

**PRODUCT SUPPLIER MANAGEMENT SYSTEM**

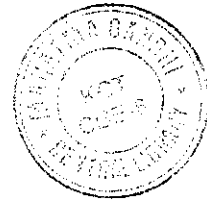
**By**

**S.BALAMURUGAN**

Roll No. 0906MBA1288

Reg. No. 68309200375

**A PROJECT REPORT**



Submitted to the

**FACULTY OF MANAGEMENT SCIENCES**

*in partial fulfillment for the award of the degree*

*of*

**MASTER OF BUSINESS ADMINISTRATION**

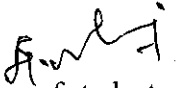


**CENTRE FOR DISTANCE EDUCATION  
ANNA UNIVERSITY CHENNAI  
CHENNAI 600 025**

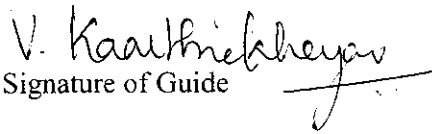
AUGUST 2011

## BONAFIDE CERTIFICATE

Certified that the Project report titled "PRODUCT SUPPLIER MANAGEMENT SYSTEM" is a bonafide work of Mr.S.BALAMURUGAN who carried out the work 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.



Signature of student



Signature of Guide

Name: S.BALAMURUGAN  
Roll No: 0906MBA1288  
Reg. No: 68309200375

Name: Mr. KAARTHIEKHEYAN V  
Designation: Associate Professor  
Address: KCT Business School, Coimbatore



Signature of Project-in-charge

Name: Dr. V.R. NEDUNCHEZHIAN,

Designation: Professor

Address: Kumaraguru College of Technology, Coimbatore



03.08.2011

**TO WHOM SO EVER IT MAY CONCERN**

This is to certify that **S.BALAMURUGAN** (Register No – 68309200375, Roll No – 0906MBA1288) M.B.A Final Year Student, Centre for Distance Education Anna University Chennai at Kumaraguru College of Technology, Coimbatore has successfully completed his project work in the company on "Product Supplier Management System" in our organization during the period between 21.04.2011 and 20.07.2011.

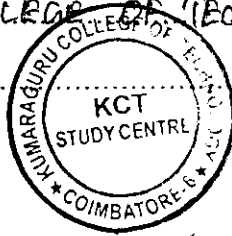
For Essae Digitronics Pvt Ltd,

M.G. Narender  
Design Manager

## Certificate of Viva-voce-Examination

This is to certify that Mr. S. BALAMURUGAN (Roll No. 0906MBA1288; Register No. 68309200375) has been subjected to Viva-voce-Examination on ...10.9.2011...

(Date) at .....9AM..... (Time) at the Study centre.....KUMARAGURU...COLLEGE OF TECHNOLOGY, COIMBATORE.- 49  
.....TAMILNADU.....



Internal Examiner

Name: Dr. V. R. NEDUNCHEZHIAN  
Designation: Prof. KCT Business School  
Address: KCT, Coimbatore - 49  
TamilNadu.

External Examiner

Name: Dr. N. SENTHIL KUNAR  
Designation: Asst. Prof (sr. Grade)  
Address: Dept of Mgmt Studies,  
Anna University, Chennai-2

Coordinator

Study centre

Name: Dr. VIJILA KENNEDY  
Designation: Professor & Director  
Address: KCT Business School

Date: Kumaraguru College of Technology,  
Coimbatore - 49.

## **ABSTRACT**

The project entitled as **Product Supplier Management System** is providing a complete automation on purchase and sales orders in the organization. The purpose of this project is the preparation of a purchase and sales requisition that requires managerial followed by finance department approval and designating whether or not the purchase is a fixed asset. Information on the nature of the purchase, vendors, and delivery dates can be defined and used to prepare and issue a purchase order. When the goods are received they are recorded and verify with purchase orders to complete the workflow.

This project supports the most comprehensive functions of Order Processing. It facilitates the order entry process by allowing generation of Quotations and Estimates, Validating Stock levels, Accepting Orders and Sales Invoices. Sales Returns can also be used in conjunction to the Inventory to increase stock.

This application will have different Category of Products. Sub category of Products and items. This application also has transaction details and reports. The administrator of this application has rights to create product, add items delete items etc. The application will provide information on available stock of the products

## **ACKNOWLEDGEMENT**

I would like to express my profound thanks to Director, Center of Distance Education, Anna University, Chennai, for giving me an opportunity to study in this esteemed institution. I would extend my thanks to The Principal, Kumaraguru College of Technology, Coimbatore.

I would like to thank Prof.Dr.Vijila Kennedy, Director, KCT Business School, Coimbatore & Coordinator, KCT Study Centre.

I would like to thank Mr.A.Senthil Kumar, Asst Professor (Sr.Grade), KCT Business School, Cbe & Counselor, Coimbatore, for his constant support and guidance throughout the project.

I would like to thank my project guide Mr. Kaarthickheyan V, Associate Professor, KCT Business School, Coimbatore, for his guidance in undertaking this project.

I would like to thank the Essae Digitronics (P) Ltd, Bangalore for supporting in undertaking the project.

I thank my parents and the almighty for his blessings.

## TABLE OF CONTENTS

CHAPTER	PARTICULARS	PAGE NO
	<b>ACKNOWLEDGEMENT</b>	<b>i</b>
	<b>ABSTRACT</b>	<b>ii</b>
<b>CHAPTER I</b>	<b>INTRODUCTION</b>	<b>1</b>
1.1	Overview	1
1.2	Need for the Study	1
1.3	About the Organization	2
1.4	Primary Objective	2
1.5	Secondary Objectives	2
1.6	Expected Deliverables	3
1.7	Limitations	3
1.8	Scope for Further Development	3
1.9	Future Enhancements	3
<b>CHAPTER II</b>	<b>SYSTEM ANALYSIS AND SPECIFICATION</b>	<b>4</b>
2.1	Existing System	4
2.1.1	Drawbacks of the Existing System	4
2.2	Proposed System	4
2.2.1	Benefits of the Proposed System	4
2.3	System Specification	5
2.3.1	Hardware Specification:	5
2.3.2	Software Specification:	5
2.4	Software Description	6
2.4.1	Visual Basic.Net	6
2.4.2	Ms-Access	8
2.4.3	Crystal Reports	8
2.5	Testing	9
2.5.1	System Testing	9
2.6	System Implementation	11
<b>CHAPTER III</b>	<b>SYSTEM DESIGN</b>	<b>13</b>
3.2	Design Process	14
3.2.1	System Database Design	14
3.2.2	System Input Design	18
<b>CHAPTER IV</b>	<b>SYSTEM DEVELOPMENT</b>	<b>21</b>
4.1	System Input Screens	21
<b>CHAPTER V</b>		<b>39</b>
	<b>CONCLUSION</b>	
	<b>BIBLIOGRAPHY</b>	
	<b>ANNEXURE - SOURCE CODE</b>	

## LIST OF FIGURES

S.NO	PARTICULARS	PAGE NO
1	Data Flow Diagram	13
2	Login Form	21
3	Main Form	21
4	Supplier Details	22
5	Customer Details	22
6	Product Details	23
7	Item Details	23
8	Item/Date Wise Purchase Report Details	24
9	Purchase Details	24
10	Purchase Return	25
11	Production Details	25
12	Sales Details	26
13	Sales Return	26
14	Material Stock Details	27
15	Product Stock Details	27
16	System Output Reports	28
17	Item Details	29
18	Product Details	30
18	Purchase Detail	31
19	Sales Details	32
20	Material Inventory Reports	33
21	Product Inventory Report	34
22	Purchase Order Report	35
23	Item/Date Wise Details Report	36
24	Purchase Return	37
25	Sales Return	38



# **CHAPTER - 1**

## **INTRODUCTION**

### **1.1 Overview**

Purchase Order Processing module handles the complete spectrum of purchasing tasks, from replenishment analysis through payment. Automatic replenishment processing creates a suggested buy report that is used to determine what, when, and how much to buy to get the best purchase price and achieve your customer service targets. Enter and track your purchase requisitions, purchase orders, and purchase receipts through a set of entry forms designed for consistency and ease of use. For maximum efficiency, a single purchase order can accommodate multiple shipments of the same item using multiple required dates.

The Sales Order Processing module provides everything required to handle orders quickly and efficiently. The order entry program provides an instant snapshot of a customer's credit status and displays backordered and shipped quantities to accurately report the order's current status. As with purchase orders, the sales order system interfaces with the inventory control module to keep quantities accurately up-to-date and also allows committing or releasing future orders. The price quote function enables pricing review and checks item availability and facilitates quickly converting quotes to order.

### **1.2 Need For The Study**

The purpose of this project is the preparation of a purchase and sales requisition that requires managerial followed by finance department approval and designating whether or not the purchase is a fixed asset. Information on the nature of the purchase, vendors, and delivery dates can be defined and used to prepare and issue a purchase order. When the goods are received they are recorded and verify with purchase orders to complete the workflow.

This project supports the most comprehensive functions of Order Processing. It facilitates the order entry process by allowing generation of Quotations and Estimates, Validating Stock levels, Accepting Orders and Sales Invoices. Sales Returns can also be used in conjunction to the Inventory to increase stock.

This application will have different Category of Products, Sub category of Products and items. This application also has transaction details and reports. The administrator of this application has rights to create product, add items delete items etc. The application will provide information on available stock of the products

### **1.3 About the Organization**

This Purchase and Sales Order Processing System project is developed at “Essae Digitronics Private Limited” started at 1996. This Unit has established to satisfying the customer by providing high standards of quality in Electronic Weighing Scales and Systems.

This unit located at Bommasandra Industrial Area manufactures all Electronic Weighing Scales and Systems meant for export, special system products, customized weighing solutions with software and all higher capacity platform scales meant for the domestic market. This facility is well manned with Production Engineers, QC Engineers and Administrative staff. The facilities in this factory include modern equipments for PCB assembly, Wire EDM, welding equipment, shearing equipment, variety of presses. lathes etc. To leverage on the excellent supply base available outside, many processes like injection molding and laser cutting are outsourced.

### **1.4 Primary Objective**

1. To design a software application focusing upon the complete automation on purchase and sales orders in the organization.

### **1.5 Secondary Objectives**

1. To study and to analyze the pitfalls in the existing manual Processing of purchase and sales order in an organization
2. To effectively identify the gap and suggest areas of improvement to eliminate redundancies
3. To deliberate the need of an effective automated system in Processing of purchases and sales orders.

## **1.6 Expected Deliverables**

The “PRODUCT SUPPLIER MANAGEMENT SYSTEM” has been developed to satisfy all proposed requirements. The system is highly scalable and user friendly. The system minimizes the problem arising in the existing manual system and it eliminates the human errors to zero level. Further it helps to

1. It satisfying all user requirements of the organization
2. Purchase and sales order details are stored separately
3. To develop and implement a completely automated system
4. Graphical oriented form designs
5. To generate correct and accurate information with appropriate reports
6. To provide maintenance of master databases
7. To enforce security, integrity, consistency, accuracy and validity of data

## **1.7 Limitations**

1. The blurring of company boundaries can cause problems in accountability, lines of responsibility, and employee morale.
2. Resistance in sharing sensitive internal information between departments can reduce the effectiveness of the software.
3. There are frequent compatibility problems with the various legacy systems of the partners.
4. The system may be over-engineered relative to the actual needs of the customer

## **1.8 Scope For Further Development**

This system is very flexible so that the maintenance and further amendments based on the changing environments and requirements can be made easily. Any change that leads to the system is prevented with security measures

## **1.9 Future Enhancements**

- Interfacing with ERP, Tally and SAP.
- High Security for Databases
- More functions for purchase and sales

## **CHAPTER - 2**

### **SYSTEM ANALYSIS AND SPECIFICATION**

#### **2.1 Existing System**

The existing systems for the flow of operations mentioned above are all manual. All the entries supplier and contractors transactions related to their tender are made manually and ledgers are maintained to record this information.

