**1 INTRODUCTION**

1.1 ABSTRACT

This application consists of a database which will provide the user with a comprehensive and an easy way for obtaining useful information on the stocks traded by their clients. The input file will consist of transactions done by the clients on a day to day basis. The user will be provided with a very easy option of updating the master database, so that he can easily evaluate different statistics such as the payments to be paid by their clients, payments to be received, highest traded stock, etc. The application will also provide a very user-friendly interface along with an option of printing the transaction details of the stocks traded by the clients.

The user will have to provide the software with a .txt file which will consist of number of values with respect to stocks traded on a particular day. The values in the text file are separated by comma. Each value corresponds to a distinct column in the master database. On clicking the option provided, the user can easily integrate the text file into the master database, after which the analysis can be done as per requirement.

The application provides additional functionalities over the ‘BUSYWIN’ by ComTek Software Solutions such as payments to be received, highest traded stock, payments to be paid by their clients, etc.

1.2 INTRODUCTION AND MOTIVATION

The Stock Exchange is a very integral part of any nation’s economy. The growth of a country can be determined by the status or nature of the stock exchange. As a result a large number of people engage in trading shares on the stock exchange. In our country the two main Stock Exchanges are –

* BSE(Bombay Stock Exchange)
* NSE(National Stock Exchange)

In addition to this the trading can also be done in FNO(Futures And Options),MCX etc. The company with which we are dealing is a stock broking firm dealing with the trading of shares for their numerous clients. The database application which will created by us will help our client to have a comprehensive analysis of all the necessary information required by the user.

1.3 PROBLEM DEFINITION

Since the company for which we are creating the application deals with the trading of stocks, they have to continuously maintain a list of the stocks traded by their clients. With help of this database, our user will be able to summarize and analyze the stock trading done by their clients in all formats i.e., individual client-wise, individual stock-wise and in other parameters as may be necessary in order for us to monitor office positions at the end of the trading session.

The application will also be able to track financial positions and liabilities of their clients and our own proprietary positions. The database would be instrumental in allowing our client to view reports based on individuals and different parameters as may be necessary and thus help our client in optimizing their business abilities.

1.4 AIMS AND OBJECTIVES

* To create a comprehensive database that will provide our client with useful information on traded stocks.
* To provide an easy way of handling payment transactions of the clients.
* To monitor holding positions of clients be it, for a day or for a week.
* To provide an option for printing the transactions done by a particular client.
* To execute a list of queries as per user requirements.

1.5 SCOPE

* The application will work for transactions taking place in all the different environments such as BSE, NSE and FNO.
* The user will be able to easily navigate through all the information such as profit, loss, payments, traded stocks, client information at the click of a button.
* An easy HELP option will be provided so that the user can get used to the environment and not have any problems.

**2 REVIEW OF LITERATURE**

2.1 DOMAIN EXPLANATION

2.1.1 Visual Basic

Visual Basic (VB) is the third-generation event-driven programming language and integrated development environment (IDE) from Microsoft for its COM programming model. VB is also considered a relatively easy to learn and use programming language, because of its graphical development features and BASIC heritage.

Visual Basic was derived from BASIC and enables the rapid application development (RAD) of graphical user interface (GUI) applications, access to databases using Data Access Objects, Remote Data Objects, or ActiveX Data Objects, and creation of ActiveX controls and objects. Scripting languages such as VBA and VBScript are syntactically similar to Visual Basic, but perform differently.

A programmer can put together an application using the components provided with Visual Basic itself. Programs written in Visual Basic can also use the Windows API, but doing so requires external function declarations.

Like the BASIC programming language, Visual Basic was designed to be easily learned and used by beginner programmers. The language not only allows programmers to create simple GUI applications, but can also develop complex applications. Programming in VB is a combination of visually arranging components or controls on a form, specifying attributes and actions of those components, and writing additional lines of code for more functionality. Since default attributes and actions are defined for the components, a simple program can be created without the programmer having to write many lines of code. Performance problems were experienced by earlier versions, but with faster computers and native code compilation this has become less of an issue.

Forms are created using drag-and-drop techniques. A tool is used to place controls (e.g., text boxes, buttons, etc.) on the form (window). Controls have attributes and event handlers associated with them. Default values are provided when the control is created, but may be changed by the programmer. Many attribute values can be modified during run time based on user actions or changes in the environment, providing a dynamic application. For example, code can be inserted into the form resize event handler to reposition a control so that it remains centered on the form, expands to fill up the form, etc. By inserting code into the event handler for a key press in a text box, the program can automatically translate the case of the text being entered, or even prevent certain characters from being inserted.

Visual Basic can create executables (EXE files), ActiveX controls, or DLL files, but is primarily used to develop Windows applications and to interface database systems. Dialog boxes with less functionality can be used to provide pop-up capabilities. Controls provide the basic functionality of the application, while programmers can insert additional logic within the appropriate event handlers. For example, a drop-down combination box will automatically display its list and allow the user to select any element. An event handler is called when an item is selected, which can then execute additional code created by the programmer to perform some action based on which element was selected, such as populating a related list.

The Visual Basic compiler is shared with other Visual Studio languages (C, C++), but restrictions in the IDE do not allow the creation of some targets (Windows model DLLs) and threading models.[3]

**Example code**

Here is an example of the language:

|  |
| --- |
| Private Sub Form\_ Load()  'Execute a simple message box that will say "Hello, World!"  MsgBox("Hello, World!")  End Sub |

Figure 2.1: Code snippet that displays a message box "Hello, World!" as the window *Form* loads

2.1.2 Microsoft SQL Server 2008

Microsoft SQL Server is a [relational model](http://en.wikipedia.org/wiki/Relational_model)[database server](http://en.wikipedia.org/wiki/Database_server) produced by [Microsoft](http://en.wikipedia.org/wiki/Microsoft). Its primary [query languages](http://en.wikipedia.org/wiki/Query_language) are [T-SQL](http://en.wikipedia.org/wiki/Transact-SQL) and [ANSI SQL](http://en.wikipedia.org/wiki/SQL).

The current version of SQL Server, SQL Server 2008, was released in 2008 and aims to make data management [self-tuning](http://en.wikipedia.org/wiki/Self-tuning), self-organizing, and self-maintaining with the development of SQL Server Always On technologies, to provide near-zero downtime. SQL Server 2008 also includes support for [structured](http://en.wikipedia.org/wiki/Structured_data) and semi-structured data, including digital media formats for pictures, audio, video and other multimedia data. In current versions, such multimedia data can be stored as [BLOBs](http://en.wikipedia.org/wiki/Binary_large_object) (binary large objects), but they are generic bit streams. Intrinsic awareness of multimedia data will allow specialized functions to be performed on them. SQL Server 2008 can be a data storage backend for different varieties of data: XML, email, time/calendar, file, document, spatial, etc. as well as perform search, query, analysis, sharing, and synchronization across all data types.

