

# **Web-Based Students' Information System for Sabaragamuwa University**

**By  
D.A.V.Wijesinghe**

**00/AS/075**

**Dissertation submitted in partial fulfillment of the requirements for  
the degree of Bachelor of Science in Physical Sciences**


**Faculty of Applied Sciences  
Sabaragamuwa University of Sri Lanka  
Buttala**

**March 2004**

## DECLARATION

I hereby declare that the work reported in the project report was exclusively carried out by me, under the supervision of Dr.R.G.N Meegama and Mr.Jayalath Ekanayake. Any part of this project report has not been submitted earlier or concurrently for same or any other degree.

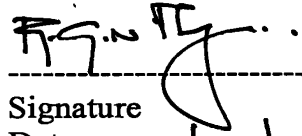
01/05/2004  
Date

  
D.A.V. Wijesinghe

To the best of my knowledge the above particulars are correct.


### External Supervisor

Dr. R.G.N.Meegama.  
Lecturer,  
Centre for computer Studies.  
Sabaragamuwa University of Sri Lanka.  
Belihuloya.

  
Signature  
Date: 06/05/2004

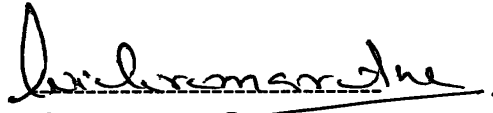
### Internal Supervisor

Mr. Jayalath Ekanayake.  
Lecturer,  
Department of Physical Sciences,  
Faculty of Applied Science,  
Sabaragamuwa University of Sri Lanka.

  
Signature  
Date: 09/05/2004

### Head of the Department

Dr, (Mrs.). N.Wickramaratne.  
Head/Dept Physical Sciences,  
Faculty of Applied Science,  
Sabaragamuwa University of Sri Lanka.

  
Signature  
Date: 10/05/2004

***DEDICATED TO  
MY EVER LOVING PARENTS  
&  
TEACHERS***

## **ACKNOWLEDGEMENTS**

I would like to express my deepest gratitude to my internal supervisor Mr. Jayalath Ekanayake, Lecturer, Department of physical Sciences, Faculty of Applied Sciences. Sabaragamuwa University of Sri Lanka (SUSL). I also extend my thanks to the external supervisor Dr. R.G.N.Meegama, Lecturer and Co-ordinator, Centre for computer studies (CCS), SUSL, Belihuloya who kindly offered me project placement with all the facilities. And my special thanks to the staff for their guidance, persistent and generous help throughout my study.

I also appreciate the patronage of Prof .I.K Perera, Director, CCS.

I express my deepest gratitude to Dr. D.B.M.Wickramaratne, Dean, Faculty of Applied Sciences, and Dr. (Mrs) N.Wickramaratne, Head, Department of Physical Sciences, Faculty of Applied Science, SUSL, for guiding me towards successful completion of the project.

## **ABSTRACT**

The Sabaragamuwa University of Sri Lanka (SUSL) is one of the youngest universities in Sri Lanka, which was established in 1996. The University now has a student population of around 1500, which has been divided into five faculties.

The main administrative premises of the University are located at Belihuloya, and handle all the student information manually. It requires a lot of man-hours. Also, it consumes a considerable amount of time to update and search records.

The main objective of this project was to develop a user-friendly, online student information system for the Sabaragamuwa University. To develop this system the traditional life cycle methodology was used. The system consists of personal information about the students and information with regard to examinations.

The functional requirements were provided by the Center for Computer Studies of the Sabaragamuwa University. The interfaces and the database were designed to meet the given functional requirements. When the interfaces were designed consistency was maintained to provide a friendly environment to the users.

The interfaces were developed using Macromedia Dream weaver MX with HTML. The database was created using Microsoft Access 2000. Then the database and the interfaces were connected using Active Server Pages (ASP) technology. ASP codes were implemented using VB Scripts. The system was developed on a Windows 2000 Professional platform which comes with built-in support for Internet Information Services (IIS).

The outcome of this software project is a user-friendly, online student information system for the Sabaragamuwa University. This system meets all the requirements given by the Center for Computer Studies of the Sabaragamuwa University.

The current system could be further enhanced by including staff details, a search facility and the facility for viewing individual examination results. The bandwidth of the Internet connection and the server performances are the dominant factors, affecting the efficiency of the system.

# CONTENTS

<b>ABSTRACT</b>	<b>I</b>
<b>ACKNOWLEDGEMENTS</b>	<b>II</b>
<b>LIST OF FIGURES</b>	<b>III</b>
<b>ABBREVIATIONS</b>	<b>IV</b>
<b>CONTENTS</b>	<b>V</b>
<b>Chapter 01 – Introduction</b>	
1.1 Introduction	1
1.2 Overview of the project	1
1.2.1 Background	1
1.2.1.1 Sabaragamuwa University of Sri Lanka	1
1.2.2 Objectives	3
1.3 Software development methodologies	4
1.3.1 Linear waterfall model	4
1.3.2 Evolutionary model	4
1.3.3 Spiral model	5
1.3.1.1 How the linear model is related to this project	5
1.3.1.2 Problem with the linear waterfall cycle	6
<b>Chapter 02 – Literature Review</b>	
2.1 Overview of Web-Based Information Systems	7
2.2 Active Server Pages at a glance	8
2.2.1 In the Beginning	8
2.2.2 Development with Scripts	13
2.2.3 Features of ASP	13
2.2.3.1 Language Independence	13
2.2.3.2 Built-In and Installable Objects	14
2.2.3.3 Built-In Objects	14
2.2.3.4 Installable Objects	14
2.2.3.5 Databases and Cookies	15
2.2.3.5.1 Data-Driven Web Pages	15
2.2.3.5.2 Cookies	16
2.2.3.6 Creating Client/Server Web Applications	16
2.2.3.7 Sessions and Web Applications	17

2.2.3.8 Using scripts to extend ASP	18
2.2.4 ASP Syntax	19
2.2.5 Browser Detection	20
2.3 Web Application development with Macromedia Dreamweaver	22
2.3.1 Basics of Web Applications	22
2.3.2 Common uses for web applications	22
2.3.3 Processing regular web pages	23
2.3.4 Processing dynamic pages	24
2.3.5 Accessing a database	25
<b>Chapter 03 – Methodology</b>	
3.1 Approach to the Development of the Student Information System	28
3.1.1 Primary Tools used for the Development	28
3.1.2 Platform used for the Development	28
3.2 Interface Building	28
3.2.1 Steps used to develop the Web Application using Dreamweaver	29
3.2.2 Connecting to the sample database	30
3.3 Database construction	30
3.4 Connecting the Database	31
3.5 Allowing the facilities for updating	32
3.6 Unit Testing	32
3.7 System Testing	32
<b>Chapter 04 – Results &amp; Discussion</b>	<b>33</b>
<b>Chapter 05 – Conclusions and Recommendations</b>	<b>34</b>
<b>REFERENCES</b>	<b>35</b>
<b>Appendix I</b>	
Dreamweaver Workspace	36
<b>Appendix II</b>	
A web page and its underlying codes	36
<b>Appendix III</b>	
The start up screen	37
The Students Personal Details Screen	37
The Students Results Screen	38
The login Fail Screen	38
The Data Entry Selection Screen	39

**Appendix IV**

The Students Personal Data Table 40

The Students Results Table 41

The Logging Data Table 41

**Appendix V**

Codes 42



## **LIST OF FIGURES**

Fig. 2.1 Web-Based Information systems	7
Fig. 2.2 How the Internet is constructed	9
Fig. 2.3 Process of Hypertext Transfer Protocol	10
Fig. 2.4 Process1 Within the web sever	23
Fig. 2.5 Process2 Within the web sever	24
Fig. 2.6 Process3 Within the web sever	26

## **ABBREVIATIONS**

<b>ASP</b>	<b>Active Sever Pages</b>
<b>CCS</b>	<b>Center for computer studies</b>
<b>CGI</b>	<b>Common Gateway Interface</b>
<b>CFML</b>	<b>Cold Fusion Markup Language</b>
<b>DSN</b>	<b>Data Source Name</b>
<b>HTML</b>	<b>Hypertext Markup Language</b>
<b>HTTP</b>	<b>Hypertext Transfer Protocol</b>
<b>IEEE</b>	<b>Institute of Electrical and Electronics Engineers</b>
<b>IE</b>	<b>Internet Explorer</b>
<b>IIS</b>	<b>Internet Information Services</b>
<b>JSP</b>	<b>Java Server Pages</b>
<b>ODBC</b>	<b>Open DataBase Connectivity</b>
<b>RDBMS</b>	<b>Relational Database Management System</b>
<b>SUSL</b>	<b>Sabaragamuwa University of Sri Lanka</b>
<b>SQL</b>	<b>Structured Query language</b>
<b>URL</b>	<b>Uniform Resource Locator</b>
<b>WWW</b>	<b>World Wide Web</b>

# **1. Introduction**

## **1.1 Introduction**

Information is one of the key aspects that is being dealt within any field. Wide ranges of data are exchanged all over the world meeting the multiple requirements of modern world. The evolution of the Internet has contributed massively in catering for the need of exchanging information. Thus, the Internet has become the integral link between countries as well as continents thereby filling the gap that existed previously as a hindrance for the exchange of information.

A University is a system where a great deal of information is handled: they include simple facts and complex facts: examination results and criteria for the award of class degrees. Today, one of the ideal ways to deal with this kind of information is to publish them in the electronic form: that is via a web site. Therefore, as a means of allowing interested parties to have the quickest possible access to the information regarding the university, a web site renders great convenience.

