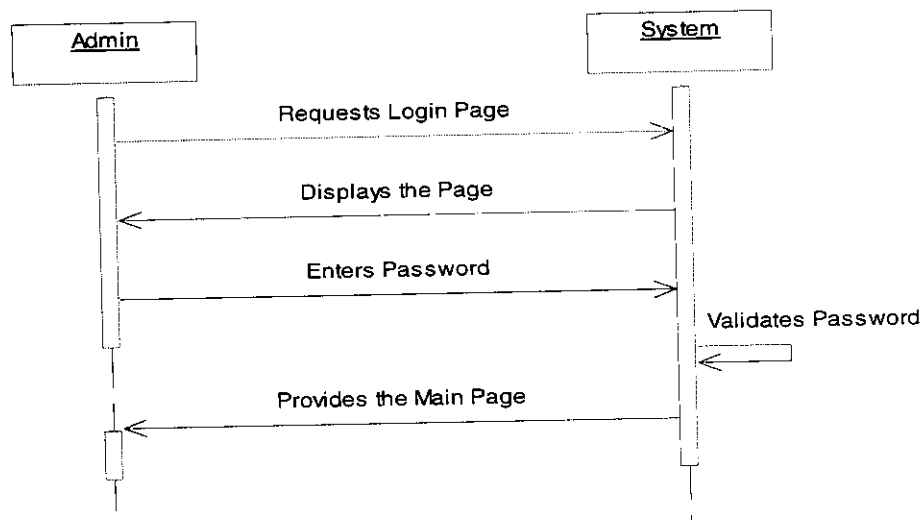


Figure 4.2.3.1 Administrator Login

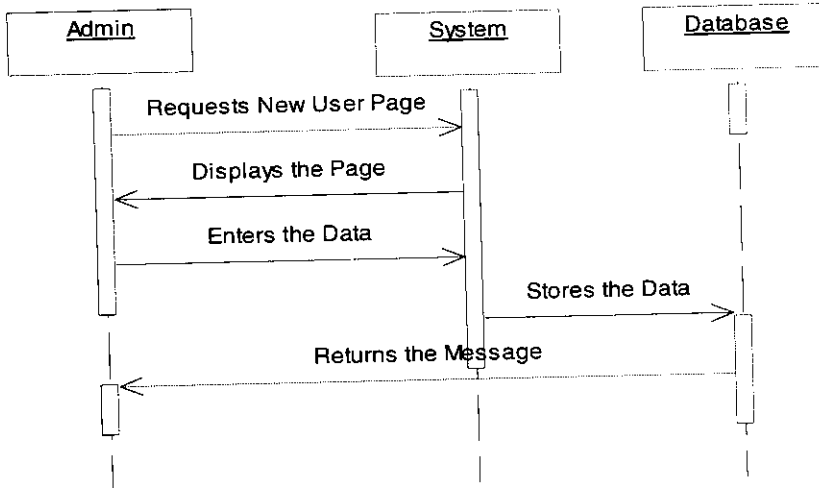


Figure 4.2.3.2 Creates New User

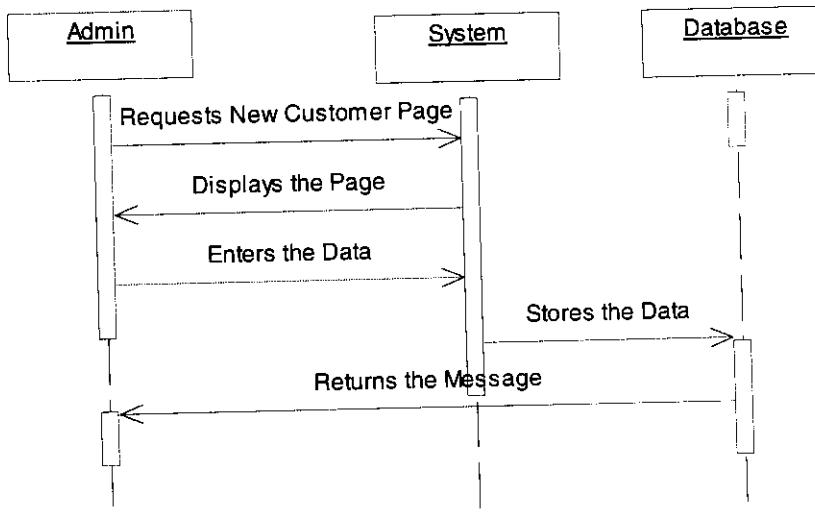


Figure 4.2.3.3 Creates New Customer

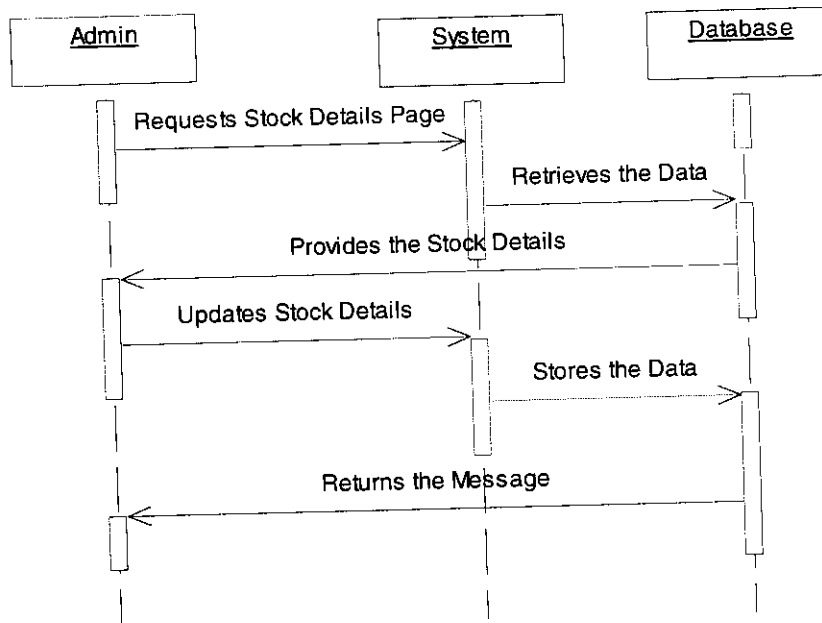


Figure 4.2.3.4 Viewing Stock Details

4.2.4 Collaboration Diagram

Collaboration diagrams, like Sequence Diagrams, show how objects interact over the course of time. However, instead of showing the sequence of events by the layout on the diagram, collaboration diagrams show the sequence by numbering the messages on the diagram. This makes it easier to show how the objects are linked together, but harder to see the sequence at a glance.