New data types include specialized date and time types and a Spatial data type for location-dependent data. Better support for unstructured and semi-structured data is provided using the new FILESTREAM data type, which can be used to reference any file stored on the file system. Structured data and metadata about the file is stored in SQL Server database, whereas the unstructured component is stored in the file system. Such files can be accessed both via [Win32](http://en.wikipedia.org/wiki/Win32) file handling [APIs](http://en.wikipedia.org/wiki/API) as well as via SQL Server using [T-SQL](http://en.wikipedia.org/wiki/T-SQL); doing the latter accesses the file data as a BLOB. Backing up and restoring the database backs up or restores the referenced files as well. SQL Server 2008 also natively supports hierarchical data, and includes [T-SQL](http://en.wikipedia.org/wiki/T-SQL) constructs to directly deal with them, without using recursive queries. The [Full-Text Search](http://en.wikipedia.org/w/index.php?title=SQL_Server_Full_Text_Search&action=edit&redlink=1) functionality has been integrated with the database engine. According to a Microsoft technical article, this simplifies management and improves performance.

SQL Server includes better compression features, which also helps in improving scalability. It enhanced the indexing algorithms and introduced the notion of filtered indexes. It also includes Resource Governor that allows reserving resources for certain users or workflows. It also includes capabilities for [transparent encryption](http://en.wikipedia.org/wiki/Transparent_encryption) of data (TDE) as well as compression of backups. SQL Server 2008 supports the [ADO.NET Entity Framework](http://en.wikipedia.org/wiki/ADO.NET_Entity_Framework) and the reporting tools, replication, and data definition will be built around the [Entity Data Model](http://en.wikipedia.org/wiki/Entity_Data_Model). [SQL Server Reporting Services](http://en.wikipedia.org/wiki/SQL_Server_Reporting_Services) will gain charting capabilities from the integration of the data visualization products from Dundas Data Visualization Inc., which was acquired by Microsoft. On the management side, SQL Server 2008 includesthe Declarative Management Framework which allows configuring policies and constraints, on the entire database or certain tables, declaratively. The version of [SQL Server Management Studio](http://en.wikipedia.org/wiki/SQL_Server_Management_Studio) included with SQL Server 2008 supports [IntelliSense](http://en.wikipedia.org/wiki/IntelliSense) for SQL queries against a SQL Server 2008 Database Engine. SQL Server 2008 also makes the databases available via [Windows PowerShell](http://en.wikipedia.org/wiki/Windows_PowerShell) providers and management functionality available as [Cmdlets](http://en.wikipedia.org/wiki/Cmdlets), so that the server and all the running instances can be managed from [Windows PowerShell](http://en.wikipedia.org/wiki/Windows_PowerShell).[5]

2.2 EXISTING SOLUTION

The company is currently using “BUSY WIN” created by Comtek Software Solutions. This software enables them to view the transactions script-wise and client-wise.These two limited features have resulted in the company entrusting us with this project for ‘Database For Traded Stock Analysis’.With this application we will able to provide the user with the details as required by them.

Moreover, if the application is upto standards with the companies requirements, it has the potential of replacing the current software,thereby enabling the company to save 15,000 rupees per month.