## **1.2 Overview of the project**

### **1.2.1 Background**

#### **1.2.1.1 Sabaragamuwa University of Sri Lanka.**

The Sabaragamuwa University of Sri Lanka (SUSL), being one of the youngest Universities in the country's university system, was established in the 1996. The University now has a student population of around 1500, which has been divided into five faculties; Faculty of Applied Sciences, Faculty of Agricultural Sciences, Faculty of Social Sciences and Languages, Faculty of

Management Studies and Department of Surveying Sciences. It offers the following degree programs through its five Faculties.

- 1 The faculty of social sciences and Languages offers B.A degree program through its two departments: Social Sciences and Languages.
- 2 The faculty of Business Studies offers B.Sc. degree program in Business studies through its two departments, the department of Business Management and the department of Accountancy and Finance.
- 3 The Faculty of Agricultural sciences offer B.Sc. degree program in Agricultural Sciences through its three departments, the department of Livestock production, the department of export Agriculture and the Department of Agribusiness Management.
- 4 The faculty of Applied Sciences offers B.Sc. degree program in food Sciences and Technology, Natural Resources and Physical sciences through its two department, the department of Natural Resources and the Department of Physical sciences.
- 5 The Department of Surveying Sciences offers B.Sc. degree program in Surveying Sciences.

The main administrative premise, is located at Belihuloya, handles all information regarding students. Presently the processing of student information is done manually. This requires a lot of man-hours and is a tedious task. Also, it consumes a considerable time to update the records. Therefore, in order to make this process more efficient, a web-based information management system was proposed.

This system consists of the following information.

1. Personal information about the students.
2. Information with regard to examination results.

The new system has been developed to provide easier access to the above information.

The web site of SUSL (<http://www.sab.ac.lk>) has been designed and maintained by the Center for Computer Studies (CCS) of the University, in order to provide as many details as possible about the campus. The other services currently provided by the CCS are e-mail system for the University and training facilities for students and staff.

### **1.2.2 Objectives**

The primary objective of this project is to develop a user-friendly, web-based student information system for the Sabaragamuwa University. This is achieved by

1. Studying the software development methodologies.
2. Analyzing the effectiveness of the software development methodologies.
3. Developing the system using the most suitable methodology.

The student information has been categorized according to the faculty and the respective departments. Following are the main information accessible through the information system.

1. Personal details: (Registration number, ID number, First name, Last name, Address, Email, Date of Birth, Gender, Faculty, Department, Registration date, Academic year)
2. Examination result categorized by the Year and Semester.

Macromedia Dreamweaver MX has been utilized as the primary software for this project supported by Active Server Pages (ASP) and Structured Query Language (SQL) technologies.

### **1.3 Software development methodologies**

There are several types of methods available to develop a system. IEEE proposed the number of standards for this purpose. Before selecting a method, the developer should really be concerned about some key features of the project such as the time availability for project development, entering releases, clear definition of the problem etc. The following models are frequently used for software development processes (Pressman, 2001).

- Linear model
- Evolutionary model
- Spiral model

#### **1.3.1 Linear model**

The linear or waterfall model is a development process that centers around planned work and is best suited for projects where the requirements can be clearly defined. The linear cycle groups development activities into sequence of consecutive phases. Each phase itself is made of more detailed activities and can only be commenced after the previous phase has been completed. Each phase usually produces one or more models or products in later phases.

#### **1.3.2 Evolutionary model**

Evolutionary model is more applicable when the requirements are not clearly defined or understood or likely to undergo significantly changes. In this process the users evaluate system capabilities (Pressman, 2001). Also, if the search space is too large this model is more suitable. As an example

Microsoft Corporation, frequently uses this model for their software development industry.

### **1.3.3 Spiral model**

This model is most appropriate for system strike requirements in terms of reliability and risk management. This process focuses attention on early error elimination, puts quality objectives up run, integrate development and maintains and provides a framework for simultaneous access. However this process is time-costly overhead and may not be possible on contractual reasons.

#### **1.3.1.1 How the linear model is related to this project**

##### **Phase I - Problem Identification**

Firstly focused on project environment and identify the problem involved in existing system. Output of this phase is principally defined user requirements for a new system. It defines the proposed solution and change processes. It consists project goal, its bounds, and the terms of references for the project.

##### **Phase II - Developing the system specification**

In this stage the major task is to find out more about the system and what users expect from new or changed system.

- Revision to the project goal.
- How the new system will work.
- A broad system specification.
- Define the expected time.

##### **Phase III - System design**

This phase consist of two parts called broad design and detailed design (Ghezzi et al, 1996). The broad design identifies the main architecture of the proposed system. The database and program modules are design during the

detail design phase. Out put of this phase specify the computer programs and data base related new system.

#### **Phase IV - System development**

At the end of this phase a working system is provided for users. This phase includes set of working programs and databases that have been initialized and Interfaces.

#### **1.3.1.2 Problems in linear model**

Linear model has an important property. That is its activities are performed in a strict sequence, one phase must be completed before the next phase starts, and no phase can be repeated.

Common problems are,

- Conceptual design cannot be implemented.
- Information gathered during analysis is not enough.
- Design system cannot be implement.



## 2. Literature Review

### 2.1 Overview of Web-Based Information Systems

In general, web-based applications all share a similar architecture that can be depicted in figure 2.1.

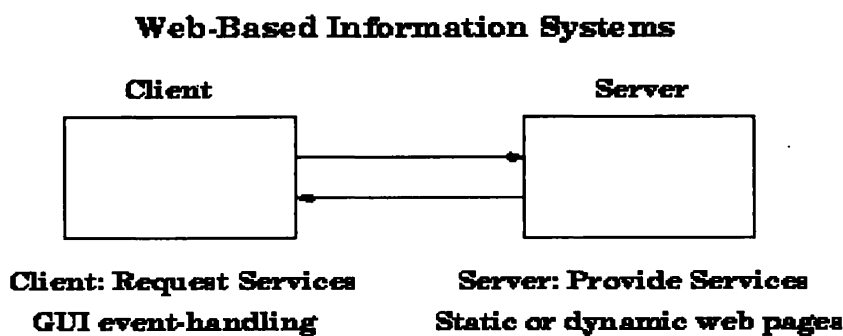


Figure 2.1

Netscape and Internet Explorer (IE) are client-side applications for web browsing. Such browsers support primarily the interpretation of static HTML, a special markup language used to compose web pages. In general, all WWW Browsers are able to universally render HTML pages *as expected*, i.e., to format the specified content in a manner intended by the author of the web page (Crouch, 2000).

Designers of web applications generally desire more and different kinds of interaction with the client-side user. Browsers accommodate this need in one of three ways:

1. Default, Client-Side WWW Browser Extensions.
2. Downloaded Client-Side WWW Browser Extensions.

3. **Server-side Web Server Extensions.** In addition to requesting static web pages from a web-server, it is also possible to request the execution of an arbitrary procedure that dynamically formats and returns web pages. For example, Microsoft's Active Server Pages (ASP), Java Servlets, or Java Server Pages (JSP) support this functionality.

Web-based servers may be composed of three components:

1. HTTP Server
2. Dynamic page generation
3. Application

A relational database is more often involving in web-based information systems. Many commercial and freeware relational databases exist, with varying degrees of features.

- o Very large, commercial RDBMS like those offered by Sybase and Oracle which offer a full range of database services.
- o Smaller, commercial packages like MS Access and Visual FoxPro offer more limited, but still very sophisticated features.
- o Other packages (mostly freeware), e.g., SQL Server and MySQL.

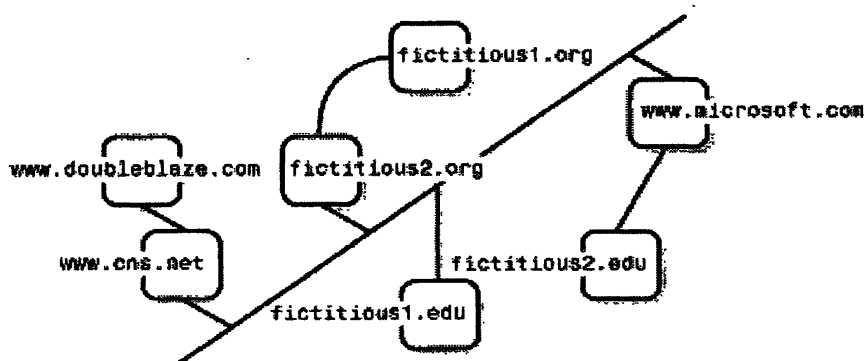
## **2.2 Active Server Pages at a glance**

### **2.2.1 In the Beginning**

The Internet, which has currently come to serve over 50 million users worldwide, had relatively humble beginnings. It began as a rather obscure network called ARPANET-a network used by the Department of Defense, USA, its contractors, and defense researchers. The ARPANET proved very useful and powerful, so much that over time it grew tremendously. Soon, a more general network called the Internet was created with the same

philosophy as ARPANET, only with a broader scope. In the late 1980s, the Internet spread across universities around the world and became more and more popular among researchers in the academic community (Crouch, 2000).

The popularity of the Internet has increased, users awareness also increased about the benefits of specific services, such as electronic mail and the ability to transfer files. Over time, the incredible potential of the Internet for sharing and distributing information became apparent as more and more people started using it. The Internet, simply put, is like a vast ocean of computers across the world connected together by a network of cables. Figure 2.2 shows a model of Internet.