##### **2.1.1 Drawbacks of the Existing System**

1. Manual work.
2. Inventory maintenance is more complexity.
3. A lot of time consumed for every orders.
4. Separate books maintained for all transactions.
5. More damaged caused for frequent usage of sales and purchase register.

#### **2.2 Proposed System**

The drawbacks, which are faced during existing system, can be eradicated by using the proposed system. The main objective of the existing system is to provide a user-friendly interface. The system, which is proposed, now computerizes all the details that are maintained manually. Once the details are fed into the computer there is no need for various persons to deal with separate sections. Only a single person is enough to maintain all the reports. The security can also be given as per the requirement of the users.

##### **2.2.1 Benefits of the Proposed System**

1. Graphical User Oriented from designs
2. More validation during the data entering forms
3. Avoid empty form filling
4. Very easy to store the orders and maintain the stocks level
5. Updateable inventory
6. Security in accessing a database and reports

## **2.3 System Specification**

### **2.3.1 Hardware Specification:**

Processor	:	Pentium IV
Speed	:	Above 500 MHz
RAM capacity	:	256 MB
Floppy disk drive	:	1.44 MB
Hard disk drive	:	40 GB
Monitor	:	15" Color Monitor
Keyboard	:	Multimedia Keyboard
Mouse	:	Scroll Mouse

### **2.3.2 Software Specification:**

Operating System	:	Windows 2000/XP
Front-end used	:	VB.NET 2005
Back-end used	:	MS Access
Report	:	Crystal Report

## **2.4 Software Description**

### **2.4.1 Visual Basic.Net**

Visual Basic.Net has revolutionized windows programming and with an object based, event driven approach to software designs. Visual basic.Net applications act as a front end to the database. Visual basic.Net application provides the interface between the user and the database. Sophisticated features that make the language truly object oriented and interfaces it with the latest in the database technology.

.NET provides a new, object-oriented API as a set of classes that will be accessible from any programming language. This book describes this framework of classes and provides a reference to what is available and how you can use this framework to write Windows applications in the brave new world of .NET.

Microsoft .NET Framework is a computing platform for developing distributed applications for the Internet. Following are the design goals of Microsoft .NET Framework.

1. To provide a very high degree of language interoperability
2. To provide a runtime environment that completely manages code execution
3. To provide a very simple software deployment and versioning model
4. To provide high-level code security through code access security and strong type checking
5. To provide a consistent object-oriented programming model
6. To facilitate application communication by using industry standards such as SOAP (Simple Object Access Protocol) and XML (Extensible Markup Language).
7. To simplify Web application development

Visual basic.net lets the user to add menus, text boxes, command buttons, option buttons, check boxes, list boxes, scroll bars, and file directory boxes to blank windows. Visual basic.net has many different tools.

## **The Common Language Runtime**

The CLR is the mechanism through which .NET code is executed. It is built upon a single, common language—IL—into which source languages are compiled and includes mechanisms for executing the compiled code. This includes code verification and just-in-time (JIT) compilation, garbage collection and enforcement of security policies, and the provision of profiling and debugging services.

The CLR provides a lot of added value to the programs it supports. Because it controls how a .NET program executes and sits between the program and the operating system, it can implement security, versioning support, automatic memory management through garbage collection, and provide transparent access to system services

### **Important Features:**

1. The application is a graphical user interface.
2. Client-Server architecture benefits picture and image box can be easily handled using bit mapped files and icons.
3. Bit mapped files and icons are used as simple debugging tools.
4. With the advent of .NET, Microsoft has introduced many new technologies that make writing component-based distributed systems easier, more flexible, and more powerful than ever before.
5. It is now easier than it has ever been to write components in any programming language that can interoperate with components on other machines, which may not be Windows-based at all.

### **2.4.2 Ms-Access**

Microsoft Office Access, previously known as Microsoft Access, is a relational database management system from Microsoft that combines the relational Microsoft Jet Database Engine with a graphical user interface and software development tools. It is a member of the Microsoft Office suite of applications and is included in the Professional and higher versions for Windows and also sold separately.

Access stores data in its own format based on the Access Jet Database Engine. It can also import or link directly to data stored in other Access databases, Excel, SharePoint lists, text, XML(Extensible Markup Language), Outlook, HTML, dBase, Paradox, Lotus 1-2-3, or any ODBC (Open Database Connectivity)-compliant data container including Microsoft SQL Server, Oracle, My SQL and Postgre SQL. Software developers and data architects can use it to develop application software and non-programmer "power users" can use it to build simple applications. Like other Office applications Access is supported by Visual Basic for Applications, an object-oriented programming language that can reference a wide variety of objects, including DAO (Data Access Objects) and ActiveX Data Objects, and many other ActiveX components provided by Microsoft or by third parties. Visual objects used in forms and reports expose their methods and properties gracefully in the VBA(visual basic applications) programming environment, and a huge selection of Windows operating system functions can be declared and called from VBA code modules, making Access a rich programming environment.

### **2.4.3 Crystal Reports**

Crystal Reports allows users to graphically design data connections and report layout. In the Database Expert, users can select and link tables from a wide variety of data sources, including Microsoft Excel spreadsheets, Oracle databases, Business Objects Enterprise business views, and local file system information. Fields from these tables can be placed on the report design surface, and can also be used in custom formulas, using either BASIC or Crystal's own syntax, which is then placed on the design surface.

Crystal Reports for Visual Studio .NET software provides developers with a fast and productive way to create and integrate presentation-quality reports into



applications, without leaving the familiar Visual Studio development environment. It also enables them to easily create interactive reports for enterprise, Web, and smart client applications that scale to meet the demands of end users.

Crystal Reports for Visual Studio .NET enables you to

1. Easily upgrade to Crystal Reports for powerful developer features
2. Create reports in your chosen application development environment
3. Save time using powerful report creation, integration, and delivery tools
4. Deliver interactive, graphical reports on any device through an XML Web services model
5. Distribute thoroughly-formatted reports in rich-client Windows environments
6. Access reports contained within the SAP Business Objects Enterprise framework when upgrading to Crystal Reports Server or SAP Business Objects Enterprise.

## **2.5 Testing**

Testing is a series of different tests that whose primary purpose is to fully exercise the computer based system. Although each test has a different purpose, all work should verify that all system element have been properly integrated and performed allocated function. Testing is the process of checking whether the developed system works according to the actual requirement and objectives of the system.

The philosophy behind testing is to find the errors. A good test is one that has a high probability of finding an undiscovered error. A successful test is one that uncovers the undiscovered error. Test cases are devised with this purpose in mind. A test case is a set of data that the system will process as an input. However the data are created with the intent of determining whether the system will process them correctly without any errors to produce the required output.

### **2.5.1 System Testing**

It is the stage of implementation, which ensures that system works accurately and effectively before the live operation commences. It is a confirmation that all are correct and opportunity to show users that the system must be tested with text data

and show that the system will operate successfully and produce expected results under expected conditions.

Software testing is a crucial element of software quality assurance and represents the unlimited review of specification, design and coding. Testing represents an interesting anomaly for the software. During the earlier definition and development phase, it was attempted to build the software from an abstract concept to tangible implementation

### **Testing Objectives:**

1. Testing is the process of analyzing program with the intent of discovering errors.
2. A good test case is one that has high probability of finding undiscovered error.

### **Types of Testing:**

1. Unit testing
2. Integration testing
3. Validation testing
4. Output testing

### **Unit Testing:**

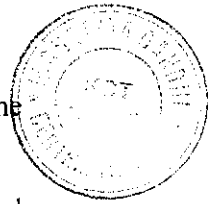
All modules were tested and individually as soon as they were completed and were checked for their correct functionality.

In this project test all modules individually tested during the running process. This testing truncate all errors this project.

### **Integration Testing:**

The entire project was split into small program; each of these single programs gives a frame as an output. These programs were tested individually; at last all these programs where combined together by creating another program where all these constructors were used. It give a lot of problem by not functioning is an integrated manner.

The user interface testing is important since the user has to declare that the arrangements made in frames are convenient and it is satisfied. When the frames



where given for the test, the end user gave suggestion. Based on their suggestions the frames were modified and put into practice.

This testing used to test the all integration coding such as common module and database connections. All other integration coding also be tested.

### **Validation Testing:**

At the culmination of the black box testing software is completely assembled as a package. Interfacing errors have been uncovered and corrected and a final series of test i.e., Validation succeeds when the software function in a manner that can be reasonably accepted by the customer.

This testing is used for checking all user input forms values in this project. All textboxes are tested under the validation testing.

### **Output Testing:**

After performing the validation testing the next step is output testing of the proposed system. Since the system cannot be useful if it does not produce the required output. Asking the user about the format in which the system is required tests the output displayed or generated by the system under consideration. Here the output format is considered in two ways. One is on screen and another one is printed format. The output format on the screen is found to be corrected as the format was designed in the system phase according to the user needs. And for the hardcopy the output comes according to the specifications requested by the user.

## **2.6 System Implementation**

The purpose of System Implementation can be summarized as follows:

It making the new system available to a prepared set of users (the deployment), and positioning on-going support and maintenance of the system within the Performing Organization (the transition). At a finer level of detail, deploying the system consists of executing all steps necessary to educate the Consumers on the use of the new system. placing the newly developed system into production, confirming that all data required at the start of operations is available and accurate, and validating that business functions that interact with the system are functioning properly. Transitioning the system support responsibilities involves changing from a system

development to a system support and maintenance mode of operation, with ownership of the new system moving from the Project Team to the Performing Organization.

List of System implementation is the important stage of project when the theoretical design is tuned into practical system.

