

A Project Report
On
Project II: PRJ 251
DEPARTMENTAL STORE MANAGEMENT SYSTEM

Submitted by:

Debid Rana Magar	17530099
Dansi Ram Aacharya	17530098
Rajendra Lungeli	17530138
Khusi Ram Mahato	17530108

Under the Guidance
Of
Mr. Sunil Kumar

Submitted to the Faculty of Science and Technology,
OCEM
In Partial Fulfilment of the Requirements for

Fourth Semester Bachelor of Computer Application (BCA)
Year: 2018



OXFORD
(AFFILIATED TO POKHARA UNIVERSITY)
COLLEGE OF ENGINEERING & MANAGEMENT



Gaindakot-2, Nawalparasi
Nepal

CERTIFICATE OF AUTHENTICATED WORK

This is to certify that the project report entitled **Departmental Store Management System** submitted to **Department of Bachelor of Computer Application, Oxford College of Engineering and Management** in partial fulfilment of Fourth semester is an original work carried out by:

Debid Rana Magar	17530099
Dansi Ram Aacharya	17530098
Rajendra Lungeli	17530138
Khusi Ram Mahato	17530108

Under our guidance and supervision. The matter embodied in this project is authentic and genuine work done by the group and has not been submitted in other college/institution/University for the fulfilment of the requirement of any course of study.

.....
Mr. Sunil Kumar
Project Guide

.....
.....
External Evaluator

.....
Suresh Baral
HOD, BCA

ACKNOWLEDGEMENT

This project is developed in order to fulfil the partial requirement of **Pokhara University (PU)** for the completion of BCA 4th semester. Although this is the individual project during our study, we made it jointly in a group of four pupils.

We are very thankful to our teacher **Mr. Sunil Kumar** for his guidance and help in research and development. His valuable suggestions were very helpful in this project. For the development of each system, proper environment and platform is so inevitable. For this, we'll always remember the Golden name **Mr. Suresh Baral** who is our friendly HOD.

Suggesting new methods and techniques was really aid to us. As such, we wouldn't forget to highlight the name of **Dipendra Silwal** Sir for helpful guidance. And we would also like to thank our all team members for their hard work.

While creating new system, suggestion and ideas given by friends was so crucial to us. So, we will like to add those name in our list who had been engaged with us directly or indirectly.

ROLES AND RESPONSIBILITIES FORM

Departmental Store Management System

14/12/2018

Name of team Members	Roles	Task and Responsibility
Debid Rana Magar	Programmer	Coding, Testing , Debuging
Dansi Ram Acharya	Analyst	System Analysis, Testing
Rajendra Lungeli	Designer	Interface & Database Design
Khusi Ram Mahato	Documentation	Analysis & Documentation

Name of Project Team Members

Signature

1: Debid Rana Magar

1.....

2: Dansi Ram Acharya

2.....

3: Rajendra Lungeli

3.....

4: Khusi Ram Mahato

4.....

Signature of the Project Guide:

14/12/2018

Abstract/Executive Summary

This project is based on "**Department Store Management System**". Which is an attempt to computerize the existing flat file system. This project enables users to perform all the operation that is needed for department management system. According to this project users can make entry of the purchase and sales details, displaying their information, modifying their record, deleting their record as well as search their details and generate monthly, yearly sales report. Our software has facility to give a unique id for every purchase and sales transactions. It includes a search facility to know the current status of purchase and sales.

The Department Store Management System can be access by using a username and password. It is accessible only to the authorized person only. They can perform CRUD operation as per user requirements. Thus, this system highly secure with encryption. The data can be Backup and restore easily. The interface is very user-friendly.

Table of Contents

Chapter 1: INTRODUCTION.....	1
1.1 Introduction	1
1.2 Objectives.....	1
1.3 Scope and Limitations	1
Chapter 2: REQUIREMENT ANALYSIS AND FEASIBILITY STUDY	2
2.1 Related works:	2
2.2 Requirement Analysis.....	3
2.2.1 Functional requirements:	3
2.2.2 Nonfunctional Requirements	4
2.3 Feasibility Analysis.....	4
2.3.1 Economical Feasibility:	4
2.3.2 Operational Feasibility:.....	5
2.3.3 Technical Feasibility:.....	5
2.3.4 Schedule Feasibility:	5
Chapter 3: SYSTEM DESIGN	8
3.1 Process Modeling.....	8
3.1.1 Context Flow Diagram.....	8
3.1.2 Data Flow Diagram.....	9
3.2 Data Modeling	11
3.2.1 ER-Diagram	11
Chapter 4: CODING Of SYSTEM	13
Chapter 5: IMPLEMENTATION ANS TESTING	27
5.1 Implementation	27
5.1.1 Tools Used	27
5.1.2 Description/List of major classes/methods	28
5.2 Testing.....	28
5.2.1 Unit Testing	28
5.2.2 System Testing.....	35
Chapter 6: CONCLUSION AND RECOMMENDATION	41
6.1 Conclusion	41
6.2 Recommendations.....	41
References / Bibliography.....	42
References.....	42
Bibliography	42

Table of Figures

Figure 1 Functional use case diagram for DSM system	3
Figure 2 Pert Chart.....	6
Figure 3 Gantt Chart	7
Figure 4 Context Flow Diagram	8
Figure 5 DFD Level 1	9
Figure 6 DFD Level 2	9
Figure 7 DFD Level 3	10
Figure 8 DFD Level 4	10
Figure 9 ER Diagram	12
Figure 10 Login Form.....	28
Figure 11 Add User Form	29
Figure 12 View User.....	29
Figure 13 Categories	30
Figure 14 Add New Product	31
Figure 15 Product Stock.....	31
Figure 16 New Sales Entry	32
Figure 17 Sales Stock.....	32
Figure 18 Data Backup & Recovery	33
Figure 19 Product Report.....	33
Figure 20 Customer Bill.....	34

Chapter 1: INTRODUCTION

1.1 Introduction

“Departmental Store Management System” is useful for computerizing billing system. This software is useful for viewing customer information, calculating taxes, updating billing details, calculating other charges etc.

“Departmental Store Management System” is developed as per seeing the increasing requirement to speed up the work and incorporate a new work culture. Thus a new software has been proposed to reduce manual work, improving work efficiency, saving time.

1.2 Objectives

- To overcome the flat file system into a computerized system
- To record the purchases and sales transactions
- To reduce the transaction time
- To add and maintain records of available maintain
- To provide a convenient solution of billing pattern
- To add and maintain new entered category of products
- To add and maintain details of new products

1.3 Scope and Limitations

- System can be used in store keeping.
- System can be used for general purpose.
- System generates bill including vat 13%.
- Access control mechanism can be seen system.
- System isn't multiuser.
- It can't be connected to the internet.
- Data loss can be occur due to technical issues, so it is necessary to maintain backups at regular basis.

Chapter 2: REQUIREMENT ANALYSIS AND FEASIBILITY STUDY

Our project “**Departmental Store Management System**” is actually a dynamic system that is adaptable to the normal stores, shops and Trade companies as well. It is all about adding, reading, deleting and updating our products and their related categories & analyzing the business records/transactions.

2.1 Related works:

Before the start of any project, it is must to investigate the problem and define problem of the system. In order to develop any project, we have to know the problem. Once we investigate the problem, we need to find out the solution for it. So, we need a problem definition of the system. Mainly, it can be done in two ways:

- **System Analysis:** System analysis is the total analysis of the system. If we cannot make a proper analysis, then we can never develop the project as per our expectations. We have analysed the possible problems by taking some sample systems through online.
- **System Design:** It is arguably the most challenging and creative part of system development. It is all about giving proper shape and size to the program and it covers all the phases from data collection to validation

We have done following works to collect the best concepts in designing our system:

- Visited a couple of Stores around Gaidakot and Narayangarh
- Collected raw data about products and its categories.
- Collected samples of bill.
- Referenced the working concept of system via Web Articles.

2.2 Requirement Analysis

2.2.1 Functional requirements:

Functional requirements are the product features or functions that developers must implement to enable users to accomplish their tasks. Given below is a use case diagram for reflecting.

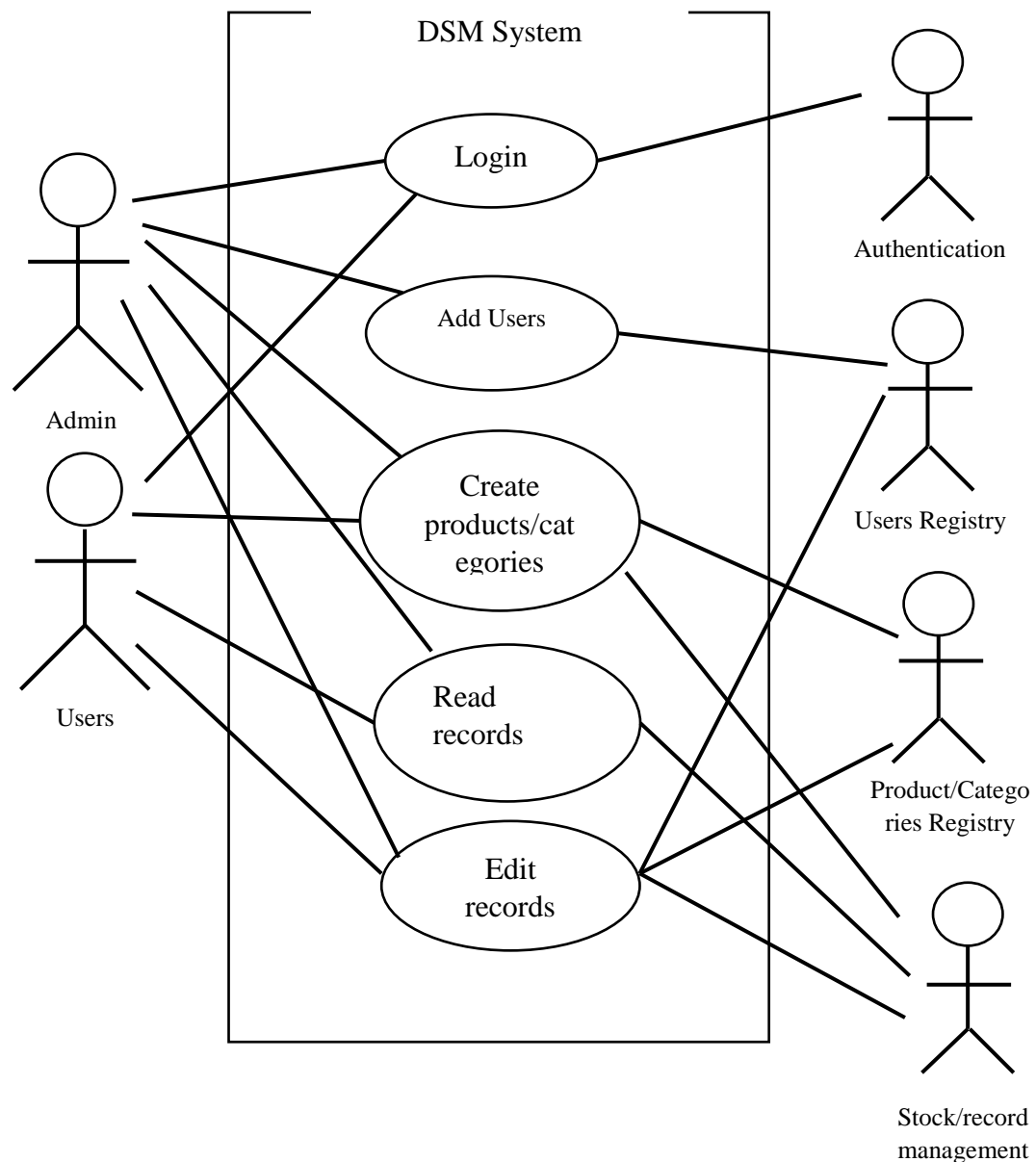


Figure 1 Functional use case diagram for DSM system

2.2.2 Nonfunctional Requirements

Non-functional requirements are those which describe the general characteristics of a system. They are also known as quality attributes. Thus, the program will still run even if these requirements are not fulfilled. Given below are a few of such requirements:

- Provision for access control mechanism,
- Service of backup and restore of data to maintain security,
- System help/guidance at runtime,
- Resetting the passwords for security purpose,
- Online access of the application.

