

**SMS APPLICATIONS:
SHORT MESSAGING & RESULT CHECKING SYSTEM
(SMRCS)**

LAU SOOK WAI

**FACULTY OF COMPUTER SCIENCE & INFORMATION
TECHNOLOGY
UNIVERSITY OF MALAYA
KUALA LUMPUR**

2002/2003

Abstract

With the advanced technologies in the Internet, SMS is growing faster and getting more popular among Internet users. Communication through the Internet is not only faster but also more convenient and time saving. Users no longer conveyed important message through simple chat on the telephone or a face-to-face act, but it can be carried out on a special simulator through the Internet.

Short Messaging and Result Checking System (SMRCS) is develops with the objective of convenience, efficient, and cost-effective. This report focuses on building an online simulator, which provide more functions and features. It contains the conceptual design for the project, the research done on various field related to the project and also methodology used to develop the software.

The system architecture of SMRCS is divided into web client, web server and database. The client sends message by connecting and pass request to the web server that will interact with the database server. The database server will process the request and result is sent back to the web server and finally to the recipient.

SMRCS will be developed using Microsoft Visual Studio .NET Professional with Visual Basic.NET on Microsoft Windows 2000 Professional platform with Internet Information Server (IIS) as the web server, utilizing database created by Microsoft SQL Server 2000. At the end, this project will implement the entire requirements into a real system and strength and weakness of the system will be analyzed.

Acknowledgement

First and foremost, I would like to express my heartiest thank you and sincere gratitude to my project supervisor, Mr. Ang Tan Fong for his valuable advice, constructive ideas and patience throughout my project. This project would not be a success without his assistance and guidance.

I would also like to thank my project moderator, Dr. Rosli Salleh for his thought, ideas and guidance in improving my project.

Special thanks must be extended for my course mates for their thoughtful comments and ideas.

Finally, I also would like to thank everyone who has guided me in developing this system whose name that I did not mention.

Contents

	Page
Abstract	ii
Acknowledgement	iii
Contents	iv
List of Tables	viii
List of Figures	ix
 Chapter 1 : Introduction	 1
1.1 Overview of SMS	1
1.2 Overview of SMRCS	3
1.3 Project Motivation	4
1.3.1 Limitation of SMRCS	4
1.3.2 Advantages of SMRCS	5
1.3.3 Features of SMRCS	7
1.4 Project Objectives	8
1.5 Project Scope	9
1.6 Expected Outcome	10
1.7 Project Schedule	11
1.8 Overall Structure of Thesis Report	14
 Chapter 2 : Review of Literature	 17
2.1 Analysis Studies	18
2.1.1 Case Study 1 – Webportal SMS	18
2.1.1.1 Result of Study	18
2.1.2 Case Study 2 – The Simulator of C – SMS	20
2.1.2.1 Result of Study	21
2.1.3 Case Study 3 – National University of Singapore	23
2.1.3.1 Result of Study	24
2.2 Software Architecture	25
2.2.1 Mainframe Architecture	25
2.2.2 Client Server Architecture	26
2.2.2.1 Advantages of Client Server	27
2.2.2.2 Disadvantages of Client Server	28
2.2.3 Two-Tier Architecture	29
2.2.3.1 Advantages of two-tier architecture	30
2.2.3.2 Disadvantages of two-tier architecture	31
2.2.4 Three-Tier Architecture	31
2.2.4.1 Advantages of three-tier architecture	32
2.2.4.2 Disadvantages of two-tier architecture	33
2.2.5 Conclusion for Software Architecture	35
2.3 Web server	35
2.3.1 Apache Web Server	35
2.3.2 Internet Information Server (IIS) v5.0	36
2.3.3 Netscape Enterprise Server	37
2.4 Operating System	38
2.4.1 Unix	38
2.4.2 Windows 2000	40

2.4.3 Linux	42
2.5 Database Server	42
2.5.1 Database Management System (DBMS)	43
2.5.2 Oracle	43
2.5.3 MySQL	44
2.5.4 Microsoft SQL Server 2000	45
2.6 Data Access Technology	46
2.6.1 OLE DB	46
2.6.2 ODBC (Open Database Connectivity)	47
2.6.3 ADO.NET	48
2.6.4 Java Database Connectivity (JDBC)	49
2.7 Language	50
2.7.1 C++	50
2.7.2 VB.Net	51
2.7.3 Microsoft Visual Basic 6.0	53
2.7.4 Java	54
2.8 Web Development Tools	55
2.8.1 Visual Interdev	55
2.8.2 Microsoft Front Page	56
2.8.3 Visual Studio .NET Professional	56
2.9 Summary of Chapter 2	62
Chapter 3 – Methodology	63
3.1 Methodology	63
3.1.1 Spiral Model	64
3.1.2 V Model	67
3.2 Summary of Chapter 3	70
Chapter 4 : System Requirement Analysis	71
4.1 Techniques Used to Define Requirements	71
4.1.1 Brainstorming	71
4.1.2 Discussions	71
4.1.3 Internet research	72
4.1.4 Library research	72
4.2 Requirement Analysis	73
4.2.1 Functional Requirements	75
4.2.1.1 User Management Module	75
4.2.1.2 Record View Module	75
4.2.1.3 Result Module	75
4.2.1.4 Top Up Module	75
4.2.1.5 Authentication Module	76
4.2.1.6 Communication Module	76
4.2.1.7 Phone Book module	76
4.2.1.8 Registration Module	76
4.2.1.9 User Top Up Module	77
4.2.1.10 Model Selection Module	77
4.2.2 Non – Functional Requirements Analysis	77
4.2.2.1 Security	77
4.2.2.2 User Friendliness	78
4.2.2.3 Reliability	78

4.2.2.4	Availability	78
4.2.2.5	Accuracy	78
4.2.2.6	Performance	79
4.2.2.7	Flexibility	79
4.2.2.8	Efficiency	79
4.3	Hardware Requirement	80
4.4	Chosen Platform, Database and Tools	81
4.4.1	Chosen Development Platform	81
4.4.2	Chosen Database Management System	82
4.4.3	Chosen Application Development Tool	82
4.4.4	Chosen Development Language	84
4.5	Summary of Chapter 4	85
Chapter 5 : System Design		86
5.1	Introduction	86
5.2	Overview of System Architecture	87
5.3	Process Design	88
5.3.1	System Structure Charts	89
5.3.2	Data Flow Diagram	92
5.3.3	Process Flowcharts	98
5.4	Database Design	101
5.4.1	Data Dictionary	101
5.5	User Interface Design	105
5.6	Summary of Chapter 5	108
Chapter 6 : System Implementation		109
6.1	Introduction	109
6.2	Development Environment	109
6.2.1	Hardware Configurations	109
6.2.2	Software Configurations	110
6.3	System Development	111
6.3.1	Data presentation	111
6.3.1.1	Still Images	111
6.3.1.2	Database Preparation	111
6.3.1.3	Input Form Design	112
6.3.2	Coding	112
6.3.2.1	Introduction	112
6.3.2.2	Coding Principles	114
6.4	Summary of Chapter 6	116
Chapter 7 : System Testing		117
7.1	Introduction	117
7.2	Types of Testing	119
7.2.1	Unit Testing	119
7.2.2	Module Testing	120
7.2.3	Integration Testing	121
7.2.4	System Testing	121
7.2.4.1	Security Testing	122
7.2.4.2	Performance Testing	122
7.3	Test Case	124

7.4 Summary of Chapter 7	126
Chapter 8 : System Evaluation	127
8.1 Problem Encountered and Solutions	127
8.1.1 Difficulties In Determining The Scope of the System	128
8.1.2 Problems in Choosing Language and Tools	128
8.1.3 Lack of Knowledge in Language and Tools	129
8.2 System Strength	129
8.2.1 User-friendliness	129
8.2.2 Password Protected Members and Administrator Sites	129
8.2.3 System Transparency	130
8.2.4 Reliable System with Efficient Errors Handling	130
8.2.5 Validation of Input Data	130
8.2.6 Dynamic Database Access Capability	131
8.3 System Constraints and Future Enhancements	131
8.4 Knowledge and Experience Gained	132
8.4.1 Time Management	132
8.4.2 Documentation	133
8.4.3 Improving skills in Programming, Database Design, System Analyzing and Designing	133
8.5 Summary of Chapter 8	134
Appendix A	135
Appendix B	159
Reference	197

List of Tables

Table 4.1 : Hardware and Software Requirements	80
Table 5.1 : DFD Symbols	93
Table 5.2 : User Profile	102
Table 5.3 : Admin Profile	102
Table 5.4 : Result	103
Table 5.5 : Message Storage	103
Table 5.6 : Server Storage	103
Table 5.7 : Card ID	103
Table 5.8 : Balance	104
Table 5.9 : User Contact	104
Table 5.10 : Credit List	104
Table 6.1 : Software Specifications	110
Table 7.2 : Performance Testing Table	123
Table 7.2 : Test Case	124

List of Figures

Figure 1.1 : Project Schedule	13
Figure 2.1 : The Main Page of Webportal SMS	18
Figure 2.2 : The Simulator of C – SMS	20
Figure 2.3 : The Main Page of National University of Singapore	23
Figure 2.4 : Message flow in two-tier client/server architecture	30
Figure 2.5 : Three Tier Architecture	34
Figure 2.6 : The Components of Microsoft .NET-Connected Software	57
Figure 3.1 : The Spiral Model	65
Figure 3.2 : The V Model	68
Figure 4.1 : The Process of Determining Requirements	73
Figure 5.1 : Three Tier Architecture of SMRCS	88
Figure 5.2 : Structure Chart of SMRCS	89
Figure 5.3 : Administrators' Structure Chart	90
Figure 5.4 : Users' Structure Chart	91
Figure 5.5 : Context Level Diagram of SMRCS	94
Figure 5.6 : Diagram 0 of SMRCS	95
Figure 5.7 : Level 1 of registration	96
Figure 5.8 : Level 1 of Authentication	96
Figure 5.9 : Level 1 of sending message	97
Figure 5.10 : Level 1 of Topup Account	97
Figure 5.11 : Flow Chart of SMRCS	99
Figure 5.12 : Flow Chart of Sending SMS	100
Figure 5.13 : Main Page of SMRCS	106

Figure 5.14 : Login page SMRCS	106
Figure 5.15 : Registration page of SMRCS	107
Figure 7.1 : The Process of System Testing	118

Universiti Malaya

Chapter 1 : Introduction

The Short Message Service (SMS) allow transmission of short messages in hundred of octets between two mobile stations. The service makes use of short message service center to store and forward the short messages. SMS is commonly used for voice mail notification and information services such as weather and stock data update. SMS is widely deployed in analog and digital wireless network such as Global System for Mobile Communications (GSM), Code Division Multiple Access (CDMA), Narrowband Advanced Mobile Phone System (NAMP) and integrated Dispatched Enhanced Network (iDEN).

1.1 Overview of SMS

SMS allow mobile user to receive and send alphanumeric messages via their mobile phones. SMS first appeared on the wireless scene in early 1991 shortly after the GSM standard was finalized where short-messaging services were included from the outset. The service makes use of a short message service center (SMSC in GSM standard) to store and forward the short messages. The main function of SMSC s to collect, store, manage and forward alphanumeric messages for delivery to the mobile station. The wireless network provides the transport for messages between the SMSC and the mobile phones.

One of the distinguish characteristics of the service is that an active mobile phone is able to receive or submit a short message at any time, independent of

whether or not a voice or data call is in progress. SMS also guarantees delivery of the short messages by the network. If temporary failures are encountered, the short message is stored in SMSC until the destination becomes available. Generally, the use of SMS in a network increases the airtime through return call, and as a result, generates extra revenue for the operator.

Initial development of SMS focus on eliminating alphanumeric pagers by permitting two-way general purposed messaging and notification services, primarily for voice mail. As the technology and networks matured, a variety of services were added on, including fax integration, paging integration, Internet paging, electronic mail, information services such as weather and stock data update. SMS is also used to facilitate 'Over-The-Air' (OTA) technology, where customized SMS is used as commands to retrieve and download specific information to and from the subscriber identity module (SIM) card. For example, some countries in Europe are using OTA technology for provisioning of mobile stations and updating the preferred roaming list.

SMS can be broadly classified into two types. The point-to-point SMS allows the network to send and receive SMS to and from the mobile station. The second type of SMS called cell broadcast allows the network to broadcast a short message to multiple users within a cell or a region.

1.2 Overview of SMRCS

SMRCS referring to Short Messaging and Result Checking System, which enable users to easily communicate with each other. By using SMRCS, users can send short text message and students are able to check their exam results. It is a computerized system, which can be access from anywhere via the Internet. The data will be stored in a database. Data can be retrieve easily and efficiently when it is needed.

SMRCS is also designed to enable users to send unlimited text messages as long as they have enough credit in the account. Besides sending message, users can use the phone book to store contact information such as their phone number and contacts name.

Users are able to change the model of the simulator. There are a few designs for users to choose from. Besides the model, user can also change the skin color of the selected simulator too.

The message send using SMRCS cannot exceed 70 characters for non-Latin alphabets such as Chinese or Arabic and 160 characters if Latin alphabets are used. Since SMRCS is a computerized system, user can type the message quickly using a keyboard, instead of entering a message on your phone using small buttons.

1.3 Project Motivation

SMS is a popular communications tools because it can send the latest information to the user using a very fast and low-cost compared to the traditional methods. SMS application is designed to enable users to quickly and cost-effectively communicate and interact with each other. With Short Messaging and Result Checking System (**SMRCS**), user can send SMS and students are able to check their exam results too.

1.3.1 Limitation of SMRCS

SMRCS consists of a few limitations. Some of the major limitations are as below:

- ✦ **Text message**

Messages are plain in nature. User can only send simple text messages. There is no scope for any graphics or audio.

- ✦ **Limited size**

The messages are limited by size. Each SMS message cannot exceed 160 characters.

- ✦ **Suitability for WAP**

The store and forward nature of SMS, though useful in many applications makes SMS not very suitable for WAP.

✦ **Authorized user**

Only authorized user can sign in and use the application of the system.

✦ **Data rate and latency**

GPRS (General Packet Radio Services) and USSD (Unstructured Supplementary Service Data) provide better data rates and lower latency compared to SMS. This is because SMS uses the slow signaling channel, which is used for many other things also in GSM. GPRS offer fast short message delivery as packets data, supported by sophisticated signaling procedures in the core network.

1.3.2 Advantages of SMRCS

Below are the advantages of SMRCS:

✦ **Cost effective**

- SMS is an inexpensive method of communication.
- For normal SMS it will cost RM0.15 for every message sent and for checking result it will cost RM 0.30.

✦ **Better Communication**

- 160 characters take up as much room as a one-second-voice call.
- Easy to keep in touch with friends, family or colleagues.

✦ Efficiency

- Messages are delivered immediately when the recipients' log on to the system.
- Messages can be stored and delivered later if the recipients' status is not active.

✦ Bandwidth

- SMS message use little bandwidth and carriers do not have to deliver them in real time (like they do with voice transmission).

✦ Alternative

- An alternative to alphanumeric paging services.

✦ Storage

- Like e-mail, user can stored the message in the inbox and reviewed it whenever they want.

✦ User friendly

- By using graphical user interface and easy to use features, user can quickly adept and use the system even without any IT background.

1.3.3 Features of SMRCS

- ✦ **Performance** – the system must maintain satisfactory response time as database size and number of client terminals increases.
- ✦ **Security** – the system must include security mechanisms that prevent unauthorized access.
- ✦ **Scalability** – the ability of a computer application to continue function well as it (or its context) is changed in size or volume in order to meet the user need.
- ✦ **Text length** – A single short message can be up to 160 characters of text in length. Those 160 characters can comprise of words or numbers or an alphanumeric combination.
- ✦ **Store and forward service** – Short messages are not sent directly from sender to recipient, but always via a web server.
- ✦ **Confirmation of message delivery** – Unlike paging, users do not simply send a short message and trust and hope that it gets delivered. Instead the sender of the short message can receive a return message back notifying them whether the short message has been delivered or not.
- ✦ **No limitation** – Send unlimited text message as long as there are enough credits in the user account.
- ✦ **Convenience** – Instead of using small keypads on mobile phone, user can type in text message using the keyboards of the computer.
- ✦ **Choices** – Beside mobile phones, user has another option for sending SMS.
- ✦ **Sender Identification** – Identify the identity of the sender so the recipient can reply the message easily.

1.4 Project Objectives

Core objectives of the project are as below:

✦ Computerized system

- The project aim is to provide a computerized system for sending message messages to the people around the world.
- User can type message quickly using a keyboard, instead of entering a message on your phone using frustratingly small buttons.
- To send text messages to a personal computer that are capable of sending and receiving SMS messages.

✦ Better Communication

- To offer a cost effective way and a simple method of sending text messages programmatically to a personal computer within a computer application.

✦ Authentication

- The system provides appropriate access control for user and administrator.
- To increase the security user can change the password from time to time.

✦ Efficient

- Once the user is log on to the system, they can send message to the person who is using the same system and receive new message too.
- It gives an easier way for students to check their exam result no matter where they are as long as they are connected to the Internet.

✦ Effective

- Information is send to the person who is intended to get the message.

1.5 Project Scope

SMS Application can be divided into two major parts, which are the System Administration Section and the User Section. The System Administration Section is to manage the system such as handling the requests from the clients, give approval to users to use the system, updating exam result databases, and manage the users account balance. The User Section allow the users to register, login, send text messages to other users who is using the same application, top up their account and even check their exam results.

The project scope for **System Administration Sections** includes:

- ✦ Develop a server module to handle the requests from the client
- ✦ Develop a server module that is capable to communicate with the client module through the network sockets
- ✦ Develop a web-based management system to manage the users who are using the system
- ✦ The administrators can update or retrieve the data that stored inside the database
- ✦ Administrators add the users to the system by approving their registration and provide them with a username and password in order for them to access the system
- ✦ Develop a exam result checking module that is able to send a short message back to the user with correct input such I.C number and Matriculation number
- ✦ Develop a top up module that is able to check the validity of the top up card before updating the users account balance

The project scope for **User Section** includes:

- ✦ Develop a client module that enables the users to send their SMS messages
- ✦ Develop a client module that is capable to communicate with the server module through the network socket
- ✦ Develop an application to enable students to check their examination result remotely from anywhere
- ✦ Users can register as a member of the system and will have a password and a unique username given to them

1.6 Expected Outcome

Expected outcomes for System Administration Section include:

- ✦ A control that can approve user application and remove users
- ✦ Input and update the exam result into the database
- ✦ Send exam result back to the user with matching IC number and Matriculation Number
- ✦ Check top up card validity before updating the users' account
- ✦ Able to change the password from time to time to increase security

Expected outcomes for User Section include:

- ✦ Registration form for new user
- ✦ A simulator
- ✦ Able to send short text message

- ✦ Able to check exam result by giving the correct I.C number and Matriculation number
- ✦ Able to top up the balance in the account in order to send SMS
- ✦ User-friendly graphical user interface that is easy-to-use
- ✦ User can log in to use the applications in the system and log out when the user want to end the session
- ✦ Able to change the password from time to time to increase security

1.7 Project Schedule

In planning for the SMRCS, a project schedule was designed for the development of the system. This is important so that the system can be implemented according to the time given. The project schedule is the operating timetable of the project. It serves as the fundamental basis for monitoring and controlling project activity. In a project environment, paper scheduling function is important because projects lack the continuity of day-to-day operations and often present much more complex problem of coordination. [Meredith & Mantel,1995]

The proposed project will be carried out in two stages, which each stage has to be completed within the period of one semester respectively. The work involve in the first stage in the project planning, research, analysis and design. The next stage involves the actual development of the system, testing and implementation.

The project schedule for developing the SMS Applications System is shown in figure 1.1. The project was schedule using Gantt Chart. Gantt Chart is chosen because it can be prepared to help schedule tasks. They show when tasks should begin and end, what tasks can be undertaken concurrently, and what tasks must proceed serially. They also help identify the consequences of early or late completion of a task. [Weber,1999]

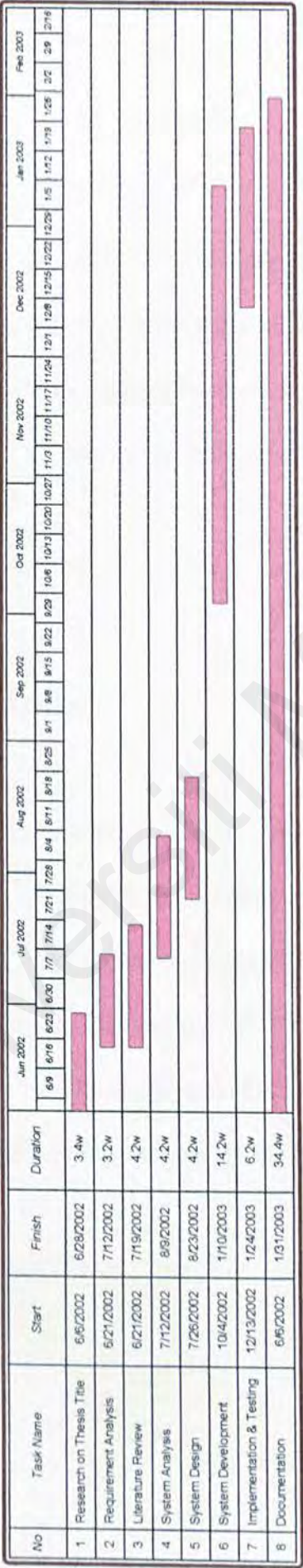


Figure 1.1 : Project Schedule

1.8 Overall Structure of Thesis Report

Chapter 1 : Introduction

This chapter gives a brief overview regarding SMRCS. There are certain purposes, objectives and scope that must be fulfilled while developing this system in order for the system to meet the requirements of user and administrators. Finally the project schedule serves as a guideline of various activities that will be taken in order to complete this project.

Chapter 2 : Literature Review

Chapter 2 lays out the process of literature review being undertaken to develop the proposed system. An analysis of the existing systems provides a framework for better understanding of the concepts, techniques and materials used in the construction of those systems. Comparison of the existing system provides in one way or another, gives an insight of the strengths and weaknesses of those systems, thus leading to a much appropriate features for the proposed system.

Chapter 3 : Methodology

This chapter drills down into methodologies that help to speed up and simplify the information system development process. Several models will be discussed in brief, namely the waterfall model, the spiral model and the V Model.

Chapter 4 : System Requirement Analysis

Chapter 4 explains the information gathering techniques needed to obtain relevant information regarding the system. Also included in the research and investigation on the system requirement analysis, which consist of functional requirements and non-functional requirements, based on similar systems available in the market. Hardware and software is another key factors that should be considered during the process of system design to ensure compatibility between users and the burden of technology integration and technical support.

Chapter 5 : System Design

This chapter looks at the system design where it documents the process of program design, input form design, graphical user interface design and database design.

Chapter 6 : System Implementation

Chapter 6 explains about the system implementation, which refers to transferring, previously planned module and algorithm into instructions than can be run using certain programming language.

Chapter 7 : System Testing

Chapter 7 makes sure the system can function according to certain requirements and specifications.

Chapter 8 : System Evaluation

This chapter evaluates the end results, problems and solutions, system constraints and future enhancements, suggestions and the summary of the whole project.

Chapter 2 : Review of Literature



- 2.1 Analysis Studies
- 2.2 Software Architecture
- 2.3 Web Server
- 2.4 Operating System
- 2.5 Database Server
- 2.6 Data Access Technology
- 2.7 Language
- 2.8 Web Development Tools
- 2.9 Summary of Chapter 2

Chapter 2 : Review of Literature

A literature review of a project is important as it places the project in the context of others, which might have similar characteristics. It helps the developer to know some of the existing feature offered by a similar system.

There is no use of reinventing the wheel that has already been invented. The developer can rather focus on learning the existing system and modify or enhance it into a more advance and powerful feature for the project.

Another important purpose of a literature review is to sufficiently equip the developer with some knowledge of the strengths and limitation of several development tools. This can help the developer to choose the right tool in system development.

2.1 Analysis Studies

2.1.1 Case Study 1 – Webportal SMS



Figure 2.1 : The main page of Webportal SMS

2.1.1.1 Result of Study

This site is the main page for Webportal SMS as shown in figure 2.1. User who wants to use the system would have to register online and the approval will be done within one week. Reload coupon for Digi, maxis, TMTouch, Celcom and TIMEcel are sold here.

Besides sending short text message, Webportal SMS offer other services such as:

- ✦ Search directory
- ✦ Online chat
- ✦ E-mail
- ✦ E-Greetings
- ✦ Auction
- ✦ Forum

Some main modules of the site are as below:

- ✦ The SMS module
 - Users are able to send SMS online to Maxis and Celcom mobile phones.
- ✦ The Forum module
 - Allow users to create new discussion topic and reply to the topic that have been created.
- ✦ The Auction module
 - Users can sell new or used items over the net.
 - Users can bid on the auction lot.

Strength:

- ✦ Simple design make it fast to load
- ✦ The main page has a clear view, nice layout and usage of space
- ✦ Website can also be viewed in Chinese
- ✦ Include many other services and quick links such as movies, weather and horoscopes
- ✦ Allow user to send free SMS to a real mobile phone

- ✦ It is user-friendly because it does not involve too many steps to send a SMS

Weaknesses:

- ✦ Only 110 characters are allowed for each message
- ✦ Allow user to send SMS to Maxis and Celcom mobile only
- ✦ Without a help page to guide the user on how to use the system

2.1.2 Case Study 2 – The Simulator of C – SMS



Figure 2.2 : The simulator of C – SMS

2.1.2.1 Result of Study

Figure 2.2 shows one of the systems used to send SMS. In order to use C – SMS, user may need to download the simulator. It will automatically install in the PC. User can send different kind of SMS such as Arabic SMS, Flash SMS, and Group Messaging.

C-SMS has a lot of features such as:

- ✦ Sending SMS
- ✦ Address Book
- ✦ Outbox
- ✦ Archiving
- ✦ Settings
- ✦ Integration with Internet Explorer
- ✦ C-SMS news ticker

Some main modules of the simulator are as the following:

- ✦ Sending SMS module
 - First enter Contact's phone number, or if he/she is already in your address book, click on his/her name and the phone number will be entered automatically. If the number is not found in the address book, C – SMS will tell notified user that it is an Unknown Contact. After sending the SMS, C – SMS will ask if user wants to add this new contact to the Address Book.

- ✦ Group Messaging module

- This module enables the user to send a single SMS to many contacts at a single click.

Strength:

- ✦ The Strip or Squeeze functions can be used when user want to put more text in the SMS but there is no space left
- ✦ Users can send unlimited SMS to mobile phone through the Internet
- ✦ Remind users to send SMS to a particular person every X days with a check mark next to a person's name in users contact's list
- ✦ When user is offline, they can store the SMS in the outbox and send it out when they are connected to the Internet
- ✦ Able to remind users to send SMS to particular person every X days with a check mark next to a person's name in users contact's list
- ✦ Users can view their previous message
- ✦ C-SMS will automatically add itself to Internet Explorer's Toolbar after it is downloaded
- ✦ A small address book is integrated in it. It allows user to store the phone numbers of their contacts
- ✦ The system provide a counter that will show how many characters remains for sending a single SMS

Weaknesses:

- ✦ Limited by size – an SMS message cannot exceed 70 characters
- ✦ The design is quite confusing and the layout not attractive. It is a bit crowded because there are too many frames in the page
- ✦ Involve a lot of steps to send a single SMS
- ✦ The users need to download the system before they can use it
- ✦ The takes a long time to load because the system has many graphic

2.1.3 Case Study 3 – National University of Singapore

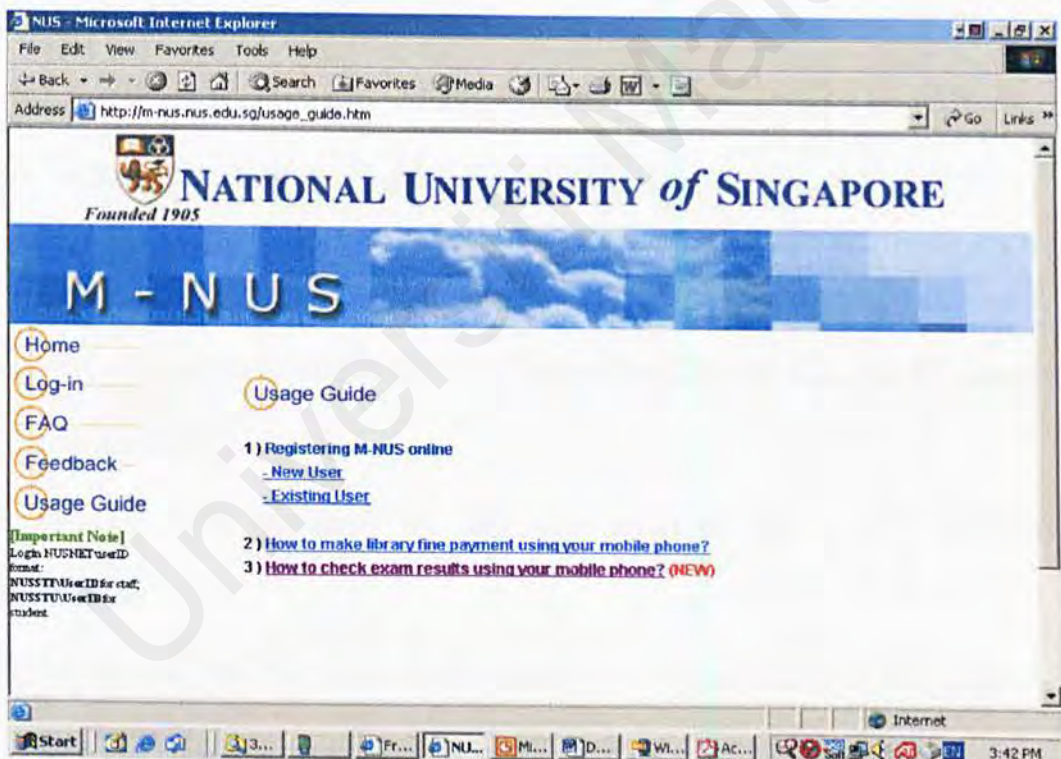


Figure 2.3 : The main page of National University of Singapore

2.1.3.1 Result of Study

M-NUS enables users to use their mobile phones to carry out system transactions. Using the WAP phones, staff can query their leave balances, and students can query their exam results.

Future mobile phone services that are coming soon includes viewing and replying of email, checking of appointments and contacts on Outlook all with just the mobile phone.

To further harness the SMS capabilities of SMS, NUS has taken part in a consortium (led by NCS) that enables users to make payment to participating merchants using mobile phones. Mobile payment services offered by NUS (e.g. payment of library fines for overdue books) will be rolled out in phases.

Strength:

- ✦ User can use SMS to bid for items like collectable watches, mobile phones and DVD players
- ✦ Installed SSL certificate on the web portal to secure your personal information
- ✦ For security reasons, exam results query can only be made for the particular user that is tied to unique mobile phone number
- ✦ User can use SMS to pay library fines
- ✦ User-friendly. The users are guided step-by-step on how to check their exam result
- ✦ Enable users to use their mobile phones to carry out system transactions

Weaknesses:

- ✦ Payment of services offered can be made only with NETS Virtual Card
- ✦ It takes too many steps to check their exam result. User waste a lot of time and money
- ✦ The system did not provide a help page to assist the user

2.2 Software Architecture

There are a few software architectures available: mainframe architecture, client-server architecture, two-tier architecture and three-tier architecture.

2.2.1 Mainframe Architecture

In mainframe system architecture, all operations are within the central host computer. User interacts with the host through a terminal that captures keystroke and sends that info to the host. Mainframe architecture is not tied to a hardware platform. User interaction can be cloned using PCs and UNIX workstations. A limitation of mainframe architecture is that it does not easily supports graphical user interface or accesses to multiple databases from graphically dispersed sites.

2.2.2 Client Server Architecture

The client/server model is an approach to software in which one application (the client) asks for and receives services from another application (the server). Client/Server is a concept of computing as seen from the end user's viewpoint - not that of the system or the application. In a client/server environment, data are manipulated at the user level. Client/server computing can be considered totally user-driven and the client/server environment can be envisioned as a multi-vendor, multi-product, multi-application implementation. Essentially, a client/server is a software-based architecture that enables distributed computing resources on a network to share common resources among groups of users at intelligent workstations. [1]

Client/server computing is an environment that satisfies the business need by appropriately allocation the application processing between the client and the server processors. The client requests services from the server; the server processes the request and returns the result to the client. The communications mechanism is a message passing interprocess communication (IPC) that enables (but does not require) distributed placement of the client and server processes. Client/server is a software model of computing not a hardware definition. Though client/server architecture can be very complex, they are generally speaking, two kinds of client/server infrastructures to choose from. They are two-tier and three-tier. The choice between two-tier and three-tier architecture should be based on the scope and complexity of a project, the time available for complexion and the expected enhancement or obsolescence of the system. [1]

2.2.2.1 Advantages of Client Server

Client/server is an open system. It offers organizations the ability to distribute processing and data across networks using powerful graphical workstations, servers and mainframes, the client/server enables rightsizing, the selection and location of computing resource according to the needs of the individuals and work groups. One of the prime benefits of a client/server system is a lower cost. Another is increased productivity from the individual to the corporation that results from better access to information and the distribution resources thorough the corporation. Additional benefits of client/server include:

- ✦ Interoperability – key components (client/server/network) work together
- ✦ Scalability – any of the key elements may be replaced when the need to either grow or reduce processing for that element dictates without major impact on the other elements
- ✦ Adaptability – new technology may be incorporated into the system.
- ✦ Cost effectiveness – Using less expensive MIPs that are available on each platform insures affordability
- ✦ Data Integrity – entity, domain and referential integrity are maintained on the database server
- ✦ Performance – performance may be optimized by hardware and process
- ✦ Security – data security is centralized on the server

2.2.2.2 Disadvantages of Client Server

Although client/server computing provides innovative solutions for a number of businesses, it may not be the right solution for all the flexibility of client/server systems and the complexity of networking require careful strategic planning up front.

