IDM ACHIEVERS INTERNATIONAL CAMPUS

Course Title	Diploma in ICT & Computing
Unit Name	Advanced Visual Programming (VB.net)
Unit Number:	Unit 11
Assessor Name:	Mr: ☐ Mrs: ☐ Ms: ☒
	Primani Amarasiri
Branch Name:	Galle

Student's Name	**Title** Mr: ☐ Mrs: ☐ Ms: ☒		
	G. L. P. Wijesinghe		
Registration Number	46472		
Due Date:	31 / 07 /2017	**Date Submitted:**	27 / 07 /2017

Learning Outcomes Covered:

1. Demonstrate and deliver a range of transferable skills.
2. Show evidence of Working and contributing to a group situation
3. Identify a given problem and provide feasible solutions
4. Monitor and review own leaning experience

Assessor Comments:

Assessor Name: **Signature:** **Date:**

Task 01	*Task 02*	*Task 03*	*Task 04*	*Task 05*	*Task 06*	*Task 07*

Total Marks	

Internal Verifier's Comments:

Internal Verifier: **Signature** **Date:**

Assignment

Advanced Visual Programming (Vb.Net)

Diploma in ICT & Computing

Student Number: 46472

G. L. P. Wijesinghe

Advanced Visual Programming
Unit 11

Stock Management System

In real world with each and every organization we can see the requirement of managing stocks (inventory). In this assignment student need to do a survey to examine the processes and functions regarding to Stock Management Systems and need to implement user friendly software to handle stock.

Assignment

Main requirements of the system:

1. The system must include managing of stock (Inventory) including the following function
    - Maintain the Stock Master File (SMF)
    - Maintaining the Transaction Procession File (TRF)
    - Supplier Registrations

2. Report generations regarding to the above system
    - Supplier details report
    - Transaction processing detail report

(You may add any additional requirement for the database and user interfaces)

Aims

To implement the software system outlined above.

- Provide an effective and an efficient performance.
- The user view of the system will be a number of interactive windows forms implemented in visual basic.
- It is critical that data validation and error trapping is comprehensive.
- Ensure that the input is acceptable to and understood by the staff

Assume you are an individual to develop this system, which leads you to perform as a project manager, system analyst, programmer, etc…

Task 1

1.1 Briefly explain the Software Development Cycle.
1.2 Briefly explain what is .Net Framework and Features of .Net Framework.
1.3 Explain about Visual Basic main IDE Components.

Task 2

2.1 Design a database to include necessary tables.
2.2 Set up the database structure by using suitable RDBMS. (MS Access, MySQL or MS SQL Server)
2.3 Populate the database with suitable test data.

Task 3

3.1 The designs of all the required algorithms are required as part of this task. You can use suitable design techniques (Class diagrams, Activity diagrams, Use case diagrams etc.)

3.2 Design necessary user interfaces.

Task 4

4.1 Using VB programming language implement the stock management system designed in Task 3.2

Task 5

5.1 Carry out complete system testing and provide user guide. System testing and user guide must include suitable screenshots. Test your complete project with suitable data.

Task 6

6.1 Produce a publishable working copy a compiled assignment together with software installation notes. The installation note should include the system requirements.

Task 7

7.1 Develop the Java Programs for Practical Assignment 1, Assignment 2 and Assignment 3 in Book II.

Outcome:

The student must submit the hard copy of the completed assignment along with an attached softcopy.

NOTE: All media should be free of viruses. Any media that contains viruses will be subjected to a Fail grade

Acknowledgement

I am taking this opportunity to express my gratitude to everyone who supported me to complete this Assignment. I am thankful for their guidance, invaluable advice during the work. I am sincerely grateful to them for sharing their truthful and illuminating views on a number of issues related to this assignment.

Specially, I express my warm thanks to Ms. Primani & Ms. Dilani for their support and guidance.

Contents

Picture Contents

Table Contents

Introduction

This documentation is about developing a Stock Management System. This system is made out of 3 major forms, 2 Crystal reports and one database. The forms are made by Microsoft Visual Studio 2010. Database structures are made by Microsoft SQL Server 2014 Management Studio. Stock Master File, Transaction Procession File and Supplier Registration File are the main forms. The database includes 3 tables such as SMF, TRF and SMF. Transaction Procession Details Report and Supplier Details Report are the reports created by Crystal Reports Reporting Wizard.

So this documentation provides a brief description of the software. Further this includes development steps. And also this documentation includes guide for installation and operational instructions for the user.

TASK 1

1.1 Briefly explain the Software Development Cycle.
1.2 Briefly explain “What is .Net Framework and Features of .Net Framework.”
1.3 Explain about Visual Basic main IDE Components

Software Development Cycle

A software development lifecycle is essentially a series of steps, or phases, that provide a model for the development and lifecycle management of an application or piece of software.

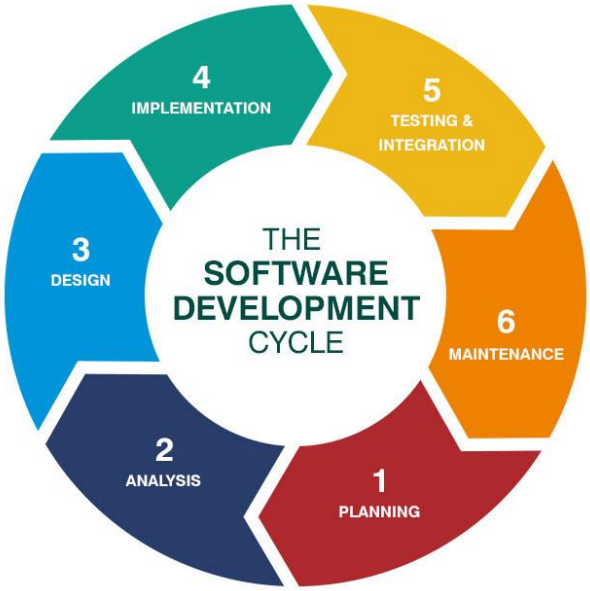


Figure 1

1. Planning

This stage is the most fundamental part of the SDLC process. It is performed primarily by the development team. This stage is the foundation of the requirement analysis. Once everyone is clear on the specifications, the team can use the information gained from outside sources to plan the basic project approach and identify potential risks. This involves determining how the project helps further the organization's business objectives.

2. Analysis

The analysis stage of the SDLC process involves defining project goals as functions. Next, the development team determines what operations the intended application should have. In general, analysis requires gathering and interpreting facts, as well as diagnosing issues with the current system and recommending improvements.

3. Design

In this stage, the team will design the product's optimal architecture. The internal design of the architecture should be defined in a complete, detailed manner. Components that should be included are screen layouts, business rules, process diagrams and other documentation. Developers and software engineers must be able to develop and deliver the system based on this information with minimal additions.

4. Implementation

During implementation, the team builds the product and creates programming code. During this stage, it is important for developers to follow the coding guidelines defined by their organization. Developers determine the language to use based on the type of software along with customer requirements. At the end of this stage, the software is put into production.

5. Testing and Integration

This stage of the SDLC involves bringing the separate parts of the project together into a dedicated testing environment to check for errors, bugs and other issues. During the testing phase, the product is checked to ensure that defects are reported, tracked, fixed, and tested again until the product meets quality standards.

6. Maintenance

When testing is complete and the product is ready for deployment, it is time for its release into the marketplace. After the product's release, maintenance is carried out for the customer. The team makes software improvements or change requests as needed. The ultimate goal of the maintenance phase is to ensure that the product remains relevant and high quality. It involves ongoing evaluations of the system's performance.

What is .NET Framework?

The Microsoft .Net Framework is an integrated and managed environment for the development and execution of your code.

The .Net Framework is a managed type-safe environment for application development and execution. The .Net Framework manages all aspects of your program's execution. It allocates memory for the storage of data and instructions, grants or denies the appropriate permissions to your application, initiates and manages application execution, and manages the reallocation of memory from resources that are no longer needed.

The .Net Framework consists of two main components:

1. Common language runtime
2. The .Net Framework class library

Common language runtime

The common language runtime can be thought of as the environment that manages code execution. It provides core services, such as code compilation, memory allocation, thread management, and garbage collection. Through the common type system (CTS), it enforces strict type-safety and ensures that code is executed in a safe environment by also enforcing code access security.

The .NET framework class library

.NET framework class library (FCL) is a comprehensive, object oriented collection of reusable types that you can use to develop applications ranging from traditional CLI or GUI applications to applications based on the latest innovations provided by ASP.NET such as web forms and XML web services. This provides user interface, data access, database connectivity, cryptography, web application development, numeric algorithms and network communications. There are pre-defined and user defined class libraries.

Some important features of .NET framework

- Assemblies – An assembly is either a .dll (dynamic line output library) or .exe (executable file) that forms a part of an application. It contains MSIL (Microsoft intermediate language) code which is executed by CLR. It is the unit on which permissions are granted. It contains a version, interfaces, classes and assembly metadata which contains information about assembly.

- Common type system – This specifies the rules related to data types that languages must follow. As programs written in all languages are ultimately converted to MSIL, data types in all languages must be convertible to certain standard data types. This is a part of cross language integration which allows

classes written in one language to be used and extended by another language.

- Cross language interoperability - .NET provides supports for language interoperability. However it doesn't mean every program written in a language can be used by another language. To enable a program to be used with other languages, it must be created by following a set of rules called cross language specifications. Cross language inheritance is the ability to create a class in C# from a class created in VB.NET. When an exception is raised by a program written in C# the exception can be handled by VB.NET. This kind of exception handling is called cross language exception handling.

Different types of applications that can be developed in .NET

1. Windows applications – typical client server applications
2. Web applications – web sites and intranet applications
3. Web services – programs that are accessible from anywhere using universal protocols like http and SOAP
4. Console applications – simple console based applications without any GUI run from command prompt .Best suited to learn fundamentals and also for applications such as server sockets
5. Mobile apps – contain web pages that run in mobile devices such as PDAs and cell phones

Visual Basic main IDE Components.

Integrated Development Environment or IDE is a software suite that combines the basic tools which developers need to write and test softwares. It consists following elements.

Microsoft Visual Studio 2010

Figure 2

Microsoft Visual Studio is an integrated development environment (IDE) from Microsoft. It is used to develop computer programs for Microsoft Windows, as well as websites, web apps, web services and mobile apps. Visual Studio uses Microsoft software development platforms. It can produce both native code and managed code. Visual Studio supports different programming languages and allows the code editor and debugger to support nearly any programing language, provided a language-specific service exists.

Menu Bar

The Menu Bar displays the commands that are required to build an application. The main Menu item has sub Menu item that can be chosen when needed. The toolbars in the menu bar provide quick access to the commonly used commands. Besides the standard File, Edit, View, Widow and Help menus, Visual Basic Menu Bar provides access to functions which are specific to programming such as Project, Format or Debug.

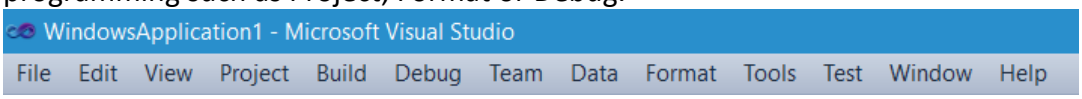


Figure 3

Tool Bar

The toolbox contains a set of controls (tools) that are used to place on a form at design time and that are used to design the interfaces of an application. Provide quick access to commonly used commands in the programming environment. You click a button on the toolbar once to carry out the action represented by that button. By default, the Standard toolbar is displayed when you start Visual Basic. Additional toolbars for editing, form design and

debugging can be toggled on or off from the Toolbars command on the View menu. Toolbars can be docked beneath the menu bar or can "float" if you select the vertical baron the left edge and drag it away from the menu bar.

Figure 4

Toolbox

This provides a set of tools that you use at design time to place controls on a form. In addition to the default toolbox layout, you can create your own custom layouts by selecting “Add Tab” from the context menu and adding controls to the resulting tab.

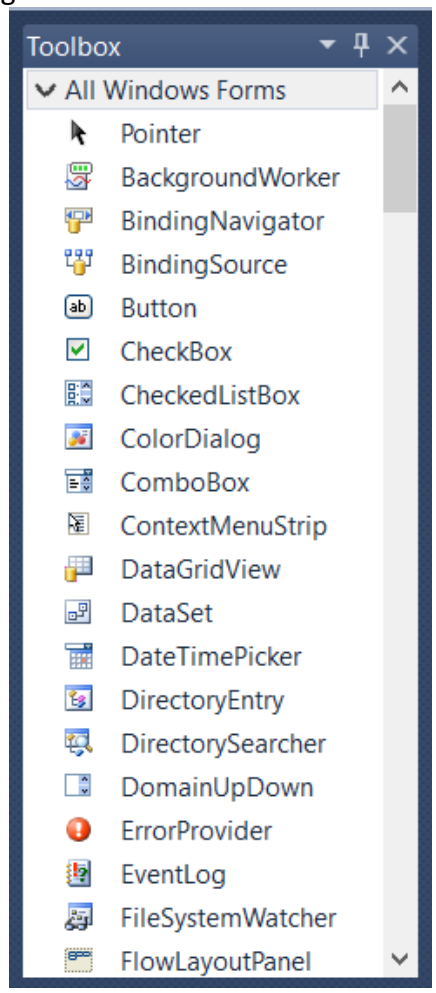


Figure 5

Solution Explorer Window

This lists the forms and modules in your current project. A project is the collection of files you use to build an application.

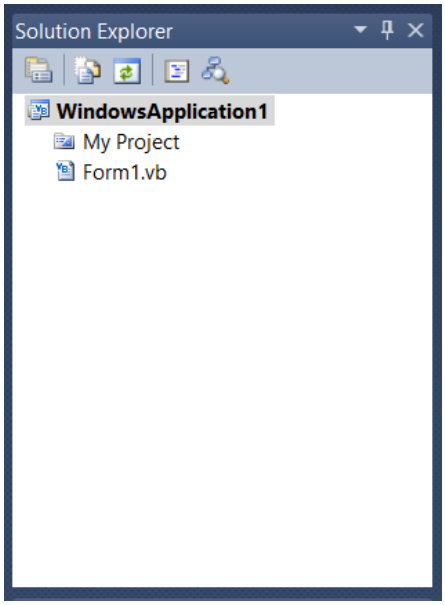


Figure 6

Properties Window

This lists the property settings for the selected form or control. A property is a characteristic of an object, such as size, caption or color.

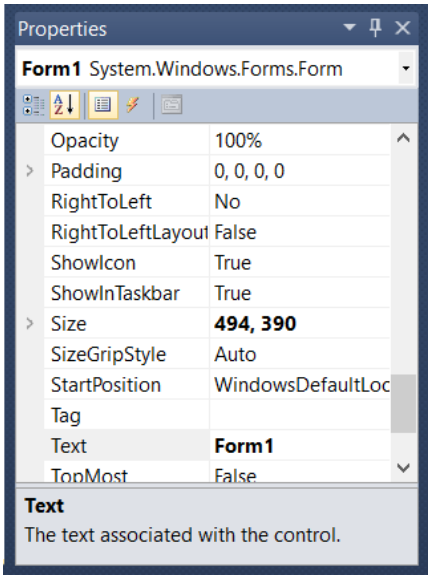


Figure 7

Object Browser

Lists objects available for use in your project and gives you a quick way to navigate through your code. You can use the Object Browser to explore objects in Visual Basic and other applications, see what methods and properties are available for those objects and paste code procedures into your application.

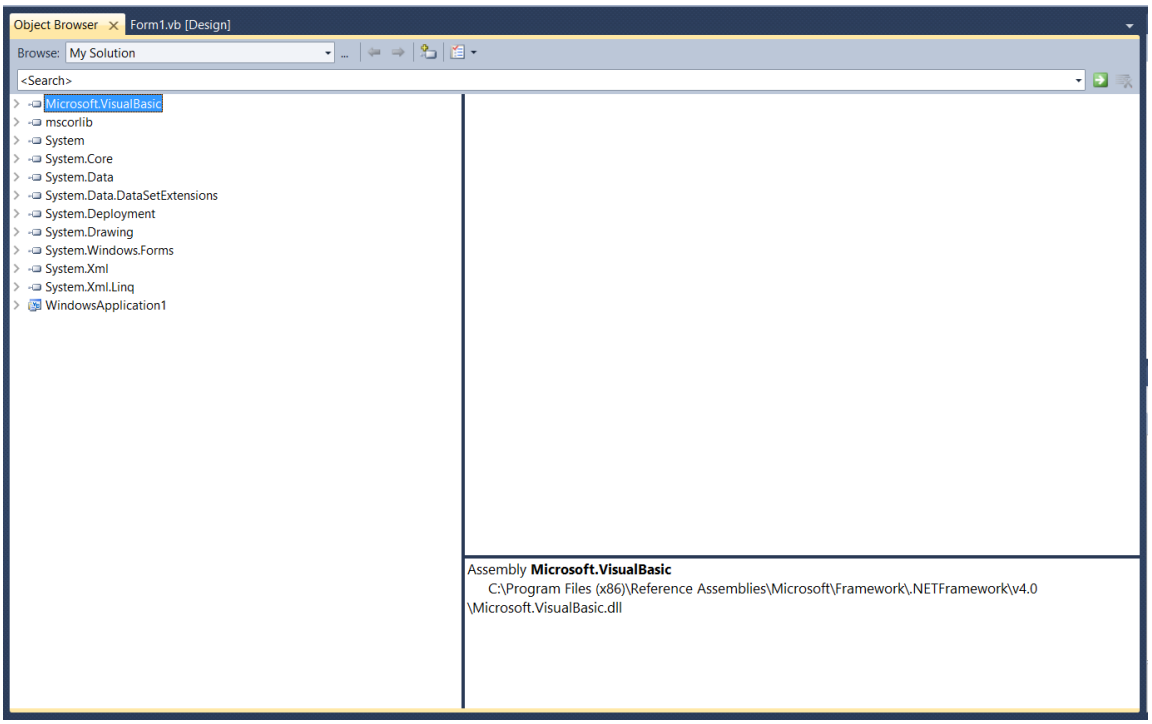


Figure 8

Form Designer

This serves as a window that you customize to design the interface of your application. You add controls, graphics, and pictures to a form to create the look you want. Each form in your application has its own form designer window.

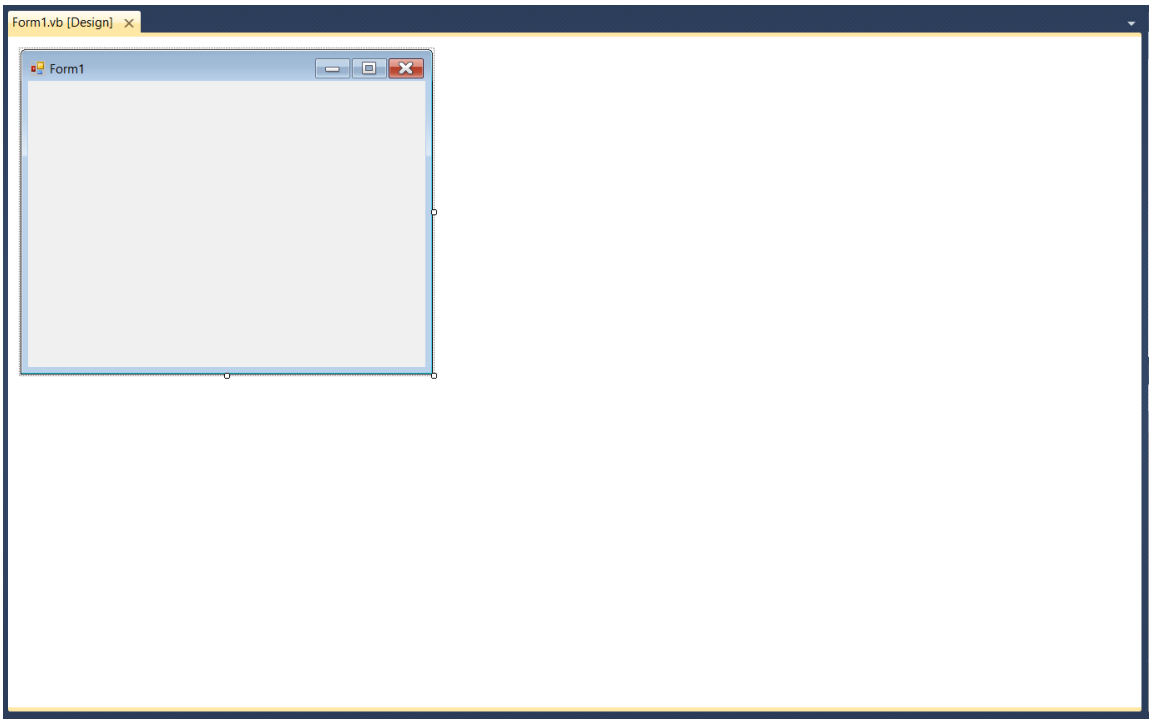


Figure 9

Code Editor Window

Visual basic code editor is used to write a code which responds to control events. For example the “load event of form control”

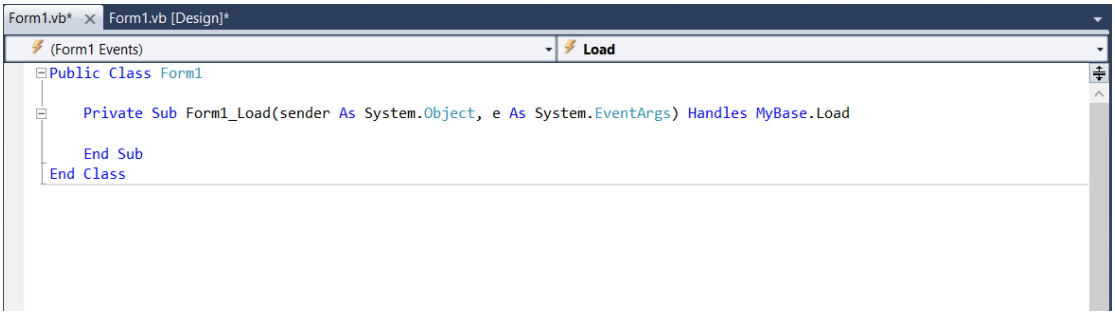


Figure 10

Task 2

2.1 Design a database to include necessary tables.
2.2 Set up the database structure by using suitable RDBMS. (MS Access, MySQL or MS SQL Server)
2.3 Populate the database with suitable test data.

Design a Database to include necessary tables

1. SMF table for Stock Master File

Column Name	Data Type
Item Code	Integer
Item Name	Varchar(50)
Item Price	Money
Stock Balance	Integer
Store Location	Varchar(50)
Stock Level	Text
Last Date of Transaction	Date

Table 1

2. TRF table for Transaction Procession File

Column Name	Data Type
Item Code	Integer
Item Name	Varchar(50)
Transaction Type	Text
Quantity	Integer
New Stock Balance	Integer
Last Date of Transaction	date

Table 2

3. SUPREG table for Supplier Registration

Column Name	Data Type
Supplier ID	Integer
Name	Varchar(50)
Address	Varchar(50)
Email	Varchar(50)
Tel M	Integer
Tel F	Integer

Table 3

Set up the database structure by using suitable RDBMS (MS Access, MySQL or MS SQL Server)

The Database structures are set up by Microsoft SQL Server Managements Studio 2014.

Microsoft SQL Server 2014 Management Studio

Figure 11

Microsoft SQL Server is a relational database management system developed by Microsoft. As a database server, it is a software product with the primary function of storing and retrieving data as requested by other software applications-which may run either on the same computer across a network (including the Internet).

Microsoft SQL Server 2014 features characteristics

A central feature of SSMS is the Object Explorer, which allows the user to browse, select, and act upon any of the objects within the server. Recent versions of SSMS are fully capable of connecting to and manage any SQL Server Express instance. Microsoft also incorporated backwards compatibility for older versions of SQL Server thus allowing a newer version of SSMS to connect to older versions of SQL Server instances.

Microsoft SQL Server Management Studio (SSMS) is an integrated environment to manage a SQL Server infrastructure. It provides a user interface and a group of tools with rich script editors that interact with SQL Server.

SSMS tools

SSMS provides tools to configure manage and administer instances of Microsoft SQL Server, and it brings together a range of graphical and visual design tools and rich script editors to simplify working with SQL Server. SSMS combined features come from Enterprise Manager, Query Analyzer and Analysis Manager, along with features

included in previous releases of SQL Server. It supports most of SQL Server's administrative tasks and maintains a single, integrated environment for SQL Server Database Engine management and authoring.

SSMS components

Microsoft SQL Server Management Studio features include Object Explorer, which can view and manage all objects in a SQL Server instance; Template Explorer, which builds and manages files of text that can be reused to speed up query and script development; Solution Explorer, which builds the projects used to manage administration items, such as queries and scripts. SSMS components customize keyboard shortcuts and viewing property pages; connect to instances of the Database Engine and Analysis Services; visual design tools; and interactively builds and debug queries and scripts.

Primary Key

A primary key is a table column or a combination of columns of a relational database designated to identify all table records uniquely. A primary key contain a unique value. It cannot contains a null value

Loading Microsoft SQL Server

Click the search icon in the task bar and type SQL. Then click SQL Server Management Studio.

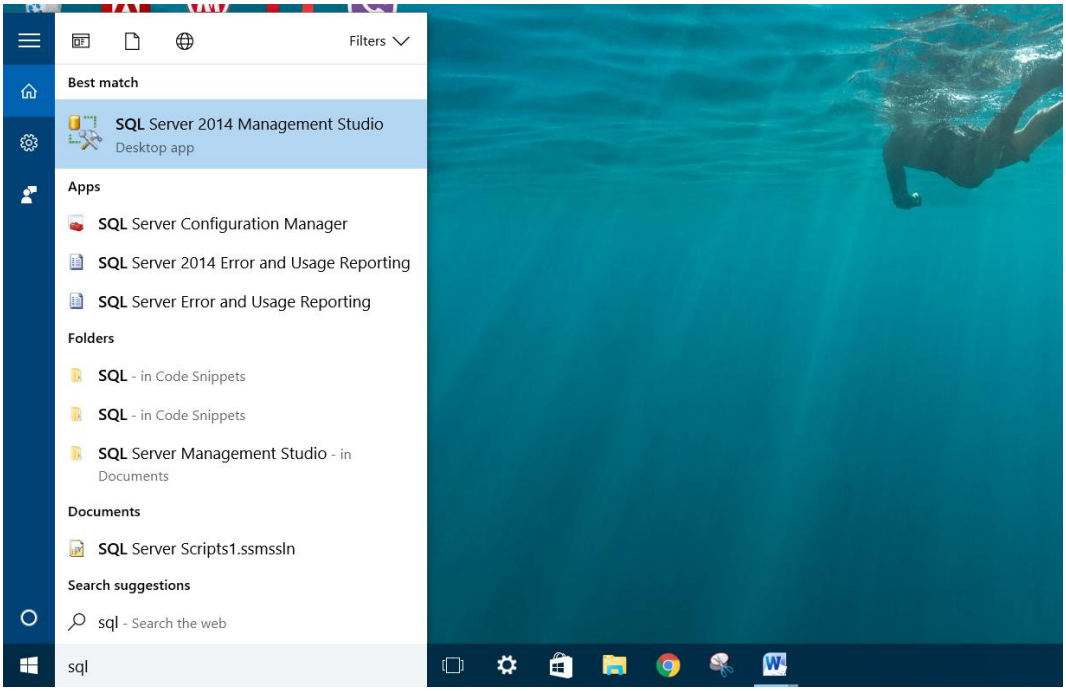


Figure 12

Establishing the Connection

Open Microsoft SQL Server Managements Studio 2014 software and click "Connect" as below diagram in order to connect to the server.

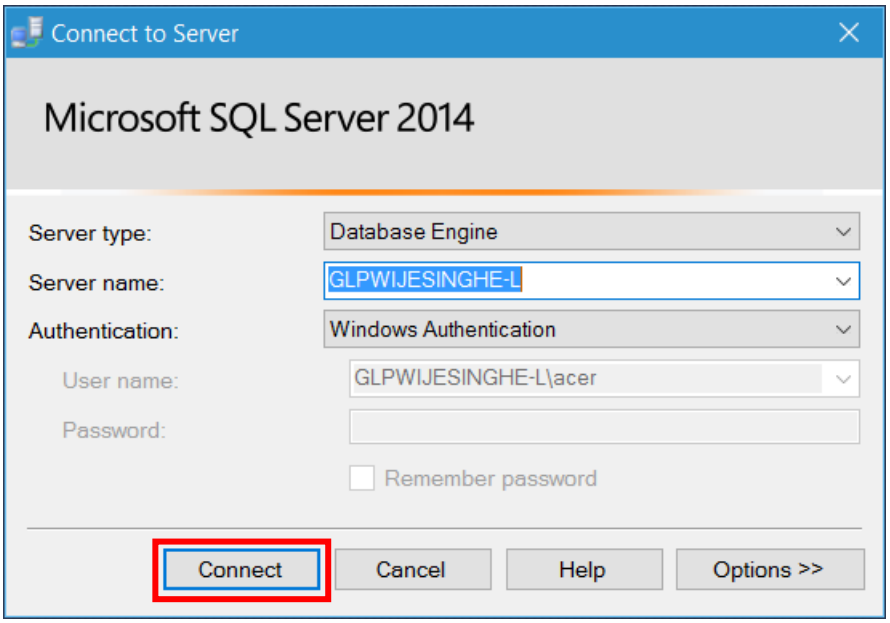


Figure 13

Creating the Database

Then Right Click on Databases from the Object Explorer tab. Then Click on "New Database…" option from the context menu.

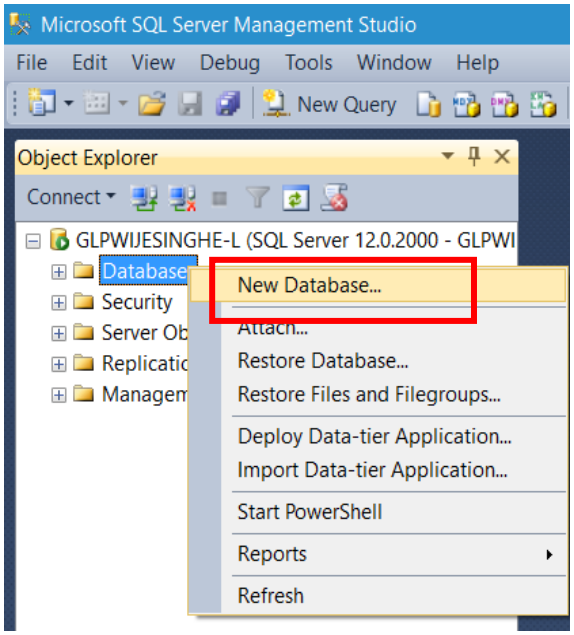


Figure 14

Then give a name for your Database and click “OK” in the window shown below.