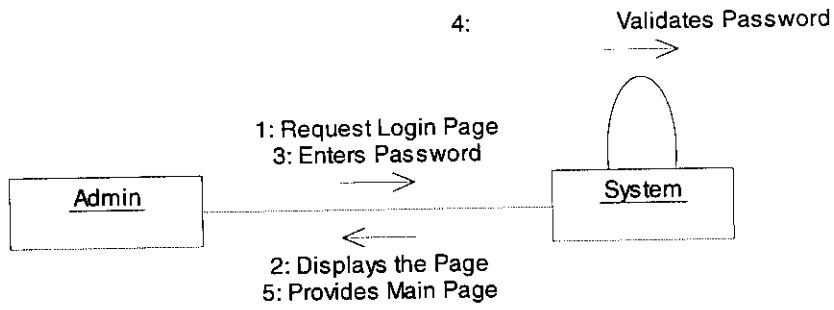


Figure 4.2.4.1 Administrator Login

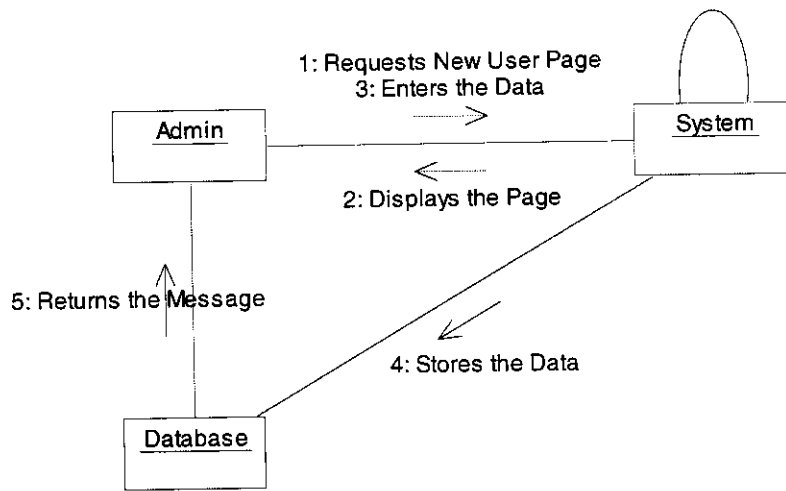


Figure 4.2.4.2 Creates New User

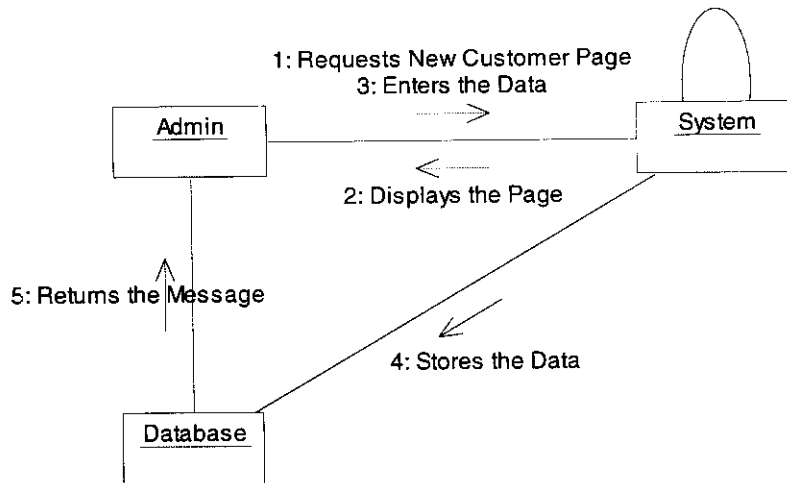


Figure 4.2.4.3 Creates New Customer

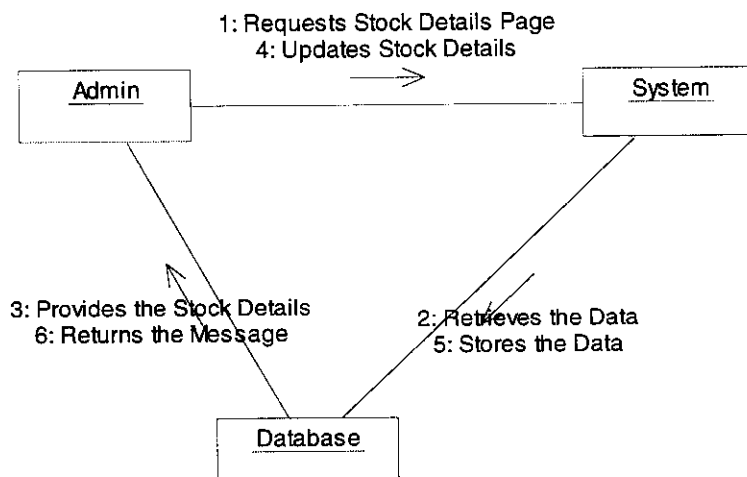


Figure 4.2.4.4 Viewing Stock Details

CHAPTER 5

SYSTEM IMPLEMENTATION

System Implementation is the stage in the project where the theoretical design is turned into a working system. The most critical stage is achieving a successful system and in giving confidence on the new system for the user that it will work efficiently and effectively.