Hardware and Software Requirements

- Hardware Requirements:
 - RAM: 2GB or above
 - Hard Disk: 256GB or above
 - Processor: i3 or above
- Software Requirements:
 - Operating system: Windows 7/8/10
 - Server : Microsoft SQL Server 2012 and later
 - Framework : .NET Framework 4.6.1 and later

2.3 Feasibility Analysis

The feasibility study defines whether the system is feasible/acceptable or not to the various fields or factors. Our system is feasible to the following fields/factors:

2.3.1 Economical Feasibility:

Economical analysis is the most frequently used method for evaluating the effectiveness of the new system. Our system is economically feasible due to the following streams of reason:

- It is low in the fees and is affordable to each mid-scale stores.
- It does not possess extra charges.

2.3.2 Operational Feasibility:

Our system is operationally feasible as it solves the problems, and takes advantage of the opportunities identified during scope definition. It is smooth in the operation and takes no more time to store our data to database and access them immediately.

Thus, our system is operationally feasible.

2.3.3 Technical Feasibility:

To implement our software 'DST System', a server which can be connected in any Standalone computer is required. So, the technical requirements for this software aren't that complicated, which makes it technically feasible.

2.3.4 Schedule Feasibility:

It is defined as the likelihood of a project being completed within its scheduled time frame. Given below is a Gantt chart and a PERT chart to show the time plan for the project assigned to us and the actual completion time respectively:-

Pert Chart

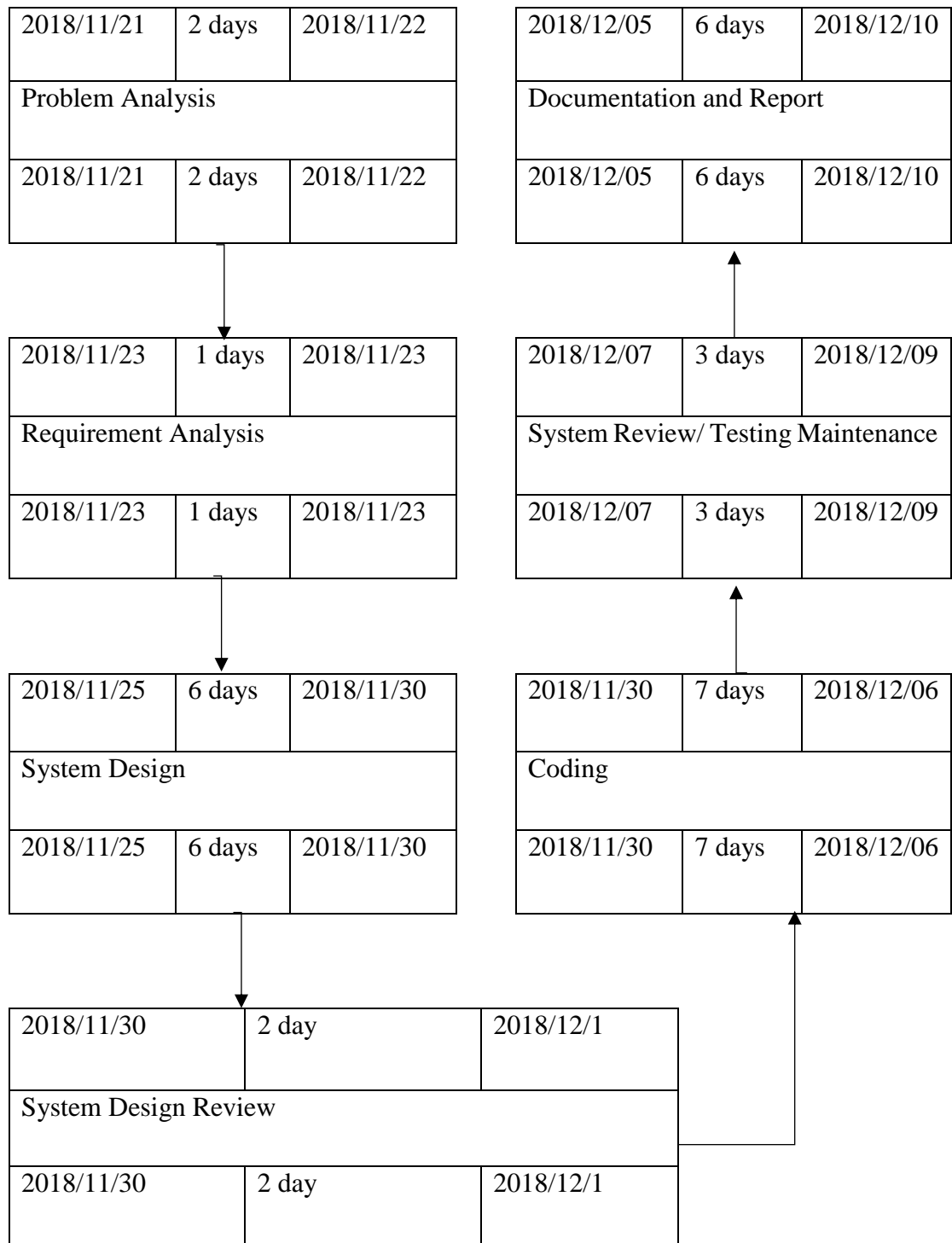


Figure 2 Pert Chart

Gantt Chart

Project Activity	Nov 21 -22	Nov 23 - 23	Nov 25 - 30	Nov 30 – Dec 1	Nov/D ec 30 - 06	Dec 07- 09	Dec 05 - 10
Problem Analysis							
Requirement Analysis							
System Design							
System Design Review							
Coding							
System testing and debugging							
Documentation and Report							