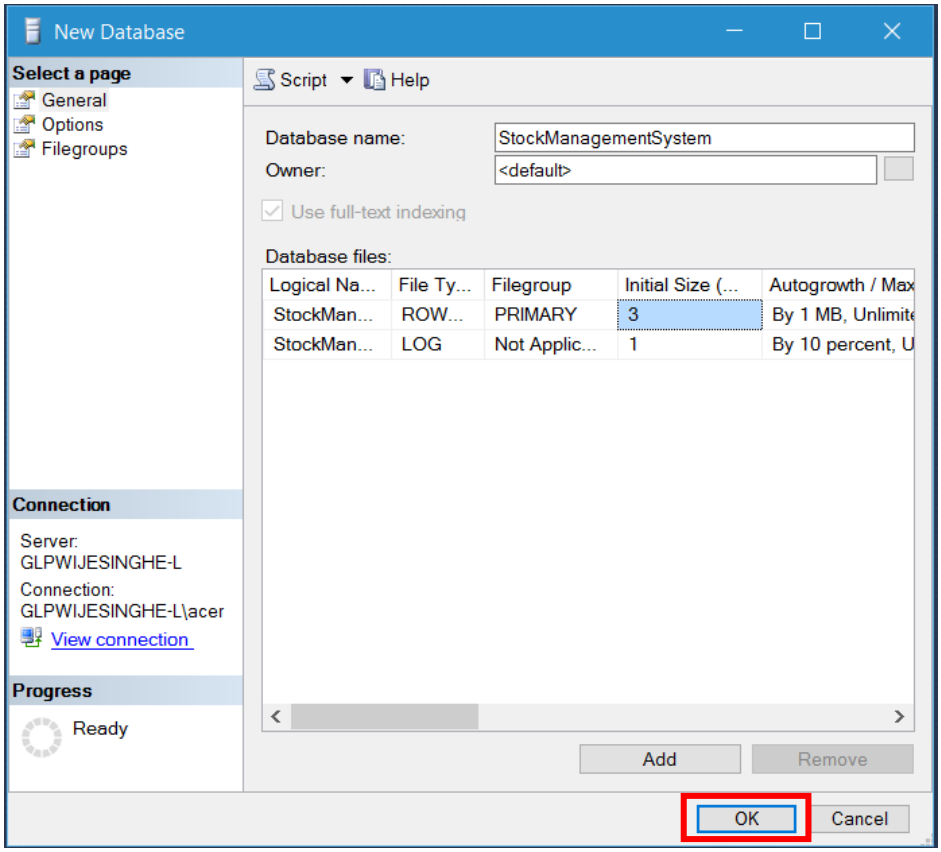


Figure 15

Now you have created the Database.

Opening the Database

Click on the “Expand mark (+)” of Databases option under the Object Explorer tab.

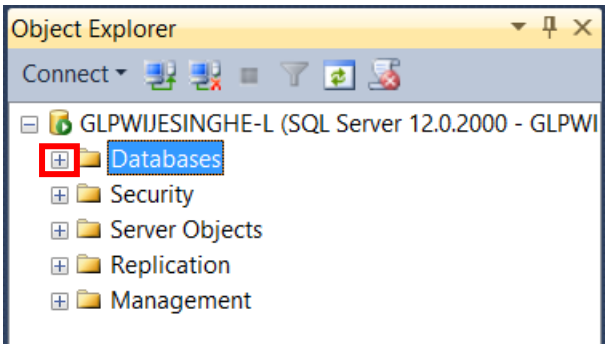


Figure 16

Then Select your database. Expand it.

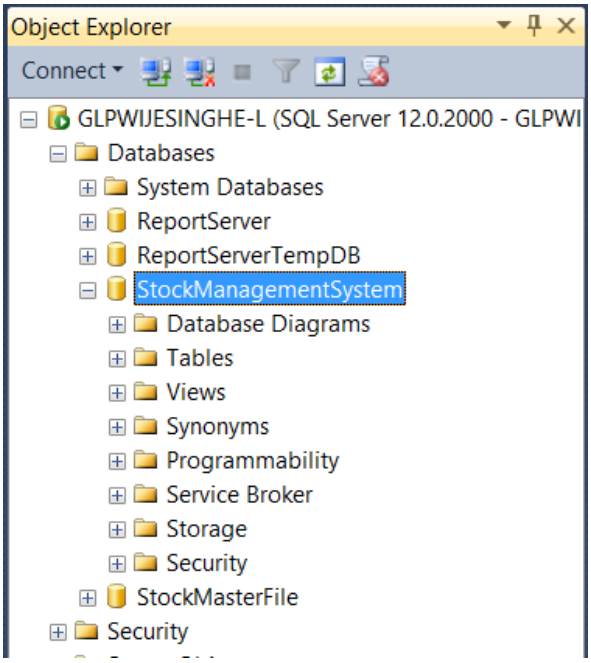


Figure 17

Creating a Table

Now Right Click on “Tables” section and from the context menu click on “Table…” option.

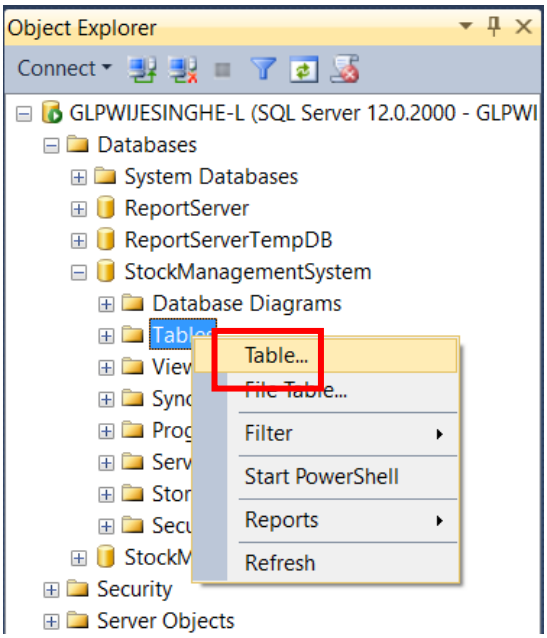


Figure 18

Now create the table design as you need with the appropriate data types.

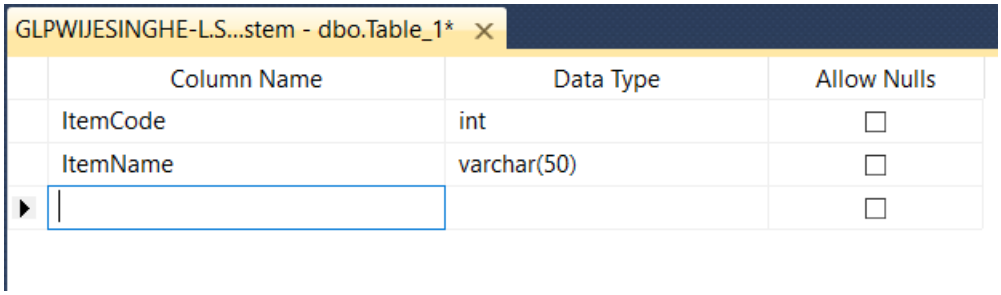


Figure 19

Data types

Data Type	Description
Text	Allows to string characters only
Varchar(50)	Allows to numeric and string characters
int	Allows to integer numbers only
Date	Allows calendar dates
Money	Allows currency

Table 4

Enter a name for the Table. Then save it.

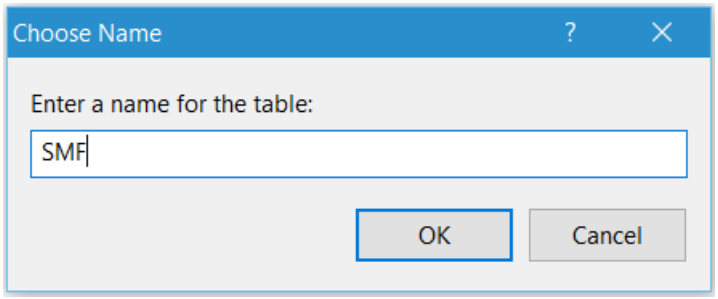


Figure 20

Design view of SMF table

GLPWIJESINGHE-L.S...tSystem - dbo.SMF

Column Name	Data Type	Allow Nulls
ItemCode	int	☐
ItemName	varchar(50)	☐
ItemPrice	money	☐
StockBalance	int	☐
StoreLocation	varchar(50)	☐
StockLevel	text	☐
LastDateofTransaction	date	☐

Figure 21

Design view of TRF table

GLPWIJESINGHE-L.S...tSystem - dbo.TRF

Column Name	Data Type	Allow Nulls
ItemCode	int	☐
ItemName	varchar(50)	☐
ItemPrice	money	☐
TransactionType	text	☐
Quantity	int	☐
TotalPrice	money	☐
NewStockBalance	int	☐
LastDateofTransaction	date	☐

Figure 22

Design view of SUPREG table

GLPWIJESINGHE-L....stem - dbo.SUPREG

Column Name	Data Type	Allow Nulls
SupplierID	int	☐
Name	varchar(50)	☐
Address	varchar(50)	☐
Email	varchar(50)	☐
TelM	int	☐
TelF	int	☐

Figure 23

Opening and Editing a Table.

Expand the Tables option under your Database. Right Click on your table name (dbo.SMF) and then from the context menu select “Edit Top 200 Rows” option.

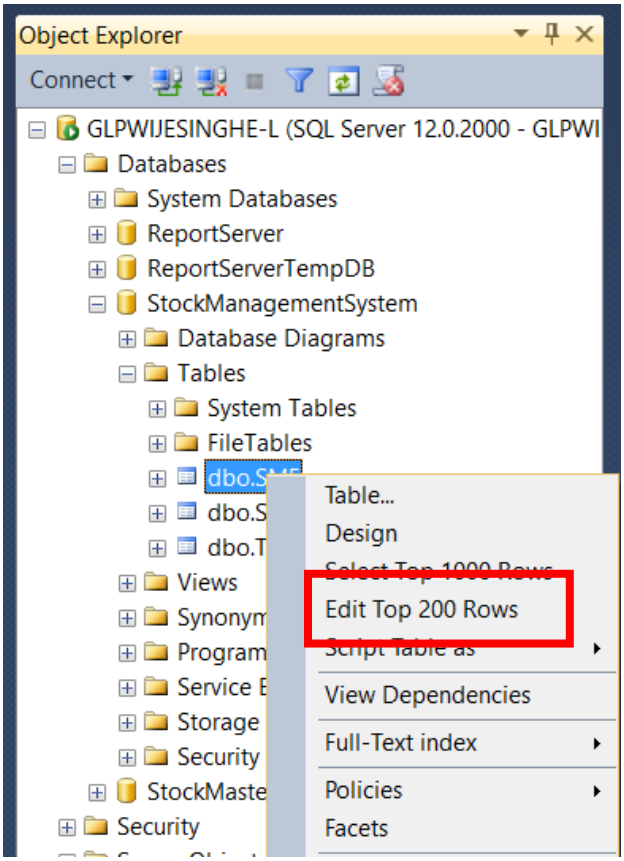


Figure 24

Now you can edit your table.

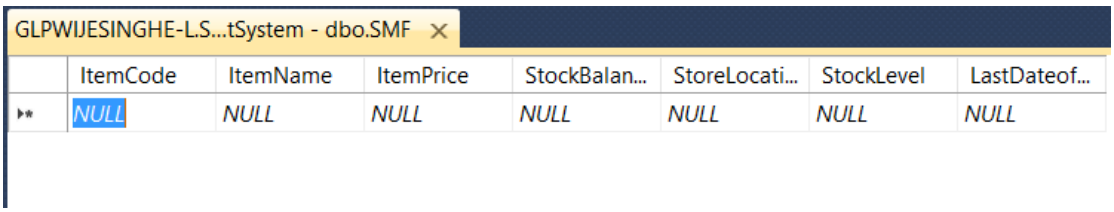


Figure 25

Populate the database with suitable test data.

1. SMF

GLPWIJESINGHE-L.S...tSystem - dbo.SMF

	ItemCode	ItemName	ItemPrice	StockBalan...	StoreLocati...	StockLevel	LastDateof...
	1001	Rice Cooker...	20000.0000	15	SR1	M	2017-07-16
	1002	Gas Cooker-...	25000.0000	20	SR1	M	2017-07-15
	1003	Microwave ...	45000.0000	10	SR1	L	2017-06-30
	1004	Acer Laptop	90000.0000	5	SR2	L	2017-06-25
	1005	LED TV 24"	45000.0000	50	SR3	H	2017-05-17
▸*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 26

2. TRF

GLPWIJESINGHE-L.S...tSystem - dbo.TRF

	ItemCode	ItemName	Transaction...	Quantity	NewStockB...	LastDateof...
	1005	LED TV 24"	Issued OUT	2	50	2017-05-17
	1002	Gas Cooker-...	Issued IN	5	20	2017-07-15
	1004	Acer Laptop	Return OUT	2	5	2017-06-25
▸*	NULL	NULL	NULL	NULL	NULL	NULL

Figure 27

3. SUPREG

GLPWIJESINGHE-L....stem - dbo.SUPREG

	SupplierID	Name	Address	Email	TelM	TelF
	9001	D. S. W. Liya...	No 1,Templ...	makalan@g...	771598546	912233445
	9002	S. R. Nawar...	No. 05, Jail ...	snawa@gm...	773692587	912245671
	9003	D. S. Geega...	"Keshara", A...	dsgee@gm...	714455669	913938028
▸*	NULL	NULL	NULL	NULL	NULL	NULL

Figure 28

Task 3

3.1 The designs of all the required algorithms are required as part of this task. You can use suitable design techniques (Class diagrams, Activity diagrams, Use case diagrams etc.)

3.2 Design necessary user interfaces.

3.1 The designs of all the required algorithms are required as part of this task. You can use suitable design techniques (Class diagrams, Activity diagrams, Use case diagrams etc.)

Algorithms

In mathematics and computer science, an algorithm is a self-contained sequence of actions to be performed. Algorithms can perform calculation, data processing and automated reasoning tasks.

An algorithm is an effective method that can be expressed within a finite amount of space and time and in a well-defined formal language for calculating a function. Starting from an initial state and initial input (perhaps empty), the instructions describe a computation that, when executed, proceeds through a finite number of well-defined successive states, eventually producing "output" and terminating at a final ending state. The transition from one state to the next is not necessarily deterministic; some algorithms, known as randomized algorithms, incorporate random input.

Steps in developing an algorithm are problem definition, development of a model, specification of algorithm, designing, checking the correctness, analysis, implementing and testing. When designing a problem using design methods, there are 3 control constructions as sequence, selection and repetition. Sequence structure specifies that one program statement after another is to be executed in an order. Selection structure shows a choice among 2 actions based on a condition. Repetition structure shows that process is to be rapidly performed while a given condition is true. In designing most of the programs variables should be used. Variable is a temporary memory location which can be used to share data values when you are running a specific computer program. When declaring variables, data types are assigned to them. Such as Integers, real, double, string, Varchar etc.

Class Diagrams

Class diagrams are the most used Unified Modeling Language (UML) diagram type which is used in object oriented software engineering. This concept is mainly focused with object and real world concept. Anything that exists in the real world which can be uniquely identified is an object. It shows the classes in a system, attributes and operations of each class and relationship between each class.

Class – An object is a category of a class or an instance of a class. It is a mechanism of binding data members and associated methods in a single unit. A class consist 3 parts as class name, class attributes and class operations.

Attributes – A characteristic or a property of a class. It describes the range of values the property may hold in objects of class. Every object of the class has a specific value of every attribute.

Operations – The behaviors or services that the object carries out.

Relationships – This allows us to indicate how many objects of one class relate to one object of another class. A class may be involved in one or more relationships with other classes. It can be one of the following types.

- Association-This is displayed as a solid line connecting 2 classes.

Figure 29

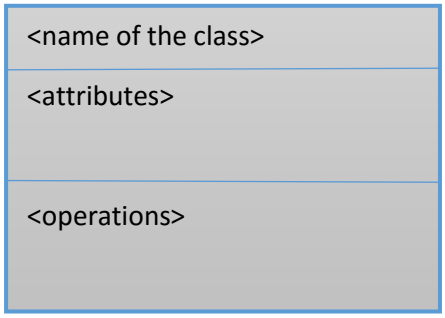
- Aggregation-This is used to illustrate the whole part relationship between 2 classes where one class is a part of the other class.

Figure 30

- Composition-This is also a form of aggregation in which the part classes use to make up the whole class but cannot exist on their own.

Figure 31

Notation use to draw class diagrams



+ denotes public attributes/operations

-denotes private attributes/operations

#denotes protected attributes/operations

Figure 32

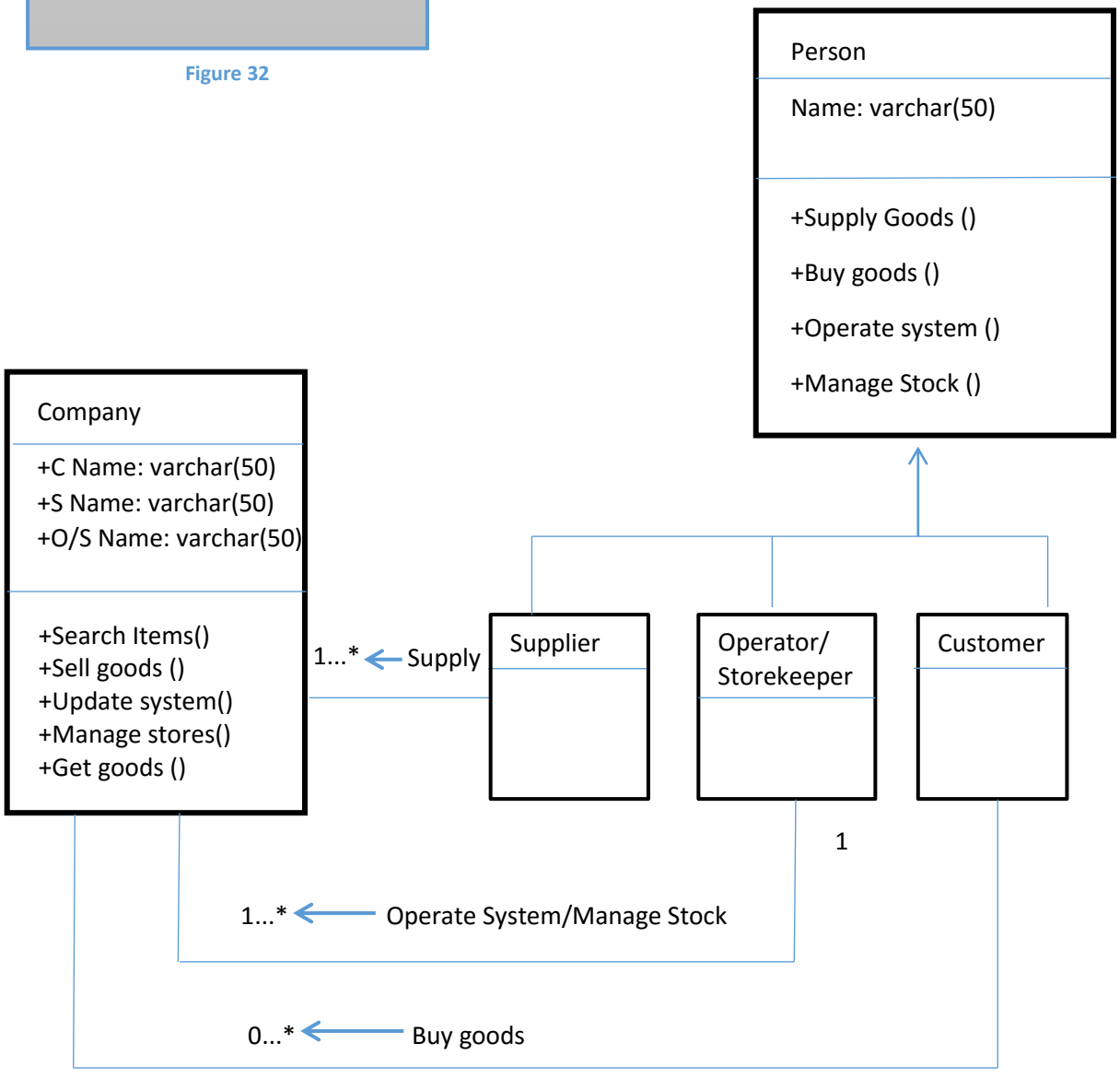


Figure 33

Activity Diagrams

Activity diagrams represent workflows in a graphical way. Activity diagrams show the flow of control between activities. These diagrams are useful in showing work flow connections and describing behavior that has a lot of parallel processing.

Control Elements

- Initial state 

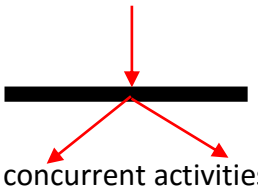
This represents the beginning of a process or work flow.

- Action state 

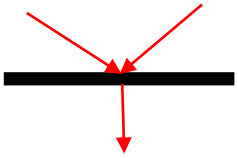
This indicates the activities that makeup a modeled process.

- Final state 

This represents the end of a process or work flow.

- Transition(fork) 

This splits a single activity flow into two concurrent activities.

- Transition(join) 

This combines two concurrent activities and reintroduces them to a new flow where only one activity occurs at a time.

- Decision 

This represents the branching or merging of various flows with the symbol acting as a frame or container.

- Control flow

This shows the flow of control from one action to the next. An incoming arrow starts a step of an activity and once the activity is completed, the flow continues with the outgoing arrow.

Actions

An action represents a single step of functionality in an activity. Actions have incoming and outgoing activity edges that specify the flow of control and data to and from other activity nodes. Structured activity, accept actions, invocation actions, link actions, object actions, structural feature actions are some examples for actions.

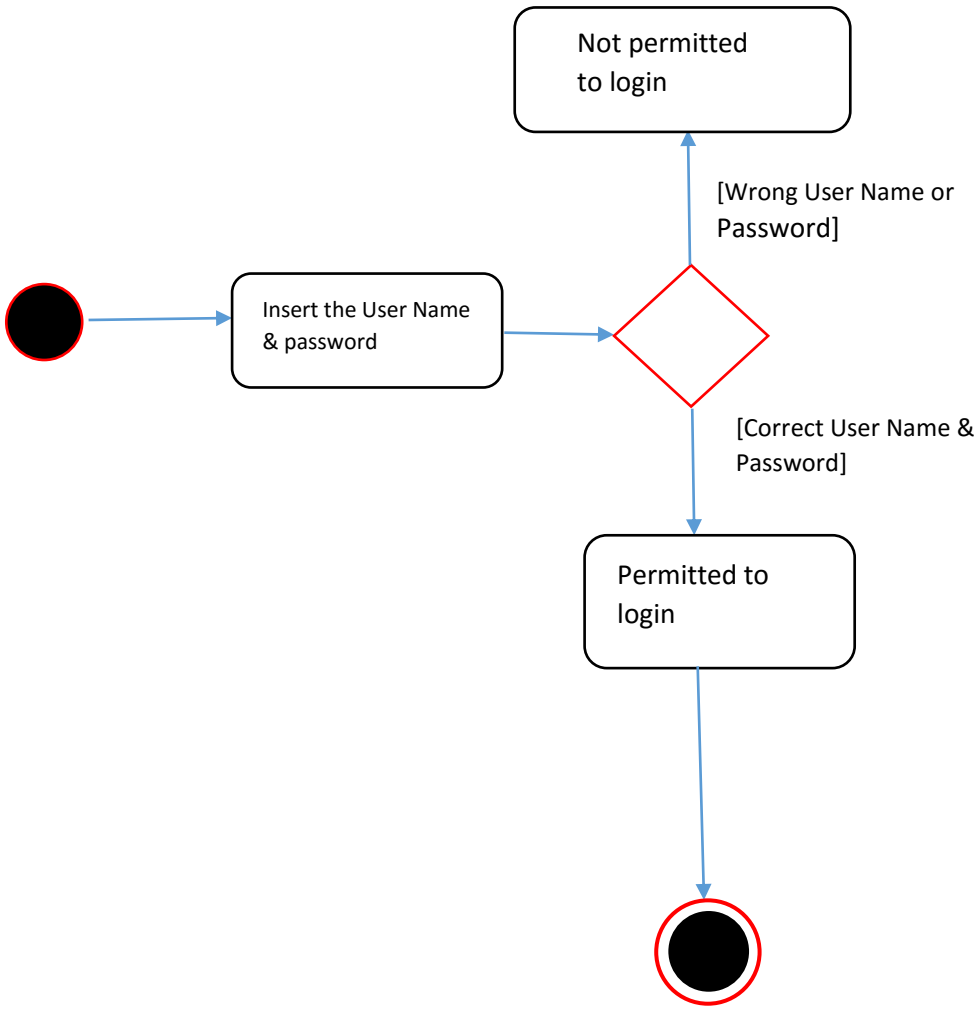


Figure 34

Use Case Diagrams

Use case diagrams are the starting point when designing a new system using the Unified Modeling Language (UML). Use case diagrams give a graphic overview of the actors involved in a system, different functions needed by those actors and how these different functions are interacted.

A use case diagram includes users, use cases and the relationships between the users and use cases within a system and possibly one or more sub systems. There are 4 basic components in a use case diagram as system, actors, use cases and relationships.

- System – A system is something that performs functions
- Actors – An actor is used to represent something/someone that uses the system. An actor can be a person or another system.
- Use cases – Use cases are the actions that a user takes on a system. A use case is illustrated by an oval with a name inside.
- Relationships – A relationship will display actions that are done by each actor. One actor can connect with many use cases. One user can connect with many actors.

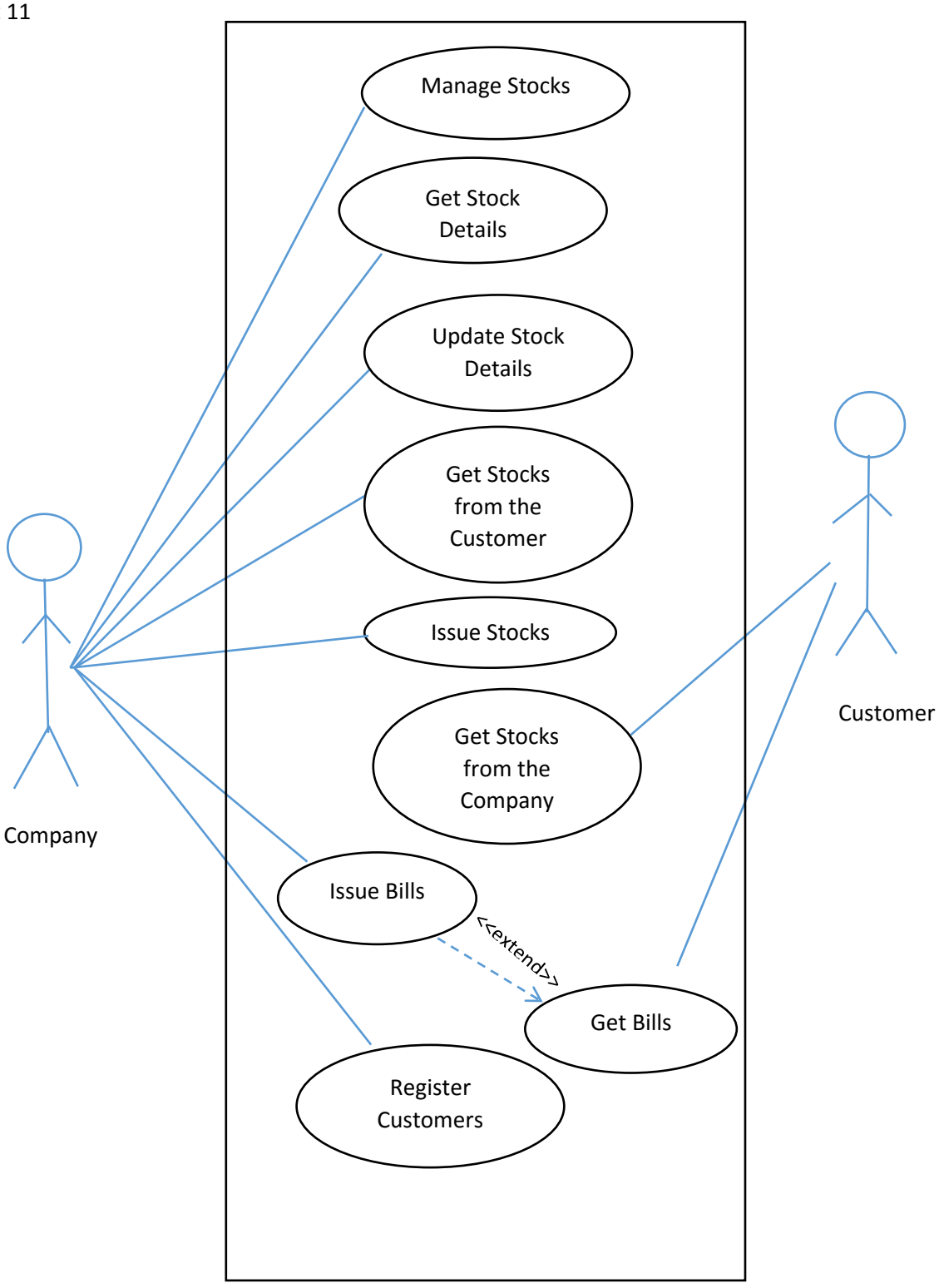

Manage Stocks
Get Stock Details
Update Stock Details
Get Stocks from the Customer
Issue Stocks
Get Stocks from the Company
Issue Bills
<<extend>>
Get Bills
Register Customers
Company
Customer