The existing system was long time process. The proposed system was developed using VB.NET and SQL. The existing system caused long time transmission process but the system developed now has a very good user-friendly tool, which has a menu-based interface, graphical interface for the end user.

Implementation is the stage where theoretical design is converted into a working system. The following are steps involved in the implementation plan.

- Test the system with sample data
- Detection and correction errors
- Make the necessary changes in the system
- Check with the existing system
- Installation of hardware and software utilities
- Training and involvement of user personnel

CHAPTER 6

TESTING

Testing is vital to the success of the system. System testing makes a logical assumption that if all the parts of the system are correct, the goal will be successfully achieved. System testing is the stage of implementation that we aimed at assuring that the system works accurately and efficiently before live operation commences.

Software testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. The user tests the developed system and changes are made according to their needs. The testing phases involve the testing of developed system using various kinds of data.

The candidate system may subject to variety of tests. In the response security and usability is tested. A series of testing is performed for the proposed system, before the system is ready for user acceptance testing.

The objectives of testing are

- Testing is the process of executing the program with the intention of finding an error
- A good test is one that has a high probability of finding an as-yet-undiscovered error
- A successful test is that which uncovers as-yet-undiscovered error

System Testing

For every software project, there is an inherent conflict of interest that occurs as testing begins. The people, who have built the software, are now asked to test the software. These same developers have a vested interest in demonstrating that the program is error free, that it will be completed on schedule and with in budget. From psychological point of

view, software analysis and testing (along with coding) is constructed works. The software engineer creates a computer program, is demonstrated and data structure.

White Box Testing

White box focuses on the program control structure. Test cases are derived to ensure that all statements in the program have executed at least once during, testing and that all logical conditions implying that this test is typically applied to small program components. The system has been tested by providing variety of inputs to ensure that all the statements are executed at least once and that too in the expected manner. All topic and transaction path from origin to destination was tested to identify and correct the possible error.

Black Box Testing

Black box testing broadens our focus and might be called testing in the large. Black box tests are designed to validate functional requirements without regards to the internal working of the program. Black box techniques focus on information domain of the software deriving test cases by partitioning input and output in a manner that provides through test coverage. The requirement for higher quality software demands a more systematic approach to testing. The specification stating what a program should do, and it should perform under various conditions are examined. The test cases are developed for each condition or combination of conditions and submitted for processing. By examining the results, the performance of the program according to its specified requirements can be determined.

Unit Testing

Unit testing is the first level of testing in this different module is tested against the specifications produced during the design of the modules. Unit testing is done for the verification of the code produced during the coding phase and to test the internal logic of

modules. It refers to the verification of the single program module in an isolated environment. Unit testing first focused on the modules independently of one another to locate errors. After coding each dialogue is tested and run individually. All unnecessary coding were removed and it was ensured that all the modules works as the programmer would expect. Logical errors found were corrected.

Integration Testing

Data can be lost across an interface, one module can have adverse affect on another, sub functions, when combined, may not produce the desired major functions. Integration testing is a systematic testing for constructing the program structure while at the same time conducting tests to uncover errors associated with in the interface. The objective of it is to take unit tested modules and build a program structure. All the modules are combined and tested as a whole. Here correction is difficult because the vast expense of the entire program complicate the isolation of courses. In integration testing steps, all the errors un-covered are corrected for the next testing step.

Validation Testing

Validation testing provides the final assurance that the software meets all the functional, behavioral and performance requirements. The software is completely assembled on a package. Validation succeeds when the software functions in a manner in which the user expects. Validation refers to the process of using software in a live environment in order to find errors. During the courses of validating the system, failures may occur and sometimes the coding has to be changed according to the requirements. Thus the feedback from validation phase generally precedes changes in the software. Once the application was made free of all logical and interface errors, inputting dummy data ensured that the software developed satisfies all the requirements of the user. This dummy data is usually known as test case.

User Acceptance Testing

User acceptance of a system is the key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with the prospective system. Users at time of development can make changes whenever required. This is done regarded to the following points.

- Input Screen design
- Output Screen design
- On-line message to guide the user
- Format of ad-hoc reports and other output

The above testing is done taking various kinds of test data; Preparation of test data plays a vital role in the system testing. After preparing the test data, the system under study is tested using that test data. While testing the system by which test data errors are again un-covered and corrected by using above testing steps and corrections are also noted for future use.

User Training and Documentation

The implementation of the system includes the training of the system. Training of the system operators includes not only the instruction on how to use the system, but also how to diagnose system errors and mal-functions and ways to resolve the same. So proper training should be provided to system operators. No training is complete without familiarizing users with simple system maintenance activities.

Change Over

Implementation of a new computer system to replace an existing is a more difficult conversion. If not properly planned, there can be many problems. Some large computer systems have taken long time to convert. The accuracy of the conversion is of at most importance both to user confidence in the system and to effective operation. There are

many types of change over, such as direct changeover, parallel running, and staged changeover.

6.1 Test Case Reports

Create New User

Test Case	Input Data	Expected Result	Actual Result	Pass/Fail
Add UserID	UserID	UserID must be created	UserID gets created	Pass
Add Username	Username	Username must be created	Username must be added	Pass
Enter Password	Password	Password must be added	Password gets added	Pass
Select Privilege	User/Admin	Privilege must be granted	Privilege gets added	Pass

Create New Customer

Test Case	Input Data	Expected Result	Actual Result	Pass/Fail
Add CustomerID	CustomerID	CustomerID must be created	CustomerID gets created	Pass
Add Customername	Customername	Customername must be created	Customername must be added	Pass
Add Date Of Birth	Date Of Birth	Date Of Birth must be added	Date Of Birth gets added	Pass
Add Phone Number	Customer Phone Number	Phone Number must be added	Phone Number gets added	Pass
Add Address	Address	Address must be added	Address gets added	Pass