Figure 3 Gantt Chart

Chapter 3: SYSTEM DESIGN

3.1 Process Modeling

3.1.1 Context Flow Diagram

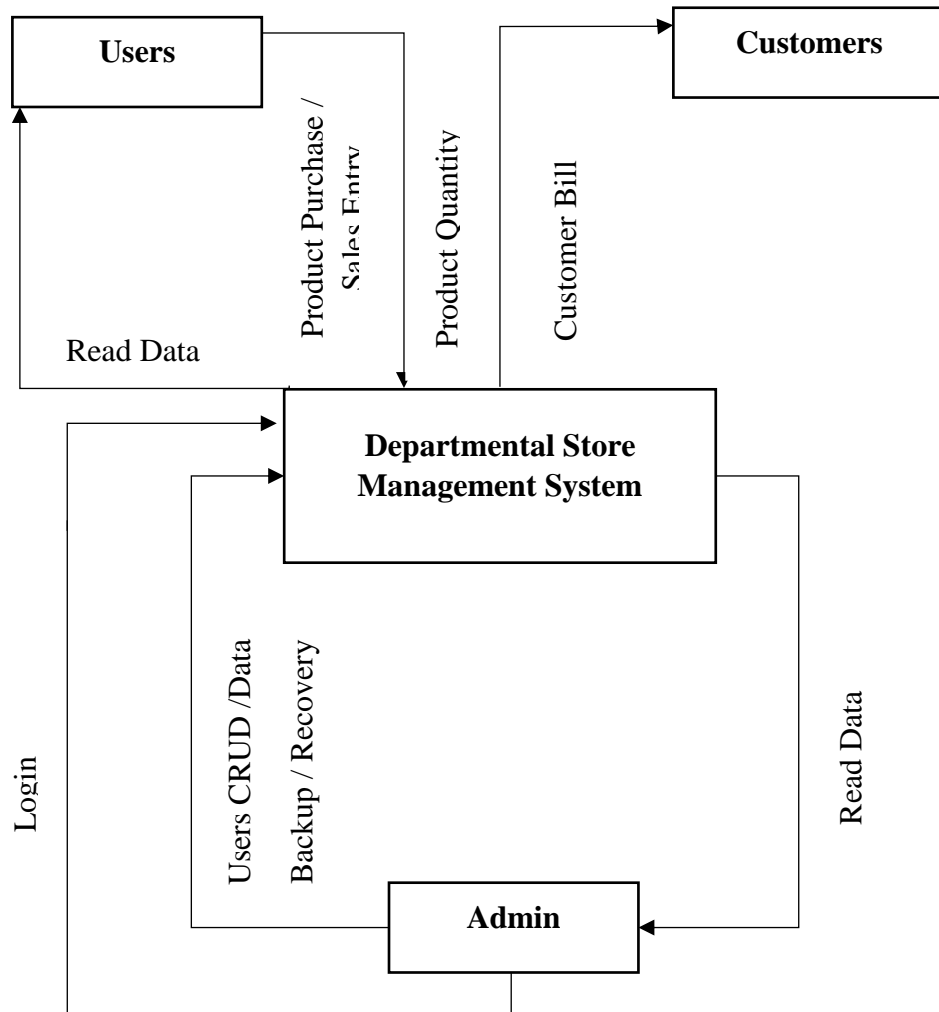


Figure 4 Context Flow Diagram

3.1.2 Data Flow Diagram

Level 1

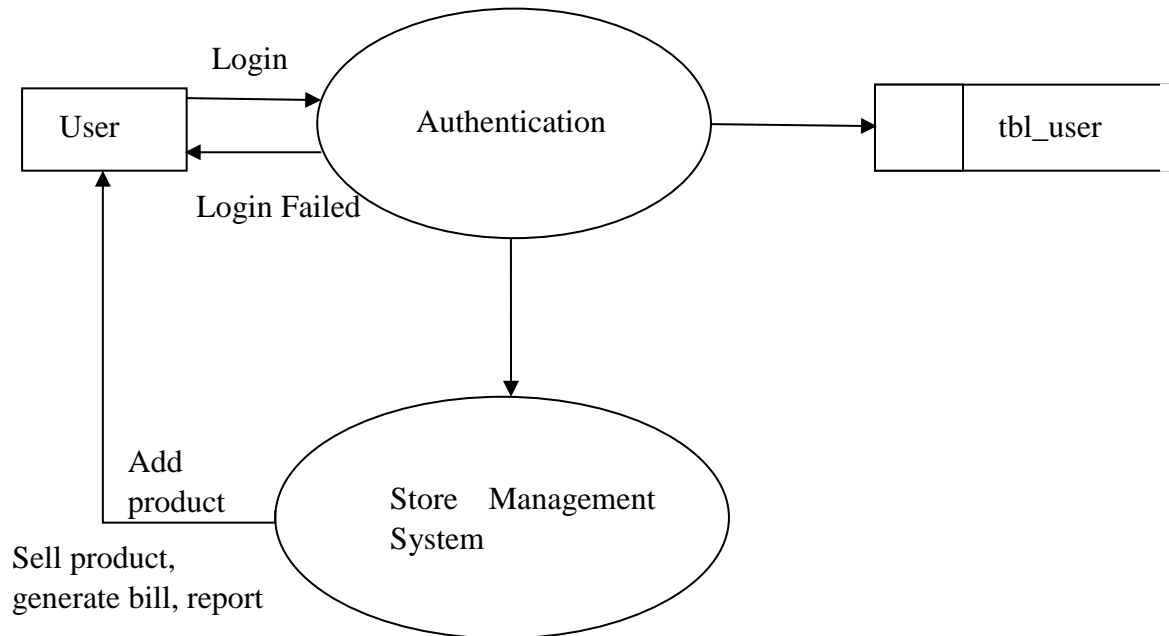


Figure 5 DFD Level 1

Level 2

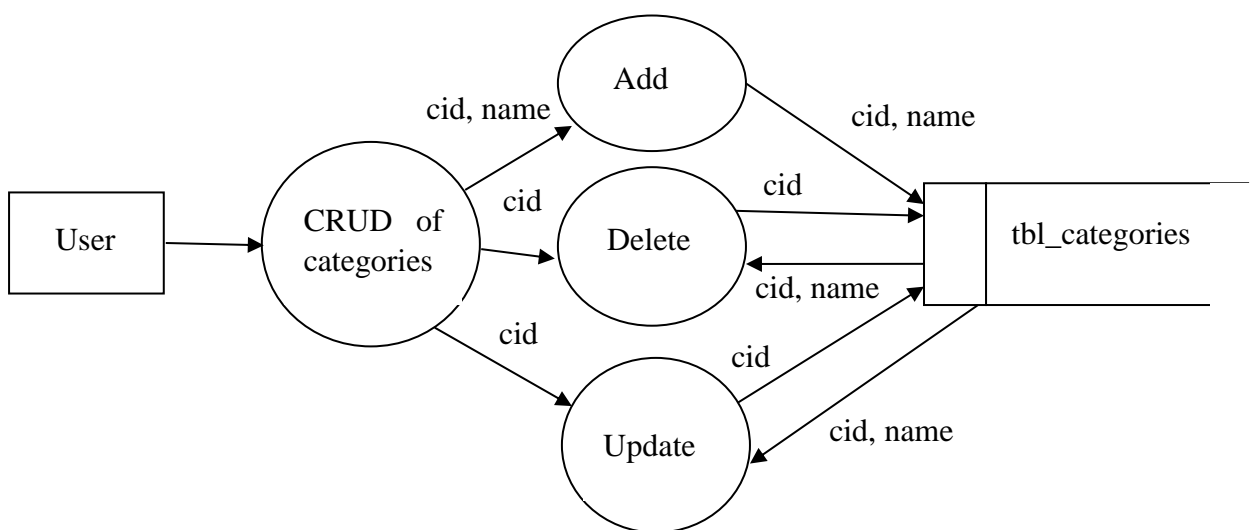


Figure 6 DFD Level 2

Level 3

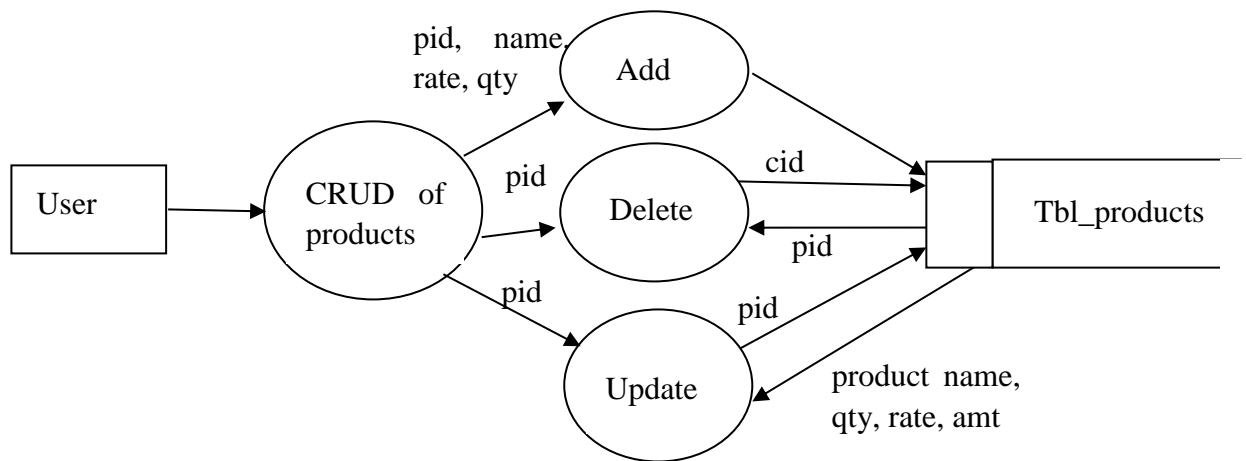


Figure 7 DFD Level 3

Level 4

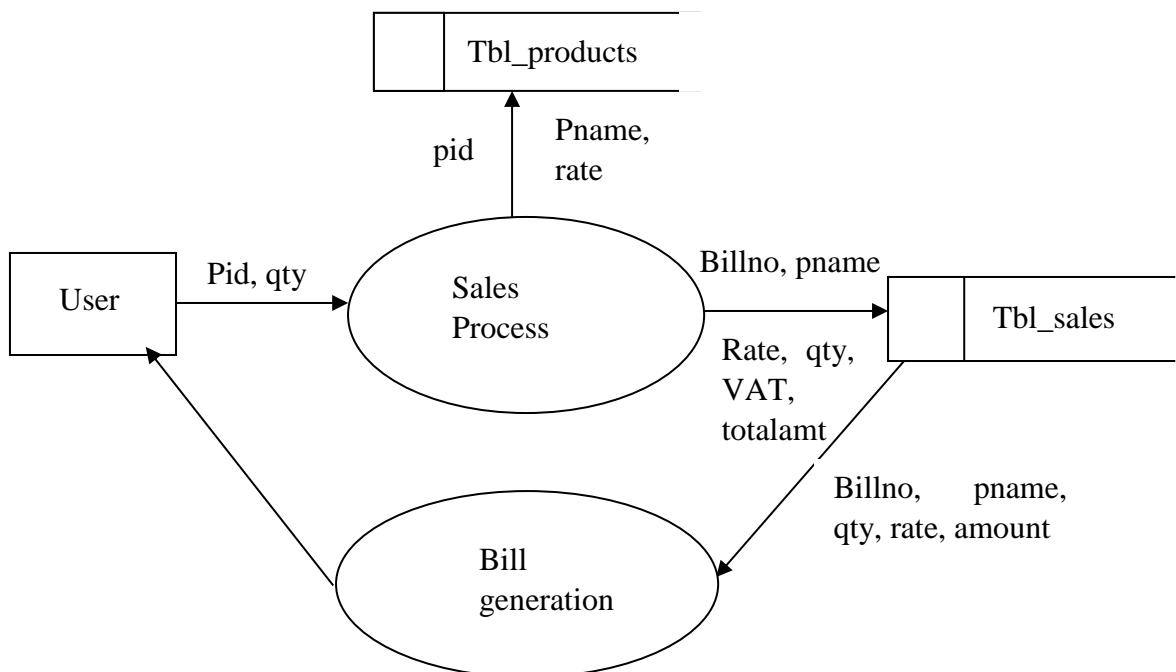
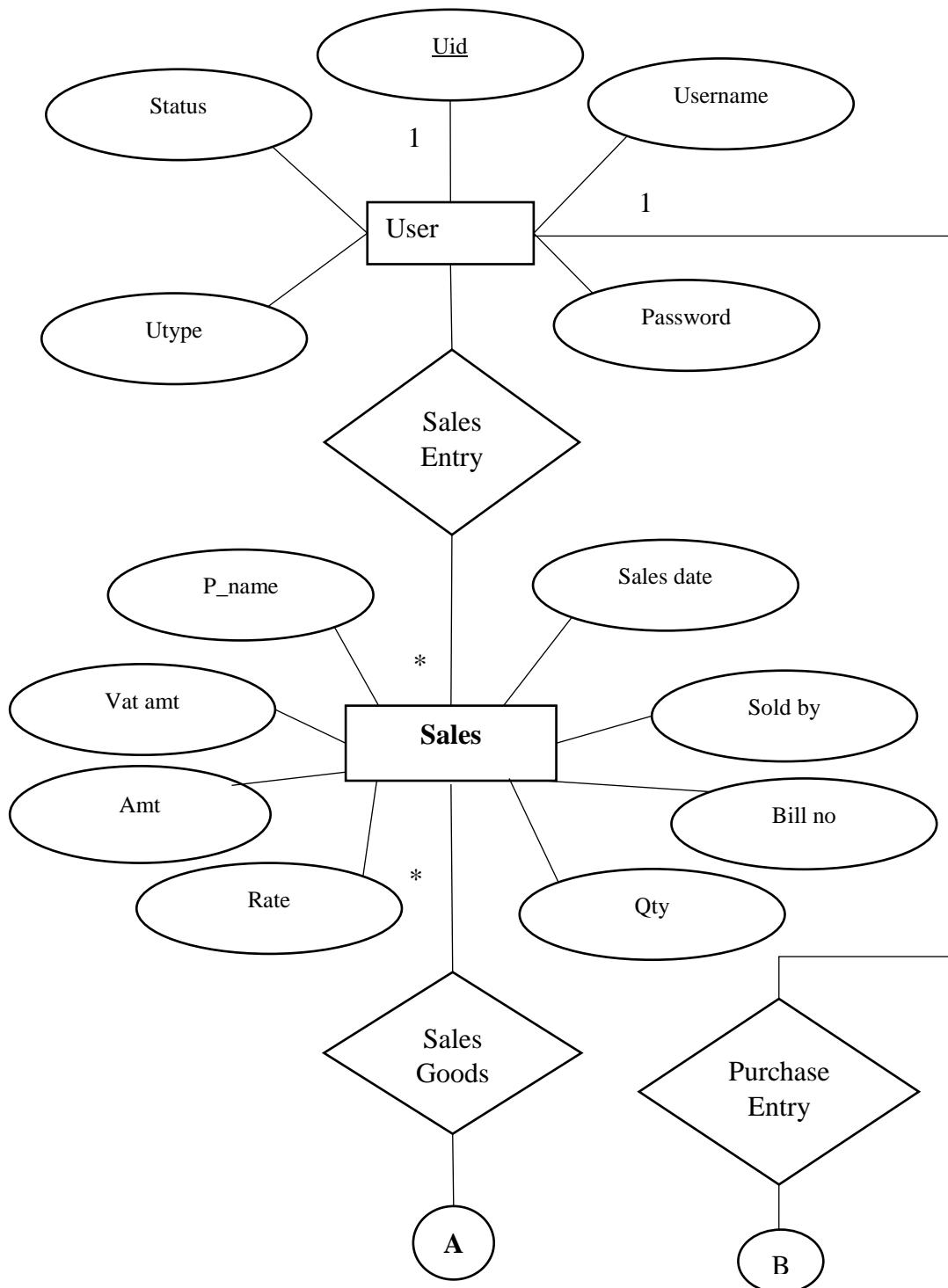


