A

Project Report On

“ Bakery Management System “

Submitted To,

SHIVAJI UNIVERSITY, KOLHAPUR

For the partial fulfillment of the requirement

For the award Degree of

BACHELOR OF COMPUTER APPLICATIONS (SEM-V)

By-

1. Amit Pravin Chavan
2. Varad Vishweshwar Chitale.
3. Jayshree Ramdas Bhise
4. Reshma Nerlekar

Under The Guidance Of

Mr. Rajane S. B.

(M.C.A)

Through

The Director

Rayat Shikshan Sanstha’s

DHANANJAYRAO GADGIL COLLEGE OF COMMERCE SATARA.

YEAR (2012-2013)

Rayat Shikshan Sanstha's

DHANANJAYRAO GADGIL COLLEGE OF COMMERCE SATARA.

CERTIFICATE

This is to certify that Amit Pravin Chavan, Varad Vishweshwar Chitale, Jayshree Ramdas Bhise,Reshma Nerlekar
are bonafide in BCA III Semester V. They have completed Project Report under entitled " Bakery Management System "__under the guidance of Mr. Rajane S.B. satisfactory & submitted to Shivaji University, Kolhapur For the partial fulfillment of the requirement For the award Degree Of Bachelor of Computer Applications.

The matter presented in project report has not been submitted earlier.

Place: Satara

Dr.N.S,Gaikwad

Principal

(D.G.college of commerce satara.)

"Education through Self-help is our Motto"-Karmveer

Rayat Shikshan Sanstha's

Dhananjayrao Gadgil College of Commerce,Satara.

CERTIFICATE

Date:- **Exam Seat No:-**

This is to certify that Miss. /Mr. ---------- is a bonafied student of our college studying in B.C.A-III. He has successfully completed his practical work in the subject: practical Lab Course-IV and prepared journal satisfactorily in the Academic year 2011-2012.

Date- / /2012

Place:- Satara

Class Teacher **External Examiner** **Head of Department**

DECLARATION

I hereby declare that the project report entitled “Bakery Management System” which is being submitted in partial fulfillment of the requirement of the course leading to the completion of the semester project is the result of the project carried out by me under the guidance and supervision of Mr.Ranjane S.B.

I further declared that I or any other person has not previously submitted this project report to any other institution/university for any other degree/ diploma or any other person.

Date-
Place- Satara

Yours Sincerely,

Amit Pravin Chavan,
VaradVishwesh warChitale,
Jayshree Ramdas Bhise,
Reshma Nerlekar

It is certified that this project has been prepared and submitted under my guidance.

Date: (Mr.Ranjane S.B.)
Place:

ACKNOWLEDGEMENT

This project is done as a semester project . We are really thankful to our course instructor, Mr.Ranjane S.B Assistant Professor, Department of BCA, for his invaluable guidance and assistance, without which the accomplishment of the task would have never been possible.

In our present project we have chosen the topic- “Bakery Management System”. This report contains the complete computerized management of Bakery. This system reflects standard structure so that any inventory management system can implements this system easily in the existing system. This system works to reduce the human efforts. Due to totally computerized occurrence of error is less. This system works smoothly when used.

We are also thankful to the persons who are indirectly involved in the project completion.
We are also thankful to friends who have supported us in the successful completion of this project.

Yours Sincerely,

Amit Pravin Chavan,
Varad Vishweshwar Chitale,
Jayshree Ramdas Bhise,
Reshma Nerlekar

INDEX

No.	Title Name	Page No.
1.	Introduction a)Information of System Introduction b) Existing System. c) Objective of proposed Information System	
2.	System requirement Specifications a. Requirement Gathering Tools i: Fact Finding Tools b. Feasibility Study i: Technical ii. Social iii. Economical c. System Analysis i:Diagrams -Site Map d. System Design i:User Interface Design	
3.	System Diagrams a. DFD	

	b. ERD	
4.	Screen Shots	
5.	Source Code	
6.	Advantages	
7.	Limitation	
8.	Future Enhancement	
9.	Bibliography	

BAKERY

MANAGEM ENT SYSTEM

1) INTRODUCTION

a. Organization

b. Existing System

c. Objectives of Proposed System

a. INTRODUCTION TO ORGANIZATION -

Bakery Management System is totally computer based software application to maintain day to day transactions in a bakery. This software helps to store all bakery items with category and sub-category. It also maintains record of purchase and sales. It maintains details of Supplier. This application generates reports of purchase, sales and stock.

The system reflects standard structure so that any inventory management system can implements this system easily in their existing system. The system works to reduce the human efforts. Due to totally computerized occurrence of error is less & works smoothly. It is user friendly system.

b. INTRODUCTION TO EXISTING SYSTEM -

I) **Manual System:**

Manual System is tedious and has lot of paperwork. It is not much accurate and ambiguity exists in the manual system. No. of registers have to be maintained. Calculations should be done manually. Stock has to be checked often.

c. OBJECTIVE OF PROPOSED SYSTEM -

- The main objective of the project is to design and develop a user friendly system.

- Easy to use and an efficient computerized system.
- To develop an accurate and flexible system, it will eliminate data redundancy.
- Computerization can be helpful as a means of saving time and money.
- To provide better Graphical User Interface (GUI).
- Less chances of information leakage.
- Provides Security to the data by using login and password method.

2 .SYSTEM REQUIREMENT SPECIFICATIONS

a. Requirement Gathering

i. Fact Finding Tools

b. Feasibility Study

i. Technical

ii. Behavioral / Operational

iii. Economical

c. System Analysis

i. Diagrams

- Data Flow Diagram

d. System Design

i. Database Design

- Data Dictionary

ii. User Interface Design

- Forms (Input Screens)

- Reports (Output Screens)

SYSTEM STUDY:

The first four phases of System Development Life Cycle:

1) PLANNING PHASE:

The primary objectives of these phases are to identify the scope of the new system, ensures that the project is feasible, develop a schedule, allocate resources, and budget for the remainder of the project. The following are the five activities in the project-planning phase:

- Define the problem
- Confirm project feasibility
- Produce the project schedule
- Staff the project
- Launch the project

2) ANALYSIS PHASE:

The primary objectives of these phases are to understand the business needs and the processing requirements of the new system. Analysis is essentially a discovery process. The key words to drive the activities during analysis are discovery and understanding. There are six primary activities considered part of this phase:

- Gather information
- Define system requirements
- Build prototypes for discovery of requirements
- Prioritize requirements
- Generate and evaluate alternatives
- Review recommendations with management

3) DESIGN PHASE:

The objective of design phase is to design solution system. High-level design consists of developing an architectural structure for software programs, database, the user interface, and the operating environment. Low-level design emails developing the detailed algorithms and data structure that are required for program development. The design phase utilizes is its input the information obtained during the analysis phase. Seven major activities must be done during design. Design activities are closely interrelated and generally have substantial overlap .

Design and integrate network

- Design the application architecture
- Design user interfaces
- Design system interfaces
- Design the integrated database
- Prototype for design details
- Design and integrate the system controls

4) IMPLEMENTATION PHASE:

During the implementation phase, the final system is built, leads, and installed. The objectives of these phases are not only to have reliable, working information system but also to ensure that the users are all trained and that the business is benefiting. All the prior activities come together during this phase to culminate in an operational system. Six major activities make up the implementation phase:

- Construct software components
- Verify the test
- Develop prototypes for tuning
- Convert data
- Train and document

EXISTING SYSTEM:

The Bakery Management System is working manually. The current system is very time consuming and costly, because it involve lot of paper work. To manually handle such a system is very difficult task. But now-a-days because of computerization this job is becoming easier. Following are the reason why the current system should be computerized.

- To increase efficiency with reduced cost.
- To reduce the burden of paper work.
- To save the time of recording details of every work undertaken by Bakery.
- To check that the request for particular product is available.
- To generate reports easily.

LIMITATION OF CURRENT SYSTEM:

- **Time consumption:** As the records are to be manually maintained it consumes a lot of time.
- **Paper Work:** Lot of paper work is involved as the records are maintained in the files and registers.
- **Storage Requirements:** As files and registers are used the storage space requirement is increased.
- **Less Reliable:** Use of papers for storing valuable data information is not at all reliable.
- **Accuracy:** As the system is in manual there are lot many chances of human errors. These cause errors in calculating mechanism or maintaining product and supplier data in registers.

PROPOSED SYSTEM

PROPOSED SYSTEM:

To reduce the inconvenience that found in the current system, it has been automated so as to provide user friendly GUI that will help data entry. This also includes report generation.

DESCRIPTION OF THE SYSTEM:

The proposed system is created in Visual studio 2010 tool as a Front end and Back end SQL Server database. The proposed system deals with very popular interface tool i.e. Visual studio 2010 as Front end; hence it provides well-placed controlled information to the end user.

The database is used i.e. SQL Server is faster and well-designed tool to the user for quick manipulation with it.

The proposed system has better I/O capabilities of each the user skills while interacting with the system.

The retrieval of the records is much faster than the present system. Hence it causes in saving users time for the further work.

The user can have fast interaction with system by inserting, deleting, updating records etc. Because not only the front end provides the faster interaction with the records but the back end database also provides the proper interaction with the records and gives or prompts the information/messages to the end user if he/she making error during work.

The proper labeling on the proposed system acts as the manual for the new user.

Searching feature is quite faster than the present system since it searches directly from the system that is from the front end. The screening of the records is very clear and précised because of the grid control used which acts like table in the database. To fire a search query to the database, user need not have to go to the database for searching the records. Hence he/she can search from the front end itself.

PROPOSED SYSTEM:

The first advantage of the proposed system is that its front end provides easy and précised information to the user to interact with the system and hence it is faster to complete the work.

The next advantage of the present system is that its faster capability to interact with the database. Even the database also provides its inbuilt features to maintain the records. It also provides the security to the records from the system itself and also from the database itself. Hence it prevents the database conjunction.

Another advantage of the proposed system is its faster capability to search the records from the database.

The proposed system provides both with the mouse or keyboard handling features to interact with the system.

SYSTEM REQUIREMENT

Hardware & Software to be used:

- Desktop Standard Requirements:
  - Pentium4 (P4) or higher version.
  - 512 MB RAM or more.
  - 100 MB free space in Hard Disk.
  - Windows XP or Windows 7 version.

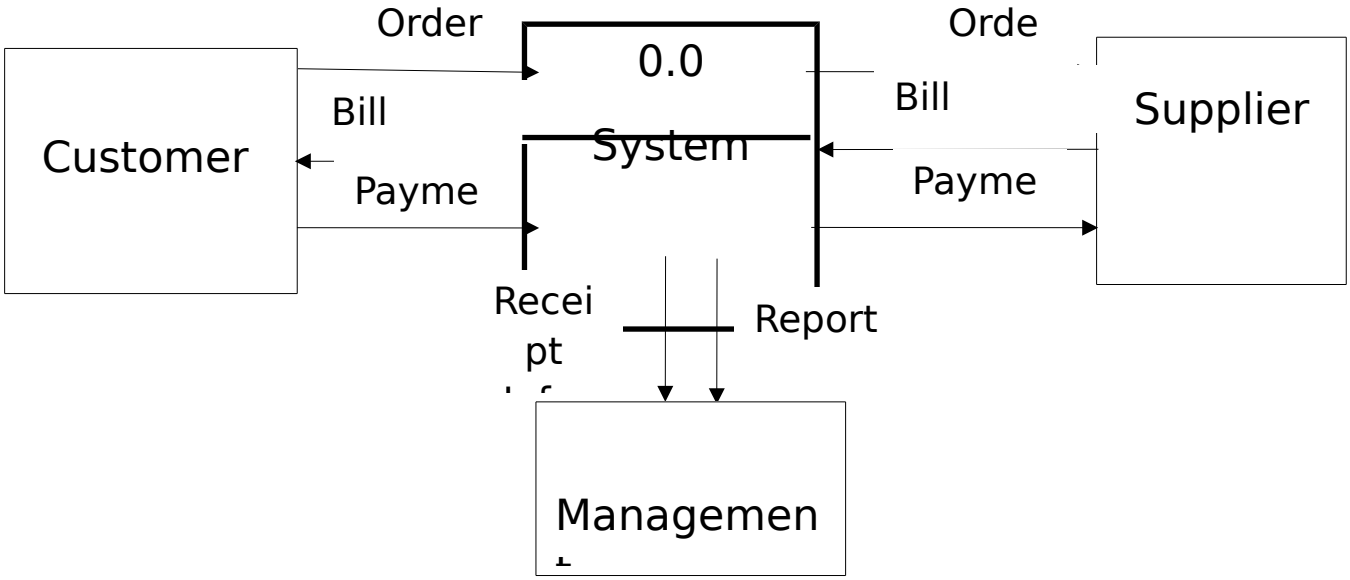
- Software:
  - SQL Server 2008

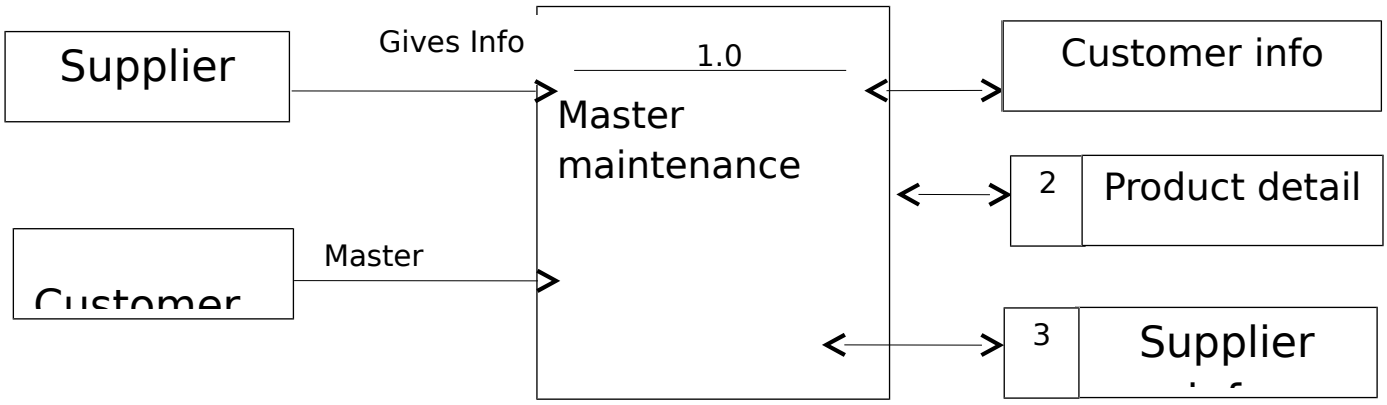
- Development Tools:
  - Microsoft Visual Studio 2010
  - SQL Server 2008