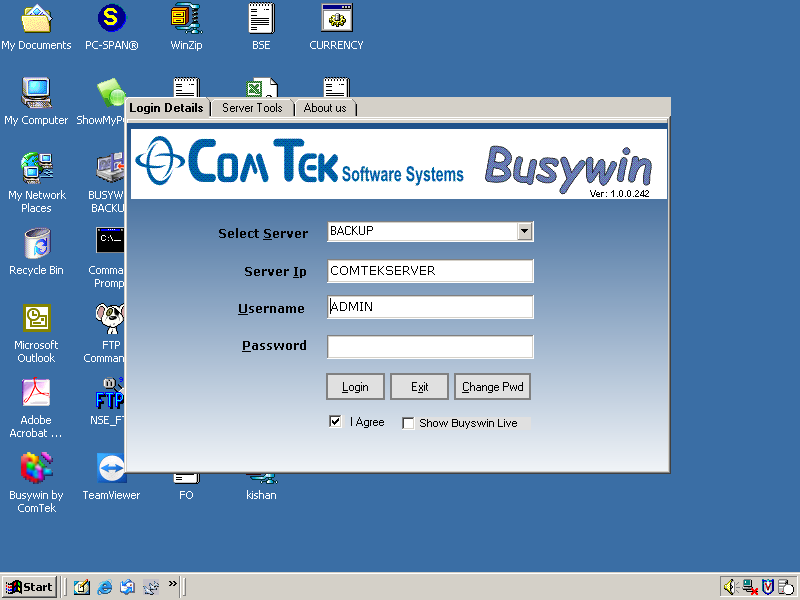


Figure 2.2:A screen shot of the login interface of Busy Win

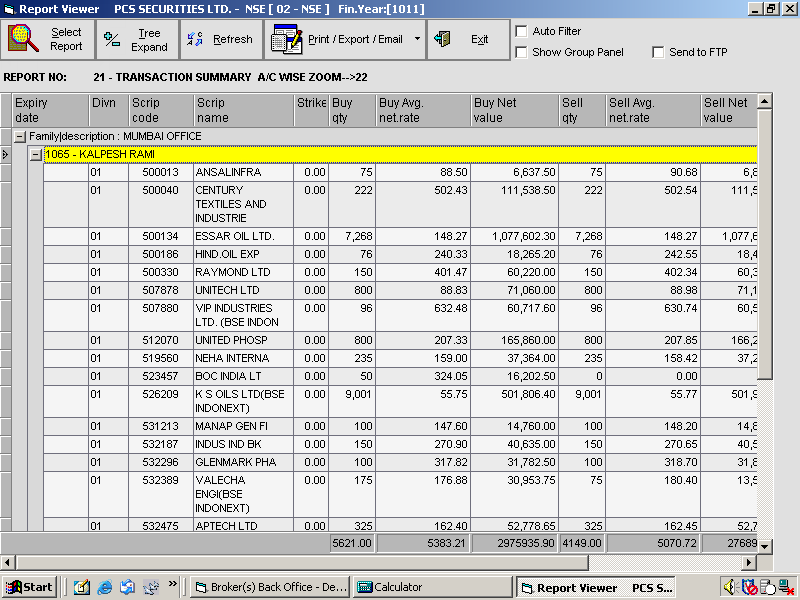


Figure 2.3: Screenshot of Scrip-wise Sorting

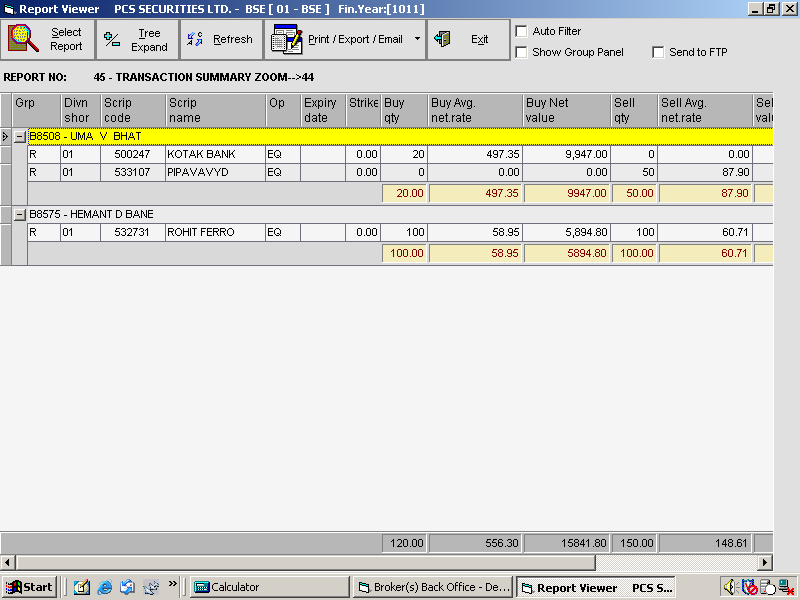


Figure 2.4:Screenshot of Client–wise Sorting

2.3 ProJECT OVERVIEW

Our software consists of following components:

* Graphical User interface
* Original Textfile
* Admin

2.3.1 Component 1 – Graphical User Interface

* First the user is greeted with a login screen. On the login screen, the user has to enter a valid username and password. For users that have not registered there is an option to register.
* If the user selects the option of registering, then the user has to enter a new username, a password and the admin has to enter his/her password to complete the creation of new user account.
* The second screen will consist of four options-

1. Client Information
2. Print Contract
3. Analysis
4. Update Database

* If the admin clicks on Client Information, the admin has to enter a valid client name after which a screen will be shown in which the client details will be displayed. The admin can also add a new client by clicking on the ‘Add Client’ option. On clicking the add client option the admin has to enter the client details and to attach a picture of the client the admin has to either specify the correct path of the picture in the provided textbox or browse the picture from a specific location using the browse option.
* If the admin clicks on Print Contract, a screen will be shown, on the left hand side the content to be printed is displayed and on the right hand side a preview of the print is displayed. There is an option on the toolbar called ‘File’ which has the following options :

1. Print – To print the contract

2. Print Preview – To see a preview of print

3. Page Setup – To adjust the page settings

* If the admin clicks on Analysis, a screen with a dropdown box will be displayed .In the dropdown box there are several queries which can executed by merely clicking on the queries.

2.3.2 Component 2 – Original Text file

The text file of the daily transactions will be stored in .txt format. This file is generated on a day to day basis consisting of the transactions done by the clients. The file has to be integrated daily to master database, so that first-hand and valid information is obtained at all times.

2.3.3 Component 3 - Admin

This module is particularly for the modifications in the software. There will be changes in the information which have to be updated periodically. The Admin will allow the administrator to add information in the Master Database.

**3 ANALYSIS AND DESIGN**

3.1 REQUIREMENT ANALYSIS

3.1.1 HARDWARE AND SOFTWARE REQUIREMENTS

Any system must meet certain requirements in order to be able to run the software on that system. The requirements are given below:

3.1.1.1 Hardware Requirements

Minimum:

* CPU: Intel Pentium 300 MHz or equivalent
* RAM: 256 MB
* Storage:500 MB of free disk space
* Screen resolution: 800 x 600

Recommended:

* CPU: Intel Pentium Dual CoreE5400or better
* RAM: 1GB or more
* Storage: 1 GB of free disk space
* Screen resolution: 1024 x 768

3.1.1.2 Software Requirements

* OS: Microsoft Windows 2000, Microsoft Windows XP, Microsoft Windows Vista, Microsoft Windows 7
* Microsoft SQL Server 2008
* Visual Studio 2008

3.1.2 Functional Requirements

* The application has to provide certain information which has to be obtained from the transactions done by the clients.
* Only the Admin will have an authority to modify the system.
* The application will have to contain sufficient documentation so that the user won’t have any problems while using it.
* The application will also have to provide the option of printing a contract/bill for a particular client.

3.1.3 Non-functional Requirements

* If provided with the resources, then multiple users can get information from the system.
* New information added or modified should not change the existing system.
* Software upgrades also should not affect the functioning of the software in any way.
* The information provided to the user should be up-to-date.
* Only the Admin should be able to login into the system.

3.2 Design Consideration

3.2.1 Tables

The application shall consist of the following tables-

|  |  |
| --- | --- |
| **Table Name** | **Function** |
| 1.tblUser | Will consist of the usernames and passwords of the users who can access the application. |
| 2.tblClient | Will store all the information regarding the clients trading stocks with  the company. |
| 3.tblMaster1 | The main database which will consist of all the transactions done by the clients which will updated on day to day basis. |

Table 3.1 Tables and Functions

3.2.1.1 Table tblUser

The Admin table will consist of the following columns-

|  |  |
| --- | --- |
| Column | Function |
| USERNAME | Stores the usernames of the users who can access the application. |
| PASSWORD | Stores the passwords of the users who can access the application. |

Table 3.2Table tblUser

Whenever the user wants to access the application the above table will be checked for the corresponding username and password only after which access will be granted.

3.2.1.2 Table tblClient

The Client table will consist of the following columns-

|  |  |
| --- | --- |
| Column | Function |
| RecordID | Unique ID which will be provided to each and every client. |
| ClientCode | Stores the unique code of the client. |
| Name | Stores the name of the client. |
| ClientAddr | Stores the address of the client. |
| PinCode | Stores the pincode of the client. |
| ClientState | Stores the state in which the client stays. |
| Country | Stores the country in which the client stays. |
| PhoneNumber | Stores the contact no. of the client. |
| DoB | Stores the date of birth of the client. |
| Age | Stores the age of the client. |
| Sex | Stores the gender of the client. |
| Nationality | Stores the nationality of the client. |
| MaritalStatus | Stores the marital status of the client. |
| Occupation | Stores the occupation of the client. |
| BankName | Stores the name of the bank in which client holds an account. |
| AccountNumber | Stores the account no. of the client |

Table 3.3 Table tblClient

Whenever any information regarding a client has to be viewed a tblClient Table can be accessed.

3.2.1.3 Table tblMaster1

The Text table will consist of the following columns-

|  |  |
| --- | --- |
| Column | Function |
| SerialID | Unique ID for each tuple. |
| ClientCode | Stores unique code for each client. |
| Exchange | Stores quantity of shares bought. |
| Symbol | Stores the scrip symbol of the company for which the transaction is done. |
| CompanyName | Stores the name of the company. |
| BuySell | Integer indicating buy or sell type of transaction. |
| BQty | Number of shares that have been bought. |
| BAvg | Stores the average cost at which the shares are bought. |
| BValue | Stores the product of BQty and BAvg. |
| SQty | Number of shares that have been sold. |
| SAvg | Stores the average cost at which the shares are sold. |
| SValue | Stores the product of SQty and SAvg. |
| NetValue | For buy transactions stores –(BValue) .  For sell transactions stores SValue . |
| TimeDate | Stores the date & time at which the transaction took place . |

Table 3.4 Table tblMaster1

This table stores the transactions done by the clients be it on a day to day basis, or weekly or monthly.