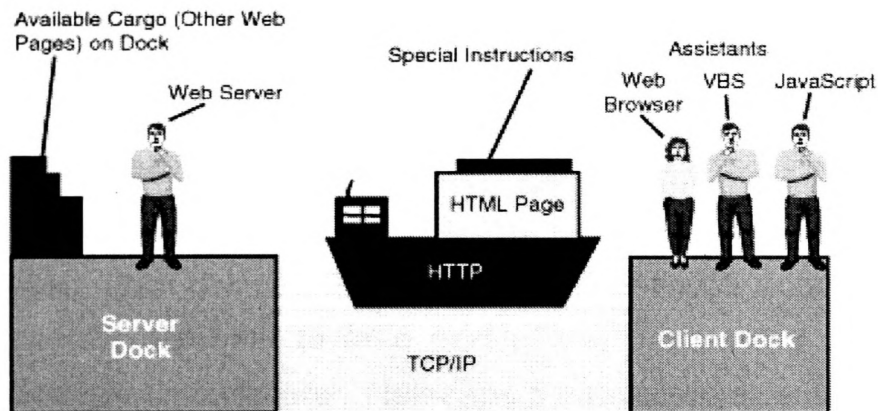


How the Internet is constructed

Figure 2.2

The World Wide Web (www) is by far the most talked-about service on the Internet today. Web pages are used within the www. The www is an information system that brings together data from many of the other Internet services under one set of protocols. It began in March 1989 when a group of high-energy physics researchers wanted a new protocol for distributing information on the Internet. The European Laboratory for Particle Physics, or CERN, actually proposed the standards, and a consortium of organizations, called the W3 Consortium, was created for continuing to develop the standards. The consortium put together a set of protocols for the www by creating the Hypertext Markup Language, or HTML. HTML is the underlying standard and means by which information is exchanged in the web. In

addition, various browser developers, most prominently Netscape and Microsoft, have at times extended HTML with their own additions. Essentially, the web consists of HTML documents, or Web pages, which are delivered by Web servers and can be interpreted by Web clients, or *browsers*. The protocol governing this procedure is called HTTP, which stands for Hypertext Transfer Protocol. A simplified version of the above process illustrated in figure 2.3.



Process of Hypertext Transfer Protocol

Figure 2.3

In the early days of the web, all information served to the client's browser was static. In other words the content for page A served to client 1 was exactly the same as the content for page A served to client 2. The web server did not dynamically generate any part of the site's content. But simply served request for static HTML pages loaded from the Web Server's file system and sent to the requesting client. There was no interactivity between the user and the server. The browser requested information and the server sent it (Weissinger, 2000).

Although the static Internet quickly evolved to include graphics and sounds the Web was still static with little interactivity and very little functionality beyond that provided by simple hyperlinks.

One of the first extensions of the static Internet was the creation of the Common Gateway Interface (CGI). The CGI provides a mechanism by which a web browser can communicate a request for the execution of an application on the web server. The result of this application is converted/formatted into a browser-readable (HTML) form and sent to the requesting browser (Weissinger, 2000).

CGI applications raised the bar on what was expected from a web site and transitioned the web from an easy way to share information to a viable platform for information processing. The response to this evolution of the web was rapidly accelerated growth and the beginning of the business world's interest in the Internet.

Part of this growth was the creation of several client-side scripting solutions that enabled client's machine to take part of the processing tasks. Among these client-side solutions are Netscape's JavaScript and Microsoft's VBScript.

During this huge growth of Internet based technology Microsoft released its Internet Information Server (IIS). Touted as being easy to use, scalable, portable, secure and extensible, it is also free and closely integrated with Microsoft Windows NT and Windows 2000 operating systems. It quickly became very popular (Weissinger, 2000).

Thus the distinction between Client-side programming and Server-side programming was recognized by this time. In Client-side programming the client's browser handles the task of interpreting a piece of code known as a script, which is usually implemented through a scripting language such as JavaScript or VBScript. On the contrary, in Server-side programming a particular task is handled by a piece of code stored in the server. Server-side programming used to be pretty difficult at this time and making something work via the Common Gateway Interface (CGI), which was the de facto standard then, required some knowledge of arcane programming languages like Perl or C. In addition, it was inefficient. Each time someone hits a CGI

script, a new process was created on the server; if your script was written in an interpreted language like Perl, the server had to start up another Perl interpreter, taking up processing time and memory. The situation got even hairier when it lived on a site that was getting a few thousand hits a day.

Also, you couldn't program in any of the nifty development interfaces, like Visual Basic or Visual C++. You were limited to watching it crash and burn, then checking out the server logs.

Microsoft attempted to change all this when they introduced Active Server Pages (ASP). ASPs are server-generated pages, which can call other programs to do things like access databases, serve different pages to different browsers - basically, anything we used to do with CGI. ASP is almost as efficient as writing code directly to the server's application program interface, and it's a lot more efficient than CGI because it runs as a service and can take advantage of multithreaded architectures.

According to the official word from the Microsoft site: "Active Server Pages is an open, compile-free application environment in which you can combine HTML, scripts, and reusable ActiveX server components to create dynamic and powerful Web-based business solutions. Active Server Pages enables server-side scripting for IIS with native support for both VBScript and Jscript."

ASP stuck around, though, because it made sense. It evolved into an "open technology framework," meaning you didn't have to use Microsoft products to create code in it, though that's still the best way to go, honestly. Nowadays, you can create ASP pages using whatever language you want, but VBScript is still the most common choice. It seems likely that more people will choose to use ASP because all of those Microsoft developer tools are actually pretty good and written to save you time. ASPs can also take advantage of COM and DCOM (Component Object Model and Distributed Component Object Model) objects with minimum effort.

## **2.2.2 Development with Scripts**

Scripting languages are great for creating applications quickly. Compared to formal programming languages, you generally need far fewer lines of script to accomplish a task. Now that Dynamic HTML and the Document Object Model have arrived, you can even combine server-side and client-side scripting to quickly develop a prototype of your ideas.

Or perhaps you simply want to serve Web pages that are sensitive to the features supported by a particular browser version. This applies to different versions of a single browser (Internet Explorer 3. x and 4.0, for example), different operating systems (Windows and UNIX, for example), and to completely different browsers (such as Navigator and Internet Explorer). You can create scripts that are appropriate for each browser, and invoke the appropriate commands to serve pages uniquely suited to the features of any browser. The Browser Capabilities object provides detailed information about the features supported for each browser version. For example, you can design an application for Internet Explorer 4.0 that displays different HTML code for down-level browsers such as Internet Explorer 3.02, or for alternate browsers, such as Netscape Navigator. This enables you to maintain one set of content pages, with conditional browser detection for features on different browsers built right into each page.

You can do a lot of development with scripts. You can incorporate business rules, for example, into a complete client/server or multi-tiered business application such as a telemarketing sales application. Or you can simply verify that a visitor has been to your site before and recall preferences for that user

## **2.2.3 Features of ASP**

### **2.2.3.1 Language Independence**

Active Server Pages (ASP) technology is language-independent. Two of the most common scripting languages are supported right out of the box: VBScript and JScript. Support for other scripting languages, such as Perl, is

available. Whatever scripting language you use, you can simply enclose script statements in special delimiters for ASP. The starting delimiter is <%, and the closing delimiter is %> (Cyber Tec, 2003).

### **2.2.3.2 Built-In and Installable Objects**

Most of the functionality you can build into an ASP page comes from objects on the server. IIS 4.0 comes with some built-in objects, as well as a number of installable objects. You can also use objects created by a developer you know, or create and use your own objects.

### **2.2.3.3 Built-In Objects**

Six objects come built-in with IIS 4.0. These objects enable your pages to communicate effectively with the server, the application, the current session, and the user:

- Request object -- Gets information from the user. You can get FORM data, read cookies, and so forth.
- Response object -- Sends information to the user. You can send text to the page, redirect to another URL, and set cookies.
- Server object -- Interacts with the server. You can access a database, read files, and find out about the capabilities of the browser.
- Session object -- Enables you to manage information about the current session (each user has one session per open browser).
- Application object -- Will store information for all sessions.
- ObjectContext object -- Used for committing and aborting transactions (new in IIS 4.0)

### **2.2.3.4 Installable Objects**

IIS 4.0 comes with a variety of installable objects, and you can also use your own custom objects on the server side. The installable objects include (Cyber Tec, 2003):



- **Ad Rotator** -- Rotates advertisements displayed on a page using a schedule
- **Browser Capabilities** -- Gives you access to information about the client browser's capabilities, type, and version
- **Database Access** -- Provides access to databases using ADO (ActiveX® Data Objects)
- **Content Linking** -- Creates tables of contents for Web pages, and links them sequentially like the pages of a book
- **File Access** -- Provides access to file input and output
- **Collaboration Data Objects for Windows NT Server** -- Provides the ability to send and receive messages from your Web page

### **2.2.3.5 Databases and Cookies**

#### **2.2.3.5.1 Data-Driven Web Pages**

IIS 4.0 and ASP make it easy to access data and put it on a Web page. You can simply display data from an ODBC-compliant database, or you can use ASP to make decisions about what to display on Web pages. Here's the process:

1. **Create an ODBC Data Source** -- Specify a Data Source Name (DSN) in the server registry or in a file, and grant access to all users, specific users, or groups.
2. **Make the Connection** -- Use the Connection object to point to a DSN, and create a Recordset object to gain interactive access to data.
3. **Execute Queries** -- You can construct queries using standard SQL syntax. You can submit queries using Connection methods, Recordset methods, or with the Command object.
4. **Reissue Queries** -- You can use the Command object to change query parameters and quickly resubmit the query. For example, you can INSERT a record into your database with a set of values, reset the values for the next record, insert it, and so on. The Command object makes it easy to construct valid queries, too, because it uses a parameter-driven interface that is easy to set up and maintain.

5. Access Data Using Forms -- With forms, your users can perform queries, read data, change data, add records, and so forth. Thus you can create a complete client-server solution. Using the transaction features built into IIS 4.0, you can even create multi-tier business solutions on the Internet.