Other disadvantages are:

- ✦ The hardware, software and communication technology is neither mature nor entirely stable, nor easy to assemble
- ✦ Because client/server is not well understood it is frequently sold inappropriately or oversold to management and unsatisfied expectations result
- ✦ Support cost can run three times the prices of the system hardware and software
- ✦ Redesign and reprogramming are not trivial exercises
- ✦ Backup and recovery in a client/server environment can be expensive
- ✦ The more the distributed the systems, the greater its vulnerability
- ✦ Client/server is an evolving technology and as such there is no standardization

In theory, client/server looks great; it allows an organization to rapidly create graphical applications that reflect changing business needs. Underneath the surface, however are unexpected costs that can make client/server systems more expensive to operate than centralized host-based systems are. [1]

2.2.3 Two-Tier Architecture

The two-tiered architecture contains two components – a client and a server with areas of logic combined on the client (shown in figure 2.4). The three components of an application-presentation, processing and data are divided among two client/server entities or tiers: client application code and database server. A robust client application development language and a versatile mechanism for transmitting client requests to other server are essential for a two-tier implementation.

Presentation is handled exclusively by the client, processing is split between client and server and data is stored on and accessed through the server. The PC client assumes the bulk of responsibility for application (functionality) logic with respect to the processing component, while the data base engine with its attendant integrity checks, query capabilities and central repository functions handles data intensive tasks. In a data access topology, a data engine would process requests sent from the clients.

Currently, the language used in these requests is most typically a form of SQL. To send the SQL, the client must know the syntax of the server or have it translated by an API (application program interface). Data returned to the client can be manipulated at the client level for further sub selection, business modeling, what-if-analysis and reporting.

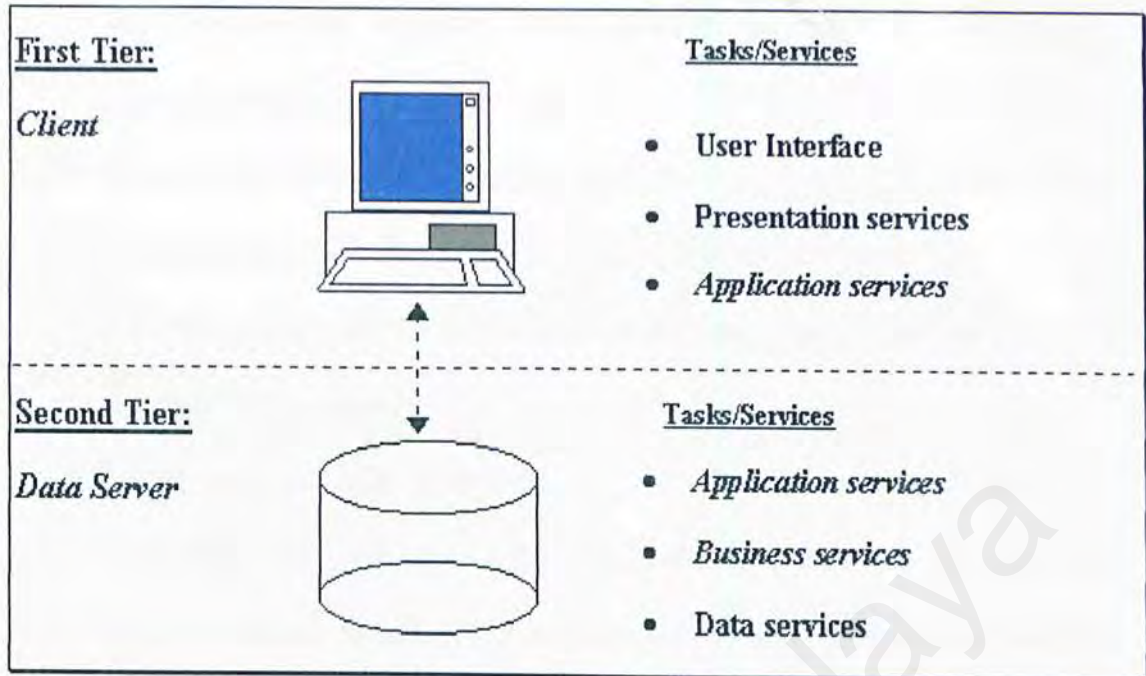


Figure 2.4 : Message flow in two-tier client/server architecture

2.2.3.1 Advantages of two-tier architecture

- ✦ Fast application development time
- ✦ Available tools are robust and lend themselves to fast prototyping to insure user need a met accurately and completely
- ✦ Conducive to environments with homogeneous clients, homogeneous applications and static business rules

2.2.3.2 Disadvantages of two-tier architecture

- ⊕ Not suitable for dispersed, heterogeneous environments with rapidly changing business rules
- ⊕ Because the bulk of the application logic is on the client, there is the problem of client software version control and new version redistribution
- ⊕ Security can be complicated because a user may require separate password for each SQL accessed
- ⊕ Client tool and SQL middleware in two tier environments tends to be proprietary. The volatility of the client/server tool market raises questions about the long-term viability of any proprietary tool. Organizations should be wary about committing to proprietary tools

2.2.4 Three-Tier Architecture

The components of three-tier architecture are divided into three layers: a presentation layer, a functionality layer and the data layer (shown in figure 2.5). Each of these layers must be logically separate. The three-tier architecture attempts to overcome some of the limitations of the two-tier scheme by separating presentation, processing and data into separate distinct entities.

The same types of tools can be used for presentation as were used in a two-tier environment, however the tools are now dedicated to handling just the presentation. When the presentation client requires calculation or data access, a call is made to a middle-tier functionality server. This tier performs calculations or makes requests as a client to additional server. This middle-tier server are typically coded in

a highly portable, nonproprietary language such as C. Middle-tier functionality servers may be multithreaded and can be accessed by multiple clients, even those from separate applications. Although three-tiers systems can be implemented using a variety of technologies, the calling mechanism from client to server in such a system is most typically the remote procedure call or RPC. Because the bulk of two-tier implementations involve SQL messaging and most three-tier system utilize RPCs, examination of the merits of these respective request or response mechanisms is warranted.

2.2.4.1 Advantages of three-tier architecture

- ✦ RPC calls from presentation client to middle-tier server provide greater overall system flexibility than the SQL calls made by clients in the two-tier architecture. This is because in RPC, the requesting client simply passes parameters needed for the request and specifies a data structure to accept returned values
- ✦ Unlike in most two-tier implementations, the three-tier presentation client is not required to understand SQL. As such, the organization names or even the overall structures of the back-end data can be changed without requiring changes to PC based presentation clients. Because SQL is no longer required, data can be organized hierarchically, relationally or in object format. This added flexibility allows a firm to access legacy data and simplifies the introduction of new data base technologies
- ✦ Having separate software entities allows for the parallel development of individual tiers by application specialists

- ✦ Provides for more flexible resource allocation. Middle-tier functionality servers are highly portable and can be dynamically allocated and shifted as the needs of the organization change. Network traffic may be reduced by having functionality servers strip data to the precise structure required before distributing it to individual clients of the local area network (LAN) level
- ✦ Modularly designed middle-tier code modules can be reused by several applications. Reusable logic reduces subsequent development efforts, minimizes the maintenance workload, and decreases migration costs when switching client applications
- ✦ Three-tier systems such as Open Software Foundation's Distributing Computing Environment (OSF/DCE) offer a variety of additional features to support distributed application development

2.2.4.2 Disadvantages of two-tier architecture

- ✦ Three-tier brings with it an increased need for network traffic management, server load balancing and fault tolerance
- ✦ Current tools are relatively immature and require more complex 3GLs for middle-tier server generation. Maintenance tools have underdeveloped facilities for maintaining server libraries a potential abstract for simplifying maintenance and promoting code reuse throughout in IS organization

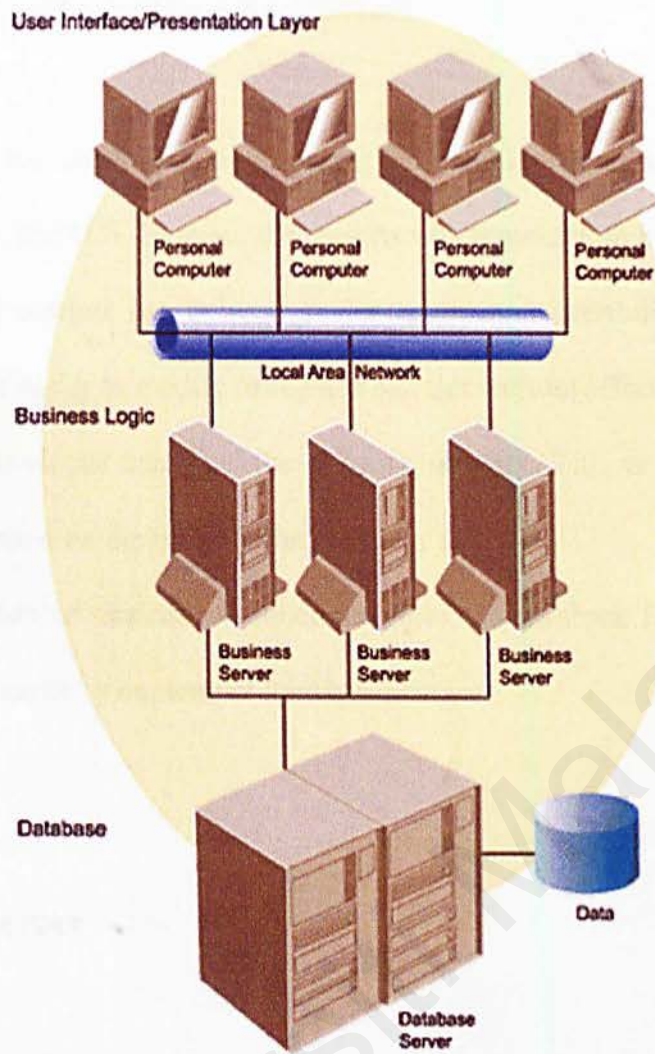


Figure 2.5 : Three tier Architecture

2.2.5 Conclusion for Software Architecture

For all the client/server computing architecture, three-tier architecture has been chosen for SMRCS. Here are the reasons why three-tier architecture is chosen:

- ✦ The architecture can deliver greater application scalability. The modularity makes it easier to modify or replace one tier without affecting the other tiers
- ✦ Each developer enhances the network security. This is to avoid fraud and interception on the information
- ✦ Separation of application functions from the database functions makes the system easier to implement load balancing

2.3 Web server

2.3.1 Apache Web Server

Apache server is a powerful yet flexible web server. It is compliant with HTTP/1.1 and implements the latest protocols, including HTTP/1.1 (RFC 2616). Apache Web Server is highly configurable as it is open source code and extensible with third-party modules and can be customized by writing 'modules' using the Apache module API. Moreover, it provides full source and comes with an unrestrictive license.

Apache web server runs on Windows NT/9x, Netware 5.x, OS/2 and most versions of UNIX as well as several other operating systems. The web server is

actively being developed and encourages user feedback through new ideas, bug reports and patches. The features of Apache Web Server include DBM databases for authentication, customizable responses to errors and problem, multiple Directory Index directives and unlimited flexible URL rewriting and aliasing. It is compatible with Windows 2000, NT, Linux, Netware 5.0 (with Service Pack 5), 5.1 (with Service Pack), UNIX, BSD, HP MPE/iX 6.0 or higher and TPF version 4.1 PUT09.

However there are limitations in Apache Web Server. To corporate web server customers, the fact that Apache is free can be a drawback signifying a lack of the explicit or implied accountability they get with vendor products. Its flexibility also can be a double edge sword. Apache is easy to setup, but for those who try to extend it had better know what they are doing. Where there is not a lot of expertise available, customers may also prefer to see features that come together and have been tested together rather than search them out from multiple sources on the Internet. The lack of software support for Apache has been confined to online resources. [Bowen & Coar, 2000]

2.3.2 Internet Information Server (IIS) v5.0

Internet Information Service 5.0 (IIS) is the Windows 2000 Web Service that makes it easy to publish information on the intranet for the Internet. It is completely integrated with Windows NT Directory Services and includes Crystal reports, a visual reporting tool. Internet Information Server 5.0 has many new features to enables user to create a scalable and flexible web applications. It allows administrators to configure servers, sites, virtual directories, subdirectories and files

individually. It also includes crash protection that allows users to run multiple applications reliably. Moreover, IIS includes tools to analyze and manage web server content and supports multiple web sites on one IP address.

2.3.3 Netscape Enterprise Server

The Netscape Enterprise Server software runs on a representative collection of operating systems: AIX, Digital Unix, HP_UX, Irix, Solaris and Windows NT. It provides a powerful development environment that support development of web-based applications that can be run on the Internet, an intranet or an extranet. Netscape Enterprise Server's content management allows user to create their own Netshares, personal home directories using an interesting method that provides services including link management, web publishing, agent services and access and version control. [2]

Like most other server programs, Netscape Enterprise Server supports dynamic application development including CGI and Netscape's own version of application program interface: Netscape Server API (NSAPI). Netscape Enterprise Server supports the Java Servlet API for Server-side applications. A Netscape product called liveware runtime environment is included in Netscape Enterprise Server and allow user to write server-side scripts that among other things, provide connectivity to databases, including Oracle, Sybase and Infomix. It's ODBC conformance means that Netscape Enterprise Server provides connectivity to other database sources as well. [Scheined & Perry, 2000]

2.4 Operating System

2.4.1 Unix

Unix is an increasingly popular operating system and it is traditionally used on minicomputers and workstations in the academic community. Unix is now available on personal computer and the business community has started to choose Unix for its openness.

Unix can run on multiple platforms and the minimum requirements are vary depending on platform chosen. There are several advantages about Unix that enables it to become one of the popular operating system among large organization. These include [2]

- ✦ **High performance** – Unix is the choice for high performance network applications. It will outperform other operating system when running on equivalent hardware. Unix is used by Yahoo, USWest and Xooms.com as their main server's operating system because of its ability to handle network traffic with high performance
- ✦ **High reliability** – Unix is extremely robust because the new file system optimizes disk input and output for high performance. It also ensures reliability for transaction-based application such as databases
- ✦ **Good development environment** – Unix includes an extensive collection of development tools such as C/C++, java, HTTP, Perl and Python. All of these are free, come in full source code and are included in the installation

Although Unix can be considered one of the popular operating system but it also have a few disadvantages such as [2]

- ✦ **Expensive** – Unix is very expensive compared to other operating system. All Unix machine are also very expensive because it is specially designed only for Unix
- ✦ **User friendliness** – The interface in Unix is based on command-line interface (CLI) and it requires user to type specific command in order to execute any applications or instructions. Many users are not familiar with CLI, so it is quite difficult for them to use Unix. Although Unix had developed a few Graphical User Interface (GUI) but it is still not as complete and friendly as the Windows desktop
- ✦ **Installation problem** – Many users will face problem during installation because the installation process needs the concept of disk partitioning and mounting of the systems, which are relatively an advanced concept for new users. The users also have to know the details of the graphics adapter card and monitor in order to provide the information the installation project requires
- ✦ **Difficult to configure and maintain** – Unix is difficult to configure and maintain because it requires the users to type a set of specific commands for configuration and maintenance. The configuration is not guided with any wizard or GUI interface

2.4.2 Windows 2000

Windows 2000 is the latest commercial version of Microsoft evolving Windows operating system (although a new version, Windows XP is now available). Previously called Windows NT 5.0, Microsoft emphasizes that Windows 2000 is evolutionary and “Built on technology”. Windows 2000 is designed to appeal to small business and professional users as well as to the more technical and larger business market for which the NT was designed. For many Windows 95 and Windows 98 users, Windows 2000 may be regarded as a step to take when purchasing their next computer.

The Windows 2000 product lines consist of four products:

- ✦ Windows Professional, aimed at individuals and businesses of all sizes. It includes security and mobile use enhancements. It is the most economical choice
- ✦ Windows 2000 Server, aimed at small-to-medium size businesses. It can function as a Web Server and/or a workgroup (or branch office) server. It can be part of a two-way symmetric multiprocessing system. NT 4.0 servers can be upgraded to this server
- ✦ Windows 2000 Advanced Server, aimed at being a network operating system and/or an application, including these involving large databases. This server facilitates clustering and load balancing. NT 4.0 servers with up to eight-way SMP can upgrade to this product
- ✦ Windows 2000 Datacenter Server, designed for large data warehouses, online transaction processing (OLTP), econometric analysis and other applications

requiring high speed computation and large databases. The Datacenter Server supports up to 16-way SMP and up to 64 gigabytes of physical memory

Windows 2000 is reported to be more stable (less apt to crash) than Windows 98/NT systems. A significant new feature is Microsoft's Active Directory, which among other capabilities, enables a company to set up virtual private networks to encrypt data locally or on the network and to give users access to shared files in a consistent way from any computer network.

Notable features of the Windows 2000 products are:

- ✦ A fully customizable administrative console that can be based on tasks rather than files, applications or users
- ✦ A new file directory approach called Active Directory that lets the administrator and other users view every file and application in the network from a simple point-of-view
- ✦ Dynamic Domain Name Server (DNS), which replicates changes in the network using the Active Directory Service (WINS) whenever a client is reconfigured
- ✦ The ability to create, extend or mirror a disk volume without having to shut down the system and to back up data to a variety of magnetic and optical storage media
- ✦ A distributed file system (DFS) that lets user see a distributed set of files in a single file structure across departments, divisions or an entire enterprise

Close integration with and support for Microsoft's Message Queue Server, Microsoft Transaction Server and Internet Information Server (IIS).

2.4.3 Linux

Linux is a Unix like operating system that was designed to provide personal computer users a free or very low cost operating system compare to traditional and usually more expensive Unix system.

Linux is a remarkably complete operating system, including a graphical user interface, on X Window System, TCP/IP, the Emacs editor and other components usually found in a comprehensive Unix system. Unlike Windows and other proprietary systems, Linux is publicly open and extensible by contributors. Because it conforms to the Portable operating system Interface standard user and programming interfaces, developers can write programs that can be ported to other operating system. [2]

2.5 Database Server

A database is a collection of data that is organized so that it's contents can be easily accessed, managed and updated. The most prevalent type of database is the relational database, a tabular database in which data is defined so that it can be reorganized and accessed in a number of different ways. A distributed database is one that can be dispersed or replicated among different points in a network. An

object oriented programming database is one that is congruent with the data defined in object classes and subclasses.

2.5.1 Database Management System (DBMS)

A Database Management System (DBMS) is a program that lets one or more computer users create and access data in a database. DBMS also ensures the integrity and security of the data. The most typical DBMS is a relational database management system (RDBMS). A standard user and program interface is the Structured Query Language (SQL).

A DBMS can be thought of as a file manager that manages data in database rather than files in file systems. A DBMS is usually an inherent part of a database product. Microsoft Access is a popular example of a single or small-group user DBMS. Microsoft's SQL Server is an example of a DBMS that serves database requests from multiple (client) users.

2.5.2 Oracle

Oracle platform is available for multiple operating systems and research proved that Oracle runs great on Unix. Oracle is more standards-based as well with a set of neat features. Oracles databases are as powerful as the users want them to be. Oracle is able to efficiently utilize hardware platform and manage multiple high-speed processors, clustered servers, high bandwidth connectivity and fault tolerant

storage technology. Java application can run perfectly with combination of Oracle database.

Oracle also provides the users with more power and flexibility with the database to meet the user requirements. Oracle able to handle handles a rapidly expanding amount of users and/or data gracefully.

One of the disadvantages is oracle has weird concept and names as well. As a result, users have to undergo specialized training or knowledge to be more familiar with oracle database management; even the experts of DBMS, like Microsoft SQL Server and Microsoft Access. Oracle needs a costly start-up solution of database management. Besides, total ownership is high for Oracle.

2.5.3 MySQL

MySQL is an open source relational database management system (RDBMS) that uses Structured Query Language (SQL), the most popular language for adding, accessing and processing data in a database. MySQL is noted mainly for its speed, realibility and flexibility. It works best when managing the content and not executing transactions.

The MySQL relational database system is fully multi-threaded using kernel threads, provides application interfaces (APIs) for C, C++, Java, Perl and PHP allow

for many column types and offer full operator and function support in the SELECT and WHERE parts of queries.

2.5.4 Microsoft SQL Server 2000

As database grows and become more complex, there is some point when a Microsoft Access database should be should be upsized to a Microsoft SQL Server database. This is to optimize database and application performance, scalability, security, reliability, and availability. The following reviews on different areas in SQL server.

- a) High performance and scalability – SQL server can support very large database up to on terabytes in contrast to only 2 gigabytes for Access. SQL server is well integrated with Microsoft NT thus make it works efficiently on the platform. Besides SQL Server 2000 can run on the stand-alone laptop computer running Windows 95/98.
- b) Increased availability – dynamic backup can be carried out while the database is being used. Users do not need to exit from the database to do backup. Hence the database can be available at all times.
- c) Improved security – SQL server integrates with Windows NT that user only needs single log-on to the network and database. A user cannot use SQL Server without accessing to the network first. This results in better security and eases the administrator.
- d) Immediate recoverability – When a system suddenly breaks down, SQL server database can have automatic recovery mechanism that recovers the database to the last state of consistency without administrator intervention.

- e) Reliable distributed data and transactions – SQL server supports atomic transactions without transactions logging. This guarantees that all changes performed within a transaction are either committed or rolled back.
- f) Server-based processing – SQL server is designed as a client/server database residing on a server. It reduces network traffic by processing database queries first before sending them to clients. Processing is always done on the server-stored procedures and triggers are also supported to be processed on the server.

2.6 Data Access Technology

2.6.1 OLE DB

OLE DB is Microsoft's strategic low-level application program interface (API) for access to different data sources. OLE DB includes not only the Structured Query Language (SQL) capabilities of the Microsoft-sponsored standard data interface Open Database Connectivity (ODBC) but also includes access to data other than SQL data.

As a design from Microsoft's Component Object Model (COM), OLE DB is a set of methods (in earlier days, these might have been called *routines*) for reading and writing data. The objects in OLE DB consist mainly of a data source object, a session object, a command object, and a row set object. An application using OLE DB would use this request sequence:

1. Initialize OLE.
2. Connect to a data source.
3. Issue a command.
4. Process the results.
5. Release the data source object and uninitialize OLE.

OLE once stood for "Object Link Embedding" and "DB" for database. However, Microsoft no longer ascribes these meanings to the letters "OLE" and "DB."

2.6.2 ODBC (Open Database Connectivity)

Open Database Connectivity (ODBC) is an open standard application-programming interface (API) for accessing a database. By using ODBC statements in a program, you can access files in a number of different databases, including Access, dBase, DB2, Excel, and Text. In addition to the ODBC software, a separate module or driver is needed for each database to be accessed. The main proponent and supplier of ODBC programming support is Microsoft.

ODBC is based on and closely aligned with The Open Group standard Structured Query Language (SQL) Call-Level Interface. It allows programs to use SQL requests that will access databases without having to know the proprietary interfaces to the databases. ODBC handles the SQL request and converts it into a request the individual database system understands.

ODBC was created by the SQL Access Group and first released in September 1992. Although Microsoft Windows was the first to provide an ODBC product, versions now exist for UNIX, OS/2, and Macintosh platforms as well.

In the newer distributed object architecture called Common Object Request Broker Architecture (CORBA), the Persistent Object Service (POS) is a superset of both the Call-Level Interface and ODBC. When writing programs in the Java language and using the Java Database Connectivity (JDBC) application program interface, you can use a product that includes a JDBC-ODBC "bridge" program to reach ODBC-accessible databases.

2.6.3 ADO.NET

From an architect's perspective ADO.NET represents the abstract design concepts used to build the data access classes within the .NET Framework. There were several main design goals driving ADO.NET:

- ✦ Explicit and factored object model. ADO.NET is designed to be a simple to use object model in which the developer has complete control over how to control data source connectivity, command execution, and data manipulation
- ✦ Disconnected data cache model. N-tier programming and XML Web service architecture require that applications can participate in a disconnected, loosely coupled manner. ADO.NET provides a comprehensive caching data model for marshalling data between applications or services and then to optimistically update the original data sources or source
- ✦ XML support. XML is the key to building interoperable applications and more robust data processing models. XML support has been built directly

into the .NET Framework. ADO.NET leverages this implementation by providing a seamless interaction with XML in a relational manner or in a native XML manner

- ✦ Leverage existing ADO knowledge. Although the ADO.NET object model is different from the existing ADO model, the basic constructs are the same. The ADO.NET object model consists of a provider, connection, and command objects. Thus current ADO developers should be able to efficiently migrate to ADO.NET

From a developer's perspective ADO.NET represents the concrete implementation of classes inside the .NET Framework used for data access. [4]

2.6.4 Java Database Connectivity (JDBC)

Java Database Connectivity (JDBC) is an application program interface (API) specification connecting programs written in Java to the data in popular database. The application program interface lets you encode access request statements in Structured Query Language (SQL) that are then passed to the program that manages the database. It returns the result through a similar interface.

JDBC is very similar to the SQL Access Group's Open Database Connectivity (ODBC) and with a small bridge program, you can use the JDBC interface to access databases through the ODBC interface.

2.7 Language

2.7.1 C++

C, the predecessor to C++, has become one of the most popular programming languages. Originally designed for system programming, C enables programmers to write efficient code and provided close access to the machine. C compilers found on practically every UNIX system are now available with most operating system.

C++ represents a significant extension of C abilities. We might then consider C to be a subset of C++. C++ supports essentially every desirable behavior and most of the undesirable ones of its predecessor, but provides general language improvements as well as adding Object Oriented Programming capabilities. User can simply create structured code that uses only C++ non-Object Oriented Programming features.

C++ have many features of the C language, such as efficiency, closeness to the machine, and a variety of built in types. A number of new features were added to C++ to make the language even more robust, many of which are not used by novice programmers. Most of these features can be summarized by two important design goals: strong compiler type checking and a user-extensible language.

C++ also enables programmers to incorporate new types into the language, through the use of classes. A class is a user-defined type. The compiler can treat new types as if they are one of the built-in types. This is a very powerful feature. In

addition, the class provides the mechanism for data abstraction and encapsulation, which is the key to object oriented programming.

2.7.2 VB.Net

To rapidly build enterprise Web applications, developers must rely on business logic that is scalable, robust, and reusable. Over the past several years, object-oriented programming has emerged as the primary methodology for building systems that meet these requirements. Using object-oriented programming languages helps make large-scale systems easier to understand, simpler to debug, and faster to update.

To enable Visual Basic developers to benefit from object-oriented design and to simplify the development of enterprise Web applications, full object-oriented language features, including implementation inheritance, will be supported in the next version of Visual Basic - Visual Basic .NET. With these new language features, Visual Basic .NET will deliver all the power required to quickly and effectively develop enterprise-critical applications while maintaining the instant accessibility that has made it the world's most popular development tool.

Visual Basic .NET will provide a first class object-oriented programming language with new features such as implementation inheritance, overloading, and parameterized constructors. Additionally, developers will be able to create highly scalable code with explicit free threading and highly maintainable code with the addition of modernized language constructs like structured exception handling.

Visual Basic will provide all the language characteristics that developers need to create robust, scalable distributed Web applications with the following new features:

New object-oriented programming features

- ✦ Inheritance
- ✦ Overloading
- ✦ Parameterized Constructors

Additional modernized language features

- ✦ Free Threading
- ✦ Structured Exception Handling
- ✦ Strict Type Checking
- ✦ Shared Members
- ✦ Initializers

Visual Basic is now a first class object-oriented programming language. Using Visual Basic .NET, developers will be able to create highly scalable code with explicit Free Threading. The code they write will be highly maintainable with the addition of modernized language constructs like Structured Exception Handling. Visual Basic will provide all the language characteristics that developers need to create robust, scalable distributed Web applications.

2.7.3 Microsoft Visual Basic 6.0

The Microsoft Visual Basic development system version 6.0 is the most productive tool for creating high performance components and applications. Visual Basic 6.0 offers developers the ability to create robust applications that reside on the client or server, or operate in the distributed n-tier environment. Visual Basic 6.0 is the Rapid Application Development (RAD) tool available either as a standalone product or as a part of the Visual Basic 6.0 suite of tools.

Visual Basic can be used to develop components for any tier within a solution. On the presentation tier, Visual Basic can be used to author WIN32, Dynamic HTML or HTML-based clients; on the middle tier, Visual Basic can be used to author thread-safe components for use in the Microsoft Transaction Server; and on the data tier, Visual Basic can be used for database and schema design as well as for stored procedure authoring and debugging.

All areas of data access have been improved to make Visual Basic easier to perform most common database activities. The most significant features include these:

- ✦ Universal Data Access with integrated ADO/OLE database support
- ✦ Visual Database Tools integrated into the Visual Basic environment
- ✦ New Oracle schema and stored procedure design capabilities
- ✦ Data Environment Designer for authoring ADO-based data access components
- ✦ New Integrated Report Writer

- ✦ Hierarchical Flex Grid Control for displaying hierarchical data
- ✦ Ability to create the OLE Database Providers
- ✦ Ability to create Data Sources
- ✦ Ability to easily remove data from machine to machine , tier to tier
- ✦ Advance Data Binding

2.7.4 Java

Java is a programming language designed for use in the distributed environment of the Internet. It was designed to have the “look and feel” of the C++ language, but it is simpler than C++ and enforces an object oriented programming model. Java can be used to create complete applications that may run on a single computer or be distributed module or applet for use as part of a web page. Applets make it possible for a web page user to interact with the page.

The major characteristics of Java are:

- ✦ The programs created are portable in the network. The source program is compiled into what Java calls byte code, which can be run anywhere in a network or a server or client that has Java Virtual Machine. The Java Virtual Machine interprets the byte code into code that will run on real computer hardware. This means that individual computer platform differences such as instructions lengths can be recognized and accommodated locally just as the page is being executed. Platform-specific versions of program are no longer needed

- ✦ The code is robust, here meaning that unlike program written in C++ and perhaps some other language, Java objects can contain no references to data extended to themselves or other known objects. This ensures that an instruction cannot contain the address of data storage in another application or in the operating system itself, either of which would cause the program and perhaps the operating system itself to terminate or “crash”. The Java Virtual machine makes a number of checks on each object to ensure integrity
- ✦ Java is object oriented, which means that among other characteristics on object can take advantages of being part of a class of objects and inherit code that is common to the class. Objects are thought of as “nouns” that a user might relate to than the traditional procedure “verbs”. A method can be thought of as one of the objects capabilities or behaviors
- ✦ In addition to being executed at the client rather than the server, a Java applet has other characteristics designed to make it run fast
- ✦ Relative to C++, Java is easier to learn

2.8 Web Development Tools

2.8.1 Visual Interdev

Visual Interdev is Microsoft’s development tool for building a dynamic, data – driven Web site. Whereas Microsoft’s Front Page is an HTML editor aimed at letting non – programmers build the pages for a web site. Visual Interdev provides the tools for programmers to build a Web site. (Front Page and Interdev are said to

interoperable). Visual Interdev offers a user interface similar to those for Visual Basic, Visual J++ and Visual Studio. Using Visual Interdev, one can assemble pages that use Microsoft's ActiveX technologies including Active Server Pages (ASP) technologies. The developer can build and insert ActiveX control or Java applet. Visual Interdev includes an HTML editor and support for dynamic HTML. The web site can be integrated with server programs written in any language and access to almost any database using Microsoft's Universal Data Access including ActiveX Data Objects, Open Database Connectivity and OLE Database.

2.8.2 Microsoft Front Page

Microsoft Front Page adds value to creating Web application by adding the visual components that is missing from Visual Interdev. Microsoft Front Page enable users to quickly generate HTML and save a lot of time and frustration spent on getting complicated HTML page layout properly adjusted. After the page is created, users can edit the HTML source code to create the dynamic content on the page while relying on the HTML tags to quickly generate the look and feel of the page.

[Johnson, Balinger & Chapman, 1997]

2.8.3 Visual Studio .NET Professional

Defining the Basic Elements of .NET

Microsoft .NET is a set of Microsoft software technologies for connecting your world of information, people, systems, and devices. It enables an unprecedented

level of software integration through the use of XML Web services: small, discrete, building-block applications that connect to each other—as well as to other, larger applications—via the Internet.

.NET is infused into the products that make up the Microsoft platform, providing the ability to quickly and reliably build, host, deploy, and utilize secure and connected solutions using XML Web services. The Microsoft platform provides a suite of developer tools, client applications, XML Web services, and servers necessary to participate in this connected world.

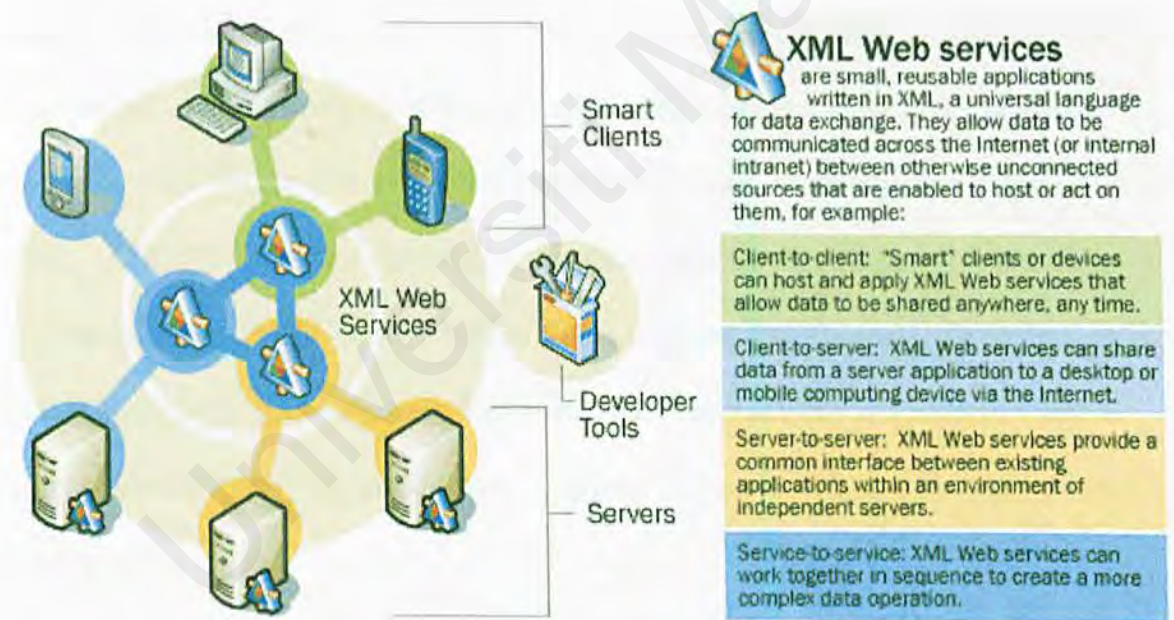


Figure 2.6 : The Components of Microsoft .NET-Connected Software

Visual Studio .NET enables developers to build the next generation of Internet applications today. Providing the most modern and feature-rich development environment, Visual Studio .NET provides developers with the tools for integrating solutions across operating systems and languages. With Visual Studio .NET, developers can easily convert existing business logic into reusable XML Web Services, encapsulating processes and making them available to applications on any platform. Developers can easily incorporate any number of Web Services that are cataloged and available in many independent UDDI directories, providing a strong foundation of services and business logic for their applications.