Figure 35

Design necessary user interfaces

Login Form

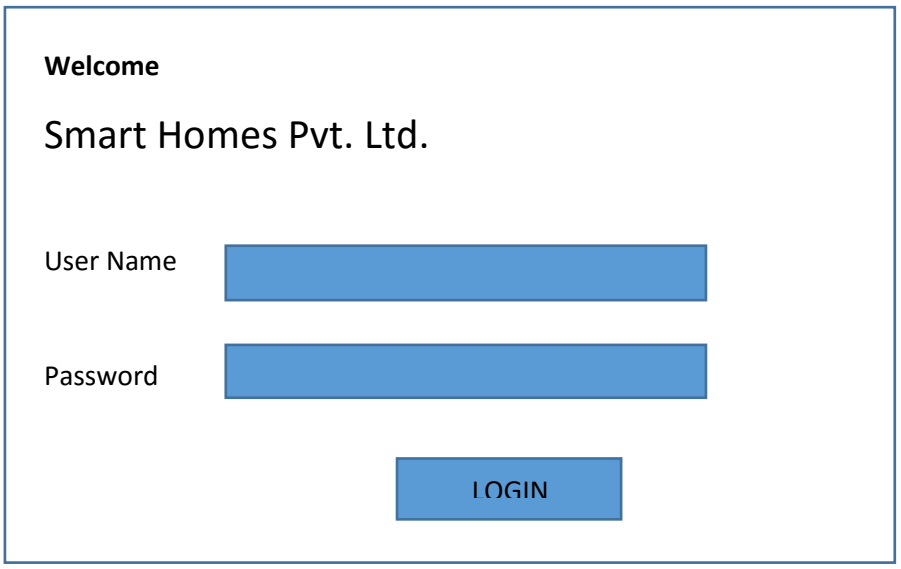


Figure 36

MDI Form

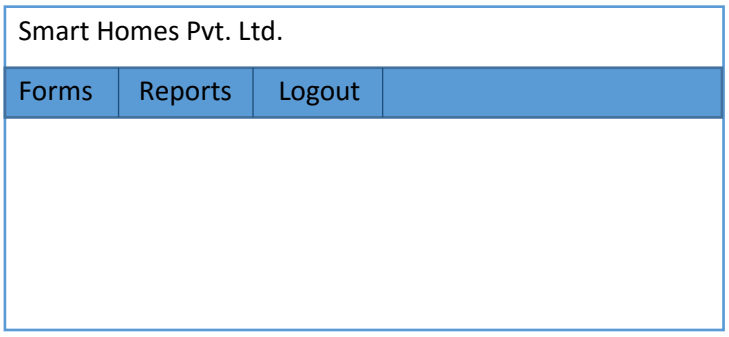


Figure 37

Menu Strip Item- Forms

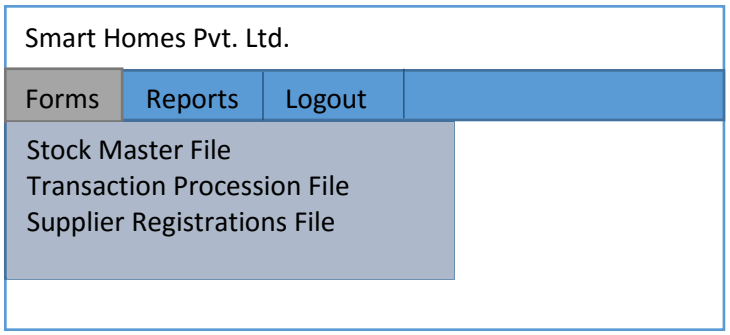


Figure 38

Menu Strip Item-Reports

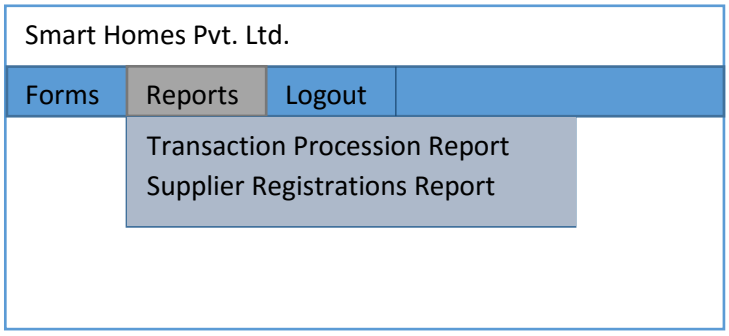


Figure 39

Stock Master File

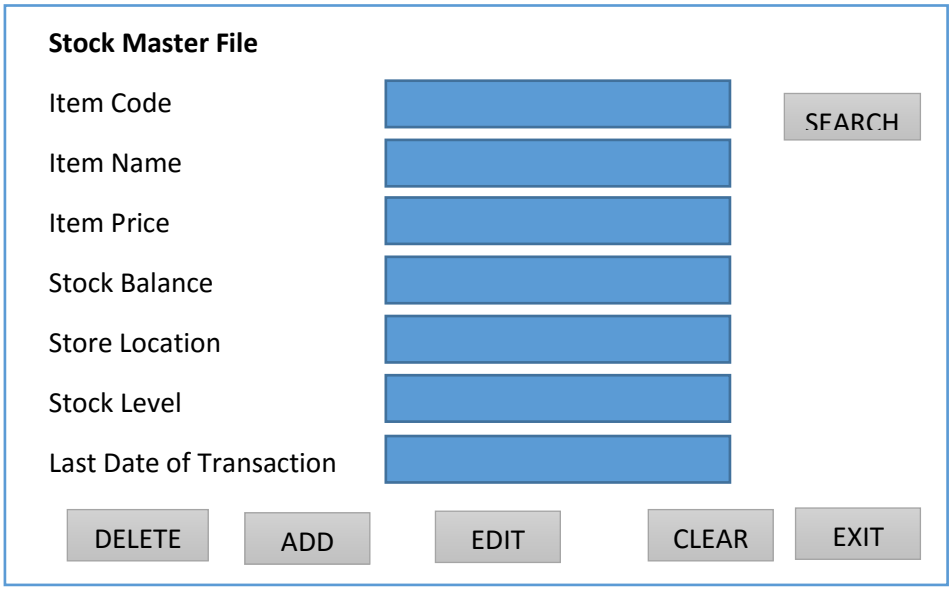


Figure 40

Transaction Procession File

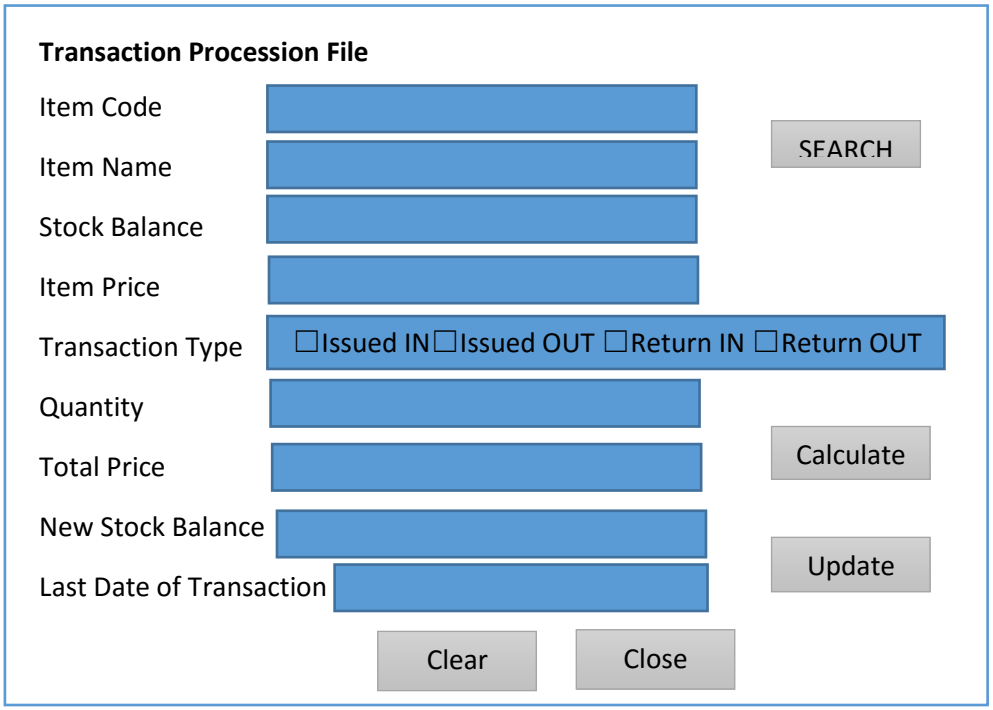


Figure 41

Supplier Registrations File

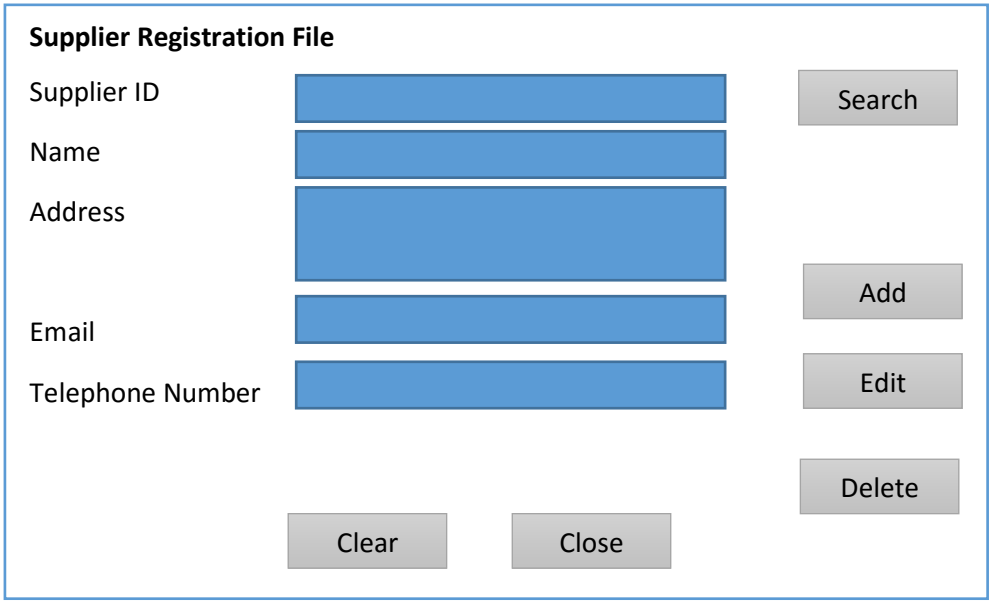


Figure 42

Task 4

4.1 Using VB programming language implement the stock management system designed in Task 3.2

Stock Master File

Form1.vb [Design]

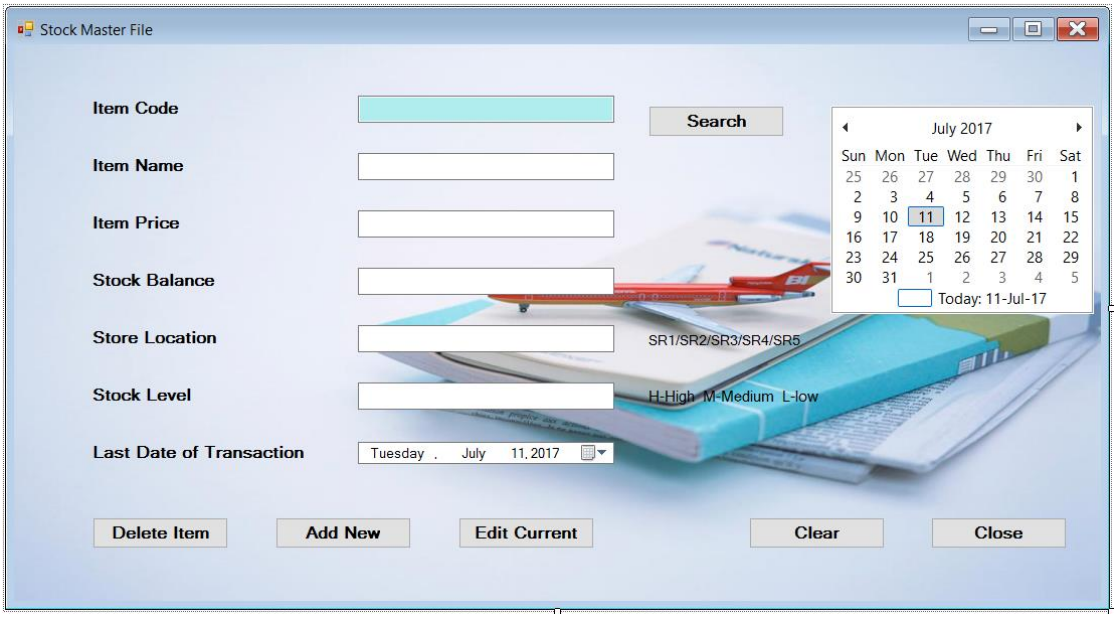


Figure 43

Property Window

Control Name	Name Property	Text Property
Label1	Label1	Item Code
Label2	Label2	Item Name
Label3	Label3	Item Price
Label5	Label5	Stock Balance
Label6	Label6	Stock Level
Label7	Label7	SR1/SR2/SR3/SR4/SR5
Label8	Label8	Store Location
Label9	Label9	Last Date of Transaction
Label10	Label10	H-High M-Medium L-low
TextBox1	txtItCode	
TextBox2	txtItName	
TextBox3	txtItPrice	
TextBox4	txtStockBal	
TextBox5	txtStoreLoc	
TextBox6	txtStockLevel	
TextBox7	txtLDOT	
Button1	btnclose	Close
Button2	btnclr	Clear
Button3	btndel	Delete Item
Button4	btnedit	Edit Current

Button5	btnsearch	Search
Button6	btnadd	Add New

Table 5

Form1.vb [Code]

```
Public Class Form1
    Private con As New System.Data.SqlClient.SqlConnection
    Private com As New System.Data.SqlClient.SqlCommand
    Private adp As New System.Data.SqlClient.SqlDataAdapter
    Private DS As New DataSet

    Private Sub Form1_load(sender As System.Object, e As System.EventArgs)
Handles MyBase.Load
        Try
            con.ConnectionString = "Data Source=GLPWIJESINGHE-L;Initial
Catalog=StockManagementSystem;Integrated Security=True"
            txtItCode.Select()
            btnedit.Enabled = False
            btnclr.Enabled = False
            btndel.Enabled = False
        Catch ex As Exception
            MsgBox(ex.Message, MsgBoxStyle.Exclamation)
        End Try
    End Sub

    Private Sub clear()
        txtItCode.Clear()
        txtItName.Clear()
        txtItPrice.Clear()
        txtStockBal.Clear()
        txtStoreLoc.Clear()
        txtStockLevel.Clear()
        txtLDOT.Clear()
        txtItCode.Focus()
    End Sub

    Private Sub btnclr_Click(sender As System.Object, e As System.EventArgs)
Handles btnclr.Click
        Dim MSG1 As Integer
        MSG1 = MsgBox("Do you want to clear the form?", MsgBoxStyle.YesNo +
MsgBoxStyle.Question, "Clear")
        If MSG1 = vbYes Then
            clear()
            txtItCode.Focus()
        Else
            txtItCode.Focus()
        End If
    End Sub

    Private Sub btnclose_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnclose.Click
        Dim MSG2 As Integer
        MSG2 = MsgBox("Do you want to close the form?", MsgBoxStyle.YesNo +
MsgBoxStyle.Question, "Close")
```

```
        If MSG2 = vbYes Then
            Me.Close()
        Else
            txtItCode.Focus()
        End If
    End Sub

    Private Sub btnadd_Click(sender As System.Object, e As System.EventArgs)
Handles btnadd.Click
        Try
            con.Open()
            com.Connection = con
            com.CommandText = "insert into SMF
(ItemCode,ItemName,ItemPrice,StockBalance,StoreLocation,StockLevel,LastDateofTr
ansaction)values('" + txtItCode.Text + "','" + txtItName.Text + "','" +
txtItPrice.Text + "','" + txtStockBal.Text + "','" + txtStoreLoc.Text + "','" +
txtStockLevel.Text + "','" + txtLDOT.Text + "') "

            If (MsgBox("Do you want Add New Item to the database?",
MsgBoxStyle.YesNo, "Add New Item") = MsgBoxResult.Yes) Then
                MsgBox("Item Added to the database Successfully!",
MsgBoxStyle.Information, "Item Added")
                com.ExecuteNonQuery()
                clear()
                txtItCode.Focus()
            Else
                clear()
                txtItCode.Focus()
            End If
        Catch ex As Exception
            MsgBox(ex.Message, MsgBoxStyle.Exclamation)
        End Try
        con.Close()
    End Sub

    Private Sub btnsearch_Click(sender As System.Object, e As System.EventArgs)
Handles btnsearch.Click

        Try
            con.Open()
            adp = New System.Data.SqlClient.SqlDataAdapter("Select * from SMF
Where ItemCode='" + txtItCode.Text + "'", con)
            DS.Clear()
            adp.Fill(DS, "StockManagementSystem")
            Dim result As Integer
            result = DS.Tables(0).Rows.Count
            If result = 1 Then

                If MsgBox("Do you want to view record?", MsgBoxStyle.YesNo,
"Record found") = MsgBoxResult.Yes Then
                    txtItName.Text =
DS.Tables("StockManagementSystem").Rows(0).Item(1)
                    txtItPrice.Text =
DS.Tables("StockManagementSystem").Rows(0).Item(2)
                    txtStockBal.Text =
DS.Tables("StockManagementSystem").Rows(0).Item(3)
                    txtStoreLoc.Text =
```

```
DS.Tables("StockManagementSystem").Rows(0).Item(4)
                    txtStockLevel.Text =
DS.Tables("StockManagementSystem").Rows(0).Item(5)
                    txtLDOT.Text =
DS.Tables("StockManagementSystem").Rows(0).Item(6)
                Else
                    txtItCode.Clear()
                    txtItCode.Focus()
                End If
            Else
                MsgBox("Record not found", MsgBoxStyle.Information, "No
Record")
                txtItCode.Clear()
                txtItCode.Focus()
            End If

        Catch ex As Exception
            MsgBox(ex.Message, MsgBoxStyle.Exclamation)
        End Try
        con.Close()
    End Sub

    Private Sub btnedit_Click(sender As System.Object, e As System.EventArgs)
Handles btnedit.Click
        Try
            con.Open()
            com.Connection = con
            com.CommandText = "Update SMF Set ItemName='" + txtItName.Text +
"',ItemPrice='" + txtItPrice.Text + "',StockBalance='" + txtStockBal.Text +
"',StoreLocation='" + txtStoreLoc.Text + "',StockLevel='" + txtStockLevel.Text
+ "',LastDateofTransaction='" + txtLDOT.Text + "' Where ItemCode='" +
txtItCode.Text + "'"

            Dim Msg As Integer
            Msg = MsgBox("Do you really want to update Item?",
MsgBoxStyle.Exclamation + MsgBoxStyle.YesNo, "Update Item")
            If Msg = vbYes Then
                com.ExecuteNonQuery()
                MsgBox("Updated successfully!", MsgBoxStyle.Information,
"Updated.")
                clear()
            Else
                clear()
                txtItCode.Focus()
            End If

        Catch ex As Exception
            MsgBox(ex.Message, MsgBoxStyle.Exclamation)
        End Try
        con.Close()
    End Sub

    Private Sub btndel_Click(sender As System.Object, e As System.EventArgs)
Handles btndel.Click
        Try
            con.Open()
            com.Connection = con
```

```
            com.CommandText = "Delete from SMF Where ItemCode='" +
txtItCode.Text + "'"

            Dim Msg As Integer
            Msg = MsgBox("Do you really want to Delete this item?",
MsgBoxStyle.YesNo + MsgBoxStyle.Critical, "Delete Item")
            If Msg = vbYes Then
                com.ExecuteNonQuery()
                MsgBox("One Item Deleted!", MsgBoxStyle.Information,
"Deleted.")
                clear()
            Else
                clear()
                txtItCode.Focus()
            End If

        Catch ex As Exception
            MsgBox(ex.Message, MsgBoxStyle.Exclamation)
        End Try
        con.Close()
    End Sub

    Private Sub txtItCode_TextChanged(sender As System.Object, e As
System.EventArgs) Handles txtItCode.TextChanged
        btnedit.Enabled = True
        btnclr.Enabled = True
        btndel.Enabled = True
    End Sub

    Private Sub btnsearch_MouseHover(sender As Object, e As System.EventArgs)
Handles btnsearch.MouseHover
        Try
            Dim msg As Integer
            If txtItCode.Text = "" Then
                msg = MsgBox("Item Code cannot be blank.",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Incomplete record")
                If msg = vbOKCancel Then
                    txtItCode.Focus()
                End If
            ElseIf Len(txtItCode.Text) <> 4 Then
                msg = MsgBox("Wrong Item Code length.", MsgBoxStyle.Exclamation
+ MsgBoxStyle.OkCancel, "Invalied record")
                If msg = vbOKCancel Then
                    txtItCode.Focus()
                    clear()
                End If

            ElseIf Not IsNumeric(txtItCode.Text) Then
                msg = MsgBox("Item Code must be numeric.",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Invalied record")
                If msg = vbOKCancel Then
                    txtItCode.Focus()
                    clear()
                End If
            End If
        Catch ex As Exception
            MsgBox(ex.Message, MsgBoxStyle.Exclamation)
```

```
        End Try
    End Sub

    Private Sub btnadd_MouseHover(sender As Object, e As System.EventArgs)
Handles btnadd.MouseHover
        Try
            Dim msg As Integer
            If txtItCode.Text = "" Then
                msg = MsgBox("Item Code cannot be blank.",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Incomplete record")
                If msg = vbOKCancel Then
                    txtItCode.Focus()
                End If

            ElseIf Len(txtItCode.Text) <> 4 Then
                msg = MsgBox("Wrong Item Code length.", MsgBoxStyle.Exclamation
+ MsgBoxStyle.OkCancel, "Invalied record")
                If msg = vbOKCancel Then
                    txtItCode.Focus()
                    txtItCode.Clear()
                End If

            ElseIf Not IsNumeric(txtItCode.Text) Then
                msg = MsgBox("Item Code must be numeric.",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Invalied record")
                If msg = vbOKCancel Then
                    txtItCode.Focus()
                    txtItCode.Clear()
                End If

            ElseIf txtItName.Text = "" Then
                msg = MsgBox("Item Name cannot be blank.",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Incomplete record")
                If msg = vbOKCancel Then
                    txtItName.Focus()
                End If

            ElseIf txtItPrice.Text = "" Then
                msg = MsgBox("Item Price cannot be blank.",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Incomplete record")
                If msg = vbOKCancel Then
                    txtItPrice.Focus()
                End If

            ElseIf txtStockBal.Text = "" Then
                msg = MsgBox("Stock Balance cannot be blank.",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Incomplete record")
                If msg = vbOKCancel Then
                    txtStockBal.Focus()
                End If

            ElseIf txtStoreLoc.Text = "" Then
                msg = MsgBox("Store Location cannot be blank.",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Incomplete record")
                If msg = vbOKCancel Then
                    txtStoreLoc.Focus()
                End If
```

```
            ElseIf txtStockLevel.Text = "" Then
                msg = MsgBox("Stock Level cannot be blank.",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Incomplete record")
                If msg = vbOKCancel Then
                    txtStockLevel.Focus()
                End If

            ElseIf txtLDOT.Text = "" Then
                msg = MsgBox("Last Date of Transaction cannot be blank.",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Incomplete record")
                If msg = vbOKCancel Then
                    txtLDOT.Focus()
                End If
            End If

            con.Open()
            adp = New System.Data.SqlClient.SqlDataAdapter("Select * from SMF
Where ItemCode='" + txtItCode.Text + "'", con)
            DS.Clear()
            adp.Fill(DS, "StockManagementSystem")
            Dim result As Integer
            result = DS.Tables(0).Rows.Count
            If result = 1 Then
                Dim Mssg As Integer
                Mssg = MsgBox("A record exists with reference to the Item Code
you entered.Do you really want to overwrite record?", MsgBoxStyle.Exclamation +
MsgBoxStyle.YesNo, "Record Exists!")
                If Mssg = vbYes Then
                    com.Connection = con
                    com.CommandText = "insert into SMF
(ItemCode,ItemName,ItemPrice,StockBalance,StoreLocation,StockLevel,LastDateofTr
ansaction)values('" + txtItCode.Text + "','" + txtItName.Text + "','" +
txtItPrice.Text + "','" + txtStockBal.Text + "','" + txtStoreLoc.Text + "','" +
txtStockLevel.Text + "','" + txtLDOT.Text + "') "
                    com.ExecuteNonQuery()
                    MsgBox("Updated successfully!", MsgBoxStyle.Information,
"Updated.")
                    clear()
                Else
                    txtItCode.Clear()
                    txtItCode.Focus()
                End If
            End If
            con.Close()

        Catch ex As Exception
            MsgBox(ex.Message, MsgBoxStyle.Exclamation)
        End Try
    End Sub

    Private Sub btnedit_MouseHover(sender As Object, e As System.EventArgs)
Handles btnedit.MouseHover
        Dim msg As Integer
        If txtItCode.Text = "" Then
            msg = MsgBox("Item Code cannot be blank.", MsgBoxStyle.Exclamation
+ MsgBoxStyle.OkCancel, "Incomplete record")
```

```
            If msg = vbOKCancel Then
                txtItCode.Focus()
            End If
        ElseIf Len(txtItCode.Text) <> 4 Then
            msg = MsgBox("Wrong Item Code length.", MsgBoxStyle.Exclamation +
MsgBoxStyle.OkCancel, "Invalied record")
            If msg = vbOKCancel Then
                txtItCode.Focus()
                txtItCode.Clear()
            End If
        ElseIf Not IsNumeric(txtItCode.Text) Then
            msg = MsgBox("Item Code must be numeric.", MsgBoxStyle.Exclamation
+ MsgBoxStyle.OkCancel, "Invalied record")
            If msg = vbOKCancel Then
                txtItCode.Focus()
                txtItCode.Clear()
            End If
        ElseIf txtItName.Text = "" Then
            msg = MsgBox("Item Name cannot be blank.", MsgBoxStyle.Exclamation
+ MsgBoxStyle.OkCancel, "Incomplete record")
            If msg = vbOKCancel Then
                txtItName.Focus()
            End If
        ElseIf txtItPrice.Text = "" Then
            msg = MsgBox("Item Price cannot be blank.", MsgBoxStyle.Exclamation
+ MsgBoxStyle.OkCancel, "Incomplete record")
            If msg = vbOKCancel Then
                txtItPrice.Focus()
            End If
        ElseIf txtStockBal.Text = "" Then
            msg = MsgBox("Stock Balance cannot be blank.",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Incomplete record")
            If msg = vbOKCancel Then
                txtStockBal.Focus()
            End If
        ElseIf txtStoreLoc.Text = "" Then
            msg = MsgBox("Store Location cannot be blank.",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Incomplete record")
            If msg = vbOKCancel Then
                txtStoreLoc.Focus()
            End If
        ElseIf txtStockLevel.Text = "" Then
            msg = MsgBox("Stock Level cannot be blank.",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Incomplete record")
            If msg = vbOKCancel Then
                txtStockLevel.Focus()
            End If
        ElseIf txtLDOT.Text = "" Then
            msg = MsgBox("Last Date of Transaction cannot be blank.",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Incomplete record")
            If msg = vbOKCancel Then
                txtLDOT.Focus()
            End If
        End If
    End Sub
End Class
```

Transaction Procession File

Form2.vb [Design]

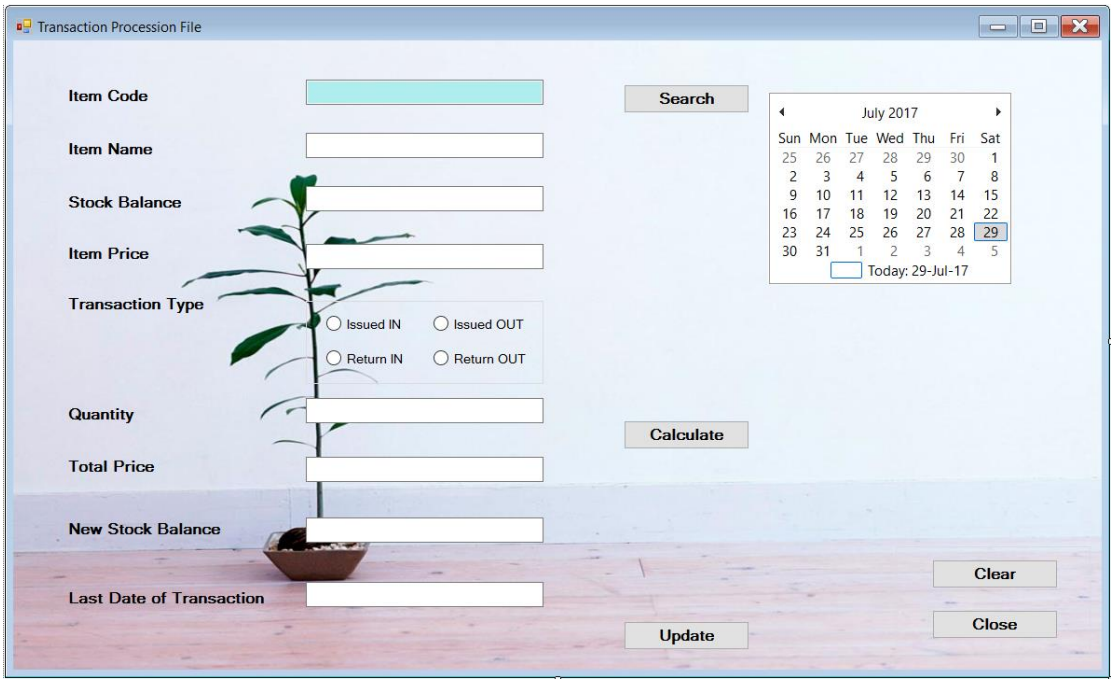


Figure 44

Property Window

Control Name	Name Property	Text Property
Label1	Label1	Item Code
Label2	Label2	Item Name
Label3	Label3	Stock Balance
Label8	Label8	Item Price
Label4	Label4	Transaction Type
Label5	Label5	Quantity
Label9	Label9	Total Price
Label6	Label6	New Stock Balance
Label7	Label7	Label Date of Transaction
TextBox1	txtItCode	
TextBox2	txtItName	
TextBox3	txtStockBal	
TextBox4	txtItPrice	
TextBox5	txtQuan	
TextBox6	txttotprice	
TextBox7	txtNwStockBal	
TextBox8	txtLDOT	
Button1	btnCal	Calculate
Button2	btnclr	Clear

Button3	btnclose	Close
Button4	btnup	Update
Button5	btnsearch	Search
RadioButton1	IssuedOut	Issued OUT
RadioButton2	IssuedIn	Issued IN
RadioButton3	Returnout	Return OUT
RadioButton4	ReturnIn	Return IN
GroupBox1	GroupBox1	
GroupBox2	GroupBox2	