The main stages in the implementation are as follows:

1. Planning
2. Training
3. System testing and
4. Changeover Planning

Planning is the first task in the system implementation. Planning will decide the method and the time scale to be adopted. At the time of implementation of any system people from different departments and system analysis involve. They are confirmed to practical problem of controlling various activities of people outside their own data processing departments. The line managers controlled through an implementation coordinating committee. The committee considers ideas, problems and complaints of user department, it must also consider:

1. The implication of system environment
2. Self selection and allocation form implementation tasks
3. Consultation with unions and resources available
4. Standby facilities and channels of communication

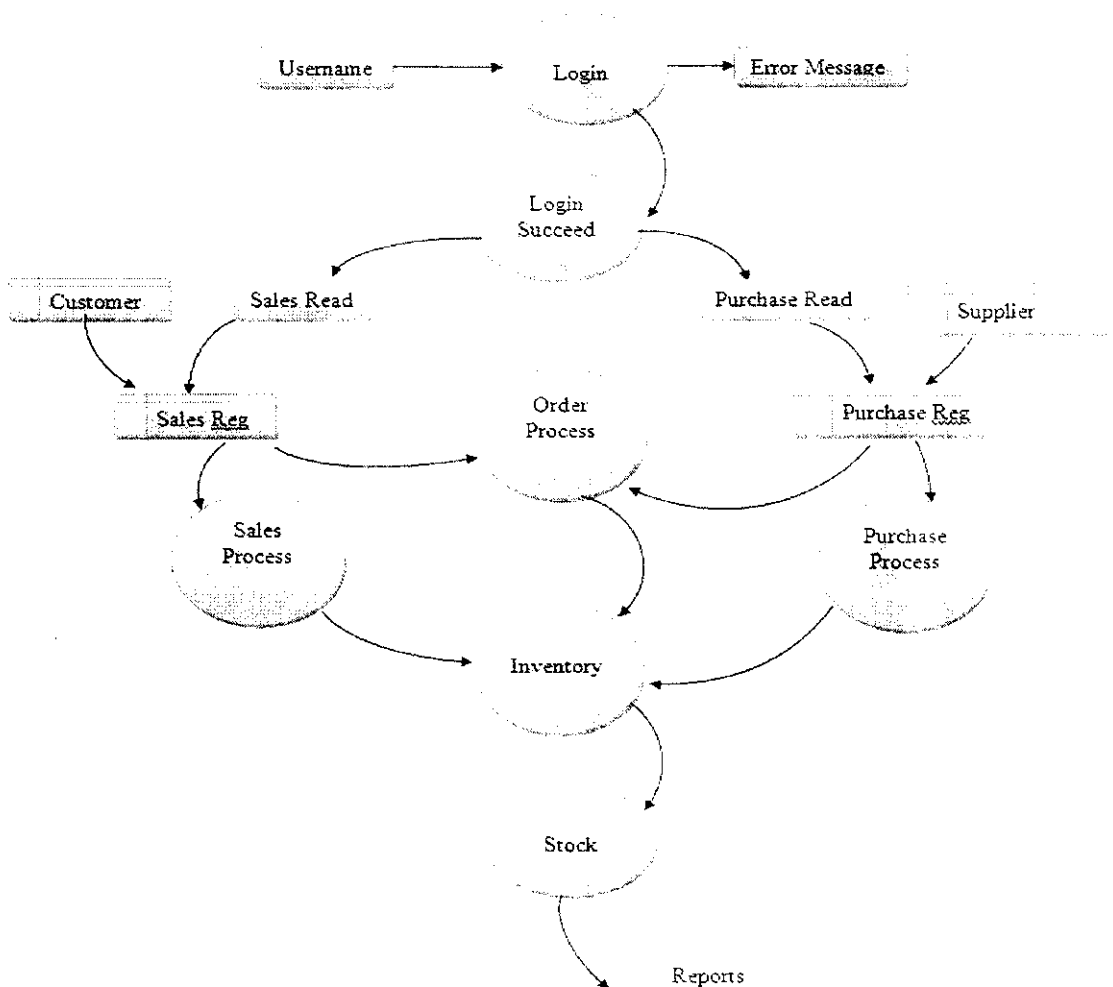
## CHAPTER - 3

### SYSTEM DESIGN

In this chapter, concepts associated with term structured system and how they are implemented in the project has been dealt with the tools used for structure system analysis are,

#### 3.1 Data Flow Diagram

A data flow diagram also known as “**bubble chart**” has the purpose of clarifying system requirements and identifying major transformation that will become program in system design. So it is the starting point of the phase that functionally decomposes the requirement specification down to the lowest level of details. A DFD contains series of bubbles joined by lines.



## **3.2 Design Process**

System design is the process of planning a new system to complement or altogether replace the old system. The purpose of the design phase is the first step in moving from the problem domain to the solution domain. The design of the system is the critical aspect that affects the quality of the software. System design is also called top-level design. The design phase translates the logical aspects of the system into physical aspects of the system.

### **3.2.1 System Database Design**

Database design is the process of producing a detailed data model of a database. This logical data model contains all the needed logical and physical design choices and physical storage parameters needed to generate a design in a Data Definition Language, which can then be used to create a database. A fully attributed data model contains detailed attributes for each entity.

The term database design can be used to describe many different parts of the design of an overall database system. Principally, and most correctly, it can be thought of as the logical design of the base data structures used to store the data. In the relational model these are the tables and views. In an object database the entities and relationships map directly to object classes and named relationships. However, the term database design could also be used to apply to the overall process of designing, not just the base data structures, but also the forms and queries used as part of the overall database application within the database management system (DBMS).

### **Database Table Design process**

The process of doing database design generally consists of a number of steps which will be carried out by the database designer. Not all of these steps will be necessary in all cases. Usually, the designer must:

1. Determine the relationships between the different data elements
2. Superimpose a logical structure upon the data on the basis of these relationships.

Within the relational model the final step can generally be broken down into two further steps that of determining the grouping of information within the system, generally determining what are the basic objects about which information is being

stored, and then determining the relationships between these groups of information, or objects. This step is not necessary with an Object database.

The tree structure of data may enforce a hierarchical model organization, with a parent-child relationship table. An Object database will simply use a one-to-many relationship between instances of an object class. It also introduces the concept of a hierarchical relationship between object classes, termed inheritance.

## Data Storage Tables

### 1. Customer Table

Field Name	Data Type	Description
Cust id	Number	Primary key
Customer Name	Text	
Customer Address	Text	
Cperson	Text	
Cno	Text	
Mail	Number	

### 2. Inventory Table

Field Name	Data Type	Description
ItemID	Number	Foreign Key
Item name	Text	
Quantity	Number	
Rlevel	Number	

### 3. Product Inventory Table

Field Name	Data Type	Description
ItemID	Number	Foreign Key
Item name	Text	
Quantity	Number	
PDate	Date/Time	

#### 4. Item Details

Field Name	Data Type	Description
Item Id	Number	Primary Key
Item Group	Text	
Item Name	Text	
Uom	Text	

#### 5. Product Details

Field Name	Data Type	Description
Pid	Number	Primary key
Pgroup	Text	
Pname	Text	
Uom	Text	

#### 6. Purchase Details

Field Name	Data Type	Description
Porderno	Number	Primary key
Supplierid	Number	Foreign key
Sname	Text	
Pdate	Date/Time	
Mname	Text	
Uom	Text	
Quantity	Number	
Uprice	Number	
Price	Number	
Tax	Number	
TPrice	Number	

#### 7. Purchase Return

Field Name	Data Type	Description
Pid	Number	Foreign key
Porderno	Number	Foreign key
Prdate	Date/time	
Qty	Number	
Price	Number	



## 8. Purchase Order

Field Name	Data Type	Description
PONo	Number	Foreign key
Sname	Text	
SAddress	Text	
Item Id	Number	Foreign key
Item name	Text	
Quantity	Number	
Uprice	Number	
Price	Number	
TaxRate	Number	
Tax	Number	
PTax	Number	

## 9. Purchase Report

Field Name	Data Type	Description
PName	Text	
Sname	Text	
Pdate	Date/Time	
Quantity	Number	
Ivno	Number	Foreign key
Price	Number	
Tax	Number	
TPrice	Number	

## 10. Sales Table

Field Name	Data Type	Description
Invoice no	Number	Primary key
Cust id	Number	Foreign key
Cname	Text	
Sdate	Data/time	
Sname	Text	
Uom	Text	
Quantity	Number	
Uprice	Number	
Price	Number	

## 11. Sales Return

Field Name	Data Type	Description
Prid	Number	Foreign key
Porderno	Number	Foreign key
Prdate	Date/time	
Qty	Number	
Price	Number	

## 12. Supplier Details

Field Name	Data Type	Description
SupplierId	Number	Primary key
Supbname	Text	
Address	Text	
Mtype	Text	
Cperson	Text	
Cno	Text	
Mailaddress	Text	
Matname	Text	

### 3.2.2 System Input Design

Input design is one of the most important phases of the system design. Input design is the process where the input received in the system are planned and designed, so as to get necessary information from the user, eliminating the information that is not required. The aim of the input design is to ensure the maximum possible levels of accuracy and also ensures that the input is accessible that understood by the user.

The input design is the part of overall system design, which requires very careful attention. If the data going into the system is incorrect then the processing and output will magnify the errors.

The input design also determines the user to interact efficiently with the system. Input design is a part of overall system design that requires special attention because it is the common source for data processing error. The goal of designing input data is to make entry easy and free from errors.

The main modules exist in these projects are listed below

1. Supplier Details
2. Customer Details
3. Purchase Details
4. Purchase Returns
5. Production Details
6. Sales Details
7. Sales Returns
8. Stock Details

### **Supplier Details**

This module is used to store the supplier details in the company. The details retrieved during the purchase orders.

### **Customer Details**

This module is used to store the customer details in the company. The details retrieved during the purchase sales orders.

### **Purchase Details**

This module is used to store the all purchase level entries in the company. It will apply the stock inventory

### **Purchase Returns**

This module stores the purchase returned goods. Damaged or low quality cottons are returned during the purchase process.

### **Production Details**

This module is dealing a process of to create raw materials into a finished product. It takes a shift details and raw goods.

**Sales details**

This module is used to store the all sales level entries in the company. It will applied the stock inventory

**Sales Returns**

Sold products are returned back to us by our customers. So these details are maintained by this module.

**Stock Details**

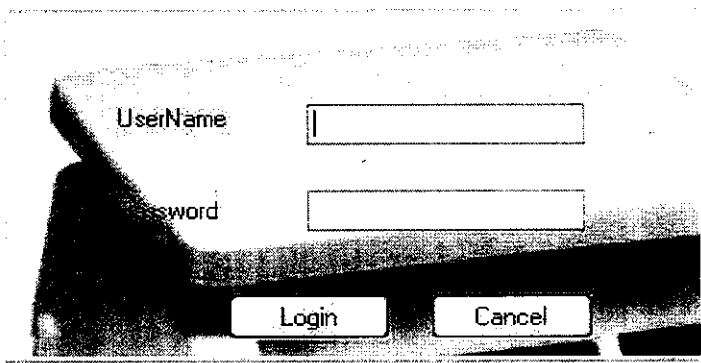
This is the main module of our project. This is a module it automatically generated. All stocks are updated during the purchase and sales process.