#### **2.2.3.5.2 Cookies**

You can use cookies, as well as session and application variables, to store and retrieve information. IIS makes cookie values available to you with the Request object's Cookies collection. You can set cookie values with the Response object. Because IIS sends all header information before the document content, you must set cookies before the <HTML> tag. For example, to check the value of a cookie, you can use this code (Cyber Tec, 2003):

```
<% Dim strNameCookie strNameCookie = Request.Cookies("nameCookie")
%>
```

and then set some session variables based on the cookie value. You can use default values if there is no nameCookie, or the existing cookie values if there is a nameCookie:

```
<% If strNameCookie = "" Then Session("strDefaultPage") = ""
Session("strDefaultLevel") = "1" Else Session("strDefaultPage") =
Request.Cookies("pageCookie") Session("strDefaultLevel") =
Request.Cookies("levelCookie") End If %>
```

#### **2.2.3.6 Creating Client/Server Web Applications**

ASP technology makes it easy to create complete Web-based applications. You don't have to use the full breadth of ASP technology on your Web pages, of course; you can simply use ASP as a template language to create flexible, easy-to-maintain Web pages. Or you can create a client/server application: define a directory for the application, and all files in that directory (and in all of its sub-directories) exist as an application. Pages can communicate with each other, or with the application object. This is possible because you can persist

selected information across a user session and/or the entire application as needed (Berson, 1992).

A typical application can include Web pages, databases, controls, server-side scripts, client-side scripts, multiple servers, and transactions. The first visitor to the application starts the application, and each visitor has a unique session. You can create just about any kind of application because of the multiple levels of control available to you: scripts, built-in ActiveX components, third-party components, custom components of your own, databases, and transactions.

Other key features of applications include:

- **Start/End Events** -- You can start and end applications as needed, with complete control over what occurs at both events. You can define subs or functions that run at the start and end of an application, enabling you to configure the application automatically each time it starts, or clean up each time it ends.
- **Isolated Processes** -- You can run each application in its own process on the server. This protects applications from each other -- should one application go down, it would not take the server or other applications with it.
- **Application Object** -- The Application object has its own properties and methods. You can set or read properties and use methods on any page that is part of the application. You can set variables with application scope when the application starts, and change them at run time as needed to indicate the application's current state. For example, you could store the number of current sessions in the Application object, and increment/decrement it as visitors come and go from your site.

#### **2.2.3.7 Sessions and Web Applications**

The Session object enables you to communicate effectively between the pages of your application, and to store information about visitors so you can easily retrieve it on their next visit.

Each visitor to your site has a Session object with a unique session ID. You can use the Session object to maintain information about that user during the session. The session ID is stored in a cookie, and you can add/read your own cookies as needed. You can use the session ID cookie to rebuild session information from one session to the next. For example, if a user visits your site and establishes preferences that you store in a database, you can use the session ID cookie to read the correct database information every time that visitor returns to your site. This makes it easy to create a "personalized" Web site.

You can store variables in the Session object that have session scope. That means you can access the variable from any page in the application that the user visits. You can also establish handlers for the session start and end events, integrate forms into your sessions, and so forth.

#### **2.2.3.8 Using scripts to extend ASP**

VBScript is one of the most popular scripting languages. A scripting language is a type of programming language used to provide control in another host environment. It is interpreted rather than compiled. This means that a program built with a scripting language must be run in the environment that contains the scripting language's interpreter and cannot be run as a stand-alone application (Moniz, 2000).

Using VBScript on the server in an ASP page is not very different from using it in applications or on ordinary Web pages. Nearly all of the VBScript commands are available for use on the server. VBScript commands that interact with the user, however, are not available. For example, imagine a command that opens a dialog box on the server. No one is around to dismiss it, and the system can do nothing until someone dismisses it!

The VBScript statements that present user interface elements are InputBox and MsgBox. In addition, the VBScript function Create Object is replaced by a method of the Server object. This is necessary to track the object instances on the server side. You can add comments to your script just as you normally

do. However, you cannot add comments inside an output expression. An output expression is an expression or value that is evaluated and written to the Web page. It is contained within the delimiters `<%=` and `%>`.

#### 2.2.4 ASP Syntax

An Active Server Page is an HTML file with some additional code in it, separated from the HTML markup by a different delimiter:

```
<% ASP Code evaluated here %>
```

Example:

```
<%@ Language=VBScript %>
<HTML>
<HEAD>
<TITLE>Example 1</TITLE>
</HEAD>
<BODY bgcolor=Lime aLink=DarkTurquoise>
<P> </P>
<% Response.Write("Hello, world!") %>
</BODY>
</HTML>
```

If this file is saved as `.asp` and put it on an ASP-ready server, the message "Hello, world!" will be displayed in the page above.

The important thing to note is that the server will assume that anything between the delimiters is code and will then try to execute it. Also, the first line, `<%@ Language=VBScript %>` is optional. If you leave it out, the server will assume a VBScript (Lycos, 2003).

VBScript, although it supports various scalar values (integers, floating-point numbers, strings, etc.), has only one primary data type: the Variant, a peculiar data type that transforms itself to fit within the context in which it is currently being used. Further, VBScript has all the requisite programming-language operators, conditional statements, and control structures.

The facilities provided by VBScripts are

- 1 **Request** - to get information from the user

- 2 **Response** - to send information to the user
- 3 **Server** - to control the Internet Information Server
- 4 **Session** - to store information about and change settings for the user's current Web-server session
- 5 **Application** - to share application-level information and control settings for the lifetime of the application

Each of these objects has a collection of functions that it can perform, called "methods," and one or more properties, each called with the usual object-orientation-style code [value = Object.Property or value = Object.Method()]. The "Hello, world!" example above uses one of the most common methods of the Response object: write. A programmer can create objects that are ASP-accessible by using the Component Object Model, Microsoft's new way of creating OO code (Lycos, 2003).

### 2.2.5 Browser Detection

Client-side scripting can be used for browser detection. However, if the client does not support client-side scripting, like Lynx, problems may arise in executing the script.

The following example gives an idea of how ASP can handle this problem(Lycos, 2003):

```
<%@ Language=VBScript %>
<HTML>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft Visual Studio 6.0">
</HEAD>
<BODY>

<%
dim BrowserType
set bc = Server.CreateObject("MSWC.BrowserType")
if bc.browser="IE" then
```

```

    BrowserType = "MSIE"
elseif bc.browser="Netscape" then
    BrowserType = "Netscape"
elseif bc.browser="Lynx" then
    BrowserType = "Lynx"
end if
%>

<%
select case BrowserType
case "Lynx"
    Response.Write("You're using Lynx! How do you manage to live
    without 3-D backgrounds and endless download times? Whatever!")
case "MSIE"
    Response.Write("You're using Internet Explorer! Thank you for
    helping keep Microsoft afloat!")
case "Netscape"
    Response.Write("You're using Netscape! And you're wearing those
    great pink pants! ASP knows everything about you!")
case else
    Response.Write("You're using some other browser I don't know
    about.")
End select
%>
<BR>
</BODY>
</HTML>

```

The result of this code tells the type of browser which accessed this code running.

Microsoft provides objects, such as MSWC.BrowserType above, which can be used to detect browser types easily. This server object's property, MSWC.BrowserType.browser, is much simpler to deal with than the usual HTTP\_USER\_AGENT because it does not say "Mozilla (compatible)" for

Internet Explorer, and a parse through the string just to get the version number is not necessary. It is another property, as is the client platform (platform). It makes for cleaner code, which is always an efficient coding practice. With a few more branches in the "if statement", a page can be served for all of the major browser, platform, and version groups providing the best browsing experience possible. For example,

```
[Code snippet: if bc.majorver = "4" and InStr(bc.platform,"Win") <> 0 then ]
```

## **2.3 Web Application development with Macromedia Dreamweaver**

### **2.3.1 Basics of Web Applications**

A web application is a website that contains pages stored on a web server with partly or entirely undetermined content. The final content of a page is determined only when the user requests a page from the web server. Because the final content of the page varies from request to request based on the user's actions, this kind of page is called a dynamic page.

### **2.3.2 Common uses for web applications**

Web applications have many uses for both users and developers, including the following:

- Let users find information quickly and easily on a content-rich website.

This kind of web application gives users the ability to search, organize, and navigate content as they see fit. Examples include company intranets, Microsoft MSDN, and Amazon.com.

- Collect, save, and analyze data provided by users.

In the past, data entered in HTML forms was sent as e-mail messages to employees or CGI applications for processing. A web application can save form data directly into a database and also extract the data and create web-based reports for analysis. Examples include online banking pages, store check-out pages, surveys, and user-feedback forms (Crouch, 2000).



- Update websites that have constantly changing content.

A web application frees the web designer from continually updating the site's HTML. Content providers such as news editors provide the web application with content and the web application updates the site automatically. Examples include the Economist and CNN.