Using XML, an industry standard technology for describing data, Visual Studio .NET developers can build high-performance data-driven applications. Developers can use built-in ADO.NET tools that target a variety of databases, including SQL Server, Oracle, or any other XML source. With intrinsic support for XML, ADO.NET enables developers to share data across disparate computing platforms. Additionally, Visual Studio .NET includes the Microsoft Data Engine (MSDE), a 100% SQL Server-compatible database that provides programmers with a viable development database and natively supports XML for maximum interoperability.

Visual Studio .NET allows programmers to create and deploy critical server-based business logic. Historically, server-based programming has been tedious to code, prone to error, and difficult to test. With Visual Studio .NET, developers can visually compose middle-tier components using the Visual Component Designer (VCD). The VCD enables developers to drag and drop non-visual objects such as

message queues, timers, and event logs, to a design surface, automatically discovering all necessary server-based resources and configuring required components. Finally, production Web applications requiring critical changes can be altered using the new Dynamic Properties feature, letting developers simply update XML files without requiring a full recompilation of the application.

Visual Studio .NET makes it simple to create solutions that span any device. With its powerful WYSIWYG designer for Web pages, HTML IntelliSense, and Style Sheet Editor, Visual Studio .NET helps developers feel comfortable authoring complex Web solutions while leveraging Visual XML designers and XML IntelliSense for drag and drop creation and manipulation of data. With automatically generated client-side validation code, Web developers can rest assured that their application works in Internet Explorer and Netscape, reducing the amount of JavaScript code necessary to write.

Windows developers will find the new Windows forms to be intuitive and efficient as they construct code using any .NET language, including Visual Basic .NET or Visual C# .NET. With Visual Inheritance, developers can greatly simplify the creation of Windows applications by centralizing in parent forms the common logic and user interface for their entire solution. Using control anchoring and docking, programmers can provide resizable forms automatically without code, while the in-place menu editor enables developers to visually author menus directly from within the Forms Designer.

As companies expand their solutions to satisfy an increasingly mobile workforce, Visual Studio .NET can help them build solutions for servers, Windows desktops, the Web, and for a new class of devices, including smart phones and the Pocket PC. By using the same languages and techniques for construction of mobile and device-based applications, developers can leverage their knowledge and existing code for maximum productivity. With the .NET Compact Framework, developers receive a rich subset of the desktop .NET Framework and can integrate XML Web services, access enterprise data, and more in their mobile solutions. And for those developers that don't yet have a device, the built-in emulator provides a cost-effective and robust design and development experience.

For the broadest possible reach to Internet-enabled devices, Visual Studio .NET provides Mobile Internet features that enable developers to build a single mobile Web interface to support a broad range of devices, including WML 1.1 for WAP cell phones, cHTML for I-mode, HTML for Pocket PC, PalmTM, and pagers. Server-side mobile controls intelligently generate the appropriate rendering and pagination for the target web device, providing a rich and consistent user experience while preserving developer flexibility.

Visual Studio .NET continues to set the benchmark for developer productivity. With the single unified Integrated Development Environment (IDE) for all languages, including Visual Basic .NET, Visual C++ .NET, and Visual C# .NET, development organizations can take advantage of a common toolbox, debugger, and task window, greatly reducing developer learning curve and ensuring that they can always choose the language most appropriate for their task and expertise. With

IntelliSense statement completion and automatic syntax error checking, Visual Studio .NET informs developers when code is incorrect and provides immediate insight into class hierarchies and APIs. Using the Solution Explorer, developers can easily reuse code across projects and even build multi-language solutions that most effectively meet their business needs. And, thanks to the fully extensible IDE, developers can enjoy the benefits of a vibrant third party add-in and component vendor community that helps them further customize and extend the environment to suit their needs.

With application wizards, project templates, and example source code developers can rapidly create Windows, Web, and device applications with minimal up-front investment. Dynamic Help and the Microsoft Developer Network (MSDN) provide assistance based on the current task and programming language, ensuring that developers are never at a loss for information on the .NET platform or their language of choice. Visual Studio Macros, like VBA macros in Office, enable automation of routine tasks within the IDE, further enhancing the overall productivity of Visual Studio developers.

Finally, developers can choose from a set of modernized languages that gives them the most appropriate means to solve their business problems. Visual Basic .NET includes the familiar syntax Visual Basic developers are accustomed to plus optional Object Oriented Programming features including inheritance and other optional power features including structured exception handling, and free-threading.

2.9 Summary of Chapter 2

Chapter 2 discusses the literature review, which is review of previous work or systems relating to the proposed system. A number of existing online systems has been reviewed in order to have a better understanding of the strengths and weaknesses of each other, thus making improvement to the proposed system.

Universiti Malaya

Chapter 3 : Methodology



- 3.1 Methodology
- 3.2 Summary of Chapter 3

Chapter 3 : Methodology

3.1 Methodology

Over the years, many systems development methodologies have evolved. A system development methodology does not just provide a set of modeling techniques, it also defines the stages of a system development project, specifies the task to be carried out in and the output expected from each stage, provides guidelines for project management and control. [Robinson, Barbara & Prior, 1995]

Good methodology characteristics are:

- ✦ Easy to use for average analyst and programmers
- ✦ Cover all phases of system development
- ✦ Relevant to the type of application being developed
- ✦ Well quality documentation is available
- ✦ Good vendor support in terms of training and consultancy

Besides, a good methodology caused the effective way of doing things is define before the project and becomes the framework to develop staff. Among the numerous benefits offered by a good methodology are: [Holloway, 1989]

- 1) Provides a standard framework that the developer does not have to reinvent the wheel project.

- 2) Each method or tool in the methodology results in the successful completion of each development tool.
- 3) Review procedures are available to identify any errors, inconsistencies and discrepancies during development.
- 4) Increase the system quality by forcing the developer to produce flexible systems and adequate documentation.
- 5) Provides better understanding of user needs and validation of user needs.
- 6) Provides the management with tool to review project progress and checklist to access tasks and deliverables.
- 7) Improve communication among management analysts, programming, users and other stakeholders by providing a communication base.
- 8) Facilitates planning and controlling the project.

3.1.1 Spiral Model

Originally proposed by Boehm, the spiral model is an evolutionary software process model that encompasses the iterative nature of prototyping and systematic aspects of the System Development Life Cycle (SDLC), and the incorporation of software quality as a specific object. [Pfleeger, 2001] As seen in figure 3.1, the spiral model could combine development activities with risk management to minimize and control risk. It is a risk – oriented model that breaks the system to a series of small mine projects. The prototyping will be used as a risk reduction mechanism and allow the developer to apply the prototype at any stage in a cycle.

The risk driven is a major strength of the spiral model, but this can also be a weakness. With each iteration, the risk analysis weighs different alternatives in light of the requirements and constraints and prototyping verifies feasibility or desirability before a particular alternative is chosen. The success of the spiral model largely depends on the considerable risk assessment expertise.

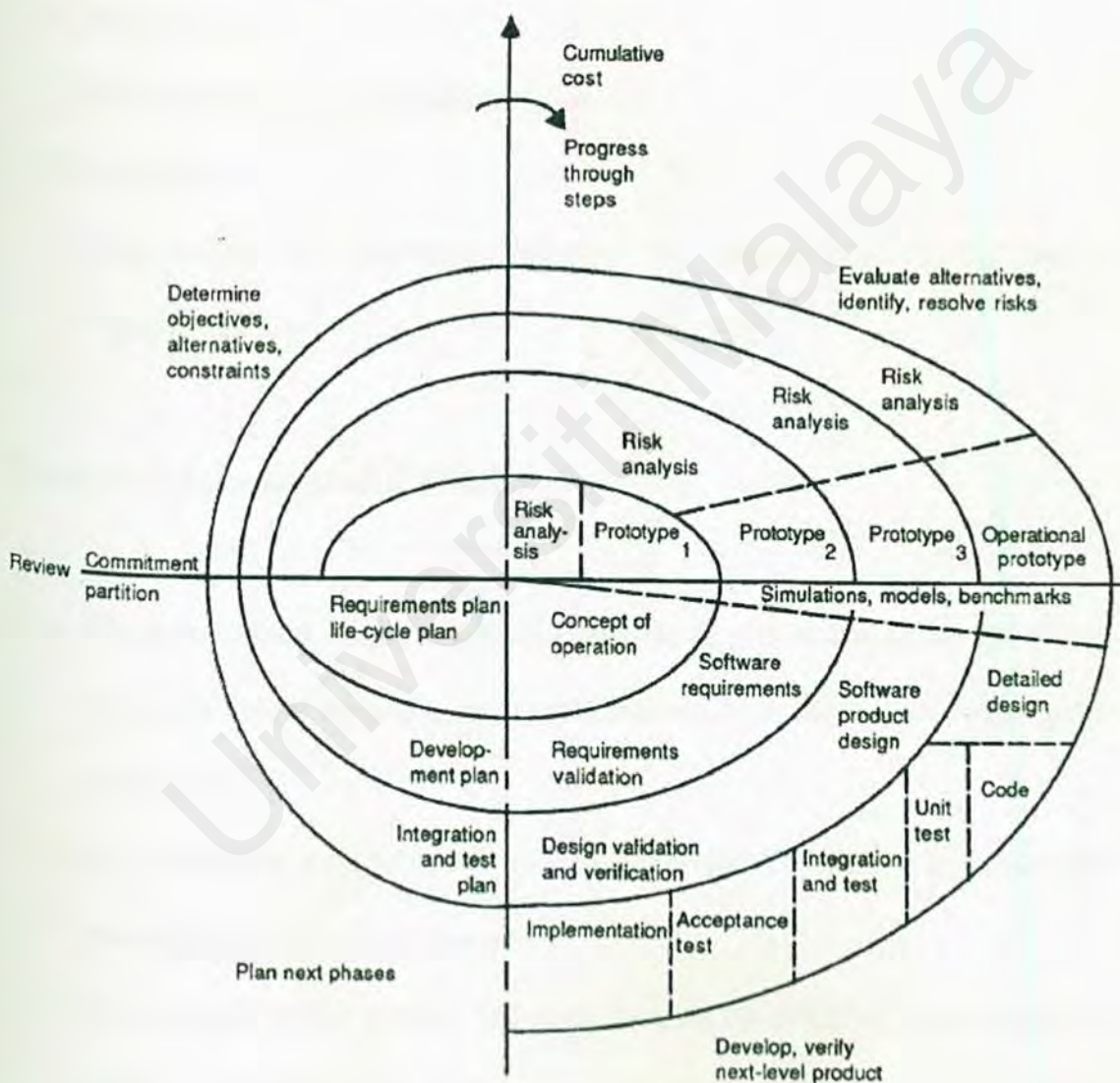


Figure 3.1 : The Spiral Model

The spiral consists of four major activities within its cycle:

a) Planning

Determine the project objective, alternatives and constraints

b) Risk Analysis

Determine the risk of the project and other alternatives to the current approach

c) Engineering

Development and testing of the system

d) Evaluation

The process of determining whether the system has met the users' requirement

Reasons for not choosing Spiral Model

- a) The spiral model is only suitable for large-scale project and has no governors to clearly define the scope; most often the length or numbers of cycles grow unbounded.
- b) It is a risk driven methodology, but scale of SMRCS is not that large and will not emphasize on the risk analysis.
- c) It is suitable for the project that is not firm on the deadline, cycles continue with no clear determination condition. But the SMRCS, the submission date has been clearly set and must be completed on time.

3.1.2 V Model

The chosen process model for SMRCS is V model. The V model is a variation of the waterfall model that demonstrates how the testing activities are related to analysis and design. As shown in figure 3.2, coding forms the point of the V, with analysis and design on the left, testing and maintenance on the right. Unit and integration testing addressed the correctness of programs. The V model suggests that unit and integration testing also be used to verify the program design. During unit and integration testing, the coders and test team members should ensure that all aspects of the program design have been implemented correctly in the code.

Similarly, system testing should verify the system design, making sure that all system design aspects are correctly implemented. Acceptance testing, which is conducted by the customer rather than the developer, validates the requirements by associating a testing step with each element of the specifications; this type of testing checks to see that all requirements have been fully implemented by the system is accepted and paid for.

The model's linkage of the left side with the right side of the V implies that if problems are found during verification and validation, then the left side of the V can be re-executed to fix and improve the requirements, design and code before the testing steps on the right side are reenacted. This aligns with the attribute of the project as it could literally incur requirements change at any point of development.

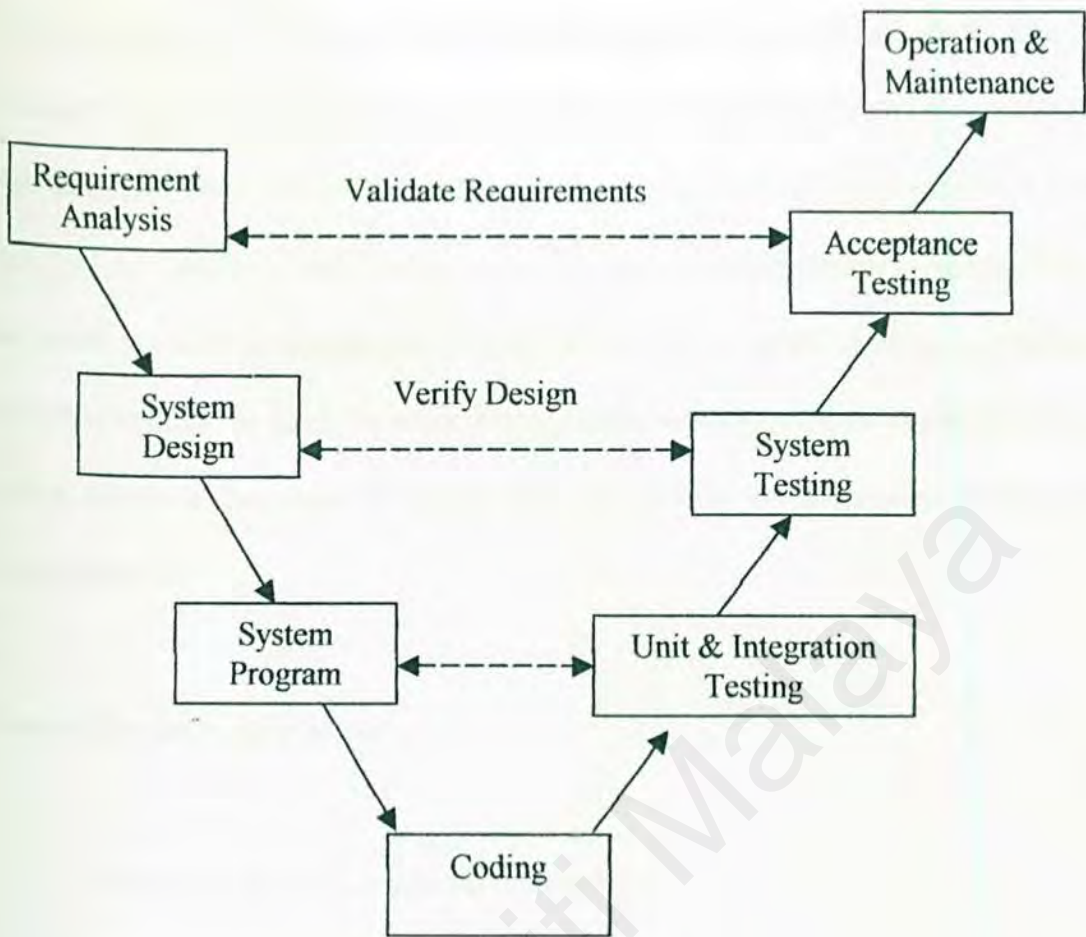


Figure 3.2 : The V Model

Defined requirements are assumed to be finalized but the project is subjected to changes in requirements. Some adjustment could emerge in the process of sports and management that prompts the demand of adding new requirements or possibly redesign parts of the previously defined system. The V Model's versatility and flexibility to embrace and accommodate these changes should they occur justify the selection of the model. Model V also allows the minimization of errors in development and guarantees that feasible requirements are fully implemented.

Though model V is derived from the waterfall model, but its advantages are more prevailing in the context of the SMRCS project. The waterfall model depicts how each major phase of development terminates in the production of some artifact such as requirement, design and code. In one perspective, there is no insight in how each activity transforms one artifact to another, such as requirements to design. Thus, the model provides no guidance to developers on how to handle changes to products and activities that are likely to occur. For instance, when requirements change during coding activities, the waterfall model does not address the subsequent changes to design and code.

Reasons for choosing V model:

- a) Testing brought earlier into the life cycle.
- b) Detailed test plan at each stage.
- c) Versatility and flexibility.
- d) Minimization of errors in development.
- e) Guarantee that flexible requirements are fully implemented.

3.2 Summary of Chapter 3

Chapter 3 discusses the methodology used to develop the system. A few model were discussed in brief, such as the spiral model and the V model. Their advantages and disadvantages are compared in order to choose a suitable methodology for this project.

Universiti Malaya

Chapter 4 : System Requirement

Analysis



- 4.1- Techniques Used to Define Requirements
- 4.2 Requirement Analysis
- 4.3 Hardware Requirement
- 4.4 Chosen Platform, Database and Tools
- 4.5 Summary of Chapter 4

Chapter 4 : System Requirement Analysis

4.1 Techniques Used to Define Requirements

In determining the requirements for the SMS Applications Systems, a few techniques had been use. The techniques include library research, Internet research, brainstorming and discussions.

4.1.1 Brainstorming

Before the project can even begin, an initial brainstorming session is hold together with my project supervisor Mr. Ang. The purpose of this session is to first understand and grab the overall concept behind the project at hand. This we define the project and draft out a few functions that the SMS Applications System should have. Brainstorming is also a good way of planning for the next move.

4.1.2 Discussions

Discussions with a few friends have been conducted to define the system requirements. The discussions had help to understand more on how the SMRCS is going to work and flow. Besides, the discussions also guide me throughout the design parts where I can draft out the systems interfaces more clearly.

4.1.3 Internet research

Nowadays, the Internet has become the main place where most people find resources and information. It has become a major resource to obtain the latest information. The Internet provides faster and a more efficient way to get information. The information obtained is then used as guidelines to help in defining the requirements of the SMS Applications System. Thus the Internet is used as one of the techniques to do research on the SMS Applications System.

4.1.4 Library research

Besides the Internet, library also helps in the finding of important information. The library also provides a lot of useful information, where it guides me in finding and searching more relevant information for the development of the SMS Applications System.

4.2 Requirement Analysis

Requirement analysis enables the system engineer to specify software elements and establishes design constraints that software must meet. A complete understanding of software requirements is essential to the success of a software development effort, no matter how well designed or well coded. A poorly analyzed and specified program will disappoint the user and bring grief to the developer.

Requirement elicitation is the critical part of the process. Variety of techniques is used to determine users and customer's need. Requirement identifies the "what" of the system and the design identify the "how" of the system.

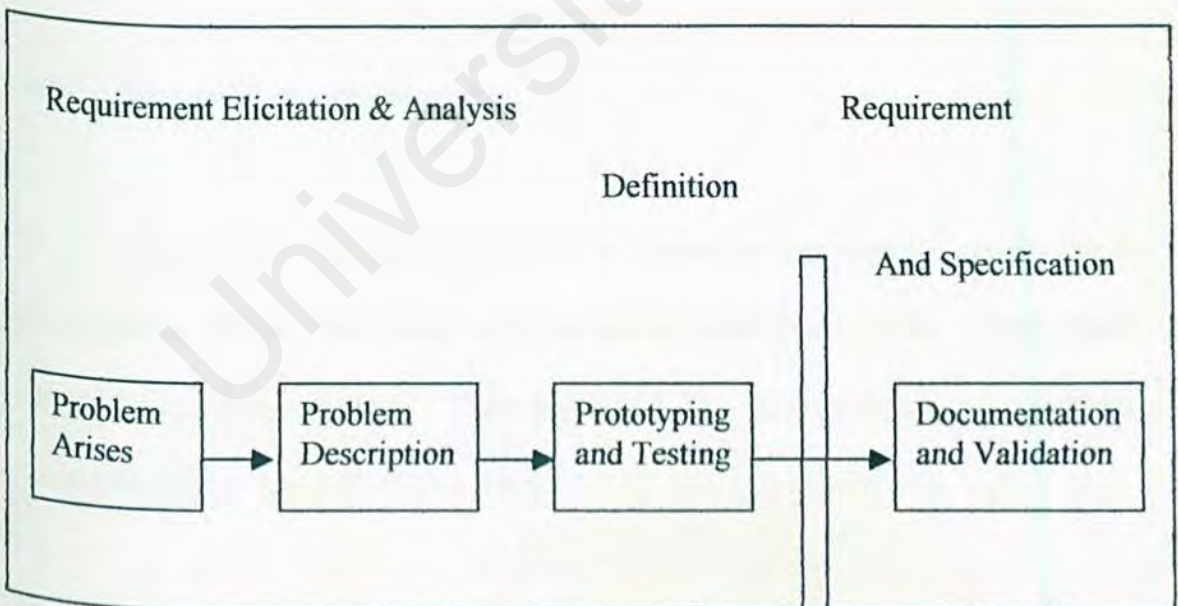


Figure 4.1 The Process of determining requirements

Requirement elicitation enables to explain the requirements definition of the system. Requirement definition is a complete listing of everything the customer expects the proposed system to do. It represents an understanding between customers' need or wants and is usually written jointly with developer. On the other hand, the requirements specifications restate the requirements definition in technical term appropriate for the development of a system design. It is the technical counterpart to the requirements definition document and requirement analysts write it.

Requirement analysis covers two main categories, which are:

- ⊕ Functional Requirements
- ⊕ Non – Functional requirements

4.2.1 Functional Requirements

A functional requirement describes an interaction between the system and its environment. Further, functional requirement also describes how the system should behave given certain stimuli. [Pfleeger, 2001] The following are the functional requirements for each of the sub-system.

4.2.1.1 User Management Module

This module helps the administrator to manage the users who are using SMRCS. This module allows administrator to approve new user applications, view and edit user information.

4.2.1.2 Record View Module

This module helps the administrator to manage the database. Administrator can view or edit the data in the database.

4.2.1.3 Result Module

This module enables the administrator to add in new results and update the result table. It will check if the user IC number and Matriculation number is match before the system send the result back to the user.

4.2.1.4 Top Up Module

This module checks the validity of the pin number before it updates the account. Administrator adds valid pin number into the database.

4.2.1.5 Authentication Module

This module is for all the users to log onto the system with their own unique user name and password to use the system. If a wrong password is entered, the system will reject the access to the system.

4.2.1.6 Communication Module

The communication module must be able to provide the users with the interface that can input their message. A send button will then submit the form to a database. An alert icon will prompt the desired users that he/she has unread message.

4.2.1.7 Phone Book module

This module allows user to keep information such as phone number, users name and id and others.

4.2.1.8 Registration Module

This module allows user to register as a new member of SMRCS. Users need to fill up some information and request a username and password that will use in authentication and authorization process.

4.2.1.9 User Top Up Module

User top up module enable the users to key in their pin number in order to top up the credits in their account. Users must make sure they have enough credits before they send a message.

4.2.1.10 Model Selection Module

This module allows user to choose from a variety of phone model that is available. User can also change the skin color of the selected phone.

4.2.2 Non – Functional Requirements Analysis

A non-functional requirement describes a restriction on the system that limits user choices for constructing a solution to the problem [Pfleeger, 2001]. It will include the system properties such as response time, the memory requirement and so on. The following are the non-functional requirements for the SMRCS.

4.2.2.1 Security

The system will have security access to login into the system. Access by users should be authenticated and validated. Each user will have a different identity to log on to the system. Different level of user will have different authority level of accessibility.

4.2.2.2 User Friendliness

The system should provide user-friendly interfaces. The system should have to use menu and graphical interface. The use of mouse, icon and menu should be common and familiar functions as a Window environment.

4.2.2.3 Reliability

The system must be reliable and produce a consistence output for every process. It must always provide quality performances and must not fail at any critical time. For example, the system should provide a consistence result each time a same input is entered into the system.

4.2.2.4 Availability

The system should be developed in the most common operating system environment to maximize adaptability of the application to the computer system. The system should be easily available when requested by new and existed users.

4.2.2.5 Accuracy

The system should provide the correct info when a certain input data are entered. This means the output process by the system must be accurate.

4.2.2.6 Performance

The webpage can load with a minimum delay time and any changes are immediately updated.

4.2.2.7 Flexibility

The system changes should be easily implemented. Changes may often take the form of upgrades or enhancements. The system should have the capability to take advantage of new technologies and resources

4.2.2.8 Efficiency

In an efficient system, there should be compatibility and integration between the different subsystems involved. For example, the speed at which data is input should closely match the speed at which data is processed to avoid backlogs of work on one hand or idle processing capacity on the other. The aim should be to achieve smooth interfaces between processes.

4.3 Hardware Requirement

The table 4.1 shows the summary of the hardware and software requirements that have been considered for this project.

	Server Side	Client Side
Hardware Requirements	<ul style="list-style-type: none">▪ Processor with 450 MHz at minimum level▪ Hard disk space with 5GB▪ 128MB RAM of memory	<ul style="list-style-type: none">▪ Processor with 400 MHz▪ Hard disk space with 2GB▪ 64MB RAM of memory▪ Network connections through existed network configuration on modem
Software Requirements	<ul style="list-style-type: none">▪ Microsoft Windows 2000 Professional as the operating system▪ Microsoft SQL Server 7.0 as the database▪ VB.Net	<ul style="list-style-type: none">▪ Microsoft Windows 2000/NT

Table 4.1 : Hardware and Software Requirements

4.4 Chosen Platform, Database and Tools

4.4.1 Chosen Development Platform

Microsoft Windows 2000 Professional

The Windows 2000 is the evolutionary and built on the NT technology. It is design especially for small business and professional users as well as to the more technical and larger business market for which the NT was designed. It was reported in earlier reviews that Windows 2000 is more stable than Windows 98/NT systems. It is likely to crash. The main reason for choosing Windows is that Windows currently enjoys a dominant position as the preferred operating system by most corporations. Microsoft's Windows enjoy a penetration rate at almost 90% of the overall market, which makes it almost the de factor choice for operating system.

Windows comprises of a user-friendly Graphics user Interface (GUI) that makes it easy for both consumers and computer professionals to use. Although Linux also provides a user-friendly GUI, it undoubtedly has a much lower usage percentage in corporations. This prompted me to choose the more popular and latest Windows2000 over Linux. Unix on the other hand does not provide its user with user-friendly GUI.

4.4.2 Chosen Database Management System

Microsoft SQL Server 2000

SMRCS will be developed using Microsoft SQL 2000. It has been chosen because it can handle a large amount of data. SQL 2000 is tightly integrated with the Windows 32-bit platform. Particularly, it is design to take advantage of the features of the Windows 2000 operating system for large-scale organization and enterprise database.

Compared with Microsoft Access, it has relatively higher data storage capacity. The main reason for not using Oracle database is because of to set up an Oracle database server need a large amount of budget that is more expensive than using Microsoft SQL Server. Although MySQL is free, it does not have a user-friendly interface to set up a database. It will take a lot of time to develop a database for the SMRCS. So MySQL will not be used in the SMRCS.

4.4.3 Chosen Application Development Tool

Microsoft Visual Studio Professional.Net

Microsoft Visual Studio Professional.Net has been chosen as the application development tool. Visual Studio .NET is the tool that provides developers the key to the next generation Internet with XML Web Services, high performance database applications with XML and RAD for the Server. It enables developers to build

solutions for the broadest range of clients from Web applications to Windows to thin-client devices to smart devices. The RAD capabilities provide a shared IDE and a choice of programming languages from Visual Basic, C++, C# and Java. Visual Studio .NET Professional is RAD for the Programmable Web.

Quickly build next-generation Internet applications. Create components that can be accessed on any platform using XML Web services. Convert existing functions to XML Web services and reuse existing Web services. Build high-performance database applications using with Microsoft Data Engine (MSDE)—a 100 percent SQL Server™-compatible engine - using ADO .NET and XML.

Create solutions that span any platform or device. Simplify development of browser-based applications with shared Web Page, XML Schema, and Style Sheet visual designers. Build powerful Microsoft Windows®-based applications and target mobile devices.

Cut time-to-market for powerful, scalable applications. Leverage an integrated development environment to share one toolbox, debugger, form designer, and task window across languages. Jump-start development by assembling applications with reusable .NET user interface, database, and server components. Choose among easy-to-use, modern programming languages.

4.4.4 Chosen Development Language

Microsoft Visual Basic.Net

Microsoft Visual Basic.Net has been selected as the web development tool for the proposed system. The reasons of choosing Visual Basic.Net are as follows:

- ✦ Quickly and easily create cutting edge XML Web services and applications that scale and integrate easily
- ✦ Improve the reliability, scalability, and security of your application using .NET Framework's application execution environment
- ✦ Tap the broadest developer talent through the .NET Framework and Visual Studio .NET support for most modern programming languages
- ✦ Enable use of existing skills to create solutions for a wide range of devices

4.5 Summary of Chapter 4

Chapter 4 discusses the system analysis used to develop the system. It explains several methods that are used for information gathering in order to obtain the appropriate information regarding the system. System functionalities, which are divided into functional requirements and non-functional requirements, are another feature that has been analyzed. Apart from that, comparison of programming language, tools and technology is essential so as to have a system that has a good compatibility of hardware and software.

Chapter 5 : System Design



- 5.1 Introduction
- 5.2 Overview of System Architecture
- 5.3 Process Design
- 5.4 Database Design
- 5.5 User Interface Design
- 5.6 Summary of Chapter 5

Chapter 5 : System Design

5.1 Introduction

System design is a process through which requirement are translated into a representation of the system through out their stage, a thorough description will show the logical flows of the system as well as how the requirements is being implemented and fulfill one by one. Design is a multi-step process in which representations of the data structure, program structure, interface characteristics and procedural details are shown.

The objective of system design are listed below:

- ✦ Specify logical design elements
 - Detailed design specification that describes the features of an information system: input, output, files and database and procedures.
- ✦ Meet user requirements

Meet user needs states in term of:

 - Performing appropriate procedures correctly.
 - Presenting proper form of information.
 - Providing accurate results.
 - Using appropriate method of interaction.
 - Providing overall reliability.
- ✦ Ease of use
- ✦ Favorable human engineering

- ✦ Ergonomic design that is physically comfortable to user effectiveness and efficiency
- ✦ Provide software specifications
Specific component and functions with adequate detail to construct application software

System design includes the following issues:

- ✦ System Architecture Design
- ✦ System Functionality Design
- ✦ User Interface Design
- ✦ Database Design

5.2 Overview of System Architecture

SMRCS client/server architecture is divided into three distinct tiers – client, server and database. Components are built into each tier to fulfill its role and then tie together to form a final solution.

Initially, if the client (the sender) wants to send SMS to the receiver, it sends through the server using socket programming. Then the server will process it and send the messages to the receiver. Or if the client wants to check their result, they need to type in their matrix and I.C. number first according to the format that had been stated. Then the message will be send to the server and the server will process

and check through the database. After a few second, the server will reply back to the sender.

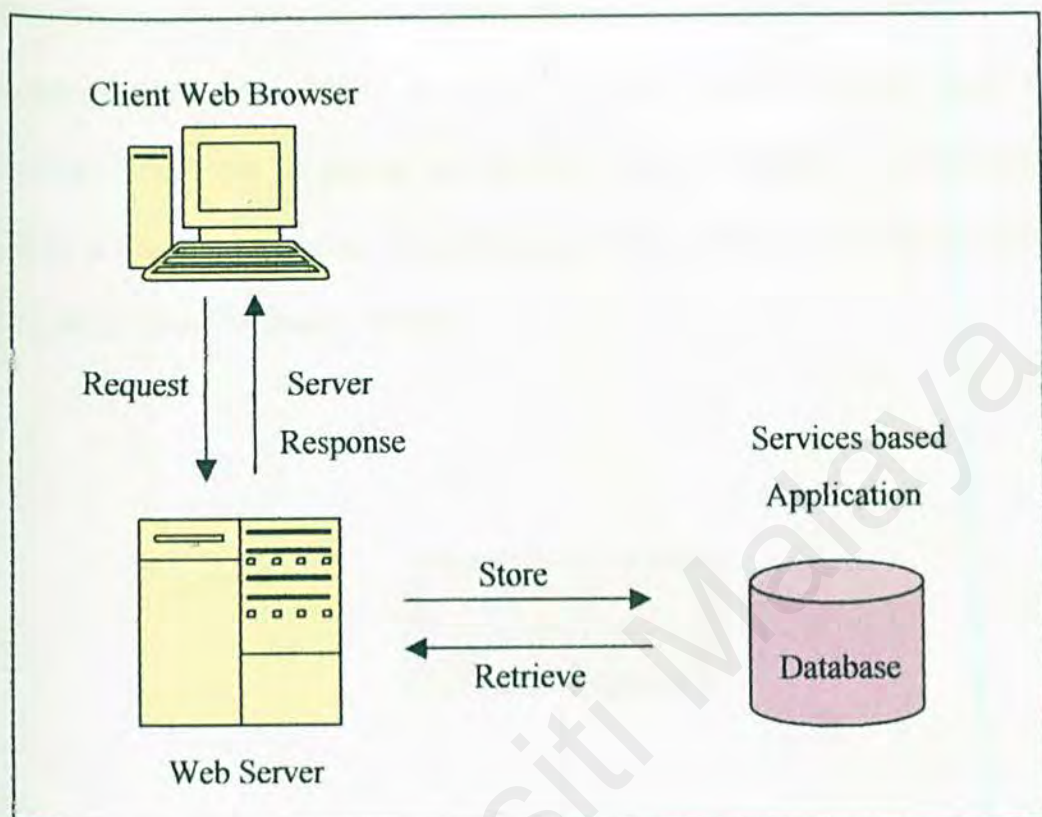


Figure 5.1 : Three Tier Architecture of SMRCS

5.3 Process Design

There is several design methodology for the process design. SMRCS is designed based on the data flow oriented-design methodology on the structured design. Data flow oriented-design has its origin in earlier concepts that stressed on modularity, top down design and structured programming. This design focuses on system design and data flow design.

5.3.1 System Structure Charts

The system structure chart is used to depict high-level abstraction of a specified system. The use of structure chart is to describe the interaction between independent modules. Major functions from the initial component part of the structured chart can be broken into detailed sub components. A structure chart is simply a diagram consisting the modules and connecting arrows. Figure 5.1 shows the system structure chart of SMRCS.