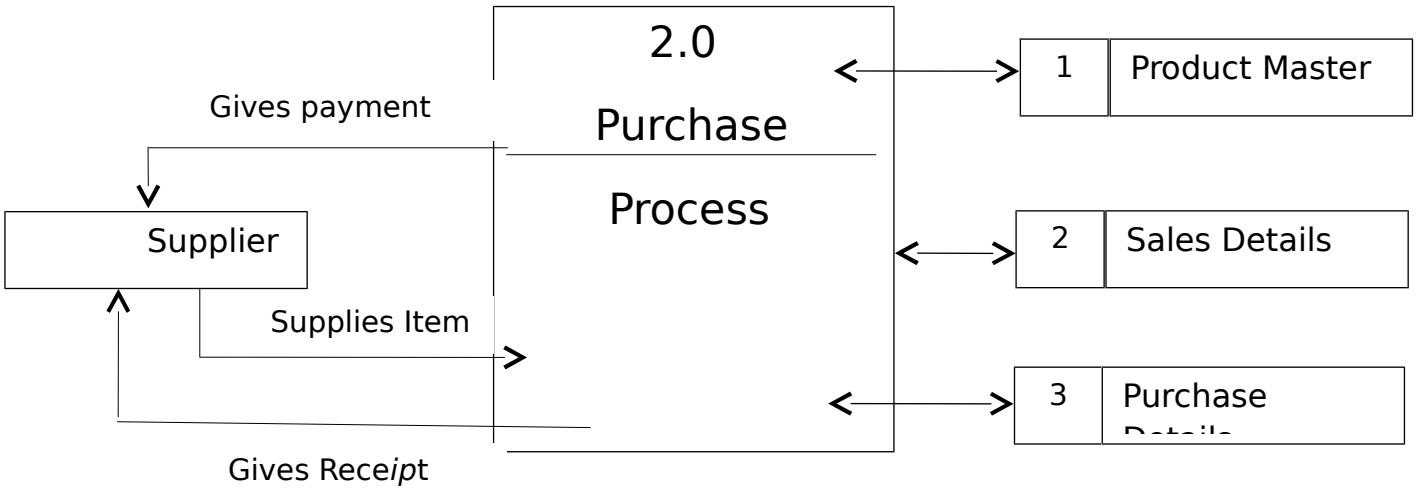
SYSTEM DIAGRAM

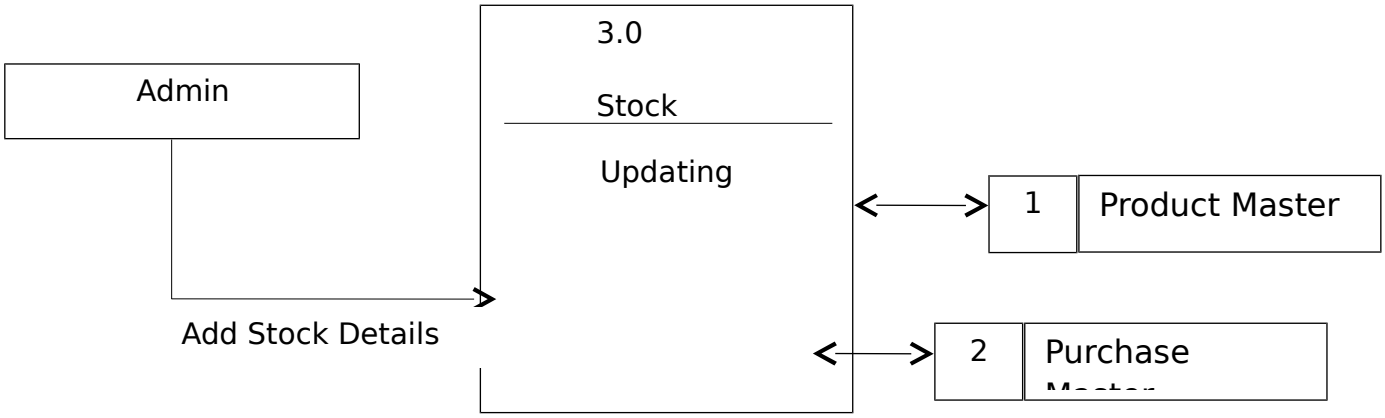
DIAGRAM

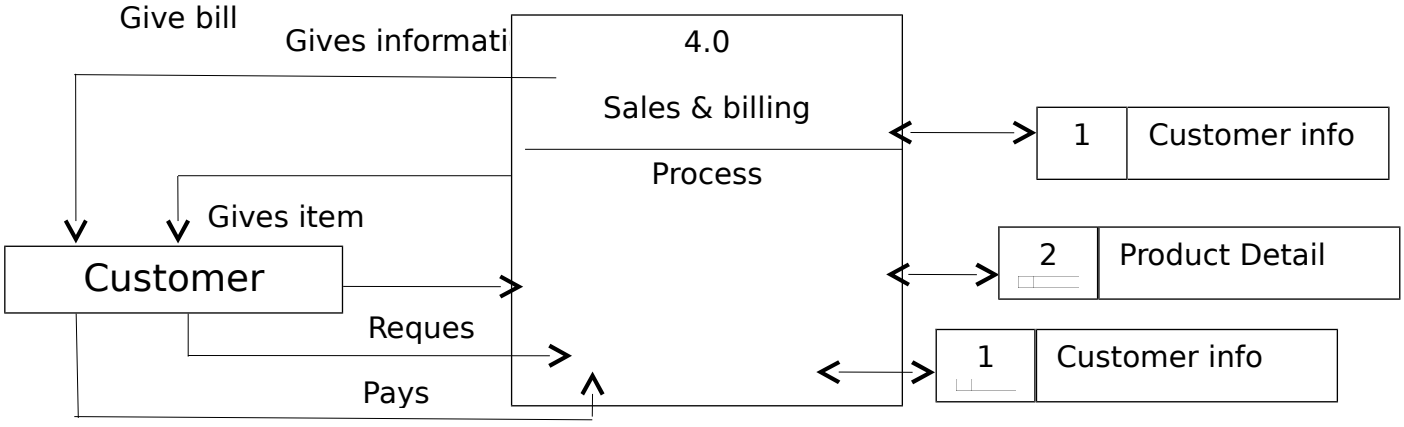
DFD


Order
0.0
Orde
Bill
Bill
Customer
System
Supplier
Payme
Payme
Recei
pt
Report
Managemen

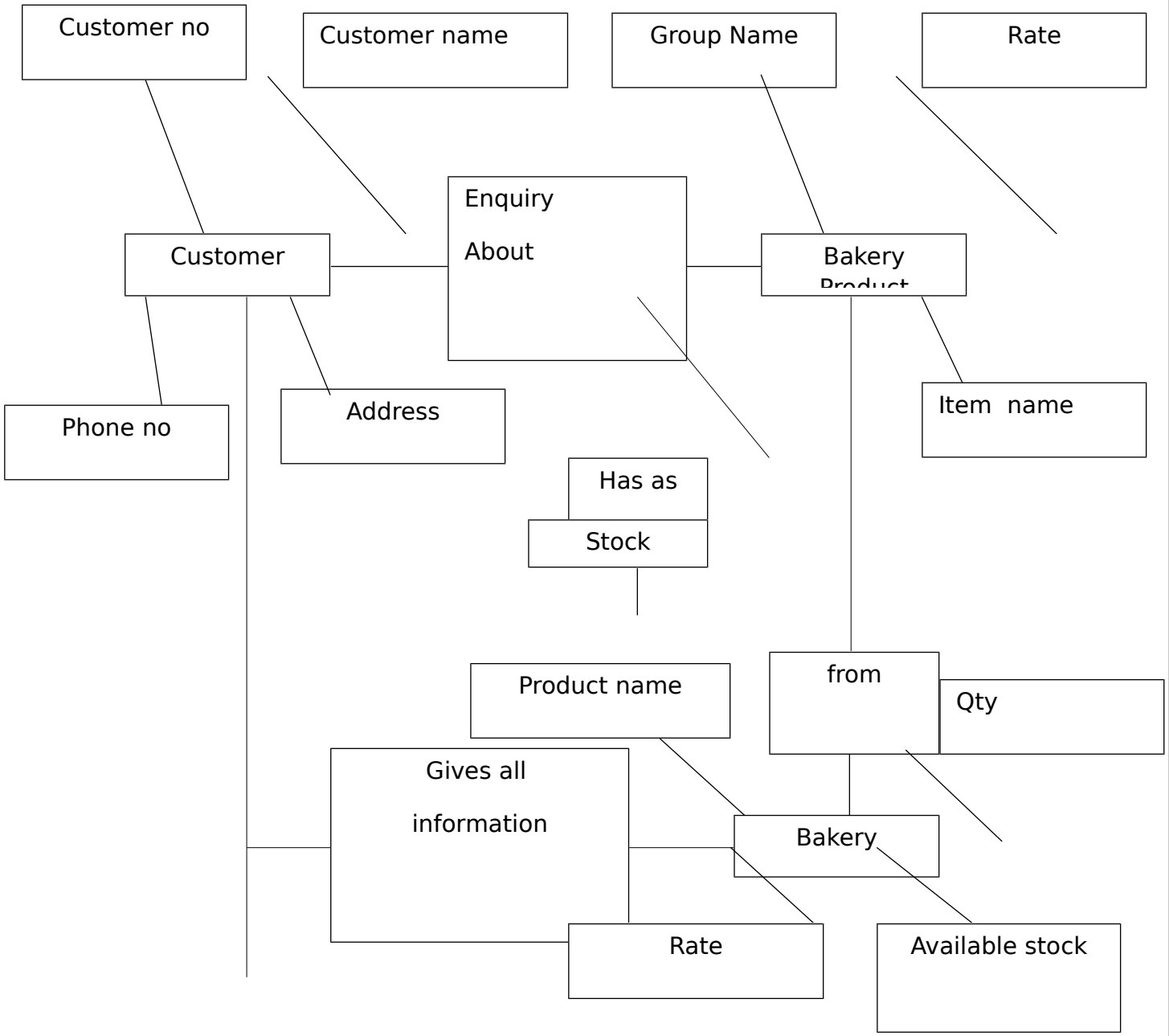

Supplier
Gives Info
1.0
Master
maintenance
Customer info
2
Product detail
Master
3
Supplier


2.0
Purchase
Process
1
Product Master
Gives payment
Supplier
2
Sales Details
Supplies Item
3
Purchase
Gives Receipt


3.0
Stock
Updating
Admin
Add Stock Details
1
Product Master
2
Purchase


Give bill
Gives informati
4.0
Sales & billing
Process
Gives item
Customer
Reques
Pays
1
Customer info
2
Product Detail
1
Customer info

ERD



SYSTEM DATABASE

Category Master

Column Name	Data Type	Allow Nulls
C_ID	int	☐
C_CSub	int	☐
C_Name	varchar(50)	☐
C_Disc	varchar(100)	☑
		☐

ITEM MASTER

Column Name	Data Type	Allow Nulls
I_ID	int	☐
I_Cat	int	☐
I_Name	varchar(50)	☐
I_Dis	varchar(100)	☐
I_Unit	varchar(50)	☐
I_SPrice	float	☐
I_MRp	float	☐
I_Disc	float	☑
I_Quantity	float	☐
		☐