Table 6

Form2.vb [Code]

```
Public Class Form2
    Private con As New System.Data.SqlClient.SqlConnection
    Private com As New System.Data.SqlClient.SqlCommand
    Private adp As New System.Data.SqlClient.SqlDataAdapter
    Private DS As New DataSet
    Public trtype As String
    Public Stockbal, quan, newstcbal, totprice, ItPrice As Integer


    Private Sub Form2_load(sender As System.Object, e As System.EventArgs)
Handles MyBase.Load
        Try
            con.ConnectionString = "Data Source=GLPWIJESINGHE-L;Initial
Catalog=StockManagementSystem;Integrated Security=True"
            txtItCode.Select()
            btnCal.Enabled = False
            btnup.Enabled = False

        Catch ex As Exception
            MsgBox(ex.Message, MsgBoxStyle.Exclamation)
        End Try
    End Sub

    Private Sub clear()
        txtItCode.Clear()
        txtItName.Clear()
        txtItPrice.Clear()
        txttotprice.Clear()
        txtQuan.Clear()
        txtStockBal.Clear()
        txtLDOT.Clear()
        txtNwStockBal.Clear()
    End Sub

    Private Sub btnclr_Click(sender As System.Object, e As System.EventArgs)
Handles btnclr.Click
        Dim MSG1 As Integer
        MSG1 = MsgBox("Do you want to clear the form?", MsgBoxStyle.YesNo +
MsgBoxStyle.Question, "Clear")
        If MSG1 = vbYes Then
            clear()
            txtItCode.Focus()
```

```
        Else
            txtItCode.Focus()
        End If
    End Sub

    Private Sub btnclose_Click(sender As System.Object, e As System.EventArgs)
Handles btnclose.Click
        Dim MSG2 As Integer
        MSG2 = MsgBox("Do you want to close the form?", MsgBoxStyle.YesNo +
MsgBoxStyle.Question, "Close")
        If MSG2 = vbYes Then
            Me.Close()
        Else
            txtItCode.Focus()
        End If
    End Sub

    Private Sub btnsearch_Click(sender As System.Object, e As System.EventArgs)
Handles btnsearch.Click
        Try
            con.Open()
            adp = New System.Data.SqlClient.SqlDataAdapter("Select * from SMF
Where ItemCode='" + txtItCode.Text + "'", con)
            DS.Clear()
            adp.Fill(DS, "StockManagementSystem")
            Dim result As Integer
            result = DS.Tables(0).Rows.Count
            If result = 1 Then

                If MsgBox("Do you want to view record?", MsgBoxStyle.YesNo,
"Record found") = MsgBoxResult.Yes Then
                    txtItName.Text =
DS.Tables("StockManagementSystem").Rows(0).Item(1)
                    txtStockBal.Text =
DS.Tables("StockManagementSystem").Rows(0).Item(3)
                    txtItPrice.Text =
DS.Tables("StockManagementSystem").Rows(0).Item(2)
                Else
                    txtItCode.Clear()
                    txtItCode.Focus()
                End If
            Else
                MsgBox("Record not found", MsgBoxStyle.Information)
                clear()
                txtItCode.Focus()
            End If

        Catch ex As Exception
            MsgBox(ex.Message, MsgBoxStyle.Exclamation)
        End Try
        con.Close()
    End Sub

    Private Sub IssuedIn_CheckedChanged(sender As System.Object, e As
System.EventArgs) Handles IssuedIn.CheckedChanged
        trtype = IssuedIn.Text
    End Sub
```

```
    Private Sub IssuedOut_CheckedChanged(sender As System.Object, e As
System.EventArgs) Handles IssuedOut.CheckedChanged
        trtype = IssuedOut.Text
    End Sub

    Private Sub ReturnIn_CheckedChanged(sender As System.Object, e As
System.EventArgs) Handles ReturnIn.CheckedChanged
        trtype = ReturnIn.Text
    End Sub

    Private Sub Returnout_CheckedChanged(sender As System.Object, e As
System.EventArgs) Handles Returnout.CheckedChanged
        trtype = Returnout.Text
    End Sub

    Private Sub txtQuan_TextChanged(sender As System.Object, e As
System.EventArgs) Handles txtQuan.TextChanged
        btnCal.Enabled = True
    End Sub

    Private Sub btnCal_Click(sender As System.Object, e As System.EventArgs)
Handles btnCal.Click
        Stockbal = txtStockBal.Text
        quan = txtQuan.Text
        ItPrice = txtItPrice.Text
        If IssuedIn.Checked Then
            newstcbal = Stockbal + quan
        ElseIf IssuedOut.Checked Then
            newstcbal = Stockbal - quan
        ElseIf ReturnIn.Checked Then
            newstcbal = Stockbal + quan
        ElseIf Returnout.Checked Then
            newstcbal = Stockbal - quan
        End If
        txtNwStockBal.Text = newstcbal
        totprice = quan * ItPrice
        txttotprice.Text = totprice
    End Sub

    Private Sub txtLDOT_TextChanged(sender As System.Object, e As
System.EventArgs) Handles txtLDOT.TextChanged
        btnup.Enabled = True
    End Sub

    Private Sub btnup_Click(sender As System.Object, e As System.EventArgs)
Handles btnup.Click
        'updating SMF
        Try
            con.Open()
            com.Connection = con
            com.CommandText = "Update SMF Set StockBalance='" +
txtNwStockBal.Text + "',LastDateofTransaction='" + txtLDOT.Text + "' Where
ItemCode='" + txtItCode.Text + "'"
            Dim Msg As Integer
            Msg = MsgBox("Do you really want to update SMF?",
MsgBoxStyle.Exclamation + MsgBoxStyle.YesNo, "Update Item")
```

```
            If Msg = vbYes Then
                com.ExecuteNonQuery()
                MsgBox("Updated successfully", MsgBoxStyle.Information,
"Updated")
            Else
                clear()
                txtItCode.Focus()
            End If
        Catch ex As Exception
            MsgBox(ex.Message, MsgBoxStyle.Exclamation)
        End Try
        con.Close()

        'UPDATING TRF
        Try
            con.Open()
            com.Connection = con
            com.CommandText = "Insert into TRF (ItemCode, ItemName, ItemPrice,
TransactionType, Quantity, TotalPrice, NewStockBalance, LastDateofTransaction)
values('" + txtItCode.Text + "','" + txtItName.Text + "','" + txtItPrice.Text +
"','" + trtype + "','" + txtQuan.Text + "','" + txttotprice.Text + "','" +
txtNwStockBal.Text + "','" + txtLDOT.Text + "')"
            Dim Msg As Integer
            Msg = MsgBox("Do you really want to update TRF?",
MsgBoxStyle.Exclamation + MsgBoxStyle.YesNo, "Update Item")
            If Msg = vbYes Then
                com.ExecuteNonQuery()
                MsgBox("Updated successfully", MsgBoxStyle.Information,
"Updated")
                clear()
                txtItCode.Focus()
            Else
                clear()
                txtItCode.Focus()
            End If
        Catch ex As Exception
            MsgBox(ex.Message, MsgBoxStyle.Exclamation)
        End Try
        con.Close()
    End Sub

    Private Sub btnsearch_MouseHover(sender As Object, e As System.EventArgs)
Handles btnsearch.MouseHover
        Try
            Dim msg As Integer
            If txtItCode.Text = "" Then
                msg = MsgBox("Item Code cannot be blank",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Incomplete record")
                If msg = vbOKCancel Then
                    txtItCode.Focus()
                    clear()
                End If
            ElseIf Len(txtItCode.Text) <> 4 Then
                msg = MsgBox("Wrong Item Code length.", MsgBoxStyle.Exclamation
+ MsgBoxStyle.OkCancel, "Invalied record")
                If msg = vbOKCancel Then
                    txtItCode.Focus()
```

```
                    clear()
                End If

            ElseIf Not IsNumeric(txtItCode.Text) Then
                msg = MsgBox("Item Code must be numeric.",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Invalied record")
                If msg = vbOKCancel Then
                    txtItCode.Focus()
                    clear()
                End If
            End If
        Catch ex As Exception
            MsgBox(ex.Message, MsgBoxStyle.Exclamation)
        End Try
    End Sub

End Class
```

Supplier Registration File

Form3.vb [Design]

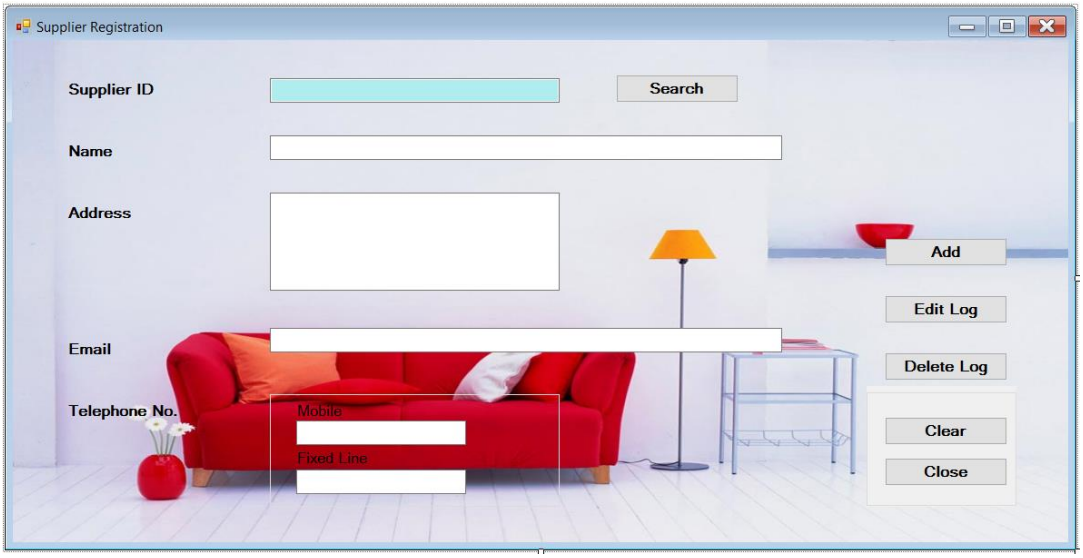


Figure 45

Property Window

Control Name	Name Property	Text Property
Label1	Label1	Supplier ID
Label2	Label2	Name
Label3	Label3	Address
Label4	Label4	Email
Label5	Label5	Telephone No.
Label6	Label6	Mobile
Label7	Label7	Fixed Line
TextBox1	txtsupID	
TextBox2	txtName	
TextBox3	txtadd	
TextBox4	txtEmail	
TextBox5	txtTelM	
TextBox6	txtTelF	
Button1	btnclose	Close
Button2	btnclr	Clear
Button3	btnsearch	Search
Button4	btndel	Delete Log
Button5	btnedit	Edit Log
Button6	btnadd	Add
GroupBox1	GroupBox1	
GroupBox2	GroupBox2	

Table 7

Form3.vb [Code]

```
Public Class Form3
    Private con As New System.Data.SqlClient.SqlConnection
    Private com As New System.Data.SqlClient.SqlCommand
    Private adp As New System.Data.SqlClient.SqlDataAdapter
    Private DS As New DataSet
    Private Sub Form3_Load(sender As System.Object, e As System.EventArgs)
Handles MyBase.Load
        txtsupID.Select()
        btnedit.Enabled = False
        btnclr.Enabled = False
        btndel.Enabled = False
        Try
            con.ConnectionString = "Data Source=GLPWIJESINGHE-L;Initial
Catalog=StockManagementSystem;Integrated Security=True"
        Catch ex As Exception
            MsgBox(ex.Message, MsgBoxStyle.Exclamation)
        End Try
    End Sub

    Private Sub clear()
        txtsupID.Clear()
        txtName.Clear()
        txtadd.Clear()
        txtEmail.Clear()
        txtTelM.Clear()
        txtTelF.Clear()
    End Sub

    Private Sub btnclr_Click(sender As System.Object, e As System.EventArgs)
Handles btnclr.Click
        Dim MSG1 As Integer
        MSG1 = MsgBox("Do you want to clear the form?", MsgBoxStyle.YesNo +
MsgBoxStyle.Question, "Clear")
        If MSG1 = vbYes Then
            clear()
            txtsupID.Focus()
        Else
            txtsupID.Focus()
        End If
    End Sub

    Private Sub btnclose_Click(sender As System.Object, e As System.EventArgs)
Handles btnclose.Click
        Dim MSG2 As Integer
        MSG2 = MsgBox("Do you want to close the form?", MsgBoxStyle.YesNo +
MsgBoxStyle.Question, "Close")
        If MSG2 = vbYes Then
            Me.Close()
        Else
            txtsupID.Focus()
        End If
    End Sub

    Private Sub btnadd_Click(sender As System.Object, e As System.EventArgs)
Handles btnadd.Click
```

```
        Try
            con.Open()
            com.Connection = con
            com.CommandText = "insert into SUPREG(SupplierID, Name, Address,
Email, TelM, TelF)values('" + txtsupID.Text + "','" + txtName.Text + "','" +
txtadd.Text + "','" + txtEmail.Text + "','" + txtTelM.Text + "','" +
txtTelF.Text + "') "
            If (MsgBox("Do you want Add New Item to the database?",
MsgBoxStyle.DefaultButton1 + MsgBoxStyle.YesNo, "Add New Item") =
MsgBoxResult.Yes) Then
                com.ExecuteNonQuery()
                MsgBox("Item added to the database Successfully!",
MsgBoxStyle.Information, "Item Added")
                clear()
                txtsupID.Focus()
            Else
                clear()
                txtsupID.Focus()
            End If
        Catch ex As Exception
            MsgBox(ex.Message, MsgBoxStyle.Exclamation)
        End Try
        con.Close()
    End Sub
    Private Sub btnedit_Click(sender As System.Object, e As System.EventArgs)
Handles btnedit.Click
        Try
            con.Open()
            com.Connection = con
            com.CommandText = "Update SUPREG Set Name='" + txtName.Text + "',
Address='" + txtadd.Text + "', Email='" + txtEmail.Text + "', TelM='" +
txtTelM.Text + "', TelF='" + txtTelF.Text + "' Where SupplierID='" +
txtsupID.Text + "'"
            If (MsgBox("Do you really want to update Item?",
MsgBoxStyle.Exclamation + MsgBoxStyle.YesNo, "Update Item") = MsgBoxResult.Yes)
Then
                com.ExecuteNonQuery()
                MsgBox("Updated successfully!", MsgBoxStyle.Information,
"Updated")
                clear()
                txtsupID.Focus()
            Else
                txtsupID.Focus()
            End If
        Catch ex As Exception
            MsgBox(ex.Message, MsgBoxStyle.Exclamation)
        End Try
        con.Close()
    End Sub
    Private Sub btndel_Click(sender As System.Object, e As System.EventArgs)
Handles btndel.Click
        Try
            con.Open()
            com.Connection = con
            com.CommandText = "Delete from SUPREG Where SupplierID='" +
txtsupID.Text + "'"
            If (MsgBox("Do you really want to Delete this item?",
```

```
MsgBoxStyle.YesNo + MsgBoxStyle.Critical, "Delete Item") = MsgBoxResult.Yes)
Then
                com.ExecuteNonQuery()
                MsgBox("One Item Deleted!", MsgBoxStyle.Information, "Deleted")
                clear()
                txtsupID.Focus()
            Else
                txtsupID.Focus()
            End If
        Catch ex As Exception
            MsgBox(ex.Message, MsgBoxStyle.Exclamation)
        End Try
        con.Close()
    End Sub
    Private Sub btnsearch_Click(sender As System.Object, e As System.EventArgs)
Handles btnsearch.Click
        Try
            btnclr.Enabled = True
            btnedit.Enabled = True
            btndel.Enabled = True
            con.Open()
            adp = New System.Data.SqlClient.SqlDataAdapter("Select * from
SUPREG Where SupplierID='" + txtsupID.Text + "'", con)
            DS.Clear()
            adp.Fill(DS, "StockManagementSystem")
            Dim result As Integer
            result = DS.Tables(0).Rows.Count
            If result = 1 Then
                If MsgBox("Do you want to view record?", MsgBoxStyle.YesNo,
"Record found") = MsgBoxResult.Yes Then
                    txtName.Text =
DS.Tables("StockManagementSystem").Rows(0).Item(1)
                    txtadd.Text =
DS.Tables("StockManagementSystem").Rows(0).Item(2)
                    txtEmail.Text =
DS.Tables("StockManagementSystem").Rows(0).Item(3)
                    txtTelM.Text =
DS.Tables("StockManagementSystem").Rows(0).Item(4)
                    txtTelF.Text =
DS.Tables("StockManagementSystem").Rows(0).Item(5)
                Else
                    txtsupID.Clear()
                    txtsupID.Focus()
                End If
            Else
                MsgBox("Record not found.Please recheck the Supplier ID and try
again.", MsgBoxStyle.Information, "Wrong Supplier ID")
                txtsupID.Clear()
                txtsupID.Focus()
            End If
        Catch ex As Exception
            MsgBox(ex.Message, MsgBoxStyle.Exclamation)
        End Try
        con.Close()
    End Sub
    Private Sub btnsearch_MouseHover(sender As Object, e As System.EventArgs)
Handles btnsearch.MouseHover
```

```
        Try
            Dim msg As Integer
            If txtsupID.Text = "" Then
                msg = MsgBox("Supplier ID cannot be blank",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Incomplete record")
                If msg = vbOKCancel Then
                    txtsupID.Focus()
                End If
            ElseIf Len(txtsupID.Text) <> 4 Then
                msg = MsgBox("Wrong Supplier ID length.",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Invalied record")
                If msg = vbOKCancel Then
                    txtsupID.Focus()
                    clear()
                End If
            ElseIf Not IsNumeric(txtsupID.Text) Then
                msg = MsgBox("Supplier ID must be numeric.",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Invalied record")
                If msg = vbOKCancel Then
                    txtsupID.Focus()
                    clear()
                End If
            End If
        Catch ex As Exception
            MsgBox(ex.Message, MsgBoxStyle.Exclamation)
        End Try
    End Sub
    Private Sub btnadd_MouseHover(sender As Object, e As System.EventArgs)
Handles btnadd.MouseHover
        Try

            Dim msg As Integer
            If txtsupID.Text = "" Then
                msg = MsgBox("Supplier ID cannot be blank",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Incomplete record")
                If msg = vbOKCancel Then
                    txtsupID.Focus()
                End If
            ElseIf Len(txtsupID.Text) <> 4 Then
                msg = MsgBox("Wrong Supplier ID length.",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Invalied record")
                If msg = vbOKCancel Then
                    txtsupID.Focus()
                    txtsupID.Clear()
                End If
            ElseIf Not IsNumeric(txtsupID.Text) Then
                msg = MsgBox("Supplier ID must be numeric.",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Invalied record")
                If msg = vbOKCancel Then
                    txtsupID.Focus()
                    txtsupID.Clear()
                End If
            ElseIf txtName.Text = "" Then
                msg = MsgBox("Supplier Name cannot be blank",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Incomplete record")
                If msg = vbOKCancel Then
                    txtName.Focus()
```

```
                End If
            ElseIf txtadd.Text = "" Then
                msg = MsgBox("Supplier Address cannot be blank",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Incomplete record")
                If msg = vbOKCancel Then
                    txtadd.Focus()
                End If
            ElseIf txtEmail.Text = "" Then
                msg = MsgBox("Supplier Email cannot be blank",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Incomplete record")
                If msg = vbOKCancel Then
                    txtEmail.Focus()
                End If
            ElseIf txtTelM.Text = "" And txtTelF.Text = "" Then
                msg = MsgBox("At least one telephone number should be
provided.", MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Incomplete
record")
                If msg = vbOKCancel Then
                    txtTelM.Focus()
                End If

            ElseIf Len(txtTelM.Text) < 9 Or Len(txtTelF.Text) < 9 Then
                msg = MsgBox("Please provide a valied Telephone number",
MsgBoxStyle.Exclamation + MsgBoxStyle.OkCancel, "Invalied record")
                If msg = vbOKCancel Then
                    txtTelM.Focus()
                    txtTelM.Clear()
                    txtTelF.Clear()
                End If
            End If

            con.Open()
            adp = New System.Data.SqlClient.SqlDataAdapter("Select * from
SUPREG Where SupplierID='" + txtsupID.Text + "'", con)
            DS.Clear()
            adp.Fill(DS, "StockManagementSystem")
            Dim result As Integer
            result = DS.Tables(0).Rows.Count
            If result = 1 Then
                Dim Mssg As Integer
                Mssg = MsgBox("A record exists with reference to the Supplier
ID you entered.Do you really want to overwrite record?",
MsgBoxStyle.Exclamation + MsgBoxStyle.YesNo, "Record Exists!")
                If Mssg = vbYes Then
                    com.Connection = con
                    com.CommandText = "insert into SUPREG(SupplierID, Name,
Address, Email, TelM, TelF)values('" + txtsupID.Text + "','" + txtName.Text +
"','" + txtadd.Text + "','" + txtEmail.Text + "','" + txtTelM.Text + "','" +
txtTelF.Text + "') "
                    com.ExecuteNonQuery()
                    MsgBox("Updated successfully!", MsgBoxStyle.Information,
"Updated.")
                    clear()
                Else
                    txtsupID.Clear()
                    txtsupID.Focus()
                End If
```

```
            End If
            con.Close()

        Catch ex As Exception
            MsgBox(ex.Message, MsgBoxStyle.Exclamation)
        End Try
    End Sub

    Private Sub txtsupID_TextChanged(sender As System.Object, e As
System.EventArgs) Handles txtsupID.TextChanged
        btnclr.Enabled = True
    End Sub

    Private Sub GroupBox1_Enter(sender As System.Object, e As System.EventArgs)
Handles GroupBox1.Enter
        btnedit.Enabled = True
        btndel.Enabled = True
    End Sub
End Class
```

Login Form

Form4.vb [Design]

Figure 46

Property Window

Control Name	Name Property	Text Property
Label1	Label2	User Name
Label2	Label3	Password
Label3	Label4	Smart Homes Pvt. Ltd.
TextBox1	txtname	
TextBox2	txtpwrd	
Button1	btnlogin	Login
GroupBox1	GroupBox1	
PictureBox2	PictureBox1	

Table 8

Form4.vb [Code]

```
Public Class Form4

    Private Sub btnlogin_Click(sender As System.Object, e As System.EventArgs)
Handles btnlogin.Click
        If txtname.Text = "admin" And txtpwrd.Text = "11111" Then
            Form5.Show()
            Me.Hide()
            txtname.Clear()
            txtpwrd.Clear()
            txtname.Focus()
        Else
            MsgBox("Sorry, username or password Incorrect!", MsgBoxStyle.OkOnly
+ MsgBoxStyle.Critical, "Invalid")
            txtname.Clear()
            txtpwrd.Clear()
            txtname.Focus()
        End If
    End Sub

    Private Sub Form4_Load(sender As System.Object, e As System.EventArgs)
Handles MyBase.Load
        txtname.Select()
    End Sub
End Class
```

MDI Form

Form5.vb [Design]

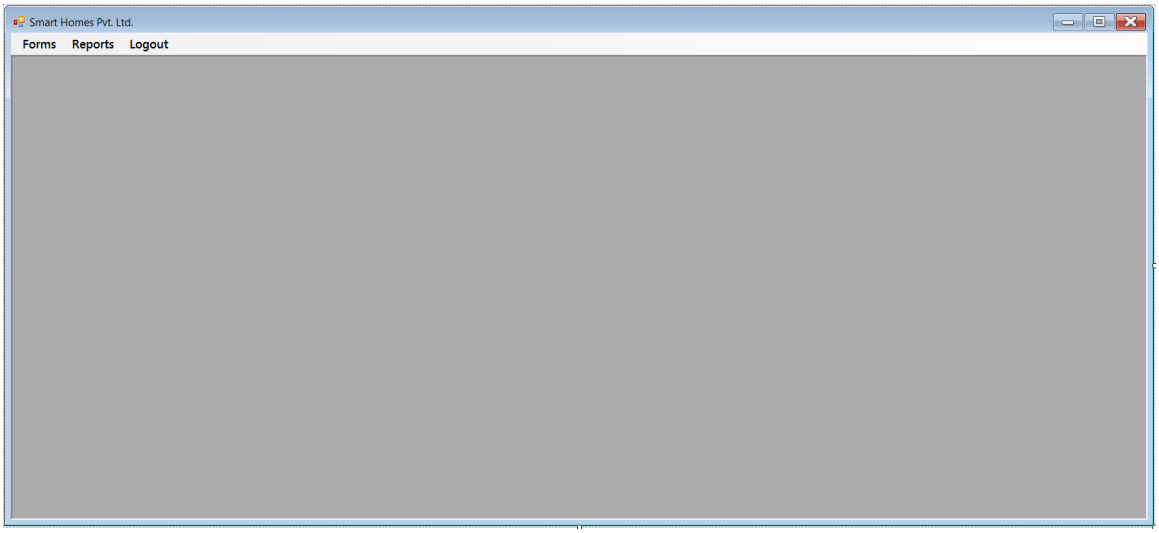


Figure 47

Menu Strip Item- Forms

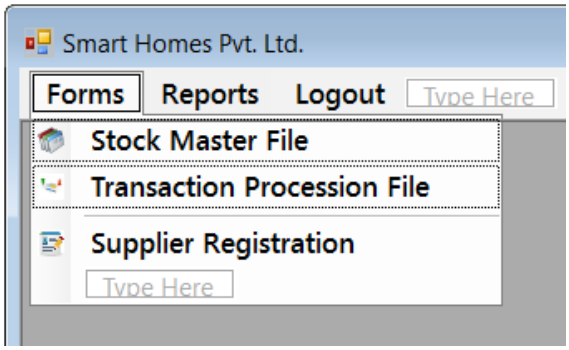


Figure 48

Menu Strip Item-Reports

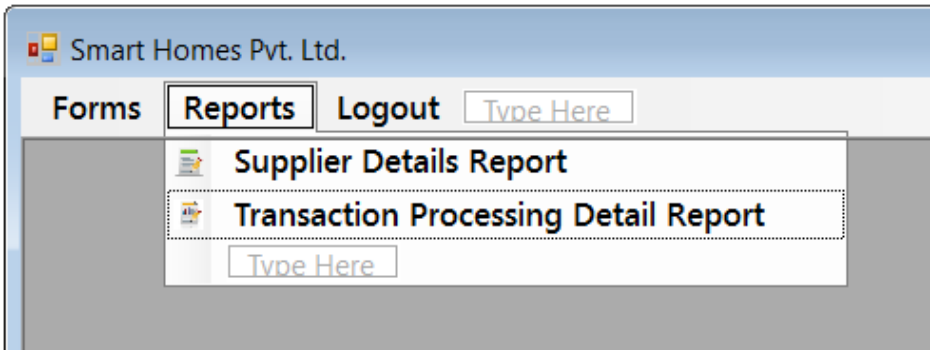


Figure 49

Property Window

Control Name	Name Property	Text Property
MenuStrip1	Menu Strip1	
ToolSripMenueItem1	FormsToolSripMenueItem	Forms
ToolSripMenueItem2	StockMasterFileToolSripMenueItem	Stock Master File
ToolSripMenueItem3	TransactionProcessionFileToolSripMenueItem	Transaction Procession File
ToolSripMenueItem4	SupplierRegistrationFileToolSripMenueItem	Supplier Registration File
ToolSripMenueItem5	ReportsToolSripMenueItem	Reports
ToolSripMenueItem6	TransactionProcessionDetailReportToolSripMenueItem	Transaction Procession Detail Report
ToolSripMenueItem7	SupplierDetailsReportToolSripMenueItem	Supplier Details Report
ToolSripMenueItem8	LogoutToolSripMenueItem	Logout

Table 9

Form5.vb [Code]

```
Public Class Form5
        Private Sub Form5_Load(sender As System.Object, e As System.EventArgs)
Handles MyBase.Load
        Dim ctl As Control
        Dim ctlMDI As MdiClient
        For Each ctl In Me.Controls
            Try
                ctlMDI = CType(ctl, MdiClient)
                ctlMDI.BackColor = Me.BackColor
            Catch exc As InvalidCastException
            End Try
        Next
    End Sub

    Private Sub StockMasterFileToolStripMenuItem_Click(sender As System.Object,
e As System.EventArgs) Handles StockMasterFileToolStripMenuItem.Click
        Form1.MdiParent = Me
        Form1.Show()
    End Sub

    Private Sub TransactionProcessionFileToolStripMenuItem_Click_1(sender As
System.Object, e As System.EventArgs) Handles
TransactionProcessionFileToolStripMenuItem.Click
        Form2.MdiParent = Me
        Form2.Show()
    End Sub
```

```
    Private Sub SupplierRegistrationToolStripMenuItem_Click_1(sender As
System.Object, e As System.EventArgs) Handles
SupplierRegistrationToolStripMenuItem.Click
        Form3.MdiParent = Me
        Form3.Show()
    End Sub

    Private Sub LogoutToolStripMenuItem_Click(sender As System.Object, e As
System.EventArgs) Handles LogoutToolStripMenuItem.Click
        Dim msg As Integer
        msg = MsgBox("Do you really want to logout?", MsgBoxStyle.Critical +
MsgBoxStyle.YesNo, "Logout")
        If msg = vbYes Then
            Form1.Close()
            Form2.Close()
            Form3.Close()
            Form6.Close()
            Form7.Close()
            Me.Close()
            MsgBox("Logged out successfully.Have a nice day!",
MsgBoxStyle.Information, "Logged out!")
            Form4.Show()
        Else
            Form1.Close()
            Form2.Close()
            Form3.Close()
            Form6.Close()
            Form7.Close()
        End If
    End Sub

    Private Sub TransactionProToolStripMenuItem_Click(sender As System.Object,
e As System.EventArgs) Handles TransactionProToolStripMenuItem.Click
        Form6.MdiParent = Me
        Form6.Show()
    End Sub

    Private Sub SupplierDetailReportToolStripMenuItem_Click(sender As
System.Object, e As System.EventArgs) Handles
SupplierDetailReportToolStripMenuItem.Click
        Form7.MdiParent = Me
        Form7.Show()
    End Sub
End Class
```

Transaction Processing Detail Report-Form6.vb [Design]