# CHAPTER - 4

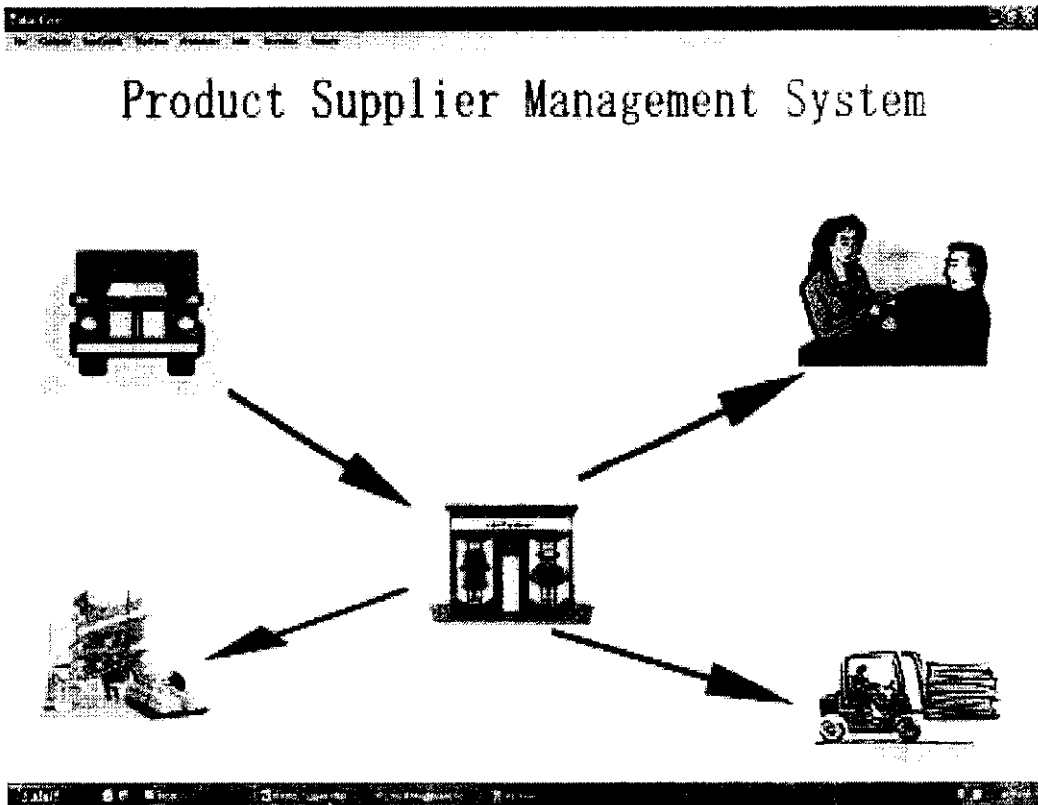
## SYSTEM DEVELOPMENT

### 4.1 System Input Screens

#### 4.1.1 Login Form



#### 4.1.2 Main Form



### 4.1.3 Supplier Details

SupplierDetails ✕

## Supplier Details

Personal Details

SupplierID	<input type="text"/>	<input type="button" value="Find"/>	Contact Person	<input type="text"/>
Supplier Name	<input type="text"/>		Contact No	<input type="text"/>
Address	<input type="text"/>		Mail Address	<input type="text"/>

### 4.1.4 Customer Details

CustomerDetails ✕

## CUSTOMER DETAILS

Personal Details

CustomerID	<input type="text"/>	<input type="button" value="Find"/>	Contact Person	<input type="text"/>
CustomerName	<input type="text"/>		Contact No	<input type="text"/>
Address	<input type="text"/>		Mail Address	<input type="text"/>

### 4.1.5 Product Details

**Product Details**

ProductId  Product Group

Product Name  UOM

### 4.1.6 Item Details

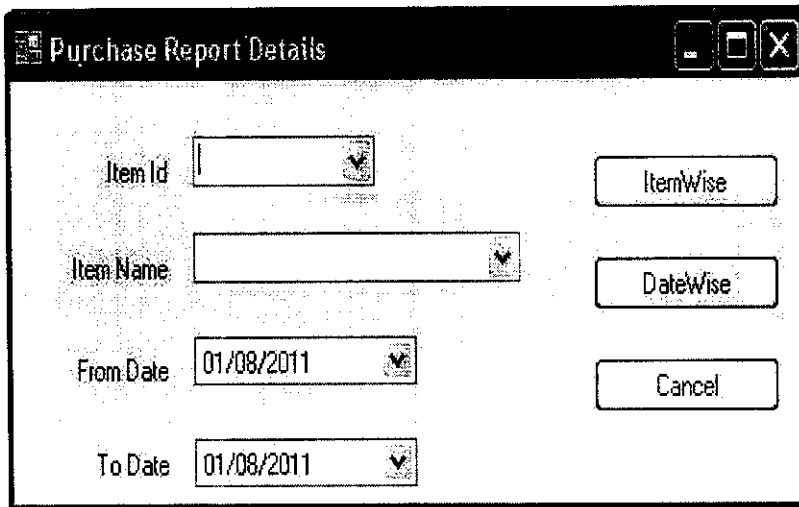
**ItemDetails**

**Item Details**

ItemID  Item Group

Item Name  UOM

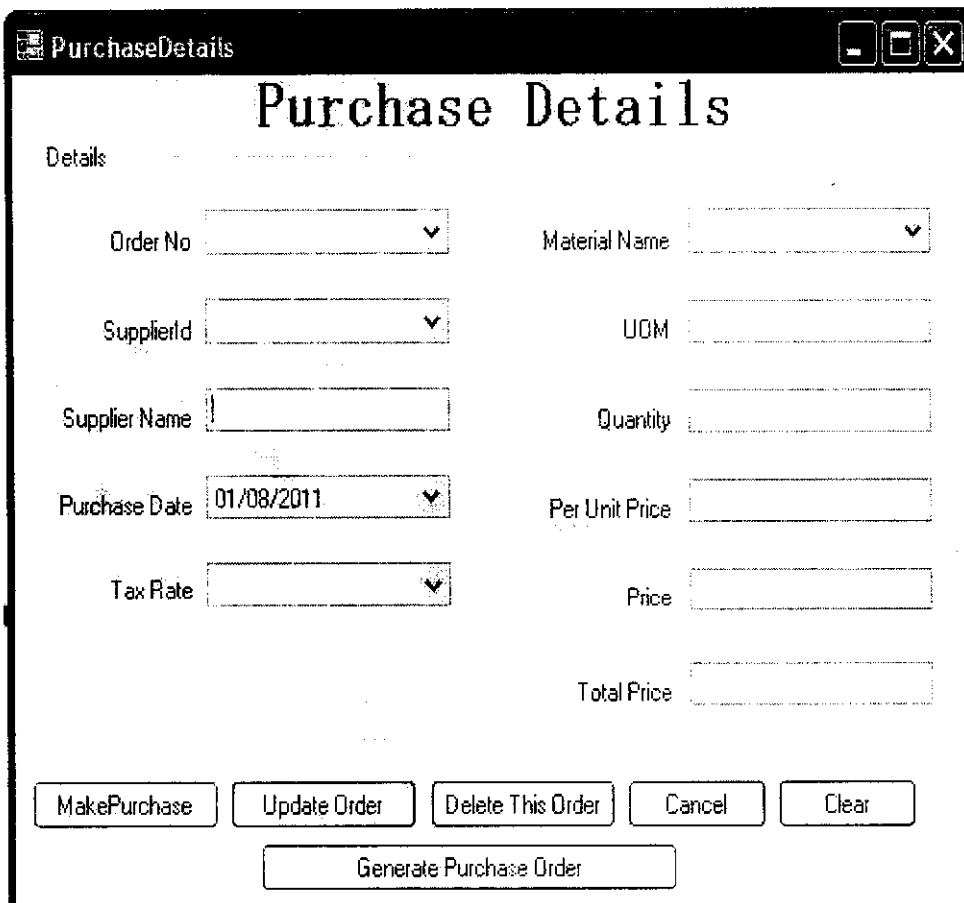
### 4.1.7 Item/Date Wise Purchase Report Details



The screenshot shows a window titled "Purchase Report Details" with standard window controls (minimize, maximize, close) in the top right corner. The window contains several input fields and buttons:

- Item Id:** A dropdown menu.
- Item Name:** A dropdown menu.
- From Date:** A date field containing "01/08/2011".
- To Date:** A date field containing "01/08/2011".
- Buttons:** "ItemWise", "DateWise", and "Cancel".

### 4.1.8 Purchase Details



The screenshot shows a window titled "PurchaseDetails" with standard window controls (minimize, maximize, close) in the top right corner. The window title is "Purchase Details". Below the title, the word "Details" is displayed. The form contains the following fields and buttons:

- Order No:** A dropdown menu.
- Material Name:** A dropdown menu.
- SupplierId:** A dropdown menu.
- UOM:** A text input field.
- Supplier Name:** A text input field.
- Quantity:** A text input field.
- Purchase Date:** A date field containing "01/08/2011".
- Per Unit Price:** A text input field.
- Tax Rate:** A dropdown menu.
- Price:** A text input field.
- Total Price:** A text input field.
- Buttons:** "MakePurchase", "Update Order", "Delete This Order", "Cancel", "Clear", and "Generate Purchase Order".



### 4.1.9 Purchase Return

**PurchaseReturns**

**PurchaseReturns**

Details

Order No	<input type="text"/>	Item Name	<input type="text"/>
SupplierId	<input type="text"/>	UOM	<input type="text"/>
Supplier Name	<input type="text"/>	Quantity	<input type="text"/>
PurchaseDate	<input type="text" value="12/07/2011"/>		
Purchase Return Date	<input type="text" value="12/07/2011"/>	ReturnId	<input type="text" value="2"/>
Quantity	<input type="text"/>	Rate	<input type="text"/>

### 4.1.10 Production Details

**Production**

**Production**

GroupBox1

Date	<input type="text" value="01/08/2011"/>	UOM	<input type="text" value="Nos"/>
SHIFT	<input type="text"/>	Used Nos	<input type="text"/>
Item	<input type="text"/>	Product Name	<input type="text"/>
		Product Quantity	<input type="text"/>

### 4.1.11 Sales Details

**SalesDetails** [min] [max] [close]

## Sales Details

Details

Invoice No	<input type="text"/>	Product Name	<input type="text"/>
CustomerId	<input type="text"/>	UOM	<input type="text"/>
Customer Name	<input type="text"/>	Quantity	<input type="text"/>
SalesDate	01/08/2011	Per Unit Price	<input type="text"/>
Tax Rate	<input type="text"/>	Price	<input type="text"/>
		Total Price	<input type="text"/>

### 4.1.12 Sales Return

**SalesReturns** [min] [max] [close]

## Sales Returns

Details

Invoice No	<input type="text"/>	Product Name	<input type="text"/>
CustomerId	<input type="text"/>	UOM	<input type="text"/>
Customer Name	<input type="text"/>	Quantity	<input type="text"/>
SalesDate	12/40/2011		
Sales Return Date	12/07/2011	ReturnId	1
Quantity	<input type="text"/>	Rate	<input type="text"/>

### 4.1.13 Material Stock Details

StockDetails

## Stock Details

Item wise Details:   All Details

	ItemID	ItemName	Quantity	Level
▶	34858	ESPD Junction B...	80	2
*				

### 4.1.14 Product Stock Details

Product Stock Details

## Stock Details

Product:   All Details

	ItemID	ItemName	Quantity	PDate
▶		120ton Rail In M...	1	7/31/2011
*				

## 4.2 System Output Reports

The output design was done so that results of processing could be communicated to the users. The various outputs have been designed in such a way that they represent the same format that the office and management used to.