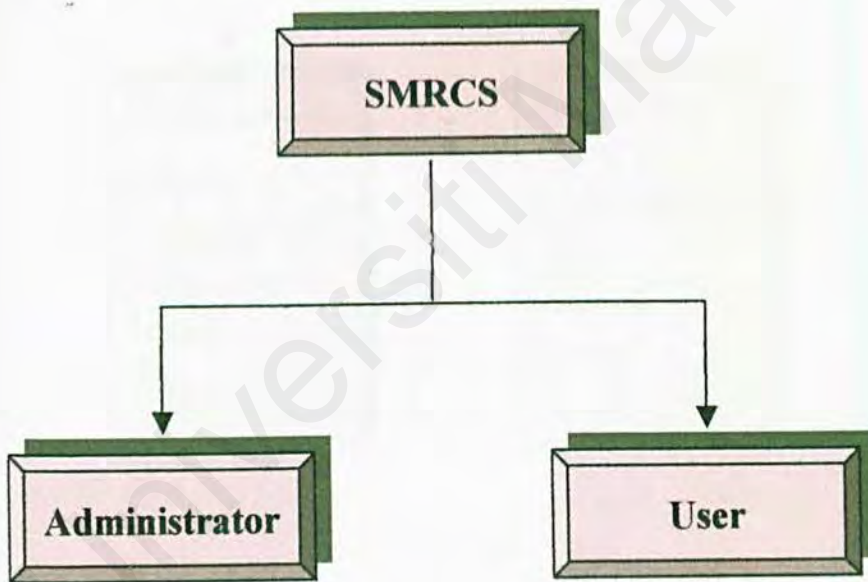


Figure 5.2 : Structure Chart of SMRCS

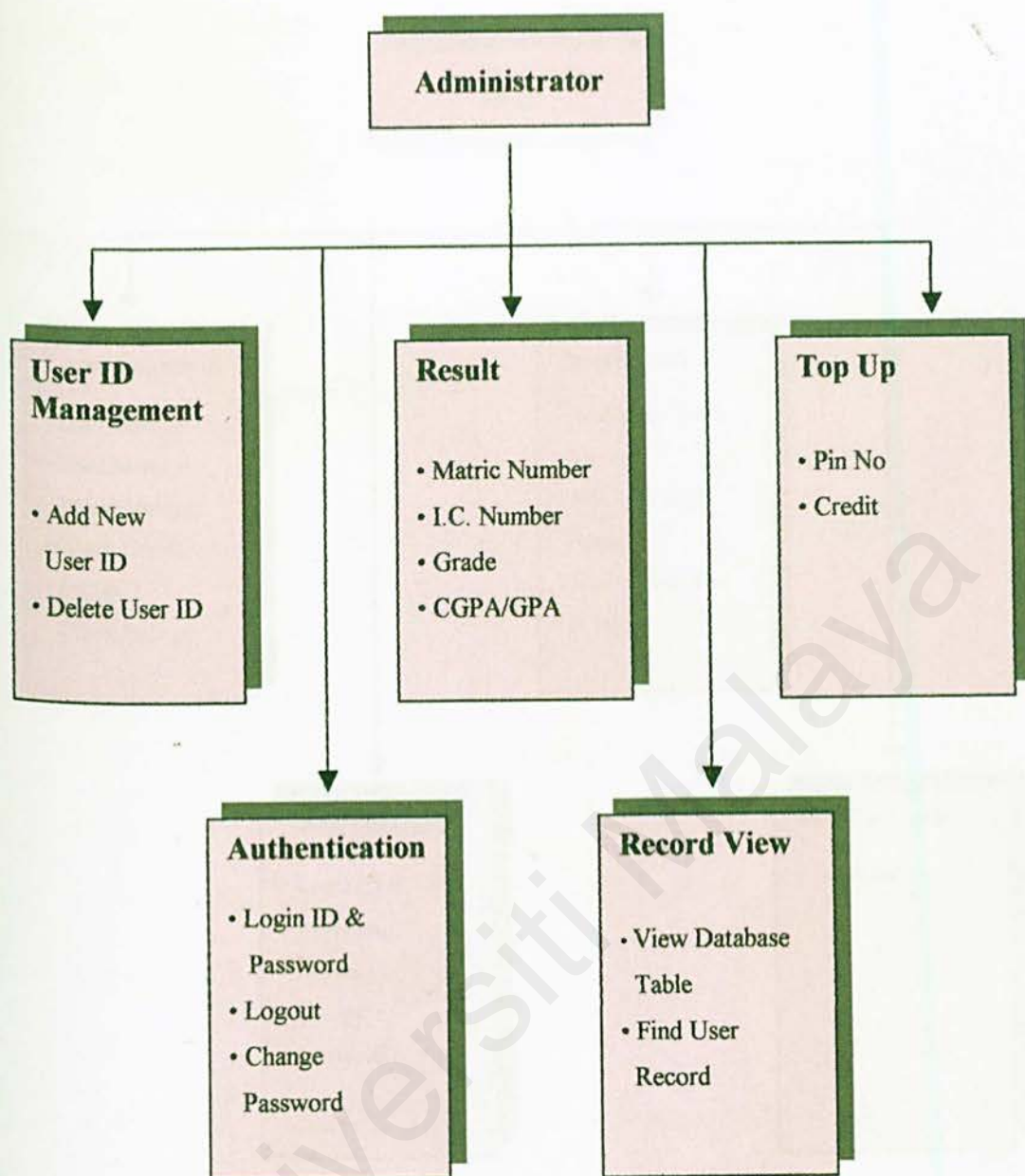


Figure 5.3 : Administrators' Structure Chart

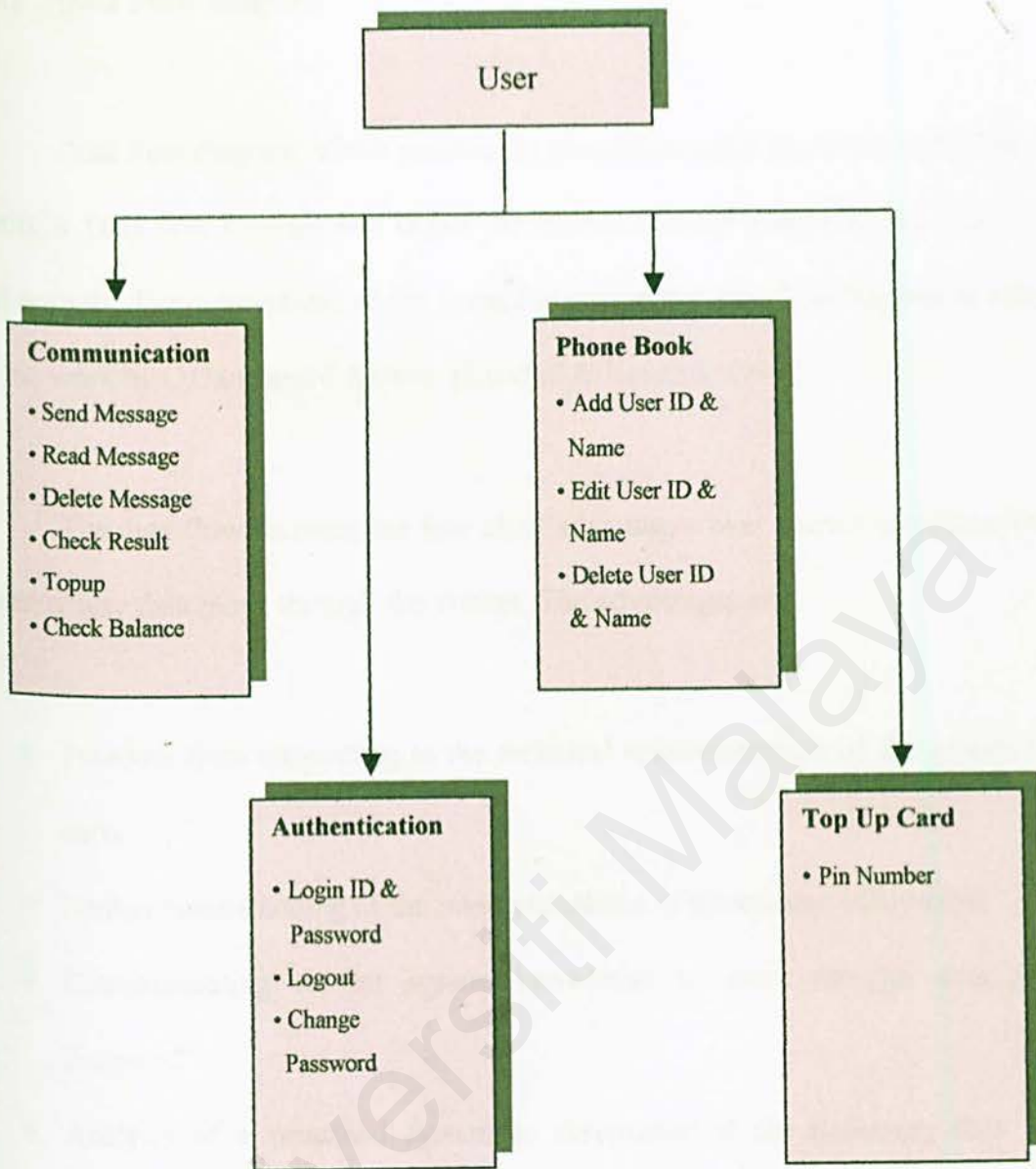


Figure 5.4 : Users' Structure Chart

5.3.2 Data Flow Diagram

Data flow diagram, which graphically characterizes and processes and flows in SMRCS. Data flow diagram will depict the overview of the system inputs, processes and outputs. The convention, which is used to design the data flow diagram is based on the work by C.Gane and T.Sarson. [Kendall & Kendall, 1998]

The data flow diagram has four chief advantages over narrative explanations of many ways data move through the system. The advantages are:

- ✦ Freedom from connecting to the technical implementation of the system too early
- ✦ Further understanding of the inter relatedness of system and subsystems
- ✦ Communicating current system knowledge to users through data flow diagrams
- ✦ Analysis of a proposed system to determine if the necessary data and processes have been defined

DFD is easy to be understood as it has symbols that specify the physical aspects of implementation. There are four basic symbols in DFD: entity, flow of data, process and data stores as shown in Table 5.1.

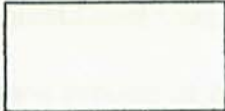

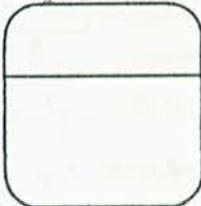
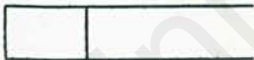
Symbols	Attribute
	<p>Entity</p> <ul style="list-style-type: none"> - source or destination of data
	<p>Flow of Data</p> <ul style="list-style-type: none"> - as flow of data - shows movement of data from one point to another
	<p>Process</p> <ul style="list-style-type: none"> - represent a process - show the occurrence of a transforming process
	<p>Data Store</p> <ul style="list-style-type: none"> - represent a manual store such as a filing cabinet or a computerized file or a database

Table 5.1 : DFD Symbols

The convention, which is used to design DFD are based on the work by C.Gane and T.Sarson. The data flow is conceptualized with a top-down perspective. So, the Context Level Diagram will be drawn, followed by the Diagram 0. Diagram 0 is an overview process of all the major modules in SMRCS that includes all the data stores, entities and process involved.

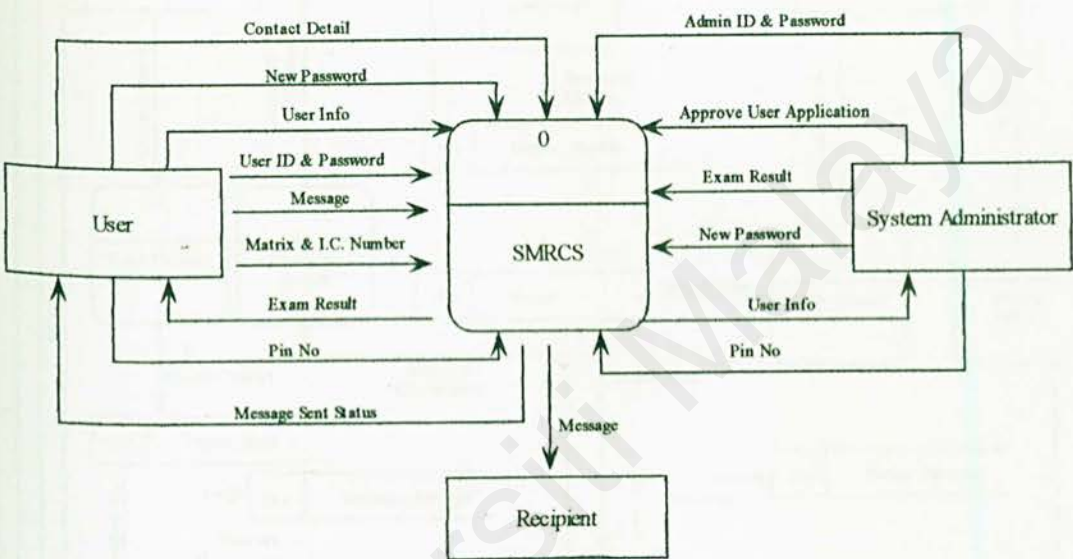


Figure 5.5 : Context Level Diagram of SMRCS

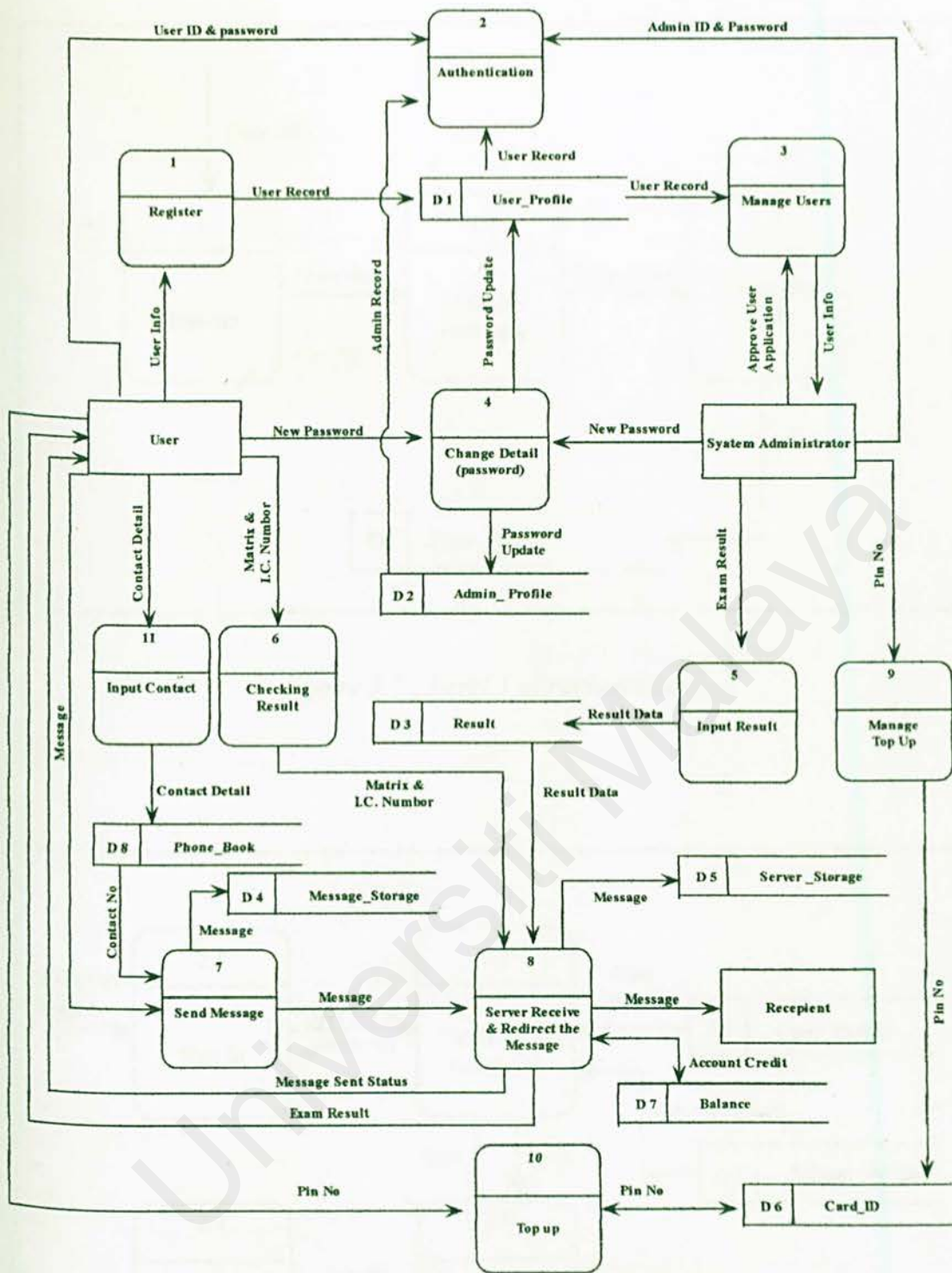


Figure 5.6 : Diagram 0 of SMRCS

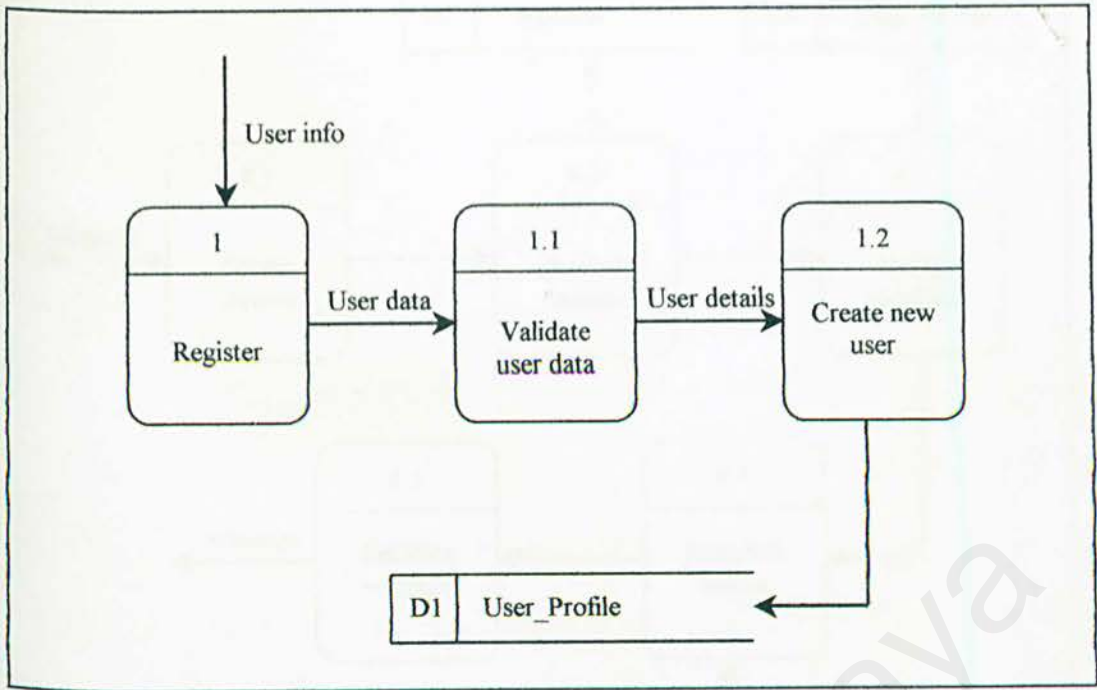


Figure 5.7 : Level 1 of registration

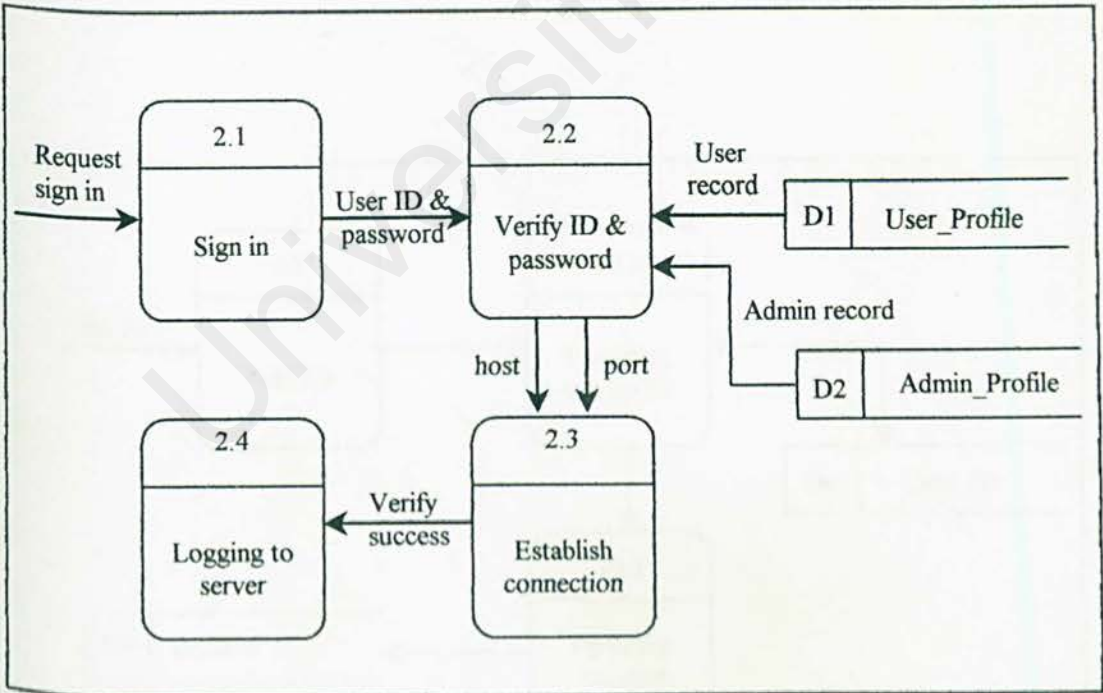


Figure 5.8 : Level 1 of Authentication

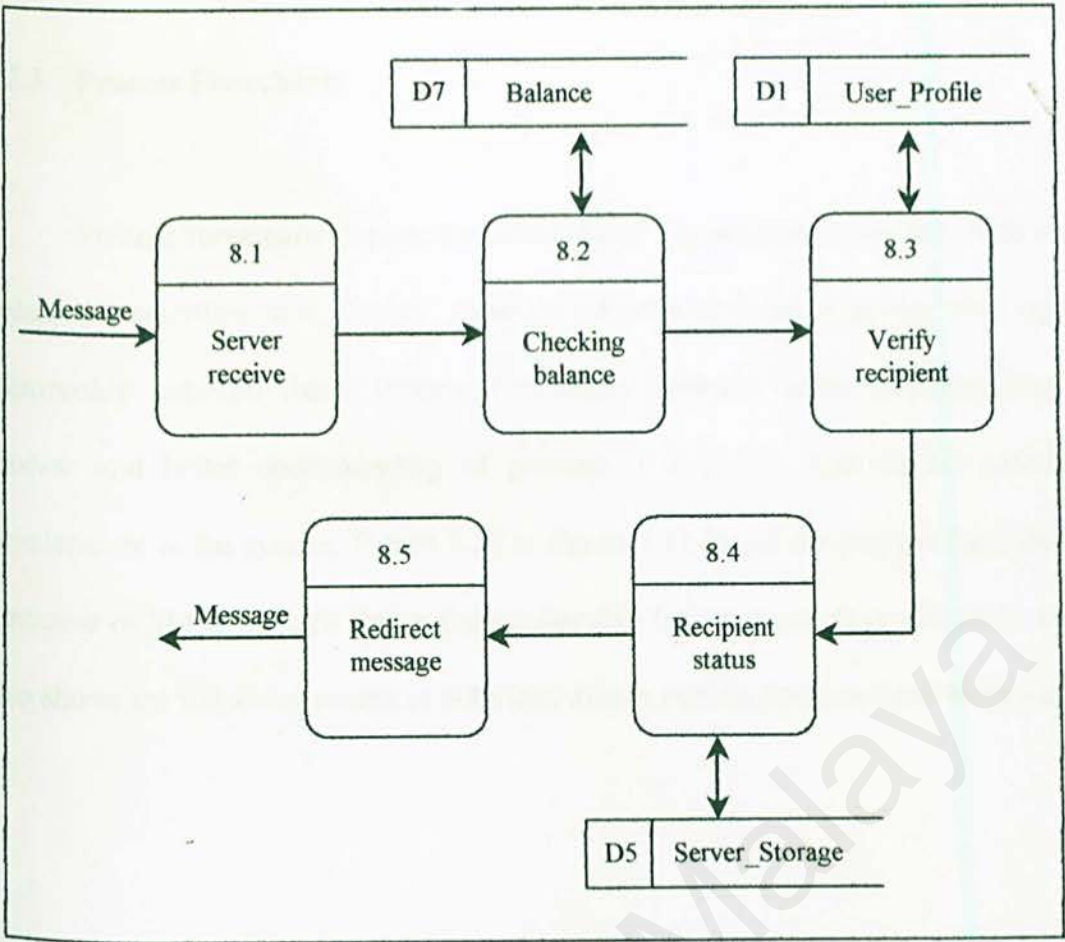


Figure 5.9 : Level 1 of sending message

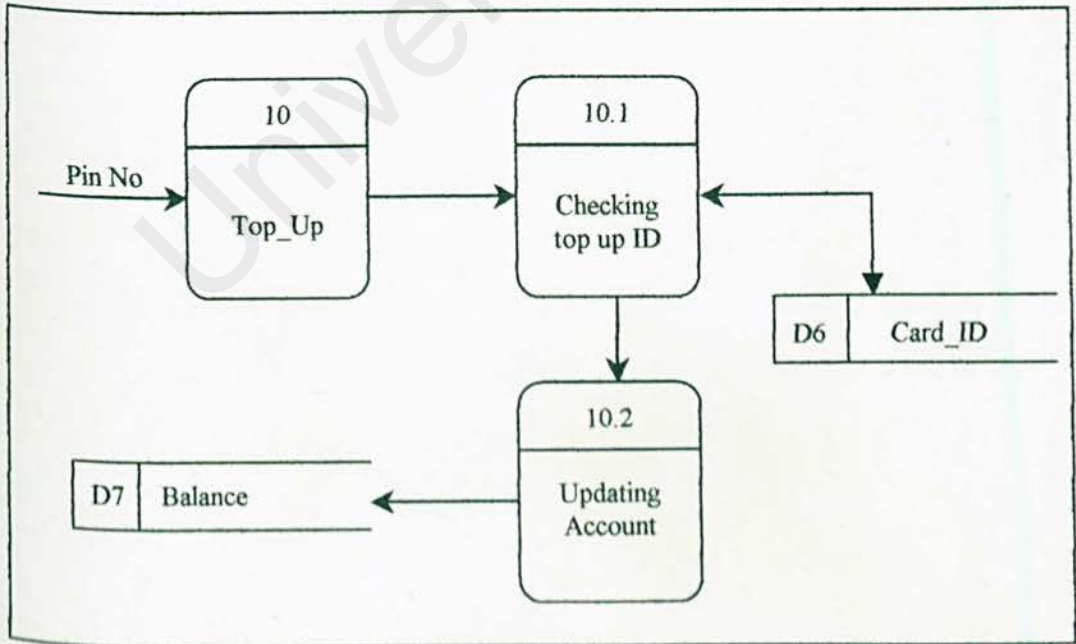


Figure 5.10 : Level 1 of top up account

5.3.3 Process Flowcharts

Process flowcharts depicts the breaking of any process down into individual events or activities and display these in shorthand form showing the logical relationship between them. Process flowcharts promote better understanding of process and better understanding of process is a pre – requisite for effective development of the system. Figure 5.10 to figure 5.11 depict the process flow for the functions of SMRCS. Each shows the relationship between events or activities. Each also shows the following events or activities after a certain decision have been made.

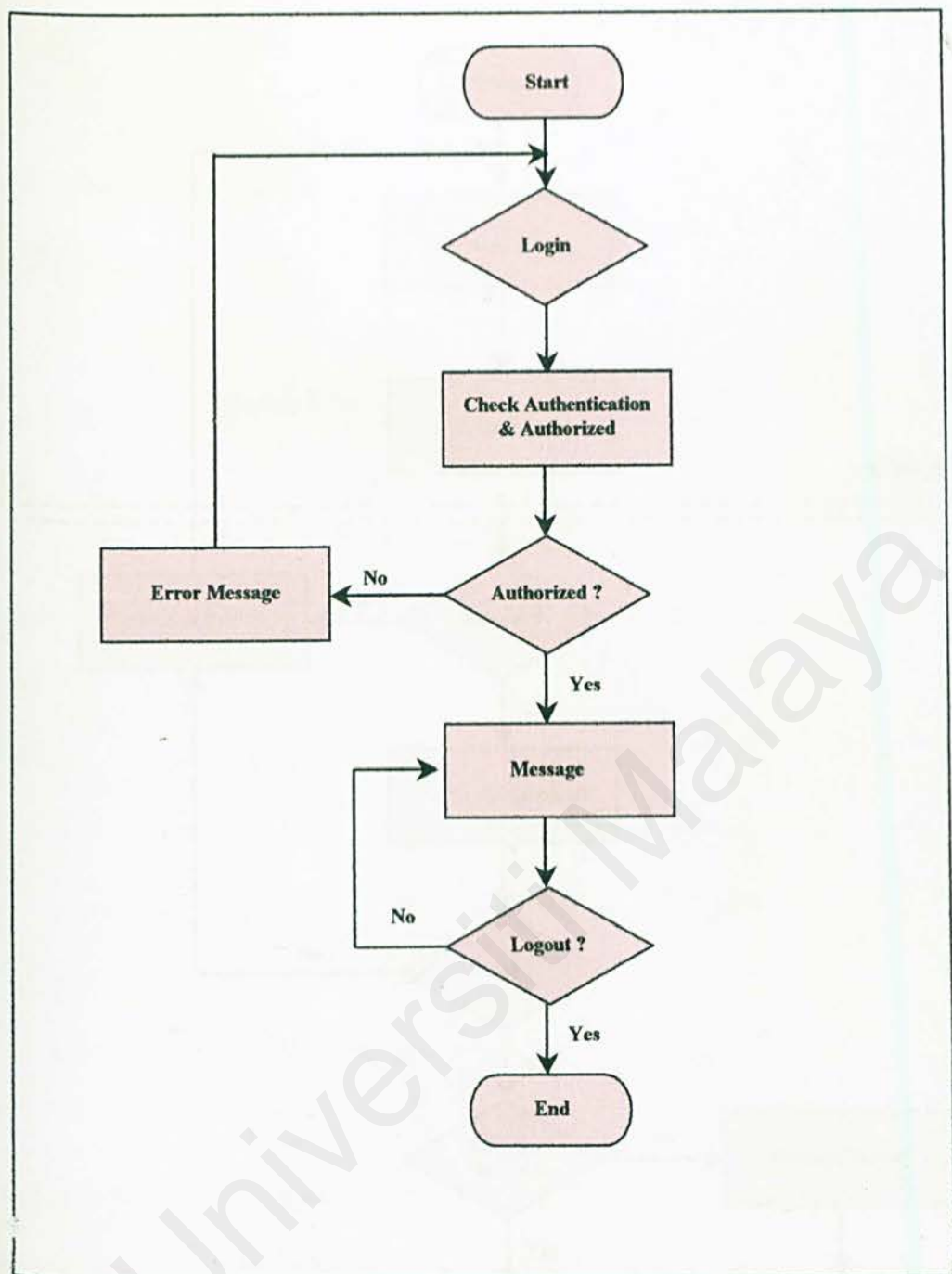


Figure 5.11 Flow Chart of SMRCS

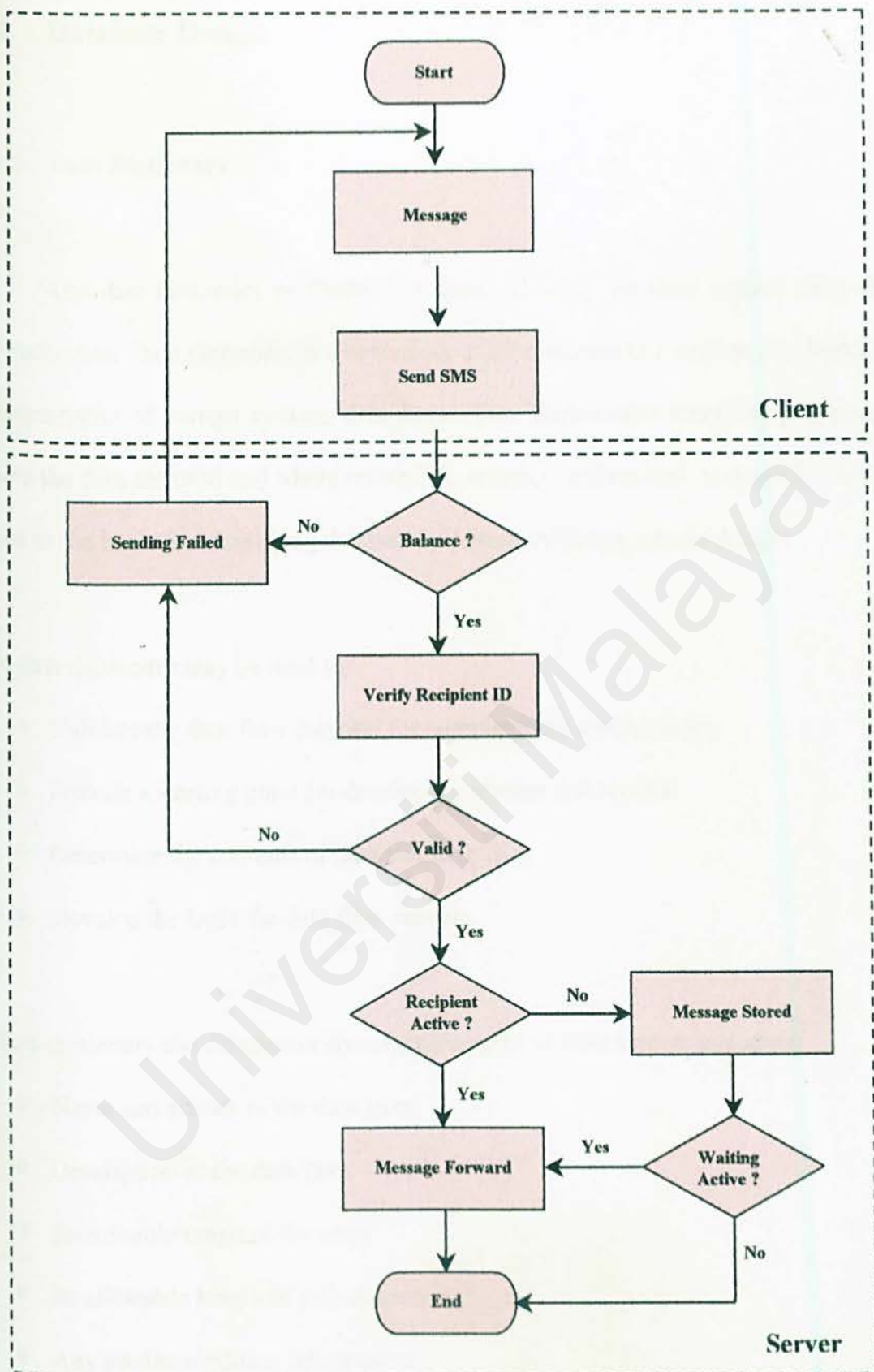


Figure 5.12 Flow Chart of Sending SMS

5.4 Database Design

5.4.1 Data Dictionary

The data dictionary of SMRCS is designed using the third normal form of normalization. Data dictionary is a repository of all elements in a system. It is logical characteristics of current systems data stores. Data dictionaries identifies processes where the data are used and where immediate access to information is needed. It also saves as the basic for identifying database requirement during system design.