Add Deposit Details

Test Case	Input Data	Expected Result	Actual Result	Pass/Fail
Add DepositID	DepositID	DepositID must be created	DepositID gets added	Pass
Add TransactionID	TransactionID	TransactionID must be added	TransactionID gets added	Pass
Add CustomerID	CustomerID	CustomerID must be added	CustomerID gets Added	Pass
Add Caution Deposit	Caution Deposit	Caution Deposit must be added	Caution Deposit gets added	Pass
Add Amount	Amount	Amount must be added	Amount gets added	Pass

Add Transaction Details

Test Case	Input Data	Expected Result	Actual Result	Pass/Fail
Add TransactionID	TransactionID	TransactionID must be added	TransactionID gets added	Pass
Add ItemNumber	ItemNumber	ItemNumber must be added	ItemNumber gets Added	Pass
Add Quantity	Quantity	Quantity must be added	Quantity gets added	Pass
Add Date and Time	Date and Time	Date and Time must be added	Date and Time gets added	Pass

CHAPTER 7

CONCLUSION AND FUTURE ENHANCEMENT

7.1 CONCLUSION

The project has been designed, developed and implemented and provides a full-fledged approach for proficient with best of results. The project satisfies each efficient user for saving his time and cost. The system is integration easily with other.

This system automates the legacy cooperative stores and allows user to analyze past trend and enables automated purchase if the quantity drops below the threshold value.

The analysis module will extract the data from various sources of the existing automated system and perform analysis. To initiate analysis and to make decision always user is involved in this system. In future machine learning technology can be used to develop fully automated intelligent system for cooperative stores.

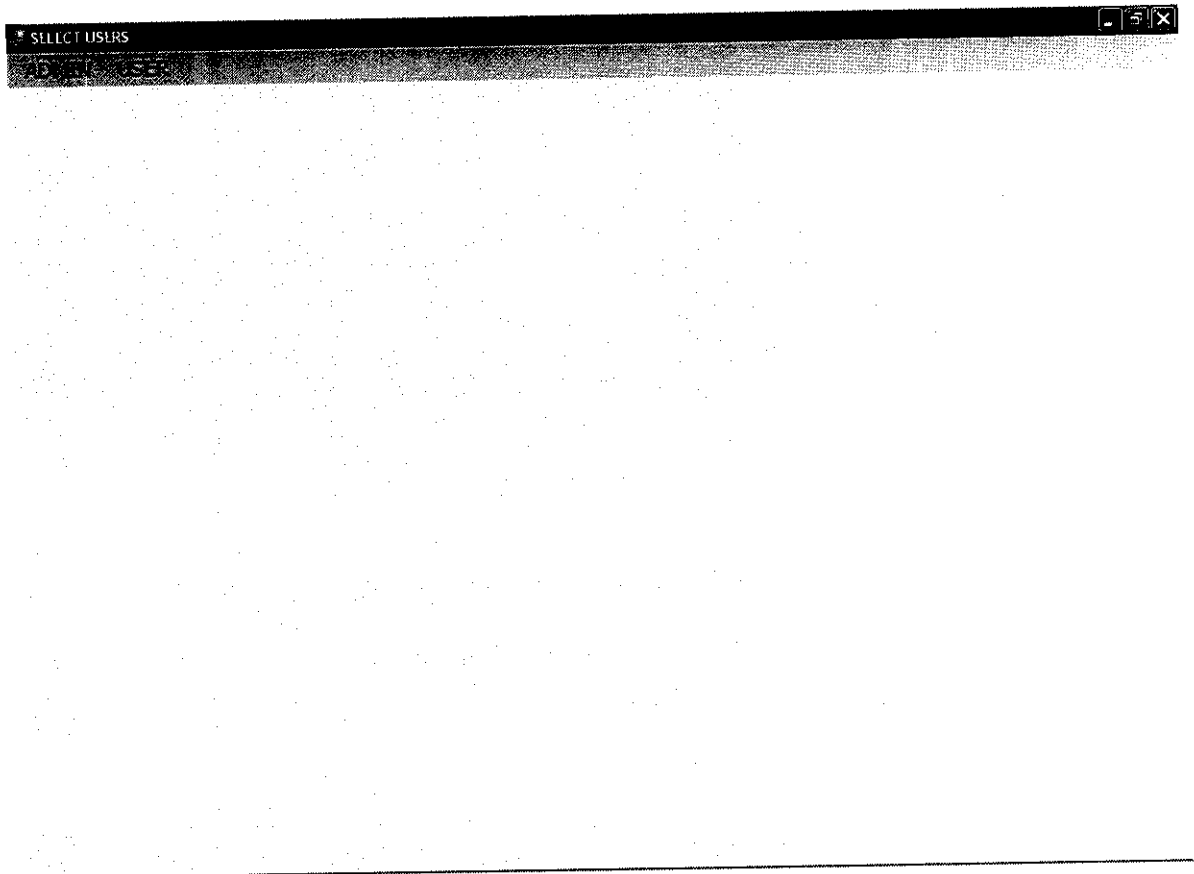
7.2. FUTURE ENHANCEMENT

The system has the capability for easy integration with other systems. New modules can be added to the existing system with less effort. Providing extra facilities can enhance the planning module.