PURCHASE MASTER

Column Name	Data Type	Allow Nulls
RegNo	int	☐
RefNo	int	☐
RefName	varchar(50)	☐
Date	date	☐
TItem	int	☐
TAMT	float	☐
		☐

PURCHASE DETAILS

Column Name	Data Type	Allow Nulls
RegNo	int	☐
IName	varchar(50)	☐
Quantity	float	☐
UPrice	float	☐
Unit	varchar(50)	☐
TAMT	float	☐
		☐

RECEIVED DETAILS

Column Name	Data Type	Allow Nulls
R_No	int	
IName	varchar(50)	
Quantity	float	
UPrice	float	
Unit	varchar(50)	
disc	float	
amt	float	

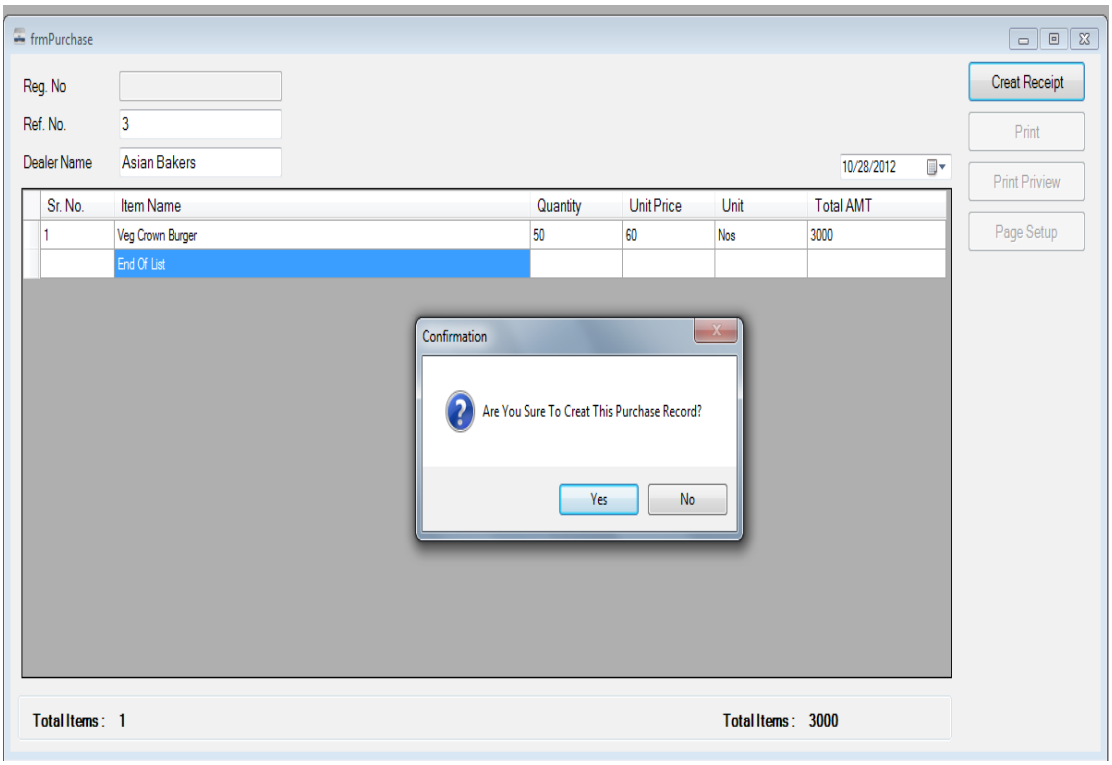
RECEIPT MASTER

Column Name	Data Type	Allow Nulls
RNo	int	
C_Name	varchar(50)	
Date	date	
TItem	int	
tamt	float	

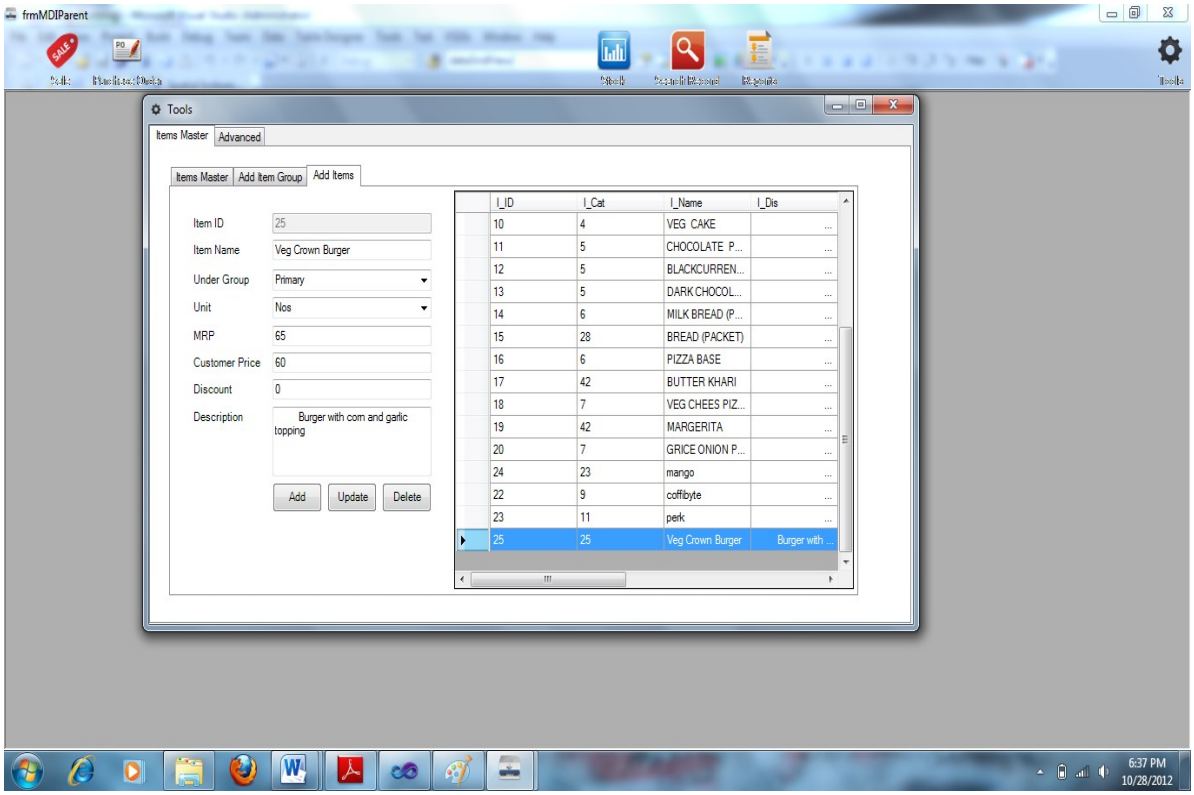
SCREEN SHOTS

LOGIN SCREEN

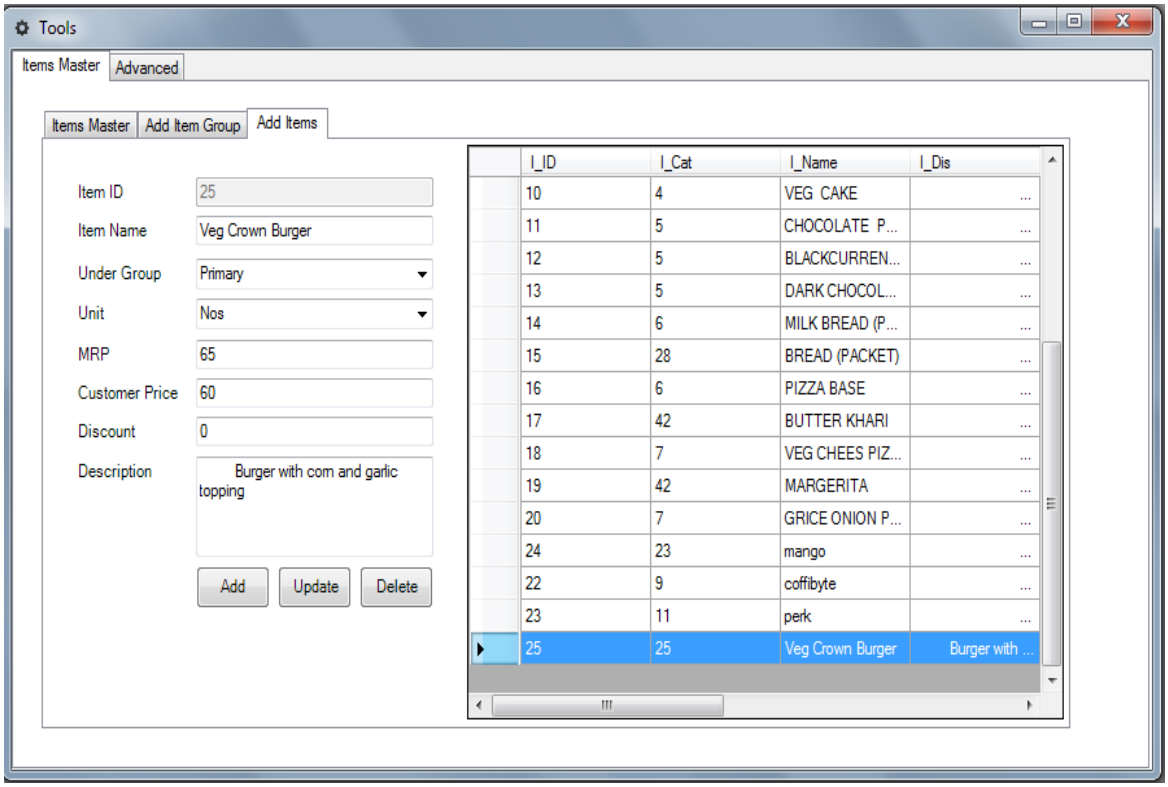
PURCHASE FORM



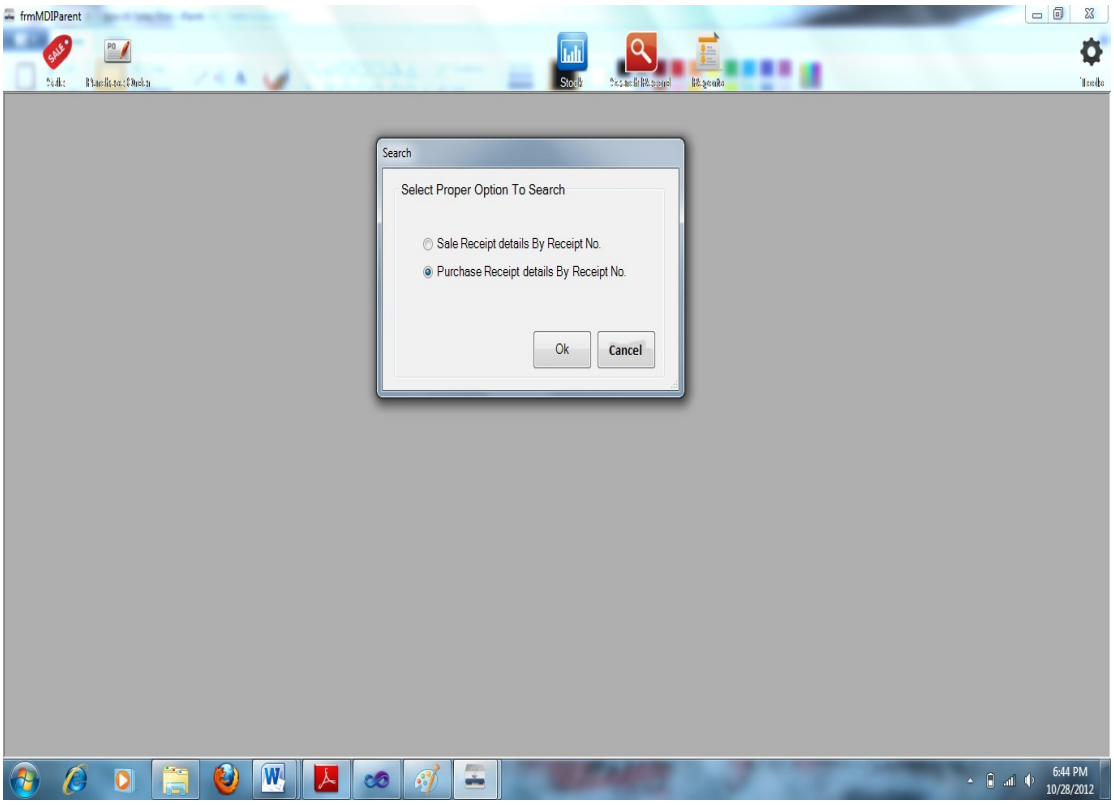
SALES FORM


frmMDIParent
Tools
Items Master
Advanced
Items Master
Add Item Group
Add Items
Item ID
25
Item Name
Veg Crown Burger
Under Group
Primary
Unit
Nos
MRP
65
Customer Price
60
Discount
0
Description
Burger with corn and garlic topping
Add
Update
Delete
I_ID
I_Cat
I_Name
I_Dis
10 4 VEG CAKE
11 5 CHOCOLATE P...
12 5 BLACKCURREN...
13 5 DARK CHOCOL...
14 6 MILK BREAD (P...
15 28 BREAD (PACKET)
16 6 PIZZA BASE
17 42 BUTTER KHARI
18 7 VEG CHEES PIZ...
19 42 MARGERITA
20 7 GRICE ONION P...
24 23 mango
22 9 coffibyte
23 11 perk
25 25 Veg Crown Burger Burger with ...
6:37 PM
10/28/2012