3.3 Design Details

3.3.1 Level 0 DFD

|  |
| --- |
|  |

Figure 3.1:Level 0 DFD providing an complete overview of the system

3.3.2 Level 1 DFD

|  |
| --- |
|  |

Figure 3.2:Level 1 DFD showing different users who are going to access the system.

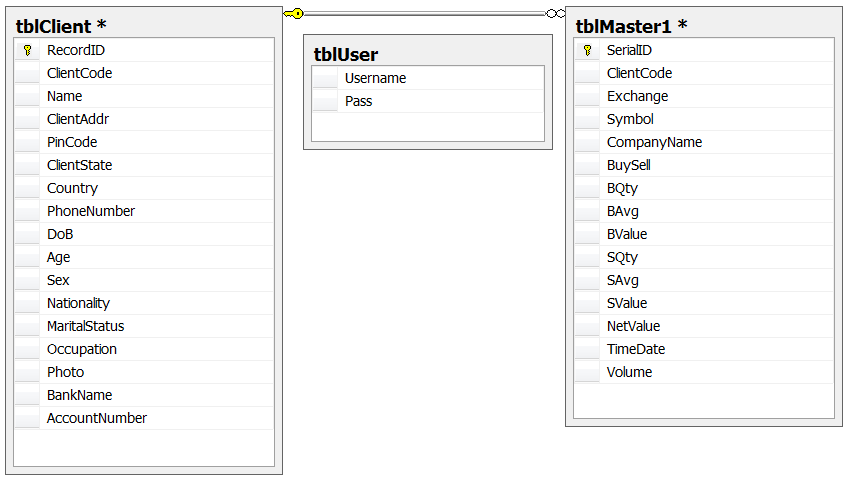


Figure 3.3: Class Diagram showing the different tables present in the system.