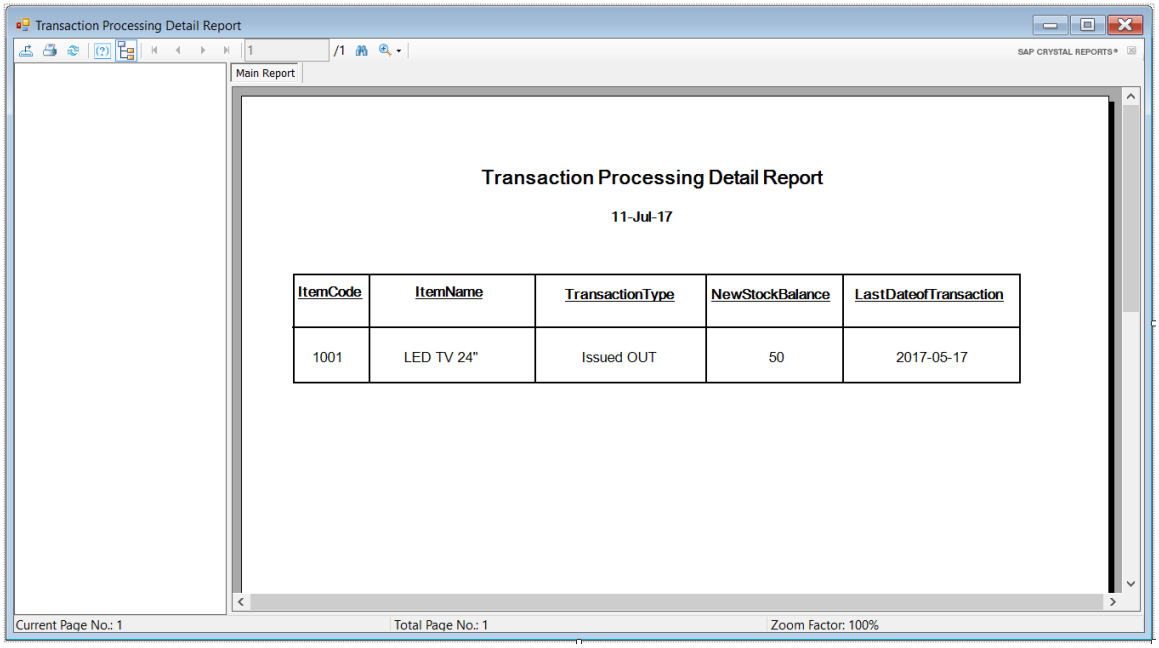


Figure 50

Supplier Details Report- Form7.vb [Design]

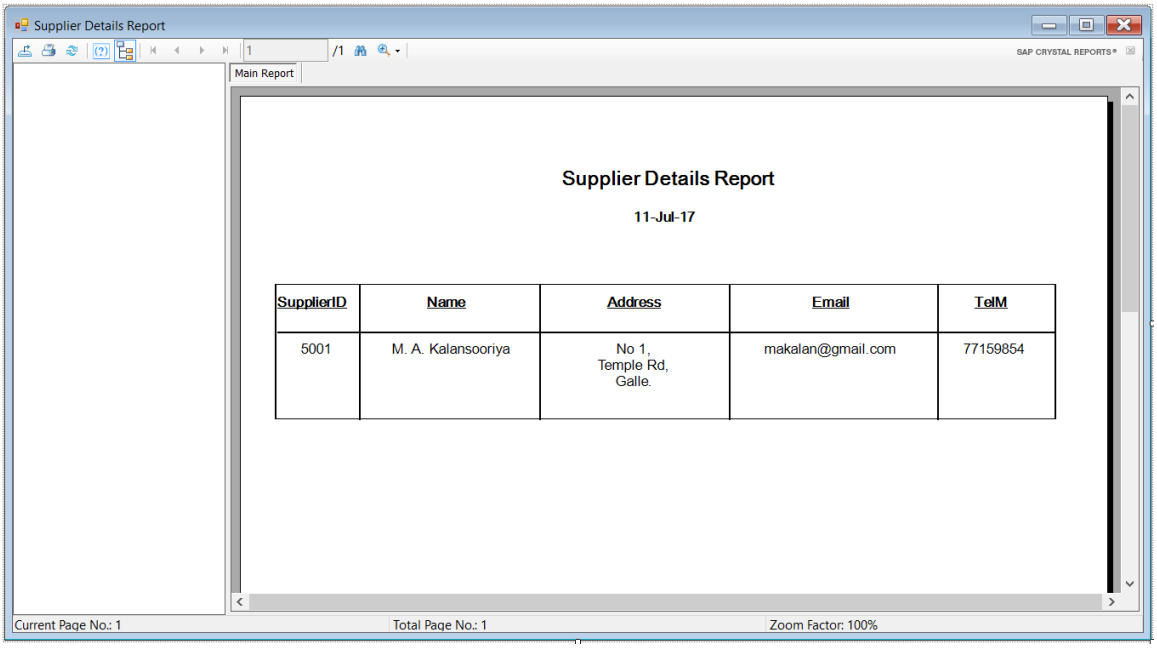


Figure 51

Crystal Reports

TRF.rpt

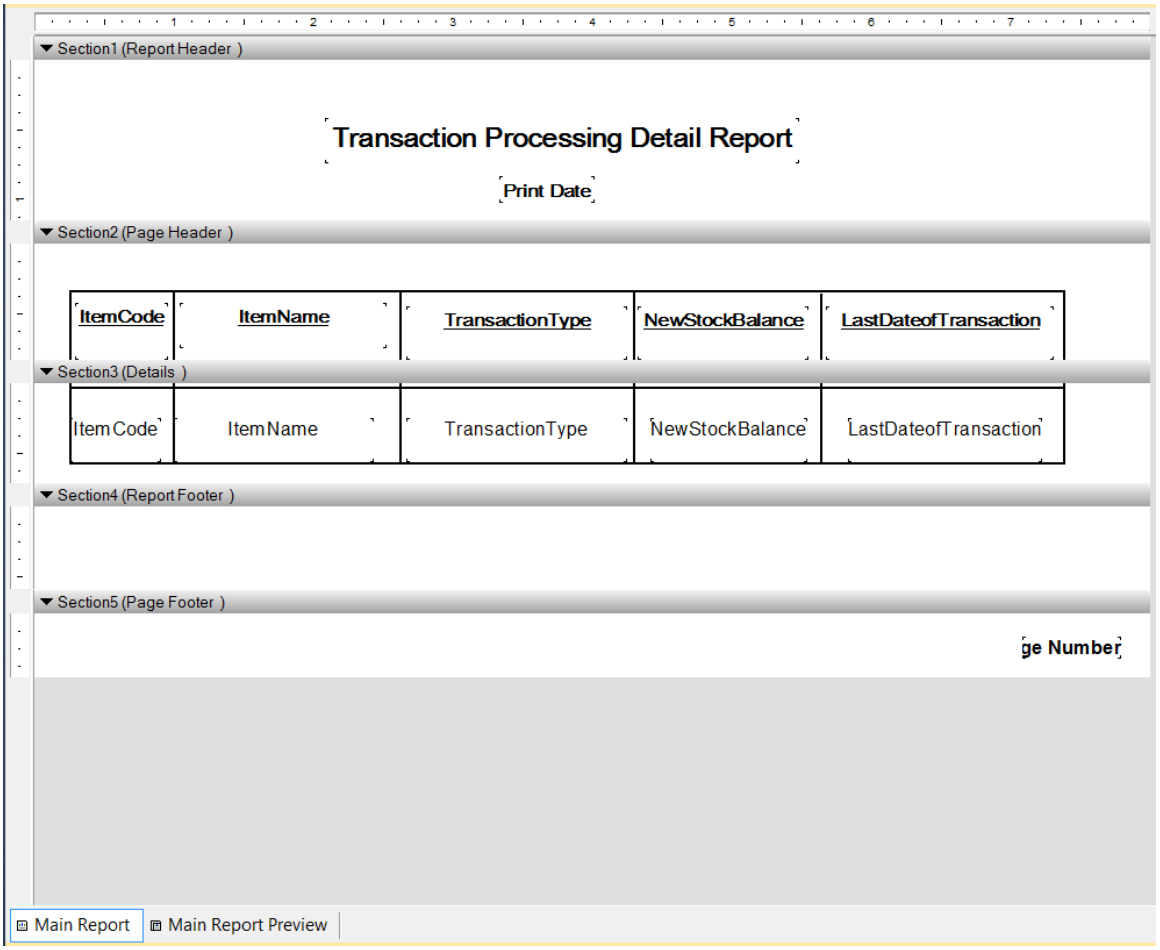


Figure 52

SUPREG.rpt

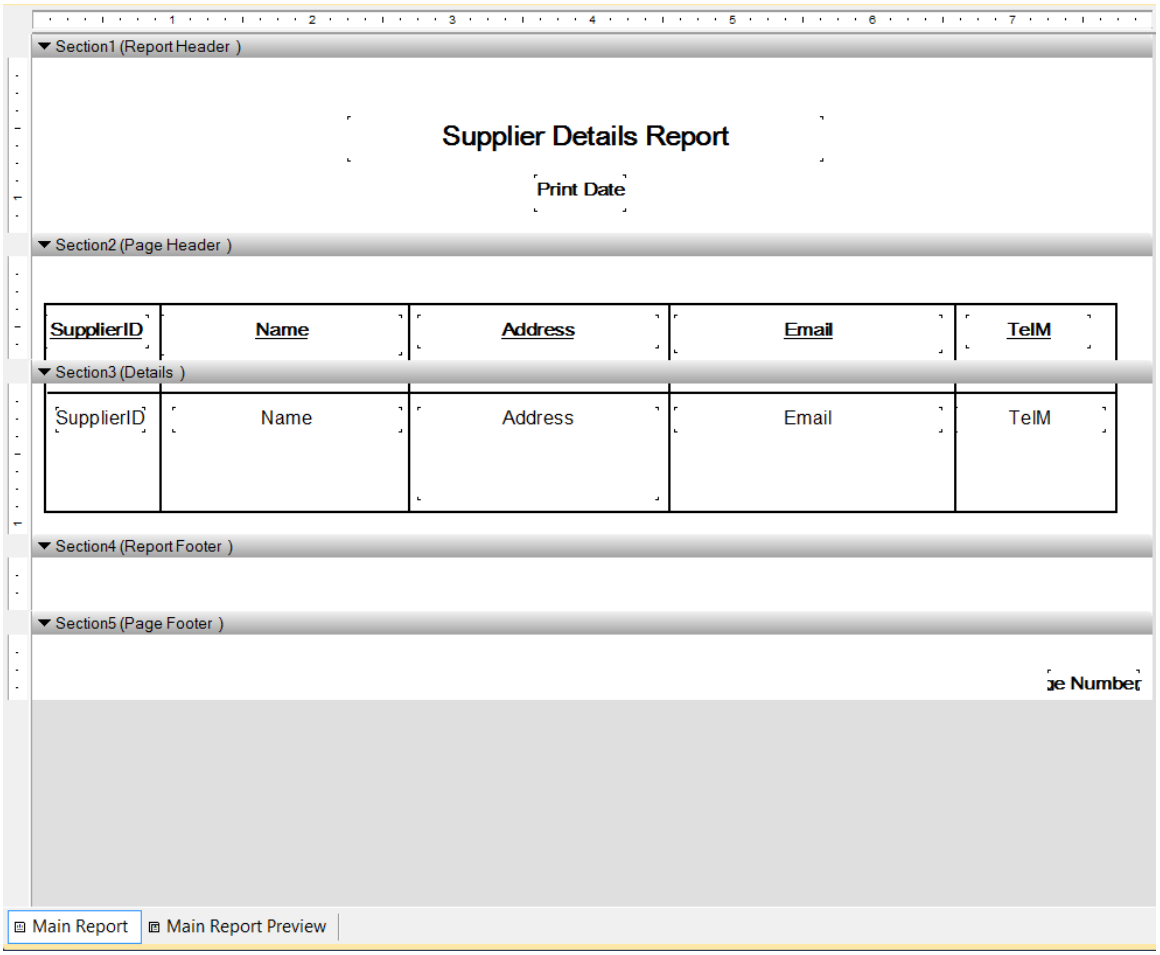


Figure 53

Task 5

5.1 Carry out complete system testing and provide user guide. System testing and user guide must include suitable screenshots. Test your complete project with suitable data.

How to Log into Stock Management System

First install the Stock Management System Software. Then open the Software from the desktop shortcut. Then you can see Stock Management System Log In window.

Figure 54

Next you have to fill your User Name and Password in the related text boxes and click “Login” button.

User Name: admin

Password: 11111

If your user name and password are both correct, you get the Stock Management interface of the Smart Homes Pvt. Ltd. Company.

If your Login information is incorrect you get an error message.

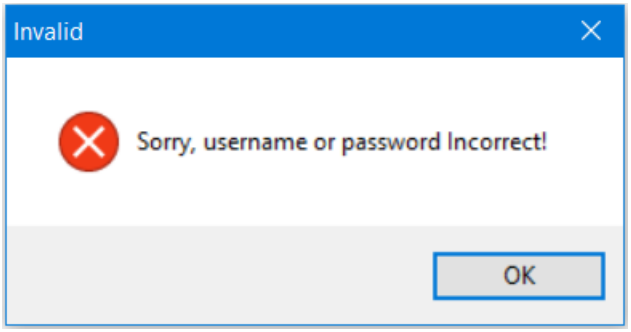


Figure 55

When you click OK you will be redirected to the Login screen to attempt again.

How to Logout from the Stock Management System

We care about your privacy. When you are stepping out for a while or if your days' work is over; we strictly advice you to "Logout" from the system to stop unwanted privacy leakage.

Click on "Logout" option.

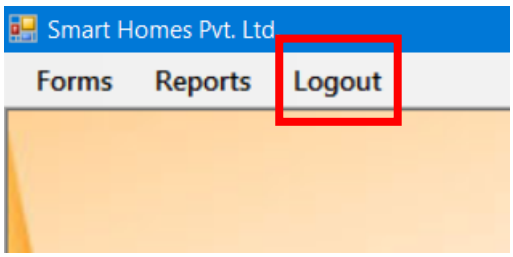


Figure 56

Then you get a message box inquiring "Do you really want to logout?" If you click "Yes" button you will be logged out and you will get a message box saying "Logged out successfully. Have a nice day!" When you click OK you will be redirected to the Login Interface.

If you clicked Logout button unintentionally or if you changed your mind of Logging out, you can click on "No" and execute the command to Logout and return to your work.

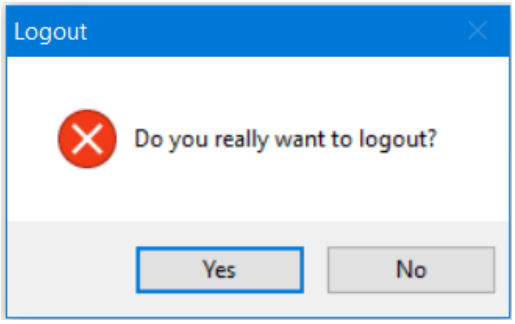


Figure 57

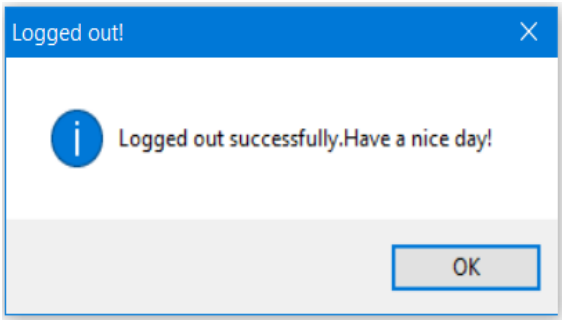


Figure 58

Working with Stock Management System main interface

There are 3 main tabs included in Stock Management System main user interface. These are Forms, Reports and Logout.

If you want to inquire about the Stock Master File, Transactions or about Suppliers you have to select Forms tab.

If you want a detailed report of Transactions or Supplier Registration you have to select Reports tab.

If you want to Logout you have to select Logout tab.

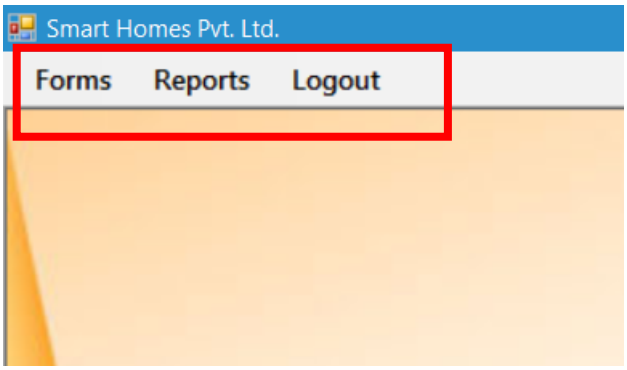


Figure 59

Working with Forms tab

Three sub tabs are included in the Forms tab. These are Stock Master File, Transaction Procession File and Supplier Registration File.

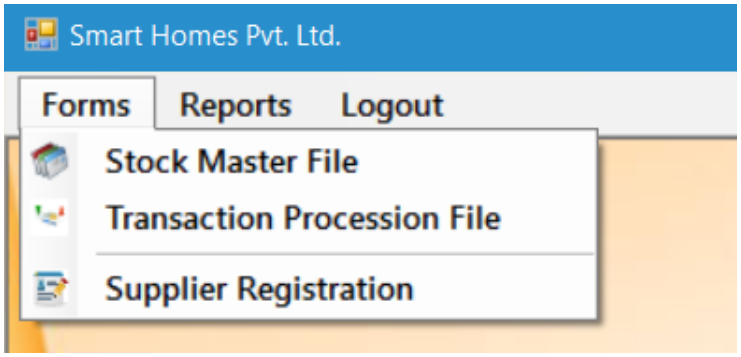


Figure 60

If you want to Search, Add, Edit or Delete an entry in the Stock Master file you have to select Stock Master File tab under Forms tab.

If you want to add details about a new transaction you have to select Transaction Procession File tab.

If you want to add, search or delete an entry in the Supplier Registry you have to select Supplier Registration File tab.

Working with the Stock Master File-SMF

As the first step, you must fill the item code text box in this window. Then the Search button and Clear buttons will be enabled. After you click on the Search button, Delete button and Edit button will be enabled.

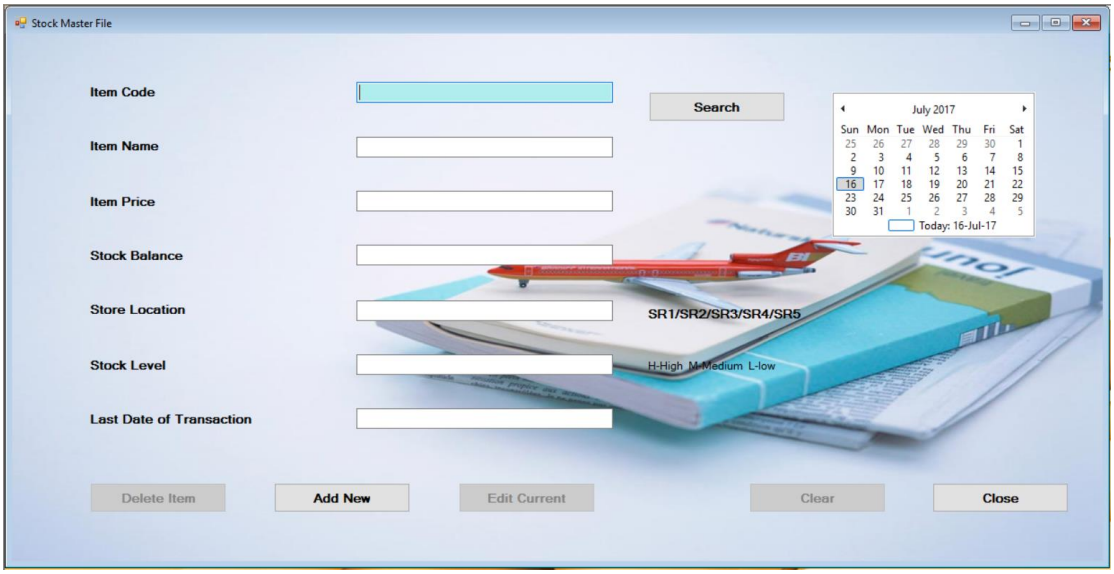


Figure 61

If you forgot to fill any textbox before clicking the Search, Add and Edit buttons you will get an error message. Then when you click OK you will be directed to the relevant blank textbox.

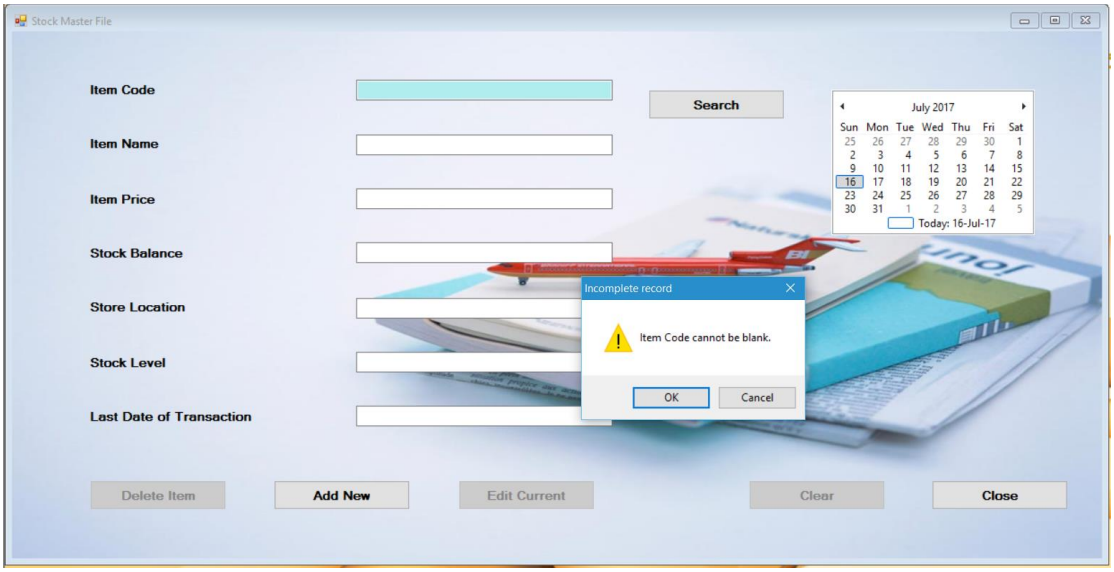


Figure 62

Add data to SQL server from the Stock Master File

Fill the all data in this window and click “Add” Button. Then you get warning massage “Do you want Add New Item to the database?”

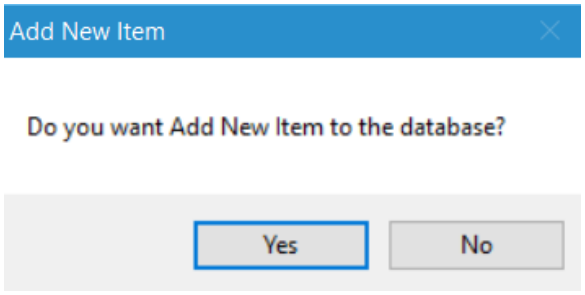


Figure 63

If you click “Yes” button, data will be added to the database and you will get a message box “Item added to the database successfully!” If you click “No” button all the text boxes will be cleared and data will not be added to the database.

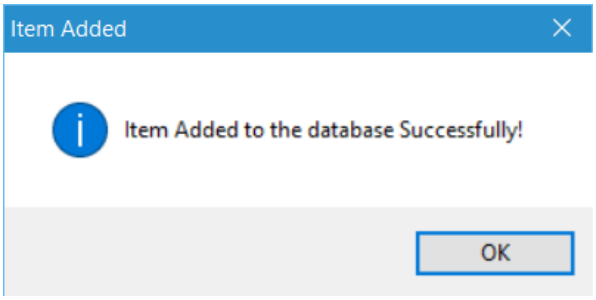


Figure 64

By viewing the SMF table in the StockManagementSystem database on SQL Server Management Studio you can ensure that your data has been successfully added

Before adding data

GLPWIJESINGHE-L.S...tSystem - dbo.SMF

ItemCode	ItemName	ItemPrice	StockBalan...	StoreLocati...	StockLevel	LastDateof...
1001	Rice Cooker...	20000.0000	15	SR1	M	2017-07-16
1002	Gas Cooker-...	30000.0000	20	SR1	M	2017-07-15
1003	Microwave ...	45000.0000	10	SR1	L	2017-06-30
1004	Acer Laptop	90000.0000	5	SR2	L	2017-06-25
1005	LED TV 24"	45000.0000	50	SR3	H	2017-05-17
NULL	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*

Figure 65

After adding data

GLPWIJESINGHE-L.S...tSystem - dbo.SMF

ItemCode	ItemName	ItemPrice	StockBalan...	StoreLocati...	StockLevel	LastDateof...
1001	Rice Cooker...	20000.0000	15	SR1	M	2017-07-16
1002	Gas Cooker-...	30000.0000	20	SR1	M	2017-07-15
1003	Microwave ...	45000.0000	10	SR1	L	2017-06-30
1004	Acer Laptop	90000.0000	5	SR2	L	2017-06-25
1005	LED TV 24"	45000.0000	50	SR3	H	2017-05-17
1006	Huawei P9 l...	37000.0000	40	SR4	H	2017-07-16
NULL	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*

Figure 66

Search data from the Stock Master File

Fill the Item code text box and click the search button. If item code is in the database you get message box “Do you want to view record?” When you click “Yes” button in the Record Found message box, the Stock Master File will be filled with data. If you click “No” button then you will be redirected to fill the Item Code text box. If the Item Code is not in the database, you get message box “Record Not Found”.

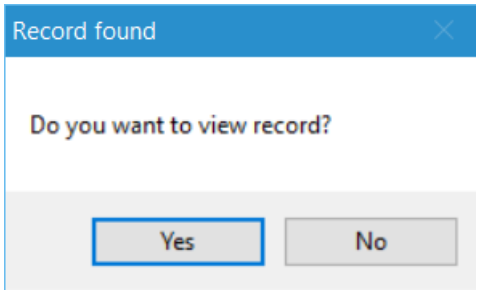


Figure 67

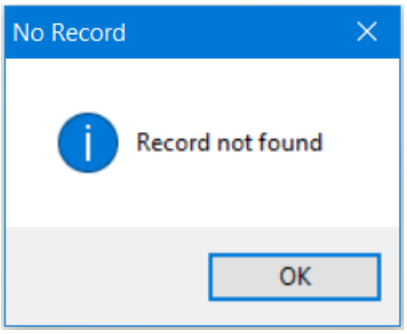


Figure 68

Edit or Update data from the Stock Master File

Fill the item code and click the Search button. Then you get a message box. Click "Yes" in it. Now you can edit the necessary textbox or textboxes. Click edit button. Then you get "Update Item" message box.

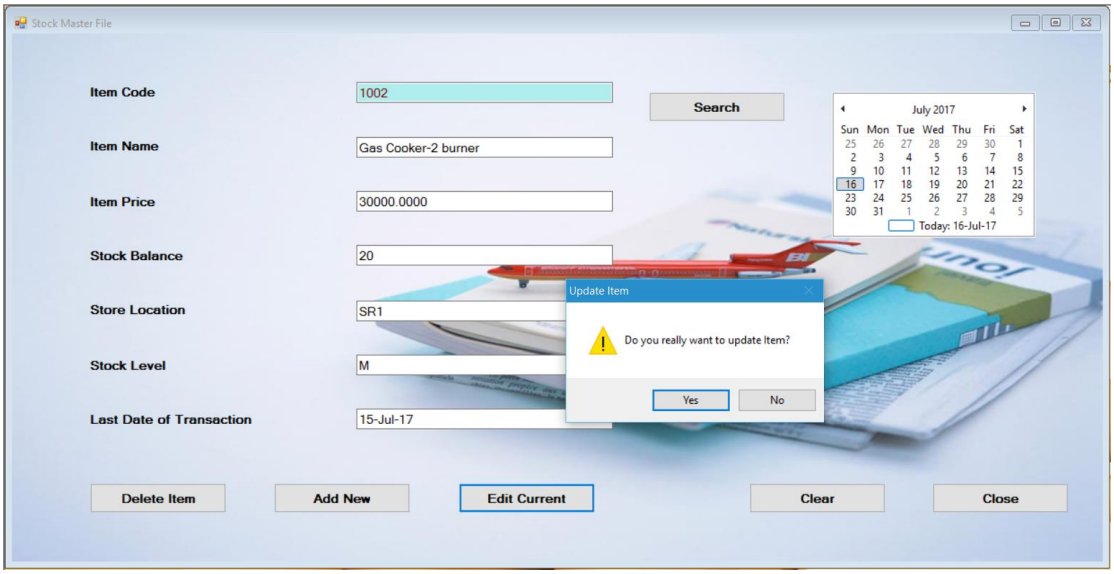


Figure 69

If you click "Yes" the database will be updated and you will get a message box telling you "Updated Successfully"

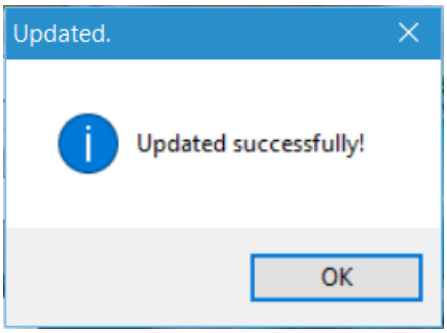


Figure 70

By viewing the SMF table in the StockManagementSystem database on SQL Server Management Studio you can ensure that your data has been successfully updated.

Before editing data

GLPWIJESINGHE-L.S...tSystem - dbo.SMF

	ItemCode	ItemName	ItemPrice	StockBalan...	StoreLocati...	StockLevel	LastDateof...
	1001	Rice Cooker...	20000.0000	15	SR1	M	2017-07-16
	1002	Gas Cooker-...	25000.0000	20	SR1	M	2017-07-15
	1003	Microwave ...	45000.0000	10	SR1	L	2017-06-30
	1004	Acer Laptop	90000.0000	5	SR2	L	2017-06-25
	1005	LED TV 24"	45000.0000	50	SR3	H	2017-05-17
▸*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 71

After editing data

GLPWIJESINGHE-L.S...tSystem - dbo.SMF

	ItemCode	ItemName	ItemPrice	StockBalan...	StoreLocati...	StockLevel	LastDateof...
▸	1001	Rice Cooker...	20000.0000	15	SR1	M	2017-07-16
	1002	Gas Cooker-...	30000.0000	20	SR1	M	2017-07-15
	1003	Microwave ...	45000.0000	10	SR1	L	2017-06-30
	1004	Acer Laptop	90000.0000	5	SR2	L	2017-06-25
	1005	LED TV 24"	45000.0000	50	SR3	H	2017-05-17
	1006	Huawei P9 l...	37000.0000	40	SR4	H	2017-07-16
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 72

Delete data from the Stock Master File

First you should select the relevant Item Code and search it. Next click “Delete” button. Then you will get the message box asking “Do you really want to delete this item?” If you are sure that you need to delete the entry click “Yes” button.

If you don’t want to delete the entry simply click “No” and your entry will not be deleted from the database.