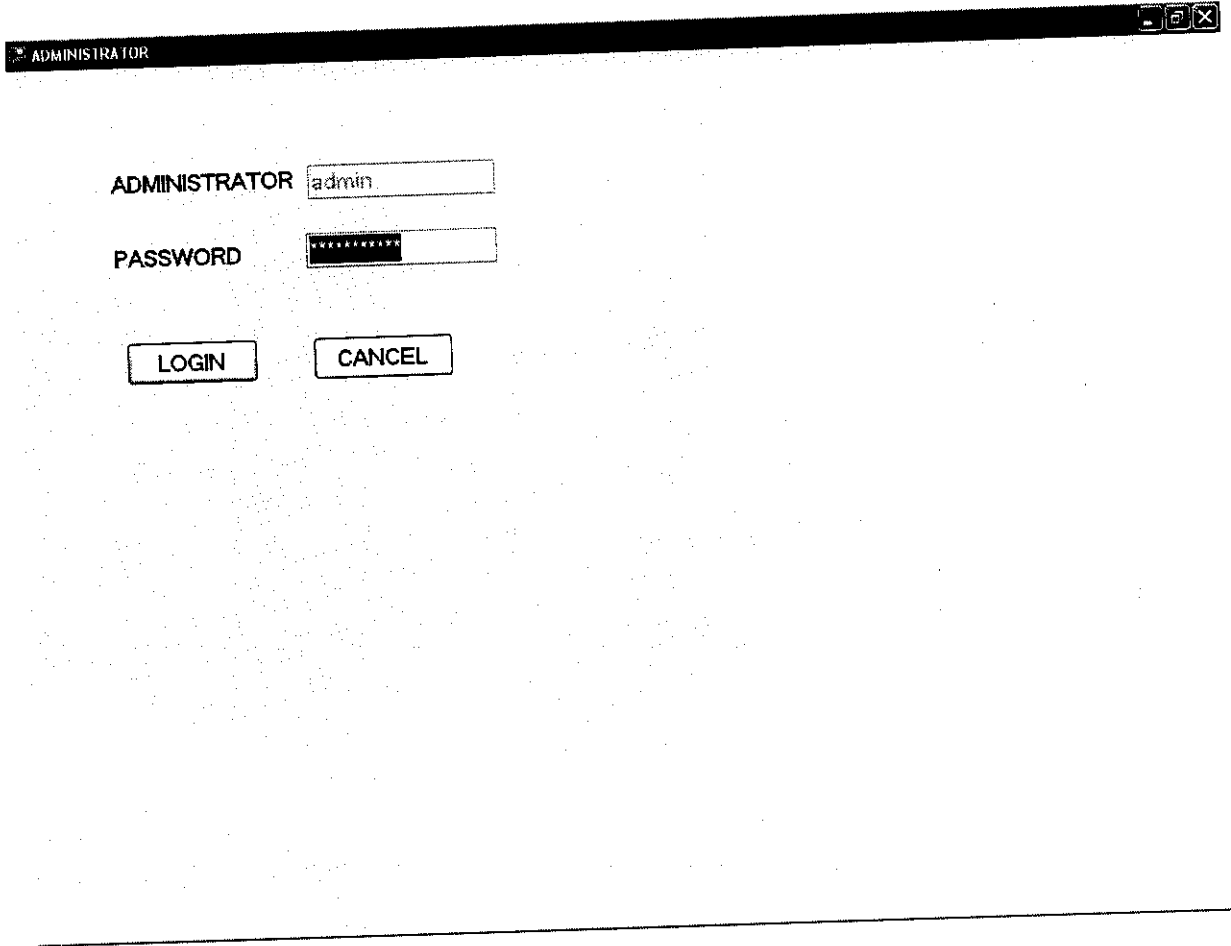
CHAPTER 8

APPENDICES

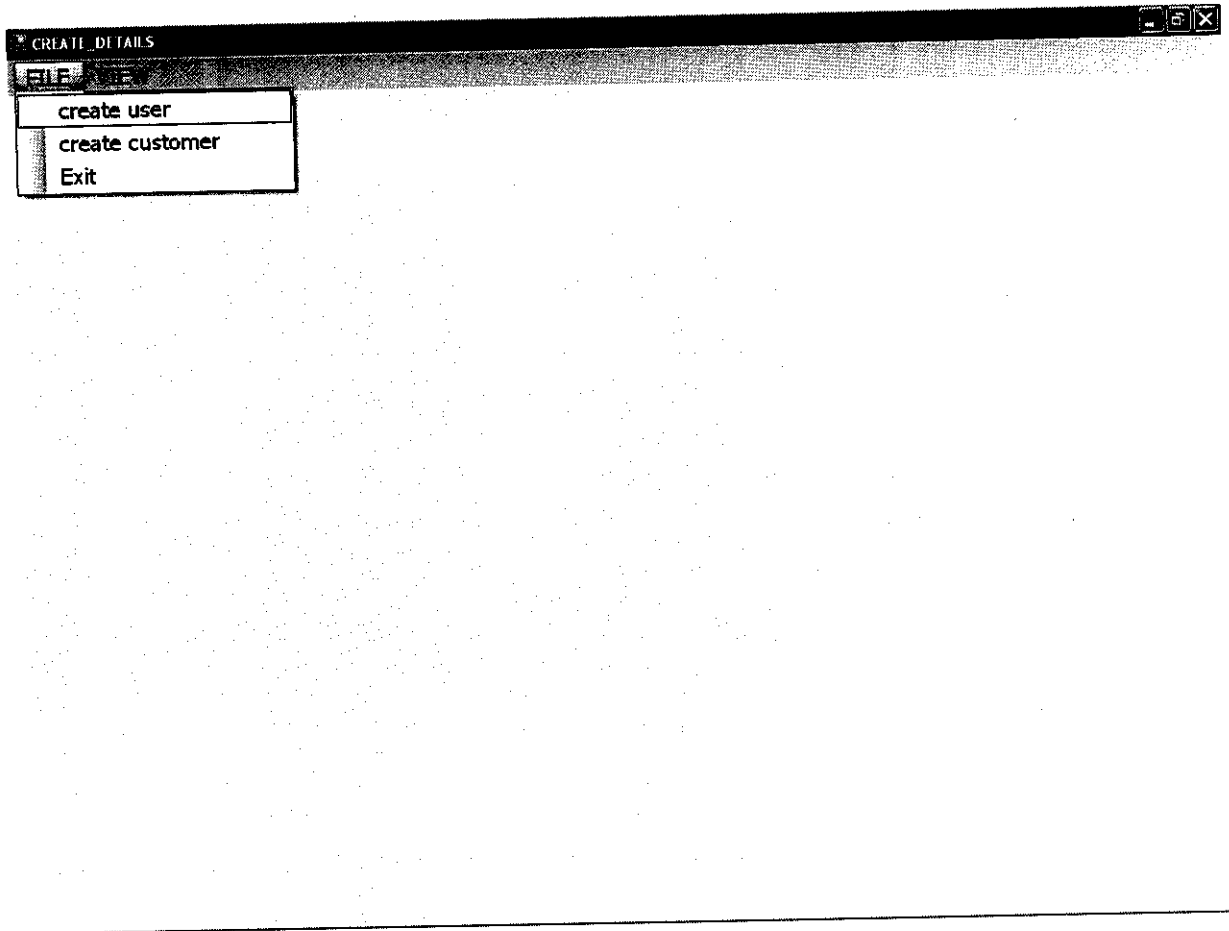
8.1. SAMPLE SCREENS



Output Screen



Admin Login Screen

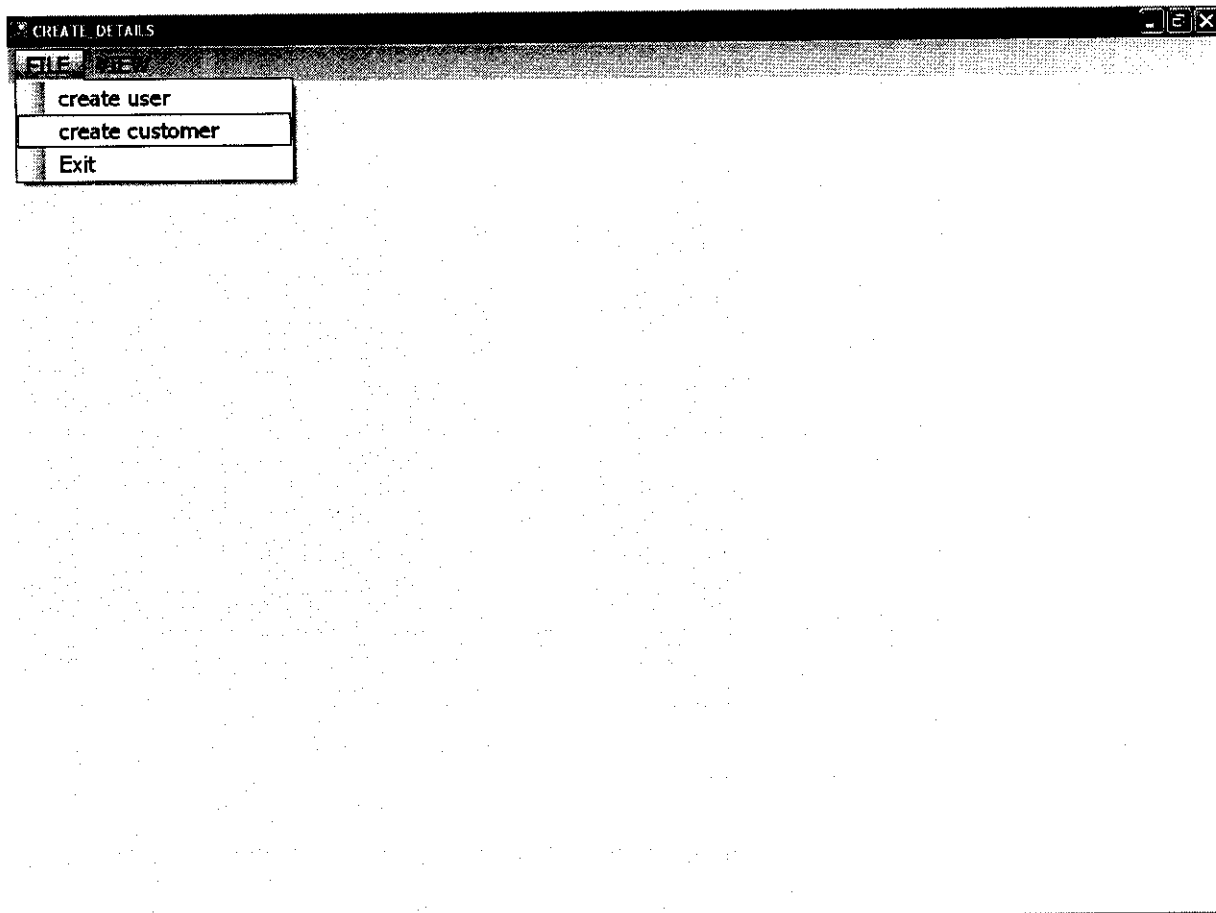


User Creation Screen

USER_DETAILS

USERID	101
USERNAME	madu
PASSWORD	*****
PRIVILEGE	USER

User Creation with privilege



Customer Creation Screen

CREATE_CUSTOMER

CUSTOMERID: 102

CUSTOMERNAME: maith

DATE OF BIRTH:

PHONE NUMBER: 0422-253666

ADDRESS: tatabad,coimbatore

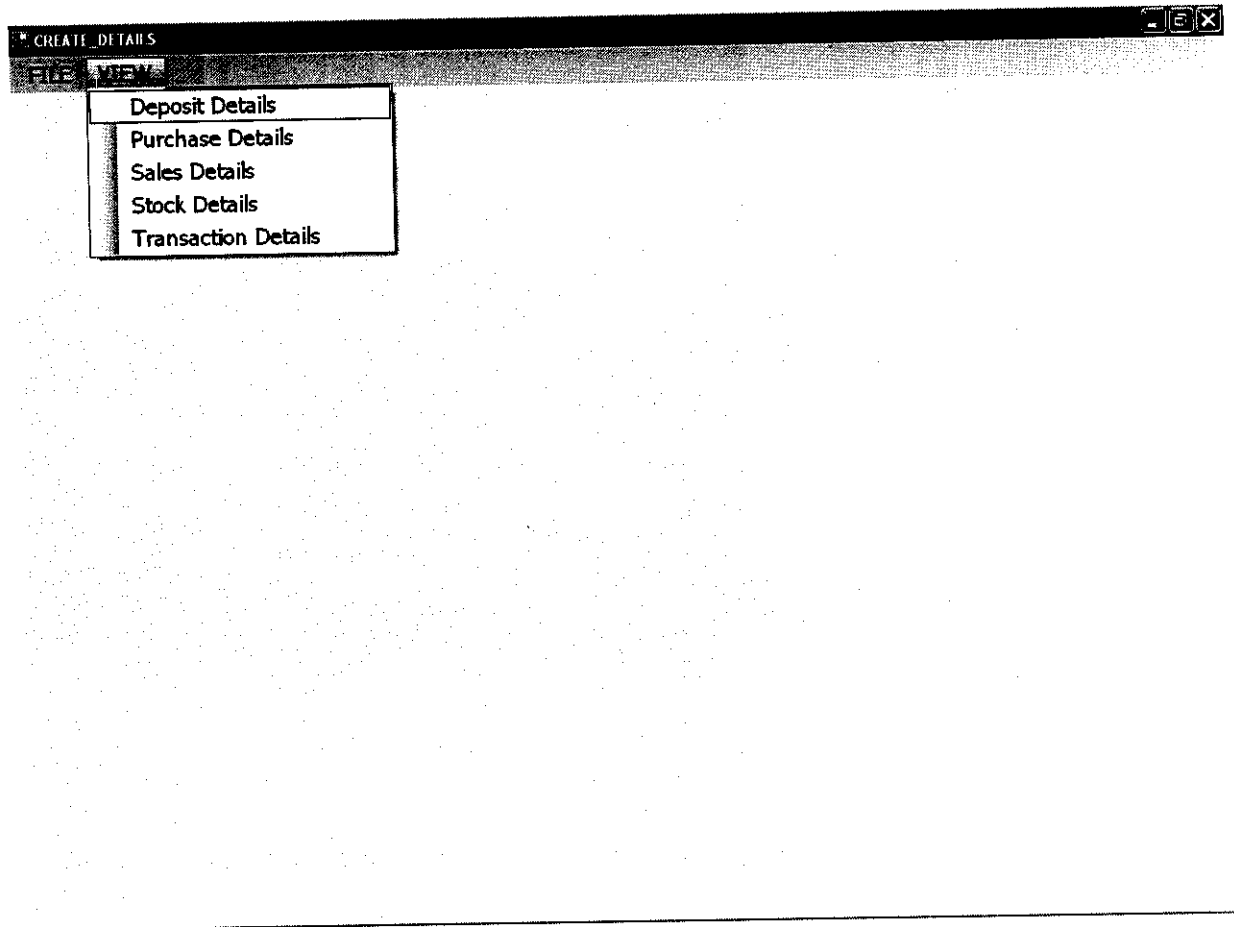
CREATE CANCEL

Tuesday , June 24, 2008

June, 2008

Sun	Mon	Tue	Wed	Thu	Fri	Sat
25	26	27	28	29	30	31
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	1	2	3	4	5
Today: 6/24/2008						