3.4 GUI Design



Figure 3.4: Screenshot of the Login Screen



Figure 3.5: Screenshot of the Create New User Screen



Figure 3.6: Screenshot of the Menu Screen

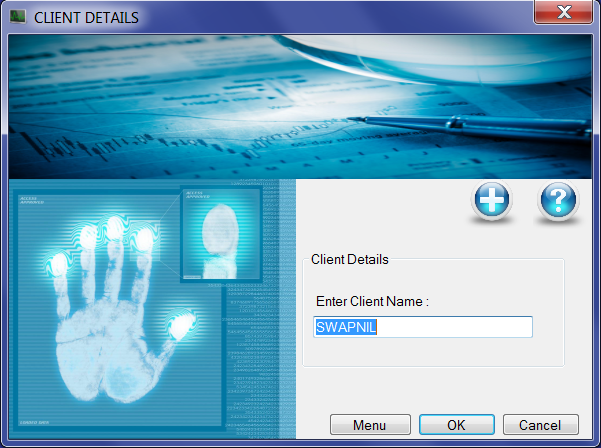


Figure 3.7: Screenshot of the Client Details\Add Client Screen

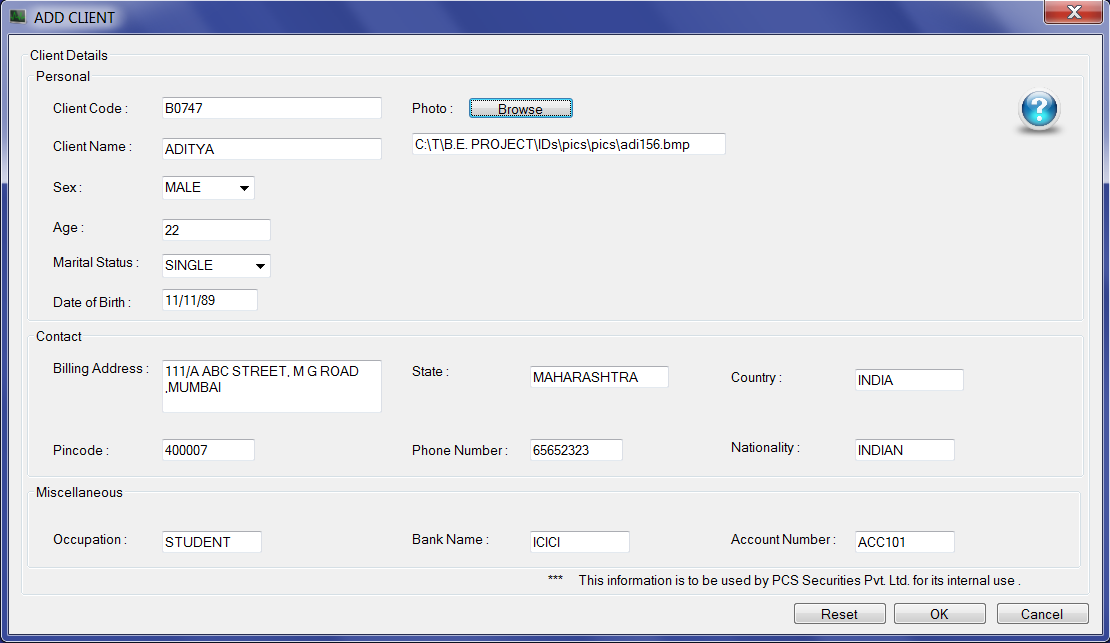


Figure 3.8: Screenshot of the Add Client Screen



Figure 3.9: Screenshot of the Client Details Screen

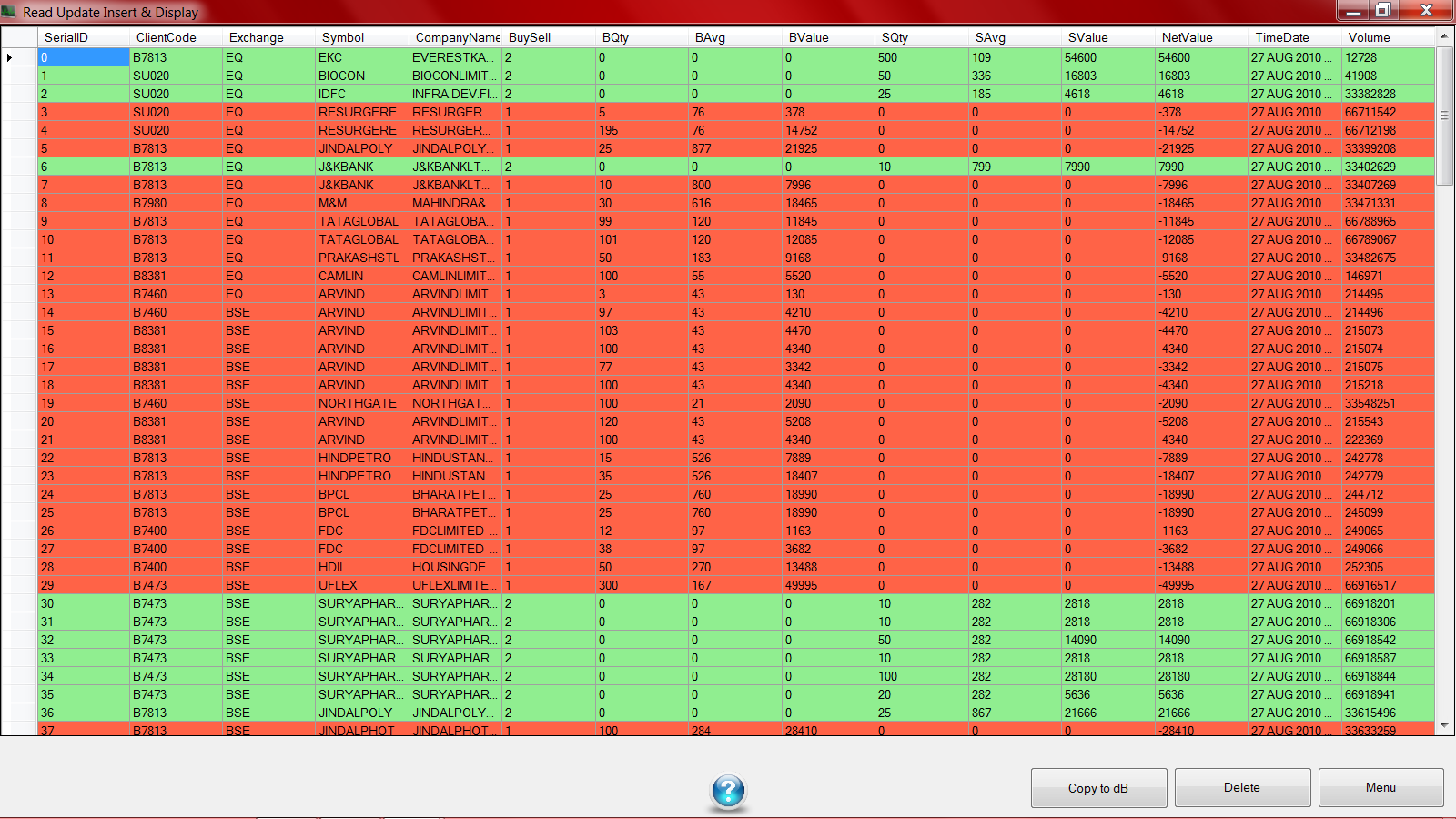


Figure 3.10: Screenshot of the Copy To Database Screen

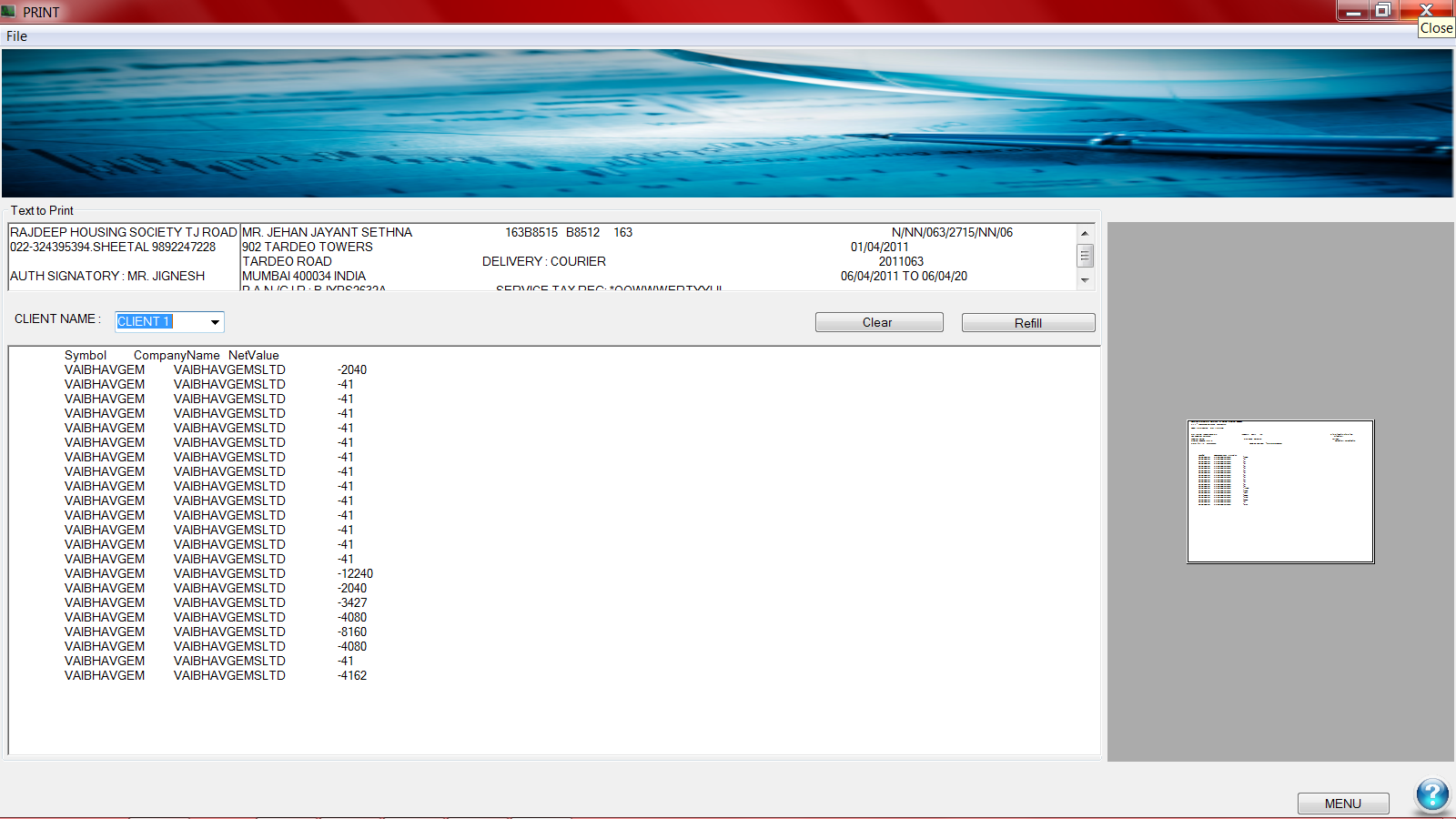