The data dictionary may be used to:

- ✦ Validate the data flow diagram for completeness and accuracy
- ✦ Provide a starting point for developing screens and reports
- ✦ Determine the contents of data stored in files
- ✦ Develop the logic for data flow process

A data dictionary should contain specific categories of information including:

- ✦ Name and aliases of the data item
- ✦ Description of the data item
- ✦ Permissible range of the entry
- ✦ Its allowable length of information
- ✦ Any pertinent editing information

The data dictionaries table in for SMRCS is shown as below:

Table Name : User Profile (D1)			
Primary Key : User_ID			
Field Name	Data Type	Length	Description
User_ID	char	10	To create an unique key for this table
First_name	varchar	50	User's first name
Last_name	varchar	50	User's last name
User_password	varchar	50	User's password for login purpose
Country	varchar	50	User's location
Date of birth	datetime	8	User's birth date
Sex	varchar	10	Male or female
Email_address	varchar	50	User's e-mail for password reminding

Table 5.2 : User Profile

Table Name : Admin Profile (D2)			
Primary Key : Admin_ID			
Field Name	Data Type	Length	Description
Admin_ID	char	10	To create an unique key for this table
First_name	varchar	50	Admin's first name
Last_name	varchar	50	Admin's last name
User_password	varchar	50	Admin's password for login purpose
Country	varchar	50	Admin's location
Date of birth	datetime	8	Admin's birth date
Sex	varchar	10	Male or female
Email_address	varchar	50	Admin's e-mail for password reminding

Table 5.3 : Admin Profile

Table Name : Result (D3)			
Primary Key : Matric No			
Field Name	Data Type	Length	Description
Matric No	varchar	10	Student's matriculation number
IC No	varchar	14	Student's identity card number
Result CGPA	float	8	Student's average result
Result GPA	float	8	Student's result for that semester

Table 5.4 : Result

Table Name : Client Storage(D4)			
Primary Key : Date Saved			
Field Name	Data Type	Length	Description
Sender ID	varchar	10	Sender's user ID
Receiver ID	varchar	10	Receiver's ID
Message	varchar	160	Message data
Date Saved	Date/Time	8	The data & time of this message being received

Table 5.5 : Message Storage

Table Name : Server Storage (D5)			
Primary Key : Date Send			
Field Name	Data Type	Length	Description
User ID	char	10	Sender's user ID
Message	varchar	160	Message data
Receiver ID	varchar	10	Destination receivers'
Date Send	Date/Time	8	The data & time of this message being received

Table 5.6 : Server Storage

Table Name : Card ID (D6)			
Primary Key : Pin No			
Field Name	Data Type	Length	Description
Pin No	char	10	10 digit unique pin number
Credit	int	4	Value of credits

Table 5.7 : Card ID

Table Name : Balance (D7)			
Primary Key : User_ID			
Field Name	Data Type	Length	Description
User_ID	char	10	User's unique ID
Account_Balance	float	8	Credit left in the account

Table 5.8 : Balance

Table Name : User_Contact (D8)			
Primary Key : No			
Field Name	Data Type	Length	Description
No	int	4	Primary key
User_ID	char	10	User's unique ID
Contact_ID	char	10	Contact's user ID
Contact_Name	varchar	50	Contact's name

Table 5.9 : User Contact

Table Name : Credit List (D9)			
Primary Key : Directory			
Field Name	Data Type	Length	Description
Directory	varchar	10	Receiver's Destination
Description	varchar	50	Description
Credit	int	4	Credit Required

Table 5.10 : Credit List

5.5 User Interface Design

User Interface Design is one of the main parts of the system design. A good interface design will help user to understand on the program or system even faster. However, User Interface Design is a tricky thing to design because different people have different style of perception.

The objective of User Interface Design is to provide the best way for user to interact with the computer systems. Computer users are not interested to the technology behind the life. Therefore a good and user-friendly User Interface Design will certainly play a big role in playing the system or program a success.

The interface is usually defined in a broad term during system specification and is designed in details during the system design. System specifications define how interfaces fit into the new process and the kinds of input and output they should provide. The detailed design describes the actual screen layouts that make up these inputs and outputs.

Figure 5.13 shows the main page of SMRCS. User can choose to login or sign in as a new user before they can use SMRCS. User interface for Login page is shown in figure 5.14. And finally the registration page is shown in figure 5.15.



Figure 5.13 : Main page of SMRCS

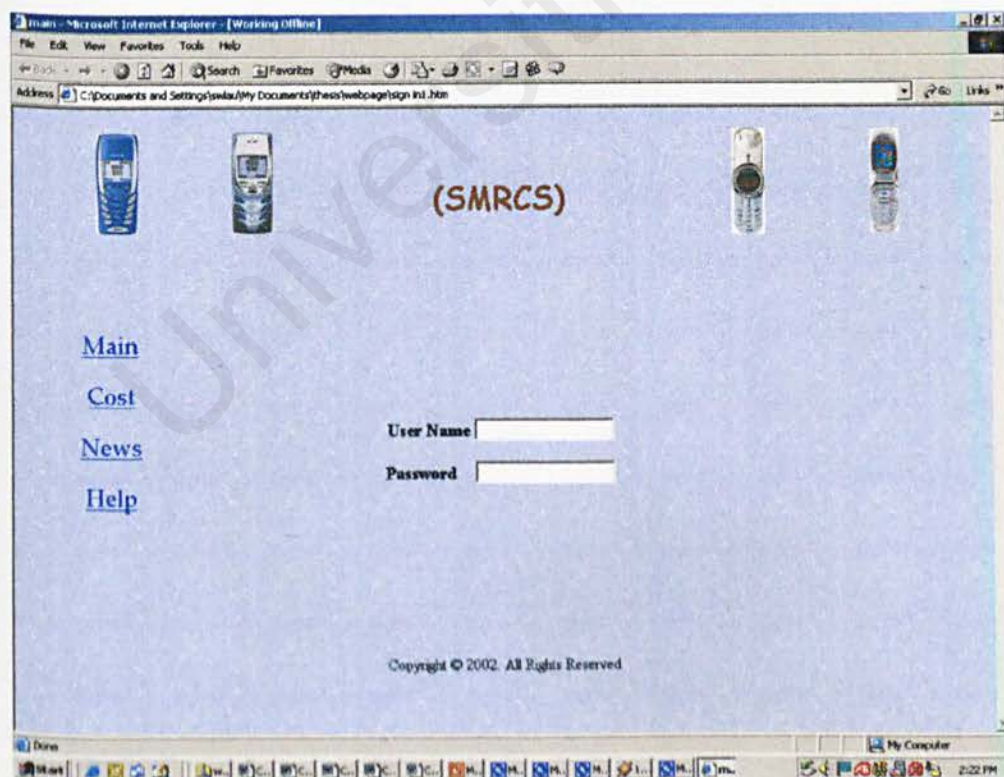


Figure 5.14 : Login page SMRCS

Registration - Microsoft Internet Explorer - [Working Offline]

File Edit View Favorites Tools Help

Address Go Links

Registration

Take a few seconds to register and you will receive great benefits!

User ID:

Password:

Confirm Password:

First Name:

Last Name:

Email:

Birthday: Day Month Year

Sex: Male

Country:

Copyright © 2002. All Rights Reserved

Done My Computer 2:35 PM

Figure 5.15 : Registration page of SMRCS

5.6 Summary of Chapter 5

Chapter 5 provides a general overview of system design for this project. This general design for the new system will eventually guide the detailed design and construction of that system. The system design includes program design, input form design, user interface design and database design. The design of user interface is crucial because user acceptance of the system is frequently dependant on a friendly and easy to use interface. Database must be carefully designed to ensure adaptability and flexibility during system lifetime.

Chapter 6 : System Implementation



- 6.1 Introduction
- 6.2 Development Environment
- 6.3 System Development
- 6.4 Summary of Chapter 6

Chapter 6 : System Implementation

6.1 Introduction

The implementation phase takes place after the system design phase. System implementation is a process that converts the system requirements and design into program codes. This phase at time involve some modifications to previous design.

6.2 Development Environment

The development environment has certain impact on the development of a system. Development environment consists of hardware and software configuration. Using the suitable hardware and software is an important factor to determine the success of the project. The hardware and software tools used to develop and document the entire system are as listed as below.

6.1.1 Hardware Configurations

The following hardware specifications have been used to develop the system.

- ❖ Intel Pentium
- ❖ 256 MB SD RAM
- ❖ 15.0 GB Hard Disk

- ⊕ 14" 256-colour monitor capable of 1024 x 768 resolution
- ⊕ 1.44 MB Floppy Drive
- ⊕ 40X CD-ROM Drive
- ⊕ Other Standard computer peripherals

6.1.2 Software Configurations

The software tools used for system development are vital to the successful implementation of SMRCS. The software specifications used in the development of this project are illustrated in table 5.1.

Software	Usage	Description
Microsoft Windows 2000	System Development	Operating System
Professional		
VB.NET	System Development	Build the application
Microsoft SQL Server 2000	Database	Build the database to store and manipulate the data
Paint Shop Pro 7	Interface Design	Graphics Editor
Microsoft Word	System Development	Documentation

Table 6.1 : Software Specifications

6.3 System Development

SMRCS is classified as windows application. All information is coded in VB.NET before being presented. Server side scripts are inserted to allow server side processing. Client side scripts are written mostly to perform validation of text input by user.

The design must be translated into the form that can be understood by the machine. Basically, the development of SMRCS is divided into three stages, which are data presentation, coding and integration.

6.3.1 Data Presentation

6.3.1.1 Still Images

Still images, mainly are phone model, included in the application with the primary objective to provide user with links to change phone model. These images are JPEG (jpg) interleaved graphic format. All of the images are created and edited using graphics editor such as Paint Shop Pro 7 and Adobe Photoshop.

6.3.1.2 Database Preparation

Microsoft SQL Server 2000 is used as the database for SMRCS. Firstly, we have to install Microsoft SQL Server 2000. By using this, creation and modification of data can be made easily. Then, the entire table listed in Chapter 5 are created using

Microsoft SQL Server 7.0. It is an important step to do before the coding of application that involves process of data input by user that involve the database.

6.3.1.3 Input Form Design

SMRCS is a windows application that involves a lot of data input process by users that involve the database. Therefore, form design need to be done carefully so that the user know how to use the system effectively. VB.Net was used to develop and design the forms.

6.3.2 Coding

6.3.2.1 Introduction

The design must be translated into the form that can be understood by the machine. The code generation step performs this task.

Visual Studio .NET allows programmers to create and deploy critical server-based business logic. Historically, server-based programming has been tedious to code, prone to error, and difficult to test. With Visual Studio .NET, developers can visually compose middle-tier components using the Visual Component Designer (VCD). The VCD enables developers to drag and drop non-visual objects such as message queues, timers, and event logs, to a design surface, automatically discovering all necessary server-based resources and configuring required components.

Visual Studio .NET makes it simple to create solutions that span any device. With its powerful WYSIWYG designer for Web pages, HTML IntelliSense, and Style Sheet Editor, Visual Studio .NET helps developers feel comfortable authoring complex Web solutions while leveraging Visual XML designers and XML IntelliSense for drag and drop creation and manipulation of data. With automatically generated client-side validation code, Web developers can rest assured that their application works in Internet Explorer and Netscape, reducing the amount of JavaScript code necessary to write.

Windows developers will find the new Windows forms to be intuitive and efficient as they construct code using any .NET language, including Visual Basic .NET or Visual C# .NET. With Visual Inheritance, developers can greatly simplify the creation of Windows applications by centralizing in parent forms the common logic and user interface for their entire solution. Using control anchoring and docking, programmers can provide resizable forms automatically without code, while the in-place menu editor enables developers to visually author menus directly from within the Forms Designer.

With application wizards, project templates, and example source code developers can rapidly create Windows, Web, and device applications with minimal up-front investment. Dynamic Help and the Microsoft Developer Network (MSDN) provide assistance based on the current task and programming language, ensuring that developers are never at a loss for information on the .NET platform or their language of choice. Visual Studio Macros, like VBA macros in Office, enable

automation of routine tasks within the IDE, further enhancing the overall productivity of Visual Studio developers.

Finally, developers can choose from a set of modernized languages that gives them the most appropriate means to solve their business problems. Visual Basic .NET includes the familiar syntax Visual Basic developers are accustomed to plus optional Object Oriented Programming features including inheritance and other optional power features including structured exception handling, and free-threading.

[5]

6.3.2.2 Coding Principles

Several principles are applied during the development of this system to ensure that the quality and the proper structure in the code generation.

Reuse

Reuse has been long been touted as a method for improving product quality throughout the software development process. It is important to create the components (classes) to be reused in the subsequent and related application. Productivity can be increased not only by reducing coding time but also by reducing testing and documentation time.

Readability

Codes should be easy to read and understandable. It is very important when it comes to the enhancement of the system in the future by other people. In addition, the meaningful variable names and statement labels will also be helpful in reading and understanding the code.

Maintainability

Codes should be easy to read, corrected and revised. Codes that perform functions for a module should be grouped together. Besides this, the code should be tried as simple as possible with doing in separate module. It is called loose coupling.

Robustness

Robustness refers to the quality that causes a system to be able to handle unexpected error and echo back with proper responses. Error handling should be done to increase the robustness of the system. Appropriate errors message should be displayed response to user's input. System failure should be minimized or avoided.

6.4 Summary of Chapter 6

The implementation phase is divided into several stages. During the implementation phase, system debugging is being carried out occasionally to prevent any of the system bugs. The next chapter discuss on system testing.

Universiti Malaya

Chapter 7 : System Testing



- 7.1 Introduction
- 7.2 Types of Testing
- 7.3 Test Case
- 7.4 Summary of Chapter 7

Chapter 7 : System Testing

7.1 Introduction

The purpose of testing is to ensure the resulting component of program as well as the program as a whole fulfills the requirements specifications and to eliminate faults in the program. Therefore, a systematic test procedure is needed to make sure the system is tested thoroughly and completely.

Generally there are four basic concepts related to software testing :

- a) Error detection
- b) Error removal
- c) Error tracking
- d) Regression testing

SMRCS follows the classical strategy for testing computer software, begins with testing in the small and works outward towards the testing in the large. Figure 7.1 shows the testing stages. The figure shows the testing process start from component testing, integration testing and finally user testing. However, the back arrow shows that the reverse testing will take place as defects, errors or fault are discovered at any stage. Programming and coding modifications are required to correct it.

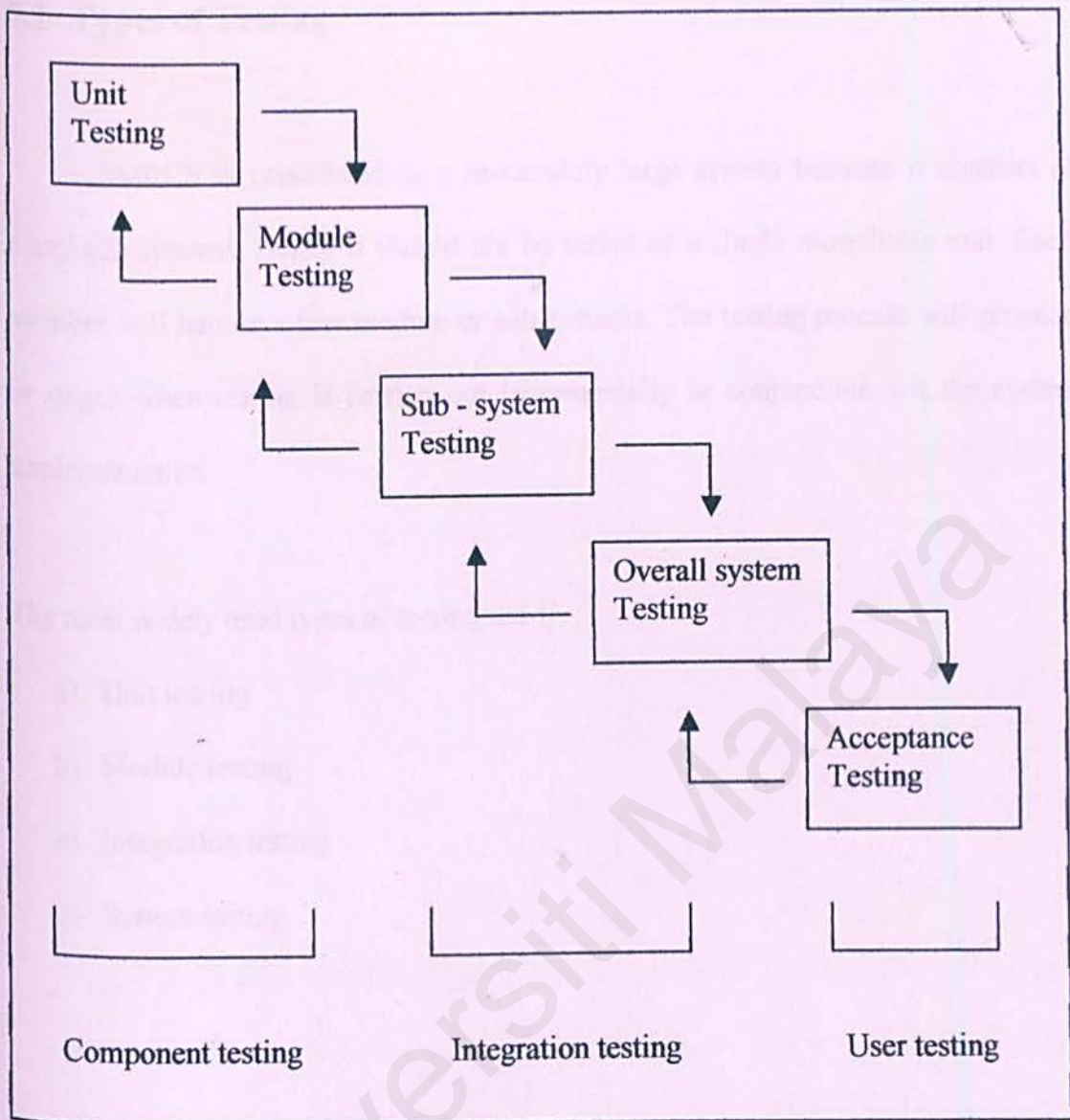


Figure 7.1 The process of System Testing

7.2 Types of Testing

SMRCS is considered as a moderately large system because it consists of many sub-systems. Hence it should not be tested as a single monolithic unit. Each member will handle a few module or sub-systems. The testing process will proceed in stages when testing is carried out incrementally in conjunction with the system implementation.

The most widely used types of testing are :

- a) Unit testing
- b) Module testing
- c) Integrating testing
- d) System testing

7.2.1 Unit Testing

This is a small unit testing where testing are done on individual components of the system to ensure that they operate correctly. Each function is tested independently without other system components. These levels, which are tested, are basically at the field or form level. In this unit testing, SMRCS is tested by form or document submission and input validation.

a) Form/Document Submission

The feedback message will show whether the document is submitted successfully. Another way to verify is by checking the database and see whether the data are already inserted in the database or not.

b) Input Validation

Input validation is to test the inserted data whether they are valid and according to the specified criteria. Each condition of the criteria is entered into the form/field and tested. If the input is incorrect, a pop-up message will appear to warn the user.

7.2.2 Module Testing

A module is a collection of dependent components such as an objective class, an abstract data type or some loose collections of procedures and functions. The main objective of doing module testing is to test the interfacing and integration between the modules that form the system.

The modules are tested with some dummy data. If an error occurs, the related error to that unit is checked and then modified. Then, these units are integrated until no error occurs on the integration module. Sometimes the error is due to the integration coding of the units. Another possibility of the error occurring in module integration is misspelling the name of the database files.

7.2.3 Integration Testing

This phase involves testing collections of modules, which have been integrated into sub-systems. Sub-systems maybe independently designed and implemented. The most common problems that arise in the large software are sub-systems interface mismatch. The integration testing should therefore concentrate on the detection of interface errors by rigorously exercising these interfaces.

Two types of integration testing that are widely used are bottom-up integration and top-down integration. Bottom-up testing is one of the popular approaches used to test large systems. Because the scale of SMRCS is not that large, so it will implement the top-down testing. Then the tested modules will be integrated and testing continues with the next lower level modules. The process is continued until all the modules are tested.

7.2.4 System Testing

When the unit testing and the integration testing had been done, the overall system will be tested. However, system testing is different with before because this test fall outside the scope of the software process. No more care about the logical conditions, loops or how well the integration of the system. The primary purpose of system testing is to fully exercise the computer-based system and some of the test case may try to force the software to fail in a variety of ways. System testing can be categorized into a few types :

- a) Security Testing
- b) Performance Testing.

7.2.4.1 Security Testing

Security attempts to verify that protection mechanism built into a system will protect it from improper penetration. During security testing, the tester plays the role of the individual who desires to penetrate the system. The tester may attempt to acquire password through external clerical means. May attack the system with custom software designed to breakdown any defenses that have been constructed or may browse through insecure data hoping to find the key to system entry.

In SMRCS, security testing is performed by entering the wrong User ID and Password and see how the system reacts. Spaces are not allowed for the User ID and Password. Once the user clicks on the logout link, the session will be terminated and the user will have to log in again.

7.2.4.2 Performance Testing

Performance testing is designed to test the run-time performance of software within the context of an integrated system. For real-time and embedded systems, software that provides required function but does not conform to performance requirements is unacceptable.

Performance testing occurs throughout all the steps in the testing process. It maybe coupled with stress testing. It does not only required the software resources, the hardware resources seem more important in performance testing. That is, it is often necessary to measure the hardware utilization, such as processor cycles. Following are two different computers that have been used in the performance testing of SMRCS.

Computer A	Computer B
AMD K62 550MHz	Intel Pentium III processor
128MB RAM	256 MB RAM
Asus P5A Super 7 Motherboard	Intel Motherboard

Table 7.2 Performance testing table

The results show that the system can perform properly but there is difference in terms of speed execution. Data entry, data retrieval and loading are much faster using Computer B. This is because Computer B consists of 256 MB RAM and a much faster processor than Computer A. Besides, Computer A is connected to the network whereby its performance is very much depending on the traffic load of the network.

7.3 Test Case

A test case is a set of input data and expected results that exercises a system with the purpose of causing failures and detecting faults. Besides reviewing the source codes, some test cases also have been used to test the system. This approach is used as observed. This strategy is needed to identify the variance between the prototype and the requirement. In this testing, different data is input into the program. A number of users were given the opportunity to test the SMRCS system using the test result case and the results were evaluated.

No.	Test Condition	Expected Result	Fail/ Pass	Remark
1	Connecting to the system	Users enter SMRCS with minimal of time	Pass	
2	Sign up as new member	Users can submit sign up form and register as new member	Pass	User ID already exist cannot be use again
3	Registered members login	Users with valid User ID and Password can login to SMRCS	Pass	
4	Edit User Profile	User can update their personal information	Pass	
5	Changing users password	Users can change their old password	Pass	
6	Change phone model	Users can change to different types of phone model	Pass	Doesn't interrupt the input of data
7	Send message	Able to send short message to other members	Pass	Must have enough credits in the account

8	Receive message	Able to receive message from other members and SMRCS	Pass	
9	Delete message	User can delete the message in their inbox and outbox.	Pass	
10	Check result	Users can check the CGPA and GPA if matric number and IC number are match	Pass	Must have enough credits in the account
11	Add Contact	User can add contact ID and contact name in to the phone book	Pass	
12	Modify Contact	User can modify current contact info in the phone book	Pass	
13	Delete Contact	User can delete contact info from the phone book	Pass	
14	Top up	User can top up to enable them to send out more message	Pass	Users must have a valid pin number
15	Check Balance	User can check the current account balance	Pass	
16	Low credits	Server send message to specific user when the credit is low.	Pass	User cannot send message. Only can receive message
17	Logout	User can logout when click on the logout link	Pass	

Table 7.2 Test Case

7.4 Summary of Chapter 7

Chapter 7 discuss how testing can be used to detect errors in a system. It provides the main elements of testing by introducing types of testing conducted. During the program design stage, the system under construction is decomposed into modules. Each module must be unit tested and module tested. The next level of testing is integration testing. Eventually, the whole system is tested.

Universiti Malaya

Chapter 8 : System Evaluation



- 8.1 Problems Encountered and Solutions
- 8.2 System Strength
- 8.3 System Constraints and Future Enhancements
- 8.4 Knowledge and Experience Gained
- 8.5 Summary of Chapter 8

Chapter 8 : System Evaluation

In this phase, SMRCS was evaluated to identify its strengths, limitations and proposals were made for future enhancements.

8.1 Problems Encountered and Solutions

The fundamental knowledge needed a foundation in building an application of this nature involves studies in fields such as the windows based system, information system and housing developer selling procedure. As this project has to be done within a short span of time and a lot of technical issues need to be resolved, a number of problems were encountered throughout the development of the system.

Solutions have been sought during testing and reference check with course mates. The following are some of the major problems encounter during the project system studies, analysis and development.

8.1.1 Difficulties In Determining The Scope of the System

SMRCS was divided into many sub-systems, which was done by two group members. It is hard to determine the project scope for each member. Besides, to build a full scale complete system is impossible within the given time frame.

The project supervisor gave some advices and opinions for us to outline the scope of the project during the initial stages. On the other hand, the results of studying on the existing system have given an outlook of the system scope.

8.1.2 Problems in Choosing Language and Tools

There are quite a number of scripting languages. All the scripting languages and tools allow the user to achieve the same end results for dynamic windows application. Thus it is difficult to determine the most appropriate languages and tools for the development of SMRCS.

To gain more information of windows application and determine the most appropriate approach to use, in depth studies and research on the windows based programming language was carried out in the earlier stage of development.

These activities include Internet surfing, reading topic related magazines and reference books and studying the existing system. To determine which approach to use, seeking advices and views from project supervisor and course mates and engaging in similar project are carried out.

8.1.3 Lack of Knowledge in the Language and Tools

As there is no prior knowledge in programming in a windows based environment, a lot of studies need to be done. New programming language like VB>NET need to be learn within a short time span. During the development of the system, a lot of time spent in looking for solutions to solve the problems that were occurred during that time.

8.2 System Strength

There are several advantages of this system as listed below :

8.2.1 User Friendliness

The system interface design is attractive, user friendly and easily understood by any users. It tells the users how to work with the system. Users have the control of the system function flow by just click on the button. It incorporates a standard homepage with a consistent environment.

8.2.2 Password Protected Members and Administrator Sites

Windows based SMRCS is a password-protected site. Giving authorized admin ID and password prohibit unauthorized admin from accessing all the members

information in the database. This also prevent unauthorized member from using the service offered by SMRCS.

8.2.3 System Transparency

System Transparency refers to the condition where the users do not need to know where the database reside, how is the system structure, its database management system and anything related to the system built. Users are just required to know how to communicate with the user interface.

8.2.4 Reliable System with Efficient Errors Handling

Data input by user is validated and verified to prevent errors caused by invalid input. If there is input failure, an error message is send to inform the user about the error. For example, there is an error message prompted for duplicated of User ID in the new member registration form.

8.2.5 Validation of Input Data

The system is precise on computations and control. Client side validation technique is used to implement on validation data. Only valid data enter, an error message will prompt the user about the error.

8.2.6 Dynamic Database Access Capability

In SMRCS, data used for display and retrieval is stored using a database. Hence, data manipulation can be done easily and efficiently. This will increase data integrity and reliability by reducing data redundancies.

8.3 System Constraints and Future Enhancements

As mentioned before, SMRCS is still not fine enough to work at its full efficiency. Some refining work needs to be done to the system to increase its usability and reliability. The aspects to be refine and some suggestions to upgrade the system are as below:

- ✦ Sound or Audio Effect

When the user receive new incoming message or the credit is low, there will have a beep sound to alert the user.

- ✦ Increase Security for Checking Result

Besides Matric number and IC number, the user has to key in a secret pin number in order to increase the security.

- ✦ Online Topup Transaction

User can buy the topup pin no or credits through online transactions by using credit cards. This offers more convenience to the users

✦ Add in Games

Games such as snakes or pairs can be added in. This offers users another kind of service and entertainment.

✦ Checking Result for Each Course

Besides CGPA and GPA, the user can check the result for each course.

8.4 Knowledge and Experience Gained

Some of the knowledge and experience that I had gained from the development of SMRCS are documented as follows.

8.4.1 Time Management

A project would not be successful without proper and effective time management skills. During the development of SMRCS, there were many workloads of other subjects. It was a struggle to concentrate in developing the system as other courses also allocated a lot of assignments and tests.

Time management must be managed in an effective way such that I could complete the coursework on time while still can dedicate sufficient time for the system in order to accomplish the system as the planned schedule. This project has helped me to realize the importance of time management.

8.4.2 Documentation

Project documentation also serves as a learning experience as the requirement for document has to follow the standard and procedure that has been stated. The entire project had been documented in a professional manner. The quality of the documentation will be measured and be applicable in the development of good software document.

8.4.3 Improving skills in Programming, Database Design, System Analyzing and Designing

Knowledge was gained through programming coding and concepts, learning the dynamic database, designing format and analyzing user's need. Through the coding of the entire system, I have increased mastery of some programming language such as VB.NET. I am more familiar now with the algorithm and logic of system coding. Besides, it is very essential to capture the user's requirements as to develop a system that issuer-friendly, readable and with clear instructions and guidance.

8.5 Summary of Chapter 8

Chapter 8 provides the context for system evaluation and its importance in a project. During system evaluation, the process focuses on the problems encountered during system development and identify alternatives solutions, the evaluation by end users, the system strength and the system constraints, the possible future enhancements that can be carried out and the knowledge and experience gained throughout the System Development Life Cycle (SDLC). This chapter concludes by mentioning the overall achievements that have been accomplish in SMRCS.

Overall, it can be said that SMRCS has actually reached its objective in developing a window-based application by fulfilling all the functional and non-functional requirements. However, there are still many rooms for improvement in SMRCS, in terms of deploying a comprehensive online application management system and moving towards online system maintenance.

A lot of research, time and effort have been involved in making this project successful and in fulfilling the system requirements. A comprehensive knowledge in building a window-based application is also necessary and important especially knowledge about the client and server. Knowledge and experience gained during the system development will no doubt be very useful in my future undertakings.

Appendix A



Universiti Malaya

Contents

Page

Appendix A – User Manual

1.0	Guide to Navigate and Browse SMRCS	135
1.1	Administration Module	135
1.1.1	Startup for Administrator	136
1.1.2	Startup for Server	141
1.2	User Module	143
1.2.1	Startup for users	143
1.2.2	Register as New User	144
1.2.3	User Login	145
1.2.4	Edit Registration Details	146
1.2.5	Sending Short Message	147
1.2.6	Message	148
1.2.7	Checking Result	150
1.2.8	Phone Book	151
1.2.9	Checking Balance	153
1.2.10	Topup	154
1.2.11	Model Selector	155
1.2.12	View SMRCS Profile	158

Appendix B - Source code for SMRCS	159
------------------------------------	-----

User Manual

1.0 Guide to Navigate and Browse SMRCS

SMRCS is divided into 2 modules, the user module and the administration module. Each module has different functionality and purposes.

1.1 Administration Module

The administration module enable administrator of SMRCS to monitor and manage the information of the current members and other details such as the database for students result and Card ID. Administration management duties involve adding, editing, deleting or updating the database, manage the server and others.

Besides the server, the administration has a different application. This is to separate the accessibility of the users and administrator. This application is strictly for SMRCS administrator with valid username and password in order to login and access the application.

1.1.1 Startup for administrator

- ✚ Begin using the system by running the administration application.
- ✚ A welcome form will appear for administrator to login.



The screenshot shows a web browser window titled "SMRCS". The main heading is "SMRCS Administrator Login" in a yellow box. Below this, it says "Welcome to Administrator Page !!". A message prompts the user to "Please enter your user name and password to login to the system". There are two input fields: "Admin ID" with the value "admin1" and "Password" with masked characters "*****". A "Sign In" button is located at the bottom right of the form.

Figure 1.1 : Administrator Welcome and Login Form

- ✚ The administrator needs to fill in a valid username and password click on the "Sign In" button.
- ✚ For successful login, it will show the user administration form and enable the administrator to view all the information of the available registered users.

SMRCS
User Administration

User_ID : 0121111111

First_name	<input type="text" value="John"/>	Sex	<input type="text" value="Male"/>
Last_name	<input type="text" value="Cheng"/>	Country	<input type="text" value="Singapore"/>
Date of Birth	<input type="text" value="23/06/1980"/>	Email Address	<input type="text" value="john@yahoo.com"/>

Administration Section

[Users](#) [Admin](#) [Result](#) [Reload Card](#) [Server Storage](#) [Credit List](#) [Logout](#)

Figure 1.2 : User Administration

When administrator clicks on the Admin link, it will lead to the Administrator Administration form as shown in figure 1.3.

SMRCS
Administrator Profile

Admin_ID : admin1

First_name	<input type="text" value="Lee"/>	Sex	<input type="text" value="Female"/>
Last_name	<input type="text" value="Dee Khoo"/>	Country	<input type="text" value="Malaysia"/>
Date of Birth	<input type="text" value="19/09/1979"/>	Email Address	<input type="text" value="lblee@smrcs.com.my"/>

Administration Section

[Users](#) [Admin](#) [Result](#) [Reload Card](#) [Server Storage](#) [Credit List](#) [Logout](#)

Figure 1.3 Administrator Administrations

- ✚ When the administrator clicks on the result link, it will lead to the Student Result Administration form.
- ✚ Administrator can choose to add, modify, delete or update the database.