Figure 8 DFD Level 4

3.2 Data Modeling

3.2.1 ER-Diagram



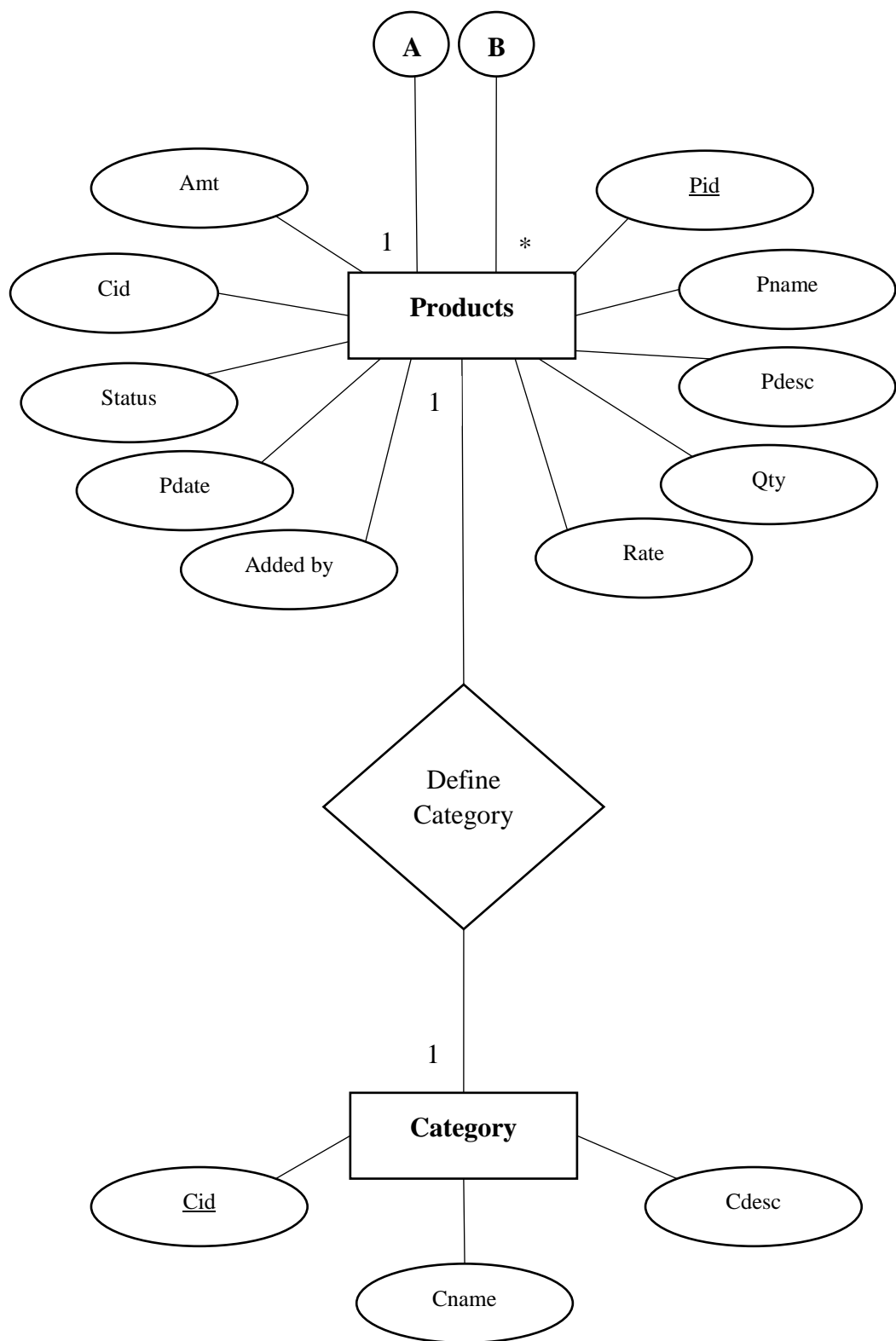


Figure 9 ER Diagram

Chapter 4: CODING Of SYSTEM

Module 1. Database Connection

```
Imports System.Data.SqlClient

Module conxnmodule
    Public cn As SqlConnection
    Public Sub con()
        Dim strConnection As String = " Data Source=DM;Initial
Catalog=db_dsms;Integrated Security=True"
        cn = New SqlConnection(strConnection)
        cn.Open()
    End Sub
End Module
```

Module 2. Login

```
Imports System.Data.SqlClient
Public Class login
    Public lvl, sts, username As String
    Private Sub login_Load(sender As Object, e As EventArgs) Handles
MyBase.Load
        conxnmodule.con()
    End Sub
    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
        cn.Close()
        cn.Open()

        Dim str As String
        If (TextBox1.Text.Length = 0 Or TextBox2.Text.Length = 0) Then
            MsgBox("check the fields, no one can be blank")
        Else
            str = "select * from users where username='" &
TextBox1.Text & "' and password='" & TextBox2.Text & "'"
            Dim cmd As New SqlCommand(str, conxnmodule.cn)
            Dim rdr As SqlDataReader
            rdr = cmd.ExecuteReader()
            If (rdr.Read = True) Then
                lvl = rdr("utype")
                username = rdr("username")
                sts = rdr("status")
                If sts = 0 Then
                    MsgBox("sorry! your account is inactive",
MsgBoxStyle.Critical, "Error")
                Else
                    MsgBox("login success", MsgBoxStyle.OkOnly,
"Success")
                    main.Show()
                    Me.Hide()
                    rdr.Close()
                End If
            Else

```

```

        MsgBox("sorry! login name or password is incorrect",
MsgBoxStyle.Critical, "Error")
    End If
End If
End Sub
End Class

```

Module 3. Add User

```

Imports System.Data.SqlClient
Public Class addnewuser
    Public Sub clear()
        TextBox1.Clear()
        TextBox2.Clear()
    End Sub
    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
        Try
            If (TextBox1.Text.Length = 0 Or TextBox2.Text.Length = 0)
Then
                MsgBox("check the fields, no one can be blank")
            Else
                Dim sts As Integer
                If ComboBox2.SelectedItem = "Active" Then
                    sts = 1
                Else
                    sts = 0
                End If
                Dim sqlcmd As New SqlCommand()
                sqlcmd.Connection = conxnmodule.cn
                Dim str As String
                str = "INSERT INTO
users(username,password,utype,status) values ('" & Me.TextBox1.Text &
"', '" & Me.TextBox2.Text & "', '" & Me.ComboBox1.SelectedItem & "', '"
& sts & "')"
                sqlcmd.CommandText = str
                sqlcmd.ExecuteNonQuery()

                MsgBox("Record Inserted Successfully")
                Me.Close()
            End If
        Catch ex As Exception
            MessageBox.Show(ex.Message)
        End Try
    End Sub

    Private Sub Button3_Click(sender As Object, e As EventArgs) Handles
Button3.Click
        Me.Close()
    End Sub

    Private Sub Button2_Click(sender As Object, e As EventArgs) Handles
Button2.Click

```

```

        Call clear()
    End Sub

End Class

```

Module 4. View User

```

Imports System.Data.SqlClient
Public Class viewusers
    Public Sub data()
        DataGridView1.Rows.Clear()
        Dim rdr As SqlDataReader
        Dim n As Integer
        Dim str As String = "select * from users"
        Dim cmd As New SqlCommand(str, conxnmodule.cn)
        rdr = cmd.ExecuteReader
        While rdr.Read()
            n = DataGridView1.Rows.Add()
            DataGridView1.Rows.Item(n).Cells(0).Value = rdr("uid")
            DataGridView1.Rows.Item(n).Cells(1).Value = rdr("username")
            DataGridView1.Rows.Item(n).Cells(2).Value = rdr("password")
            DataGridView1.Rows.Item(n).Cells(3).Value = rdr("utype")
            DataGridView1.Rows.Item(n).Cells(4).Value = rdr("status")

        End While
        rdr.Close()
    End Sub
    Private Sub viewusers_Load(sender As Object, e As EventArgs)
        Handles MyBase.Load
        conxnmodule.con()
        Call data()
    End Sub
    Private Sub DataGridView1_CellDoubleClick(sender As Object, e As
DataGridViewCellEventArgs) Handles DataGridView1.CellDoubleClick
        Dim sts As Integer
        TextBox1.Text = DataGridView1.CurrentRow.Cells.Item(0).Value
        TextBox2.Text = DataGridView1.CurrentRow.Cells.Item(1).Value
        TextBox3.Text = DataGridView1.CurrentRow.Cells.Item(2).Value

        sts = DataGridView1.CurrentRow.Cells.Item(4).Value
        If sts = 1 Then
            TextBox4.Text = "Active"
        Else
            TextBox4.Text = "Inactive"
        End If
    End Sub
    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
        cn.Close()
        cn.Open()
        Dim sts As Integer
        If TextBox4.Text = "Active" Then
            sts = 1
        Else

```

```

        sts = 0
    End If
    If (TextBox1.Text.Length = 0) Then
        MsgBox("please select category first for edit",
MsgBoxStyle.Critical, "Error")
    Else
        Dim cmd1 As New SqlCommand
        cmd1.Connection = cn
        cmd1.CommandText = "UPDATE users set username='" &
Trim(TextBox2.Text) & "',password='" & Trim(TextBox3.Text) &
"',utype='" & ComboBox1.SelectedItem & "',status='" & sts & "' where
uid='" & TextBox1.Text & "'"
        cmd1.ExecuteNonQuery()
        MsgBox("Users updated sucessfully")
        TextBox1.Clear()
        TextBox2.Clear()
        TextBox3.Clear()
        Call data()
    End If
End Sub

Private Sub Button2_Click(sender As Object, e As EventArgs) Handles
Button2.Click
    If (TextBox1.Text.Length = 0) Then
        MsgBox("Check the fields, no one can be blank! please
select item first", MsgBoxStyle.Critical, "Error")
    Else
        Dim res As Integer

        res = MsgBox("Do you want delete it",
MsgBoxStyle.OkCancel, "Confirmation")
        If (res = 1) Then

            cn.Close()
            cn.Open()
            Dim cmd1 As New SqlCommand
            cmd1.Connection = cn
            cmd1.CommandText = "delete from users where uid='" &
TextBox1.Text & "'"
            cmd1.ExecuteNonQuery()
            MsgBox("record is deleted",
MsgBoxStyle.DefaultButton1, "Success")
            TextBox1.Clear()
            TextBox2.Clear()
            TextBox3.Clear()
            TextBox4.Clear()

            DataGridView1.Rows.Clear()
            Call data()
        End If
    End If
End Sub

Private Sub Button3_Click(sender As Object, e As EventArgs) Handles
Button3.Click
    Me.Close()
End Sub

Private Sub ComboBox1_SelectedIndexChanged(sender As Object, e As
EventArgs) Handles ComboBox1.SelectedIndexChanged