Figure 3.11: Screenshot of the Contract Print Screen

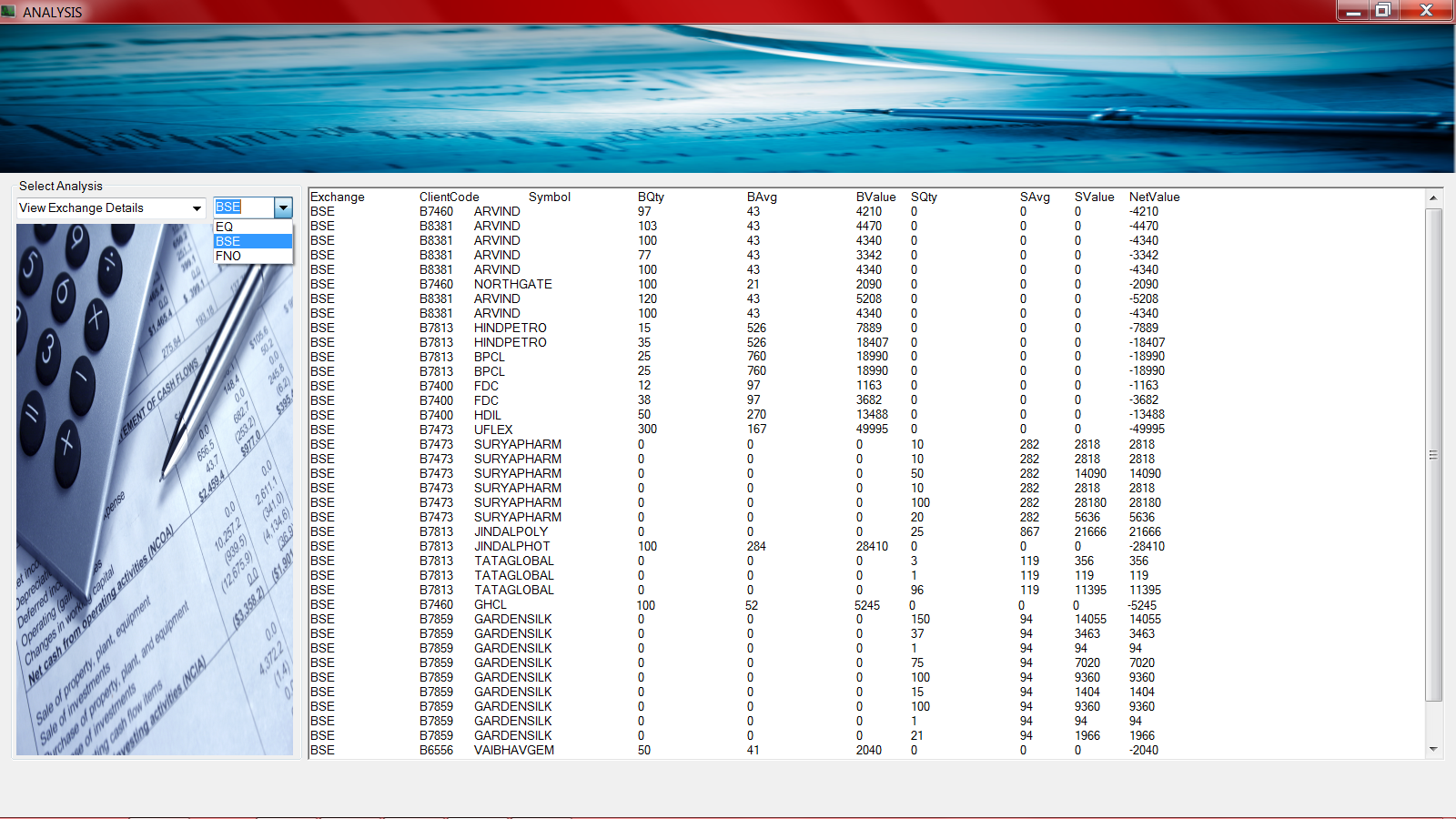
****

Figure 3.12: Screenshot of the Analysis Screen

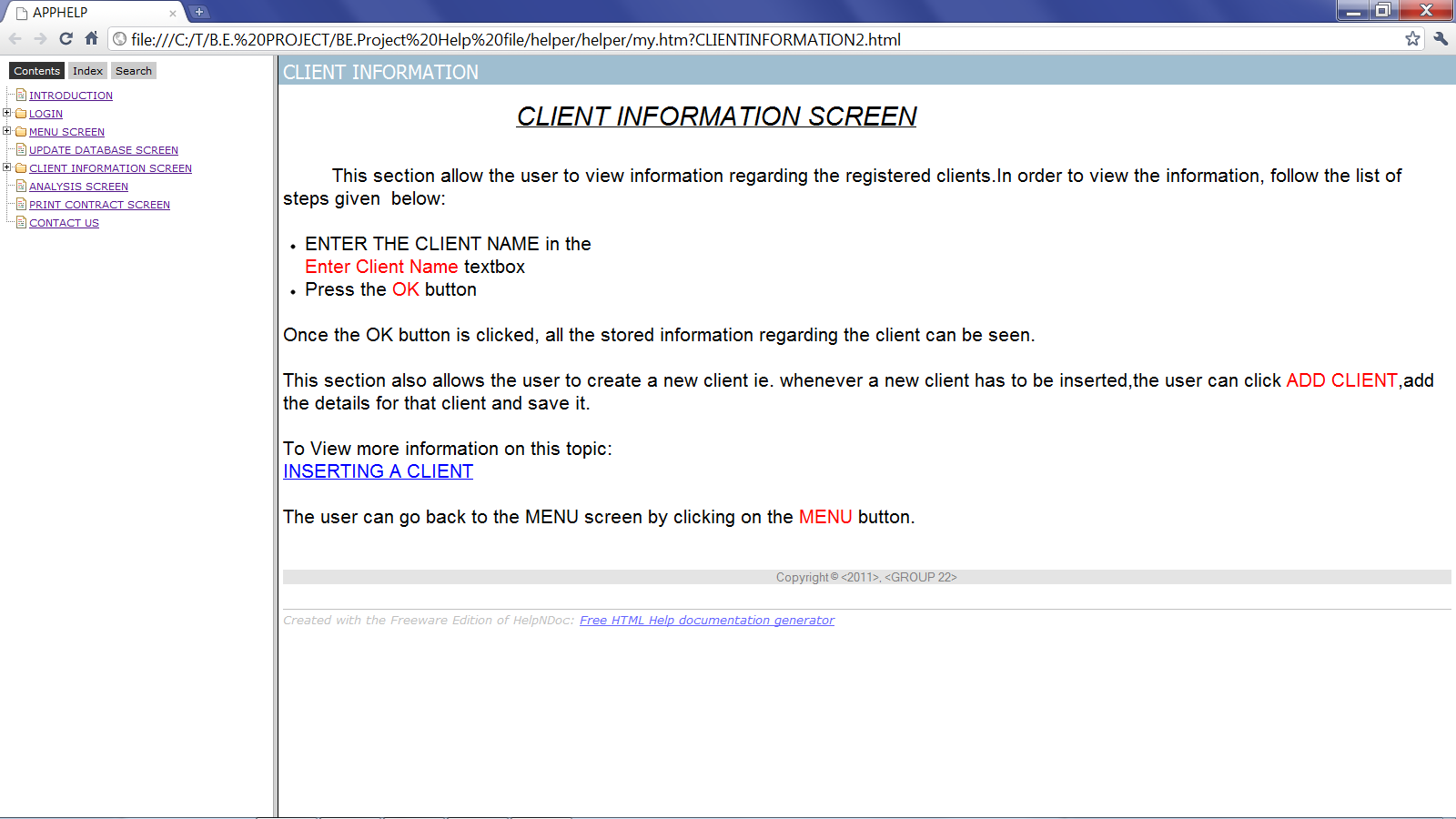
****

Figure 3.13: Screenshot of the Help File

**4 IMPLEMENTATION**

4.1 IMPLEMENTATION DETAILS

4.1.1 Week 1 –3 : Module 1- Copy to Master Database

This module required us to read the transaction file i.e, the text file provided by our client into application. Once the file is read at the click of a button, the data is copied  into the master database stored in the SQL Server.