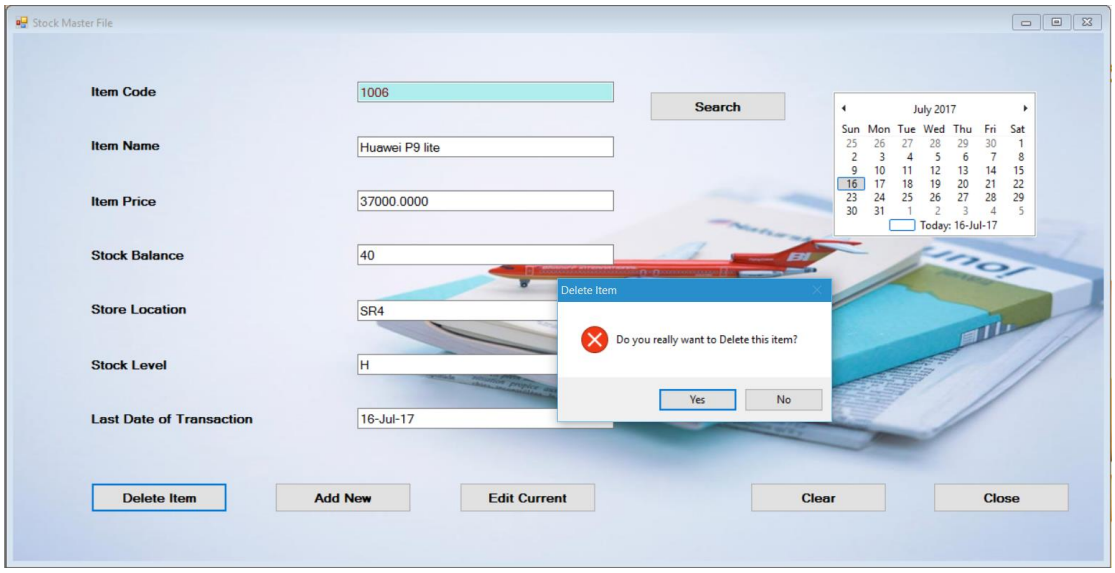


Figure 73

When you click “Yes” you will get a message box informing you that “One item deleted” Then your data has been deleted successfully.

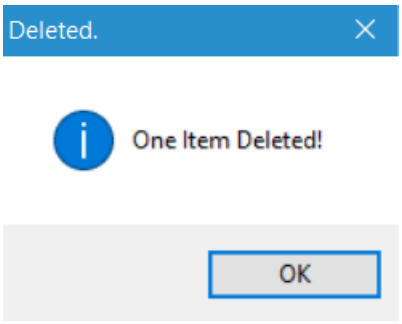


Figure 74

By viewing the SMF table in the StockManagementSystem database on SQL Server Management Studio you can ensure that your data has been successfully deleted.

Before deleing data

GLPWIJESINGHE-L.S...tSystem - dbo.SMF

	ItemCode	ItemName	ItemPrice	StockBalan...	StoreLocati...	StockLevel	LastDateof...
▶	1001	Rice Cooker...	20000.0000	15	SR1	M	2017-07-16
	1002	Gas Cooker-...	30000.0000	20	SR1	M	2017-07-15
	1003	Microwave ...	45000.0000	10	SR1	L	2017-06-30
	1004	Acer Laptop	90000.0000	5	SR2	L	2017-06-25
	1005	LED TV 24"	45000.0000	50	SR3	H	2017-05-17
	1006	Huawei P9 l...	37000.0000	40	SR4	H	2017-07-16
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 75

After deleting data

GLPWIJESINGHE-L.S...tSystem - dbo.SMF

	ItemCode	ItemName	ItemPrice	StockBalan...	StoreLocati...	StockLevel	LastDateof...
▶	1001	Rice Cooker...	20000.0000	15	SR1	M	2017-07-16
	1002	Gas Cooker-...	30000.0000	20	SR1	M	2017-07-15
	1003	Microwave ...	45000.0000	10	SR1	L	2017-06-30
	1004	Acer Laptop	90000.0000	5	SR2	L	2017-06-25
	1005	LED TV 24"	45000.0000	50	SR3	H	2017-05-17
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Figure 76

Clear the Stock Master File

Click Clear button, click "Yes" in the message box and then your form will be cleared.

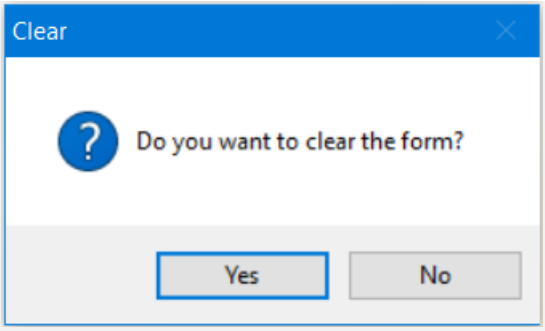


Figure 77

Close the Stock Master File

Click Close button, click "Yes" in the message box and then your form is closed.

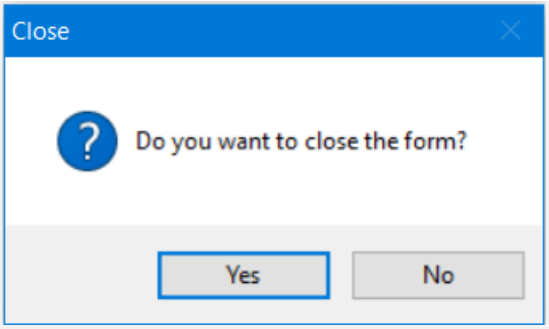


Figure 78

Working with transaction processing file

Fill the item code text box and click Search button [1]. Then you will get a message box saying “Do you want to view record?”[2]

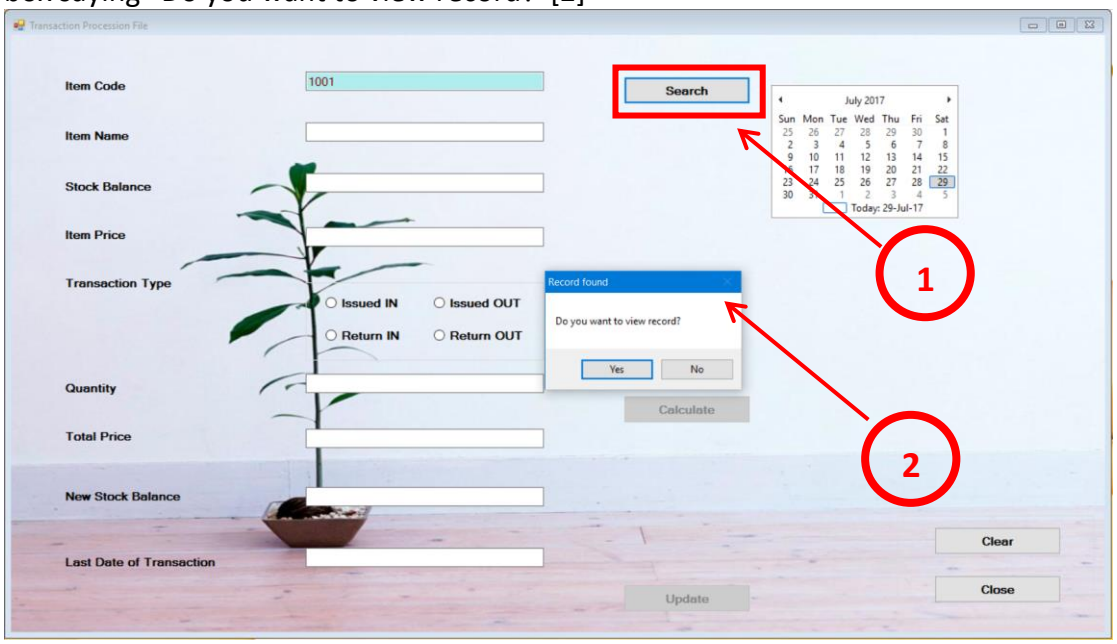


Figure 79

If you click “Yes” button you will see that the Item Name, Stock Balance and the Item Price have been automatically filled.

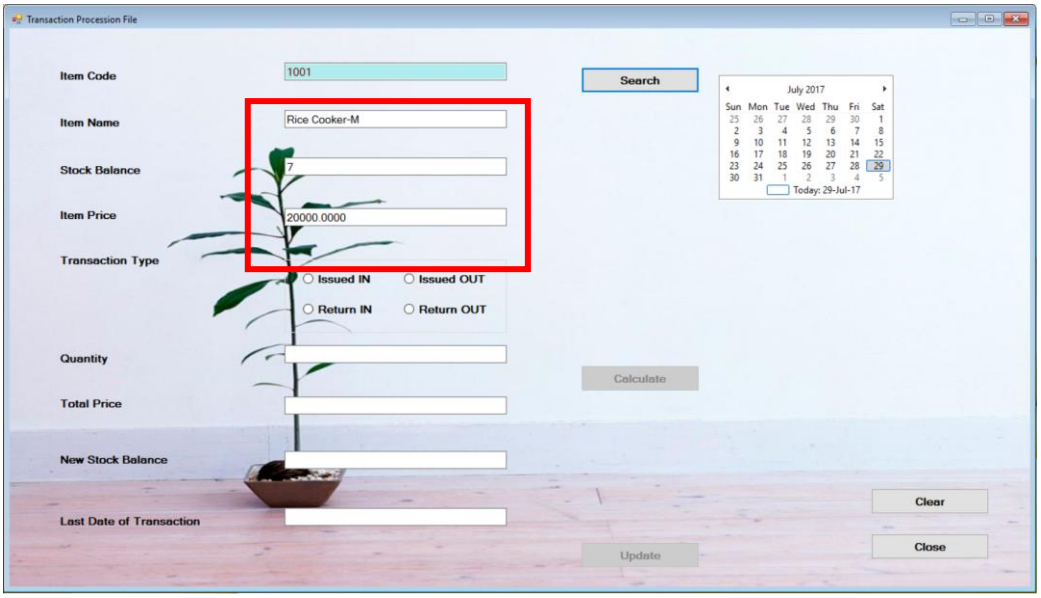


Figure 80

If you click “No” button you will be redirected to enter a new Item Code.

Select the appropriate transaction type and enter the Quantity value [1]. Then click "Calculate" button [2].

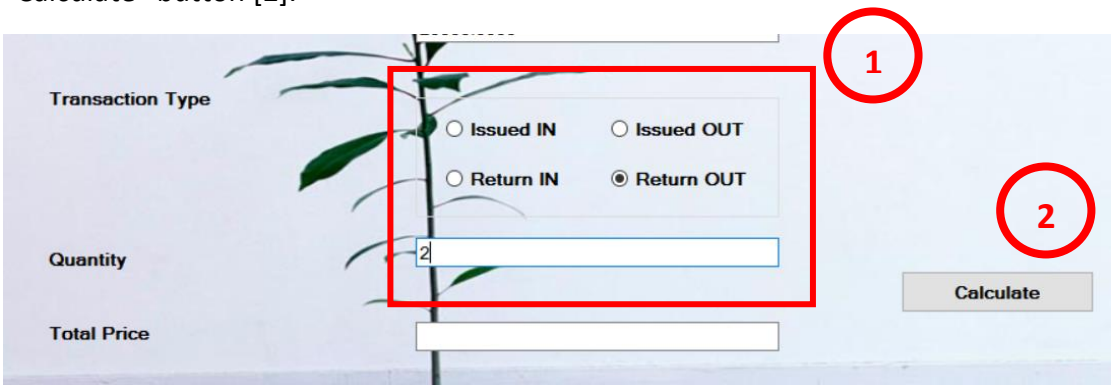


Figure 81

And you will get Total Price, New Stock Balance.

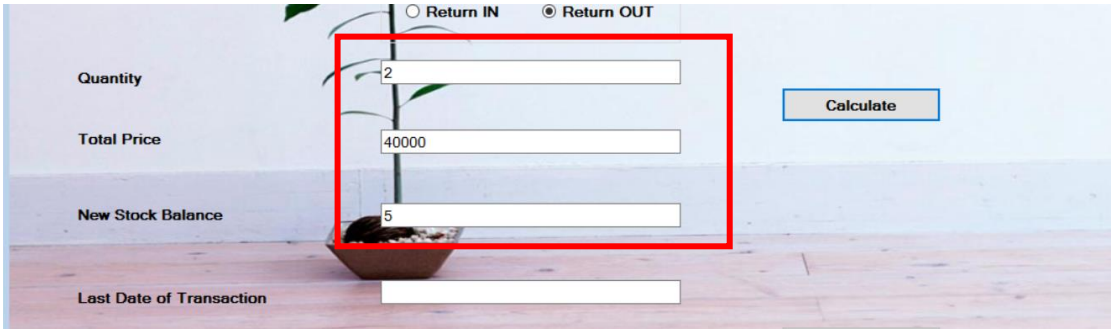


Figure 82

Next fill Last Date of Transaction. Then click "Update" button

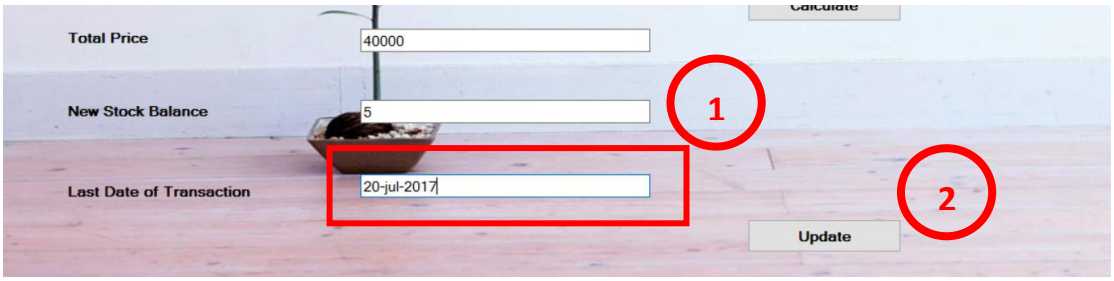


Figure 83

Note: date format should be DD/MMM/YY

First when you click Update button you will get a message box saying “Do you really want to update SMF?” If you click “Yes” button your data will be added to the SMF table in the database and you will get “Updated Successfully” message.

If you click “No” button in the “Update Item” message box the SMF table will not be updated.

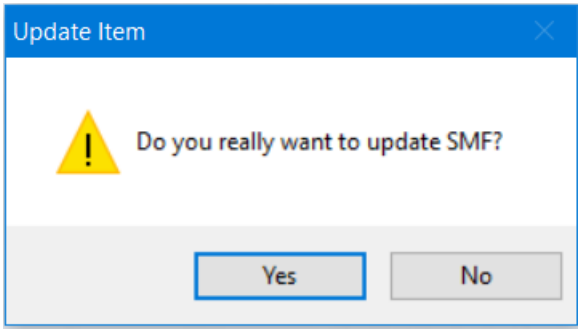


Figure 84

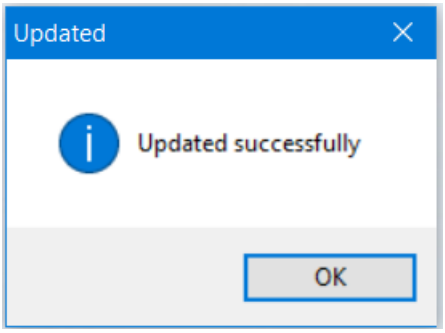


Figure 85

Next you will get a message box saying “Do you really want to update TRF?” If you click “Yes” button your data will be added to the TRF table in the database and you will get “Updated Successfully” message.

If you click “No” button in the “Update Item” message box the TRF table will not be updated.

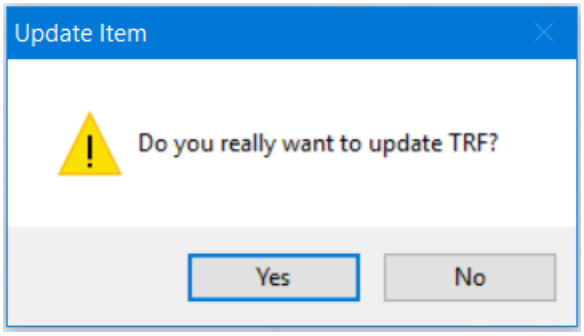


Figure 86

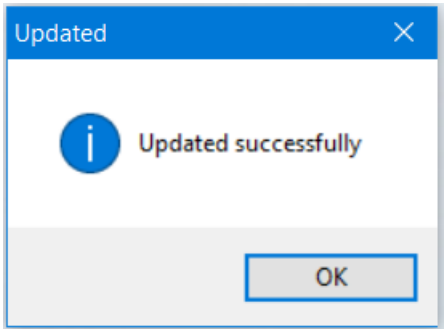


Figure 87

By viewing the SMF table in the StockManagementSystem database on SQL Server Management Studio you can ensure that your data has been successfully updated.

Before updating -SMF table in SQL server

GLPWIJESINGHE-L.S...tSystem - dbo.SMF

ItemCode	ItemName	ItemPrice	StockBalan...	StoreLocati...	StockLevel	LastDateof...
1001	Rice Cooker...	20000.0000	15	SR1	M	2017-07-16
1002	Gas Cooker-...	30000.0000	20	SR1	M	2017-07-15
1003	Microwave ...	45000.0000	10	SR1	L	2017-06-30
1004	Acer Laptop	90000.0000	5	SR2	L	2017-06-25
1005	LED TV 24"	45000.0000	50	SR3	H	2017-05-17
NULL	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*

Figure 88

After updating - SMF table in SQL server

GLPWIJESINGHE-L.S...tSystem - dbo.SMF

ItemCode	ItemName	ItemPrice	StockBalan...	StoreLocati...	StockLevel	LastDateof...
1001	Rice Cooker...	20000.0000	16	SR1	M	2017-07-16
1002	Gas Cooker-...	30000.0000	20	SR1	M	2017-07-15
1003	Microwave ...	45000.0000	10	SR1	L	2017-06-30
1004	Acer Laptop	90000.0000	5	SR2	L	2017-06-25
1005	LED TV 24"	45000.0000	50	SR3	H	2017-05-17
NULL	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*

Figure 89

By viewing the TRF table in the StockManagementSystem database on SQL Server Management Studio you can ensure that your data has been successfully updated in the TRF table also.

After creating new record in TRF table in SQL server

GLPWIJESINGHE-L.S...tSystem - dbo.TRF

ItemCode	ItemName	ItemPrice	Transaction...	Quantity	TotalPrice	NewStockB...	LastDateof...
1001	Rice Cooker...	20000.0000	Return IN	1	20000.0000	16	2017-07-16
1002	Gas Cooker-...	30000.0000	Issued IN	5	150000.0000	20	2017-07-15
1004	Acer Laptop	90000.0000	Return OUT	2	180000.0000	5	2017-06-25
1005	LED TV 24"	45000.0000	Issued OUT	2	90000.0000	50	2017-05-17
NULL	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*

Figure 90

Clear the Transaction Procession File

Click Clear button, click “Yes” in the message box and then your form will be cleared.

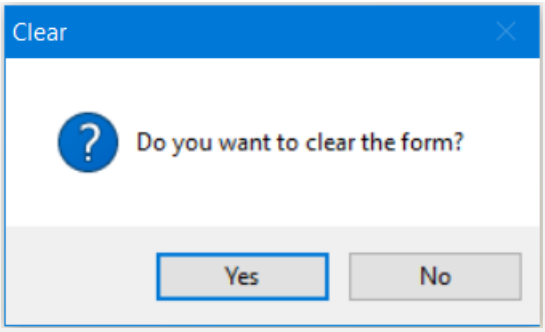


Figure 91

Close the Transaction Procession File

Click Close button, click “Yes” in the message box and then your form is closed.

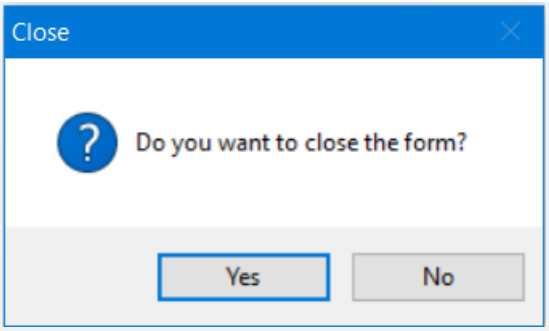


Figure 92

Working with Supplier Registration window

As the first step in this window you must fill the Supplier ID text box. Then Search, Clear buttons will be accessible. If you fill the whole form, all the buttons in this form will be accessible.

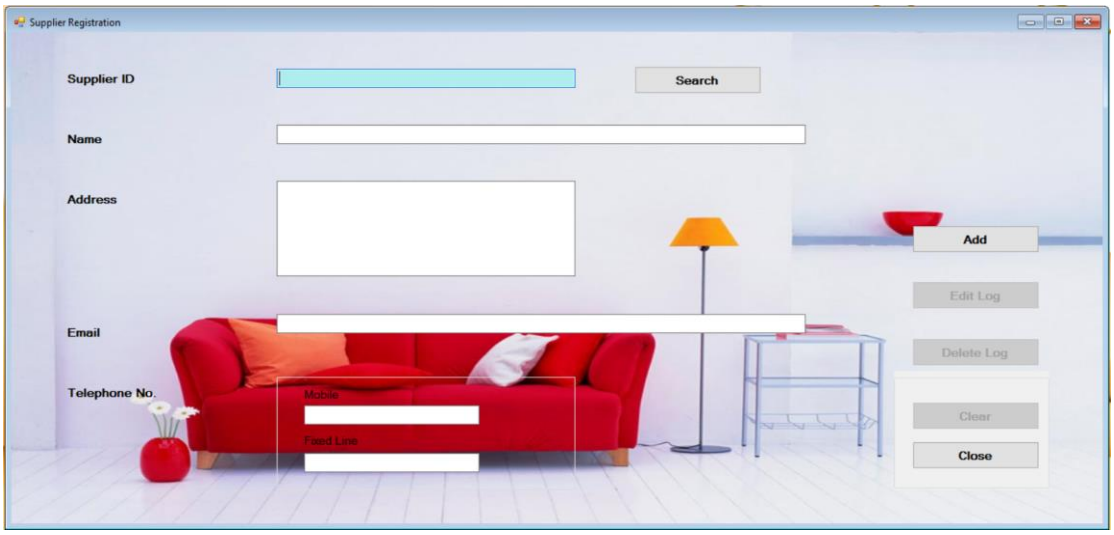


Figure 93

Adding data to Supplier Registration File

Fill the all data in this window and click “Add” Button. Then you get warning massage “Do you want add data to data base?”

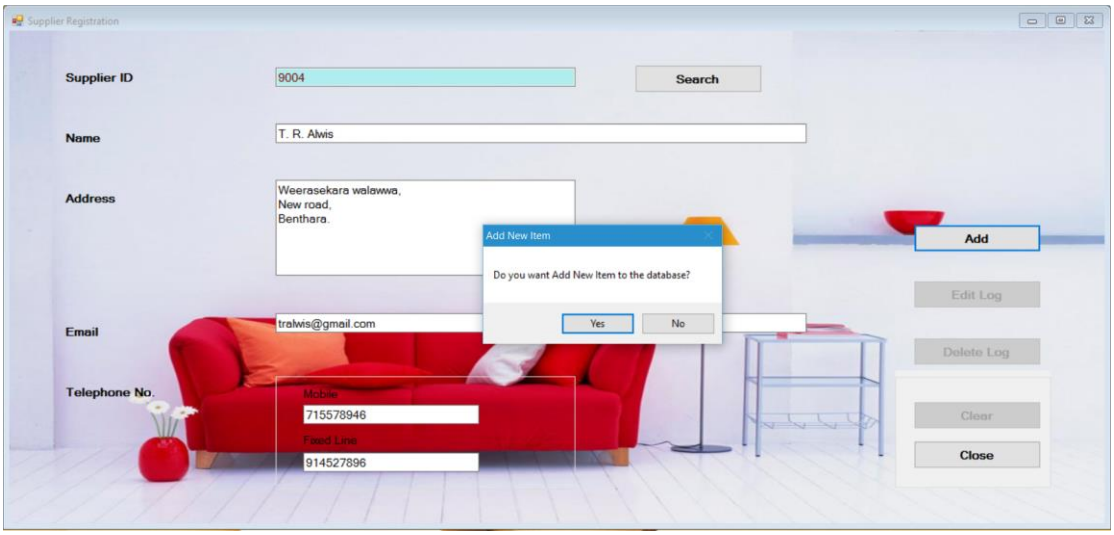


Figure 94

If you click “Yes” button data will be added to the database and you will get message box “Item added to the database successfully”

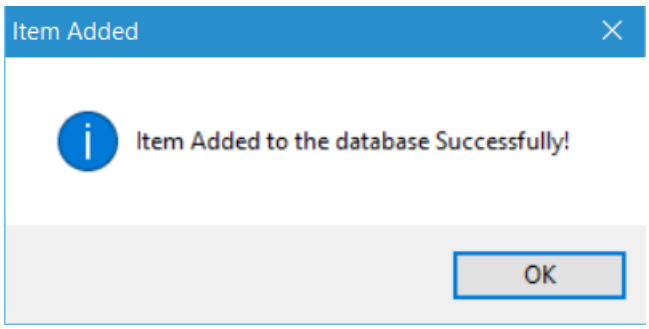


Figure 95

If you click “No” button all the text boxes will be cleared and data will not be added to the database.

Before adding data

GLPWIJESINGHE-L....stem - dbo.SUPREG

SupplierID	Name	Address	Email	TelM	TelF
9001	D. S. W. Liya...	No 1,Templ...	dswliyanag...	771598546	912233445
9002	S. R. Nawar...	No. 05, Jail ...	snawa@gm...	773692587	912245671
9003	D. S. Geega...	"Keshara", A...	dsgee@gm...	714455669	913938028
NULL	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*

Figure 96

After adding data

GLPWIJESINGHE-L....stem - dbo.SUPREG

SupplierID	Name	Address	Email	TelM	TelF
9001	D. S. W. Liya...	No 1,Templ...	dswliyanag...	771598546	912233445
9002	S. R. Nawar...	No. 05, Jail ...	snawa@gm...	773692587	912245671
9003	D. S. Geega...	"Keshara", A...	dsgee@gm...	714455669	913938028
9004	T. R. Alwis	Weerasekar...	tralwis@gm...	715578946	914527896
NULL	*NULL*	*NULL*	*NULL*	*NULL*	*NULL*

Figure 97

Search data from Supplier Registrations window

Fill the Supplier ID text box and click the Search button. If item code is in the database you get message box “Do you want to view record”

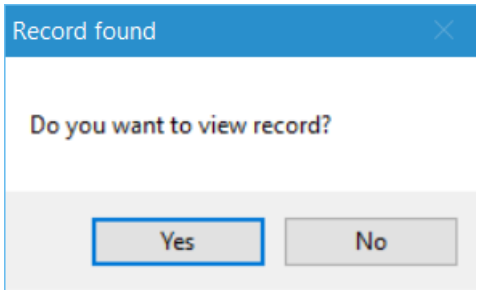


Figure 98

If you click “Yes” button you will see that the form is automatically filled with the existing data. If you click “No” button form will be blank and you will be directed to enter Supplier ID.

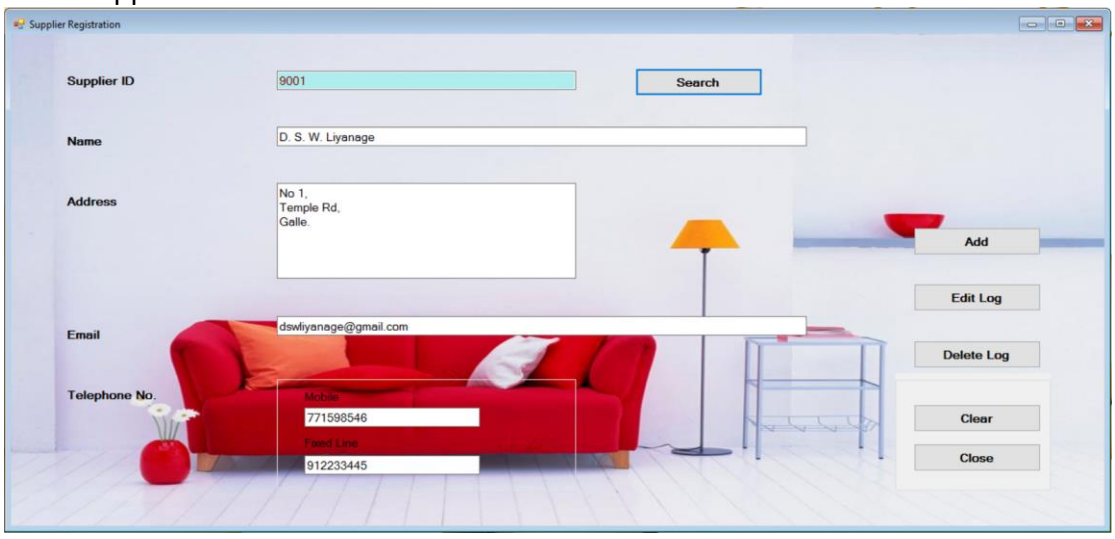


Figure 99

If the Item code is not in the database you get message box “Record Not Found”

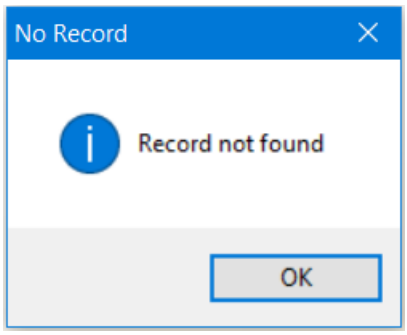


Figure 100

Edit data from Supplier Registration window

Fill the item code and click the search button. Then you get a message box. Click yes button. Then you can display the current data in the textboxes.