Computer output is the most important and direct source of information to the user. Efficient, intelligible output design should improve the systems relationships with the user and help in decision making. A major form of output is hardcopy from the printer

1. Screen Output
2. Output to be stored as files in storage media
3. Hard copy of the output

Output requirements are designed during system analysis. A good starting point for the output design is the Data Flow Diagram (DFD). Human factors reduce issues for design involves addressing internal controls to ensure readability

It consists of following types of reports:

1. Item Reports
2. Purchase Reports
3. Purchase Return Reports
4. Sales Reports
5. Sales Return Reports
6. Material Inventory Reports
7. Item/Date Wise Purchase Reports
8. Product Reports
9. Product Inventory Reports
10. Purchase Order Reports

# System Output Reports

## 4.2.1 Item Details

Item Details Report			
			R02811
Item#	ItemGroup	ItemName	Quantity
1	TrackSole	2.4mX3m MS Platform	1
2	TrackSole	Top Plate - Type-1 (Welded to Platform)	1
3	TrackSole	Top Plate - Type-2	1
4	TrackSole	40 lbs DESB Load Cell	1
5	TrackSole	40 lbs Top Frame	1
6	TrackSole	40 lbs Base Plate Assembly	1
7	TrackSole	40 lbs Tension Link	1
8	TrackSole	M20x70 Hex HT Bolt with Spring Washer	1
9	TrackSole	M12x50 Hex HT Bolt with Nut & Spring Washer	1
10	TrackSole	M12x70 Hex HT Bolt with Nut & Spring Washer	1
11	TrackSole	M12x20 Hexagonal Bolt with Nut	1
12	TrackSole	Essex Trans-Flat Shaker	1
13	TrackSole	Forge Casting Cable (see 13) note of the Length	1
14	TrackSole	Copper Cable with legs 40 inch	1
15	TrackSole	Combined 16:2 ESDP with an 800000000 Assy	1
16	TrackSole	Digital Indicator Assy (Transducer Model 49201)	1
17	TrackSole	UB300x7.0m Long	1
18	TrackSole	200k Plate 300mmx300mm	1
19	TrackSole	200k Plate 300mmx300mm	1
20	TrackSole	100k Plate 300mmx300mm	1
21	TrackSole	12mm HR Plate (1.25mX3.0m)	1
22	TrackSole	12mm HR Plate (1.25mX3.0m)	1
23	TrackSole	102x20x3.0m Long	1

## 4.2.2 Product Details

Product Details			
Product Id	Product Group	Product Name	LOM
3	RIM	100 RAIL IN MOTION	Yes
1	TrackScale	1R2-100: Motion Weighted	Yes
2	TrackScale	2Axle (CAPACITY: 50)	Yes
4	TrackScale	16x3m-BL (CAPACITY: 100)	Yes
5	TrackScale	3mX1 (CAPACITY: 60)	Yes
6	TrackScale	16x1m-BL (CAPACITY: 100)	Yes
7	TrackScale	16x100 TRACK SCALE	Yes
8	TrackScale	2.4x3.5m AXLE WEIGHT BRIDGE	Yes
9	TrackScale	3m-60: Track Scale	Yes

### 4.2.3 Purchase Detail

Purchase Report									
4/12/2011									
Order No	Supplier Name	Date	Material Name	Quantity	Unit Price	Price	Tax	T.Price	
34858	Jindal Steel Plant Pvt	07/31/2011	[REDACTED]	100	1320	132000	12.50	149500	
435634	Jindal Steel Plant Pvt	03/08/2011	[REDACTED]	6	13000	60000	12.50	67500	
45675	Jindal Steel Plant Pvt	03/08/2011	[REDACTED]	4	5800	23200	12.50	26100	
						187700		205600	

## 4.2.4 Sales Details

Map Legend

**Sales Report** 03/2011

---

InvoiceNo	Cust Id	Cust Name	Date	Product Name	Quantity	U Price	Price	Tax	TPrice
34534	1	Geshu Martin Pvt Ltd	07/01/2011	120mm Flat Ir- Moben	1	130000	130000	12.50	146250
456754	1	Geshu Martin Pvt Ltd	02/02/2011	120mm Flat Ir- Moben	1	130000	130000	12.50	146250
67857	1	Geshu Martin Pvt Ltd	01/03/2011	120mm Flat Ir- Moben	2	124500	249000	12.00	277700
							509000		510200

Page 1 of 1



## 4.2.5 Material Inventory Reports

### Inventory Report

7/12/2011

Item ID	ItemName	ReorderLevel
1	2.4mX3m MS Platform	3
2	Top Plate - Type-I (Welded to Platform)	3
3	Top Plate - Type-II	3
4	40 klb DESB Load Cell	3
5	40 klb Top Frame	3
6	40 klb Base Plate Assembly	3
7	40 klb Tension Link	3
8	M20x70 Hex H.T Bolt with Spring Washer	3
9	M12x50 Hex H.T Bolt with Nut & Spring Washer	3
10	M12x70 Hex H.T Bolt with Nut & Spring Washer	3
11	M8x30 Hexagonal Bolt with Nut	3
12	Essae Trans Peel Sticker	3
13	Surge Earthing Cable lugs @ ends of 2m Length	3
14	Copper Cable with lugs @ ends	3
15	Combined JB & ESPD with cable connector Assy	3
16	Digital Indicator Assy(Transformer Model -DS451 )	3
17	UB300x7.6m Long	3
18	20thk Plate 3000mmx290mm	3
19	20thk Plate 3000mmx400mm	3
20	16thk Plate 3200mmx400mm	3
21	12mm HR Plate (1.25mX3.0m)	3
22	12mm HR Plate (0.5mX3.0m)	3
23	UB250x3.2m Long	3
24	En-B Bush	3
25	Loadcell - Axle Type	3

## 4.2.6 Product Inventory Report

Product Inventory Report		
1/1/2011		
S.No	Product Name	Quantity
1	125 GAL IN WGT ON	3
2	16x3-1500 Monitor Amplifier	0
3	0.5x1m CAPACITY 90-	4
4	0.4-0.5 Trunk Scale	1

## 4.2.7 Purchase Order Report

Page 4/27

# Purchase Order

03/20/11

PO # 43434

**Ship To**

Jindal Steel Works Pvt Ltd

Batli, Orissa

Item No	Description	Quantity	Unit Price	Price
1	304 SS Plate	1000	10000	1000000
<b>Sub Total</b>				1000000
<b>Tax Rate</b>				10%
<b>Tax</b>				100000
<b>Grand Total</b>				1100000

**Other Comments or Special Instructions**

**Place Bangalore**

**Date 03/20/11**

**Approved By**

© 2011 SAP AG. All rights reserved. SAP and SAP logo are registered trademarks of SAP AG in Germany and several other countries.

## 4.2.8 Item/Date Wise Details Report

Item/Date Wise Report								
08/31/2011								
Order No	Item Name	Supplier Name	Date	Quantity	Uprice	Price	Tax	TPrice
439834	60M X 60M Rail	Jindal Steel Plant Pvt	Jd0008/2011	3	12000	36000	12.00	40320
43073	Top Plate Type	Jindal Steel Plant Pvt	Jd0008/2011	4	3690	14760	12.00	16632
						50760		56952

## 4.2.9 Purchase Return

Purchase Return Report					6/7/2011
Product Id	Product Order No	Date	Quantity	Price	
1	34856	07/31/2011	10	11250	
2	34856	07/31/2011	10	11250	
3	45015	08/03/2011	4	21804	
				44304	

## 4.2.10 Sales Return

Sales Return Report				
NOV 2011				
S. No	Invoice No	Return Date	Quantity	Price
1	34534	07/31/2011	1	140250
2	34534	07/31/2011	1	140250
3	67967	08/01/2011	2	555520
4	67967	08/01/2011	1	277760
				1123780

## **CHAPTER - 5**

### **CONCLUSION**

The “PRODUCT SUPPLIER MANAGEMENT SYSTEM” has been developed to satisfy all proposed requirements. The system is highly scalable and user friendly. Almost all the system objectives have been met. The system has been tested under all criteria. The system minimizes the problem arising in the existing manual system and it eliminates the human errors to zero level.

The system reduces the manual work of maintenance of the records. It has also resulted in quick retrieval and reference of required information, which is vital to the degrees of the organization. The entire system is documented and can be easily understood by the end users. The form are very user friendly and also easy to handle even by the beginners with very little effort and guidance.

## BIBLIOGRAPHY

- Elias M awas, “System Analysis and Design ” Galgotia publication, Second Edition 1996
- Harvey M. Deitel , Paul J. Deitel, Tem R. Nieto “Visual Basic .NET How to Program” 2nd Edition
- Rajib mall, “Fundamentals of Software Engineering”, Prentice-Hall of India Pvt.Ltd, 2004
- Richard Fairley, ”SOFTWARE ENGINEERING CONCEPTS”,Tata Mc Graw Hill Publication, Second Edition,1997
- Distributed .NET Programming in VB .NET by Tom Barnaby
- Professional VB.NET, 2nd Edition by Fred Barwell, et al
- The .NET Languages: A Quick Translation Guide by Brian Bischof
- Programming VB.NET: A Guide for Experienced Programmers by Gary Cornell, Jonathan Morrison
- Learning Visual Basic.NET Through Applications by Clayton Crooks II
- Visual Basic .NET How to Program (2nd Edition) by Harvey M. Deitel, Paul J. Deitel, Tem R. Nieto



## ANNEXURE

### SOURCE CODE

#### Purchase Details

Imports System.Data

Imports System.Data.OleDb

Public Class PurchaseDetails

Dim conn As OleDbConnection

Dim comm, comm1 As OleDbCommand

Dim adp As OleDbDataAdapter

Dim ds As DataSet

Dim read, read1, read2 As OleDbDataReader

Dim a, b, c, i As Integer

Dim tprice As Double

Dim tax As Double

Dim saddress As String

Private Sub Button1\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

    If ComboBox1.Text = "" Then

        MsgBox("Please Select a OrderNo", MsgBoxStyle.Information, "Purchase Details")

        Exit Sub

    End If

    If ComboBox3.Text = "" Then

        MsgBox("Please Select a SupplierId", MsgBoxStyle.Information, "Purchase Details")

        Exit Sub

    End If

    If ComboBox2.Text = "" Then

        MsgBox("Please Select a MaterialName", MsgBoxStyle.Information, "Purchase Details")

        Exit Sub

    End If

    If ComboBox4.Text = "" Then

        MsgBox("Please Select a TaxRate", MsgBoxStyle.Information, "Purchase Details")

        Exit Sub

    End If

    If TextBox7.Text = "" Then

        MsgBox("Please Enter the UOM", MsgBoxStyle.Information, "Purchase Details")

```

Exit Sub
End If

If TextBox6.Text = "" Then
    MsgBox("Please Enter the Quantity", MsgBoxStyle.Information, "Purchase
Details")
    Exit Sub
End If

If TextBox9.Text = "" Then
    MsgBox("Please Enter the Unit Price", MsgBoxStyle.Information, "Purchase
Details")
    Exit Sub
End If

tprice = TextBox8.Text + ((TextBox8.Text * ComboBox4.Text) / 100)
TextBox1.Text = tprice

dataBase()

' MsgBox("The Insert Query" & "insert into PurchaseDetails values(" &
ComboBox1.Text & "," & ComboBox3.Text & "," & TextBox2.Text & "," &
DateTimePicker1.Text & "," & ComboBox2.Text & "," & TextBox7.Text & "," &
TextBox6.Text & "," & TextBox9.Text & "," & TextBox8.Text & ")")
comm = New OleDbCommand("insert into PurchaseDetails values(" &
ComboBox1.Text & "," & ComboBox3.Text & "," & TextBox2.Text & "," &
DateTimePicker1.Text & "," & ComboBox2.Text & "," & TextBox7.Text & "," &
TextBox6.Text & "," & TextBox9.Text & "," & TextBox8.Text & "," &
ComboBox4.Text & "," & TextBox1.Text & ")")
comm.ExecuteNonQuery()
conn.Close()

If ComboBox2.Text = "" Then
    MsgBox("Please Select a Item", MsgBoxStyle.Information, "Purchase
Details")
    Exit Sub
End If

dataBase()
' MsgBox("The Select Query" & "select Quantity from inventory where
itemname=" & ComboBox2.Text & """)
comm = New OleDbCommand("select Quantity from inventory where
ItemName=" & ComboBox2.Text & """)
read1 = comm.ExecuteReader
a = 0

If read1.Read Then
    a = read1(0)

End If