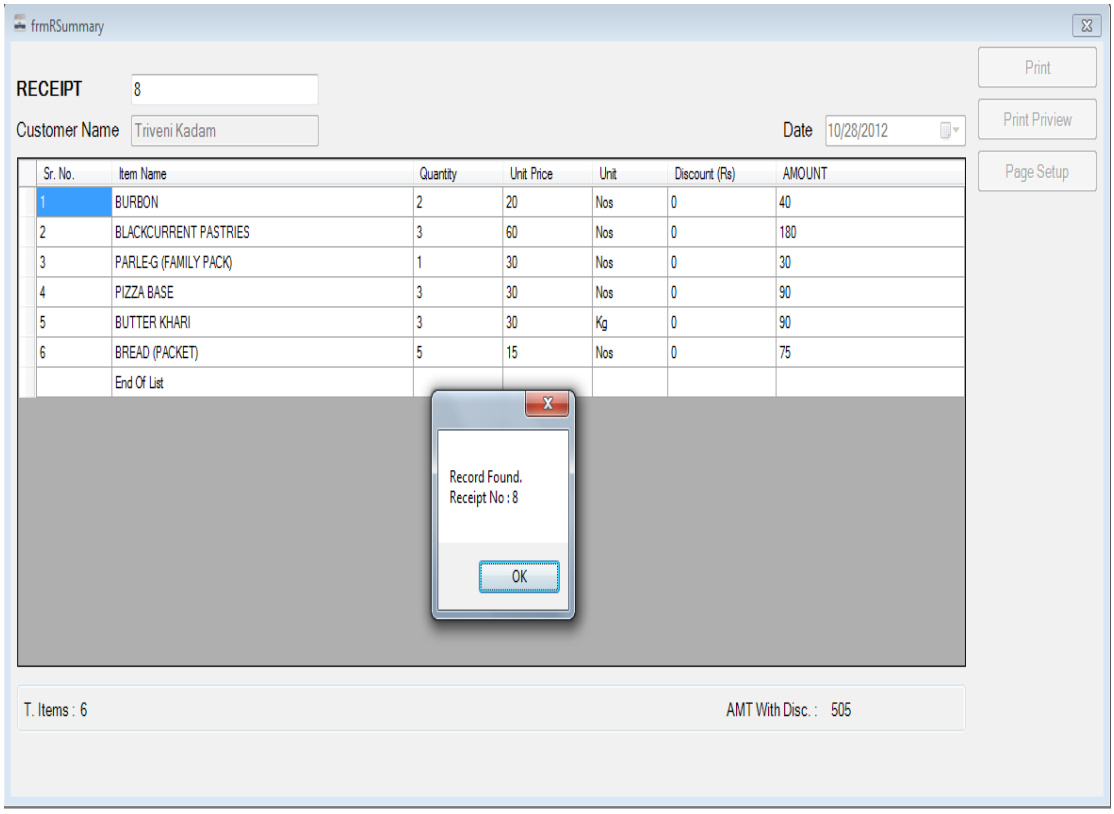
Add item Form



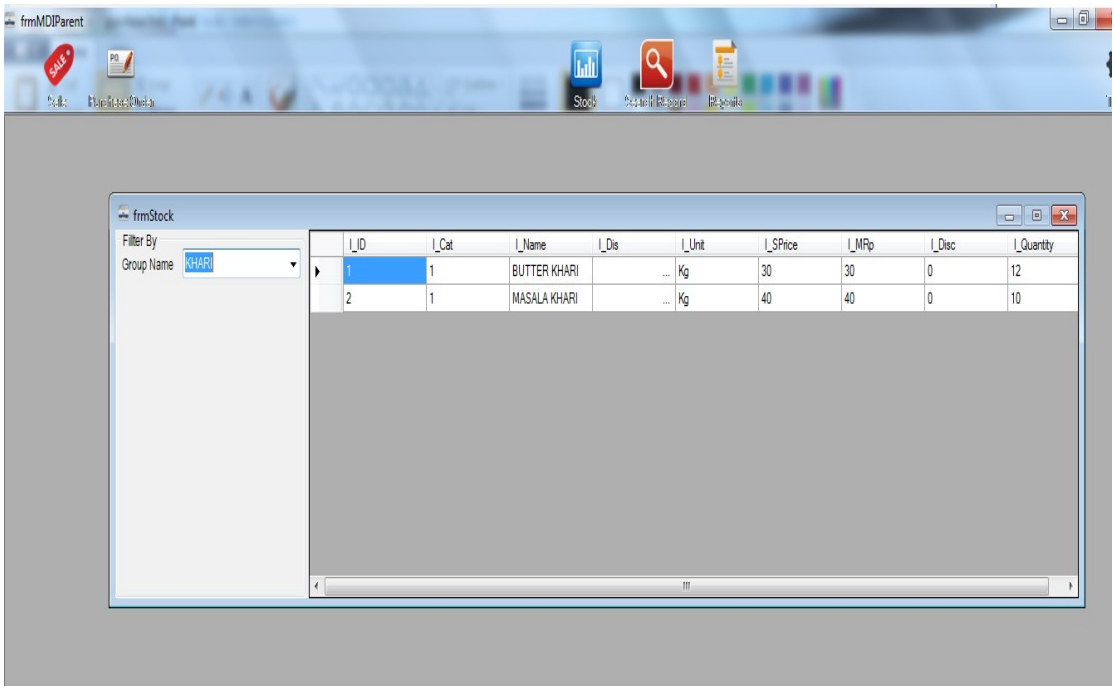
SEARCH FORM


frmMDIParent
Search
Select Proper Option To Search
Sale Receipt details By Receipt No.
Purchase Receipt details By Receipt No.
Ok
Cancel
6:44 PM
10/28/2012