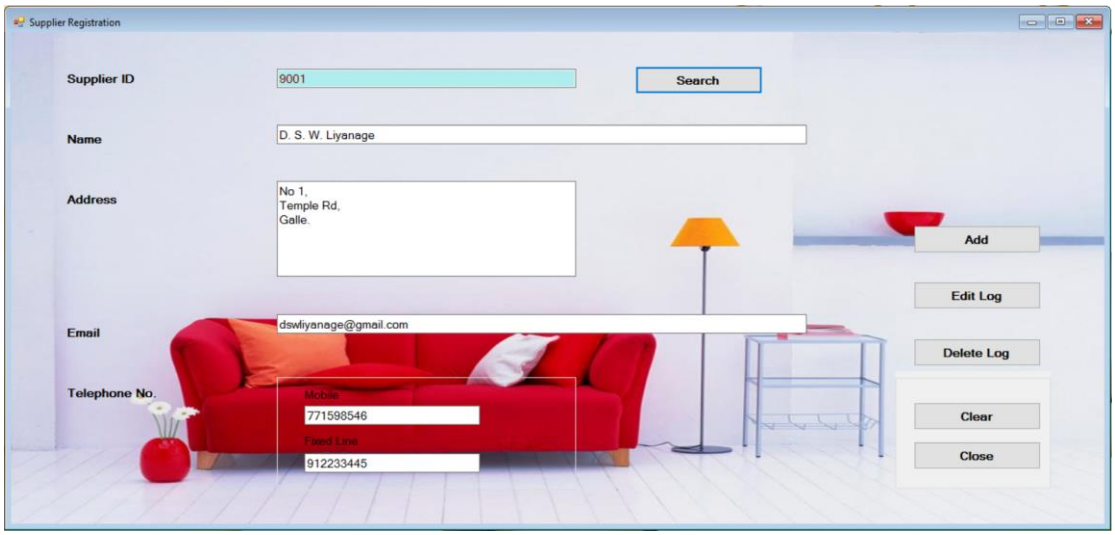


Figure 101

Now you can edit the necessary textbox or textboxes. Next click Edit button. Then you will get “Updated successfully” message box.

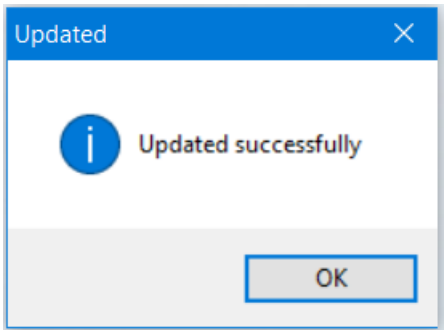


Figure 102

By viewing the SUPREG table in the StockManagementSystem database on SQL Server Management Studio you can ensure that your data has been successfully edited.

Before editing data

GLPWIJESINGHE-L....stem - dbo.SUPREG

	SupplierID	Name	Address	Email	TelM	TelF
	9001	D. S. W. Liya...	No 1,Templ...	dswliyanag...	771598546	912233445
	9002	S. R. Nawar...	No. 05, Jail ...	snawa@gm...	773692587	912245671
	9003	D. S. Geega...	"Keshara", A...	dsgee@gm...	714455669	913938028
	9004	T. R. Alwis	Weerasekar...	tralwis@gm...	715578946	914527896
▸*	NULL	NULL	NULL	NULL	NULL	NULL

Figure 103

After editing data

GLPWIJESINGHE-L....stem - dbo.SUPREG

	SupplierID	Name	Address	Email	TelM	TelF
▸	9001	D. S. W. Liya...	No 18,Temp...	dswliyanag...	702222121	912233445
	9002	S. R. Nawar...	No. 05, Jail ...	snawa@gm...	773692587	912245671
	9003	D. S. Geega...	"Keshara", A...	dsgee@gm...	714455669	913938028
	9004	T. R. Alwis	Weerasekar...	tralwis@gm...	715578946	914527896
*	NULL	NULL	NULL	NULL	NULL	NULL

Figure 104

Delete data from Supplier Registration window

First you select relevant data and search it. Click delete button. Then you will get a message box “Do you really want to delete this item?”

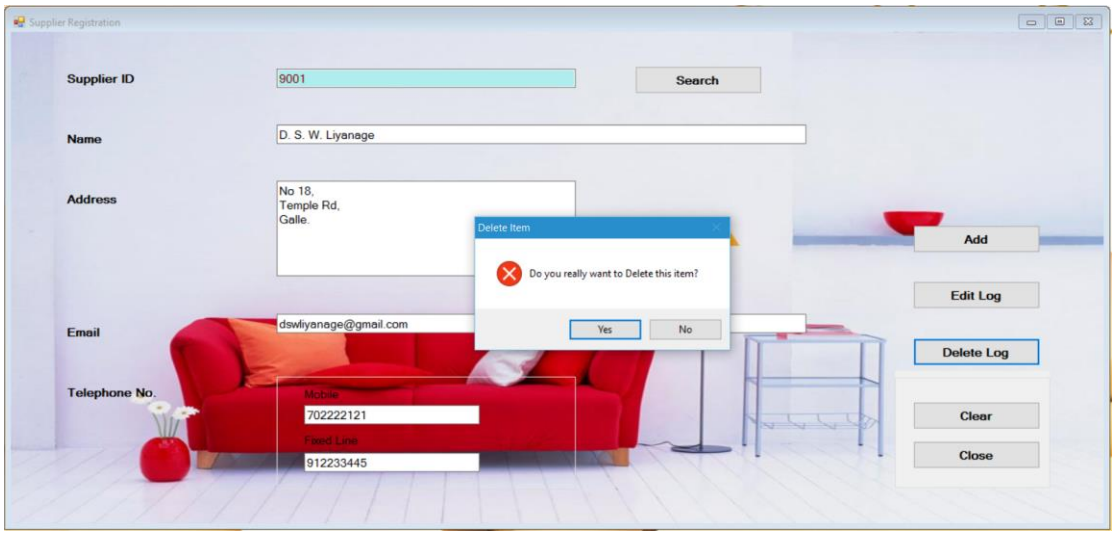


Figure 105

If you click “Yes” your data will be deleted successfully.

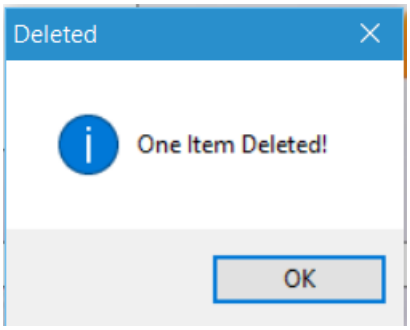


Figure 106

If you click “No” your data will not be affected.

By viewing the SUPREG table in the StockManagementSystem database on SQL Server Management Studio you can ensure that your data has been successfully deleted.

Before deleting data

GLPWIJESINGHE-L....stem - dbo.SUPREG

	SupplierID	Name	Address	Email	TelM	TelF
▶	9001	D. S. W. Liya...	No 18,Temp...	dswliyanag...	702222121	912233445
	9002	S. R. Nawar...	No. 05, Jail ...	snawa@gm...	773692587	912245671
	9003	D. S. Geega...	"Keshara", A...	dsgee@gm...	714455669	913938028
	9004	T. R. Alwis	Weerasekar...	tralwis@gm...	715578946	914527896
*	NULL	NULL	NULL	NULL	NULL	NULL

Figure 107

After deleting data

GLPWIJESINGHE-L....stem - dbo.SUPREG

	SupplierID	Name	Address	Email	TelM	TelF
▶	9002	S. R. Nawar...	No. 05, Jail ...	snawa@gm...	773692587	912245671
	9003	D. S. Geega...	"Keshara", A...	dsgee@gm...	714455669	913938028
	9004	T. R. Alwis	Weerasekar...	tralwis@gm...	715578946	914527896
*	NULL	NULL	NULL	NULL	NULL	NULL

Figure 108

Clear the Supplier Registration File

Click Clear button, click "Yes" in the message box and then your form will be cleared.

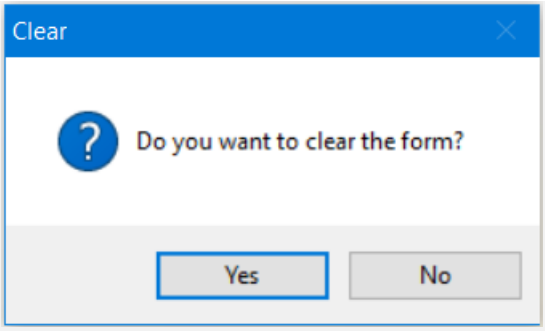


Figure 109

Close the Supplier Registration File

Click Close button, click "Yes" in the message box and then your form is closed.

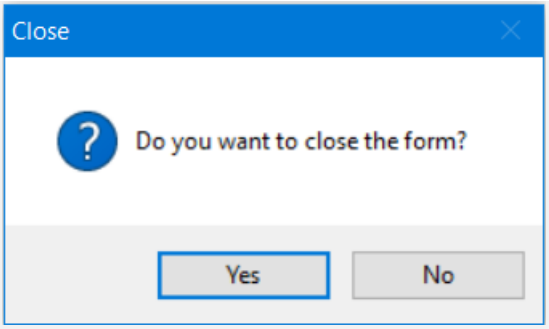


Figure 110

Working with Reports tab

Two sub tabs include in the Reports tab. They are Supplier Details Report and Transaction Processing Detail Report.

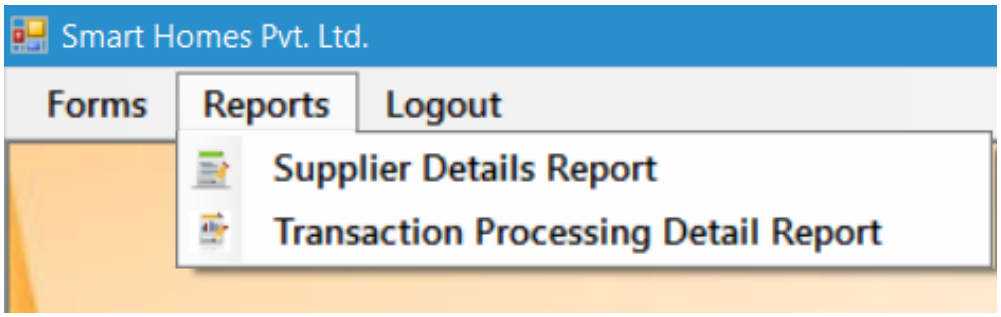


Figure 111

Working with Supplier Details Report

Click Supplier Details Report. Then you get Supplier Details Report window. Here you can see all the supplier details records.

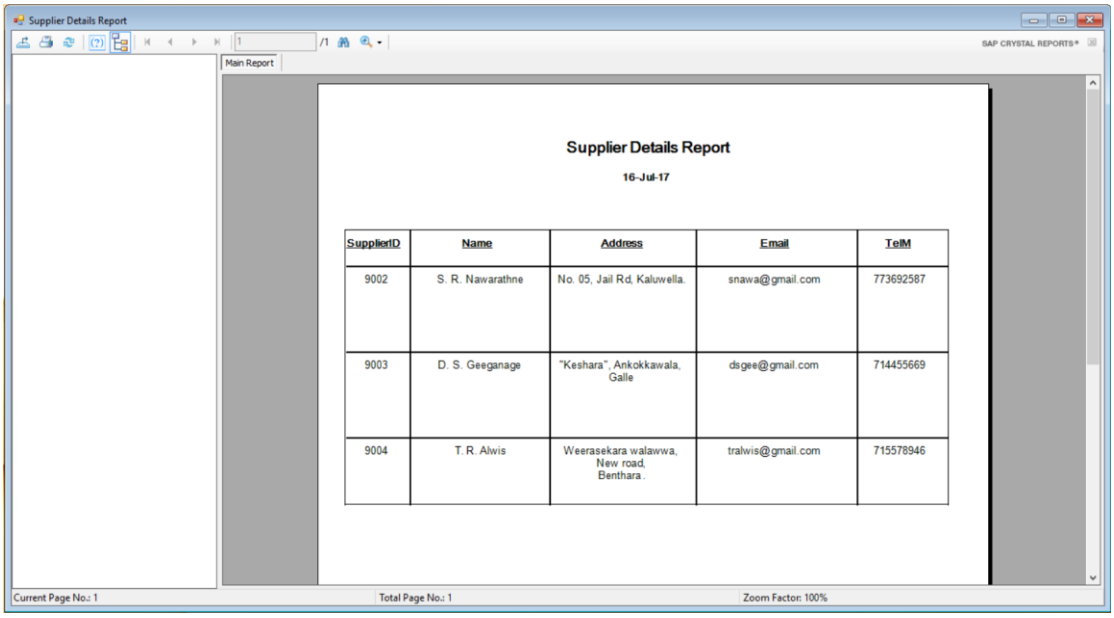


Figure 112

Then you can print this record.

Working with Transaction Processing Report

Click Transaction Processing Report tab. Then you get Transaction Processing Report window. Here you can see all the transaction records.

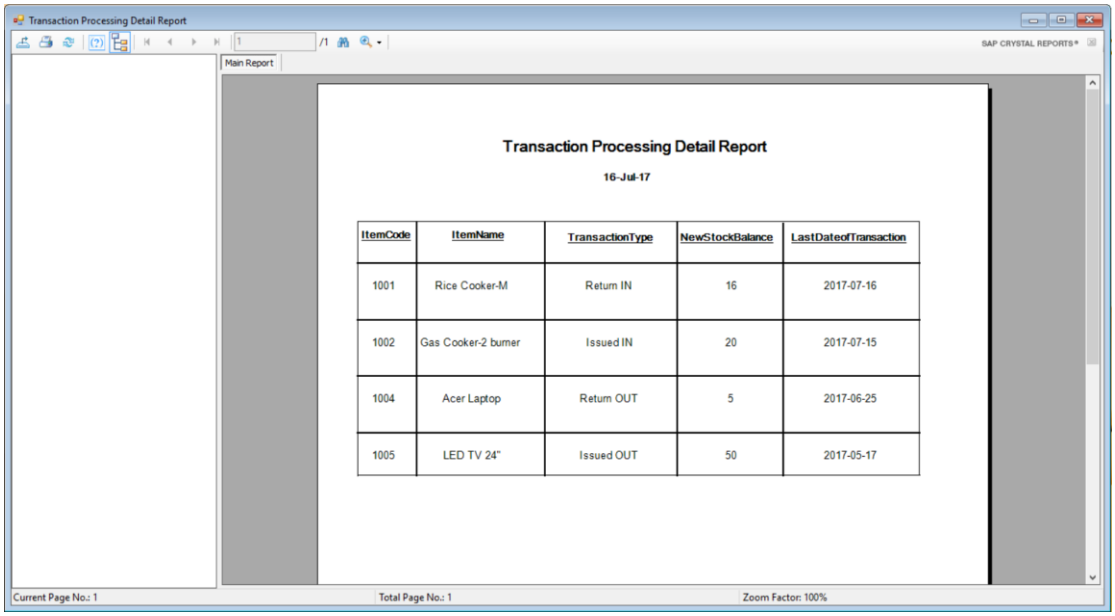


Figure 113

Task 6

6.1 Produce a publishable working copy, a compiled version of the completed assignment together with software installation notes. The installation note should include the system requirements.

Publish the Software

First open your Visual Studio Project. Then click Build and select Publish Stock Management System.

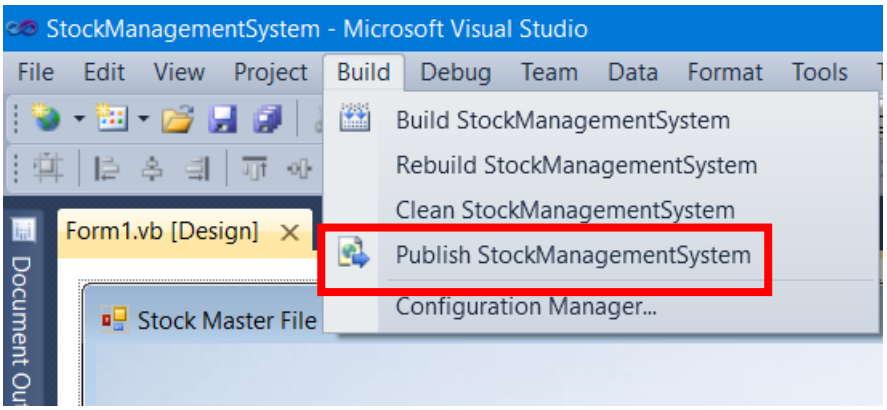


Figure 114

Then click “Next” in the Publish Wizard.

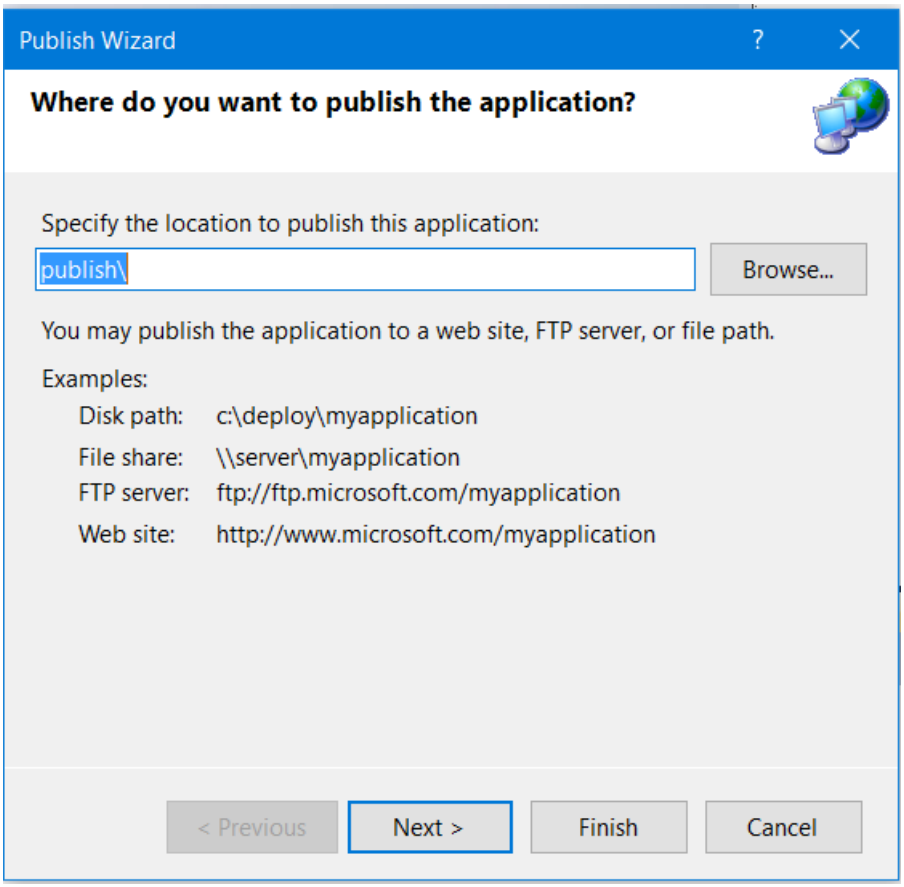


Figure 115

In the next window select the radio button “From a CD-ROM or DVD-ROM” and click “Next”

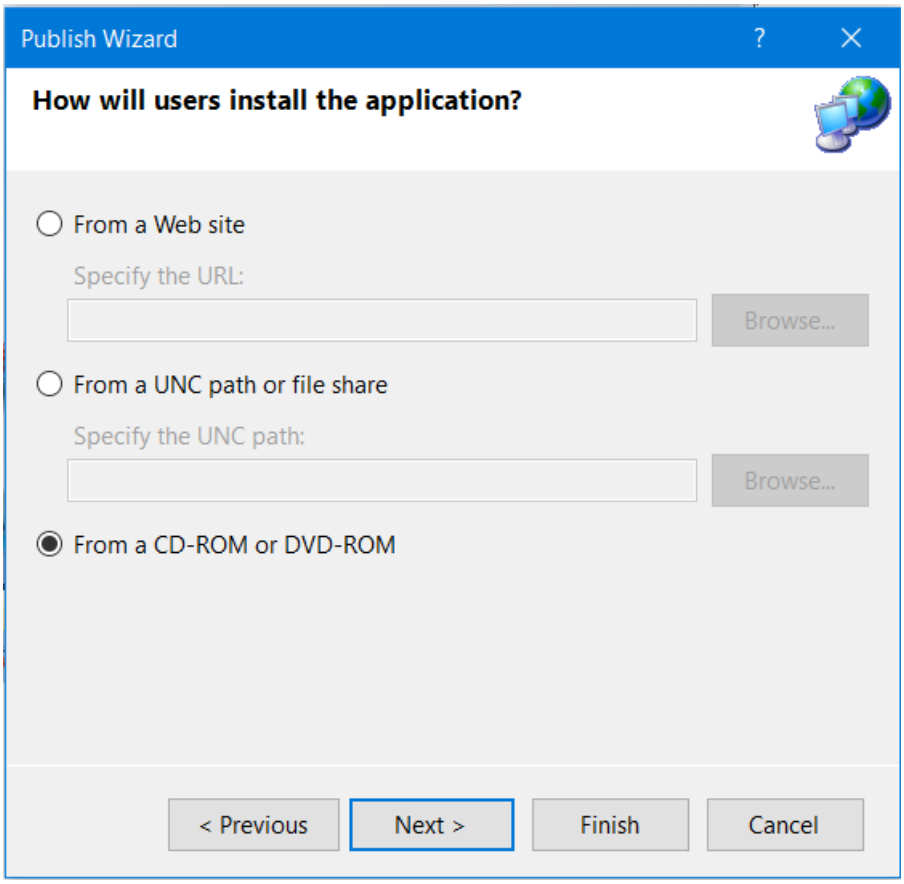


Figure 116

Then you select radio button “**The application will not check for updates”** Then click **Next** button.

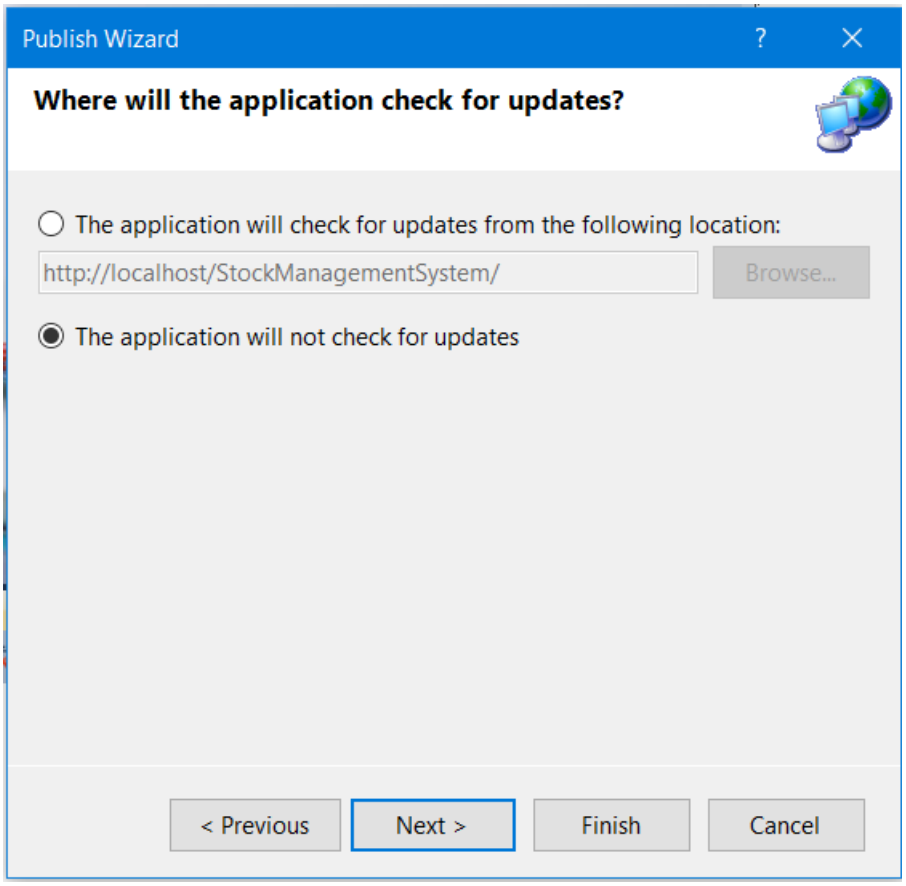


Figure 117

Then click **Finish** button.

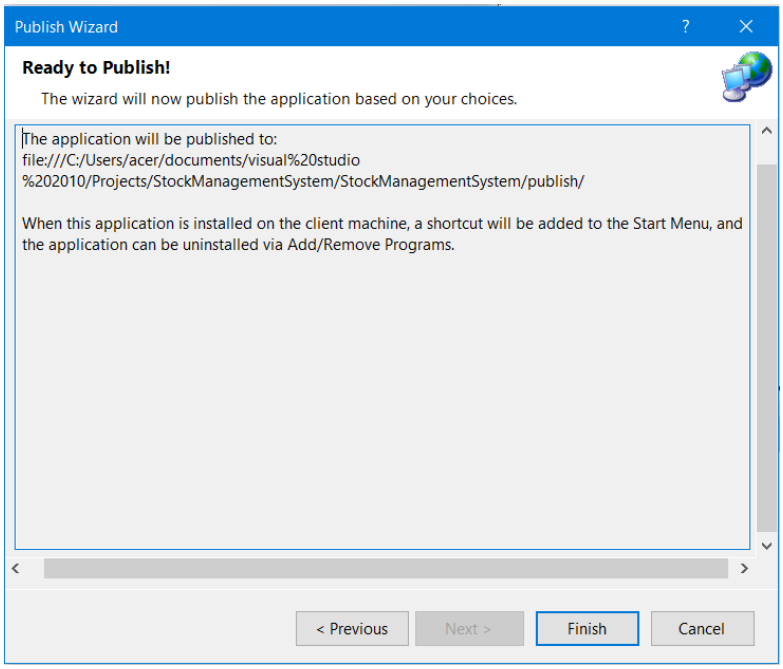


Figure 118

You can find the setup in the location specified by you.

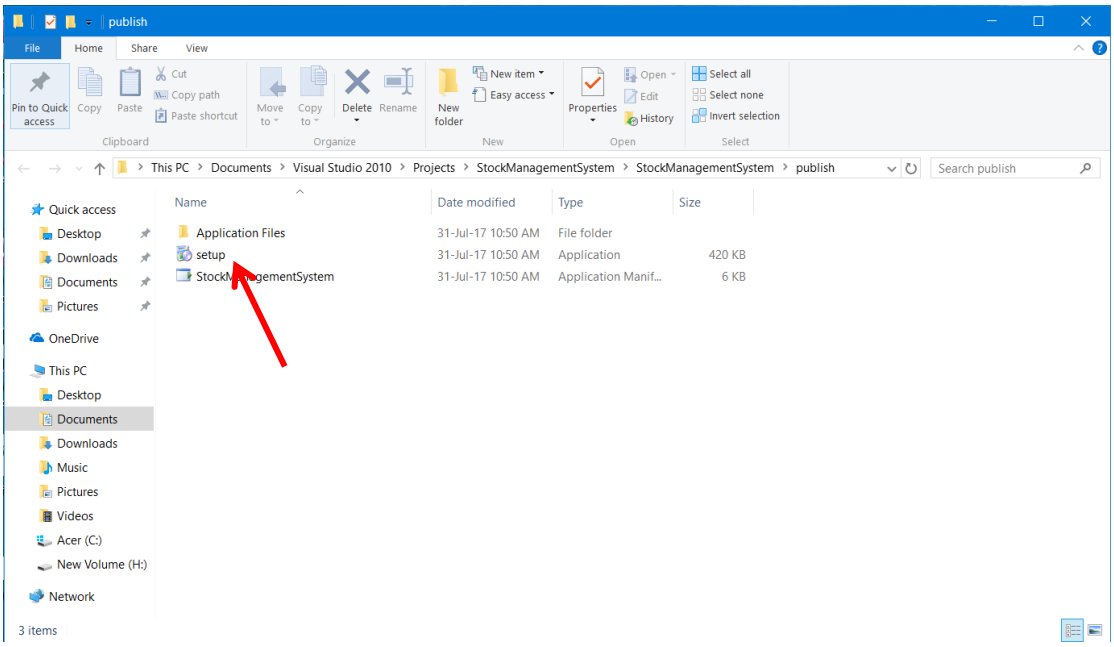


Figure 119

Software installation notes

Double click on the icon of the “StockManagementSystem.exe” file.

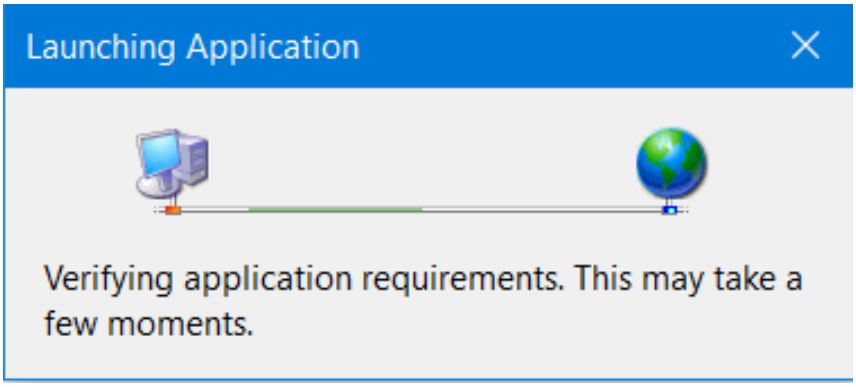


Figure 120

Then click “Install” in the “Application Install- Security Warning” window.

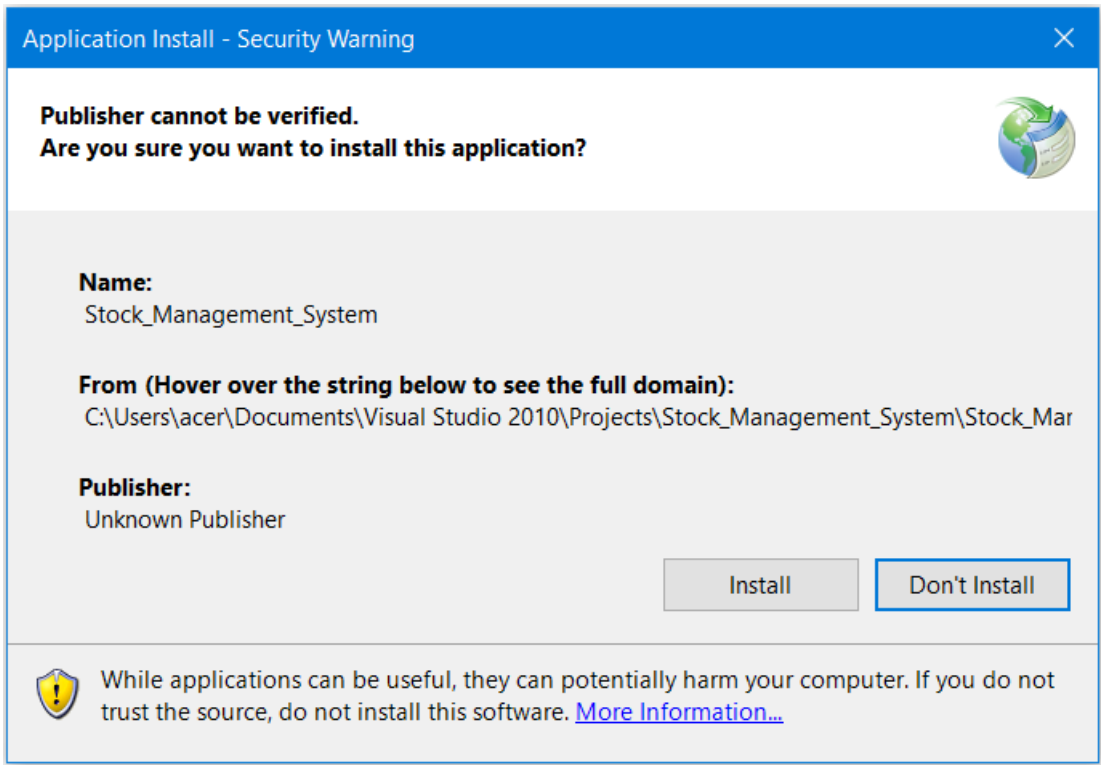


Figure 121

Then wait till the progress bar completes. Now you have successfully installed the Stock Management System in your PC.