Metric_No	IC_No	Result_CPA	Result_CPA
WEK 000001	800101015111	2.58	3.69
WEK 000002	790202025222	3.21	3.15
WEK 000003	790303035333	3.44	3.43
WEK 000004	810404045444	3.03	3.03
WEK 000005	800505055555	2.98	2.56

Update Database

Administration Section

[Users](#)
[Admin](#)
[Result](#)
[Reload Card](#)
[Server Storage](#)
[Credit List](#)
[Logout](#)

Figure 1.4 : Student Result Administration

- ✚ When the user clicks on the Reload Card link, it will lead to the Topup Card Administration form.
- ✚ Administrator can add in new Pin Number and to update the table.

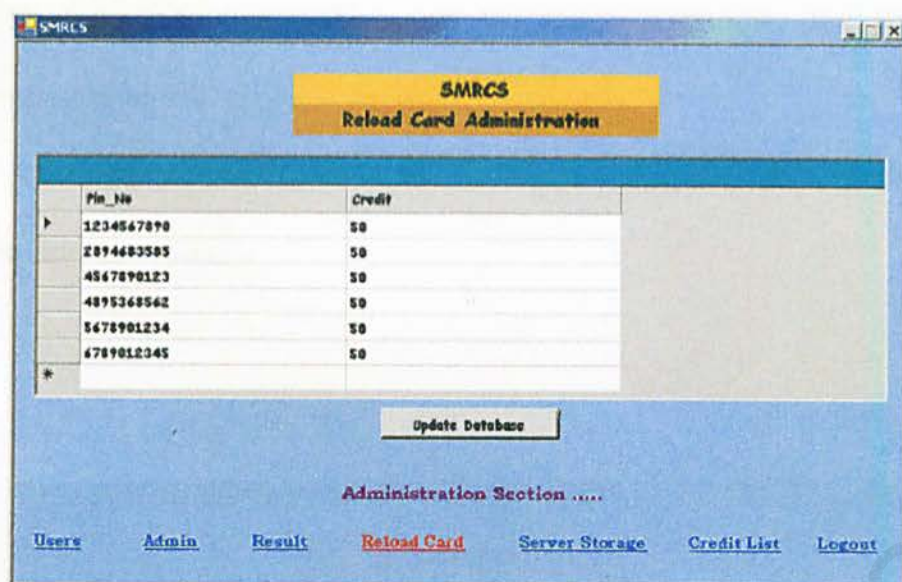


Figure 1.5 : Topup Card Administration

When the administrator clicks on the Server Storage link, the server storage administration form will appear.

Administrator can view the message and update the table.

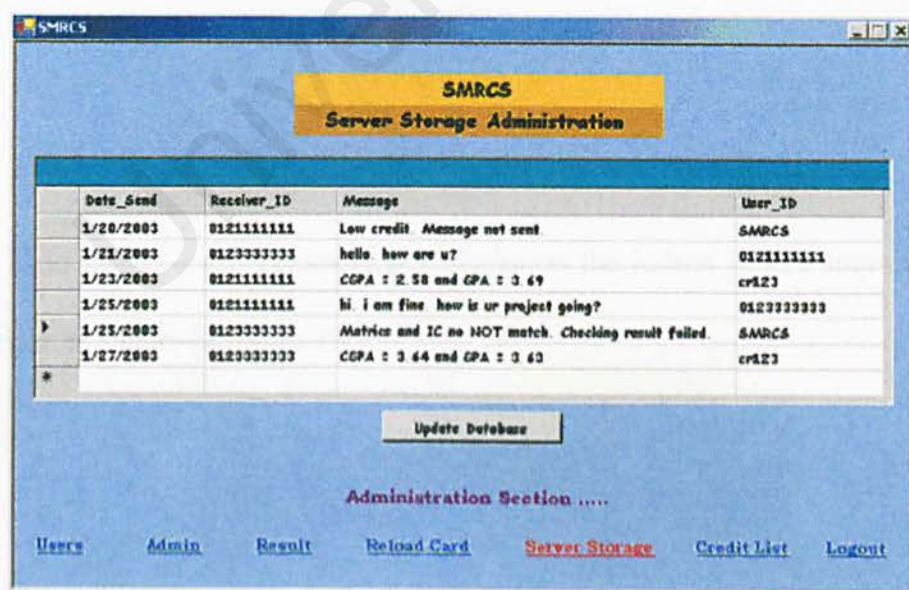


Figure 1.6 : Server Storage Administration

✚ If the administrator clicks on the Credit List link, the Credit List Administration will appear.

✚ Administrator can choose to add in new record, modify the current record or even delete a record.

Directory	Description	Credit
cb123	Checking Balance	0
Contact_ID	Sending SMS	3
cr123	Checking Result	4
* tu123	TopUp Balance	0

Update Database

Administration Section

[Users](#) [Admin](#) [Result](#) [Reload Card](#) [Server Storage](#) [Credit List](#) [Logout](#)

Figure 1.7 : Credit List Administration

✚ And lastly, if the administrator clicks on the logout link, a logout form will appear as shown in figure 1.7.

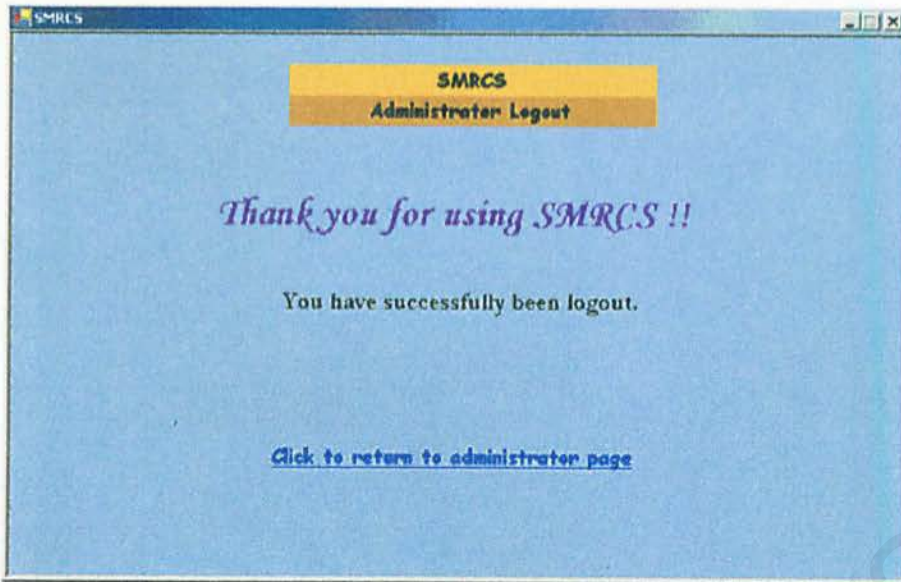


Figure 1.8 : Logout form

✚ When administrator clicks on the return to administrator page link, the administrator welcome and login form will appear as shown in figure 1.1 above.

1.1.2 Startup for server

✚ The server side has to be active before the user can use the services provided by SMRCS.

✚ The figure below shows when the server started.

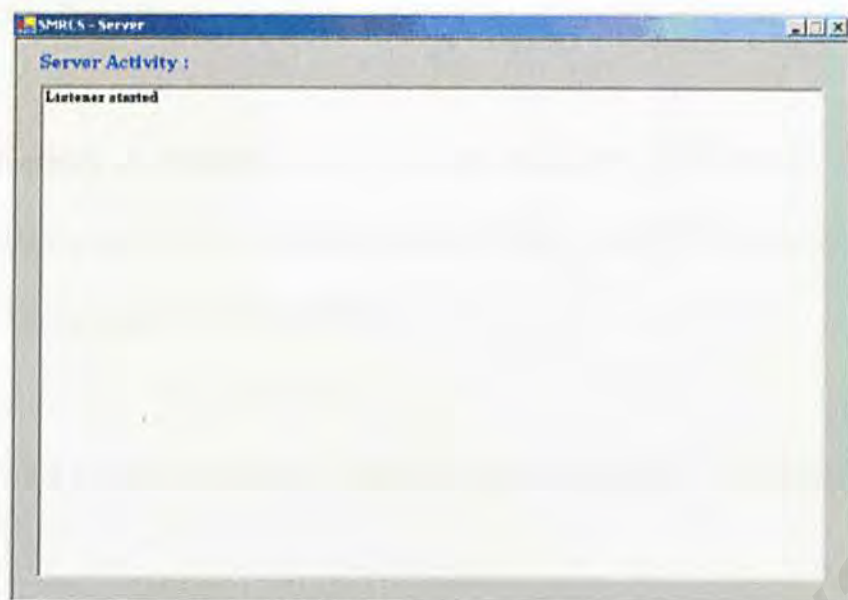


Figure 1.2 : SMRCS Server

✚ The server will list out all the activities for each user. This provides easier management for the administrator.

✚ After the server has run for some time, it will look like the figure 1.3.

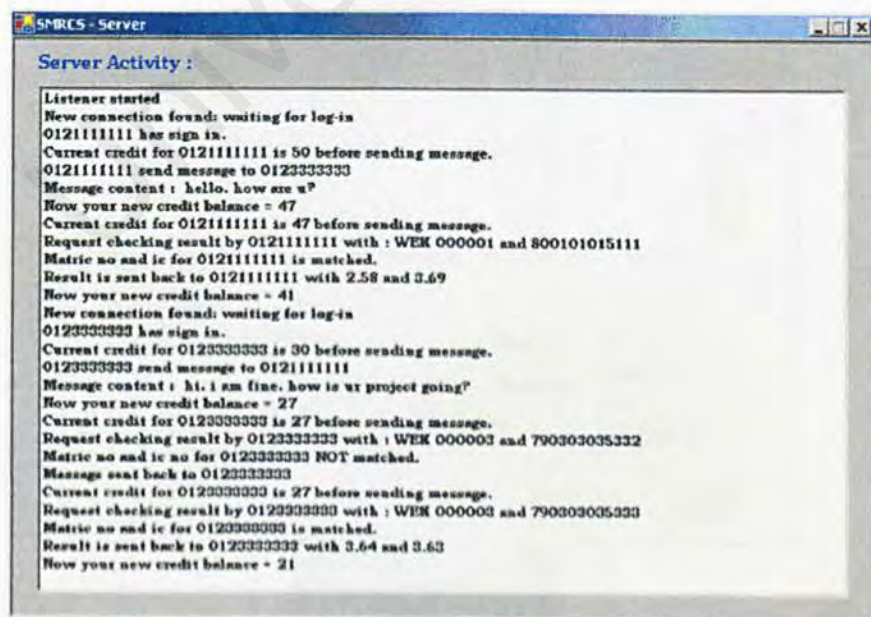


Figure 1.3 : Server Activity

1.2 User Module

Members of SMRCS need to login before they can access the services provided by the application. For example, members can send short messages, check result, and save contact's information.

The following are the step-by-step guide on how to use SMRCS for users :

1.2.1 Startup for users

- ✚ Begin using the system by running the SMRCS Client.exe file.
- ✚ The client side windows will look like figure 2.1



Figure 2.1 Client Form

1.2.2 Register As New User

✚ If the user is not a member of SMRCS, they can register as new user by clicking on the New User link.

✚ The registration form will pop up as shown in figure 2.2



The screenshot shows a Windows-style window titled "User Registration" from the "SMRCS - Client" application. The window has a blue title bar and standard window controls. The main area is white with a decorative border featuring two mobile phones. The title "Registration" is centered at the top in a red, cursive font. Below the title, there are several input fields and dropdown menus. The "User ID" field contains "0128888888" with a note "(10 digit integer)". The "Password" field is masked with asterisks and has a note "(max 6 digit character)". The "First Name" field contains "Cheong", "Last Name" contains "Siew Yee", and "Email" contains "choong@smrct.com". The "DOB" field has dropdowns for "0", "Aug", and "1988". The "Sex" dropdown is set to "Female". The "Country" dropdown is set to "Hong Kong". At the bottom right, there are two yellow buttons: "Cancel" and "Submit".

Field	Value	Notes
User ID	0128888888	(10 digit integer)
Password	*****	(max 6 digit character)
First Name	Cheong	
Last Name	Siew Yee	
Email	choong@smrct.com	
DOB	0 / Aug / 1988	
Sex	Female	
Country	Hong Kong	

Figure 2.2 : Registration Form

✚ After filling in all the fields in the registration form, the user have to click on the submit button to submit the form.

✚ A popup message box will tell the user if the registration is successful.

1.2.3 User Login

✚ For registered user, they have to fill in a valid username and password before clicking on the Login button.



Figure 2.3 : Login Form

✚ Successful login will lead the user to the phone simulator, which will allow the user to start using all the services provided by SMRCS.



Figure 2.4 : Phone Simulator

1.2.4 Edit Registration Details

As a registered user, you can update your information by clicking on the Edit Profile link on the menu bar.

A form containing your current information will appear as shown in figure 2.5.

SMRC5 - Client
File Help

Edit Profile

Edit Profile

User ID 0121111111 (10 digit integer)

Password cccc (max: 6 digit character)

First Name John **DOB** 25/05/1980 (dd/mm/yyyy)

Last Name Chong **Sex** Male

Email john@yahoo.com **Country** Singapore

Cancel Save

Figure 2.5 : Form Edit Profile

After updating all the information, user can submit the changes by clicking on the Save button at the bottom of the form. The changes in the profile will be saved.

1.2.5 Sending Short Message

- Click on the menu button and the menu screen will appear.
- Click on Send SMS link and the send SMS screen will appear.
- Type in the message and click on the OK button.
- The directory screen will appear. Key in the directory or click on the Select directory link to select a directory from the phone book.

- Click OK button after key in the directory
- The message sent screen would show the user that the message had been successfully sent.

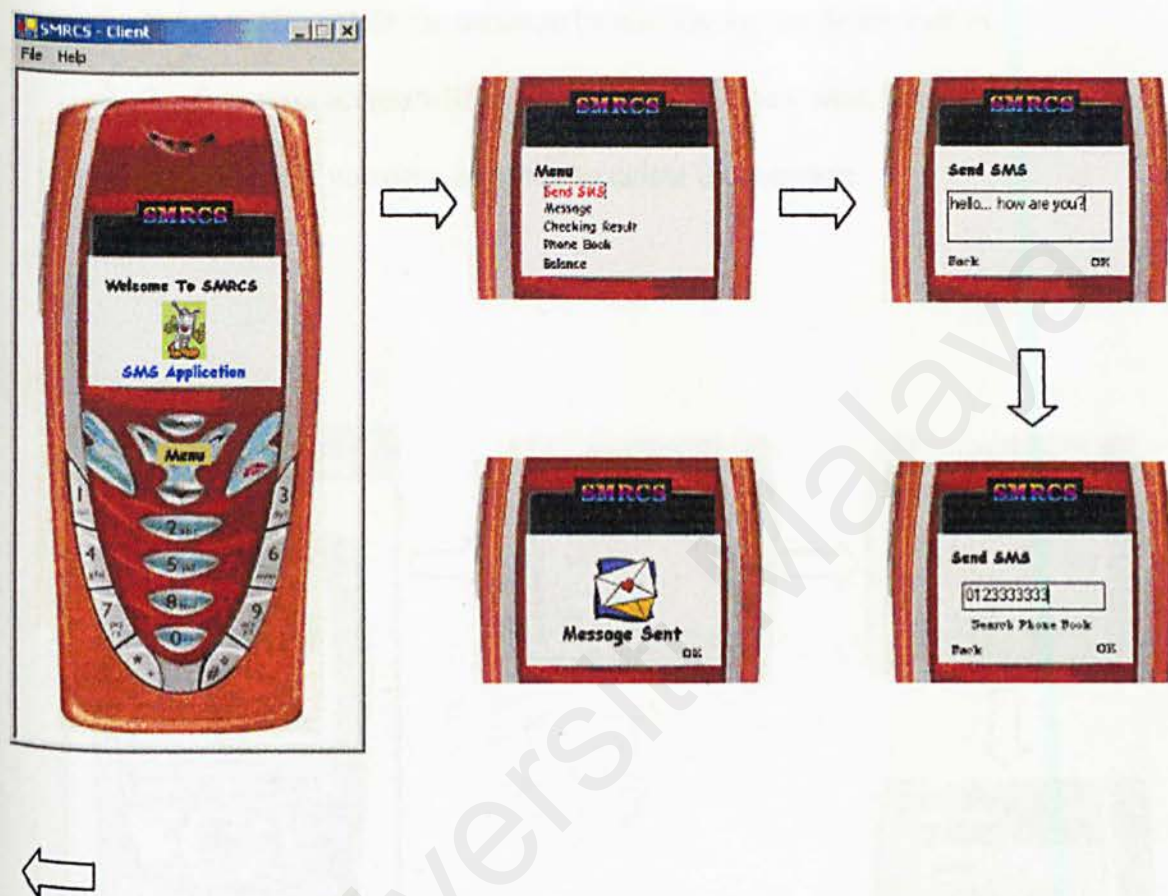


Figure 2.6 : Process of sending message

1.2.6 Message

- Click on the menu button and the menu screen will appear.
- Click on the message link and you can choose between Inbox and Outbox.

Click on the Inbox if you want to read incoming message and click on the Outbox if you want to check your message outbox.

Inbox screen or Outbox screen will be shown depending on what you choose.

You can click the next or back button to read the next or previous message.

You can also delete the message by clicking on the delete button.

Confirmation screen will appear. Click OK if you want to delete the message or click Cancel if you does not want to delete the message.



Figure 2.7 : Process for Inbox



Figure 2.8 : Process for Outbox

1.2.7 Checking Result

- Click on the menu button and the menu screen will appear.
- Click on the Checking result link and the checking result screen will appear.
- Type in the Matric number and IC number in the box provided and click on the OK button.

✚ The directory screen will appear. Key in the directory, which is “rc123” and click on the OK button.

✚ The message sent screen would show the user that the message had been successfully sent.



Figure 2.9 : Process of Checking Result

1.2.8 Phone Book

- ✚ Click on the menu button and the menu screen will appear.
- ✚ Click on the phonebook link and the display contact screen will appear.
- ✚ Click on the next or previous button to view the entire phonebook.
- ✚ You can also choose to add, modify or delete the contact in the phonebook.

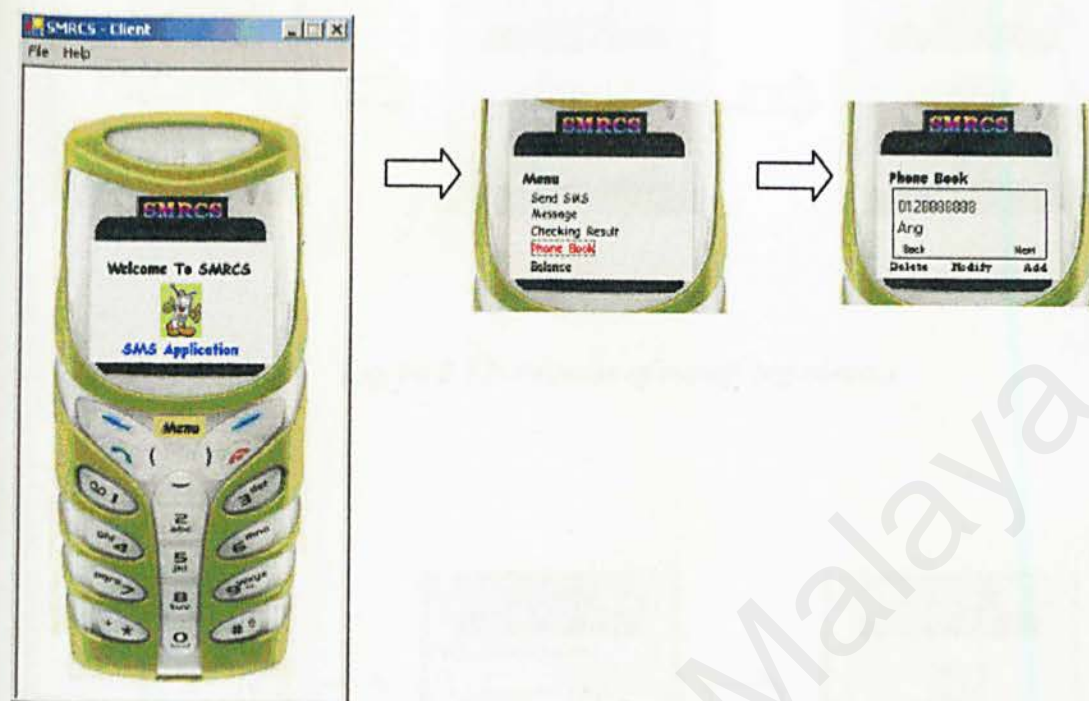


Figure2.10 : Process of displaying phonebook



Figure 2.11: Process of adding new contact



Figure 2.12: Process of modifying contact



Figure 2.13: Process of deleting contact

1.2.9 Checking Balance

- ✚ Click on the menu button and the menu screen will appear.
- ✚ Click on the balance link.
- ✚ Screen directory will appear.
- ✚ key in the directory or select from the phonebook. The directory for checking balance is "cb123".

Screen message sent will appear that indicates that the message has been successfully sent to server.

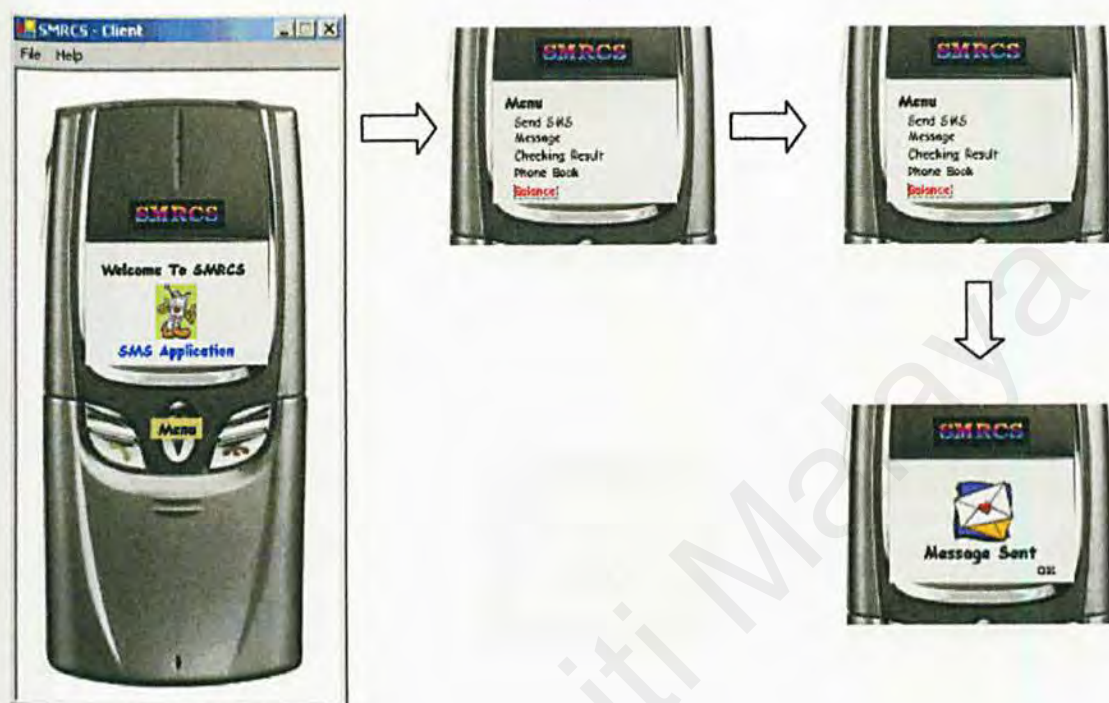


Figure 2.14: Process of Checking Balance

1.2.10 Topup

- Click on the menu button and the menu screen will appear.
- Click on the Send SMS link and the screen send SMS will appear.
- Type in the 10 digit pin no in the text box and then click the OK button.
- The screen directory will appear. Key in the directory or select from the phonebook. The directory is "tu123".
- Click the OK button after key in the directory.

✚ Screen message sent will appear that indicates that the message has been successfully sent to server.

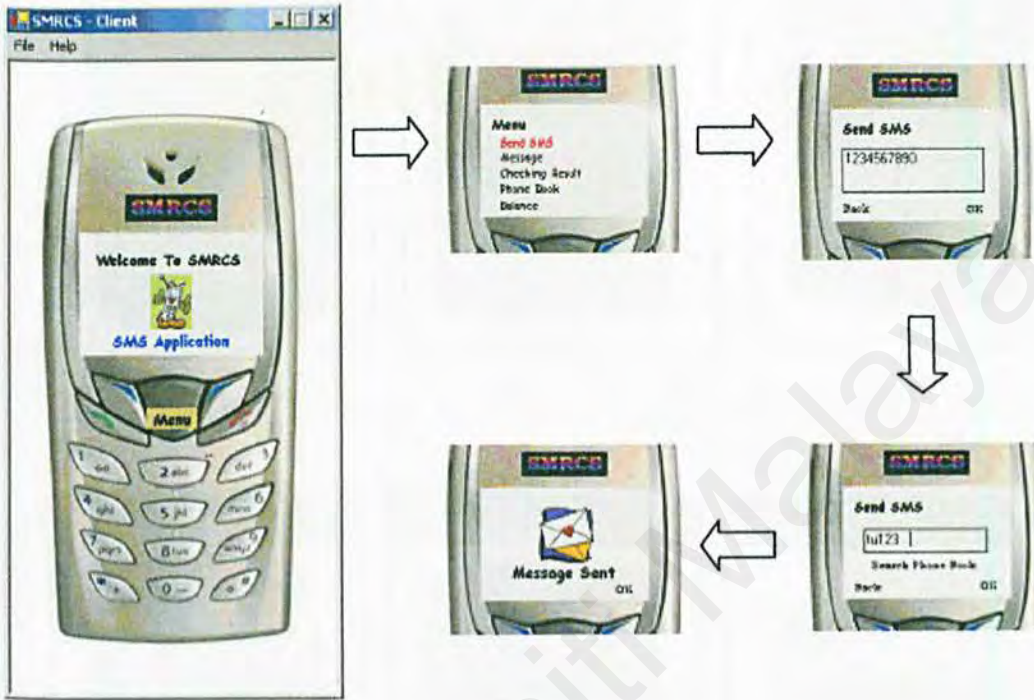


Figure 2.15: Process of Topup

1.2.11 Model Selector

✚ To change the phone model, click on the menu, model selector and the model you want to choose.

✚ Here are all the phone models available for selection.





1.2.12 View SMRCS Profile

✚ To view SMRCS Profile (About SMRCS), click on the Help – About on the menu bar.

✚ Click about SMRCS and the form will appear as shown in figure 2.16.



Figure 2.16 : About SMRCS Form

Appendix B



Universiti Malaya

Source Code For SMRCS

User Module

Client Side

```
Imports System.Net.Sockets
Imports System.Text
Imports System.Data
Imports System.Data.SqlClient
Imports System.IO

Public Class frmMain
    Inherits System.Windows.Forms.Form

    Dim dsUser_Contact As New DataSet()
    Dim dsServer_Storage As New DataSet()
    Dim dsClient_Storage As New DataSet()
    Dim daUser_Contact As New SqlDataAdapter()
    Dim daServer_Storage As New SqlDataAdapter()
    Dim daClient_Storage As New SqlDataAdapter()
    Dim connsms As SqlConnection = New
    SqlConnection("Data Source=localhost;" & "Initial
    Catalog=sms;Integrated Security=SSPI")
    Dim cmdUser_Contact As New SqlCommand()

    Const READ_BUFFER_SIZE As Integer = 255
    Const PORT_NUM As Integer = 8088

    Private client As TcpClient
    Private readBuffer(READ_BUFFER_SIZE) As Byte
    Dim strusername As String

    Private Sub AttemptLogin()

        Dim frmConnectUser As New frmConnectUser()
        frmConnectUser.StartPosition =
        FormStartPosition.CenterParent
        frmConnectUser.ShowDialog(Me)
        SendData("CONNECT|" & frmConnectUser.txtUserLogin.Text & "|"
        & frmConnectUser.txt_password.Text)
        strusername = frmConnectUser.txtUserLogin.Text

        frmConnectUser.Hide()
    End Sub

    ' Send the contents of the Send textbox if it isn't blank.
    Private Sub btnOK_Click(ByVal sender As System.Object, ByVal e
    As System.EventArgs) Handles btnOK.Click
        txtDirectory.Clear()
        txtDirectory.DataBindings.Clear()
        If txtSend.Text <> "" Then
            skrinSend.Hide()
            skrinDirectory.Show()
        End If
    End Sub

    Private Sub btnCROK_Click(ByVal sender As System.Object, ByVal e
```



```

As System.EventArgs) Handles btnCROK.Click
    If txtMatric_No.Text <> "" And txtIC_No.Text <> "" Then
        skrinCResult.Hide()
        skrinDirectory.Show()
    End If
End Sub

Private Sub btnSendOK_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles btnSendOK.Click

    If txtDirectory.Text = "cr123" Then
        SendData("REQUESTRESULTS|" & txtMatric_No.Text & "|" &
txtIC_No.Text)
    ElseIf txtDirectory.Text = "cb123" Then
        txtSend.Text = "1"
        SendData("SEND|" & txtSend.Text & "|" &
txtDirectory.Text)
    ElseIf txtDirectory.Text = "tu123" Then
        SendData("SEND|" & txtSend.Text & "|" &
txtDirectory.Text)
    Else
        If txtDirectory.Text <> "" Then
            SendData("SEND|" & txtSend.Text & "|" &
txtDirectory.Text)

            Dim strSelectClient_Storage As String = "SELECT *
FROM Client_Storage ORDER BY Date_Saved"
            Dim strConnString As String = "server=localhost;
database=sms;integrated security=SSPI;"
            Dim connsms As New SqlConnection(strConnString)
            Dim strInsertCommand As String = "INSERT INTO
Client_Storage(Date_Saved,Receiver_ID,User_ID,Message)
" &
"VALUES(@Date_Saved,@Receiver_ID,@User_ID,@Message)"
            Dim daClient_Storage As New SqlDataAdapter()
            Dim dsClient_Storage As New DataSet()
            Dim cmdSelectClient_Storage As SqlCommand = New
SqlCommand(strSelectClient_Storage, connsms)
            Dim cmdInsertClient_Storage As New
SqlCommand(strInsertCommand, connsms)
            daClient_Storage.SelectCommand =
cmdSelectClient_Storage
            daClient_Storage.InsertCommand =
cmdInsertClient_Storage

            Try
                connsms.Open()
                daClient_Storage.Fill(dsClient_Storage,
"dtClient_Storageetable")
                cmdInsertClient_Storage.Parameters.Add _
(New SqlParameter
("@Date_Saved", SqlDbType.DateTime, 8)).Value =
System.DateTime.Now
                cmdInsertClient_Storage.Parameters.Add _
(New SqlParameter
("@Receiver_ID", SqlDbType.VarChar, 10)).Value =
txtDirectory.Text
                cmdInsertClient_Storage.Parameters.Add _
(New SqlParameter
("@User_ID", SqlDbType.VarChar, 10)).Value =
strusername
            End Try
        End If
    End If
End Sub

```



```

cmdInsertClient_Storage.Parameters.Add _
(New SqlParameter _
("@Message", SqlDbType.VarChar, 160)).Value =
txtSend.Text
cmdInsertClient_Storage.ExecuteNonQuery()
connsms.Close()

```

```

Catch ex As Exception
    MsgBox("User save message Error: ")
End Try

```

```

End If
End If
txtSend.Clear()
txtMatric_No.Clear()
txtIC_No.Clear()
txtDirectory.Clear()

```

```

skrinSuccess.Show()
skrinDirectory.Hide()

```

End Sub

```

Private Sub btnCRCancel_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnCRCancel.Click
    skrinMenu.Show()
    skrinCResult.Hide()
    txtMatric_No.Clear()
    txtIC_No.Clear()
End Sub

```

End Sub

```

Private Sub DoRead(ByVal ar As IAsyncResult)
    Dim BytesRead As Integer
    Dim strMessage As String
    Dim strDirectory As String

```

Try

```

' Finish asynchronous read into readBuffer and return
number of bytes read.

```

```

BytesRead = client.GetStream.EndRead(ar)

```

```

If BytesRead < 1 Then

```

```

    ' If no bytes were read server has close. Disable
    input window.

```

```

    MarkAsDisconnected()

```

```

    Exit Sub

```

```

End If

```

```

' Convert the byte array the message was saved into,
minus two for the

```

```

' Chr(13) and Chr(10)

```

```

strMessage = Encoding.ASCII.GetString(readBuffer, 0,
BytesRead - 2)

```

```

'strDirectory = Encoding.ASCII.GetString(readBuffer, 0,
BytesRead - 2)

```

```

ProcessCommands(strMessage)

```

```

' Start a new asynchronous read into readBuffer.