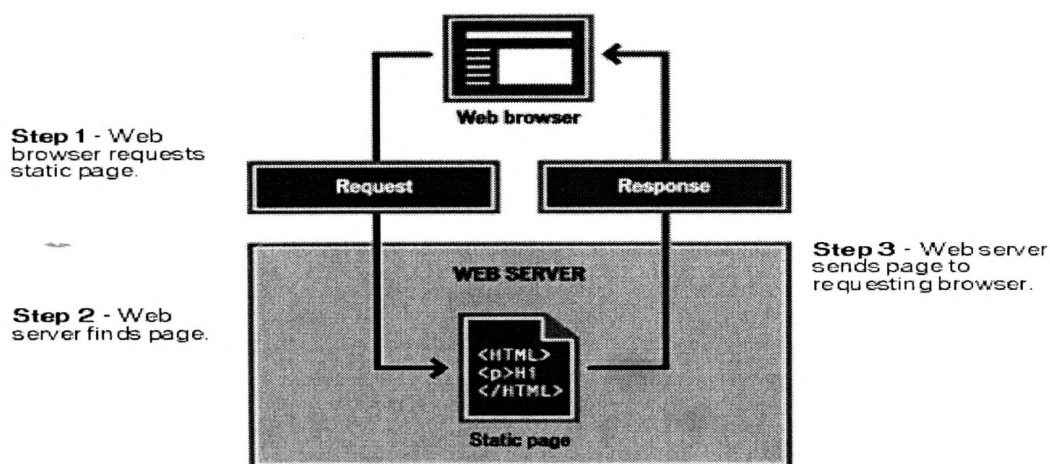
### 2.3.3 Processing regular web pages

A regular website comprises a set of related HTML pages and files hosted on a computer running a web server.

A web server is software that serves web pages in response to requests from web browsers. A page request is generated when a user clicks a link on a web page, chooses a bookmark in a browser, or enters a URL in a browser's Address text box and clicks Go.

The final content of a regular web page is determined by the page designer and doesn't change when the page is requested.

When the web server receives a request for a static page, the server reads the request, finds the page, and sends it to the requesting browser, as shown in the following figure:



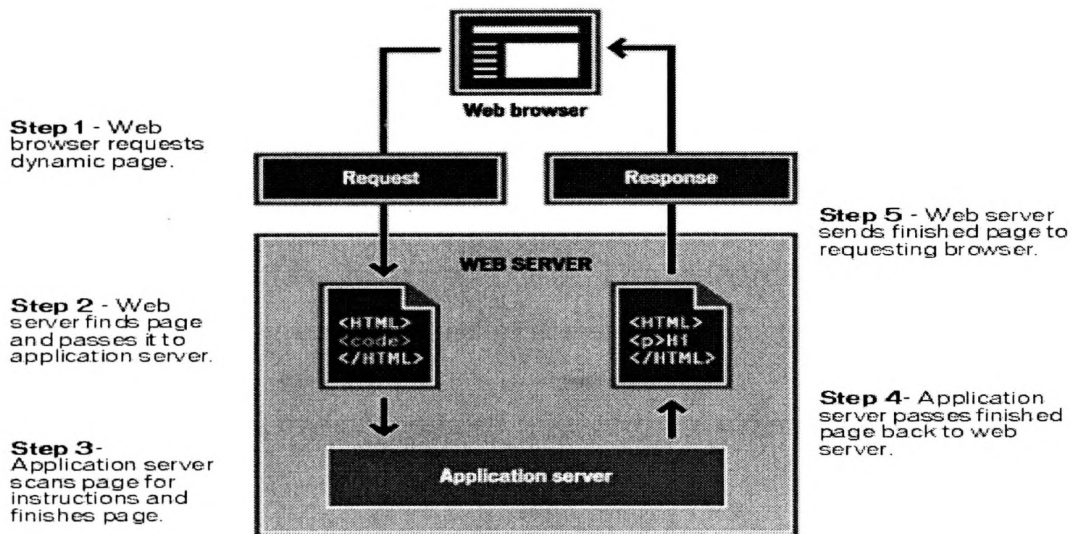
Process 1 within the web server

Figure 2.4

### 2.3.4 Processing dynamic pages

When a web server receives a request for a regular web page, the server sends the page to the requesting browser without further ado. The web server reacts differently when it receives a request for a dynamic page: it passes the page to a special software extension pble for finishing the page. This special software is called an application server.

The application server reads the code on the page, finishes the page according to the instructions in the code, then removes the code from the page. The result is a static page that the application server passes back to the web server, which then sends the page to the requesting browser. All the browser gets when the page arrives is pure HTML. Here's a view of the process:



Process 2 within the web server

Figure 2.5

### **2.3.5 Accessing a database**

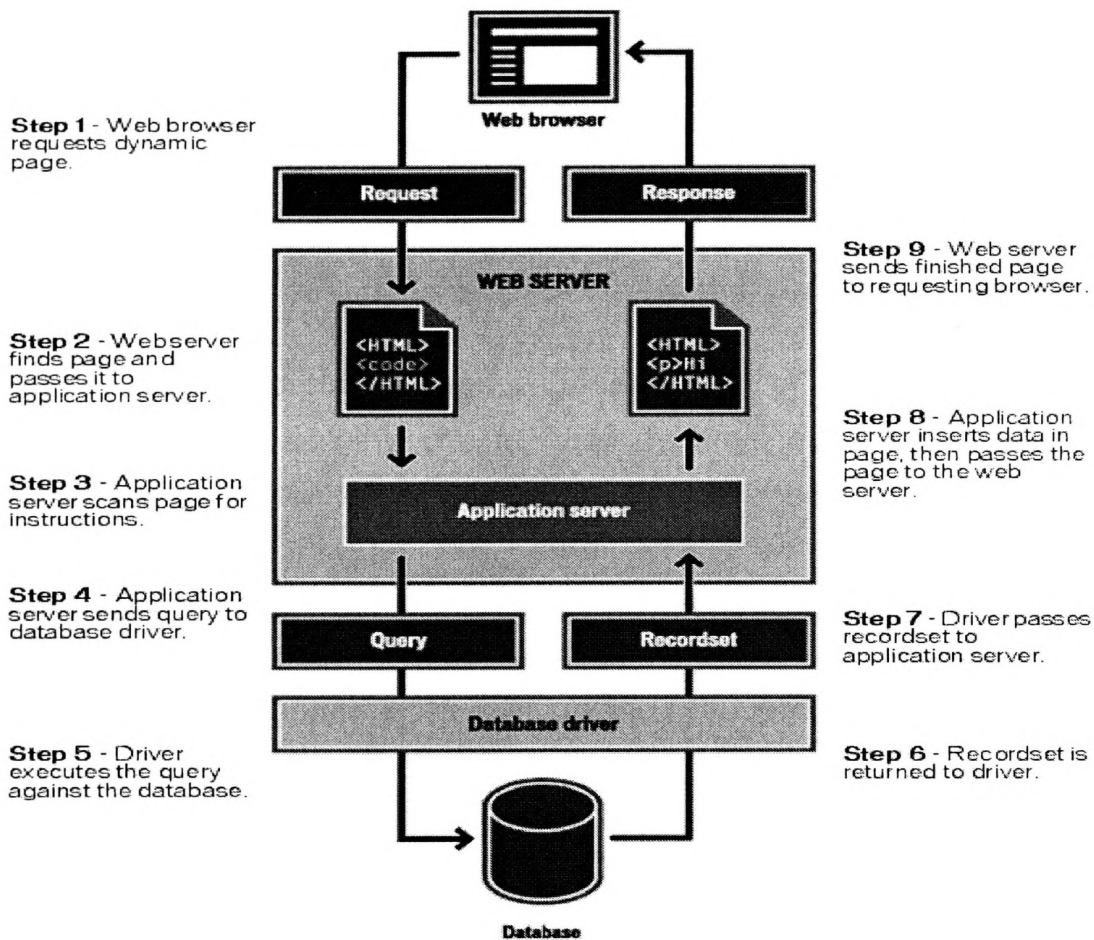
An application server lets you work with server-side resources such as databases. For example, a dynamic page may instruct the application server to extract data from a database and insert it into the page's HTML (Crouch, 2000).

The instruction to extract data from a database is called a database query. A query consists of search criteria expressed in a database language called SQL (Structured Query Language). The SQL query is written into the page's server-side scripts or tags.

An application server cannot communicate directly with a database because the database's proprietary format renders the data undecipherable in much the same way that a Word document opened in Notepad is undecipherable. The application server can communicate only through the intermediary of a database driver. A database driver is software that acts like an interpreter between the application server and the database.

After the driver establishes communication, the query is executed against the database and a recordset is created. A recordset is a subset of data extracted from one or more tables in a database. The recordset is returned to the application server and the data used in the dynamic page.

Here's an illustration of the process of querying a database and returning data to the browser



Process 3 within the web server

Figure 2.6

Macromedia Dreamweaver MX is a professional HTML editor for designing, coding, and developing websites, web pages, and web applications. Whether you enjoy the control of hand-coding HTML or prefer to work in a visual editing environment, Dreamweaver provides you with helpful tools to enhance your web creation experience.

The visual editing features in Dreamweaver let you quickly create pages without writing a line of code. You can view all your site elements or assets and drag them from an easy-to-use panel directly into a document. You can streamline your development workflow by creating and editing images in

Macromedia Fireworks, then importing them directly into Dreamweaver, or by adding Macromedia Flash objects you create directly in Dreamweaver.

Dreamweaver also includes many coding-related tools and features, including code editing tools in the Code view (such as code coloring and tag completion); reference material on HTML, CSS, JavaScript, CFML, ASP, and JSP; and a JavaScript Debugger. Macromedia Roundtrip HTML technology imports your hand-coded HTML documents without reformatting the code; you can then choose to reformat code with your preferred formatting style.

Dreamweaver now incorporates and expands on all of the capabilities from Macromedia UltraDev, helping you to build dynamic database-backed web applications using server languages such as ASP, ASP.NET, ColdFusion Markup Language (CFML), JSP, and PHP. Dreamweaver is fully customizable. You can create your own objects and commands, modify keyboard shortcuts, and even write JavaScript code to extend Dreamweaver capabilities with new behaviors, Property inspectors, and site reports.

## **3. Methodology**

### **3.1 Approaches to the Development of the Student Information System**

The tasks within the purview of the project were divided into following phases.