Code:

PrivateSub copyBtn\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles copyBtn.Click

Try

Dim filename AsString

Filename = "C:\Users\Tirth\Documents\VisualStudio 2008\Projects\TradeAnalysis\TradeAnalysis\bin\Debug\MTRADE.txt"

If File.Exists(filename) Then

ioline = iofile.ReadLine

WhileNot ioline = ""

Dim mysplit = Split(ioline, ",")

f1 = mysplit(0)

f2 = mysplit(1)

f23 = mysplit(22)

f24 = mysplit(23)

/\*Comment\*/Read from file line by line and split at “,”( comma )

comm.CommandText = "INSERT INTO [tblMaster1](SerialID, ClientCode, Exchange, Symbol, CompanyName, BuySell, BQty, BAvg, BValue, SQty, SAvg, SValue, NetValue, TimeDate) VALUES('"& t &"','"& f15 &"','"& f4 &"','"& f3 &"','"& f5 &"','"& f11 &"','"& 0 &"','"& 0 &"','"& 0 &"','"& f12 &"','"& f13 &"','"& f12 \* f13 &"','"& f12 \* f13 &"','"& f20 &"');"

EndWhile

insert into tblMaster1 table columnwise only the required values

Dim r AsInteger

For r = 0 To dg.Rows.Count - 1

If dg.Rows(r).Cells(11).Value > 0 Then

dg.Rows(r).DefaultCellStyle.BackColor = Color.LightBlue

Else

dg.Rows(r).DefaultCellStyle.BackColor = Color.LightPink

EndIf

Next

Color logic based on netvalue(skip this if you want)

Else

MsgBox(filename + "does not exist!")

EndIf

Catch ex As Exception

MsgBox(ex.ToString)

EndTry

Error msg if file not present at specified location

EndSub

4.1.2 Week 4 & 5 : Module 2 - Help File

This module solves the question of “how to ?”. It deals with the documentation of the project .This help file will provide our client with an easy to use manual to navigate the application. A software called as ‘helpndoc’ was used to create this module which was then successfully integrated into our application. This module took one week extra as the original approach which we tried didn’t work as intended.

Code:

PrivateSub LinkLabel1\_LinkClicked(ByVal sender As System.Object, ByVal e As System.Windows.Forms.LinkLabelLinkClickedEventArgs)Handles LinkLabel1.LinkClicked

Process.Start("C:\T\B.E. PROJECT\BE.Project Help file\help.chm")

EndSub

Display the mentioned help file created by 3rd party software

4.1.2 Week 6-9 : Module 3- GUI & Analysis

This module includes two important aspects of our application, the GUI and the analysis part. The GUI designing included designing the different screens/interfaces which will be provided to the user. The analysis page included a dropdown box consisting of the different details which are required for analyzing the data provided in the transaction file. This module has taken the longest time since the GUI is being continuously modified as per the requirements of our client.

Code:

PrivateSub QueryCombo\_Load(ByVal sender As System.Object, ByVal e As System.EventArgs) HandlesMyBase.Load

ComboBox1.Items.Add("Query 1:Client Code")

ComboBox1.Items.Add("Query 2:Pin Code")

.

.

Items displayed in dropdown box

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

SelectCase ComboBox1.SelectedIndex

Case 0

Dim queryString AsString = "SELECT \* FROM tblMaster1;"

TextBox1.Clear()

Query to be executed

Dim command AsNew SqlCommand(queryString, cs)

cs.Open()

Dim reader As SqlDataReader = command.ExecuteReader()

Dim fNextResult AsBoolean = True

Execute query

DoUntilNot fNextResult

TextBox1.AppendText(vbTab & reader.GetName(0) &" "& reader.GetName(1) & vbTab & reader.GetName(2) & vbTab & reader.GetName(3) & vbTab & reader.GetName(4) & Environment.NewLine)

DoWhile reader.Read()

TextBox1.AppendText(vbTab & reader.GetString(0) & vbTab & reader.GetString(1) & vbTab & reader.GetString(2) & vbTab & reader.GetString(3) & vbTab & reader.GetString(4) & Environment.NewLine)

Loop

fNextResult = reader.NextResult()

Loop

Append result in textbox

4.1.2 Week 10 & 11 : Module 4- Print Contract

This module primarily deals with printing the details ofthe transactions done by a client on a day to day basis. This module requires to enter only the client no. and transaction date ,after which a list of transactions done by the client can be printed.

Code:

PrivateSub PrintDocument1\_PrintPage(ByVal sender AsObject, ByVal e As System.Drawing.Printing.PrintPageEventArgs) Handles PrintDocument1.PrintPage

'PrintPage is the foundational printing event. This event gets fired for every

' page that will be printed

Static intCurrentChar As Int32

' declaring a static variable to hold the position of the last printed char

Dim font AsNew Font("Verdana", 8)

' initializing the font to be used for printing

Dim PrintAreaHeight, PrintAreaWidth, marginLeft, marginTop As Int32

With PrintDocument1.DefaultPageSettings

' initializing local variables that contain the bounds of the printing area rectangle

PrintAreaHeight = .PaperSize.Height - .Margins.Top - .Margins.Bottom

PrintAreaWidth = .PaperSize.Width - .Margins.Left - .Margins.Right

' initializing local variables to hold margin values that will serve

' as the X and Y coordinates for the upper left corner of the printing

' area rectangle.

marginLeft = .Margins.Left

marginTop = .Margins.Top

' X and Y coordinate

EndWith

If PrintDocument1.DefaultPageSettings.Landscape Then

Dim intTemp As Int32

intTemp = PrintAreaHeight

PrintAreaHeight = PrintAreaWidth

PrintAreaWidth = intTemp

' if the user selects landscape mode, swap the printing area height and width

EndIf

Dim intLineCount As Int32 = CInt(PrintAreaHeight / font.Height)

' calculating the total number of lines in the document based on the height of

' the printing area and the height of the font

Dim rectPrintingArea AsNew RectangleF(marginLeft, marginTop, PrintAreaWidth, PrintAreaHeight)

' initializing the rectangle structure that defines the printing area

Dim fmt AsNew StringFormat(StringFormatFlags.LineLimit)

'instantiating the StringFormat class, which encapsulates text layout information

Dim intLinesFilled, intCharsFitted As Int32

e.Graphics.MeasureString(Mid(RichTextBox1.Text, intCurrentChar + 1), font, New SizeF(PrintAreaWidth, PrintAreaHeight), fmt, intCharsFitted, intLinesFilled)

' calling MeasureString to determine the number of characters that will fit in

e.Graphics.DrawString(RichTextBox1.Text, font, Brushes.Black, 20, 0)

'Draws the specified text string at the specified location with the specified [Brush](http://msdn.microsoft.com/en-us/library/system.drawing.brush.aspx) and [Font](http://msdn.microsoft.com/en-us/library/system.drawing.font.aspx) objects.