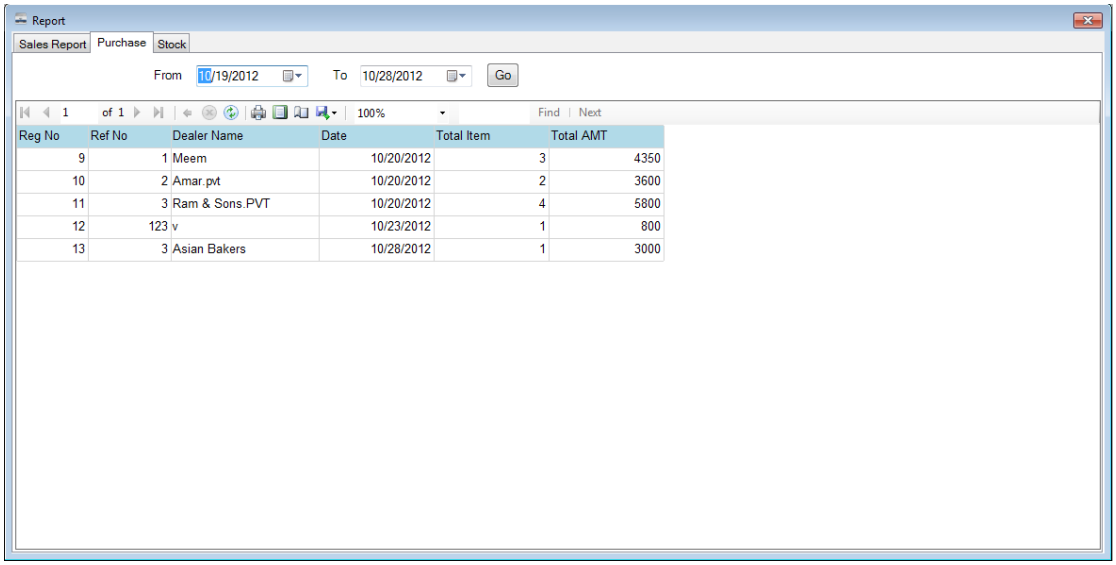
SALES SEARCH FORM



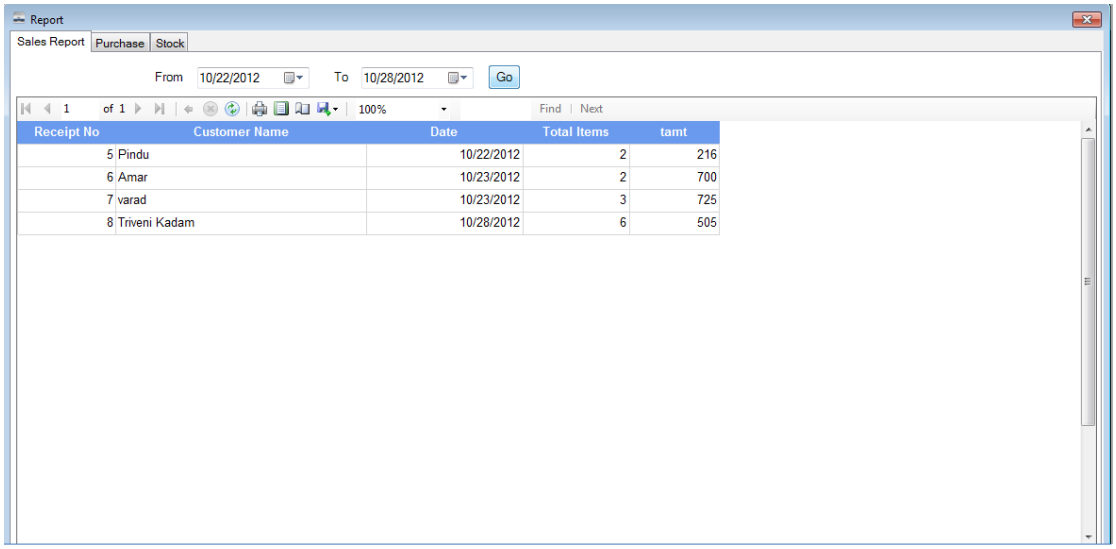
STOCK FORM



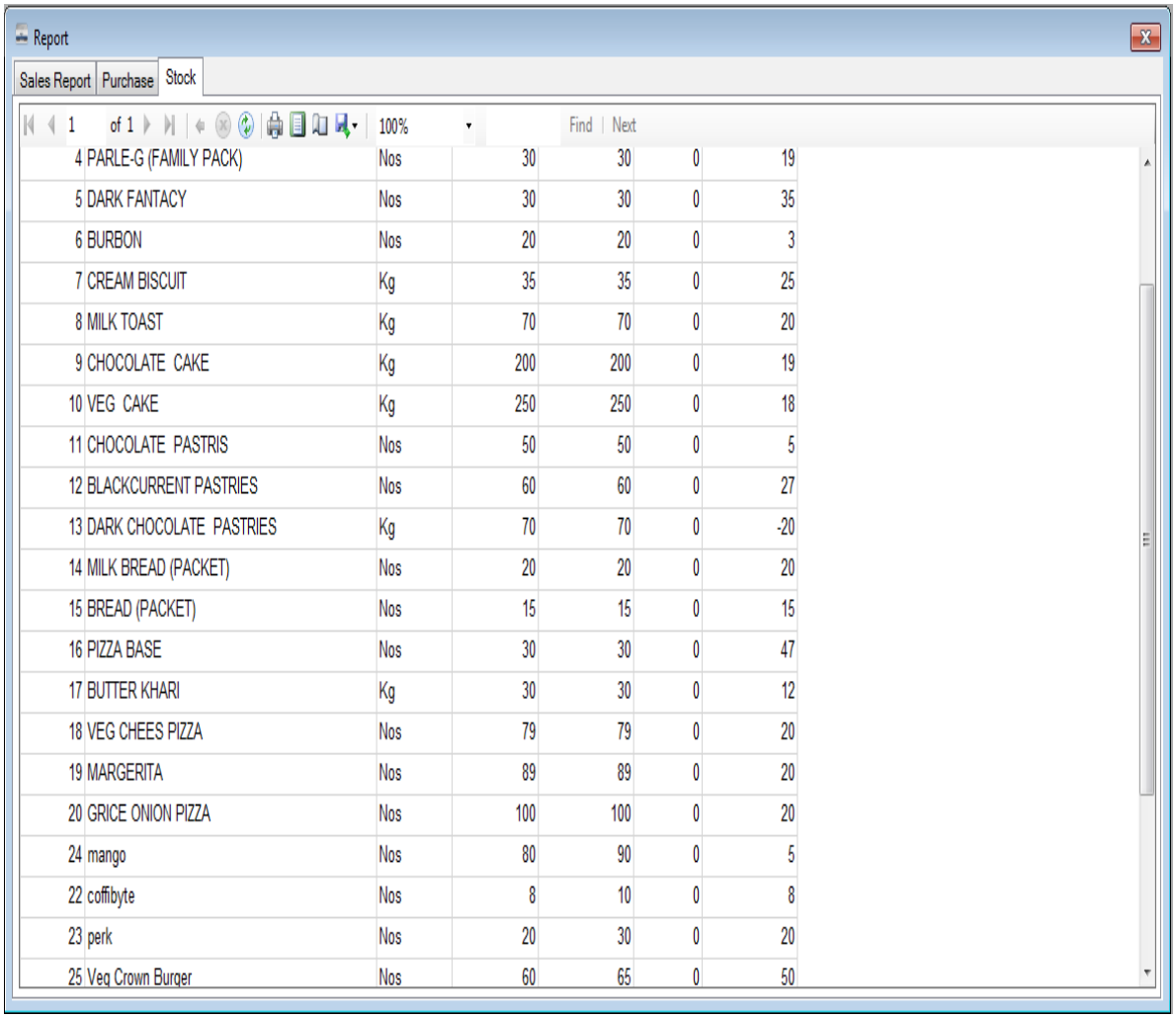
PURCHASE REPORT



SALES REPORT



STOCK REPORT



4	PARLE-G (FAMILY PACK)	Nos	30	30	0	19
5	DARK FANTACY	Nos	30	30	0	35
6	BURBON	Nos	20	20	0	3
7	CREAM BISCUIT	Kg	35	35	0	25
8	MILK TOAST	Kg	70	70	0	20
9	CHOCOLATE CAKE	Kg	200	200	0	19
10	VEG CAKE	Kg	250	250	0	18
11	CHOCOLATE PASTRIS	Nos	50	50	0	5
12	BLACKCURRENT PASTRIES	Nos	60	60	0	27
13	DARK CHOCOLATE PASTRIES	Kg	70	70	0	-20
14	MILK BREAD (PACKET)	Nos	20	20	0	20
15	BREAD (PACKET)	Nos	15	15	0	15
16	PIZZA BASE	Nos	30	30	0	47
17	BUTTER KHARI	Kg	30	30	0	12
18	VEG CHEES PIZZA	Nos	79	79	0	20
19	MARGERITA	Nos	89	89	0	20
20	GRICE ONION PIZZA	Nos	100	100	0	20
24	mango	Nos	80	90	0	5
22	coffibyte	Nos	8	10	0	8
23	perk	Nos	20	30	0	20
25	Veg Crown Burger	Nos	60	65	0	50