```

```

client.GetStream.BeginRead(readBuffer, 0,
READ_BUFFER_SIZE, AddressOf DoRead, Nothing)

```

```

Catch e As Exception

```

```

    MarkAsDisconnected()

```

```

    End Try
End Sub

' Send the server a disconnect message
Private Sub frmMain_Closing(ByVal sender As Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
    ' Send only if server is still running.
    If btnOK.Enabled = True Then
        SendData("DISCONNECT")
    End If
End Sub

Private Sub mnuLogout_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles mnuLogout.Click

    ' Send only if server is still running.
    If btnOK.Enabled = True Then
        SendData("DISCONNECT")
    End If
    System.Environment.Exit(System.Environment.ExitCode)
End Sub

' When the form starts, this subroutine will connect to the
server and attempt to
' log in.
Private Sub frmMain_Load(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles MyBase.Load

    Dim frmConnectUser As New frmConnectUser()

    Try
        ' The TcpClient is a subclass of Socket, providing
        higher level functionality like streaming.
        client = New TcpClient("localhost", PORT_NUM)
        'client = New TcpClient("202.185.109.179", PORT_NUM)

        ' Start an asynchronous read invoking DoRead to avoid
        lagging the user interface.
        client.GetStream.BeginRead(readBuffer, 0,
        READ_BUFFER_SIZE, AddressOf DoRead, Nothing)

        ' Make sure the window is showing before popping up
        connection dialog.
        Me.Show()
        AttemptLogin()

    Catch Ex As Exception
        MsgBox("Server is not active. Please start server and
        try again.", MsgBoxStyle.Exclamation, Me.Text)
        Me.Dispose()
    End Try
    updatemsg()
    updateoutbox()
    updatepbook()
End Sub

' When the server disconnects, prevent further messages from
being sent.
Private Sub MarkAsDisconnected()
    MsgBox("Server is not active!!")
    'txtSend.ReadOnly = True

```



```

    btnOK.Enabled = False
    btnSendOK.Enabled = False
    btnCROK.Enabled = False
End Sub

' Process the command received from the server, and take
appropriate action.
Private Sub ProcessCommands(ByVal strMessage As String)
    Dim dataArray() As String

    ' Message parts are divided by "|" Break the string into an
    array accordingly.
    dataArray = strMessage.Split(Chr(124))

    ' dataArray(0) is the command.
    Select Case dataArray(0)

        Case "SEND"

        Case "REFUSE"
            ' Server refused login with this user name, try to
            log in with another.
            AttemptLogin()

    End Select
End Sub

' Use a StreamWriter to send a message to server.
Private Sub SendData(ByVal data As String)
    Dim writer As New IO.StreamWriter(client.GetStream)
    writer.Write(data & vbCr)
    writer.Flush()
End Sub

Private Sub btnsentok_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles btnsentok.Click

    skrinWelcome.Show()
    skrinSuccess.Hide()
    txtDirectory.Clear()
    txtSend.Clear()

End Sub

Private Sub btnBack1_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles btnBack1.Click

    skrinMenu.Show()
    skrinDirectory.Hide()
    txtSend.Clear()
    txtDirectory.Clear()

End Sub

Private Sub btnMenu_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles btnMenu.Click

    skrinWelcome.Hide()
    skrinMenu.Show()
    skrinSend.Hide()
    skrinDirectory.Hide()

```

```

    skrinSelectDirectory.Hide()
    skrinSuccess.Hide()
    skrinMessage.Hide()
    skrinInbox.Hide()
    skrinConfirmDeleteInbox.Hide()
    skrinDeleteInbox.Hide()
    skrinOutbox.Hide()
    skrinConfirmDeleteOutbox.Hide()
    skrinDeleteOutbox.Hide()
    skrinCResult.Hide()
    skrinPBook.Hide()
    skrinContactSaved.Hide()
    skrinAddContact.Hide()
    skrinModifyContact.Hide()
    skrinConfirmDeleteContact.Hide()
    skrinDeleteContact.Hide()
    skrinBalance.Hide()
    txtSend.Clear()
    txtDirectory.Clear()
    txtMatric_No.Clear()
    txtIC_No.Clear()

```

End Sub

```

Private Sub LinkSSMS_LinkClicked(ByVal sender As System.Object,
ByVal e As System.Windows.Forms.LinkLabelLinkClickedEventArgs)
Handles linkSSMS.LinkClicked

```

```

    skrinSend.Show()
    skrinMenu.Hide()

```

End Sub

```

Private Sub btnBack_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles btnBack.Click

```

```

    txtSend.Clear()
    skrinSend.Hide()
    skrinMenu.Show()

```

End Sub

```

Private Sub LinkMessage_LinkClicked(ByVal sender As
System.Object, ByVal e As
System.Windows.Forms.LinkLabelLinkClickedEventArgs) Handles
linkMessage.LinkClicked

```

```

    skrinMessage.Show()
    skrinMenu.Hide()

```

End Sub

```

Private Sub btnMessageBack_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnMessageBack.Click

```

```

    skrinMessage.Hide()
    skrinMenu.Show()

```

End Sub

```

Private Sub txtTimer_TextChanged(ByVal sender As System.Object,

```



```

ByVal e As System.EventArgs)
    'display the current date and time
    txtTimer.Text = System.DateTime.Now

End Sub

Private Sub Timer1_Tick(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles Timer1.Tick
    'for every click of the timer (1 sec.) update the date and
    time
    txtTimer.Text = System.DateTime.Now
    updatemsg()
    updateoutbox()
    updatepbook()
End Sub

Private Sub LinkPBook_LinkClicked(ByVal sender As System.Object,
ByVal e As System.Windows.Forms.LinkLabelLinkClickedEventArgs)
Handles linkPBook.LinkClicked
    dsUser_Contact.Clear()
    updatepbook()
    skrinPBook.Show()
    skrinMenu.Hide()

End Sub

Private Sub updatepbook()
    Try
        cmdUser_Contact = connsms.CreateCommand
        cmdUser_Contact.CommandText = "SELECT * FROM
        dbo.User_Contact WHERE User_ID = '" & strusername & "'"
        daUser_Contact.SelectCommand = cmdUser_Contact
        ' Automatically generate the Update, Insert,
        ' and Delete commands
        Dim cb As SqlCommandBuilder = New
        SqlCommandBuilder(daUser_Contact)
        daUser_Contact.Fill(dsUser_Contact, "User_Contact")

        txtDisplayContactID.DataBindings.Clear()
        txtDisplayContactID.DataBindings.Add( _
        "Text", dsUser_Contact.Tables("User_Contact"),
        "Contact_ID")

        txtDisplayContactName.DataBindings.Clear()
        txtDisplayContactName.DataBindings.Add( _
        "Text", dsUser_Contact.Tables("User_Contact"),
        "Contact_Name")

    Catch ex As Exception
        MsgBox("Error: " & ex.Source & ": " & ex.Message, _
        MsgBoxStyle.OKOnly, "frmUser_Contact_Load")
    End Try

    skrinPBook.Show()
    skrinMenu.Hide()
End Sub

Private Sub btnMoveNext_Click(ByVal sender As System.Object, _
ByVal e As System.EventArgs) Handles btnMoveNext.Click

    Try

```



```

    skrinPBook.BindingContext( _
    dsUser_Contact.Tables("User_Contact")).Position += 1

```

```

Catch ex As Exception

```

```

    MsgBox("Error: " & ex.Source & ": " & ex.Message, _
    MsgBoxStyle.OKOnly, "btnMoveNext_Click")

```

```

End Try

```

```

End Sub

```

```

Private Sub btnMovePrevious_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnMovePrevious.Click

```

```

    Try

```

```

        skrinPBook.BindingContext( _
        dsUser_Contact.Tables("User_Contact")).Position -= 1

```

```

    Catch ex As Exception

```

```

        MsgBox("Error: " & ex.Source & ": " & ex.Message, _
        MsgBoxStyle.OKOnly, "btnMovePrevious_Click")

```

```

    End Try

```

```

End Sub

```

```

Private Sub btnAdd_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles btnAdd.Click

```

```

    skrinPBook.Hide()

```

```

    skrinAddContact.Show()

```

```

End Sub

```

```

Private Sub btnAddOK_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles btnAddOK.Click

```

```

    If txtAddContactID.Text = "" Or txtAddContactName.Text = ""

```

```

Then

```

```

        btnAddOK.Enabled = True

```

```

Else

```

```

    Dim strSelectUser_Contact As String = "SELECT * FROM
    User_Contact"

```

```

    Dim strInsertCommand As String = _
    "INSERT INTO User_Contact(Contact_ID, Contact_Name,
    User_ID) " & _
    "VALUES (@Contact_ID,@Contact_Name,@User_ID)"

```

```

    Dim daUser_Contact As New SqlDataAdapter()
    Dim dsUser_Contact As New DataSet()
    Dim cmdSelectUser_Contact As SqlCommand = New SqlCommand
    (strSelectUser_Contact, connsms)
    Dim cmdInsertUser_Contact As New
    SqlCommand(strInsertCommand, connsms)
    daUser_Contact.SelectCommand = cmdSelectUser_Contact
    daUser_Contact.InsertCommand = cmdInsertUser_Contact
    connsms.Open()

```

```

    daUser_Contact.Fill(dsUser_Contact,
    "dtUser_Contacttable")
    cmdInsertUser_Contact.Parameters.Add _
    (New SqlParameter
    ("@Contact_ID", SqlDbType.Char, 10)).Value =
    txtAddContactID.Text
    cmdInsertUser_Contact.Parameters.Add _

```



```

        (New SqlParameter
        ("@Contact_Name", SqlDbType.VarChar, 50)).Value =
        txtAddContactName.Text
        cmdInsertUser_Contact.Parameters.Add _
        (New SqlParameter
        ("@User_ID", SqlDbType.Char, 10)).Value = strusername

        cmdInsertUser_Contact.ExecuteNonQuery()
        connsms.Close()
        skrinAddContact.Hide()
        skrinContactSaved.Show()
    End If
End Sub

Private Sub btnSavedOK_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnSavedOK.Click
    dsUser_Contact.Clear()
    updatepbook()
    txtAddContactID.Clear()
    txtAddContactName.Clear()
    skrinContactSaved.Hide()
    skrinPBook.Show()
End Sub

Private Sub btnAddCancel_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnAddCancel.Click

    Try
        skrinPBook.BindingContext(dsUser_Contact.Tables
        ("User_Contact")).CancelCurrentEdit()

    Catch ex As Exception
        MsgBox("Error: " & ex.Source & ": " & ex.Message, _
        MsgBoxStyle.OKOnly, "btnAddCancel_Click")
    End Try

    skrinAddContact.Hide()
    skrinPBook.Show()
    txtAddContactID.Clear()
    txtAddContactName.Clear()

End Sub

Private Sub btnModify_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles btnModify.Click

    Try
        txtContact_ID.DataBindings.Clear()
        txtContact_ID.DataBindings.Add( _
        "Text", dsUser_Contact.Tables("User_Contact"),
        "Contact_ID")
        txtContact_Name.DataBindings.Clear()
        txtContact_Name.DataBindings.Add( _
        "Text", dsUser_Contact.Tables("User_Contact"),
        "Contact_Name")

    Catch ex As Exception
        MsgBox("Error: " & ex.Source & ": " & ex.Message, _
        MsgBoxStyle.OKOnly, "frmUser_Contact_Load")
    End Try

```



```

        skrinPBook.Hide()
        skrinModifyContact.Show()
End Sub

Private Sub btnModifyOK_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnModifyOK.Click

    Try
        Me.BindingContext( _
            dsUser_Contact.Tables("User_Contact")).EndCurrentEdit()
        daUser_Contact.Update(dsUser_Contact.Tables("User_Contact"))

    Catch ex As Exception
        MsgBox("Error: " & ex.Source & ": " & ex.Message, _
            MsgBoxStyle.OKOnly, "btnModify_Click")
    End Try

    skrinModifyContact.Hide()
    skrinContactSaved.Show()
End Sub

Private Sub btnModifyCancel_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnModifyCancel.Click

    Try
        skrinPBook.BindingContext( _
            dsUser_Contact.Tables("User_Contact")).CancelCurrentEdit()

    Catch ex As Exception
        MsgBox("Error: " & ex.Source & ": " & ex.Message, _
            MsgBoxStyle.OKOnly, "btnAddCancel_Click")
    End Try

    skrinModifyContact.Hide()
    skrinPBook.Show()

End Sub

Private Sub btnDelete_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles btnContactDelete.Click
    skrinPBook.Hide()
    skrinConfirmDeleteContact.Show()
End Sub

Private Sub btnCDContactOK_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnCDContactOK.Click

    Try

        skrinPBook.BindingContext(dsUser_Contact.Tables
            ("User_Contact")).RemoveAt(skrinPBook.BindingContext( _
            dsUser_Contact.Tables("User_Contact")).Position)
        daUser_Contact.Update(dsUser_Contact.Tables("User_Contact"))

    Catch ex As Exception
        MsgBox("Error: " & ex.Source & ": " & ex.Message, _
            MsgBoxStyle.OKOnly, "btnDeleteYes_Click")
    End Try

    skrinConfirmDeleteContact.Hide()
    skrinDeleteContact.Show()

End Sub

```



```
Private Sub btnCDContactCancel_Click(ByVal sender As  
System.Object, ByVal e As System.EventArgs) Handles  
btnCDContactCancel.Click
```

```
Try  
    skrinPBook.BindingContext(  
        dsUser_Contact.Tables("User_Contact")).CancelCurrentEdit()
```

```
Catch ex As Exception  
    MsgBox("Error: " & ex.Source & ": " & ex.Message, _  
        MsgBoxStyle.OKOnly, "btnAddCancel_Click")  
End Try
```

```
skrinConfirmDeleteContact.Hide()  
skrinPBook.Show()
```

```
End Sub
```

```
Private Sub btnDCOK_Click(ByVal sender As System.Object, ByVal e  
As System.EventArgs) Handles btnDCOK.Click  
    dsUser_Contact.Clear()  
    updatepbook()  
    skrinDeleteContact.Hide()  
    skrinPBook.Show()
```

```
End Sub
```

```
Private Sub btnDirectory_Click(ByVal sender As System.Object,  
ByVal e As System.EventArgs) Handles btnDirectory.Click
```

```
Try  
    txtSelectContactID.DataBindings.Clear()  
    txtSelectContactID.DataBindings.Add(_  
        "Text", dsUser_Contact.Tables("User_Contact"),  
        "Contact_ID")  
    txtSelectContactName.DataBindings.Clear()  
    txtSelectContactName.DataBindings.Add(_  
        "Text", dsUser_Contact.Tables("User_Contact"),  
        "Contact_Name")
```

```
Catch ex As Exception  
    MsgBox("Error: " & ex.Source & ": " & ex.Message, _  
        MsgBoxStyle.OKOnly, "frmUser_Contact_Load")  
End Try
```

```
skrinSelectDirectory.Show()  
skrinDirectory.Hide()
```

```
End Sub
```

```
Private Sub btnSelectNext_Click(ByVal sender As System.Object, _  
ByVal e As System.EventArgs) Handles btnSelectNext.Click
```

```
Try  
    Me.BindingContext(_  
        dsUser_Contact.Tables("User_Contact")).Position += 1
```

```
Catch ex As Exception  
    MsgBox("Error: " & ex.Source & ": " & ex.Message, _  
        MsgBoxStyle.OKOnly, "btnSelectNext_Click")  
End Try
```


End Sub

```
Private Sub btnSelectBack_Click(ByVal sender As System.Object, _  
ByVal e As System.EventArgs) Handles btnSelectBack.Click
```

```
Try
```

```
Me.BindingContext(_  
dsUser_Contact.Tables("User_Contact")).Position -= 1
```

```
Catch ex As Exception
```

```
MsgBox("Error: " & ex.Source & ": " & ex.Message, _  
MsgBoxStyle.OKOnly, "btnSelectBack_Click")
```

```
End Try
```

End Sub

```
Private Sub btnSelect_Click(ByVal sender As System.Object, ByVal  
e As System.EventArgs) Handles btnSelect.Click
```

```
txtDirectory.DataBindings.Clear()  
txtDirectory.DataBindings.Add(_  
"Text", dsUser_Contact.Tables("User_Contact"), "Contact_ID")  
txtDirectory.Clear()  
skrinSelectDirectory.Hide()  
skrinDirectory.Show()
```

End Sub

```
Private Sub btnCancelSelect_Click(ByVal sender As System.Object,  
ByVal e As System.EventArgs) Handles btnCancelSelect.Click
```

```
skrinDirectory.Show()  
skrinSelectDirectory.Hide()
```

End Sub

```
Private Sub lnkInbox_Click(ByVal sender As System.Object, ByVal  
e As System.EventArgs) Handles lnkInbox.Click
```

```
dsServer_Storage.Clear()  
updatemsg()  
skrinInbox.Show()  
skrinMessage.Hide()
```

End Sub

```
Private Sub btnInboxBack_Click(ByVal sender As System.Object,  
ByVal e As System.EventArgs) Handles btnInboxCancel.Click
```

```
skrinMessage.Show()  
skrinInbox.Hide()
```

End Sub

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

End Sub

```
Private Sub updatemsg()
```

```
Dim connsms As SqlConnection = New  
SqlConnection("Data Source=localhost;" &  
"Initial Catalog=sms;Integrated Security=SSPI")  
Dim cmdServer_Storage As New SqlCommand()
```

```
Try
```



```

cmdServer_Storage = connsms.CreateCommand
cmdServer_Storage.CommandText = "SELECT * FROM
dbo.Server_Storage WHERE Receiver_ID = '" & strusername
& "'" ORDER BY Date_Send desc"
daServer_Storage.SelectCommand = cmdServer_Storage

```

```

Dim cb As SqlCommandBuilder = New
SqlCommandBuilder(daServer_Storage)
daServer_Storage.Fill(dsServer_Storage, "Server_Storage")
tbI1.DataBindings.Clear()
tbI1.DataBindings.Add("Text", dsServer_Storage.Tables(0),
"Date_Send")
tbI1.Visible = False
tbI2.DataBindings.Clear()
tbI2.DataBindings.Add("Text", dsServer_Storage.Tables(0),
"User_ID")
tbI2.Visible = False
tbI3.DataBindings.Clear()
tbI3.DataBindings.Add("Text", dsServer_Storage.Tables(0),
"Message")
tbI3.Visible = False

```

```

txtInbox.Text = tbI1.Text & vbCrLf + tbI2.Text & vbCrLf
+ tbI3.Text

```

```

Catch ex As Exception
'MsgBox("Error: " & ex.Source & ": " & ex.Message,
MsgBoxStyle.OKOnly, "skrinInbox_Load")

```

```

End Try

```

```

End Sub

```

```

Private Sub btnInBack_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles btnInBack.Click

```

```

Me.BindingContext(dsServer_Storage.Tables(0)).Position -= 1
txtInbox.Text = tbI1.Text & vbCrLf + tbI2.Text & vbCrLf +
tbI3.Text

```

```

End Sub

```

```

Private Sub btnInNext_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles btnInNext.Click

```

```

Me.BindingContext(dsServer_Storage.Tables(0)).Position += 1
txtInbox.Text = tbI1.Text & vbCrLf + tbI2.Text & vbCrLf +
tbI3.Text

```

```

End Sub

```

```

Private Sub btnInboxDelete_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnInboxDelete.Click

```

```

skrinConfirmDeleteInbox.Show()
skrinInbox.Hide()

```

```

End Sub

```

```

Private Sub btnCDInboxOK_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnCDInboxOK.Click

```

```

Try

```

```

Me.BindingContext(dsServer_Storage.Tables

```



```

        ("Server_Storage")).RemoveAt(Me.BindingContext
        (dsServer_Storage.Tables("Server_Storage")).Position)
daServer_Storage.Update(dsServer_Storage.Tables("Server_Storage"))
Catch ex As Exception
    MsgBox("Error: " & ex.Source & ": " & ex.Message, _
    MsgBoxStyle.OKOnly, "btnCDInbox_Click")
End Try
    skrinConfirmDeleteInbox.Hide()
    skrinDeleteInbox.Show()
End Sub

```

```

Private Sub btnCDInboxCancel_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnCDInboxCancel.Click
    skrinInbox.Show()
    skrinConfirmDeleteInbox.Hide()

```

End Sub

```

Private Sub btnDIOK_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles btnDIOK.Click
    dsServer_Storage.Clear()
    updatemsg()
    skrinInbox.Show()
    skrinDeleteInbox.Hide()

```

End Sub

```

Private Sub lnkOutbox_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles lnkOutbox.Click
    dsClient_Storage.Clear()
    updateoutbox()
    skrinOutbox.Show()
    skrinMessage.Hide()

```

End Sub

```

Private Sub btnOutboxBack_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnOutboxCancel.Click

    skrinMessage.Show()
    skrinOutbox.Hide()

```

End Sub

```

Private Sub updateoutbox()
    Dim connsms As SqlConnection = New
    SqlConnection("Data Source=localhost;" &
    "Initial Catalog=sms;Integrated Security=SSPI")
    Dim cmdClient_Storage As New SqlCommand()
    Try
        cmdClient_Storage = connsms.CreateCommand
        cmdClient_Storage.CommandText = "SELECT * FROM
        dbo.Client_Storage WHERE User_ID = '" & strusername & "'
        ORDER BY Date_Saved desc "
        daClient_Storage.SelectCommand = cmdClient_Storage

        Dim cb As SqlCommandBuilder = New
        SqlCommandBuilder(daClient_Storage)
        daClient_Storage.Fill(dsClient_Storage, "Client_Storage")
        tbl.DataBindings.Clear()
        tbl.DataBindings.Add("Text", dsClient_Storage.Tables(0),

```



```

        "Date_Saved")
        tb1.Visible = False
        tb2.DataBindings.Clear()
        tb2.DataBindings.Add("Text", dsClient_Storage.Tables(0),
        "Receiver_ID")
        tb2.Visible = False
        tb3.DataBindings.Clear()
        tb3.DataBindings.Add("Text", dsClient_Storage.Tables(0),
        "Message")
        tb3.Visible = False
        txtOutbox.Text = tb1.Text & vbCrLf + tb2.Text & vbCrLf +
        tb3.Text
    Catch ex As Exception
        MsgBox("Error: " & ex.Source & ": " & ex.Message, _
        MsgBoxStyle.OKOnly, "skrinOutbox_Load")
    End Try
End Sub

Private Sub btnOutBack_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnOutBack.Click
    Me.BindingContext(dsClient_Storage.Tables(0)).Position -= 1
    txtOutbox.Text = tb1.Text & vbCrLf + tb2.Text & vbCrLf +
    tb3.Text
End Sub

Private Sub btnOutNext_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnOutNext.Click
    Me.BindingContext(dsClient_Storage.Tables(0)).Position += 1
    txtOutbox.Text = tb1.Text & vbCrLf + tb2.Text & vbCrLf +
    tb3.Text
End Sub

Private Sub btnOutboxDelete_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnOutboxDelete.Click

    skrinConfirmDeleteOutbox.Show()
    skrinOutbox.Hide()

End Sub

Private Sub btnCDOutboxOK_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnCDOutboxOK.Click

    Try
        Me.BindingContext(dsClient_Storage.Tables
        ("Client_Storage")).RemoveAt(Me.BindingContext(
        dsClient_Storage.Tables("Client_Storage")).Position)
        daClient_Storage.Update(dsClient_Storage.Tables("Client_Storage"))
    Catch ex As Exception
        MsgBox("Error: " & ex.Source & ": " & ex.Message, _
        MsgBoxStyle.OKOnly, "btnOutboxDelete_Click")
    End Try
    skrinConfirmDeleteOutbox.Hide()
    skrinDeleteOutbox.Show()
End Sub

Private Sub btnCDOutboxCancel_Click(ByVal sender As
System.Object, ByVal e As System.EventArgs) Handles
btnCDOutboxCancel.Click

    skrinOutbox.Show()

```



```

        skrinConfirmDeleteOutbox.Hide()

End Sub

Private Sub btnDOOK_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles btnDOOK.Click
    dsClient_Storage.Clear()
    updateoutbox()
    skrinOutbox.Show()
    skrinDeleteOutbox.Hide()

End Sub

Private Sub mnueditprofile_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles mnuEditProfile.Click
    Dim editprof As New frmEditProfile()
    editprof.Show()

End Sub

Private Sub linkBalance_LinkClicked(ByVal sender As
System.Object, ByVal e As
System.Windows.Forms.LinkLabelLinkClickedEventArgs) Handles
linkBalance.LinkClicked
    skrinMenu.Hide()
    skrinDirectory.Show()

End Sub

Private Sub btnBalanceOK_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnBalanceOK.Click
    skrinBalance.Hide()
    skrinWelcome.Show()

End Sub

Private Sub mnuAbout_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles mnuAbout.Click
    Dim about As New frmAbout()
    about.Show()

End Sub

Private Sub linkCResult_LinkClicked(ByVal sender As
System.Object, ByVal e As
System.Windows.Forms.LinkLabelLinkClickedEventArgs) Handles
linkCResult.LinkClicked

    skrinCResult.Show()
    skrinMenu.Hide()

End Sub

Private Sub mnuGray_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles mnuGray.Click
    Me.BackgroundImage = Me.pbGray.Image
    pbGray.Visible = True
    pbDimGray.Visible = False
    pbSilver.Visible = False
    pbCoralRed.Visible = False
    pbWhite.Visible = False
    pbAntiqueWhite.Visible = False
    pbBlue.Visible = False

```



```

pbMidnightBlue.Visible = False
pbYellowGreen.Visible = False
pbPurple.Visible = False
mnuGray.Checked = True
mnuDimGray.Checked = False
mnuSilver.Checked = False
mnuCoralRed.Checked = False
mnuWhite.Checked = False
mnuAntiqueWhite.Checked = False
mnuBlue.Checked = False
mnuMidnightBlue.Checked = False
mnuYellowGreen.Checked = False
mnuPurple.Checked = False

```

End Sub

```

Private Sub mnuDimGray_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles mnuDimGray.Click

```

```

    Me.BackgroundImage = Me.pbDimGray.Image
    pbGray.Visible = False
    pbDimGray.Visible = True
    pbSilver.Visible = False
    pbCoralRed.Visible = False
    pbWhite.Visible = False
    pbAntiqueWhite.Visible = False
    pbBlue.Visible = False
    pbMidnightBlue.Visible = False
    pbYellowGreen.Visible = False
    pbPurple.Visible = False
    mnuGray.Checked = False
    mnuDimGray.Checked = True
    mnuSilver.Checked = False
    mnuCoralRed.Checked = False
    mnuWhite.Checked = False
    mnuAntiqueWhite.Checked = False
    mnuBlue.Checked = False
    mnuMidnightBlue.Checked = False
    mnuYellowGreen.Checked = False
    mnuPurple.Checked = False

```

End Sub

```

Private Sub mnuSilver_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles mnuSilver.Click

```

```

    Me.BackgroundImage = Me.pbSilver.Image
    pbGray.Visible = False
    pbDimGray.Visible = False
    pbSilver.Visible = True
    pbCoralRed.Visible = False
    pbWhite.Visible = False
    pbAntiqueWhite.Visible = False
    pbBlue.Visible = False
    pbMidnightBlue.Visible = False
    pbYellowGreen.Visible = False
    pbPurple.Visible = False
    mnuGray.Checked = False
    mnuDimGray.Checked = False
    mnuSilver.Checked = True
    mnuCoralRed.Checked = False
    mnuWhite.Checked = False
    mnuAntiqueWhite.Checked = False
    mnuBlue.Checked = False
    mnuMidnightBlue.Checked = False

```



```

mnuYellowGreen.Checked = False
mnuPurple.Checked = False
End Sub

```

```

Private Sub mnuCoralRed_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles mnuCoralRed.Click
    Me.BackgroundImage = Me.pbCoralRed.Image
    pbGray.Visible = False
    pbDimGray.Visible = False
    pbSilver.Visible = False
    pbCoralRed.Visible = True
    pbWhite.Visible = False
    pbAntiqueWhite.Visible = False
    pbBlue.Visible = False
    pbMidnightBlue.Visible = False
    pbYellowGreen.Visible = False
    pbPurple.Visible = False
    mnuGray.Checked = False
    mnuDimGray.Checked = False
    mnuSilver.Checked = False
    mnuCoralRed.Checked = True
    mnuWhite.Checked = False
    mnuAntiqueWhite.Checked = False
    mnuBlue.Checked = False
    mnuMidnightBlue.Checked = False
    mnuYellowGreen.Checked = False
    mnuPurple.Checked = False
End Sub

```

```

Private Sub mnuWhite_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles mnuWhite.Click
    Me.BackgroundImage = Me.pbWhite.Image
    pbGray.Visible = False
    pbDimGray.Visible = False
    pbSilver.Visible = False
    pbCoralRed.Visible = False
    pbWhite.Visible = True
    pbAntiqueWhite.Visible = False
    pbBlue.Visible = False
    pbMidnightBlue.Visible = False
    pbYellowGreen.Visible = False
    pbPurple.Visible = False
    mnuGray.Checked = False
    mnuDimGray.Checked = False
    mnuSilver.Checked = False
    mnuCoralRed.Checked = False
    mnuWhite.Checked = True
    mnuAntiqueWhite.Checked = False
    mnuBlue.Checked = False
    mnuMidnightBlue.Checked = False
    mnuYellowGreen.Checked = False
    mnuPurple.Checked = False
End Sub

```

```

Private Sub mnuAntiqueWhite_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles mnuAntiqueWhite.Click
    Me.BackgroundImage = Me.pbAntiqueWhite.Image
    pbGray.Visible = False
    pbDimGray.Visible = False
    pbSilver.Visible = False
    pbCoralRed.Visible = False

```



```

pbWhite.Visible = False
pbAntiqueWhite.Visible = True
pbBlue.Visible = False
pbMidnightBlue.Visible = False
pbYellowGreen.Visible = False
pbPurple.Visible = False
mnuGray.Checked = False
mnuDimGray.Checked = False
mnuSilver.Checked = False
mnuCoralRed.Checked = False
mnuWhite.Checked = False
mnuAntiqueWhite.Checked = True
mnuBlue.Checked = False
mnuMidnightBlue.Checked = False
mnuYellowGreen.Checked = False
mnuPurple.Checked = False

```

End Sub

```

Private Sub mnuBlue_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles mnuBlue.Click

```

```

Me.BackgroundImage = Me.pbBlue.Image
pbGray.Visible = False
pbDimGray.Visible = False
pbSilver.Visible = False
pbCoralRed.Visible = False
pbWhite.Visible = False
pbAntiqueWhite.Visible = False
pbBlue.Visible = True
pbMidnightBlue.Visible = False
pbYellowGreen.Visible = False
pbPurple.Visible = False
mnuGray.Checked = False
mnuDimGray.Checked = False
mnuSilver.Checked = False
mnuCoralRed.Checked = False
mnuWhite.Checked = False
mnuAntiqueWhite.Checked = False
mnuBlue.Checked = True
mnuMidnightBlue.Checked = False
mnuYellowGreen.Checked = False
mnuPurple.Checked = False

```

End Sub

```

Private Sub mnuMidnightBlue_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles mnuMidnightBlue.Click

```

```

Me.BackgroundImage = Me.pbMidnightBlue.Image
pbGray.Visible = False
pbDimGray.Visible = False
pbSilver.Visible = False
pbCoralRed.Visible = False
pbWhite.Visible = False
pbAntiqueWhite.Visible = False
pbBlue.Visible = False
pbMidnightBlue.Visible = True
pbYellowGreen.Visible = False
pbPurple.Visible = False
mnuGray.Checked = False
mnuDimGray.Checked = False
mnuSilver.Checked = False
mnuCoralRed.Checked = False
mnuWhite.Checked = False

```



```

mnuAntiqueWhite.Checked = False
mnuBlue.Checked = False
mnuMidnightBlue.Checked = True
mnuYellowGreen.Checked = False
mnuPurple.Checked = False
End Sub

```

```

Private Sub mnuYellowGreen_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles mnuYellowGreen.Click
    Me.BackgroundImage = Me.pbYellowGreen.Image
    pbGray.Visible = False
    pbDimGray.Visible = False
    pbSilver.Visible = False
    pbCoralRed.Visible = False
    pbWhite.Visible = False
    pbAntiqueWhite.Visible = False
    pbBlue.Visible = False
    pbMidnightBlue.Visible = False
    pbYellowGreen.Visible = True
    pbPurple.Visible = False
    mnuGray.Checked = False
    mnuDimGray.Checked = False
    mnuSilver.Checked = False
    mnuCoralRed.Checked = False
    mnuWhite.Checked = False
    mnuAntiqueWhite.Checked = False
    mnuBlue.Checked = False
    mnuMidnightBlue.Checked = False
    mnuYellowGreen.Checked = True
    mnuPurple.Checked = False
End Sub

```

```

Private Sub mnuPurple_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles mnuPurple.Click
    Me.BackgroundImage = Me.pbPurple.Image
    pbGray.Visible = False
    pbDimGray.Visible = False
    pbSilver.Visible = False
    pbCoralRed.Visible = False
    pbWhite.Visible = False
    pbAntiqueWhite.Visible = False
    pbBlue.Visible = False
    pbMidnightBlue.Visible = False
    pbYellowGreen.Visible = False
    pbPurple.Visible = True
    mnuGray.Checked = False
    mnuDimGray.Checked = False
    mnuSilver.Checked = False
    mnuCoralRed.Checked = False
    mnuWhite.Checked = False
    mnuAntiqueWhite.Checked = False
    mnuBlue.Checked = False
    mnuMidnightBlue.Checked = False
    mnuYellowGreen.Checked = False
    mnuPurple.Checked = True
End Sub
End Class

```


Server Side

```
Imports System.Net.Sockets
Imports System.Data.SqlClient
```

```
Public Class frmMain
    Inherits System.Windows.Forms.Form
```

```
Const PORT_NUM As Integer = 8088
```

```
Private clients As New Hashtable()
Private listener As TcpListener
Private listenerThread As Threading.Thread
Dim credit, new_balance As String
```

```
Private Sub ConnectUser(ByVal userName As String, ByVal
userPword As String, ByVal sender As UserConnection)
```

```
    Dim con As SqlConnection
    Dim sql, id As String
    Dim cmd As SqlCommand
    sender.Name = userName
    sender.Pword = userPword
```

```
    con = New SqlConnection("data source= localhost; initial
catalog=sms; integrated security=SSPI;")
    sql = "SELECT * FROM User_Profile WHERE User_ID='{0}' and
Password='{1}'"
    sql = String.Format(sql, userName, userPword)
    cmd = New SqlCommand(sql, con)
    con.Open()
    Try
        id = CType(cmd.ExecuteScalar(), String)
    Finally
        con.Close()
    End Try
    If Not id Is Nothing Then
        UpdateStatus(userName & " has sign in.")
        clients.Add(userName, sender)
        SendToClients("SEND|" & sender.Name & " has sign in.",
sender)
    Else
        ReplyToSender("REFUSE", sender)
    End If
```

```
End Sub
```

```
Private Sub DisconnectUser(ByVal sender As UserConnection)
    UpdateStatus(sender.Name & " has sign out.")
    SendToClients("SEND|" & sender.Name & " has sign out.",
sender)
    clients.Remove(sender.Name)
End Sub
```

```
' This subroutine is used as a background listener thread to
allow reading incoming
' messages without lagging the user interface.
Private Sub DoListen()
    Try
```



```

' Listen for new connections.
listener = New TcpListener(PORT_NUM)
listener.Start()
Do
    ' Create a new user connection using TcpClient
    returned by
    ' TcpListener.AcceptTcpClient()
    Dim client As New
    UserConnection(listener.AcceptTcpClient)

    ' Create an event handler to allow the
    UserConnection to communicate
    ' with the window.
    AddHandler client.LineReceived, AddressOf
    OnLineReceived
    UpdateStatus("New connection found: waiting for log-
    in")
    Loop Until False
Catch
End Try
End Sub

' When the window closes, stop the listener.
Private Sub frmMain_Closing(ByVal sender As Object, ByVal e As
System.ComponentModel.CancelEventArgs) Handles MyBase.Closing
    listener.Stop()
End Sub

' Start the background listener thread.
Private Sub frmMain_Load(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles MyBase.Load
    listenerThread = New Threading.Thread(AddressOf DoListen)
    listenerThread.Start()
    UpdateStatus("Listener started")
End Sub

Private Sub OnLineReceived(ByVal sender As UserConnection, ByVal
data As String)
    Dim dataArray() As String

    ' Message parts are divided by "|" Break the string into an
    array accordingly.
    dataArray = data.Split(Chr(124))

    ' dataArray(0) is the command.
    Select Case dataArray(0)
        Case "CONNECT"
            ConnectUser(dataArray(1), dataArray(2), sender)
        Case "SEND"
            SendMessage(dataArray(1), dataArray(2), sender)
        Case "DISCONNECT"
            DisconnectUser(sender)
        Case "REQUESTRESULTS"
            CResult(dataArray(1), dataArray(2), sender)
        Case Else
            UpdateStatus("Unknown message:" & data)
    End Select
End Sub

' This subroutine sends a response to the sender.
Private Sub ReplyToSender(ByVal strMessage As String, ByVal