EndSub

4.1.2 Week 12 & 13: Module 5 – Final Review

This module deals with final checking and review of the system. It also deals with checking the flow of the system i.e. whether the links to the corresponding pages are working or not. It also deals with checking of whether the output matches the requirements or not.

4.2 Implementation Issues

This section lists out the various technical problems that were faced during implementation and decisions that were taken to tackle with the issues.

Help File:

The original approach which we tried in order to get the functionality of the help file was by using the ‘tree view option’. But since individuals categories couldn’t be selected we looked for an alternative and found a software called as ‘helpndoc’ which suited our needs perfectly. Initially the integration of this file also proved to be a problem but was later successfully solved.

Contract Printing:

The main problem faced during this module was of positioning the data. Our first approach of placing all the data into a single textbox didn’t provide the correct  
results. We then decided to use different textboxes which resulted in the problem of arrangement and alignment of the text on the copy to be printed.

Login:

At first, the login page looked the easiest of all,since we had done it before. But  Visual Basic 2008 turned out to be a slightly bigger test. Since  SQL is case-insensitive finding a solution for lowercase and uppercase characters created a problem.

Solution:

The solution to the above Login problem is the use of ‘COLLATE SQL\_Latin1\_General\_CP1\_CS\_AS’ in the login query. It is case sensitive collation which differentiates between A-Z & a-z.

**5 TESTING**

System testing is an expensive but critical process that can take as much as 50% of the budget for the problem development. Therefore the most useful and practical approach is with intention of finding errors that can make the program fail.

5.1 Levels of testing

Systems are not designed as entire systems nor are tested as single systems. The analyst must perform following testing methods.

5.1.1 Unit testing

In unit testing the analyst tests the program making up the system. The various system modules and routines are assembled and integrated to perform specific functions. This project has undergone the unit testing phase of every module for error free working.

5.1.2 System testing

System testing does not test the system entirely but rather the integration of each module in the system. This project has been integrated with all the modules to work as a single system and errors after integration process are removed.

5.1.3 Functional testing

It is sub phase of unit testing in which various functions of a module are tested independently.

5.1.4 Data testing

Data testing is an integral part of all above testing methods. In data testing various validations are tested on user data which user may enter. Different types of data are entered for various fields to check the validations. The various validation testing is as follows:

Test Case 1:

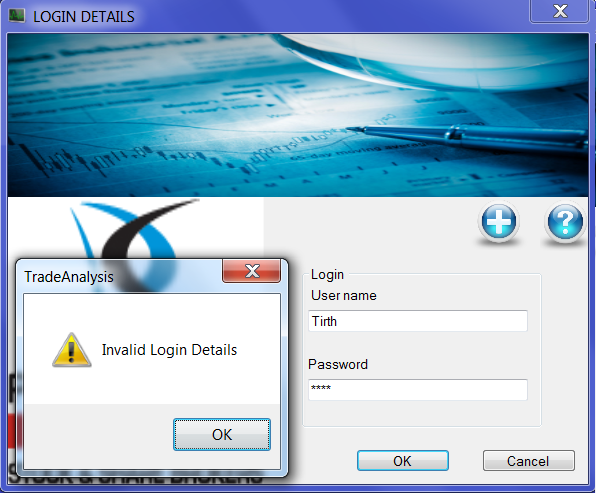


Figure 5.1: Enter valid username & password

Test Case 2:

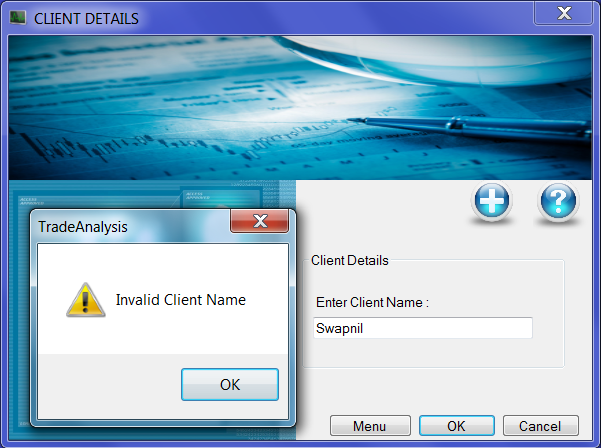


Fig 5.2: Enter valid Client Name

Test Case 3:

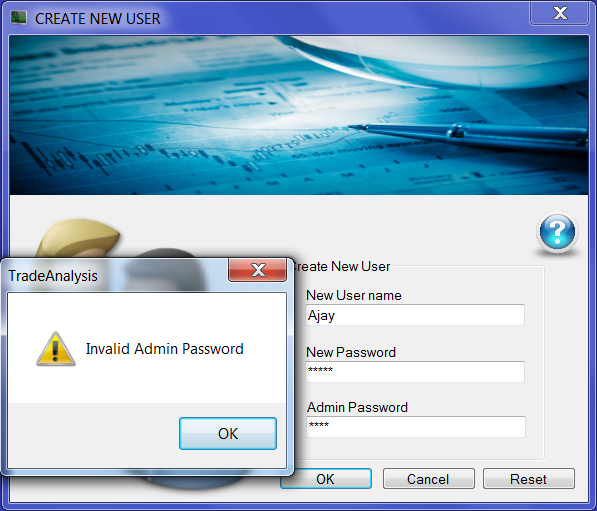


Fig 5.3: Enter valid Admin Password

# 6 CONCLUSION AND FURTHER WORK

6.1 CONCLUSION

As part of our semester 8 syllabus, we have successfully implemented our project titled DATABASE FOR TRADED STOCK ANALYSIS for our client PCS Securities Pvt. Ltd. The project has been implemented as per the requirements of our client, with useful suggestions provided by us, some useful, some not. This project has helped us to gain educational knowledge in terms of a new language vb.net, a new database software SQL Server 2008 and also gaining a substantial amount of knowledge about the world of the share market.

6.2 FURTHER WORK

As per the original requirement provided by our client PCS Securities Pvt. Ltd., all the requirements have been met. However for the analysis part, additional requirements are expected from the client, which will be met as soon as we receive them. In addition to this, we were planning to implement the concept of triggers. Our aim was to set a trigger which would send an email to the client whenever he had to pay an amount greater than Rs 50000/-.