```

```

End Sub
End Class

```

Module 5. Add New Product

```

Imports System.Data.SqlClient
Public Class addproduct
    Dim rate, qty, amt As Integer
    Public cid As Integer
    Public Sub addcat()
        conxnmodule.con()

        Dim cn As SqlConnection = conxnmodule.cn
        Dim rdr As SqlDataReader
        Try
            cn.Close()
            cn.Open()
            Dim sqlString As String
            sqlString = "SELECT * From categories"
            Dim cmd As New SqlCommand(sqlString, cn)
            rdr = cmd.ExecuteReader()
            While rdr.Read
                Dim uname = rdr.GetString(rdr.GetOrdinal("cname"))
                ComboBox1.Items.Add(uname)
            End While
            cn.Close()
        Catch ex As Exception
            MessageBox.Show(ex.Message)
        End Try
    End Sub

    Private Sub purchchase_form_Load(sender As Object, e As EventArgs)
Handles MyBase.Load
        Call addcat()
    End Sub

    Private Sub Button5_Click(sender As Object, e As EventArgs) Handles
Button5.Click
        TextBox1.Clear()
        TextBox2.Clear()
        TextBox3.Clear()
        TextBox4.Clear()

    End Sub

    Private Sub Button6_Click(sender As Object, e As EventArgs) Handles
Button6.Click
        Me.Close()
    End Sub

    Private Sub ComboBox1_SelectedIndexChanged(sender As Object, e As
EventArgs) Handles ComboBox1.SelectedIndexChanged
        Dim cn As SqlConnection = conxnmodule.cn
        Dim rdr As SqlDataReader
        Try
            cn.Close()

```



```

        cn.Open()
        Dim sqlString As String
        sqlString = "SELECT * From categories where cname='" &
ComboBox1.SelectedItem & "'"
        Dim cmd As New SqlCommand(sqlString, cn)
        rdr = cmd.ExecuteReader()
        If (rdr.Read = True) Then
            cid = rdr("cid")
        End If
        cn.Close()
    Catch ex As Exception
        MessageBox.Show(ex.Message)
    End Try
End Sub

Private Sub Button4_Click(sender As Object, e As EventArgs) Handles
Button4.Click
    Try
        If (TextBox1.Text.Length = 0 Or TextBox2.Text.Length = 0)
Then
            MsgBox("check the fields, no one can be blank")

        Else

            rate = TextBox4.Text
            qty = TextBox3.Text
            amt = rate * qty
            cn.Close()
            cn.Open()
            Dim sqlcmd As New SqlCommand()
            sqlcmd.Connection = conxnmodule.cn
            Dim str As String
            str = "INSERT INTO
products(pname,pdesc,qty,rate,amt,cid,addedby) values ('" &
Me.TextBox1.Text & "','" & Me.TextBox2.Text & "','" & qty & "','" &
rate & "','" & amt & "','" & cid & "','" & login.username & "')"
            sqlcmd.CommandText = str
            sqlcmd.ExecuteNonQuery()
            MsgBox("Record Inserted Successfully")
            Me.Close()
        End If
    Catch ex As Exception
        MessageBox.Show(ex.Message)
    End Try
End Sub

End Class

```

Module 6. New Sales Entry

```

Imports System.Data.SqlClient
Public Class sales
    Public billno As Integer
    Public Sub data()
        DataGridView1.Rows.Clear()
        conxnmodule.con()
    Try
        'Dim cmd As SqlCommand

```

```

        Dim rdr As SqlDataReader
        Dim n As Int16
        Dim str As String = "SELECT
pid,pname,pdesc,cname,qty,rate,addedby from products join categories
on products.cid = categories.cid"
        Dim cmd As New SqlCommand(str, conxnmodule.cn)
        rdr = cmd.ExecuteReader

        While rdr.Read()
            n = DataGridView1.Rows.Add()
            DataGridView1.Rows.Item(n).Cells(0).Value =
rdr("pid")
            DataGridView1.Rows.Item(n).Cells(1).Value =
rdr("pname")
            DataGridView1.Rows.Item(n).Cells(2).Value =
rdr("pdesc")
            DataGridView1.Rows.Item(n).Cells(3).Value =
rdr("cname")
            DataGridView1.Rows.Item(n).Cells(4).Value =
rdr("qty")
            DataGridView1.Rows.Item(n).Cells(5).Value =
rdr("rate")
            DataGridView1.Rows.Item(n).Cells(6).Value =
rdr("addedby")
        End While
        rdr.Close()
        Catch ext As Exception
            MsgBox(ext.Message)
        End Try
    End Sub

    Private Sub sales_Load(sender As Object, e As EventArgs) Handles
MyBase.Load
        conxnmodule.cn()
        Dim r As SqlDataReader
        Dim st As String = "SELECT TOP 1 * from sales ORDER BY billno
DESC"
        Dim cm As New SqlCommand(st, conxnmodule.cn)
        r = cm.ExecuteReader()
        If r.Read = True Then
            TextBox1.Text = r("billno") + 1
        Else
            TextBox1.Text = 1
        End If

        r.Close()
        Call data()
        Label5.Visible = False

        Dim cn As SqlConnection = conxnmodule.cn
        Dim rdr As SqlDataReader
        Try
            cn.Close()
            cn.Open()
            Dim sqlString As String
            sqlString = "SELECT * From categories"
            Dim cmd As New SqlCommand(sqlString, cn)
            rdr = cmd.ExecuteReader()
            While rdr.Read
                Dim uname = rdr.GetString(rdr.GetOrdinal("cname"))
                ComboBox1.Items.Add(uname)
            End While
        Catch
        End Try
    End Sub

```

```

        End While

        cn.Close()
    Catch ex As Exception
        MessageBox.Show(ex.Message)
    End Try
End Sub

Private Sub ComboBox1_SelectedIndexChanged(sender As Object, e As
EventArgs) Handles ComboBox1.SelectedIndexChanged
    ComboBox3.Items.Clear()
    Dim cid As Integer
    Dim cn As SqlConnection = conxnmodule.cn
    Dim rdr As SqlDataReader
    Try
        cn.Close()
        cn.Open()
        Dim sqlString As String
        sqlString = "SELECT * From categories where cname='" &
ComboBox1.SelectedItem & "'"
        Dim cmd As New SqlCommand(sqlString, cn)
        rdr = cmd.ExecuteReader()
        If (rdr.Read = True) Then
            cid = rdr("cid")
        End If
        cn.Close()
    Catch ex As Exception
        MessageBox.Show(ex.Message)
    End Try
    Label5.Visible = False
    Dim rdr1 As SqlDataReader
    Try

        cn.Close()
        cn.Open()
        Dim sqlString As String
        sqlString = "SELECT * From products where cid='" & cid &
"""
        Dim cmd As New SqlCommand(sqlString, cn)
        rdr1 = cmd.ExecuteReader()
        If (rdr1.HasRows = True) Then
            While rdr1.Read

                ComboBox3.Items.Add(rdr1("pid"))
            End While
        Else
            Label5.Visible = True
        End If
        cn.Close()
    Catch ex As Exception
        MessageBox.Show(ex.Message)
    End Try
End Sub

Public rate, qty, lamt, amt, uamt, discount, sqty, vat, pid As
Integer
Public pname As String

Private Sub Button4_Click(sender As Object, e As EventArgs) Handles
Button4.Click
    billno = TextBox1.Text

```

```

If (TextBox2.Text.Length = 0 Or TextBox3.Text.Length = 0) Then
    MsgBox("check the fields, no one can be blank")

Else
    pname = TextBox5.Text
    rate = TextBox3.Text
    qty = TextBox2.Text
    If (TextBox4.Text.Length = 0) Then
        discount = 0
    Else
        discount = TextBox4.Text
    End If
    lamt = rate * qty
    vat = lamt * 13 / 100
    amt = lamt
    amt -= discount
    amt += vat
    ' MsgBox(amt)
    Dim rdr As SqlDataReader
    Try
        cn.Close()
        cn.Open()
        Dim sqlString As String
        sqlString = "SELECT * From products where pname='" &
TextBox5.Text & "'"
        Dim cmd As New SqlCommand(sqlString, cn)
        rdr = cmd.ExecuteReader()
        If (rdr.Read = True) Then
            sqty = rdr("qty")
            pid = rdr("pid")
        End If
        cn.Close()
    Catch ex As Exception
        MessageBox.Show(ex.Message)
    End Try

    If qty <= sqty Then

        cn.Close()
        cn.Open()
        Dim sqlcmd, sqlcmd1 As New SqlCommand()
        sqlcmd.Connection = conxnmodule.cn
        sqlcmd1.Connection = conxnmodule.cn
        Dim Str, Str1 As String
        Str = "INSERT INTO
sales(billno,pname,cname,qty,rate,amt,discount,vatamt,salesby) values
('" & TextBox1.Text & "','" & Me.TextBox5.Text & "','" &
Me.ComboBox1.SelectedItem & "','" & qty & "','" & rate & "','" & amt
& "','" & discount & "','" & vat & "','" & login.username & "')"
        sqty = sqty - qty
        uamt = sqty * rate
        Str1 = "UPDATE products set qty='" & sqty & "',amt='"
& uamt & "'" where pid='" & pid & "'"
        sqlcmd.CommandText = Str
        sqlcmd1.CommandText = Str1
        sqlcmd.ExecuteNonQuery()
        sqlcmd1.ExecuteNonQuery()
        MsgBox("Sales Entry Inserted Successfully")

        Dim rs As Integer

```

```

        rs = MsgBox("Do you want entry again",
MsgBoxStyle.YesNo, "Confirmation")
        If rs = DialogResult.Yes Then

            Else
                TextBox1.Clear()
                generatingbill.Show()
            End If

        Else
            MsgBox("Selected product is out of stock")

        End If
    End If
End Sub

Private Sub ComboBox3_SelectedIndexChanged(sender As Object, e As
EventArgs) Handles ComboBox3.SelectedIndexChanged
    Dim rdr As SqlDataReader
    Try

        TextBox3.Clear()
        TextBox5.Clear()
        cn.Close()
        cn.Open()
        Dim sqlString As String
        sqlString = "SELECT * From products where pid='" &
ComboBox3.SelectedItem & "'"
        Dim cmd As New SqlCommand(sqlString, cn)
        rdr = cmd.ExecuteReader()
        If (rdr.Read = True) Then

            TextBox5.Text = rdr("pname")
            TextBox3.Text = rdr("rate")
        End If
        cn.Close()
    Catch ex As Exception
        MessageBox.Show(ex.Message)
    End Try
End Sub

End Class