1. Building interfaces
2. Constructing database for the storage of student data
3. Connecting the database through the interfaces developed and testing the connectivity
4. Allowing the facilities for updating the database

#### **3.1.1 Primary Tools used for the Development**

- a. Macromedia Dreamweaver MX (Version 6.0)
- b. Microsoft Access 2000 (Version 9.0.3821 SR-1)

#### **3.1.2 Platform used for the Development**

The above phases were carried out in a computer system with following specifications.

- Processor: Intel Pentium IV
- Operating System: Windows 2000 Professional Server with built-in support for IIS
- Main Memory: 128 MB DDR SDRAM
- Video Memory: 32 MB

### **3.2 Interface Building**

This was carried out using Macromedia Dreamweaver MX. This software facilitates the use of a user-friendly designing environment for the construction

of web pages and hence was selected for this particular project work. The Dreamweaver workspace appears in Appendix I.

Automatic integration with the ASP technology was one of the key benefits associated with the above software.

### **3.2.1 Steps used to develop the Web Application using Dreamweaver**

#### **1. Defining a new site**

- i. Site > New Site was chosen from the menu bar
- ii. The steps in the wizard were continued
- iii. The button named Done was pressed to view the site details

#### **2. Creating and saving a new page**

- i. File> New was selected from the menu bar
- ii. In the category list on the left, the Page Designs category was selected.
- iii. In the Page Designs list the item named Text: Article D with Navigation was selected.
- iv. Clicking on Create allowed for selecting the location for the new page.

#### **3. Setting a page title**

In the Title text box, where it says "Untitled Document," a title was typed for the page. Then Enter was pressed to see the page title update in the Document window's title bar.

#### **4. Adding images**

- i. An existing image placeholder can be used for this purpose
- ii. Double-clicking on this opened the Image Source Dialogue Box
- iii. An image was selected by browsing the folders
- iv. OK was pressed and the image was inserted

#### **5. Setting the background colour**

- i. In the Property inspector, the Background Color button was clicked that appeared next to the lower of the two Bg labels.
- ii. Using the color picker a colour was selected and applied.

#### **6. Viewing the HTML code for the document**

- i. In the Document toolbar, the Code and Design view button was clicked.
  - ii. The window splitted showing the underlying HTML code .This is shown in Appendix II.
7. Adding text links between pages
- i. The Property inspector was opened by choosing Window > Properties.
  - ii. The folder icon next to the Link text box in the Property inspector was clicked and the file to be linked was selected by browsing.
8. Previewing in Browser
- i. The desired document was selected.
  - ii. F 12 was pressed.

### **3.2.2 Connecting to the sample database**

The following steps were used in generating the automatic connection.

1. Any ASP page was opened in Dreamweaver, then the Databases panel (Window > Databases) was opened.
2. The plus (+) button on the panel was clicked and the Data Source Name (DSN) was selected from the pop-up menu. The Data Source Name (DSN) dialog box appeared.
3. **connGlobal** was entered as the connection name.
4. The Using Local DSN option was selected.
5. GlobalCar was selected from the list of DSNs.
6. The Test button was clicked
7. The OK button was clicked and the new connection appeared in the Database panel.

### **3.3 Database construction**

This was achieved using Microsoft Access 2000. The key areas into which the student details to be categorized into: see Appendix IV.



- i. First Name
- ii. Last Name
- iii. Registration No
- iv. NIC No.
- v. Gender
- vi. Postal Address
- vii. E-mail address
- viii. Telephone No.
- ix. Date Registered
- x. Faculty
- xi. Department
- xii. Degree Course
- xiii. Academic Year

The above data were gathered with respect to each student and stored in respective table in the database.

A separate table was designed to store examination results and the key fields in the table were identified as follows.

- i. Registration No
- ii. Year and Semester
- iii. Subject and Result

### **3.4 Connecting the Database**

Structured Query Language (SQL) was utilized as the tool for establishing the connection between the database and the respective interface. The piece of code governing this task was also generated through Dreamweaver MX. Corresponding codings are displayed in Appendix V.

The connectivity was tested using the built-in support provided by IIS through the Operating System (Windows 2000).

### **3.5 Allowing the facilities for updating**

This was also implemented through the Dreamweaver software. Access to update views was restricted using a username and password.

### **3.6 Unit Testing**

In this stage each interface was considered individually and tested according to the test plan. If the given output was not the desired out that bug was reported and fixed it. This procedure was repeated with all other interfaces.

### **3.7 System Testing**

The system was installed in the web server of the intranet. Then Logged into the server form another computer (Client), which was in the Intranet. The client computer's web browser was used to access the student information system. The home page of the system was named as *http://IP/main.asp* where IP was the IP address of the server. Once the homepage appeared the system testing was done according to the test plan.

## **4. Results & Discussion**

The result of this project was a user-friendly, web-based student information system for the Sabaragamuwa University. This was achieved by following the correct software development methodologies. The system provides all the student information including their personal and examination. The system meets all the functional specifications given by the Center for Computer of the University. Hence the current manual system could be replaced using the new system.

### **Components of the developed system**

The final system contains the following.

- Interfaces: 06 interfaces including the main interface
- Databases: 01 database with 03 tables in it

## **5. Conclusions and Recommendations**

The output of the project work could replace the current manual data processing system of the University. The use of the ASP technology integrated with the Dreamweaver MX software allows easy maintenance of this system. The system can easily be linked with the web server to make it available online. The primary advantages of the system will include the following.

1. Handling of bulk amounts of data
2. Fast processing of data
3. Local access as well as remote access

The major factor determining the efficiency of the system will be the bandwidth of the link used in the case of remote access and the server performances.

### **Recommendations for further work**

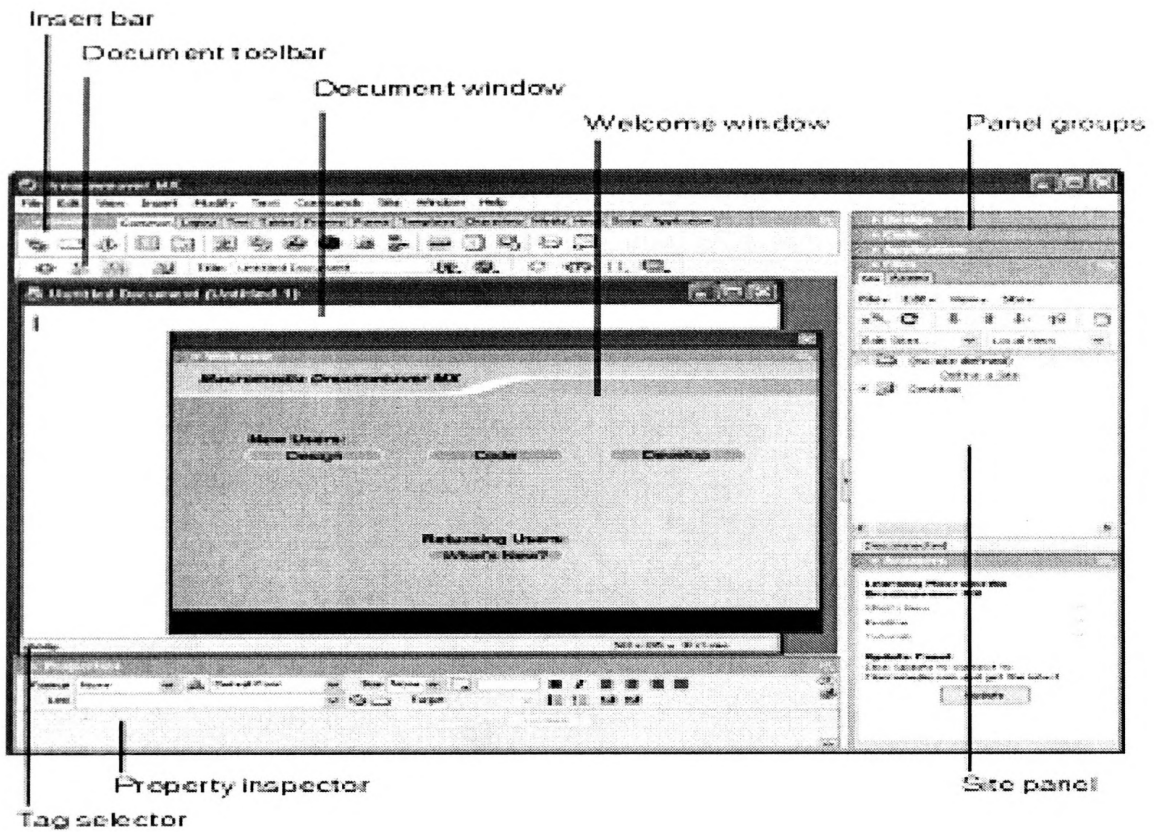
The system could be further enhanced in the following ways.

1. The system can further be extended to include staff details.
2. A search facility could be developed to incorporate with the database.
3. The facility for viewing individual examination results could also be provided.
4. The SQL database that gives more facilities in relational database manipulation could be used for back end.

## REFERENCES

- Berson, A. (1992) Client/Server Architecture. McGraw-Hill Book Co, Singapore, 452 p.
- CyberTec. (2003) ASP Technology. CyberTec Communications Group, Inc., 2101 NW 98th Avenue Sunrise, FL 33322 (Accessed from: <http://www.cybertec.net/asp.html>).
- Crouch, M. J. (2000) Web programming with ASP and COM. Addison Wesley Longman pte. Ltd, Singapore, 393p.
- Ghezzi, C. (1996) Fundamentals of Software Engineering. Prentice-Hall pte. Ltd., India, New Delhi, 573 p.
- Lycos. (2003) Introduction to active server pages. Lycos® is a registered trademark of Carnegie Mellon University (Accessed from: <http://hotwired.lycos.com/webmonkey/98/39/index2a.html?tw=programming>)
- Moniz, J. (2000) Enterprise Application Architecture-with VB MTS and ASP. Shroff publishers, Moniz, 786 p.
- Pressman, R.S. (2001) Software Engineering. 5<sup>th</sup> Ed. McGraw-Hill companies, Inc., New York, 860p.
- Weissinger, A. K. (2000) ASP IN A NUTSHELL. 2<sup>nd</sup> Ed. Shroff publisher, Mumbai, 473 p.