SOURCE CODE

Purchase form

```
using System;
```

```
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace Dairy_Mng
{
   public partial class frmPurchase : Form
   {
      #region Connection obj
      SqlConnection cn = new SqlConnection("Data
Source=.\\SQLEXPRESS;AttachDbFilename=D:\\Prj_Oct_2012\\Dairy_Mng\\Dairy_Mng\\d
b.mdf;Integrated Security=True;User Instance=True");
      SqlCommand cmd;
      SqlDataAdapter da;
      DataTable dt;
      #endregion

      #region gbl obj
      string fAdd = Properties.Settings.Default.Add;
      string mob = Properties.Settings.Default.Mob;
      TextBox txtName;
      bool edit;
      #endregion

      #region functions
      public int funMax()
      {
         int max = 0;
         try
         {
            cn.Open();
            cmd = new SqlCommand("SELECT MAX(RegNo) FROM tblPMaster", cn);
            max = Convert.ToInt32(cmd.ExecuteScalar());
         }
         catch (Exception ex)
         {
         }
         finally
         {
            cn.Close();
            max = max + 1;
         }
         return max;
      }
      public void funInitializedComp()
      {
         txtRegNo.Text = "";
         btnCRecipt.Text = "Creat Receipt";
         txtDName.Clear();
         txtRefNo.Clear();
         txtDName.Enabled = true;
         flp.Enabled = false;
         dgvMList.Rows.Clear();
         dgvMList.Rows.Add("1", "", "", "", "", "");
         lblTAMT.Text = "0";
         lblTItm.Text = "0";
         btnPrint.Enabled = false;
         btnPPV.Enabled = false;
```

```
            btnPSUp.Enabled = false;
            txtRefNo.Focus();
        }
        public void funArrangeSrNo()
        {
            float amt = 0;
            for (int i = 0; i < dgvMList.Rows.Count - 1; i++)
            {
                dgvMList.Rows[i].Cells[0].Value = i + 1;
                amt = amt + float.Parse(dgvMList.Rows[i].Cells[5].Value.ToString());
                lblTItm.Text = (i + 1).ToString();
            }
            lblTAMT.Text = amt.ToString();
        }
        #endregion

        public frmPurchase()
        {
            txtName = new TextBox();
            InitializeComponent();
        }

        private void frmPurchase_Load(object sender, EventArgs e)
        {
            #region Load Items Data
            try
            {
                cn.Open();
                cmd = new SqlCommand("SELECT * FROM tblItmMaster", cn);
                da = new SqlDataAdapter(cmd);
                dt = new DataTable();
                da.Fill(dt);
                for (int i = 0; i < dt.Rows.Count; i++)
                {

txtName.AutoCompleteCustomSource.Add(dt.Rows[i].ItemArray[2].ToString());
                }
                txtName.AutoCompleteCustomSource.Add("End Of List");
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
            finally
            {
                cn.Close();
            }
            #endregion
            funInitializedComp();
        }
        private void dgvMList_EditingControlShowing(object sender,
DataGridViewEditingControlShowingEventArgs e)
        {
            TextBox txtACS = e.Control as TextBox;
            txtACS.AutoCompleteMode = AutoCompleteMode.Suggest;
            txtACS.AutoCompleteSource = AutoCompleteSource.CustomSource;
            int swt = dgvMList.CurrentCell.ColumnIndex;
            switch (swt)
            {
                case 1:
                    txtACS.AutoCompleteCustomSource =
txtName.AutoCompleteCustomSource;
                    break;
```

```
            case 2:
                txtACS.AutoCompleteCustomSource = null;
                break;
            default:
                break;
        }
    }
    private void frmPurchase_FormClosing(object sender, FormClosingEventArgs e)
    {
        e.Cancel = false;
    }
    private void btnExit_Click(object sender, EventArgs e)
    {
        this.Close();
    }
    private void txtDName_Leave(object sender, EventArgs e)
    {
        if (string.IsNullOrWhiteSpace(txtDName.Text))
        {
            txtDName.Focus();
        }
    }
    private void txtRefNo_Leave(object sender, EventArgs e)
    {
        int rout;
        if (string.IsNullOrWhiteSpace(txtRefNo.Text) || int.TryParse(txtRefNo.Text,out
rout)==false)
        {
            txtRefNo.Focus();
        }
    }
    private void dgvMList_KeyDown(object sender, KeyEventArgs e)
    {
        if (e.KeyCode == Keys.Enter)
        {
            if (dgvMList.CurrentRow.Cells[5].Selected == true &&
dgvMList.CurrentRow.Index == dgvMList.Rows.Count-1 &&
                dgvMList.Rows[dgvMList.Rows.Count - 1].Cells[1].Value.ToString() != "End
Of List")
            {
                dgvMList.Rows.Add("", "", "", "", "", "");
                dgvMList.CurrentRow.Cells[1].Selected = true;
            }
        }

    }
    private void dgvMList_CellBeginEdit(object sender,
DataGridViewCellCancelEventArgs e)
    {
        edit = true;
    }
    private void dgvMList_CellEndEdit(object sender, DataGridViewCellEventArgs e)
    {
        edit = false;
    }
    private void dgvMList_RowsAdded(object sender,
DataGridViewRowsAddedEventArgs e)
    {
        funArrangeSrNo();
    }
    private void dgvMList_RowsRemoved(object sender,
DataGridViewRowsRemovedEventArgs e)
    {
```

```
            funArrangeSrNo();
        }
        #region menu validation
        private void dgvMList_CellValidating(object sender,
DataGridViewCellValidatingEventArgs e)
        {
            switch(e.ColumnIndex)
            {
                case 1:
                    #region M Name
                    if (string.IsNullOrWhiteSpace(e.FormattedValue.ToString()))
                    {
                        e.Cancel = true;
                    }
                    else if
(txtName.AutoCompleteCustomSource.Contains(e.FormattedValue.ToString()) ==
false)
                    {
                        MessageBox.Show("Spelling Error");
                        e.Cancel = true;
                    }
                    else
                    {

                    }
                    #endregion
                    break;
                case 2:
                    #region quantity
                    if (dgvMList.Rows[e.RowIndex].Cells[1].Value.ToString() != "End Of List")
                    {
                        float f;
                        if (string.IsNullOrWhiteSpace(e.FormattedValue.ToString()))
                        {
                            e.Cancel = true;
                        }
                        else if (float.TryParse(e.FormattedValue.ToString(), out f) == false)
                        {
                            e.Cancel = true;
                        }
                        else
                        {

                        }
                    }
                    #endregion
                    break;
                case 3:
                    #region U Price
                    if (dgvMList.Rows[e.RowIndex].Cells[1].Value.ToString() != "End Of List")
                    {
                        float f;
                        if (string.IsNullOrWhiteSpace(e.FormattedValue.ToString()))
                        {
                            e.Cancel = true;
                        }
                        else if (float.TryParse(e.FormattedValue.ToString(), out f) == false)
                        {
                            e.Cancel = true;
                        }
                        else
                        {
```

```
                }
            }
            #endregion
            break;
        default:
            break;
    }
}
private void dgvMList_CellValidated(object sender, DataGridViewCellEventArgs e)
{
    int r = e.RowIndex;
    int c = e.ColumnIndex;
    switch (c)
    {
        case 1:
            #region Menu Name
            if (edit == true)
            {
                if (dgvMList.Rows[e.RowIndex].Cells[1].Value.ToString() != "End Of
List")
                {
                    dgvMList.Rows[e.RowIndex].Cells[2].Value = "0";
                    int ilx =
txtName.AutoCompleteCustomSource.IndexOf(dgvMList.Rows[e.RowIndex].Cells[1].Val
ue.ToString());
                    dgvMList.Rows[e.RowIndex].Cells[3].Value = "0";
                    dgvMList.Rows[e.RowIndex].Cells[4].Value =
dt.Rows[ilx].ItemArray[4].ToString();
                    dgvMList.Rows[e.RowIndex].Cells[5].Value =
Convert.ToDecimal(dgvMList.Rows[e.RowIndex].Cells[2].Value) *

Convert.ToDecimal(dgvMList.Rows[e.RowIndex].Cells[3].Value);
                }
                else
                {
                    for (int i = 2; i < 5; i++)
                    {
                        dgvMList.CurrentRow.Cells[i].Value = "";
                    }
                    float amt = 0;
                    for (int i = 0; i < dgvMList.Rows.Count - 1; i++)
                    {
                        lblTItm.Text = (i + 1).ToString();
                        amt = amt +
float.Parse(dgvMList.Rows[i].Cells[5].Value.ToString());
                    }
                    flp.Enabled = true;
                    btnCRecipt.Focus();
                }
            }
            #endregion
            break;
        case 2:
            #region Quantity
            if (dgvMList.Rows[e.RowIndex].Cells[1].Value.ToString() != "End Of List")
            {
                dgvMList.Rows[e.RowIndex].Cells[5].Value =
Convert.ToDecimal(dgvMList.Rows[e.RowIndex].Cells[2].Value) *

Convert.ToDecimal(dgvMList.Rows[e.RowIndex].Cells[3].Value);
            }
            else
```

```
                {
                    dgvMList.Rows[e.RowIndex].Cells[2].Value = "";
                }
                #endregion
                break;
            case 3:
                #region Quantity
                if (dgvMList.Rows[e.RowIndex].Cells[1].Value.ToString() != "End Of List")
                {
                    dgvMList.Rows[e.RowIndex].Cells[5].Value =
Convert.ToDecimal(dgvMList.Rows[e.RowIndex].Cells[2].Value) *

Convert.ToDecimal(dgvMList.Rows[e.RowIndex].Cells[3].Value);
                }
                else
                {
                    dgvMList.Rows[e.RowIndex].Cells[3].Value = "";
                }
                #endregion
                break;
            default:
                break;
        }
    }
    #endregion
    private void btnCRecipt_Click(object sender, EventArgs e)
    {
        if (btnCRecipt.Text == "Creat Receipt")
        {
            if (MessageBox.Show("Are You Sure To Creat This Purchase Record?",
"Confirmation",
                MessageBoxButtons.YesNo, MessageBoxIcon.Question) ==
DialogResult.Yes)
            {
                int max = funMax();
                #region save & update record
                try
                {
                    cn.Open();
                    cmd = new SqlCommand("INSERT INTO tblPMaster (RegNo, RefNo,
RefName, Date, TItem, TAMT) VALUES (@regno,@refno,@name,@dt,@ti,@amt)", cn);
                    cmd.Parameters.Add("@regno", SqlDbType.Int).Value = max;
                    cmd.Parameters.Add("@refno", SqlDbType.Int).Value =
int.Parse(txtRefNo.Text);
                    cmd.Parameters.Add("@name", SqlDbType.VarChar).Value =
txtDName.Text;
                    cmd.Parameters.Add("@dt", SqlDbType.Date).Value = dtp.Value;
                    cmd.Parameters.Add("@ti", SqlDbType.Int).Value =
int.Parse(lblTItm.Text);
                    cmd.Parameters.Add("@amt", SqlDbType.Float).Value =
int.Parse(lblTAMT.Text);
                    cmd.ExecuteReader();
                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message);
                }
                finally
                {
                    cn.Close();
                }
                try
                {
```

```
                    for (int i = 0; i < dgvMList.Rows.Count - 1; i++)
                    {
                        cn.Open();
                        cmd = new SqlCommand("INSERT INTO tblPDetails (RegNo, IName,
Quantity, UPrice, Unit, TAMT) VALUES (@no,@nm,@qt,@up,@u,@amt)", cn);
                        cmd.Parameters.Add("@no", SqlDbType.Int).Value = max;
                        cmd.Parameters.Add("@nm", SqlDbType.VarChar).Value =
dgvMList.Rows[i].Cells[1].Value.ToString();
                        cmd.Parameters.Add("@qt", SqlDbType.Int).Value =
int.Parse(dgvMList.Rows[i].Cells[2].Value.ToString());
                        cmd.Parameters.Add("@up", SqlDbType.Float).Value =
float.Parse(dgvMList.Rows[i].Cells[3].Value.ToString());
                        cmd.Parameters.Add("@u", SqlDbType.VarChar).Value =
dgvMList.Rows[i].Cells[4].Value.ToString();
                        cmd.Parameters.Add("@amt", SqlDbType.Float).Value =
float.Parse(dgvMList.Rows[i].Cells[5].Value.ToString());
                        cmd.ExecuteReader();
                        cn.Close();
                    }
                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message);
                }
                finally
                {
                }
                try
                {
                    for (int i = 0; i < dgvMList.Rows.Count - 1; i++)
                    {
                        cn.Open();
                        cmd = new SqlCommand("UPDATE tblItmMaster SET  I_Quantity =
I_Quantity + @qt WHERE (I_Name = @nm)", cn);
                        cmd.Parameters.Add("@qt", SqlDbType.Float).Value =
float.Parse(dgvMList.Rows[i].Cells[2].Value.ToString());
                        cmd.Parameters.Add("@nm", SqlDbType.VarChar).Value =
dgvMList.Rows[i].Cells[1].Value.ToString();
                        cmd.ExecuteReader();
                        cn.Close();
                    }
                    txtRegNo.Text = max.ToString();
                    MessageBox.Show("Purchase Record Added Successfully.");
                    btnCRecipt.Text = "Creat New";
                    btnPrint.Enabled = true;
                    btnPPV.Enabled = true;
                    btnPSUp.Enabled = true;
                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message);
                }
                finally
                {
                }
                #endregion
            }
        }
        else
        {
            btnCRecipt.Text = "Create Receipt";
            funInitializedComp();
        }
```

```
        }
        private void btnPrint_Click(object sender, EventArgs e)
        {
            pDoc.Print();
        }
        private void btnPPV_Click(object sender, EventArgs e)
        {
            ppViewD.ShowDialog();
        }
        private void btnPSUp_Click(object sender, EventArgs e)
        {
            pSetUp.ShowDialog();
        }

        private void pDoc_PrintPage(object sender,
System.Drawing.Printing.PrintPageEventArgs e)
        {
            int pWidth = pSetUp.PageSettings.PaperSize.Width - 10;
            int pHeight = pSetUp.PageSettings.PaperSize.Height;
            #region Font
            Font fName = new System.Drawing.Font("Arial Black", 30, FontStyle.Bold);
            Font ffAdd = new System.Drawing.Font("Arial", 14, FontStyle.Bold);
            Font ffListH = new System.Drawing.Font("Arial", 11, FontStyle.Bold);
            #endregion

            #region header
            e.Graphics.DrawString("BAKE-WELL", fName, System.Drawing.Brushes.Blue,
(pWidth / 2) - 105, 25);
            e.Graphics.DrawString(fAdd, ffAdd, System.Drawing.Brushes.Black, pWidth -
fAdd.Length * 11, 80);
            e.Graphics.DrawString(mob, ffAdd, System.Drawing.Brushes.Black, pWidth -
mob.Length * 14, 100);
            e.Graphics.DrawLine(Pens.Blue, 10, 130, pWidth - 10, 130);
            #endregion
            #region recipt Details
            e.Graphics.DrawString("Reg No: " + txtRegNo.Text, ffAdd,
System.Drawing.Brushes.Black, 20, 150);
            e.Graphics.DrawString("Date: " + dtp.Text, ffAdd,
System.Drawing.Brushes.Black, pWidth - mob.Length * 14, 150);
            e.Graphics.DrawString("Ref. No: " + txtRefNo.Text, ffAdd,
System.Drawing.Brushes.Black, 20, 180);
            e.Graphics.DrawString("Dealer Name:" + txtDName.Text, ffAdd,
System.Drawing.Brushes.Black, 20, 210);
            e.Graphics.DrawString("Menu List:", ffListH, System.Drawing.Brushes.Blue, 20,
260);
            #endregion
            #region Menu List
            Rectangle MHeader = new Rectangle(20, 285, pWidth - 40, 35);
            Rectangle SrNo = new Rectangle(20, 285, 80, 35);
            Rectangle IName = new Rectangle(100, 285, 300, 35);
            Rectangle Q = new Rectangle(400, 285, 100, 35);
            Rectangle uP = new Rectangle(500, 285, 100, 35);
            Rectangle u = new Rectangle(600, 285, 80, 35);
            Rectangle amt = new Rectangle(680, 285, 140, 35);
            e.Graphics.FillRectangle(Brushes.LightBlue, MHeader);

            Font fmHeader = new System.Drawing.Font("Arial", 11, FontStyle.Bold);
            StringFormat sf = new StringFormat();
            sf.Alignment = StringAlignment.Center;
            e.Graphics.DrawRectangle(Pens.Blue, SrNo);
            e.Graphics.DrawRectangle(Pens.Blue, IName);
            e.Graphics.DrawRectangle(Pens.Blue, Q);
            e.Graphics.DrawRectangle(Pens.Blue, u);
```

```
            e.Graphics.DrawRectangle(Pens.Blue, uP);
            e.Graphics.DrawRectangle(Pens.Blue, amt);

            e.Graphics.DrawString("Sr No", fmHeader, System.Drawing.Brushes.Blue,
SrNo, sf);
            e.Graphics.DrawString("Item Name", fmHeader, System.Drawing.Brushes.Blue,
IName, sf);
            e.Graphics.DrawString("Quantity", fmHeader, System.Drawing.Brushes.Blue,
Q, sf);
            e.Graphics.DrawString("Unit", fmHeader, System.Drawing.Brushes.Blue, u, sf);
            e.Graphics.DrawString("Rs/Unit", fmHeader, System.Drawing.Brushes.Blue, uP,
sf);
            e.Graphics.DrawString("Total AMT", fmHeader, System.Drawing.Brushes.Blue,
amt, sf);
            #endregion

            #region Menu List Item
            SrNo.Height = 30;
            IName.Height = 30;
            Q.Height = 30;
            uP.Height = 30;
            u.Height = 30;
            amt.Height = 30;
            Font fitm = new System.Drawing.Font("Arial", 10, FontStyle.Regular);
            sf.LineAlignment = StringAlignment.Center;
            int y = 320;
            for (int i = 0; i < dgvMList.Rows.Count - 1; i++)
            {
                SrNo.Y = y;
                IName.Y = y;
                Q.Y = y;
                u.Y = y;
                uP.Y = y;
                amt.Y = y;
                e.Graphics.DrawRectangle(Pens.Black, SrNo);
                e.Graphics.DrawRectangle(Pens.Black, IName);
                e.Graphics.DrawRectangle(Pens.Black, Q);
                e.Graphics.DrawRectangle(Pens.Black, u);
                e.Graphics.DrawRectangle(Pens.Black, uP);
                e.Graphics.DrawRectangle(Pens.Black, amt);

                e.Graphics.DrawString((i + 1).ToString(), fitm, Brushes.Black, SrNo, sf);
                e.Graphics.DrawString(dgvMList.Rows[i].Cells[1].Value.ToString(), fitm,
Brushes.Black, IName, sf);
                e.Graphics.DrawString(dgvMList.Rows[i].Cells[2].Value.ToString(), fitm,
Brushes.Black, Q, sf);
                e.Graphics.DrawString(dgvMList.Rows[i].Cells[3].Value.ToString(), fitm,
Brushes.Black, uP, sf);
                e.Graphics.DrawString(dgvMList.Rows[i].Cells[4].Value.ToString(), fitm,
Brushes.Black, u, sf);
                e.Graphics.DrawString(dgvMList.Rows[i].Cells[5].Value.ToString(), fitm,
Brushes.Black, amt, sf);

                y += 30;
            }
            #endregion
            #region total Menu
            e.Graphics.DrawString("Total :", fmHeader, System.Drawing.Brushes.Blue, 20,
y + 15);
            y = y + 40;
            MHeader.Y = y;
            sf.LineAlignment = StringAlignment.Center;
```

```
            Rectangle TI = new Rectangle(20, y, 150, 35);
            Rectangle blank = new Rectangle(170, y, 430, 35);
            Rectangle TAMT = new Rectangle(710, y, 110, 35);
            e.Graphics.FillRectangle(Brushes.LightPink, MHeader);
            e.Graphics.DrawRectangle(Pens.Red, MHeader);
            e.Graphics.DrawString("Total Item", fmHeader, System.Drawing.Brushes.Blue,
TI, sf);
            e.Graphics.DrawString("", fmHeader, System.Drawing.Brushes.Blue, blank, sf);
            e.Graphics.DrawString("Total AMT", fmHeader, System.Drawing.Brushes.Blue,
TAMT, sf);
            sf.LineAlignment = StringAlignment.Center;
            y = y + 35;
            TI.Y = y;
            TAMT.Y = y;
            MHeader.Y = y;

            e.Graphics.FillRectangle(Brushes.LightBlue, MHeader);
            e.Graphics.DrawRectangle(Pens.Red, MHeader);
            e.Graphics.DrawString((dgvMList.Rows.Count - 1).ToString(), fmHeader,
System.Drawing.Brushes.Black, TI, sf);
            e.Graphics.DrawString(lblTAMT.Text, fmHeader, System.Drawing.Brushes.Black,
TAMT, sf);

            #endregion
            #region Footer
            e.Graphics.DrawString("Sign & Stamp Here", fmHeader,
System.Drawing.Brushes.Black, 600, y + 60);
            #endregion
        }


    }
}
```

Sales form

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Windows.Forms;
using System.Data.SqlClient;

namespace Dairy_Mng
{
    public partial class frmSale : Form
    {
        #region Connection obj
        SqlConnection cn = new SqlConnection("Data
Source=.\\SQLEXPRESS;AttachDbFilename=D:\\Prj_Oct_2012\\Dairy_Mng\\Dairy_Mng\\d
b.mdf;Integrated Security=True;User Instance=True");
        SqlCommand cmd;
        SqlDataAdapter da;
        DataTable dt;
        #endregion

        #region gbl obj
        string fAdd = Properties.Settings.Default.Add;
        string mob = Properties.Settings.Default.Mob;
        clsUnitConvertor clsUC = new clsUnitConvertor();
        TextBox txtName;
        frmMViewer fmv;
```

```
        bool edit;
        #endregion

        #region functions
        public int funMax()
        {
            int max = 0;
            try
            {
                cn.Open();
                cmd = new SqlCommand("SELECT MAX(RNo) FROM tblRecipt", cn);
                max = Convert.ToInt32(cmd.ExecuteScalar());
            }
            catch (Exception ex)
            {
            }
            finally
            {
                cn.Close();
                max = max + 1;
            }
            return max;
        }
        public void funInitializedComp()
        {
            txtCName.Text = "";
            txtCName.Enabled = true;
            dgvMList.Enabled = true;
            txtRNo.Text = "";
            dgvMList.Rows.Clear();
            dgvMList.Rows.Add("1", "", "", "", "", "","");
            lblTAMT.Text = "0";
            lblTItm.Text = "0";
            flp.Enabled = false;
            btnPrint.Enabled = false;
            btnPPV.Enabled = false;
            btnPSUp.Enabled = false;
            txtCName.Focus();
        }
        public void funLoadItemsData()
        {
            try
            {
                cn.Open();
                cmd = new SqlCommand("SELECT * FROM tblItmMaster", cn);
                da = new SqlDataAdapter(cmd);
                dt = new DataTable();
                da.Fill(dt);
                for (int i = 0; i < dt.Rows.Count; i++)
                {

txtName.AutoCompleteCustomSource.Add(dt.Rows[i].ItemArray[2].ToString());
                }
                txtName.AutoCompleteCustomSource.Add("End Of List");
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
            finally
            {
                cn.Close();
            }
```