```



```

sender As UserConnection)
    sender.SendData(strMessage)
End Sub

Private Sub balance(ByVal sender As UserConnection)

    'Get value of credit
    Try
        Dim con As SqlConnection
        Dim cmd As SqlCommand
        Dim dr As SqlDataReader

        con = New SqlConnection("data source= localhost;
initial catalog=sms; integrated security=SSPI;")
        con.Open()
        cmd = con.CreateCommand
        cmd.CommandText = " SELECT Account_Balance FROM Balance
WHERE User_ID = '" & sender.Name & "' "
        dr = cmd.ExecuteReader
        While dr.Read()
            credit = dr.GetValue(0)
        End While
        dr.Close()
        con.Close()
        UpdateStatus("Current credit for " & sender.Name & " is
" & credit & " before sending message.")
    Catch exc As Exception
        MsgBox("Check credit Error", MsgBoxStyle.Exclamation,
Me.Text)
    End Try
End Sub

Private Sub SendMessage(ByVal message As String, ByVal directory
As String, ByVal sender As UserConnection)
    balance(sender)

    If directory = "cb123" Then

        Dim strSelectServer_Storage As String = "SELECT * FROM
Server_Storage ORDER BY Date_Send"
        Dim strConnString As String = "server=localhost;
database=sms;integrated security=SSPI;"
        Dim connsms As New SqlConnection(strConnString)
        Dim strInsertCommand As String = "INSERT INTO
Server_Storage(User_ID,Receiver_ID,Message,Date_Send) "
        & "VALUES (@User_ID,@Receiver_ID,@Message,@Date_Send) "
        Dim daServer_Storage As New SqlDataAdapter()
        Dim dsServer_Storage As New DataSet()
        Dim cmdSelectServer_Storage As SqlCommand = New
SqlCommand(strSelectServer_Storage, connsms)
        Dim cmdInsertServer_Storage As New
SqlCommand(strInsertCommand, connsms)
        daServer_Storage.SelectCommand = cmdSelectServer_Storage
        daServer_Storage.InsertCommand = cmdInsertServer_Storage

        Try
            connsms.Open()
            daServer_Storage.Fill(dsServer_Storage,
"dtServer_Storageetable")
            cmdInsertServer_Storage.Parameters.Add _
(New SqlParameter _

```



```

        ("@User_ID", SqlDbType.VarChar, 10)).Value = "cb123"
cmdInsertServer_Storage.Parameters.Add _
(New SqlParameter
("@Receiver_ID", SqlDbType.VarChar, 10)).Value =
sender.Name
cmdInsertServer_Storage.Parameters.Add _
(New SqlParameter
("@Message", SqlDbType.VarChar, 160)).Value = "Your
current credit balance is " & credit
cmdInsertServer_Storage.Parameters.Add _
(New SqlParameter
("@Date_Send", SqlDbType.DateTime, 8)).Value =
System.DateTime.Now
cmdInsertServer_Storage.ExecuteNonQuery()
connsms.Close()
UpdateStatus("Request checking balance by " &
sender.Name)
UpdateStatus("Sending credit balance = " & credit & "
back to " & sender.Name)
Catch ex As Exception
MsgBox("Reply Check balance Error: ")
End Try

ElseIf directory = "tu123" Then
Dim con As SqlConnection
Dim sql, id As String
Dim cmd As SqlCommand

con = New SqlConnection("data source= localhost;
initial catalog=sms; integrated security=SSPI;")
sql = "SELECT * FROM Card_ID WHERE Pin_No='{0}'"
sql = String.Format(sql, message)
cmd = New SqlCommand(sql, con)
con.Open()
Try
    id = CType(cmd.ExecuteScalar(), String)
Finally
    con.Close()
End Try
If Not id Is Nothing Then
Dim mda As New SqlDataAdapter()
Dim mds As New DataSet()
Dim dr As DataRow
Dim cnn As SqlClient.SqlConnection = New
SqlClient.SqlConnection("Data Source= localhost;"
& "Initial Catalog=sms;Integrated Security=SSPI")
Dim cmdSelect As New SqlClient.SqlCommand()
Dim cmdUpdate As New SqlClient.SqlCommand()
Dim cmdDelete As New SqlClient.SqlCommand()
Dim prm As SqlClient.SqlParameter

Try
    ' Create the select command to grab the initial
    data
    cmdSelect = cnn.CreateCommand
    cmdSelect.CommandText = "SELECT User_ID,
Account_Balance FROM Balance"
    mda.SelectCommand = cmdSelect

    ' The update command handles updates to existing
    rows

```



```

cmdUpdate = cnn.CreateCommand
cmdUpdate.CommandText = "UPDATE Balance " & _
"SET Account_Balance = @Account_Balance " & _
"WHERE User_ID = @User_ID"
' Now create the parameters that will be passed
to this command
prm =
cmdUpdate.Parameters.Add("@Account_Balance", _
SqlDbType.Float, 8, "Account_Balance")
prm = cmdUpdate.Parameters.Add("@User_ID", _
SqlDbType.VarChar, 10, "User_ID")
prm.SourceVersion = DataRowVersion.Original
mda.UpdateCommand = cmdUpdate

mda.Fill(mds, "Balance")

new_balance = credit + 50

Dim adrEdit() = mds.Tables("Balance").Select( _
"User_ID = '" & sender.Name & "'")

If UBound(adrEdit, 1) > -1 Then
    dr = adrEdit(0)
    dr("Account_Balance") = new_balance
    mda.Update(mds, "Balance")
Else
    MsgBox("Topup failed!")
End If

Catch ex As Exception
    MsgBox("Topup Error: ")
End Try

'code for delete pin no in card_ID table

Dim sConnectionString As String = "Initial
Catalog=sms;Data Source=localhost;integrated
security=sspi;"
Dim objConn As New SqlConnection(sConnectionString)
objConn.Open()
Dim sSQL As String = "DELETE FROM Card_ID WHERE
Pin_No = @Pin_No"
Dim objCmd As New SqlCommand(sSQL, objConn)

objCmd.Parameters.Add("@Pin_No", SqlDbType.Char, 10)
objCmd.Parameters.Item("@Pin_No").Value = message

Try
    objCmd.ExecuteNonQuery()
    Console.WriteLine("Record Deleted")
Catch exc As Exception
    Console.WriteLine(exc.ToString)
End Try
Console.WriteLine("Record Deleted")
Console.ReadLine()

Dim connsms As SqlClient.SqlConnection = New
SqlClient.SqlConnection("Data Source=localhost;"
& "Initial Catalog=sms;Integrated Security=SSPI")

```



```

Dim cmdCard_ID As New SqlCommand()
Dim daCard_ID As SqlDataAdapter
Dim dsCard_ID As DataSet

Try
    cmdCard_ID = connsms.CreateCommand
    cmdCard_ID.CommandText = "SELECT * FROM
    dbo.Card_ID WHERE Pin_No = message "
    daCard_ID.SelectCommand = cmdCard_ID

    Dim cb As SqlCommandBuilder = New
    SqlCommandBuilder(daCard_ID)
    daCard_ID.Fill(dsCard_ID, "Card_ID")
    tbpin.DataBindings.Clear()
    tbpin.DataBindings.Add("Text",
    dsCard_ID.Tables(0), "Pin_No")
    tbpin.Visible = False
    tbcredit.DataBindings.Clear()
    tbcredit.DataBindings.Add("Text",
    dsCard_ID.Tables(0), "Credit")
    tbcredit.Visible = False

    Me.BindingContext(dsCard_ID.Tables
    ("Card_ID")).RemoveAt
    Me.BindingContext(
    dsCard_ID.Tables("Card_ID").Position)
    daCard_ID.Update(dsCard_ID.Tables("Card_ID"))

Catch ex As Exception
    'MsgBox("delete Card_ID Error; ")
End Try

'reply back to client
Dim strSelectServer_Storage As String = "SELECT *
FROM Server_Storage ORDER BY Date_Send"
Dim strConnString As String = "server=localhost;
database=sms;integrated security=SSPI;"
Dim strInsertCommand As String = _
"INSERT INTO
Server_Storage(User_ID,Receiver_ID,Message,Date_Send)
" & "VALUES
(@User_ID,@Receiver_ID,@Message,@Date_Send)"
Dim daServer_Storage As New SqlDataAdapter()
Dim dsServer_Storage As New DataSet()
Dim cmdSelectServer_Storage As SqlCommand = New
SqlCommand(strSelectServer_Storage, connsms)
Dim cmdInsertServer_Storage As New
SqlCommand(strInsertCommand, connsms)
daServer_Storage.SelectCommand =
cmdSelectServer_Storage
daServer_Storage.InsertCommand =
cmdInsertServer_Storage

Try
    connsms.Open()
    daServer_Storage.Fill(dsServer_Storage,
    "dtServer_Storageetable")
    cmdInsertServer_Storage.Parameters.Add _
(New SqlParameter
    ("@User_ID", SqlDbType.VarChar, 10)).Value =

```



```

        "tu123"
        cmdInsertServer_Storage.Parameters.Add _
        (New SqlParameter
        ("@Receiver_ID", SqlDbType.VarChar, 10)).Value =
        sender.Name
        cmdInsertServer_Storage.Parameters.Add _
        (New SqlParameter
        ("@Message", SqlDbType.VarChar, 160)).Value =
        "Topup successful. Your new credit balance is "
        & new_balance
        cmdInsertServer_Storage.Parameters.Add _
        (New SqlParameter
        ("@Date_Send", SqlDbType.DateTime, 8)).Value =
        System.DateTime.Now
        cmdInsertServer_Storage.ExecuteNonQuery()
        connsms.Close()

        UpdateStatus(sender.Name & " has successfully
        topup. Your new credit = " & new_balance)
    Catch ex As Exception
        MsgBox("Reply Check balance Error: ")
    End Try

    'UpdateStatus(sender.Name & " has topup.Current
    credit is " & new_balance)
Else
    'MsgBox("Invalid Topup ID")
    Dim strSelectServer_Storage As String = "SELECT *
    FROM Server_Storage ORDER BY Date_Send"
    Dim strConnString As String = "server=localhost;
    database=sms;integrated security=SSPI;"
    Dim connsms As New SqlConnection(strConnString)
    Dim strInsertCommand As String = _
    "INSERT INTO
    Server_Storage(User_ID,Receiver_ID,Message,Date_Send)
    " & "VALUES
    (@User_ID,@Receiver_ID,@Message,@Date_Send)"
    Dim daServer_Storage As New SqlDataAdapter()
    Dim dsServer_Storage As New DataSet()
    Dim cmdSelectServer_Storage As SqlCommand = New
    SqlCommand(strSelectServer_Storage, connsms)
    Dim cmdInsertServer_Storage As New
    SqlCommand(strInsertCommand, connsms)
    daServer_Storage.SelectCommand =
    cmdSelectServer_Storage
    daServer_Storage.InsertCommand =
    cmdInsertServer_Storage

    Try
        connsms.Open()
        daServer_Storage.Fill(dsServer_Storage,
        "dtServer_Storageetable")
        cmdInsertServer_Storage.Parameters.Add _
        (New SqlParameter
        ("@User_ID", SqlDbType.VarChar, 10)).Value =
        "tu123"
        cmdInsertServer_Storage.Parameters.Add _
        (New SqlParameter
        ("@Receiver_ID", SqlDbType.VarChar, 10)).Value =
        sender.Name
        cmdInsertServer_Storage.Parameters.Add _

```



```

(New SqlParameter
("@Message", SqlDbType.VarChar, 160)).Value =
"Topup failed. Invalid Pin Number."
cmdInsertServer_Storage.Parameters.Add _
(New SqlParameter
("@Date_Send", SqlDbType.DateTime, 8)).Value =
System.DateTime.Now
cmdInsertServer_Storage.ExecuteNonQuery()
connsms.Close()

UpdateStatus("Topup by " & sender.Name & "
failed. Invalid Pin No.")
UpdateStatus("Message sent back to " &
sender.Name)

Catch ex As Exception
'MsgBox("Reply Check balance Error: ")
End Try
End If

```

```

ElseIf directory <> "" AndAlso credit > 3 Then
Dim strSelectServer_Storage As String = "SELECT * FROM
Server_Storage ORDER BY Date_Send"
Dim strConnString As String = "server= localhost;
database=sms;integrated security=SSPI;"
Dim connsms As New SqlConnection(strConnString)
Dim strInsertCommand As String = _
"INSERT INTO
Server_Storage (User_ID,Receiver_ID,Message,Date_Send) "
& "VALUES (@User_ID,@Receiver_ID,@Message,@Date_Send)"
Dim daServer_Storage As New SqlDataAdapter()
Dim dsServer_Storage As New DataSet()
Dim cmdSelectServer_Storage As SqlCommand = New
SqlCommand (strSelectServer_Storage, connsms)
Dim cmdInsertServer_Storage As New
SqlCommand(strInsertCommand, connsms)
daServer_Storage.SelectCommand = cmdSelectServer_Storage
daServer_Storage.InsertCommand = cmdInsertServer_Storage

Try
connsms.Open()
daServer_Storage.Fill(dsServer_Storage,
"dtServer_Storagetable")
cmdInsertServer_Storage.Parameters.Add _
(New SqlParameter
("@User_ID", SqlDbType.VarChar, 10)).Value =
sender.Name
cmdInsertServer_Storage.Parameters.Add _
(New SqlParameter
("@Receiver_ID", SqlDbType.VarChar, 10)).Value =
directory
cmdInsertServer_Storage.Parameters.Add _
(New SqlParameter
("@Message", SqlDbType.VarChar, 160)).Value =
message
cmdInsertServer_Storage.Parameters.Add _
(New SqlParameter
("@Date_Send", SqlDbType.DateTime, 8)).Value =
System.DateTime.Now
cmdInsertServer_Storage.ExecuteNonQuery()

```



```

connsms.Close()

UpdateStatus(sender.Name & " send message to " &
directory)
UpdateStatus("Message content : " & message)
UpdateStatus("Now your new credit balance = " &
credit - 3)

' SendToClients("SEND|" & sender.Name & ": " &
message, sender)

'UPDATE credit
Dim mda As New SqlDataAdapter()
Dim mds As New DataSet()
Dim dr As DataRow
Dim cnn As SqlConnection = New
SqlConnection("Data Source= localhost;"
& "Initial Catalog=sms;Integrated Security=SSPI")
Dim cmdSelect As New SqlCommand()
Dim cmdUpdate As New SqlCommand()
Dim prm As SqlParameter

Try
    ' Create the select command to grab the initial
    data
    cmdSelect = cnn.CreateCommand
    cmdSelect.CommandText = "SELECT User_ID,
Account_Balance FROM Balance"
    mda.SelectCommand = cmdSelect

    ' The update command handles updates to existing
    rows
    cmdUpdate = cnn.CreateCommand
    cmdUpdate.CommandText = "UPDATE Balance " & _
"SET Account_Balance = @Account_Balance " & _
"WHERE User_ID = @User_ID"

    ' Now create the parameters that will be
    ' passed to this command
    prm =cmdUpdate.Parameters.Add("@Account_Balance",
SqlDbType.Float, 8, "Account_Balance")
    prm = cmdUpdate.Parameters.Add("@User_ID", _
SqlDbType.VarChar, 10, "User_ID")
    prm.SourceVersion = DataRowVersion.Original
    mda.UpdateCommand = cmdUpdate

    mda.Fill(mds, "Balance")

    new_balance = credit - 3

    Dim adrEdit() = mds.Tables("Balance").Select( _
"User_ID = '" & sender.Name & "'")

    If UBound(adrEdit, 1) > -1 Then
        dr = adrEdit(0)
        dr("Account_Balance") = new_balance
        mda.Update(mds, "Balance")
    Else
        MsgBox(sender.Name & " not found!")
    End If

```



```

        Catch ex As Exception
            MsgBox("Minus credit Error: ")
        End Try

    Catch exc As Exception
        MsgBox("Save Message Error", MsgBoxStyle.Exclamation,
            Me.Text)
    End Try

Else
    UpdateStatus("Low credit. Message not sent")

    Dim strSelectServer_Storage As String = "SELECT * FROM
    Server_Storage ORDER BY Date_Send"
    Dim strConnString As String = "server= localhost;
    database=sms;integrated security=SSPI;"
    Dim connsms As New SqlConnection(strConnString)
    Dim strInsertCommand As String = _
    "INSERT INTO
    Server_Storage(User_ID,Receiver_ID,Message,Date_Send) "
    & "VALUES (@User_ID,@Receiver_ID,@Message,@Date_Send)"
    Dim daServer_Storage As New SqlDataAdapter()
    Dim dsServer_Storage As New DataSet()
    Dim cmdSelectServer_Storage As SqlCommand = New
    SqlCommand(strSelectServer_Storage, connsms)
    Dim cmdInsertServer_Storage As New
    SqlCommand(strInsertCommand, connsms)
    daServer_Storage.SelectCommand = cmdSelectServer_Storage
    daServer_Storage.InsertCommand = cmdInsertServer_Storage

    connsms.Open()
    daServer_Storage.Fill(dsServer_Storage,
    "dtServer_Storageetable")
    cmdInsertServer_Storage.Parameters.Add _
    (New SqlParameter
    ("@User_ID", SqlDbType.VarChar, 10)).Value = "SMRCS"
    cmdInsertServer_Storage.Parameters.Add _
    (New SqlParameter
    ("@Receiver_ID", SqlDbType.VarChar, 10)).Value =
    sender.Name
    cmdInsertServer_Storage.Parameters.Add _
    (New SqlParameter
    ("@Message", SqlDbType.VarChar, 160)).Value = "Low
    credit. Message not sent."
    cmdInsertServer_Storage.Parameters.Add _
    (New SqlParameter
    ("@Date_Send", SqlDbType.DateTime, 8)).Value =
    System.DateTime.Now
    cmdInsertServer_Storage.ExecuteNonQuery()
    connsms.Close()
End If

End Sub

Private Sub CResult(ByVal matric As String, ByVal ic As String,
    ByVal sender As UserConnection)

    Dim credit, new_balance As String
    Try
        Dim conn As SqlConnection

```



```

Dim scmd As SqlCommand
Dim dr As SqlDataReader

conn = New SqlConnection("data source= localhost;
initial catalog=sms; integrated security=SSPI;")
conn.Open()
scmd = conn.CreateCommand
scmd.CommandText = " SELECT Account_Balance FROM Balance
WHERE User_ID = '" & sender.Name & "' "
dr = scmd.ExecuteReader
While dr.Read()
    credit = dr.GetValue(0)
End While
dr.Close()
conn.Close()
UpdateStatus("Current credit for " & sender.Name & " is
" & credit & " before sending message.")
Catch exc As Exception
    MsgBox("Check credit Error", MsgBoxStyle.Exclamation,
    Me.Text)
End Try

Dim CGPA, GPA As String

Try
    Dim cnn As New SqlConnection()
    Dim cd As New SqlCommand()
    Dim dr As SqlDataReader
    Dim sq As String
    cnn.ConnectionString = "Data Source= localhost;
Initial Catalog=sms;Integrated Security=SSPI"
    cnn.Open()
    cd = cnn.CreateCommand
    cd.CommandText = "SELECT Result_CGPA,Result_GPA FROM
Result WHERE Matric_No ='" & matric & "' "
    dr = cd.ExecuteReader
    While dr.Read()
        CGPA = dr.GetValue(0)
        GPA = dr.GetValue(1)
    End While
    dr.Close()
    cnn.Close()
Catch ex As Exception
    MsgBox("CGPA Error:")
End Try

UpdateStatus("Request checking result by " & sender.Name & "
with : " & matric & " and " & ic)
If credit > 6 Then
    Dim con As SqlConnection
    Dim sql, id As String
    Dim cmd As SqlCommand

    con = New SqlConnection("data source= localhost;
initial catalog=sms; integrated security=SSPI;")
    sql = "SELECT * FROM Result WHERE Matric_No='{0}' and
IC_No='{1}'"
    sql = String.Format(sql, matric, ic)
    cmd = New SqlCommand(sql, con)
    con.Open()
    Try

```

```

        id = CType(cmd.ExecuteScalar(), String)
    Finally
        con.Close()
    End Try

    If Not id Is Nothing Then

        UpdateStatus("Matric no and ic for " & sender.Name &
            " is matched.")
        UpdateStatus("Result is sent back to " & sender.Name
            & " with " & CGPA & " and " & GPA)
        UpdateStatus("Now your new credit balance = " &
            credit - 6)
        SendToClients("SEND|" & sender.Name & ": " & matric,
            sender)

        Dim strSelectServer_Storage As String = "SELECT *
            FROM Server_Storage ORDER BY Date_Send"
        Dim strConnString As String = "server=localhost;
            database=sms;integrated security=SSPI;"
        Dim connsms As New SqlConnection(strConnString)
        Dim strInsertCommand As String = _
            "INSERT INTO
            Server_Storage(User_ID,Receiver_ID,Message,Date_Send)
            " & "VALUES
            (@User_ID,@Receiver_ID,@Message,@Date_Send)"
        Dim daServer_Storage As New SqlDataAdapter()
        Dim dsServer_Storage As New DataSet()
        Dim cmdSelectServer_Storage As SqlCommand = New
            SqlCommand(strSelectServer_Storage, connsms)
        Dim cmdInsertServer_Storage As New
            SqlCommand(strInsertCommand, connsms)
        daServer_Storage.SelectCommand =
            cmdSelectServer_Storage
        daServer_Storage.InsertCommand =
            cmdInsertServer_Storage

    Try
        connsms.Open()
        daServer_Storage.Fill(dsServer_Storage,
            "dtServer_Storagetable")
        cmdInsertServer_Storage.Parameters.Add _
            (New SqlParameter
            ("@User_ID", SqlDbType.VarChar, 10)).Value =
            "cr123"
        cmdInsertServer_Storage.Parameters.Add _
            (New SqlParameter
            ("@Receiver_ID", SqlDbType.VarChar, 10)).Value =
            sender.Name
        cmdInsertServer_Storage.Parameters.Add _
            (New SqlParameter
            ("@Message", SqlDbType.VarChar, 160)).Value =
            "CGPA = " & CGPA & " and GPA = " & GPA
        cmdInsertServer_Storage.Parameters.Add _
            (New SqlParameter
            ("@Date_Send", SqlDbType.DateTime, 8)).Value =
            System.DateTime.Now
        cmdInsertServer_Storage.ExecuteNonQuery()
        connsms.Close()

        Dim mda As New SqlDataAdapter()

```



```

Dim mds As New DataSet()
Dim dr As DataRow
Dim cnn As SqlConnection = New
SqlConnection("Data
Source= localhost;" & "Initial
Catalog=sms;Integrated Security=SSPI")
Dim cmdSelect As New SqlCommand()
Dim cmdUpdate As New SqlCommand()
Dim prm As SqlParameter

Try
    ' Create the select command to grab the
    initial data
    cmdSelect = cnn.CreateCommand
    cmdSelect.CommandText = "SELECT User_ID,
Account_Balance FROM Balance"
    mda.SelectCommand = cmdSelect

    ' The update command handles updates to
    existing rows
    cmdUpdate = cnn.CreateCommand
    cmdUpdate.CommandText = "UPDATE Balance " &
"SET Account_Balance = @Account_Balance " &
"WHERE User_ID = @User_ID"

    ' Now create the parameters that will be
    passed to this command
    prm =
cmdUpdate.Parameters.Add("@Account_Balance",
SqlDbType.Float, 8, "Account_Balance")
    prm = cmdUpdate.Parameters.Add("@User_ID", _
SqlDbType.VarChar, 10, "User_ID")
    prm.SourceVersion = DataRowVersion.Original
    mda.UpdateCommand = cmdUpdate
    mda.Fill(mds, "Balance")

    new_balance = credit - 6

    Dim adrEdit() =
mds.Tables("Balance").Select(
"User_ID = '" & sender.Name & "'" )
    If UBound(adrEdit, 1) > -1 Then
        dr = adrEdit(0)
        dr("Account_Balance") = new_balance
        mda.Update(mds, "Balance")
    Else
        MsgBox(sender.Name & " not found!")
    End If

Catch ex As Exception
    MsgBox("Minus credit Error:(under checking
result) ")
End Try

Catch exc As Exception
    MsgBox(" Save Result Error ",
MsgBoxStyle.Exclamation, Me.Text)
End Try
Else
    UpdateStatus("Matric no and ic no for " &
sender.Name & " NOT matched.")

```



```

UpdateStatus("Message sent back to " & sender.Name)
Dim strSelectServer_Storage As String = "SELECT *
FROM Server_Storage ORDER BY Date_Send"
Dim strConnString As String = "server= localhost;
database=sms;integrated security=SSPI;"
Dim connsms As New SqlConnection(strConnString)
Dim strInsertCommand As String = _
"INSERT INTO
Server_Storage(User_ID,Receiver_ID,Message,Date_Send)
" & "VALUES
(@User_ID,@Receiver_ID,@Message,@Date_Send)"
Dim daServer_Storage As New SqlDataAdapter()
Dim dsServer_Storage As New DataSet()
Dim cmdSelectServer_Storage As SqlCommand = New
SqlCommand(strSelectServer_Storage, connsms)
Dim cmdInsertServer_Storage As New
SqlCommand(strInsertCommand, connsms)
daServer_Storage.SelectCommand =
cmdSelectServer_Storage
daServer_Storage.InsertCommand =
cmdInsertServer_Storage
connsms.Open()
daServer_Storage.Fill(dsServer_Storage,
"dtServer_Storageetable")
cmdInsertServer_Storage.Parameters.Add _
(New SqlParameter _
("@User_ID", SqlDbType.VarChar, 10)).Value = "SMRCS"
cmdInsertServer_Storage.Parameters.Add _
(New SqlParameter _
("@Receiver_ID", SqlDbType.VarChar, 10)).Value =
sender.Name
cmdInsertServer_Storage.Parameters.Add _
(New SqlParameter _
("@Message", SqlDbType.VarChar, 160)).Value =
"Matrics and IC no NOT match. Checking result
failed."
cmdInsertServer_Storage.Parameters.Add _
(New SqlParameter _
("@Date_Send", SqlDbType.DateTime, 8)).Value =
System.DateTime.Now
cmdInsertServer_Storage.ExecuteNonQuery()
connsms.Close()
End If
Else
UpdateStatus("Low credit. Message for checking result
not sent")

End If
End Sub

' This subroutine sends a message to all attached clients except
the sender.
Private Sub SendToClients(ByVal strMessage As String, ByVal
sender As UserConnection)
Dim client As UserConnection
Dim entry As DictionaryEntry

' All entries in the clients Hashtable are UserConnection so
it is possible
' to assign it safely.
For Each entry In clients

```



```

        client = CType(entry.Value, UserConnection)

        ' Exclude the sender.
        If client.Name <> sender.Name Then
            client.SendData(strMessage)
        End If
    Next
End Sub

' This subroutine adds line to the Status listbox
Private Sub UpdateStatus(ByVal statusMessage As String)
    lstStatus.Items.Add(statusMessage)
End Sub

End Class

```

Universiti Malaya

Administrator Module

Source Code For Admin Login

The following shows the coding for frmLogin.vb.

```
Imports System.Threading
Imports System.Security.Principal
Imports System.Data.SqlClient

Public Class frmLogin
    Inherits System.Windows.Forms.Form
    Private Sub btnsignin_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles btnsignin.Click
        Dim con As SqlConnection
        Dim sql As String
        Dim cmd As SqlCommand
        Dim id As String
        con = New SqlConnection("data source=localhost; initial
catalog=sms; integrated security=SSPI;")

        sql = "SELECT * FROM Admin_Profile WHERE Admin_ID='{0}' and
Admin_Password='{1}'"

        sql = String.Format(sql, txtAdminID.Text, txtPassword.Text)
        cmd = New SqlCommand(sql, con)
        con.Open()
        Try
            id = CType(cmd.ExecuteScalar(), String)
        Finally
            con.Close()
        End Try
        If Not id Is Nothing Then

            Dim adminprofile As New frmAdminProfile()
            Me.Hide()
            adminprofile.Show()
        Else
            MsgBox("Invalid userID or password")
        End If

    End Sub
End Class
```


Source Code For User Administration

The following shows the coding for frmUsersAdministration.vb. This source codes are quite similar to frmAdminProfile.vb.

```
Public Class frmUsersAdministration
    Inherits System.Windows.Forms.Form
    Dim dsUser_Profile As New DataSet()

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles MyBase.Load

        'declare a new instance of the common class
        Dim cCommon As New clsCommon()
        'call the PopulateDataSet function in the common class and
        pass it to the SQL statement to retrieve the records from
        the database
        dsUser_Profile = cCommon.PopulateDataSet("SELECT *
        "FROM dbo.User_Profile")

        'bind the Text property to the field in the DataTable
        lblUser_IDData.DataBindings.Add("Text",
        dsUser_Profile.Tables(0), "User_ID")

        txtFirst_name.DataBindings.Add("Text",
        dsUser_Profile.Tables(0), "First_Name")

        txtLast_name.DataBindings.Add("Text",
        dsUser_Profile.Tables(0), "Last_Name")

        txtDateofBirth.DataBindings.Add("Text",
        dsUser_Profile.Tables(0), "DateofBirth")

        txtSex.DataBindings.Add("Text",
        dsUser_Profile.Tables(0), "Sex")

        txtCountry.DataBindings.Add("Text",
        dsUser_Profile.Tables(0), "Country")

        txtEmail_Address.DataBindings.Add("Text",
        dsUser_Profile.Tables(0), "Email_Address")

    End Sub

    Private Sub btnMovePrevious_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnMovePrevious.Click
        Me.BindingContext(dsUser_Profile.Tables(0)).Position -= 1
    End Sub

    Private Sub btnMoveNext_Click(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles btnMoveNext.Click
        Me.BindingContext(dsUser_Profile.Tables(0)).Position += 1
    End Sub
End Class
```


Source Code For Student Result Administration

The following shows the coding for frmStudentResult.vb. This source codes are quite similar to frmServerStorage.vb, frmReloarCard.vb and frmCreditList.vb.

```
Public Class frmStudentsResult
    Inherits System.Windows.Forms.Form

    Dim mda As New SqlClient.SqlDataAdapter()
    Dim mds As New DataSet()

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

        Dim cnn As SqlClient.SqlConnection = _
New SqlClient.SqlConnection("server=localhost;
database=sms;integrated security=SSPI;")
        Dim cmdSelect As New SqlClient.SqlCommand()

        Try
            ' Create the select command to grab the initial data
            cmdSelect = cnn.CreateCommand
            cmdSelect.CommandText = "SELECT Matric_No, IC_No,
Result_CGPA, Result_GPA " &
"FROM Result ORDER BY Matric_No;"
            mda.SelectCommand = cmdSelect

            ' Automatically generate the Update, Insert,
            ' and Delete commands
            Dim cb As SqlClient.SqlCommandBuilder = _
New SqlClient.SqlCommandBuilder(mda)

            ' Fill the DataSet and display it on the UI
            mda.Fill(mds, "Result")
            dgResults.DataSource = mds
            dgResults.DataMember = "Result"

        Catch ex As Exception
            MsgBox("Error: " & ex.Source & ": " & ex.Message, _
MsgBoxStyle.OKOnly, "frmUpdate_Load")
        End Try

    End Sub

    Private Sub btnUpdate_Click(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles btnUpdate.Click

        Try
            mda.Update(mds, "Result")
        Catch ex As Exception
            MsgBox("Error: " & ex.Source & ": " & ex.Message, _
MsgBoxStyle.OKOnly, "btnSaveUpdates")
        End Try

    End Sub

End Class
```


Reference



Universiti Malaya

Reference

- [Bowen & Coar, 2000] Bowen, R., & Coar, K., et Apache Server. Indiana : Sams, 2000
- [Holloway, 1989] Holloway, Simon (1989) Methodology handbook for information managers. United States of America : Gover Technical.
- [Johnson, Balinger & Scot Johnson, Keith Ballinger, Davis Chapman, Special Chapman, 1997] edition using ASP, Que Corporation, 1997
- [Kendall and Kendall, Kenneth E. Kendall and Julie E. Kendall, "System Analysis and Design", Prentice Hall Inc. Fourth Edition, 1998
- [Meredith & Mantel, Meredith, J.; and Mantel, Samuel Jr. Project Management : A Managerial Approach. John Wiley & Sons, Inc., 1995.
- [Pfleeger, 2001] Pfleeger, S.H. Software Engineering: Theory And Practice. New Jersey Prentise Hall., 2001
- [Robinson, Barbara & Robinson, Barbara & Prior, Mary (1995) System Analysis Prior, 1995] techniques. United States of America: International Thomson Computer Press.
- [Scheined & Perry, Gray P. Scheined & James T. Perry, Electronic Commerce, 2000] Course Technology, 2000
- [Weber, 1999] Ron Weber, Information System Control And Audit. Prentice Hall., 1999

- [1] Fastie, W (1999). Understanding Client/Server Computing.
Retrieved from the World Wide Web :
<http://www.officewizard.com/books/clientserver/ClientServerComputing.htm>
- [2] Retrieved from the World Wide Web :
<http://www.whatis.com>
- [3] Retrieved from the World Wide Web :
http://www.vbip.com/books/1861001800/chapter_1800_01.asp
- [4] Retrieved from the World Wide Web :
<http://msdn.microsoft.com>
- [5] Retrieved from the World Wide Web :
http://Microsoft_Visual_Studio_Net_Professional_is_a_complete_development_tools_suite.htm