```

Module 7. Backup and recovery

```

Imports System.Data.SqlClient
Public Class BackupRestore

    Private Sub BackupRestore_Load(sender As Object, e As EventArgs)
Handles MyBase.Load
        conxnmodule.con()

    End Sub

    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
        forbackup.connect()
    End Sub

```

```

        forbackup.cmd1 = New SqlCommand("backup database db_dsms to
disk='D:\\College Project\\dbbackup.bak'", forbackup.con1)
        forbackup.cmd1.ExecuteNonQuery()
        MsgBox("Data is backpuped")
    End Sub

    Private Sub Button2_Click(sender As Object, e As EventArgs) Handles
Button2.Click
        forbackup.connect()
        forbackup.cmd1 = New SqlCommand("restore database db_dsms from
disk='D:\\College Project\\dbbackup.bak' with replace",
forbackup.con1)
        forbackup.cmd1.ExecuteNonQuery()
        MsgBox("Database is stored")
    End Sub

End Class

```

Module 8. Billing

```

Imports System.Data.SqlClient
Public Class biling
    Public tamt As Integer
    Dim lamt As Integer
    Dim str As String
    Dim k As Integer
    Dim A(30) As Integer
    Dim b As Boolean
    Private Sub biling_Load(sender As Object, e As EventArgs) Handles
MyBase.Load
        Label8.Text = "Date: " & Date.Today
        Label7.Text = sales.billno
        DataGridView1.Rows.Clear()
        conxnmodule.con()
        Try
            Dim cmd As SqlCommand
            Dim rdr As SqlDataReader
            Dim n As Int16
            Dim str As String = "SELECT * from sales where billno='"
& sales.billno & "'"
            Dim str As String = "SELECT * from sales where
billno='1'"
            Dim cmd As New SqlCommand(str, conxnmodule.cn)
            rdr = cmd.ExecuteReader

            While rdr.Read()

                n = DataGridView1.Rows.Add()
                DataGridView1.Rows.Item(n).Cells(0).Value = n + 1
                DataGridView1.Rows.Item(n).Cells(1).Value =
rdr("pname")
                DataGridView1.Rows.Item(n).Cells(2).Value =
rdr("qty")
                DataGridView1.Rows.Item(n).Cells(3).Value =
rdr("rate")
                DataGridView1.Rows.Item(n).Cells(4).Value =
DataGridView1.Rows.Item(n).Cells(2).Value *
DataGridView1.Rows.Item(n).Cells(3).Value
            End While
        Catch ex As Exception
            MsgBox(ex.Message)
        End Try
    End Sub
End Class

```

```

                                lamt = lamt +
DataGridView1.Rows.Item(n).Cells(4).Value
End While
rdr.Close()

Label25.Text = lamt

Dim str1 As String = "SELECT sum(vatamt) from sales where
billno='" & sales.billno & "'"
Dim cmd1 As New SqlCommand(str1, conxnmodule.cn)
Label26.Text = cmd1.ExecuteScalar().ToString
Dim str2 As String = "SELECT sum(amt) from sales where
billno='" & sales.billno & "'"
Dim cmd2 As New SqlCommand(str2, conxnmodule.cn)
tamt = cmd2.ExecuteScalar().ToString
Label27.Text = tamt

Label9.Text = login.username
Catch ext As Exception
MsgBox(ext.Message)
End Try

'Dim nn As Integer
'TextBox1.Text = Val(nn)
Dim curr As String = " "
Dim l As Integer
Dim num As Long
Dim i As Integer
b = False

ResetArray()
str = tamt & "."
str = Mid(tamt, 1, InStr(str, ".") - 1)
l = Len(str)
num = Val(str)
For i = 0 To l
    A(i) = num Mod 10
    num = num \ 10
Next

For i = 1 To 0 Step -1
    k = i
    Select Case i
        Case 10 : curr = curr & Convert1(A(i - 1), A(i),
"Billion ")
        Case 9 : curr = curr & Convert1(A(i - 1), , "Hundred")
        Case 8 : curr = curr & Convert3(A(i - 1))
        Case 7 : curr = curr & Convert1(A(i - 1), A(i), "
Million ")
        Case 6 : curr = curr & Convert1(A(i - 1), , " Hundred
")
        Case 5 : curr = curr & Convert3(A(i - 1))
        Case 4 : curr = curr & Convert1(A(i - 1), A(i), "
Thousand ")
        Case 3 : curr = curr & Convert1(A(i - 1), , " Hundred
")
        Case 2 : curr = curr & Convert3(A(i - 1))
        Case 1 : curr = curr & Convert1(A(i - 1), A(i))
    End Select

```

```

Next
If curr = " " Then curr = "Zero"
Label10.Text = curr + " Only"

End Sub

Private Function Convert2(ByVal num As Integer, Optional ByVal
Digit As String = "") As String

    Dim curRRR As String = " "
    Select Case num
        Case 9 : curRRR = "Nineteen"
        Case 8 : curRRR = "Eighteen"
        Case 7 : curRRR = "Seventeen"
        Case 6 : curRRR = "Sixteen"
        Case 5 : curRRR = "Fifteen"
        Case 4 : curRRR = "Fourteen"
        Case 3 : curRRR = "Thirteen"
        Case 2 : curRRR = "Twelve"
        Case 1 : curRRR = "Eleven"
        Case 0 : curRRR = "Ten"
    End Select

    If k = 7 Then b = True
    If num = 0 Then
        If k = 7 Then GoTo c
        If b = False And k = 4 Then GoTo c
        Digit = ""
    End If

c:
    Convert2 = curRRR & Digit

End Function

Private Function Convert1(ByVal num As Integer, Optional ByVal
Nexttime As Integer = 0, Optional ByVal Digit As String = "") As String

    Dim curR As String = ""
    If Nexttime = 1 Then
        Convert1 = Convert2(num, Digit)
        Exit Function
    End If
    Select Case num
        Case 9 : curR = "Nine"
        Case 8 : curR = "Eight"
        Case 7 : curR = "Seven"
        Case 6 : curR = "Six"
        Case 5 : curR = "Five"
        Case 4 : curR = "Four"
        Case 3 : curR = "Three"
        Case 2 : curR = "Two"
        Case 1 : curR = "One"
    End Select

    If k = 7 Then b = True
    If num = 0 Then
        If k = 7 Then GoTo c
        If b = False And k = 4 Then GoTo c
        Digit = ""

```



```

        End If
c:
        Convert1 = curR & Digit

    End Function

    Private Function Convert3(ByVal num As Integer, Optional ByVal
Digit As String = "") As String
        Dim curRR As String = ""
        Select Case num
            Case 9 : curRR = "Ninety"
            Case 8 : curRR = "Eighty"
            Case 7 : curRR = "Seventy"
            Case 6 : curRR = "Sixty"
            Case 5 : curRR = "Fifty"
            Case 4 : curRR = "Fourty"
            Case 3 : curRR = "Thirty"
            Case 2 : curRR = "Twenty"
        End Select

        If k = 7 Then b = True
        If num = 0 Then
            If k = 7 Then GoTo c
            If b = False And k = 4 Then GoTo c
            Digit = ""
        End If
c:
        Convert3 = curRR & Digit
    End Function

    Private Sub ResetArray()
        Dim i As Integer
        For i = LBound(A) To UBound(A)
            A(i) = 0
        Next
    End Sub

    Private Sub Button1_Click(sender As Object, e As EventArgs) Handles
Button1.Click
        Button1.Hide()
        Button2.Hide()
        Me.PrintForm1.PrintAction
        Printing.PrintAction.PrintToPreview
        Me.PrintForm1.Print()
        Button1.Show()
        Button2.Show()
    End Sub

    Private Sub Button2_Click(sender As Object, e As EventArgs) Handles
Button2.Click
        Me.Close()
    End Sub
End Class

```

Chapter 5: IMPLEMENTATION AND TESTING

5.1 Implementation

The term “implementation” in software engineering refers to the process that depicts how the information system will be deployed, installed and transitioned into an operational system. The process contains an overview of the system, a brief description of the major tasks involved in the implementation, the overall resources needed to support the implementation effort (such as hardware, software and materials). The plan is developed during the Design phase and is updated during the Development phase; the final version is provided in the Integration and Test phase and is used for guidance during the Implementation Phase. The outline shows the structure of the Implementation plan.

For implementing “Departmental Store Management System” the plan is to train the staff of the organization for one day. When staff run the system in real time basis, the problem is solved if the operator don’t finds any problem.

Direct changeover is one of the methods to change from an existing system to a new one. With this method of implementation the users stop using the manual system and start using the computer system from a given date. The advantage of this method is that it is less costly in terms of effort and time than any other method of implementation. Its disadvantage is that if any problem occurs the users do not have any alternative apart from returning to the manual system unless any solid solution is found.

Parallel running is another way to change from an existing system to a new one. With parallel running, the new system is introduced alongside the existing system. Both systems (manual and computer, or old and new computer system) will be in operation at the same time. Its advantage is that the results from the new system can be compared with those of the old system.