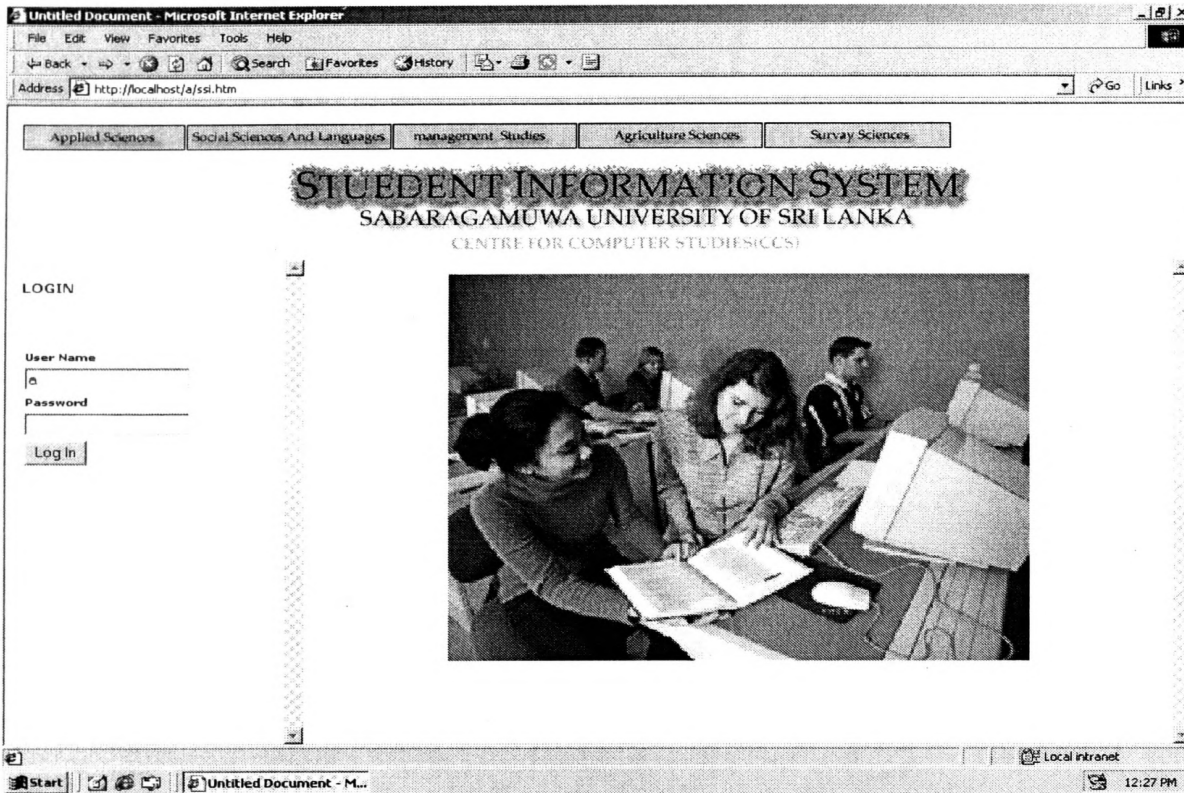
## Appendix I - Dreamweaver Workspace



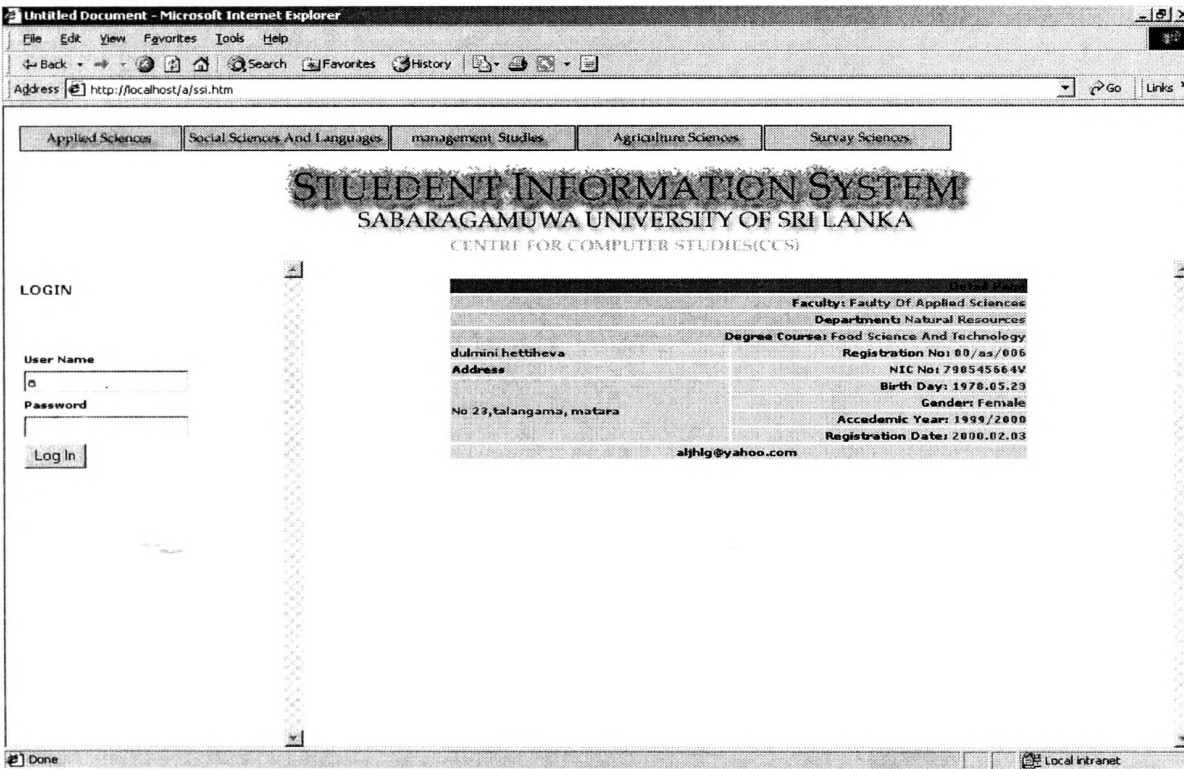
## Appendix II - A web page and its underlying codes



### Appendix III



The start up screen



The Students Personal Details Screen

Untitled Document - Microsoft Internet Explorer - [Working Offline]

Address: http://localhost/university/view\_result\_all.asp

Results									
Reg No	Year And Semester	Computer Literacy	Mechnics	Physics	A C Theory	Basic Electrcy	Organic chemistry	Inorganic chemistry	Management
00/as/54	Year 1 Semester 1	D	B	C	B	C	A	A	B
00/as/52	Year 1 Semester 1	D	B	C	B	C	A	C	B
00/as/53	Year 1 Semester 1	D	B	C	B	C	A	D	B
00/as/55	Year 1 Semester 1	D	B	C	B	C	A	D	B
00/as/56	Year 1 Semester 1	D	B	C	B	C	A	A	A
00/as/57	Year 1 Semester 1	D	B	C	B	C	A	A	D
00/as/58	Year 1 Semester 1	D	B	C	B	C	A	A	E
00/as/59	Year 1 Semester 1	A	B	C	B	C	A	A	E

Done Local Intranet 5:31 PM

The Students Results Screen

Untitled Document - Microsoft Internet Explorer

Address: http://localhost/university/logging\_fail.asp

**:: Login Fail ..!**

**Password or Username is not correct.**

**Make sure lowercase is activated**

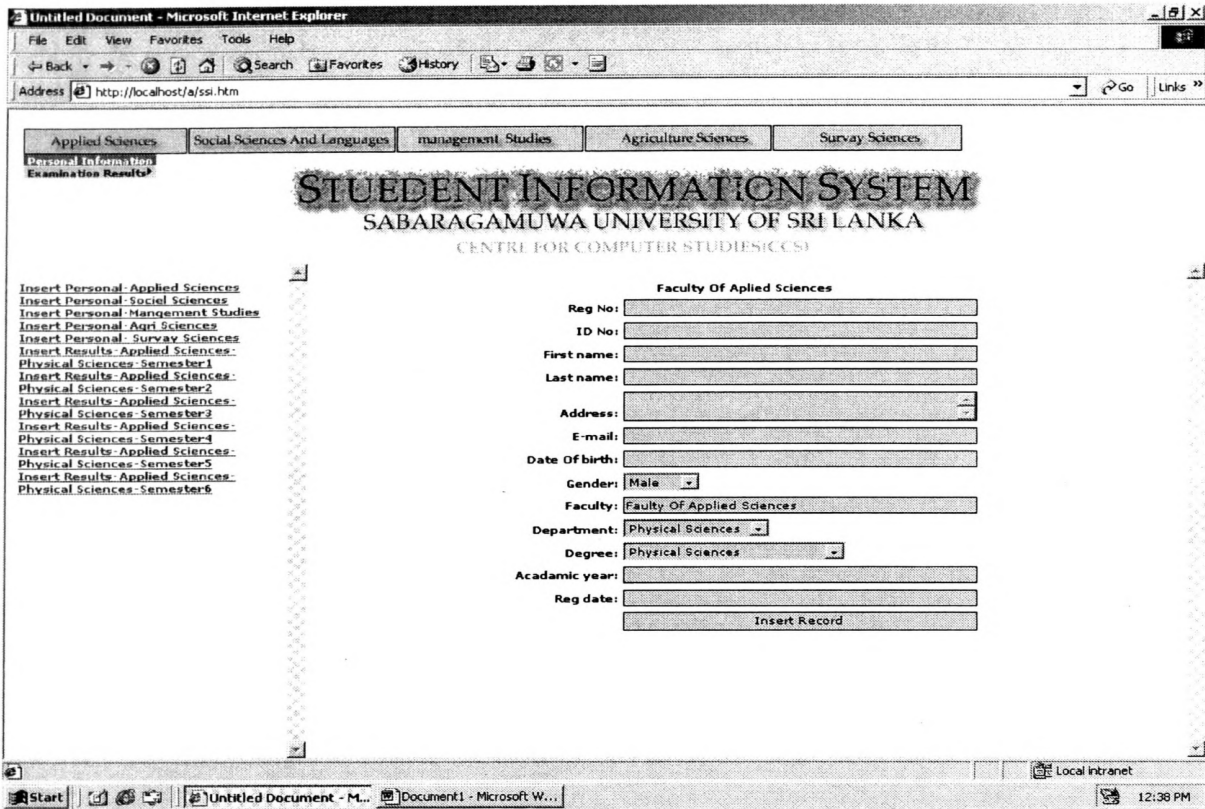
[If you wish to try again click here](#)