```
        }
        public void funMngSrno()
        {
            for (int i=0; i < dgvMList.Rows.Count - 1; i++)
            {
                dgvMList.Rows[i].Cells[0].Value = i+1;
            }
        }
        public bool funCheckIsExist(string s, int d)
        {
            bool b = false;
            for (int i = 0; i < dgvMList.Rows.Count - 1; i++)
            {
                if (dgvMList.Rows[i].Cells[1].Value.ToString() == s && d!=i)
                {
                    b = true;
                    break;
                }
            }
            return b;
        }
        #endregion

        public frmSale()
        {
            txtName = new TextBox();
            fmv = new frmMViewer();
            InitializeComponent();
            fmv.Location = new Point(dgvMList.Location.X + 100, dgvMList.Location.Y);
        }

        private void frmSale_Load(object sender, EventArgs e)
        {
            funInitializedComp();
            funLoadItemsData();
        }
        private void frmSale_FormClosing(object sender, FormClosingEventArgs e)
        {
            e.Cancel = false;
        }
        private void txtCName_Leave(object sender, EventArgs e)
        {
            if (string.IsNullOrWhiteSpace(txtCName.Text))
            {
                txtCName.Focus();
            }
        }
        private void dgvMList_EditingControlShowing(object sender,
DataGridViewEditingControlShowingEventArgs e)
        {
            TextBox txtACS = e.Control as TextBox;
            txtACS.AutoCompleteMode = AutoCompleteMode.Suggest;
            txtACS.AutoCompleteSource = AutoCompleteSource.CustomSource;
            int swt = dgvMList.CurrentCell.ColumnIndex;
            switch (swt)
            {
                case 1:
                    txtACS.AutoCompleteCustomSource =
txtName.AutoCompleteCustomSource;
                    break;
                case 2:
                    txtACS.AutoCompleteCustomSource = null;
                    break;
```

```
            case 3:
                txtACS.AutoCompleteCustomSource = null;
                break;
            case 4:
                txtACS.AutoCompleteCustomSource = null;
                break;
            case 5:
                txtACS.AutoCompleteCustomSource = null;
                break;
            case 6:
                txtACS.AutoCompleteCustomSource = null;
                break;
            default:
                break;
        }
    }
    private void dgvMList_CellBeginEdit(object sender,
DataGridViewCellCancelEventArgs e)
    {
        edit = true;
        if (e.ColumnIndex == 1)
        {
            if (fmv.ShowDialog() == DialogResult.OK)
            {
                dgvMList.CurrentRow.Cells[1].Value = fmv.selecteditm;
            }
        }
    }
    private void dgvMList_CellEnter(object sender, DataGridViewCellEventArgs e)
    {
        edit = false;
    }
    private void dgvMList_KeyDown(object sender, KeyEventArgs e)
    {
        if (e.KeyCode == Keys.Enter)
        {
            if (dgvMList.CurrentRow.Cells[6].Selected == true &&
dgvMList.CurrentRow.Index == dgvMList.Rows.Count - 1 &&
                dgvMList.Rows[dgvMList.Rows.Count - 1].Cells[1].Value.ToString() != "End
Of List")
            {
                dgvMList.Rows.Add("", "", "", "", "", "", "");
                dgvMList.CurrentRow.Cells[1].Selected = true;
            }
        }
    }
    private void dgvMList_RowsAdded(object sender,
DataGridViewRowsAddedEventArgs e)
    {
        funMngSrno();
    }
    private void dgvMList_RowsRemoved(object sender,
DataGridViewRowsRemovedEventArgs e)
    {
        funMngSrno();
    }
    private void dgvMList_CellValidating(object sender,
DataGridViewCellValidatingEventArgs e)
    {
        switch (e.ColumnIndex)
        {
            case 1:
                #region M Name
```

```
                if (string.IsNullOrWhiteSpace(e.FormattedValue.ToString()))
                {
                    e.Cancel = true;
                }
                else if
(txtName.AutoCompleteCustomSource.Contains(e.FormattedValue.ToString()) ==
false)
                {
                    MessageBox.Show("Spelling Error");
                    e.Cancel = true;
                }
                if (funCheckIsExist(e.FormattedValue.ToString(), e.RowIndex) && edit ==
true)
                {
                    MessageBox.Show("Item Is All Reaady Exist In List");
                    e.Cancel = true;
                    edit = false;
                }

                #endregion
                break;
            case 2:
                #region quantity
                if (dgvMList.Rows[e.RowIndex].Cells[1].Value.ToString() != "End Of List")
                {
                    int i;
                    float f;
                    if (string.IsNullOrWhiteSpace(e.FormattedValue.ToString()))
                    {
                        e.Cancel = true;
                    }
                    else if (float.TryParse(e.FormattedValue.ToString(), out f) == false)
                    {
                        e.Cancel = true;
                        MessageBox.Show("Quantity Must Be Neumric", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                    }
                    if (int.TryParse(e.FormattedValue.ToString(), out i) == false &&
dgvMList.CurrentRow.Cells[4].Value.ToString() == "Nos" &&
float.TryParse(e.FormattedValue.ToString(), out f) == true)
                    {
                        MessageBox.Show("Quantity Can Not Decimal For Unit Nos", "Error",
MessageBoxButtons.OK, MessageBoxIcon.Error);
                        e.Cancel = true;
                    }

                }
                #endregion
                break;
            case 5:
                #region U Price
                if (dgvMList.Rows[e.RowIndex].Cells[1].Value.ToString() != "End Of List")
                {
                    float f;
                    if (string.IsNullOrWhiteSpace(e.FormattedValue.ToString()))
                    {
                        e.Cancel = true;
                    }
                    else if (float.TryParse(e.FormattedValue.ToString(), out f) == false)
                    {
                        e.Cancel = true;
                    }
                    else
```

```
                {

                }
            }
            #endregion
            break;
        default:
            break;
    }
}
private void dgvMList_CellLeave(object sender, DataGridViewCellEventArgs e)
{
    int r = e.RowIndex;
    int c = e.ColumnIndex;
    switch (c)
    {
        case 1:
            #region Menu Name
            if (edit == true)
            {
                if (dgvMList.Rows[e.RowIndex].Cells[1].Value.ToString() != "End Of
List")
                {
                    try
                    {
                        dgvMList.Rows[e.RowIndex].Cells[2].Value = "0";
                        int ilx =
txtName.AutoCompleteCustomSource.IndexOf(dgvMList.Rows[e.RowIndex].Cells[1].Val
ue.ToString());
                        dgvMList.Rows[e.RowIndex].Cells[3].Value =
dt.Rows[ilx].ItemArray[5].ToString();
                        dgvMList.Rows[e.RowIndex].Cells[4].Value =
dt.Rows[ilx].ItemArray[4].ToString();
                        dgvMList.Rows[e.RowIndex].Cells[5].Value =
dt.Rows[ilx].ItemArray[7].ToString();
                        dgvMList.Rows[e.RowIndex].Cells[6].Value = "0";
                    }
                    catch (Exception ex)
                    {
                    }
                    finally { }
                }
                else
                {
                    for (int i = 2; i < 6; i++)
                    {
                        dgvMList.CurrentRow.Cells[i].Value = "";
                    }
                    float amt = 0;
                    for (int i = 0; i < dgvMList.Rows.Count-1; i++)
                    {
                        lblTItm.Text = (i+1).ToString();
                        amt = amt +
float.Parse(dgvMList.Rows[i].Cells[6].Value.ToString());

                    }
                    lblTAMT.Text = amt.ToString();
                    flp.Enabled = true;
                    btnCRecipt.Focus();
                }
            }
            #endregion
            break;
```

```
            case 2:
               #region Quantity
               if (dgvMList.Rows[e.RowIndex].Cells[1].Value.ToString() != "End Of List")
               {
                  dgvMList.Rows[e.RowIndex].Cells[6].Value =
float.Parse(dgvMList.Rows[e.RowIndex].Cells[2].Value.ToString()) *

float.Parse(dgvMList.Rows[e.RowIndex].Cells[3].Value.ToString()) -

float.Parse(dgvMList.Rows[e.RowIndex].Cells[2].Value.ToString()) *

float.Parse(dgvMList.Rows[e.RowIndex].Cells[5].Value.ToString())
                                                      ;
               }
               else
               {
                  dgvMList.Rows[e.RowIndex].Cells[2].Value = "";
               }
               #endregion
               break;
            case 5:
               #region Quantity
               if (dgvMList.Rows[e.RowIndex].Cells[1].Value.ToString() != "End Of List")
               {
                  dgvMList.Rows[e.RowIndex].Cells[6].Value =
float.Parse(dgvMList.Rows[e.RowIndex].Cells[2].Value.ToString()) *

float.Parse(dgvMList.Rows[e.RowIndex].Cells[3].Value.ToString()) -

float.Parse(dgvMList.Rows[e.RowIndex].Cells[2].Value.ToString()) *

float.Parse(dgvMList.Rows[e.RowIndex].Cells[5].Value.ToString())
                                                      ;
               }
               else
               {
                  dgvMList.Rows[e.RowIndex].Cells[5].Value = "";
               }
               break;
               #endregion
            default:
               break;
         }
      }
      private void dgvMList_CellValidated(object sender, DataGridViewCellEventArgs e)
      {
         int r = e.RowIndex;
         int c = e.ColumnIndex;
         switch (c)
         {
            case 1:
               #region Menu Name
               if (edit == true)
               {
                  if (dgvMList.Rows[e.RowIndex].Cells[1].Value.ToString() != "End Of
List")
                  {
                     dgvMList.Rows[e.RowIndex].Cells[2].Value = "0";
                     int ilx =
txtName.AutoCompleteCustomSource.IndexOf(dgvMList.Rows[e.RowIndex].Cells[1].Val
ue.ToString());
```

```
                    dgvMList.Rows[e.RowIndex].Cells[3].Value =
dt.Rows[ilx].ItemArray[5].ToString();
                    dgvMList.Rows[e.RowIndex].Cells[4].Value =
dt.Rows[ilx].ItemArray[4].ToString();
                    dgvMList.Rows[e.RowIndex].Cells[5].Value =
dt.Rows[ilx].ItemArray[7].ToString();
                    dgvMList.Rows[e.RowIndex].Cells[6].Value = "0";
                }
                else
                {
                    for (int i = 2; i < 6; i++)
                    {
                        dgvMList.CurrentRow.Cells[i].Value = "";
                    }
                    float amt=0;
                    for (int i = 0; i < dgvMList.Rows.Count-1; i++)
                    {
                        lblTItm.Text = (i+1).ToString();
                        amt = amt +
float.Parse(dgvMList.Rows[i].Cells[6].Value.ToString());

                    }
                    lblTAMT.Text = amt.ToString();
                    flp.Enabled = true;
                    btnCRecipt.Focus();
                }
            }
            #endregion
            break;
        case 2:
            #region Quantity
            if (dgvMList.Rows[e.RowIndex].Cells[1].Value.ToString() != "End Of List")
            {
                dgvMList.Rows[e.RowIndex].Cells[6].Value =
float.Parse(dgvMList.Rows[e.RowIndex].Cells[2].Value.ToString()) *

float.Parse(dgvMList.Rows[e.RowIndex].Cells[3].Value.ToString()) -

float.Parse(dgvMList.Rows[e.RowIndex].Cells[2].Value.ToString()) *

float.Parse(dgvMList.Rows[e.RowIndex].Cells[5].Value.ToString())
                                                    ;
            }
            else
            {
                dgvMList.Rows[e.RowIndex].Cells[2].Value = "";
            }
            #endregion
            break;
        case 5:
            #region Quantity
            if (dgvMList.Rows[e.RowIndex].Cells[1].Value.ToString() != "End Of List")
            {
                dgvMList.Rows[e.RowIndex].Cells[6].Value =
float.Parse(dgvMList.Rows[e.RowIndex].Cells[2].Value.ToString()) *

float.Parse(dgvMList.Rows[e.RowIndex].Cells[3].Value.ToString()) -

float.Parse(dgvMList.Rows[e.RowIndex].Cells[2].Value.ToString()) *

float.Parse(dgvMList.Rows[e.RowIndex].Cells[5].Value.ToString())
                                                    ;
```