5.1.1 Tools Used

- Server : Microsoft SQL Server 2012
- Developing Language : Visual Basic
- IDE : Microsoft Visual Studio 2017
- Framework : .NET Framework 4.6.1

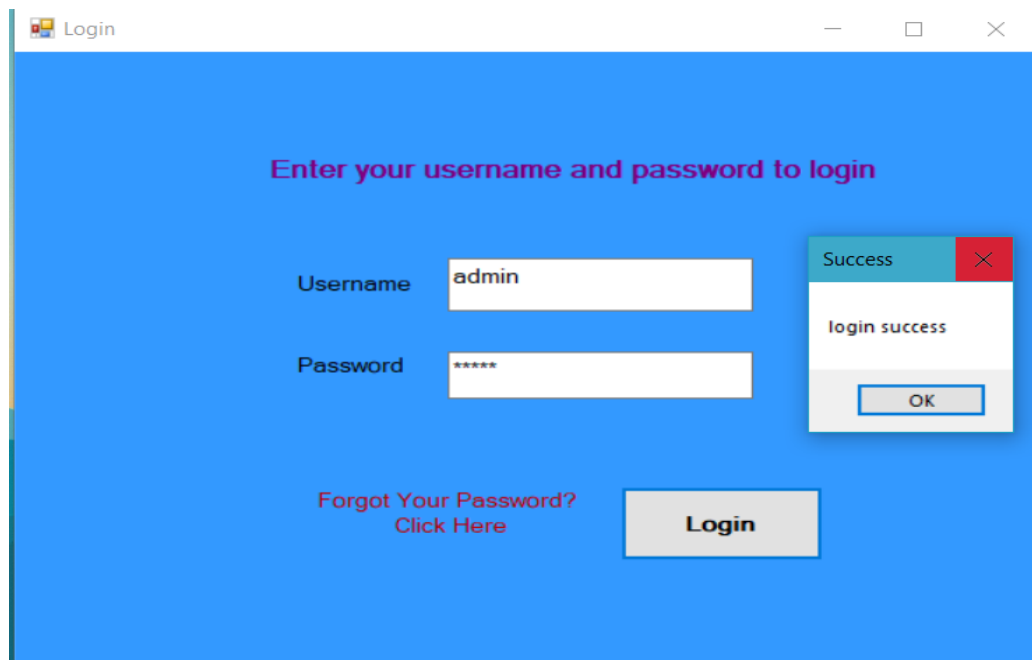
5.1.2 Description/List of major classes/methods

- Connection Module
- Login Class
- Add Product Class
- View Product Class
- Sales Entry Class
- Bill Prepare Class
- Sales Report Class
- Product Report ClassTesting

5.2 Testing

5.2.1 Unit Testing

Login Form



The screenshot shows a Windows-style application window titled "Login". The window has a blue background and contains the following elements:

- A heading: "Enter your username and password to login" in red text.
- Two input fields: "Username" with the value "admin" and "Password" with masked characters "*****".
- A "Login" button in a grey box.
- A link: "Forgot Your Password? Click Here" in red text.
- A small success dialog box in the top right corner with a green header "Success", the text "login success", and an "OK" button.

Figure 10 Login Form

Add User Form

The screenshot shows a window titled "Add New User" with a blue background. It contains four input fields: "Username" with the value "admin", "Password" with the value "DepartmentStoreManagementSystem", "User Type" (partially visible), and "Status" (partially visible). A white modal dialog box is centered over the form, displaying the message "Record Inserted Successfully" and an "OK" button. At the bottom of the window are three buttons: "Add User" (yellow), "Clear" (green), and "Cancel" (grey).

Figure 11 Add User Form

View User

The screenshot shows a window titled "View Users" with a blue background. On the left, there are input fields for "User ID" (value: 2), "Username" (value: admin), "Password" (masked with asterisks), "User Type" (dropdown menu showing "Administration"), and "Status" (value: Active). On the right, there is a table with the following data:

User ID	Username	Password	User Type	Status
1	david	david	Administration	1
2	admin	admin	Administration	1

Below the table is a large grey rectangular area. At the bottom of the window are three buttons: "Edit" (green), "Delete" (orange), and "Close" (grey).

Figure 12 View User

Category

The screenshot shows a Windows desktop with a taskbar at the top. The active window is titled 'Categories' and has a blue background. It contains a form for managing categories and a table of existing categories.

Form Fields:

- Category ID:** A text box containing the value '5'.
- Name:** A text box containing the value 'Drinking'.
- Description:** A text area containing the value 'drinking items only'.

Buttons:

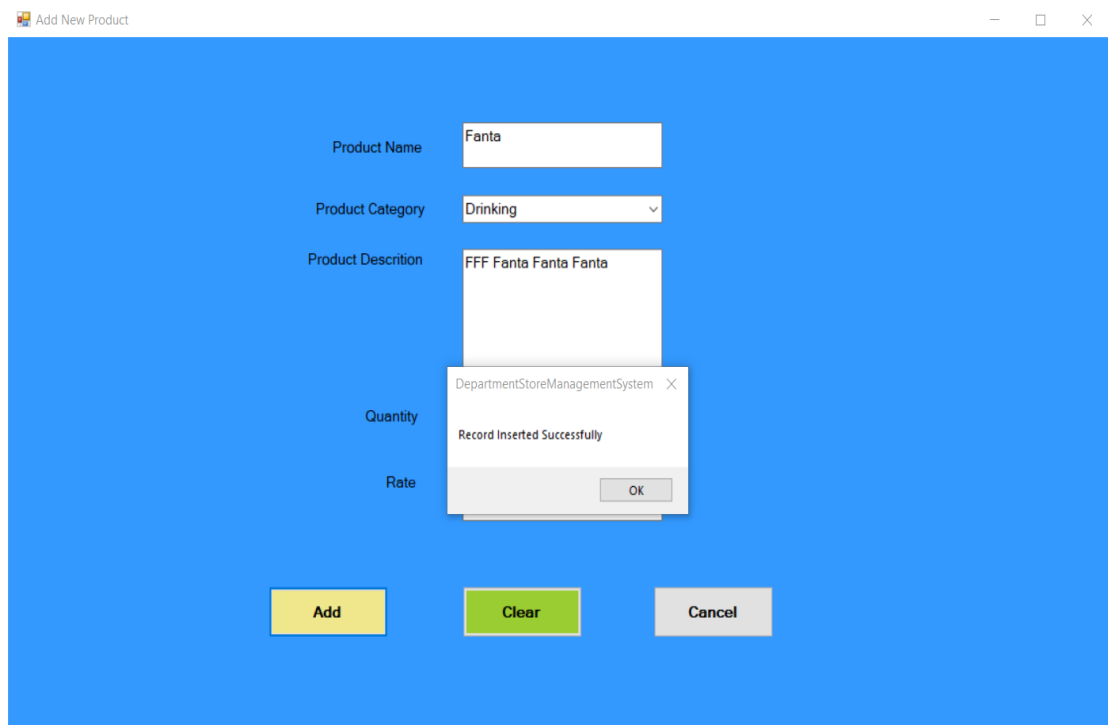
- Add New:** A yellow button.
- Edit:** A green button.
- Delete:** An orange button.
- Clear:** A green button.

Table:

Category ID	Name	Description
1	Fashion	Category for anything related to fashion.
2	Sports	Category for anything related to sports
4	Cooking	Cooking Materials
5	Drinking	drinking items only

Figure 13 Categories

Add Product



Product Name: Fanta

Product Category: Drinking

Product Description: FFF Fanta Fanta Fanta

Quantity:

Rate:

DepartmentStoreManagementSystem

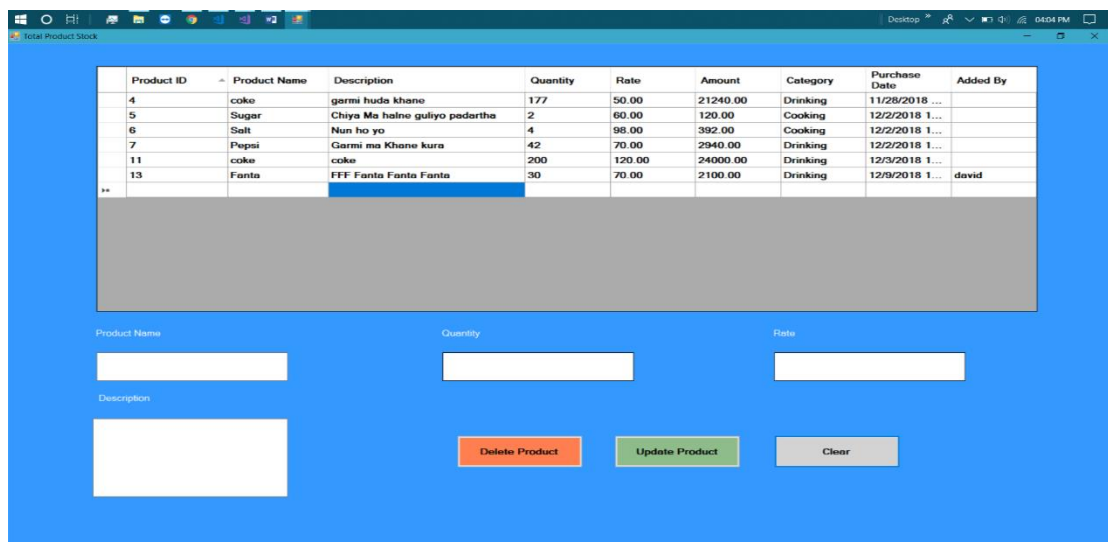
Record Inserted Successfully

OK

Add Clear Cancel

Figure 14 Add New Product

Product Stock



Product ID	Product Name	Description	Quantity	Rate	Amount	Category	Purchase Date	Added By
4	coke	garmi huda khane	177	50.00	21240.00	Drinking	11/28/2018 ...	
5	Sugar	Chiya Ma halne guliyo padartha	2	60.00	120.00	Cooking	12/2/2018 1...	
6	Salt	Nun ho yo	4	98.00	392.00	Cooking	12/2/2018 1...	
7	Pepsi	Garmi ma Khane kura	42	70.00	2940.00	Drinking	12/2/2018 1...	
11	coke	coke	200	120.00	24000.00	Drinking	12/3/2018 1...	
13	Fanta	FFF Fanta Fanta Fanta	30	70.00	2100.00	Drinking	12/9/2018 1...	david

Product Name:

Quantity:

Rate:

Description:

Delete Product Update Product Clear

Figure 15 Product Stock

New Sales Entry

Available Products For Sale

Product ID	Product Name	Description	Category	Quantity	Rate	AddedBy
4	coke	garmi hude ...	Drinking	177	50.00	
5	Sugar	Chiya Ma ha ...	Cooking	2	60.00	
6	Salt	Nun ho yo	Cooking	4	98.00	
7	Pepsi	Garmi ma K.	Drinking	42	70.00	
11	coke		Drinking	200	120.00	
13	Fanta	FFF Fanta F...	Drinking	30	70.00	devid

DepartmentStoreManagementSystem X

Sales Entry Inserted Successfully

OK

Figure 16 New Sales Entry

Sales Stock

Sales ID	Product Name	Category Name	Quantity	Rate	Discount	Vat	Amount	Sales Date	Sales By
10	coke	Drinking	10	50.00	0.00	65.00	565.00	12/2/2018 12:00	devid
11	Sugar	Cooking	3	60.00	0.00	23.00	203.00	12/2/2018 12:00	devid
11	Salt	Cooking	9	98.00	0.00	115.00	997.00	12/2/2018 12:00	devid
12	coke	Drinking	4	50.00	0.00	26.00	226.00	12/2/2018 12:00	devid
13	Sugar	Cooking	9	60.00	0.00	70.00	610.00	12/2/2018 12:00	devid
14	Sugar	Cooking	4	60.00	0.00	31.00	271.00	12/2/2018 12:00	devid
15	Sugar	Cooking	3	60.00	0.00	23.00	203.00	12/2/2018 12:00	devid
15	coke	Drinking	8	50.00	0.00	52.00	452.00	12/2/2018 12:00	devid
16	Sugar	Cooking	1	60.00	0.00	8.00	68.00	12/2/2018 12:00	devid
16	coke	Drinking	2	50.00	0.00	13.00	113.00	12/2/2018 12:00	devid
17	coke	Drinking	1	50.00	0.00	6.00	56.00	12/2/2018 12:00	devid
17	Pepsi	Drinking	2	70.00	0.00	18.00	158.00	12/2/2018 12:00	devid
20	Jacket	Fashion	50	1500.00	0.00	9750.00	94750.00	12/2/2018 12:00	devid
21	Jacket	Fashion	10	1500.00	0.00	1950.00	16950.00	12/2/2018 12:00	devid

Sales ID

Activate Windows
Go to Settings to activate Windows.