```

```

'comm.ExecuteNonQuery()
conn.Close()

dataBase()
If a.Equals(0) Then
    comm = New OleDbCommand("insert into inventory values(" &
ComboBox1.Text & "," & ComboBox2.Text & "," & TextBox6.Text & ".3)", conn)
    comm.ExecuteNonQuery()
    conn.Close()
Else
    MsgBox("The update Query" & "update inventory set Quantity=" & a +
Val(TextBox6.Text) & " where itemID=" & ComboBox1.Text & " and ItemName="
& ComboBox2.Text & """)
    a = a + Val(TextBox6.Text)
    ' MsgBox("Already Stock" & a)
    comm = New OleDbCommand("update inventory set Quantity=" & a & "
where ItemName=" & ComboBox2.Text & """, conn)
    comm.ExecuteNonQuery()
    conn.Close()
End If
MsgBox("Data Added Successfully....", MsgBoxStyle.Information, "Purchase
Details")
combofill()
clear()
End Sub
Function combofill()
    dataBase()
    comm = New OleDbCommand("select * from PurchaseDetails", conn)
    adp = New OleDbDataAdapter(comm)
    ds = New DataSet
    adp.Fill(ds, "PurchaseDetails")
    i = ds.Tables("PurchaseDetails").Rows.Count
    For a = 0 To i - 1
        ComboBox1.Items.Remove(ds.Tables("PurchaseDetails").Rows(a)(0))
        ComboBox1.Items.Add(ds.Tables("PurchaseDetails").Rows(a)(0))
    Next
    conn.Close()

    conn.Open()
    comm = New OleDbCommand("select * from SupplierDetails", conn)
    adp = New OleDbDataAdapter(comm)
    ds = New DataSet
    adp.Fill(ds, "SupplierDetails")
    i = ds.Tables("SupplierDetails").Rows.Count
    For a = 0 To i - 1
        ComboBox3.Items.Remove(ds.Tables("SupplierDetails").Rows(a)(0))
        ComboBox3.Items.Add(ds.Tables("SupplierDetails").Rows(a)(0))
    Next
    conn.Close()

```

```

conn.Open()
comm = New OleDbCommand("select * from ItemDetails", conn)
adp = New OleDbDataAdapter(comm)
ds = New DataSet
adp.Fill(ds, "ItemDetails")
i = ds.Tables("ItemDetails").Rows.Count
For a = 0 To i - 1
    ComboBox2.Items.Remove(ds.Tables("ItemDetails").Rows(a)(2))
    ComboBox2.Items.Add(ds.Tables("ItemDetails").Rows(a)(2))
Next
conn.Close()

```

```
Return (0)
```

```
End Function
```

```
Function clear()
```

```

ComboBox1.Text = ""
ComboBox2.Text = ""
ComboBox3.Text = ""
TextBox2.Text = ""
TextBox6.Text = ""
TextBox7.Text = ""
TextBox8.Text = ""
TextBox9.Text = ""
ComboBox4.Text = ""
TextBox1.Text = ""
Return 0

```

```
End Function
```

```
Function DataBase()
```

```

Dim appbase As String = System.AppDomain.CurrentDomain.BaseDirectory
Dim p As String
Dim conqry As String
p = appbase.Substring(0, appbase.LastIndexOf(New Char() {"\"c}))
For i = 0 To 1
    p = p.Substring(0, p.LastIndexOf(New Char() {"\"c}))
Next i
conqry = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & p &
"\database\StockManagement.mdb;Persist Security Info=False"
conn = New OleDbConnection(conqry)
conn.Open()

```

```
Return (0)
```

```
End Function
```

```

Private Sub PurchaseDetails_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
    combofill()
End Sub

```

```
Private Sub TextBox9_LostFocus(ByVal sender As Object, ByVal e As System.EventArgs) Handles TextBox9.LostFocus
    TextBox8.Text = Val(TextBox6.Text) * Val(TextBox9.Text)
End Sub
```

```
Private Sub ComboBox1_SelectedIndexChanged(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles ComboBox1.SelectedIndexChanged
    Dim dread As OleDbDataReader
    If ComboBox1.Text = "" Then
        MsgBox("Please Select a OrderNo", MsgBoxStyle.Information, "Purchase Details")
        Exit Sub
    End If
    dataBase()
    comm = New OleDbCommand("select * from PurchaseDetails where porderno=" & ComboBox1.Text & "", conn)
    comm.ExecuteNonQuery()
    dread = comm.ExecuteReader()
    If (dread.Read) Then
        ComboBox3.Text = dread(1)
        TextBox2.Text = dread(2)
        DateTimePicker1.Text = dread(3)
        ComboBox2.Text = dread(4)

        TextBox7.Text = dread(5)
        TextBox6.Text = dread(6)
        TextBox9.Text = dread(7)
        TextBox8.Text = dread(8)
        ComboBox4.Text = dread(9)
        TextBox1.Text = dread(10)
    End If
    conn.Close()
End Sub
```

```
Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click
    If ComboBox1.Text = "" Then
        MsgBox("Please Select a OrderNo", MsgBoxStyle.Information, "Purchase Details")
        Exit Sub
    End If
    dataBase()
    comm = New OleDbCommand("delete from PurchaseDetails where porderno=" & ComboBox1.Text & "", conn)
    comm.ExecuteNonQuery()
    MsgBox("Data Deleted Successfully....", MsgBoxStyle.Information, "Purchase Details")
    combofill()
    clear()
End Sub
```

```

Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button3.Click
    If ComboBox3.Text = "" Then
        MsgBox("Please Select a Supplier Id", MsgBoxStyle.Information, "Purchase
Details")
        Exit Sub
    End If
    dataBase()
    comm = New OleDbCommand("update PurchaseDetails set supplierid=" &
ComboBox3.Text & ",sname=" & TextBox2.Text & ",pdate=" &
DateTimePicker1.Text & ",mname=" & ComboBox2.Text & ",uom=" &
TextBox7.Text & ",quantity=" & TextBox6.Text & ",uprice=" & TextBox9.Text &
",price=" & TextBox8.Text & " where porderno=" & ComboBox1.Text & "", conn)
    ' insert into PurchaseDetails values(" & ComboBox1.Text & "," &
ComboBox3.Text & "," & TextBox2.Text & "," & DateTimePicker1.Text & "," &
ComboBox2.Text & "," & TextBox7.Text & "," & TextBox6.Text & "," &
TextBox9.Text & "," & TextBox8.Text & ")", conn)
    comm.ExecuteNonQuery()
    MsgBox("Data Updated Successfully....", MsgBoxStyle.Information, "Purchase
Details")
    combofill()
    clear()
End Sub

```

```

Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button4.Click
    Me.Close()
End Sub

```

```

Private Sub Button5_Click_1(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button5.Click
    clear()
End Sub

```

```

Private Sub ComboBox2_SelectedValueChanged(ByVal sender As Object, ByVal
e As System.EventArgs) Handles ComboBox2.SelectedValueChanged
    dataBase()
    comm = New OleDbCommand("select * from ItemDetails where itemname=" &
ComboBox2.Text & "", conn)
    read = comm.ExecuteReader
    If read.Read Then
        TextBox7.Text = read(3)
    End If

    conn.Close()
End Sub

```

```

Private Sub TextBox8_LostFocus(ByVal sender As Object, ByVal e As
System.EventArgs)
    If TextBox8.Text <> "" And ComboBox4.Text <> "" Then

```

```

    tprice = TextBox8.Text + ((TextBox8.Text * ComboBox4.Text) / 100)
    TextBox1.Text = tprice
End If
End Sub

```

```

Private Sub ComboBox3_SelectedValueChanged(ByVal sender As Object, ByVal e As System.EventArgs) Handles ComboBox3.SelectedValueChanged

```

```

    dataBase()
    comm = New OleDbCommand("select * from SupplierDetails where supplierid=" & ComboBox3.Text & "", conn)
    read = comm.ExecuteReader
    If read.Read Then
        TextBox2.Text = read(1)
    End If

    conn.Close()

```

```

End Sub

```

```

Private Sub ComboBox4_SelectedValueChanged(ByVal sender As Object, ByVal e As System.EventArgs) Handles ComboBox4.SelectedValueChanged

```

```

    If TextBox8.Text <> "" And ComboBox4.Text <> "" Then
        tprice = TextBox8.Text + ((TextBox8.Text * ComboBox4.Text) / 100)
        TextBox1.Text = tprice
    End If
End Sub

```

```

Private Sub MakeOrder_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MakeOrder.Click

```

```

    If ComboBox1.Text = "" Then
        MsgBox("Please Enter the PurchaseOrder No", MsgBoxStyle.Information, "PurchaseOrder")
        Exit Sub
    End If

```

```

    If ComboBox4.Text = "" Then
        MsgBox("Please Enter the SupplierId", MsgBoxStyle.Information, "PurchaseOrder")
        Exit Sub
    End If

```

```

    dataBase()

```

```

    comm = New OleDbCommand("delete * from PurchaseOrder", conn)
    comm.ExecuteNonQuery()

```

```

    comm = New OleDbCommand("select * from PurchaseDetails where norderno=" & TextBox1.Text & " ", conn)

```

```
read = comm.ExecuteReader()  
tax = TextBox8.Text * (ComboBox4.Text / 100)
```

```
While read.Read()
```

```
    comm = New OleDbCommand("select * from SupplierDetails where  
supplierid=" & ComboBox3.Text & "", conn)  
    read1 = comm.ExecuteReader()
```

```
    If read1.Read Then  
        saddress = read1(2)  
    End If
```

```
    comm = New OleDbCommand("insert into  
PurchaseOrder(PONo,SName,SAddress,ItemName,Quantity,UnitPrice,Price,TaxRate,  
Tax,PTax) values(" & ComboBox1.Text & "," & TextBox2.Text & "," & saddress &  
"," & ComboBox2.Text & "," & TextBox6.Text & "," & TextBox9.Text & "," &  
TextBox8.Text & "," & ComboBox4.Text & "," & tax & "." & TextBox1.Text & ")",  
conn)  
    comm.ExecuteNonQuery()
```

```
End While
```

```
conn.Close()  
Me.Close()  
PurchaseOrderReportViewer.Show()
```

```
End Sub  
End Class
```

## Purchase Return

```
Imports System.Data  
Imports System.Data.OleDb  
Public Class PurchaseReturns  
    Dim conn As OleDbConnection  
    Dim comm As OleDbCommand  
    Dim adp As OleDbDataAdapter  
    Dim ds As DataSet  
    Dim read, read1 As OleDbDataReader  
    Dim a, b, c, i As Integer  
    Private Sub PurchaseReturns_Load(ByVal sender As System.Object, ByVal e As  
System.EventArgs) Handles MyBase.Load  
        combofill()  
        dataBase()  
    End Sub
```

```
Function combofill()  
    dataBase()
```



```

comm = New OleDbCommand("select * from PurchaseDetails", conn)
adp = New OleDbDataAdapter(comm)
ds = New DataSet
adp.Fill(ds, "PurchaseDetails")
i = ds.Tables("PurchaseDetails").Rows.Count
For a = 0 To i - 1
    ComboBox1.Items.Remove(ds.Tables("PurchaseDetails").Rows(a)(0))
    ComboBox1.Items.Add(ds.Tables("PurchaseDetails").Rows(a)(0))
Next