Customer Information Screen



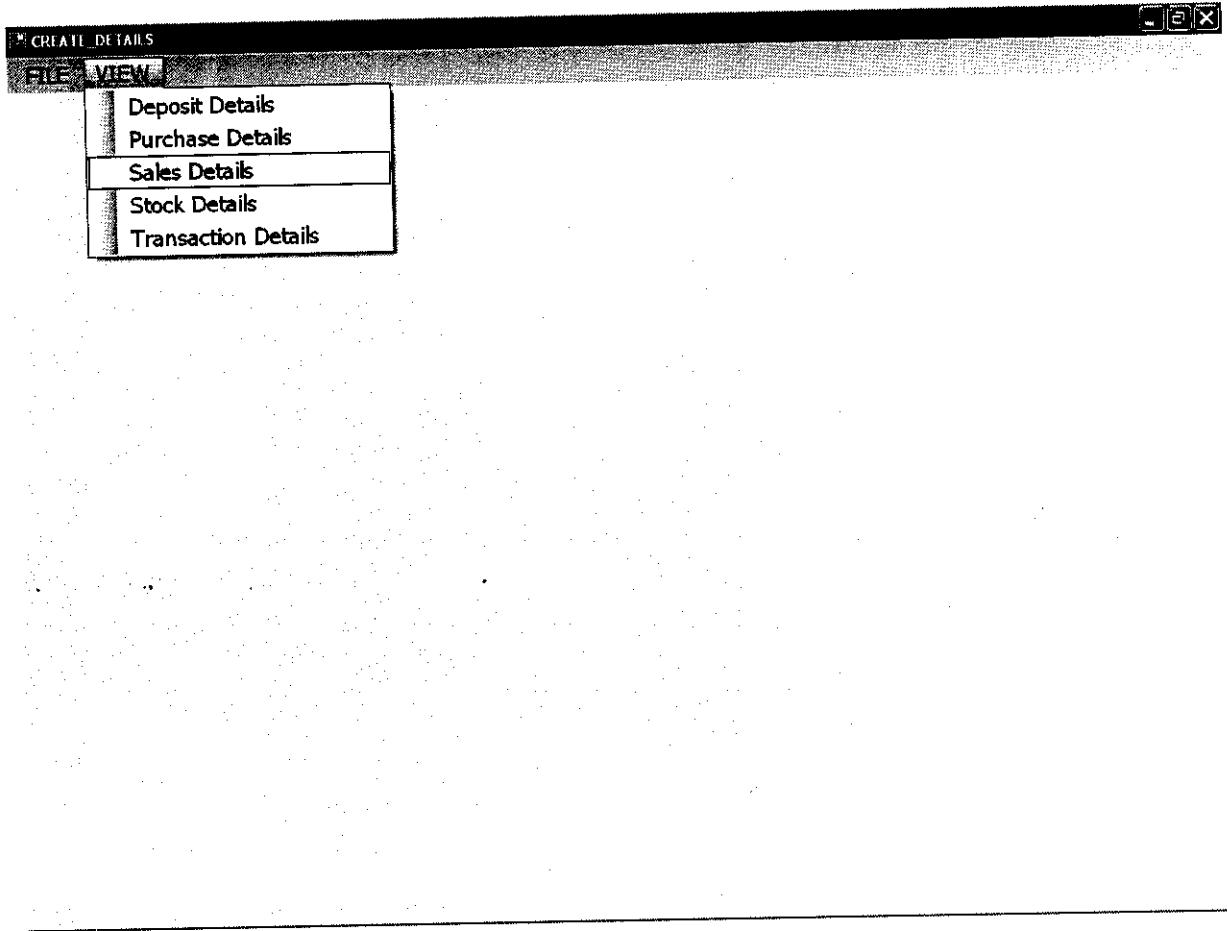
Screen for Viewing Deposit Details

DEPOSITE_DETAILS

DEPOSITID	101
TRANSACTIONID	120
CUSTOMERID	102
CAUTION DEPOSIT	1000
AMOUNT	5000

OK

Deposit Details Screen



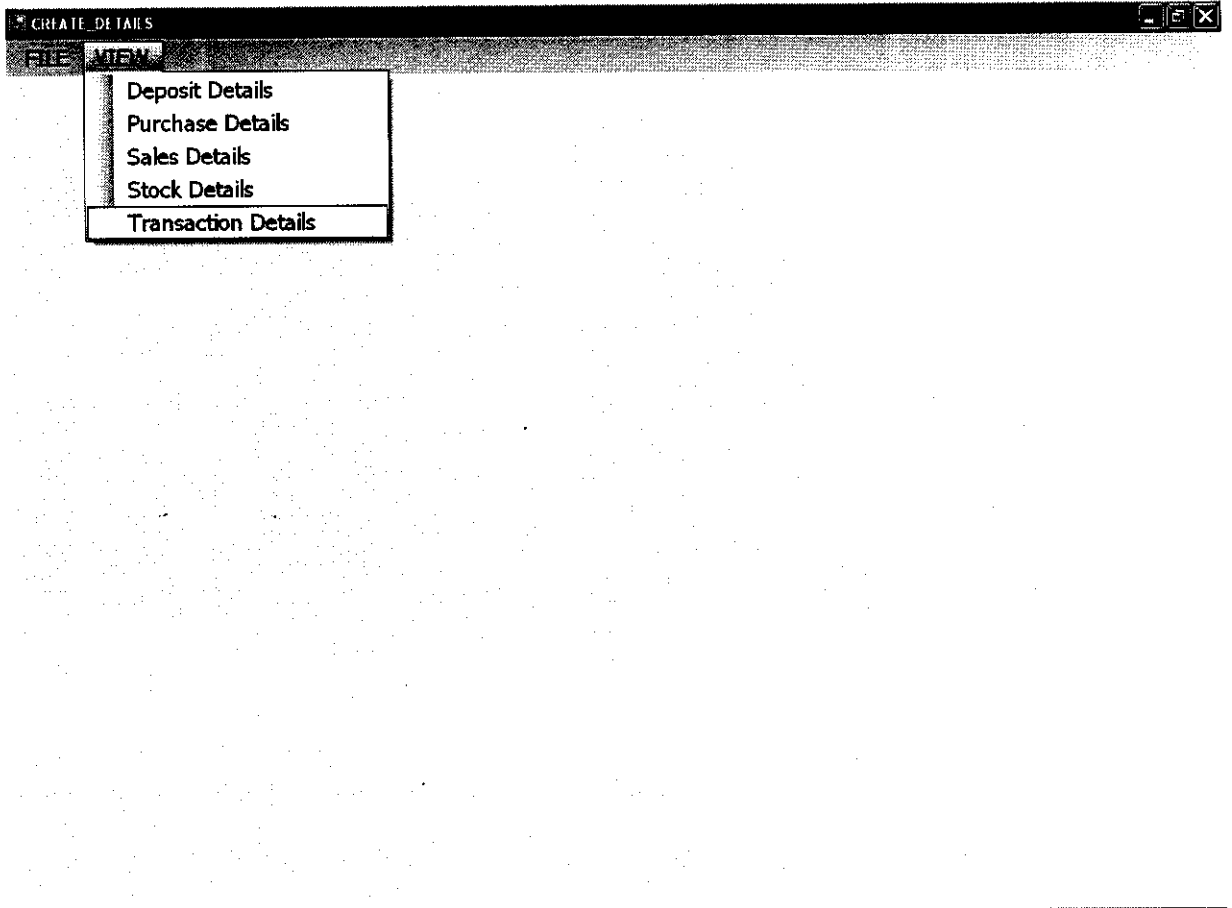
Screen for Viewing Sales Details

SALES_DETAILS

TRANSACTIONID	1011
CUSTOMERID	101
COST	25000

OK

Sales Details Screen



Screen for Viewing Transaction Details

TRANSACTION DETAILS

TRANSACTION ID: 1011

ITEM NUMBER: 2536

QUANTITY: 15

DATE AND TIME: Tuesday, June 10, 2008

OK

June, 2008						
Sun	Mon	Tue	Wed	Thu	Fri	Sat
25	26	27	28	29	30	31
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	1	2	3	4	5
Today: 6/19/2008						

Transaction Details Screen

CHAPTER 9

REFERENCES

- Matthew MacDonald, “The VB.NET: Complete Reference”, TataMacGrawHill Publications, 2002.
- Mirdula Pariha, Jeff Webb, “Microsoft VB.NET”, TataMacGrawHill.
- Paul Nielson, “Microsoft SQL Server”, TataMacGrawHill.
- Rama Ramachandran, Bill Evjen Hollis, Rockford “Professional VB.NET 2003”.
- Wrox Publications Roger.S.Pressman, “Software Engineering”, TataMacGrawHill Publications, 2003.