Figure 17 Sales Stock

Data Backup / Recovery

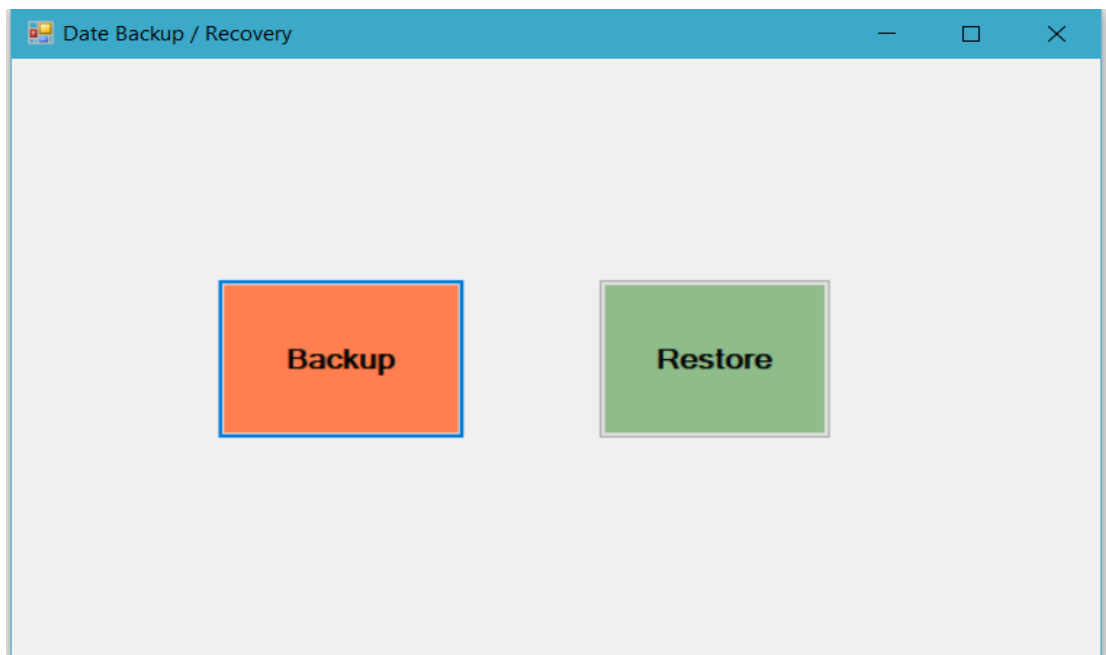


Figure 18 Data Backup & Recovery

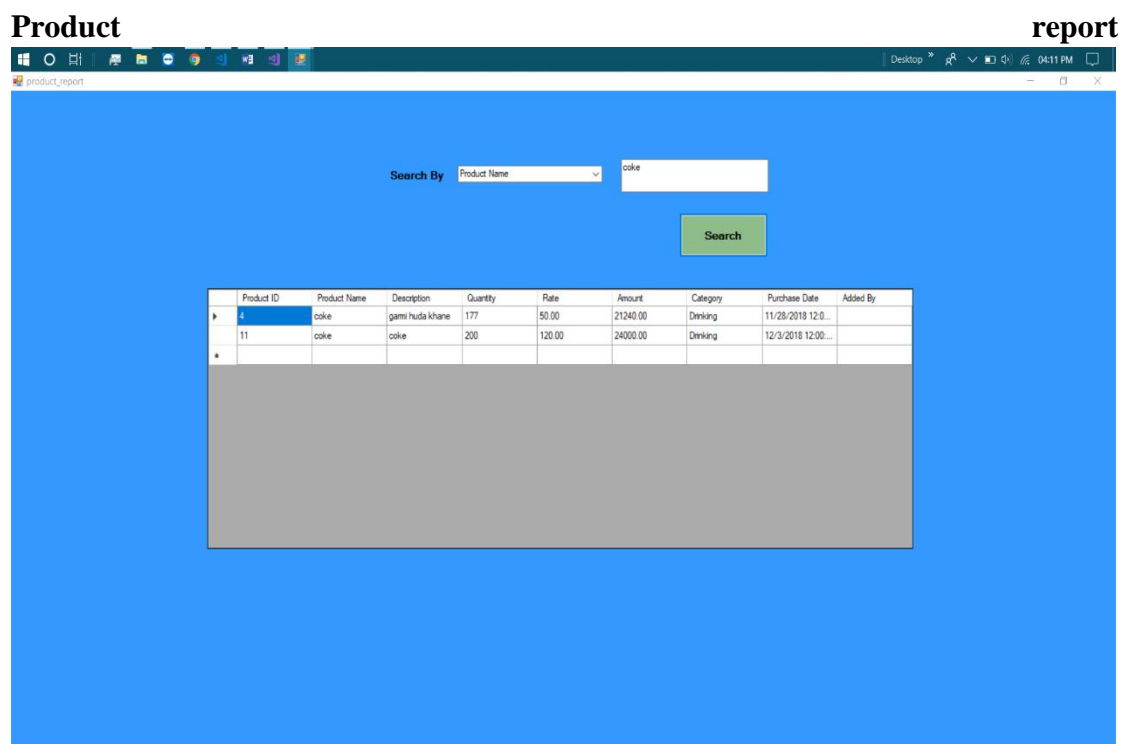


Figure 19 Product Report

Customer Bill

Bill Prepared

Business Invoice

Store Management System
Gaindakot
Phone No: 0565224187
Email : linkwithdm@gmail.com

Date: 12/9/2018
Invoice : 30

	S.N	Product Name	Quantity	Rate	Amount
▶	1	coke	5	120.00	600.00
	2	Sugar	1	60.00	60.00
	3	Salt	2	98.00	196.00

Sub Total856
Vat(13%)111.00
Total Due967

Prepared By: david

Total Due : Nine Hundred SixtySeven Only

Notes :

Print

Cancel

Figure 20 Customer Bill

5.2.2 System Testing Test Cases

Test Case No.	TC01	
Test Phase	:	Unit Test
Functionality	:	Add User's information to the database by the admin user
Environment	:	Add user in the Master Menu of main form.

Procedure	:	<ol style="list-style-type: none"> 1. Input user information such as Username, type, ,password and status. 2. Click on 'Add user' button. 3. A 'confirm' message will appear. Click 'Ok' button to store the entered user information in the database.
Expected Outcome	:	The information relating to the newly added user should be displayed in the "View User" page by the admin.

Test Case No.	TC02	
Test Phase	:	Unit Test
Functionality	:	Display Products Stock
Environment	:	Report form of Master menu of main form.
Procedure	:	<ol style="list-style-type: none"> 1. Go to Master → Report → Products Report 2. Select the option from and to then click on search button.
Expected Outcome	:	All the products reports should be displayed between the option selected above.

Test Case No.	TC03	
Test Phase	:	Unit Test
Functionality	:	Add Product's information to the database by the Admin and user
Environment	:	Add product's in the Master Menu of main form.

Procedure	:	<ol style="list-style-type: none"> 1. Input products information such as name, quantity, rate, amount, date etc. 2. Click on 'Add Product' button. 3. A 'confirm' message will appear. Click 'Ok' button to store the entered products information in the database.
-----------	---	--

Test Case No.	TC04	
Test Phase	:	Unit Test
Functionality	:	Add Sales Entry information to the database by the Admin and user
Environment	:	New Sales Entry in the Master Menu of main form.
Procedure	:	<ol style="list-style-type: none"> 1. Choose Category and Choose Product id and en quantitya and discount. 2. Click on 'Add Sales. 3. Click button to store the entered products information in the database and generate bill.
Expected Outcome	:	The information relating to the newly added products should be displayed in the "View Product Stock" page by the admin And user.

Test Case No.	TC05	
Test Phase	:	Unit Test
Functionality	:	Display All System Users
Environment	:	View Users form of Master menu of main form.
Procedure	:	1. Go to Master → Users → View Users.
Expected Outcome	:	All the users information should be displayed.

Test Case No.	TC06	
Test Phase	:	Unit Test
Functionality	:	Display Sales Stock
Environment	:	Report form of Master menu of main form.
Procedure	:	1. Go to Master → Report → Sales Report 2. Select the option from and to then click on search button.
Expected Outcome	:	All the sales reports should be displayed between the option selected above.

Test Case No.	TC07	
Test Phase	:	Unit Test
Functionality	:	Data Backup & Recovery
Environment	:	Backup / Recovery form of Master menu of main form.

Procedure	:	1. Go to Master → Backup / Recovery 2. Select the backup for data backup and restore for data recovery.
Expected Outcome	:	All the data should be stored in the respected drive and restored from drive.

Test Cases

Test Case No	Date	Pass/Fail	Comments
TC01	7-Dec-18	PASS	The newly added member details should be displayed at the registration page and can be viewed by the admin.
TC02	7-Dec-18	PASS	The products details should be displayed in the products report at appropriate format.
TC03	7-Dec-18	PASS	The newly added product details should be displayed at the page and can be viewed by the admin.
TC04	7-Dec-18	PASS	The newly added sales entry details should be displayed at the sales stock page and can be viewed by the admin.

TC05	7-Dec-18	PASS	The users details should be displayed in the view user page at appropriate format.
TC06	7-Dec-18	PASS	The sales details should be displayed in the sales report at appropriate format.
TC07	7-Dec-18	PASS	Data backup success and restore success if a problem occurs in database.

Chapter 6: CONCLUSION AND RECOMMENDATION

6.1 Conclusion

Building a awesome project is really a tough. It takes a lot of mind labor during each phase of Development cycle. Our project is built taking long time in designing and coding phase. We have took a couple of sample projects during the designing and developing phase as well. It is build by using several modules and integrating them in a single piece. Our Project has achieved simplicity and hassel-free environment by the end of development phase. Using our system, store admin and users can easily deal with products and their issues.

Visual Basic is chosen as the front end application. VB applications are typically compiled to make an executable file that can run on any computer regardless of computer architecture. Visual Basic is one of the most popular programming languages in use, particularly for client-server applications, with millions of users. SQL server is chosen for back end application

6.2 Recommendations

On the basis of the aforementioned description of this project, we recommend this software to be used in any mid and small departmental stores as well as shops. Though it isn't perfect, it can prove to be quite good enough in the context of countries like Nepal. It can easily solve the burden on manual data management by implementing the computerized form. For best results and performance of this software, we recommend all the necessary hardware and software requirements mentioned above to be fulfilled. If so, we are absolutely convinced that it won't let the operator down.

References / Bibliography

References

- <https://docs.microsoft.com/en-us/>
- <https://stackoverflow.com/>

Bibliography

- Pradeep K. Sinha and Priti Sinha (2010), Foundations of Computing , 3rd edn, BPB Publications