```

```

dataBase()
comm = New OleDbCommand("select * from PurchaseReturns", conn)
adp = New OleDbDataAdapter(comm)
ds = New DataSet
adp.Fill(ds, "PurchaseReturns")
i = ds.Tables("PurchaseReturns").Rows.Count
TextBox4.Text = i + 1

```

Return 0

End Function

Function dataBase()

```

Dim appbase As String = System.AppDomain.CurrentDomain.BaseDirectory
Dim p As String
Dim conqry As String
p = appbase.Substring(0, appbase.LastIndexOf(New Char() {"\c"}))
For i = 0 To 1
    p = p.Substring(0, p.LastIndexOf(New Char() {"\c"}))
Next i
conqry = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & p &
"\database\StockManagement.mdb;Persist Security Info=False"
conn = New OleDbConnection(conqry)
conn.Open()
Return (0)
End Function

```

Private Sub ComboBox1\_SelectedIndexChanged(ByVal sender As System.Object,  
ByVal e As System.EventArgs) Handles ComboBox1.SelectedIndexChanged

```

If ComboBox1.Text = "" Then
    MsgBox("Please Select a OrderNo", MsgBoxStyle.Information,
"PurchaseReturns")

```

Exit Sub

End If

dataBase()

```

comm = New OleDbCommand("select * from PurchaseDetails where
porderno=" & ComboBox1.Text & "", conn)

```

comm.ExecuteNonQuery()

read = comm.ExecuteReader()

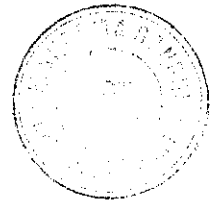
If (read.Read) Then

TextBox1.Text = read(1)

```

    TextBox2.Text = read(2)
    DateTimePicker1.Text = read(3)
    TextBox5.Text = read(4)
    TextBox6.Text = read(5)
    TextBox7.Text = read(6)
    c = Val(read(7))
    c = c + (c * read(9) / 100)
    TextBox8.Text = read(8)

```



```

End If
conn.Close()
End Sub

```

```

Private Sub TextBox8_LostFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles TextBox8.LostFocus
    If Val(TextBox8.Text) > Val(TextBox7.Text) Then
        MsgBox("Please enter the Valid Quantity", MsgBoxStyle.Information,
"PurchaseReturns")
    Else
        TextBox9.Text = Val(TextBox8.Text) * c
    End If

```

```

End Sub

```

```

Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
    If ComboBox1.Text = "" Then
        MsgBox("Please Select a OrderNo", MsgBoxStyle.Information,
"PurchaseReturns")
        Exit Sub
    End If
    dataBase()
    comm = New OleDbCommand("insert into PurchaseReturns values(" &
TextBox4.Text & "," & ComboBox1.Text & "," & DateTimePicker1.Text & "," &
TextBox8.Text & "," & TextBox9.Text & ")", conn)
    comm.ExecuteNonQuery()
    conn.Close()

```

```

    dataBase()
    'MsgBox("The Select Query" & "select Quantity from inventory where
itemname=" & ComboBox2.Text & """)
    comm = New OleDbCommand("select Quantity from inventory where
itemName=" & TextBox5.Text & """, conn)
    read1 = comm.ExecuteReader()
    a = 0
    b = 0
    If read1.Read Then
        a = read1(0)
        b = read1.FieldCount
    End If

```

```

'comm.ExecuteNonQuery()
conn.Close()

dataBase()
If a.Equals(0) And b.Equals(0) Then
    MsgBox("Invalid Goods Return", MsgBoxStyle.Critical, "PurchaseReturns")
    'comm = New OleDbCommand("insert into inventory values(" &
ComboBox1.Text & "," & ComboBox2.Text & "," & TextBox6.Text & ",2)", conn)
    'comm.ExecuteNonQuery()
    ' conn.Close()
Else
    'MsgBox("The update Query" & "update inventory set Quantity=" & a +
Val(TextBox6.Text) & " where itemID=" & ComboBox1.Text & " and ItemName="
& ComboBox2.Text & """)
    a = a - Val(TextBox8.Text)
    MsgBox("Now Available Stock is " & a)
    comm = New OleDbCommand("update inventory.set Quantity=" & a & "
where ItemName=" & TextBox5.Text & """, conn)
    comm.ExecuteNonQuery()
    conn.Close()
End If
MsgBox("Goods Returned Successfully", MsgBoxStyle.Information,
"PurchaseReturns")
clear()
End Sub
Function clear()
    TextBox1.Text = ""
    TextBox2.Text = ""
    TextBox4.Text = ""
    TextBox5.Text = ""
    TextBox6.Text = ""
    TextBox8.Text = ""
    TextBox9.Text = ""
    TextBox7.Text = ""
    ComboBox1.Text = ""

Return 0

End Function

Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button4.Click
    Me.Close()
End Sub
End Class

```

## Sales Details

Imports System.Data

Public Class SalesDetails

Dim conn As OleDbConnection

Dim comm As OleDbCommand

Dim adp As OleDbDataAdapter

Dim ds As DataSet

Dim read, read1 As OleDbDataReader

Dim a, b, c, i As Integer

Dim tax, tprice As Double

Private Sub Button1\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

If ComboBox1.Text = "" Then

MsgBox("Please Select a InvoiceNo", MsgBoxStyle.Information, "SalesDetails")

Exit Sub

End If

If ComboBox3.Text = "" Then

MsgBox("Please Select the CustomerId", MsgBoxStyle.Information, "SalesDetails")

Exit Sub

End If

If ComboBox2.Text = "" Then

MsgBox("Please Select the ProductName", MsgBoxStyle.Information, "SalesDetails")

Exit Sub

End If

If TextBox6.Text = "" Then

MsgBox("Please Enter the Quantity", MsgBoxStyle.Information, "SalesDetails")

Exit Sub

End If

If ComboBox4.Text = "" Then

MsgBox("Please Select a Tax", MsgBoxStyle.Information, "SalesDetails")

Exit Sub

End If

If TextBox9.Text = "" Then

MsgBox("Please Enter the UnitPrice", MsgBoxStyle.Information, "SalesDetails")

Exit Sub

End If

tprice = TextBox8.Text + ((TextBox8.Text \* ComboBox4.Text) / 100)

TextBox3.Text = tprice

dataBase()

```

comm = New OleDbCommand("insert into SalesDetails values(" &
ComboBox1.Text & "," & ComboBox3.Text & "," & TextBox2.Text & "," &
DateTimePicker1.Text & "," & ComboBox2.Text & "," & TextBox7.Text & "," &
TextBox6.Text & "," & TextBox9.Text & "," & TextBox8.Text & "," &
ComboBox4.Text & "," & TextBox3.Text & ") ", conn)
comm.ExecuteNonQuery()
conn.Close()

```

```

If ComboBox2.Text = "" Then
    MsgBox("Please Select a Item", MsgBoxStyle.Information, "SalesDetails")
Exit Sub
End If

```

```

dataBase()
'MsgBox("The Select Query" & "select Quantity from inventory where
itemname=" & ComboBox2.Text & """)
comm = New OleDbCommand("select Quantity from pinventory where
itemName=" & ComboBox2.Text & """, conn)
read1 = comm.ExecuteReader
a = 0

```

```

If read1.Read Then
    a = read1(0)

```

```

End If
'comm.ExecuteNonQuery()
conn.Close()

```

```

dataBase()
If a.Equals(0) Then
    comm = New OleDbCommand("insert into pinventory values(" &
ComboBox1.Text & "," & ComboBox2.Text & "," & TextBox6.Text & "," &
DateTimePicker1.Text & ") ", conn)
    comm.ExecuteNonQuery()
    conn.Close()
Else
    'MsgBox("The update Query" & "update inventory set Quantity=" & a +
Val(TextBox6.Text) & " where itemID=" & ComboBox1.Text & " and ItemName="
& ComboBox2.Text & """)
    a = a - Val(TextBox6.Text)
    'MsgBox("Already Stock" & a)
    comm = New OleDbCommand("update pinventory set Quantity=" & a & "
where ItemName=" & ComboBox2.Text & """, conn)
    comm.ExecuteNonQuery()
    conn.Close()

```

```

End If

```

```

MsgBox("Data Added Successfully....", MsgBoxStyle.Information,
"SalesDetails")
    combofill()
    clear()
End Sub
Function combofill()
    dataBase()
    comm = New OleDbCommand("select * from SalesDetails", conn)
    adp = New OleDbDataAdapter(comm)
    ds = New DataSet
    adp.Fill(ds, "SalesDetails")
    i = ds.Tables("SalesDetails").Rows.Count
    For a = 0 To i - 1
        ComboBox1.Items.Remove(ds.Tables("SalesDetails").Rows(a)(0))
        ComboBox1.Items.Add(ds.Tables("SalesDetails").Rows(a)(0))
    Next
    conn.Close()

    conn.Open()
    comm = New OleDbCommand("select * from CustomerDetails", conn)
    adp = New OleDbDataAdapter(comm)
    ds = New DataSet
    adp.Fill(ds, "CustomerDetails")
    i = ds.Tables("CustomerDetails").Rows.Count
    For a = 0 To i - 1
        ComboBox3.Items.Remove(ds.Tables("CustomerDetails").Rows(a)(0))
        ComboBox3.Items.Add(ds.Tables("CustomerDetails").Rows(a)(0))
    Next
    conn.Close()

    conn.Open()
    comm = New OleDbCommand("select * from ProductDetails", conn)
    adp = New OleDbDataAdapter(comm)
    ds = New DataSet
    adp.Fill(ds, "ProductDetails")
    i = ds.Tables("ProductDetails").Rows.Count
    For a = 0 To i - 1
        ComboBox2.Items.Remove(ds.Tables("ProductDetails").Rows(a)(2))
        ComboBox2.Items.Add(ds.Tables("ProductDetails").Rows(a)(2))
    Next
    conn.Close()

    Return (0)
End Function
Function clear()
    ComboBox1.Text = ""
    ComboBox2.Text = ""
    ComboBox3.Text = ""
    TextBox2.Text = ""

```

```
TextBox6.Text = ""
TextBox7.Text = ""
TextBox8.Text = ""
TextBox9.Text = ""
ComboBox4.Text = ""
TextBox3.Text = ""
Return 0
```

End Function

Function dataBase()

```
Dim appbase As String = System.AppDomain.CurrentDomain.BaseDirectory
Dim p As String
Dim conqry As String
p = appbase.Substring(0, appbase.LastIndexOf(New Char() {"\c"}))
For i = 0 To 1
    p = p.Substring(0, p.LastIndexOf(New Char() {"\c"}))
Next i
conqry = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & p &
"\database\StockManagement.mdb;Persist Security Info=False"
conn = New OleDbConnection(conqry)
conn.Open()

Return (0)
End Function
```

```
Private Sub PurchaseDetails_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load
    combofill()
End Sub
```