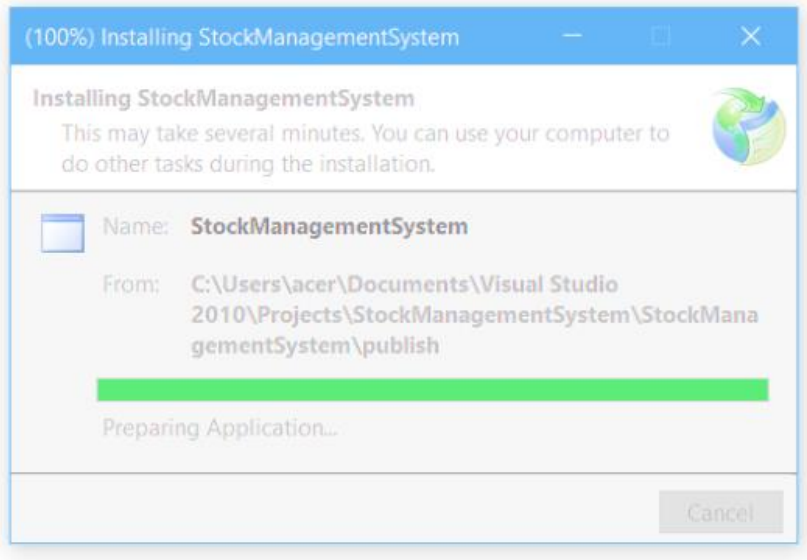


Figure 122

To launch the Application you can double click on the desktop icon of Stock Management System.

System Requirements

Hardware Requirements

- Processor – Intel® Core™ i5-6200U with Turbo Boost up to 2.80GHz
- RAM – 4GB DDR4 Memory

Software Requirements

- Operating System – Windows 10
- 86-bit operating system, x86-based processor
- Microsoft Visual Studio 2010
- Microsoft SQL Server 2014 Management Studio

Task 7

7.1 Develop the java programs for **Practical Assignment 1, Assignment 2 and Assignment 3** in **Book II**.

Java

Figure 123

Java is a general-purpose computer programming language that is concurrent, class-based, object-oriented, and specifically designed to have as few implementation dependencies as possible. It is intended to let application developers "write once, run anywhere" (WORA), meaning that compiled Java code can run on all platforms that support Java without the need for recompilation. Java applications are typically compiled to byte code that can run on any Java virtual machine (JVM) regardless of computer architecture. As of 2016, Java is one of the most popular programming languages in use, particularly for client-server web applications, with a reported 9 million developers. Java was originally developed by James Gosling at Sun Microsystems (which has since been acquired by Oracle Corporation) and released in 1995 as a core component of Sun Microsystems' Java platform. The language derives much of its syntax from C and C++, but it has fewer low-level facilities than either of them.

Practical Assignment 1

```
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;
public class aas3 extends JFrame {
	public static void main(String[] args) {
		new aas3();
				// TODO Auto-generated method stub
	}
	private JLabel l1,l2,l3,l4;
	private JTextField t1,t2,t3,t4;
	private JButton b1;
	ClickOne c=new ClickOne();

	public aas3(){
		this.setSize(400,200);
		this.setTitle("Practical Assingnment 1");
		this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
		this.setLayout(null);

		l1=new JLabel("Employee Number");
		l1.setBounds(20, 20,150,20);

		l2=new JLabel("Hours Worked");
		l2.setBounds(20, 40, 150, 20);

		l3=new JLabel("Hourly Rate");
		l3.setBounds(20, 60, 150, 20);

		l4=new JLabel("Gross Wage");
		l4.setBounds(20, 80, 150, 20);

		t1=new JTextField(10);
		t1.setBounds(175,20,150,20);

		t2=new JTextField(10);
		t2.setBounds(175, 40, 150, 20);

		t3=new JTextField(10);
		t3.setBounds(175, 60, 150, 20);

		t4=new JTextField(10);
		t4.setBounds(175, 80, 150, 20);

		b1=new JButton("Calculate");
		b1.setBounds(100,110, 100, 20);
		b1.addActionListener(c);

		this.add(l1);
```

```
            this.add(t1);
            this.add(l2);
            this.add(t2);
            this.add(l3);
            this.add(t3);
            this.add(l4);
            this.add(t4);
            this.add(b1);
            this.setVisible(true);

      }
      private class ClickOne implements ActionListener{
            //this is an inner class; contained within GUI{
            protected ClickOne(){}

            public void actionPerformed(ActionEvent e){
                  if(e.getSource()==b1){
                        int hw=Integer.parseInt(t2.getText());
                        int hr=Integer.parseInt(t3.getText());
                        int gw=hw*hr;
                        t4.setText(String.valueOf(gw));
                        }
            }
      }
}
```

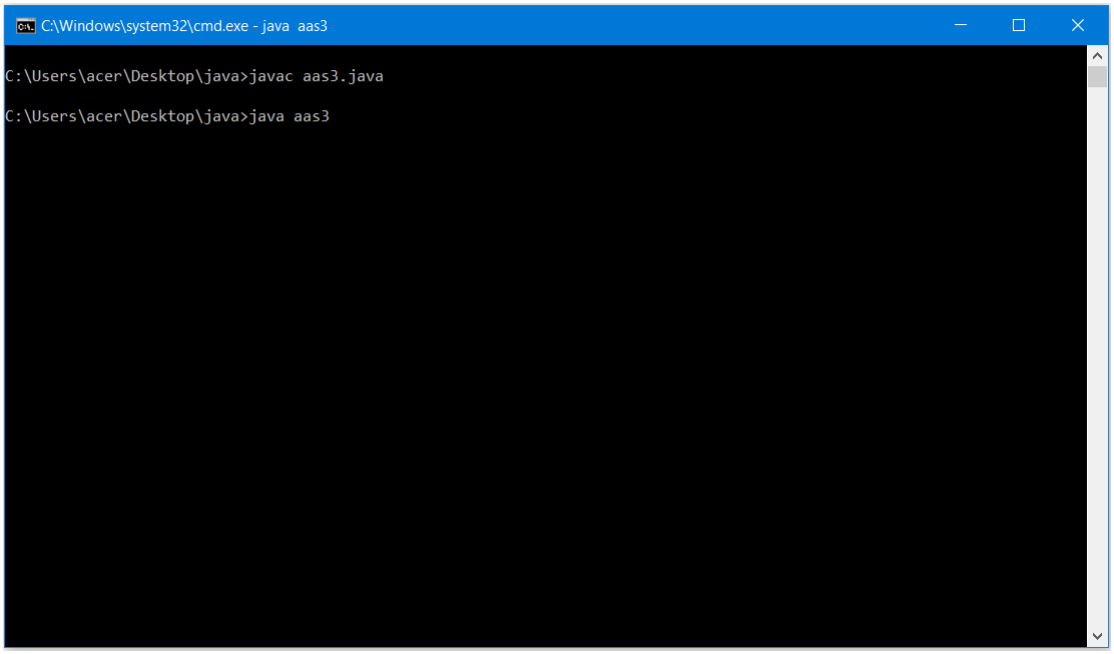


Figure 124

Interface

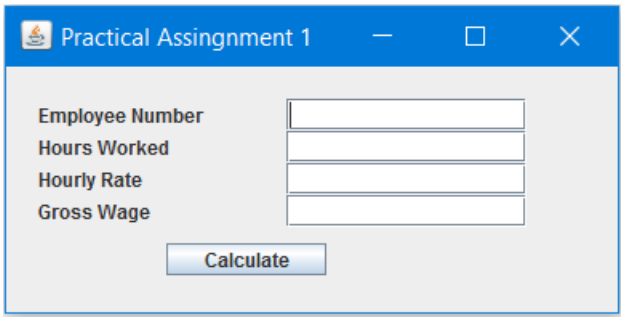


Figure 125

Enter the appropriate data in the cages and click “Calculate” button.

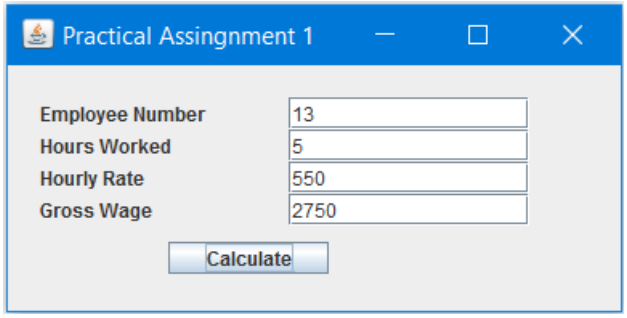


Figure 126

Practical Assignment 2

```
import java.awt.Color;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;

public class pra2 extends JFrame {

	public static void main(String[] args) {
		new pra2();
		// TODO Auto-generated method stub
	}
private JRadioButton b1,b2,b3,b4;
private JPanel p1,p2;
private JLabel l1,l2;
ClickOne c=new ClickOne();

public pra2(){
	this.setSize(400,400);
	this.setTitle("Practical Assignment 2");
	this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
	this.setLayout(null);

	b1=new JRadioButton("Red");
	b1.setBounds(150, 150, 100, 20);
	b1.addActionListener(c);
	this.add(b1);

	b2=new JRadioButton("Green");
	b2.setBounds(150, 170, 100, 20);
	b2.addActionListener(c);
	this.add(b2);

	b3=new JRadioButton("Yellow");
	b3.setBounds(150, 190, 100, 20);
	b3.addActionListener(c);
	this.add(b3);

	b4=new JRadioButton("Blue");
	b4.setBounds(150, 210, 100, 20);
	b4.addActionListener(c);
	this.add(b4);
	ButtonGroup grp=new ButtonGroup();
	grp.add(b1);
	grp.add(b2);
	grp.add(b3);
	grp.add(b4);

	this.setVisible(true);
```

```
      p1=new JPanel();
      p1.setBounds(140, 140,120, 100);
      p1.setBackground(Color.BLACK);
      p1.setLayout(null);
      this.add(p1);

      p2=new JPanel();
      p2.setBounds(70, 20, 250, 100);
      p2.setBackground(Color.WHITE);
      p2.setLayout(null);
      this.add(p2);

      l1=new JLabel("You have selected ");
      l1.setBounds(100, 260, 150,20);
      this.add(l1);
      l2=new JLabel("");
      l2.setBounds(230, 260, 150, 20);
      this.add(l2);

}
private class ClickOne implements ActionListener{
      //this is an inner class; contained within GUI{
      protected ClickOne(){}
      public void actionPerformed(ActionEvent e){
            if(e.getSource()==b1){
                  p2.setBackground(Color.RED);
                  l2.setText("Red Colour");

            }
            if(e.getSource()==b2){
                  p2.setBackground(Color.GREEN);
                  l2.setText("Green Colour");
            }
            if(e.getSource()==b3){
                  p2.setBackground(Color.YELLOW);
                  l2.setText("Yellow Colour");
            }
            if(e.getSource()==b4){
                  p2.setBackground(Color.BLUE);
                  l2.setText("Blue Colour");
            }
      }
}
}
```

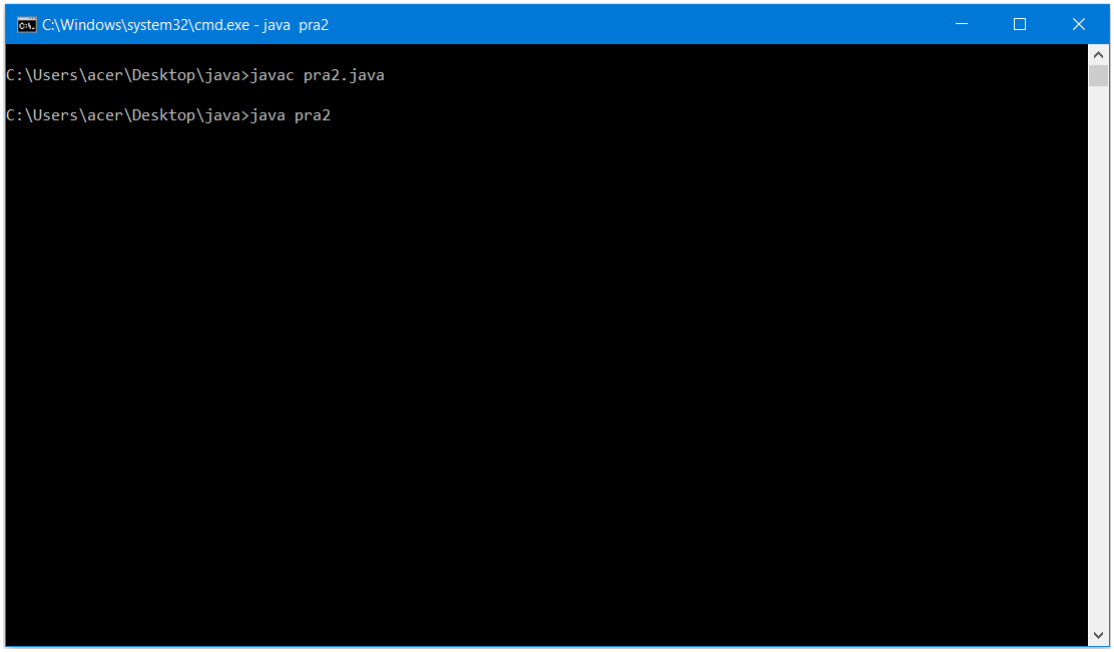


Figure 127

Interface

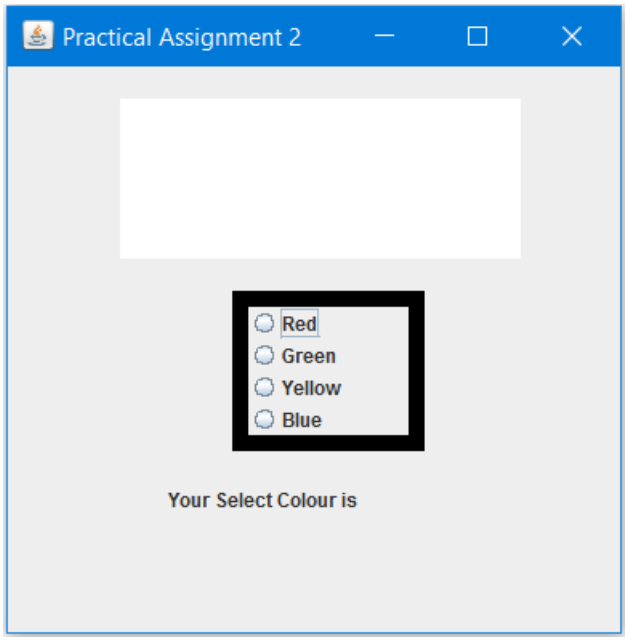


Figure 128

Select the color you want.

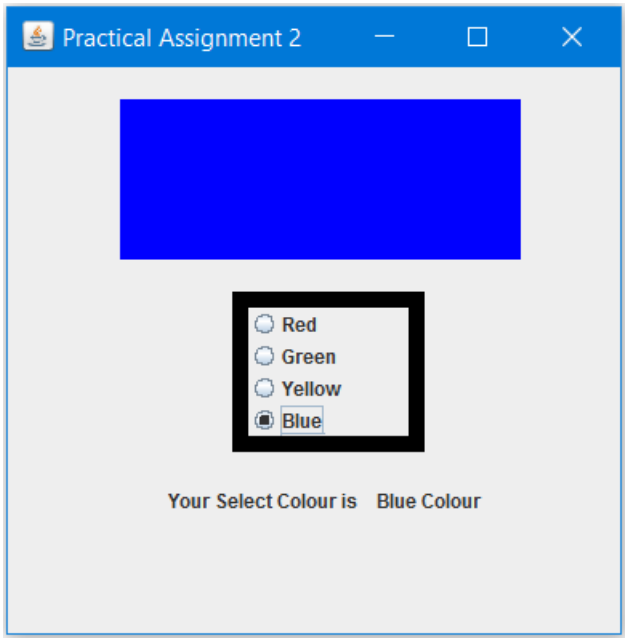

Practical Assignment 2
Red
Green
Yellow
Blue
Your Select Colour is Blue Colour

Figure 129

Practical Assignment 3

```
import java.awt.Font;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import javax.swing.*;

public class pra3 extends JFrame {

	public static void main(String[] args) {
		new pra3();
		// TODO Auto-generated method stub

	}
public JLabel l1,l2,l3,l4,l5,l6,l7,l8,l9,l10,l11,l12,l13,l14;
public JTextField t1,t2,t3,t4,t5,t6;
public JButton b1,b2,b3;
int m1,m2,m3,m4,tot;
float avg;
String tx1,tx2,tx3,tx4;
Clickone c=new Clickone();


public pra3(){
	this.setSize(500,600);
	this.setTitle("Practical Assignment 3");
	this.setLayout(null);
	this.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
	this.setVisible(true);

	l1=new JLabel("Student Number");
	l1.setBounds(30, 20, 150, 20);
	this.add(l1);

	l2=new JLabel("Student Name");
	l2.setBounds(30, 60, 150, 20);
	this.add(l2);

	l3=new JLabel("Marks 1");
	l3.setBounds(30, 100, 150, 20);
	this.add(l3);

	l4=new JLabel("Marks 2");
	l4.setBounds(30, 140, 150, 20);
	this.add(l4);

	l5=new JLabel("Marks 3");
	l5.setBounds(30, 180, 150, 20);
	this.add(l5);

	l6=new JLabel("Marks 4");
```

```
        l6.setBounds(30, 220, 150, 20);
        this.add(l6);

        l7=new JLabel("Total");
        l7.setBounds(30, 260, 150, 20);
        this.add(l7);

        l8=new JLabel("Average");
        l8.setBounds(30, 300, 150, 20);
        this.add(l8);

        l9=new JLabel("Grade");
        l9.setBounds(30, 340, 150, 20);
        this.add(l9);

        t1=new JTextField();
        t1.setBounds(150, 20, 150, 20);
        this.add(t1);

        t2=new JTextField();
        t2.setBounds(150, 60, 150, 20);
        this.add(t2);

        t3=new JTextField();
        t3.setBounds(150, 100, 150, 20);
        this.add(t3);

        t4=new JTextField();
        t4.setBounds(150, 140, 150, 20);
        this.add(t4);

        t5=new JTextField();
        t5.setBounds(150, 180, 150, 20);
        this.add(t5);

        t6=new JTextField();
        t6.setBounds(150, 220, 150, 20);
        this.add(t6);

        l10=new JLabel("NULL");
        l10.setBounds(150, 260, 150, 20);
        this.add(l10);

        l11=new JLabel("NULL");
        l11.setBounds(150, 300, 150, 20);
        this.add(l11);

        l12=new JLabel("NULL");
        l12.setBounds(150, 340, 150, 20);
        this.add(l12);
```

```
      l13=new JLabel("");
      l13.setBounds(150, 395, 150, 20);
      this.add(l13);

      l14=new JLabel("");
      l14.setBounds(150, 380, 150, 20);
      this.add(l14);

      b1=new JButton("Marks");
      b1.setBounds(40, 460, 100, 20);
      b1.addActionListener(c);
      this.add(b1);

      b2=new JButton("Grade");
      b2.setBounds(140, 460, 100, 20);
      b2.addActionListener(c);
      this.add(b2);

      b3=new JButton("Clear");
      b3.setBounds(240, 460, 100, 20);
      b3.addActionListener(c);
      this.add(b3);

}
private class Clickone implements ActionListener{
      protected Clickone(){}

      public void actionPerformed(ActionEvent e){
            if(e.getSource()==b1){
                  m1=Integer.parseInt(t3.getText());
                  m2=Integer.parseInt(t4.getText());
                  m3=Integer.parseInt(t5.getText());
                  m4=Integer.parseInt(t6.getText());

                  tot=m1+m2+m3+m4;
                  l10.setText(String.valueOf(tot));
                  avg=tot/4;
                  l11.setText(String.valueOf(avg));
            }
            if(e.getSource()==b2){
                  tx3=String.valueOf(t2.getText());
                  tx4=String.valueOf(t1.getText());
                  if (avg>=80){
                        tx1="A";
                        tx2="You have an Excellent pass";
                  }
                  else if (avg>=65){
                        tx1="B";
                        tx2="You have a good pass";
                  }
                  else if (avg>=50){
```

```
                        tx1="C";
                        tx2="You have Credit Pass";
                  }
                  else if (avg>=35){
                        tx1="S";
                        tx2="You have a simple pass";
                  }
                  else{
                        tx1="W";
                        tx2="You have a week pass";
                  }
                  l12.setText(String.valueOf(tx1));
                  l13.setText(String.valueOf(tx2));

      l14.setText(String.valueOf(tx3)+"..,"+"_"+"["+String.valu
eOf(tx4)+"]");
            }
            if(e.getSource()==b3){
                  t1.setText("");
                  t2.setText("");
                  t3.setText("");
                  t4.setText("");
                  t5.setText("");
                  t6.setText("");
                  l10.setText("NULL");
                  l11.setText("NULL");
                  l12.setText("NULL");
                  l13.setText("");
                  l14.setText("");
            }

      }
}
}
```

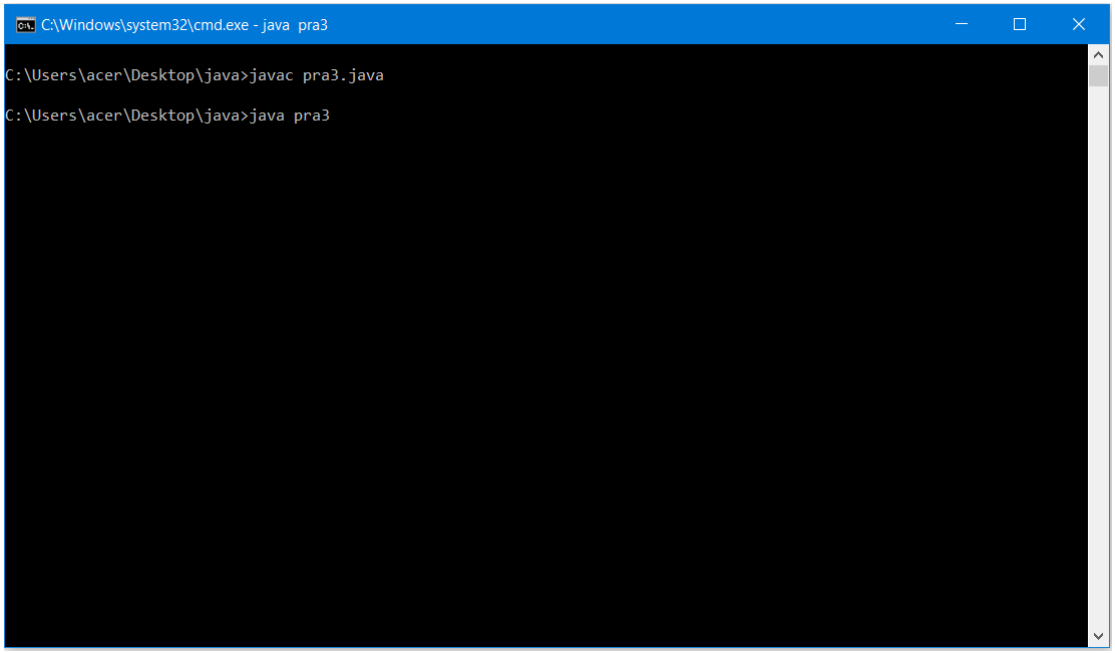

C:\Windows\system32\cmd.exe - java pra3
C:\Users\acer\Desktop\java>javac pra3.java
C:\Users\acer\Desktop\java>java pra3

Figure 130

Interface

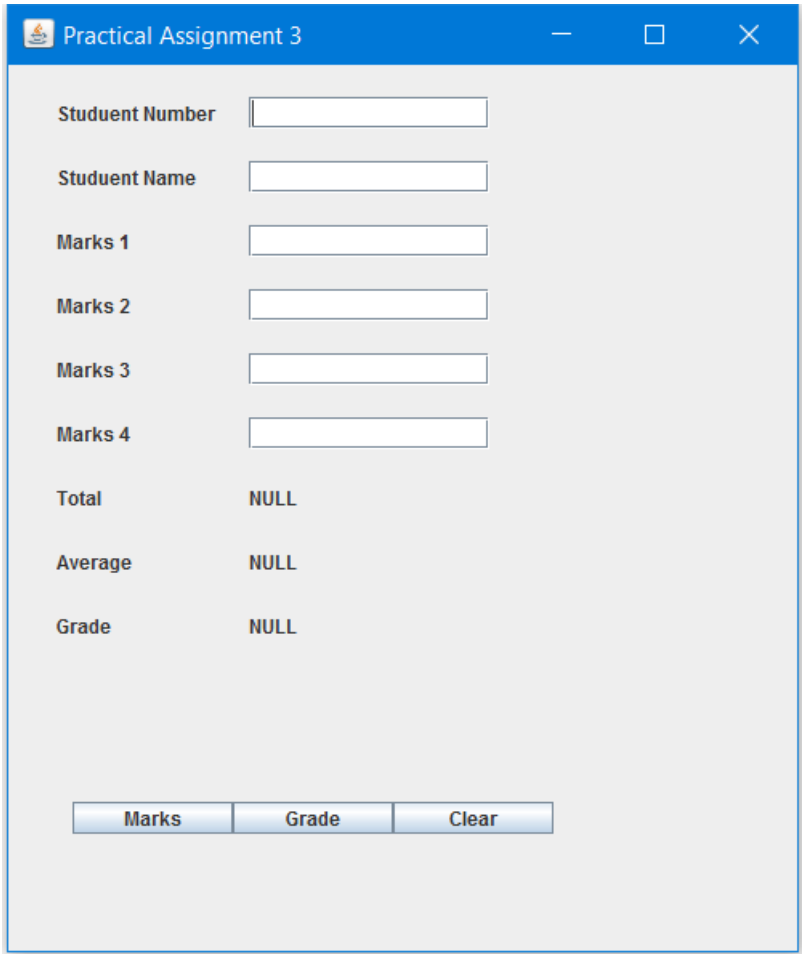


Figure 131

Enter the appropriate data to the cages and click on “Marks”, “Grade” respectively. If you want to clear data click “Clear”

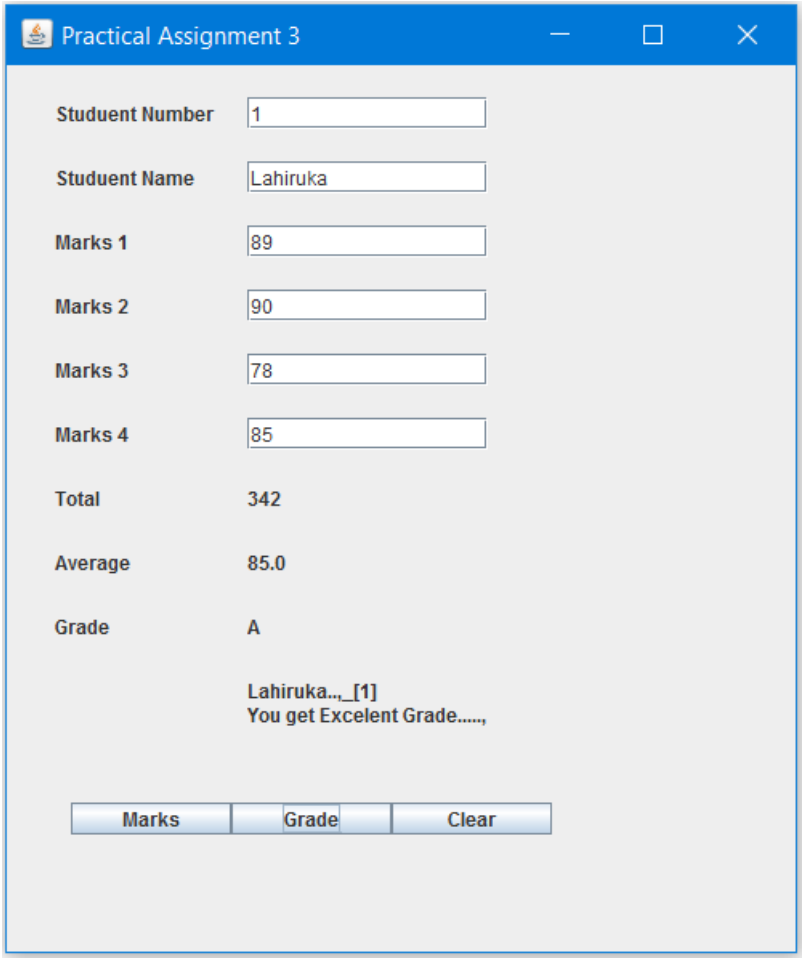

Practical Assignment 3
Studuent Number
1
Studuent Name
Lahiruka
Marks 1
89
Marks 2
90
Marks 3
78
Marks 4
85
Total
342
Average
85.0
Grade
A
Lahiruka...,_[1]
You get Excelent Grade.....,
Marks
Grade
Clear

Figure 132

Summary

This Documentation is about a Stock Management System. To design the Software, Microsoft Visual Studio 2010 was used. The language which was used to write codes was visual basic. Through this web site we were able to practice our knowledge regarding creating database in SQL Server Management Studio, Microsoft Visual Studio 2010 and Photoshop.

Further Java language was used to solve three different kinds of tasks. As a result Java language could be studied. And also using Java language to write programs could be practiced well.

While designing the software I could improve my skills on self-study, self-problem solving and visual basic language implementing.

References

www.vbforums.com

www.codeguru.com

www.wikipaedia.com

www.microsottechupdate.com