```
                }
                else
                {
                    dgvMList.Rows[e.RowIndex].Cells[5].Value = "";
                }
                break;
                #endregion
            default:
                break;
        }
    }
    private void btnCRecipt_Click(object sender, EventArgs e)
    {
        if (btnCRecipt.Text == "Creat Receipt")
        {
            if (MessageBox.Show("Are You Sure To Creat Recipt?", "Confirmation",
MessageBoxButtons.YesNo) == DialogResult.Yes)
            {
                int max = funMax();
                txtRNo.Text = max.ToString();
                #region recipt basic
                try
                {
                    cn.Open();
                    cmd = new SqlCommand("INSERT INTO tblRecipt (RNo, C_Name, Date,
TItem, tamt) VALUES (@no,@nm,@dt,@titm,@amt)", cn);
                    cmd.Parameters.Add("@no", SqlDbType.Int).Value = max;
                    cmd.Parameters.Add("@nm", SqlDbType.VarChar).Value =
txtCName.Text;
                    cmd.Parameters.Add("@dt", SqlDbType.Date).Value = dtp.Value;
                    cmd.Parameters.Add("@titm", SqlDbType.Int).Value =
int.Parse(lblTItm.Text);
                    cmd.Parameters.Add("@amt", SqlDbType.Float).Value =
float.Parse(lblTAMT.Text);
                    cmd.ExecuteReader();
                }
                catch (Exception ex)
                {
                    MessageBox.Show(ex.Message);
                }
                finally
                {
                    cn.Close();
                }
                #endregion
                #region details
                try
                {
                    for (int i = 0; i < dgvMList.Rows.Count - 1; i++)
                    {
                        cmd = new SqlCommand("INSERT INTO tblRDetails (R_No, IName,
Quantity, UPrice, Unit, disc, amt) VALUES (@no,@nm,@qt,@up,@u,@disc,@amt)", cn);
                        cn.Open();
                        cmd.Parameters.Add("@no", SqlDbType.Int).Value = max;
                        cmd.Parameters.Add("@nm", SqlDbType.VarChar).Value =
dgvMList.Rows[i].Cells[1].Value.ToString();
                        cmd.Parameters.Add("@qt", SqlDbType.Float).Value =
float.Parse(dgvMList.Rows[i].Cells[2].Value.ToString());
                        cmd.Parameters.Add("@up", SqlDbType.VarChar).Value =
dgvMList.Rows[i].Cells[3].Value.ToString();
                        cmd.Parameters.Add("@u", SqlDbType.VarChar).Value =
dgvMList.Rows[i].Cells[4].Value.ToString();
```

```
                    cmd.Parameters.Add("@disc", SqlDbType.Float).Value =
float.Parse(dgvMList.Rows[i].Cells[5].Value.ToString());
                    cmd.Parameters.Add("@amt", SqlDbType.Float).Value =
float.Parse(dgvMList.Rows[i].Cells[6].Value.ToString());
                    cmd.ExecuteReader();
                    cn.Close();
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
            finally
            {
            }
            #endregion
            #region Update Item stock
            try
            {
                for (int i = 0; i < dgvMList.Rows.Count - 1; i++)
                {
                    cmd = new SqlCommand("UPDATE tblItmMaster SET I_Quantity =
I_Quantity - @qt WHERE (I_Name = @nm)", cn);
                    cn.Open();
                    cmd.Parameters.Add("@qt", SqlDbType.Float).Value =
float.Parse(dgvMList.Rows[i].Cells[2].Value.ToString());
                    cmd.Parameters.Add("@nm", SqlDbType.VarChar).Value =
dgvMList.Rows[i].Cells[1].Value.ToString();
                    cmd.ExecuteReader();
                    cn.Close();
                }
                MessageBox.Show("Recipt Added Successfully. \n" + "Recipt No: " +
(max).ToString());
                btnPrint.Enabled = true;
                btnPPV.Enabled = true;
                btnPSUp.Enabled = true;
                txtCName.Enabled = false;
                dgvMList.Enabled = false;
                btnCRecipt.Text = "New Receipt";
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
            finally
            {
            }
            #endregion

        }
    }
    else
    {
        funInitializedComp();
    }
}
private void btnPrint_Click(object sender, EventArgs e)
{
    pDoc.Print();
}
private void btnPPV_Click(object sender, EventArgs e)
{
    ppViewD.ShowDialog();
```

```
        }
        private void btnPSUp_Click(object sender, EventArgs e)
        {
            pSetUp.ShowDialog();
        }
        private void pDoc_PrintPage(object sender,
System.Drawing.Printing.PrintPageEventArgs e)
        {

            int pWidth = pSetUp.PageSettings.PaperSize.Width - 10;
            int pHeight = pSetUp.PageSettings.PaperSize.Height;
            #region Font
            Font fName = new System.Drawing.Font("Arial Black", 30, FontStyle.Bold);
            Font ffAdd = new System.Drawing.Font("Arial", 14, FontStyle.Bold);
            Font ffListH= new System.Drawing.Font("Arial", 11, FontStyle.Bold);
            #endregion

            #region header
            e.Graphics.DrawString("BAKE-WELL", fName, System.Drawing.Brushes.Blue,
(pWidth / 2) - 105, 25);
            e.Graphics.DrawString(fAdd, ffAdd, System.Drawing.Brushes.Black, pWidth -
fAdd.Length * 11, 80);
            e.Graphics.DrawString(mob, ffAdd, System.Drawing.Brushes.Black, pWidth -
mob.Length * 14, 100);
            e.Graphics.DrawLine(Pens.Blue, 10, 130, pWidth - 10, 130);
            #endregion
            #region recipt Details
            e.Graphics.DrawString("Recipt No: " + txtRNo.Text, ffAdd,
System.Drawing.Brushes.Black, 20, 155);
            e.Graphics.DrawString("Date: " + dtp.Text, ffAdd,
System.Drawing.Brushes.Black, pWidth - mob.Length * 14, 155);
            e.Graphics.DrawString("Customer Name:" + txtCName.Text, ffAdd,
System.Drawing.Brushes.Black, 20, 190);
            e.Graphics.DrawString("Menu List:", ffListH, System.Drawing.Brushes.Blue, 20,
245);
            #endregion
            #region Menu List
            Rectangle MHeader = new Rectangle(20, 285, pWidth - 40, 35);
            Rectangle SrNo = new Rectangle(20, 285, 80, 35);
            Rectangle IName = new Rectangle(100, 285, 250, 35);
            Rectangle Q = new Rectangle(350, 285, 100, 35);
            Rectangle uP = new Rectangle(450, 285, 100, 35);
            Rectangle u = new Rectangle(550, 285, 60, 35);
            Rectangle dis = new Rectangle(610, 285, 100, 35);
            Rectangle amt = new Rectangle(710, 285, 110, 35);
            e.Graphics.FillRectangle(Brushes.LightBlue, MHeader);

            Font fmHeader = new System.Drawing.Font("Arial", 11, FontStyle.Bold);
            StringFormat sf = new StringFormat();
            sf.Alignment = StringAlignment.Center;
            e.Graphics.DrawRectangle(Pens.Blue, SrNo);
            e.Graphics.DrawRectangle(Pens.Blue, IName);
            e.Graphics.DrawRectangle(Pens.Blue, Q);
            e.Graphics.DrawRectangle(Pens.Blue, u);
            e.Graphics.DrawRectangle(Pens.Blue, uP);
            e.Graphics.DrawRectangle(Pens.Blue, dis);
            e.Graphics.DrawRectangle(Pens.Blue, amt);

            e.Graphics.DrawString("Sr No", fmHeader, System.Drawing.Brushes.Blue,
SrNo, sf);
            e.Graphics.DrawString("Item Name", fmHeader, System.Drawing.Brushes.Blue,
IName, sf);
```

```
            e.Graphics.DrawString("Quantity", fmHeader, System.Drawing.Brushes.Blue,
Q, sf);
            e.Graphics.DrawString("Unit", fmHeader, System.Drawing.Brushes.Blue, u, sf);
            e.Graphics.DrawString("Rs/Unit", fmHeader, System.Drawing.Brushes.Blue, uP,
sf);
            e.Graphics.DrawString("Disc.(Rs)/Item", fmHeader,
System.Drawing.Brushes.Blue, dis, sf);
            e.Graphics.DrawString("Total AMT", fmHeader, System.Drawing.Brushes.Blue,
amt, sf);
            #endregion

            #region Menu List Item
            SrNo.Height = 30;
            IName.Height = 30;
            Q.Height = 30;
            uP.Height = 30;
            u.Height = 30;
            dis.Height = 30;
            amt.Height = 30;
            Font fitm = new System.Drawing.Font("Arial", 10, FontStyle.Regular);
            sf.LineAlignment = StringAlignment.Center;
            int y = 320;
            for (int i = 0; i < dgvMList.Rows.Count - 1; i++)
            {
                SrNo.Y = y;
                IName.Y = y;
                Q.Y = y;
                u.Y = y;
                uP.Y = y;
                dis.Y = y;
                amt.Y = y;
                e.Graphics.DrawRectangle(Pens.Black, SrNo);
                e.Graphics.DrawRectangle(Pens.Black, IName);
                e.Graphics.DrawRectangle(Pens.Black, Q);
                e.Graphics.DrawRectangle(Pens.Black, u);
                e.Graphics.DrawRectangle(Pens.Black, uP);
                e.Graphics.DrawRectangle(Pens.Black, dis);
                e.Graphics.DrawRectangle(Pens.Black, amt);

                e.Graphics.DrawString((i + 1).ToString(), fitm, Brushes.Black, SrNo, sf);
                e.Graphics.DrawString(dgvMList.Rows[i].Cells[1].Value.ToString(), fitm,
Brushes.Black, IName, sf);
                e.Graphics.DrawString(dgvMList.Rows[i].Cells[2].Value.ToString(), fitm,
Brushes.Black, Q, sf);
                e.Graphics.DrawString(dgvMList.Rows[i].Cells[3].Value.ToString(), fitm,
Brushes.Black, uP, sf);
                e.Graphics.DrawString(dgvMList.Rows[i].Cells[4].Value.ToString(), fitm,
Brushes.Black, u, sf);
                e.Graphics.DrawString(dgvMList.Rows[i].Cells[5].Value.ToString(), fitm,
Brushes.Black, dis, sf);
                e.Graphics.DrawString(dgvMList.Rows[i].Cells[6].Value.ToString(), fitm,
Brushes.Black, amt, sf);

                y += 30;
            }
            #endregion
            #region total Menu
            e.Graphics.DrawString("Total :", fmHeader, System.Drawing.Brushes.Blue, 20,
y + 15);
            y = y + 40;
            MHeader.Y = y;
            sf.LineAlignment = StringAlignment.Center;
```

```
            Rectangle TI = new Rectangle(20, y, 150, 35);
            Rectangle blank = new Rectangle(170, y, 430, 35);
            Rectangle TAMT = new Rectangle(710, y, 110, 35);
            e.Graphics.FillRectangle(Brushes.LightPink, MHeader);
            e.Graphics.DrawRectangle(Pens.Red, MHeader);
            e.Graphics.DrawString("Total Item", fmHeader, System.Drawing.Brushes.Blue,
TI, sf);
            e.Graphics.DrawString("", fmHeader, System.Drawing.Brushes.Blue, blank, sf);
            e.Graphics.DrawString("Total AMT", fmHeader, System.Drawing.Brushes.Blue,
TAMT, sf);
            sf.LineAlignment = StringAlignment.Center;
            y = y + 35;
            TI.Y = y;
            TAMT.Y = y;
            MHeader.Y = y;

            e.Graphics.FillRectangle(Brushes.LightBlue, MHeader);
            e.Graphics.DrawRectangle(Pens.Red, MHeader);
            e.Graphics.DrawString((dgvMList.Rows.Count - 1).ToString(), fmHeader,
System.Drawing.Brushes.Black, TI, sf);
            e.Graphics.DrawString(lblTAMT.Text, fmHeader, System.Drawing.Brushes.Black,
TAMT, sf);

            #endregion
            #region Footer
            e.Graphics.DrawString("Sign & Stamp Here", fmHeader,
System.Drawing.Brushes.Black, 600, y + 60);
            #endregion
        }



    }
}
```

ADVANTAGES:

- The system allows only authorized users to access the data.
- System is use of friendly and gives quick results.
- Data is stored in a systematic way in a data base.

- Reports are accurately generated on timely basis.
- Data can be retrieved efficiently as and when required by the user.
- Reports are accurately generated by selecting the appropriate identification code.
- Printing of report is possible.
- User has different access support.

LIMITATION OF PROPOSED SYSTEM:

- ✓ The limitation of the system is that it is centralized.

- ✓ Also the data to be stored may require data entry to be done.
- ✓ Production house and godown have not been maintained

Future Enhancement

1. We can provide the online facility.

2. We can maintain the payroll System.

3. We can maintain tax details.

4. We can maintain details of production and godown.

Bibliography

Black Book of C#.net
--Mathew Telles
C#.net
-Rajendra Salokhe

My SQL
-Ivan Bayross

www.sourcecode.com

www.java2s.com