End Sub

```
Private Sub ComboBox3_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles ComboBox3.SelectedIndexChanged
    dataBase()
    comm = New OleDbCommand("select * from CustomerDetails", conn)
    read = comm.ExecuteReader
    If read.Read Then
        TextBox2.Text = read(1)
    End If

    conn.Close()
End Sub
```

End Sub

```

Private Sub ComboBox2_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles ComboBox2.SelectedIndexChanged
    dataBase()
    comm = New OleDbCommand("select * from ProductDetails", conn)
    read = comm.ExecuteReader
    If read.Read Then
        TextBox7.Text = read(3)
    Else

    End If
    conn.Close()

    qfill()

End Sub

```

```

Private Sub TextBox6_LostFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles TextBox6.LostFocus
    If Val(TextBox6.Text) > Val(TextBox1.Text) Then
        MsgBox("Invalid Sales Quantity", MsgBoxStyle.Information, "SalesDetails")
        TextBox6.Text = ""
        TextBox6.Focus()
    End If
End Sub

```

```

Private Sub TextBox9_LostFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles TextBox9.LostFocus
    TextBox8.Text = Val(TextBox6.Text) * Val(TextBox9.Text)
End Sub

```

```

Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button2.Click
    If ComboBox1.Text = "" Then
        MsgBox("Please Select a InvoiceNo", MsgBoxStyle.Information,
"SalesDetails")
        Exit Sub
    End If
    dataBase()
    comm = New OleDbCommand("delete from SalesDetails where invoiceno=" &
ComboBox1.Text & "", conn)
    comm.ExecuteNonQuery()
    MsgBox("Data Deleted Successfully....", MsgBoxStyle.Information,
"SalesDetails")
    combofill()
End Sub

```

```

Private Sub Button3_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button3.Click
    If ComboBox3.Text = "" Then

```



```

        MsgBox("Please Select a CustomerId", MsgBoxStyle.Information,
"SalesDetails")
        Exit Sub
    End If
    dataBase()
    comm = New OleDbCommand("update SalesDetails set custid=" &
ComboBox3.Text & ",cname=" & TextBox2.Text & ",sdate=" &
DateTimePicker1.Text & ",sname=" & ComboBox2.Text & ",uom=" &
TextBox7.Text & ",quantity=" & TextBox6.Text & ",uprice=" & TextBox9.Text &
",price=" & TextBox8.Text & " where porderno=" & ComboBox1.Text & "", conn)
    ' insert into PurchaseDetails values(" & ComboBox1.Text & "," &
ComboBox3.Text & "," & TextBox2.Text & "," & DateTimePicker1.Text & "," &
ComboBox2.Text & "," & TextBox7.Text & "," & TextBox6.Text & "," &
TextBox9.Text & "," & TextBox8.Text & ")", conn)
    comm.ExecuteNonQuery()
    MsgBox("Data Updated Successfully....", MsgBoxStyle.Information,
"SalesDetails")
    combofill()

End Sub

```

```

Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button4.Click
    Me.Close()
End Sub
Function qfill()
    If ComboBox2.Text = "" Then
        MsgBox("Please Select a Item", MsgBoxStyle.Information, "SalesDetails")
        Return 0
    Exit Function
    End If
    dataBase()
    comm = New OleDbCommand("select * from plnventory where itemName=" &
ComboBox2.Text & "", conn)
    read = comm.ExecuteReader
    If read.Read Then
        TextBox1.Text = read(2)
    Else
        TextBox1.Text = "0"
    End If

    conn.Close()
    Return (0)

End Function

```

```

Private Sub ComboBox4_SelectedValueChanged(ByVal sender As Object, ByVal
e As System.EventArgs) Handles ComboBox4.SelectedValueChanged
    If TextBox8.Text <> "" And ComboBox4.Text <> "" Then

```

```

        tprice = TextBox8.Text + ((TextBox8.Text * ComboBox4.Text) / 100)
        TextBox3.Text = tprice
    End If
End Sub

Private Sub TextBox8_LostFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles TextBox8.LostFocus
    If TextBox8.Text <> "" And ComboBox4.Text <> "" Then
        tprice = TextBox8.Text + ((TextBox8.Text * ComboBox4.Text) / 100)
        TextBox3.Text = tprice
    End If
End Sub

Private Sub ComboBox1_SelectedValueChanged(ByVal sender As Object, ByVal
e As System.EventArgs) Handles ComboBox1.SelectedValueChanged
    Dim dread As OleDbDataReader
    If ComboBox1.Text = "" Then
        MsgBox("Please Select a InvoiceNo", MsgBoxStyle.Information,
"SalesDetails")
        Exit Sub
    End If
    dataBase()
    comm = New OleDbCommand("select * from salesdetails where invoiceno=" &
ComboBox1.Text & "", conn)
    comm.ExecuteNonQuery()
    dread = comm.ExecuteReader()
    If (dread.Read) Then
        ComboBox3.Text = dread(1)
        TextBox2.Text = dread(2)
        DateTimePicker1.Text = dread(3)
        ComboBox2.Text = dread(4)
        TextBox7.Text = dread(5)
        TextBox6.Text = dread(6)
        TextBox9.Text = dread(7)
        TextBox8.Text = dread(8)
        ComboBox4.Text = dread(9)
        TextBox3.Text = dread(10)
    End If
    conn.Close()
End Sub
End Class

```

## Sales Return

```

Imports System.Data
Imports System.Data.OleDb
Public Class SalesReturns
    Dim conn As OleDbConnection
    Dim comm As OleDbCommand

```

```
Dim adp As OleDbDataAdapter
Dim ds As DataSet
Dim read, read1 As OleDbDataReader
Dim a, b, c, i As Integer
Dim dat As DateTime
```

```
Private Sub PurchaseReturns_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles MyBase.Load
```

```
    DateTimePicker1.CustomFormat = "dd/mm/yyyy"
    combofill()
    dataBase()
```

```
End Sub
```

```
Function combofill()
```

```
    dataBase()
    comm = New OleDbCommand("select * from SalesDetails", conn)
    adp = New OleDbDataAdapter(comm)
    ds = New DataSet
    adp.Fill(ds, "SalesDetails")
    i = ds.Tables("SalesDetails").Rows.Count
    For a = 0 To i - 1
        ComboBox1.Items.Remove(ds.Tables("SalesDetails").Rows(a)(0))
        ComboBox1.Items.Add(ds.Tables("SalesDetails").Rows(a)(0))
    Next
```

```
    dataBase()
    comm = New OleDbCommand("select * from SalesReturns", conn)
    adp = New OleDbDataAdapter(comm)
    ds = New DataSet
    adp.Fill(ds, "SalesReturns")
    i = ds.Tables("SalesReturns").Rows.Count
    TextBox4.Text = i + 1
```

```
Return 0
```

```
End Function
```

```
Function dataBase()
```

```
    Dim appbase As String = System.AppDomain.CurrentDomain.BaseDirectory
    Dim p As String
    Dim conqry As String
    p = appbase.Substring(0, appbase.LastIndexOf(New Char() {"\c"}))
    For i = 0 To 1
        p = p.Substring(0, p.LastIndexOf(New Char() {"\c"}))
    Next i
    conqry = "Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" & p &
"\database\StockManagement.mdb;Persist Security Info=False"
    conn = New OleDbConnection(conqry)
    conn.Open()
```

```
Return (0)
End Function
```

```
Private Sub ComboBox1_SelectedIndexChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles ComboBox1.SelectedIndexChanged
    If ComboBox1.Text = "" Then
        MsgBox("Please Select a InvoiceNo", MsgBoxStyle.Information,
"SalesReturn")
        Exit Sub
    End If
    dataBase()
    comm = New OleDbCommand("select * from SalesDetails where invoiceno=" &
ComboBox1.Text & "", conn)
    comm.ExecuteNonQuery()
    read = comm.ExecuteReader()
    If (read.Read) Then
        TextBox1.Text = read(1)
        TextBox2.Text = read(2)
        dat = read(3)
        DateTimePicker1.CustomFormat = "dd/MM/yyyy"
        DateTimePicker1.Value = dat
        TextBox5.Text = read(4)
        TextBox6.Text = read(6)
        TextBox7.Text = read(5)
        c = read(10)
    End If

End Sub
```

```
Private Sub TextBox8_LostFocus(ByVal sender As Object, ByVal e As
System.EventArgs) Handles TextBox8.LostFocus
    If Val(TextBox8.Text) > Val(TextBox6.Text) Then
        MsgBox("Invalid Quantity", MsgBoxStyle.Information, "SalesReturn")
        TextBox8.Focus()
    Else
        TextBox9.Text = Val(TextBox8.Text) * c
    End If

End Sub
```

```
Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles Button1.Click
    If ComboBox1.Text = "" Then
        MsgBox("Please Select a InvoiceNo", MsgBoxStyle.Information,
"SalesReturn")
        Exit Sub
    End If
```

```
dataBase()
```

```

comm = New OleDbCommand("select Quantity from Pinventory where
itemname='" & TextBox5.Text & "'", conn)
comm.ExecuteNonQuery()
read = comm.ExecuteReader()

a = 0
b = 0

If read.Read() Then
    a = read(0)
    b = read.FieldCount
End If

conn.Close()

dataBase()
MsgBox("The Details " & "insert into SalesReturns values(" & TextBox4.Text &
"," & ComboBox1.Text & "," & DateTimePicker1.Text & "," & TextBox8.Text & ","
& TextBox9.Text & ")")
comm = New OleDbCommand("insert into SalesReturns values(" &
TextBox4.Text & "," & ComboBox1.Text & "," & DateTimePicker1.Text & "," &
TextBox8.Text & "," & TextBox9.Text & ")". conn)
comm.ExecuteNonQuery()
conn.Close()

dataBase()
If a.Equals(0) And b.Equals(0) Then
    MsgBox("This Item Cannot be Sale", MsgBoxStyle.Information,
"SalesReturn Details")
    'comm = New OleDbCommand("insert into inventory values(" &
ComboBox1.Text & "," & ComboBox2.Text & "," & TextBox6.Text & ",2)", conn)
    'comm.ExecuteNonQuery()
    ' conn.Close()
Else
    'MsgBox("The update Query" & "update inventory set Quantity=" & a +
Val(TextBox6.Text) & " where itemID=" & ComboBox1.Text & " and ItemName="
& ComboBox2.Text & "'")
    a = a + Val(TextBox8.Text)
    MsgBox("Now Total Stock is " & a)
    comm = New OleDbCommand("update pinventory set Quantity=" & a & "
where ItemName=" & TextBox5.Text & "'", conn)
    comm.ExecuteNonQuery()
    conn.Close()

End If

```

```
    MsgBox("Sales Returned Successfully", MsgBoxStyle.Information,  
"SalesReturn Details")  
    clear()  
End Sub  
Function clear()  
    TextBox1.Text = ""  
    TextBox2.Text = ""  
    TextBox4.Text = ""  
    TextBox5.Text = ""  
    TextBox6.Text = ""  
    TextBox8.Text = ""  
    TextBox7.Text = ""  
    ComboBox1.Text = ""  
  
    Return 0  
  
End Function  
  
Private Sub Button4_Click(ByVal sender As System.Object, ByVal e As  
System.EventArgs) Handles Button4.Click  
    Me.Close()  
End Sub  
End Class
```