Center for computer studies (CCS)

Local Intranet 10:57 AM

The login Fail Screen





The Data Entry Selection Screen

## Appendix IV

reg_no	id_no	first_name	last_name	address	e_mail	phone	reg_date	faculty	department	acc
00/as/075	790591674v	vijekumara	wijesinghe	seedeve,ihalabel	wdakumara@yc	0777454731	2001.03.05	applied	Physical	200
00/as/12	798656342v	saman	silva	No 1, Beligalla,	ssg@yahoo.com	0472278990	2001.03.12	Applied	Natural Resourc	200
00/as/13	312313545v	kumara	ranasinghe	No2,kahawatta	fgh@yahoo.com	0472278996	12/21/03	Applied	Natural Resourc	200
00/as/14	534756697v	nalin	perera	No5, baseline rc	nal@yahoo.com	0112346576	2001.03.12	Business studie	Accountancy	200
00/as/167	790591674v	MAnjula	mohomed	kalutara	kuma@yahoo.c	0472278990	2001.03.12	Socail sciences	Languages	200
00/as/19	803344264v	mlani	perera	No34/2,yatiyan	ads@yahoo.com	0412346576	2001.03.05	applied	Physical	200
00/as/23	790591674v	MAnjula	mohomed	kalutara	kuma@yahoo.c	0472278990	2001.03.12	Socail sciences	Languages	200
00/as/34	790591674v	MAnjula	mohomed	kalutara	kuma@yahoo.c	0472278990	2001.03.12	Socail sciences	Languages	200
00/as/45	123456789v	kasun	gunasinghe	No7/34, colomb	skuma@yahoo.	0112346576	2001.03.12	applied	Physical	200
00/as/47	123456689v	nimal	weerasinghe	No7/45, colomb	weera@yahoo.c	0112346654	2001.03.12	applied	Physical	200
00/as/48	166456689v	kamal	silva	No7/45, Nigamt	kamall@yahoo.	0112346633	2001.03.12	applied	Physical	200
00/as/49	766456666v	ruwani	perera	No7/45, Badulla	ruwa44@yahoo.	0552234663	2001.03.12	applied	Physical	200
00/as/50	766456666v	thanuja	perera	No99/45, Badull	thanu2@yahoo.	0552234664	2001.03.12	applied	Physical	200
00/as/51	766456666v	kumari	perera	No99/45, Badull	fthu@yahoo.com	0552234664	2001.03.12	applied	Physical	200
00/as/52	834756697v	ruwani	Samaranayake	No99/45, rathna	ruwan44@yahoo.	0452234664	2001.03.12	applied	Physical	200
00/as/53	834756697v	anjula	Samaranayake	No99/45, rathna	anju22@yahoo.	0452234664	2001.03.12	applied	Physical	200
00/as/54	834756697v	chathuri	perera	No99/45, rathna	chathu@yahoo.	0452234664	2001.03.12	applied	Physical	200
00/as/55	834756697v	gayani	perera	No99/45, rathna	gay22@yahoo.c	0452234664	2001.03.12	applied	Physical	200
00/as/56	790591674v	MAnjula	mohomed	kalutara	kuma@yahoo.c	0472278990	2001.03.12	Socail sciences	Languages	200
00/as/57	834756697v	kasuni	Samaranayake	No99/45, rathna	kasu2@yahoo.c	0452234664	2001.03.12	applied	Physical	200
00/as/911	800591674v	gayan	mohomed	kalutara	kuma@yahoo.c	0472278990	2001.03.12	Socail sciences	Languages	200
00/as/912	800591674v	gayan	mohomed	kalutara	kuma@yahoo.c	0472278990	2001.03.12	Socail sciences	Languages	200
00/as/923	800591674v	gayan	mohomed	kalutara	kuma@yahoo.c	0472278990	2001.03.12	Socail sciences	Languages	200
00/as/93	790591674v	MAnjula	mohomed	kalutara	kuma@yahoo.c	0472278990	2001.03.12	Socail sciences	Languages	200
00/as/934	790591674v	MAnjula	mohomed	kalutara	kuma@yahoo.c	0472278990	2001.03.12	Socail sciences	Languages	200
00/as/94	800591674v	gayan	mohomed	kalutara	kuma@yahoo.c	0472278990	2001.03.12	Socail sciences	Languages	200
00/as/956	800591674v	gayan	mohomed	kalutara	kuma@yahoo.c	0472278990	2001.03.12	Socail sciences	Languages	200
00/as/98	790591674v	MAnjula	mohomed	kalutara	kuma@yahoo.c	0472278990	2001.03.12	Socail sciences	Languages	200
00/as/989	800591674v	gayan	mohomed	kalutara	kuma@yahoo.c	0472278990	2001.03.12	Socail sciences	Languages	200
00/as/999	800591674v	gayan	mohomed	kalutara	kuma@yahoo.c	0472278990	2001.03.12	Socail sciences	Languages	200
00/sc/075	834756697v	kumari	vitharana	no45/5,Badulla	kuma@yahoo.c	05522345678	2001.03.12	Socail sciences	Languages	200
00/ss/14	790591674v	das	Samaranayake	No34/1,Malmbe	webs@yahoo.c	0472278996	2001.03.05	Survey	Survey Science	200
99/hc/73	99134245v	sunil	mohomed	No3 trincomalee	sunil@yahoo.com	22123456	2001.03.05	Arnyculture	Exnort Arnycult	200

The Students Personal Data Table

Microsoft Access - [result : Table]

File Edit View Insert Format Records Tools Window Help

reg no year\_and\_sem sub1 grade1 sub2 grade2 sub3 grade3 sub4 grade4 s

00/as/075	Year 1 Semester	Computer Litara C		Mechnics	B	Physics	B	A C Theory	C	Basic
00/as/12	Year 1 Semester	Computer Litara B		Mechnics	B	Physics	C	A C Theory	B	Basic
00/as/14	Year 1 Semester	Computer Litara C		Mechnics	B	Physics	B	A C Theory	B	Basic
00/as/167	Year 1 Semester	Computer Litara C		Mechnics	B	Physics	C	A C Theory	B	Basic
00/as/32	Year 1 Semester	Computer Litara B		Mechnics	B	Physics	C	A C Theory	B	Basic
00/as/37	Year 1 Semester	Computer Litara B		Mechnics	B	Physics	C	A C Theory	B	Basic
00/as/45	Year 1 Semester	Computer Litara B		Mechnics	B	Physics	C	A C Theory	B	Basic
00/as/46	Year 1 Semester	Computer Litara D		Mechnics	B	Physics	C	A C Theory	B	Basic
00/as/47	Year 1 Semester	Computer Litara B		Mechnics	B	Physics	C	A C Theory	B	Basic
00/as/48	Year 1 Semester	Computer Litara B		Mechnics	B	Physics	C	A C Theory	B	Basic
00/as/49	Year 1 Semester	Computer Litara B		Mechnics	B	Physics	C	A C Theory	B	Basic
00/as/52	Year 1 Semester	Computer Litara D		Mechnics	B	Physics	C	A C Theory	B	Basic
00/as/53	Year 1 Semester	Computer Litara D		Mechnics	B	Physics	C	A C Theory	B	Basic
00/as/54	Year 1 Semester	Computer Litara D		Mechnics	B	Physics	C	A C Theory	B	Basic
00/as/55	Year 1 Semester	Computer Litara D		Mechnics	B	Physics	C	A C Theory	B	Basic
00/as/56	Year 1 Semester	Computer Litara D		Mechnics	B	Physics	C	A C Theory	B	Basic
00/as/57	Year 1 Semester	Computer Litara D		Mechnics	B	Physics	C	A C Theory	B	Basic
00/as/58	Year 1 Semester	Computer Litara D		Mechnics	B	Physics	C	A C Theory	B	Basic
00/as/59	Year 1 Semester	Computer Litara A		Mechnics	B	Physics	C	A C Theory	B	Basic
00/as/60	Year 1 Semester	Computer Litara A		Mechnics	B	Physics	C	A C Theory	B	Basic
00/as/61	Year 1 Semester	Computer Litara A		Mechnics	B	Physics	C	A C Theory	B	Basic
00/as/62	Year 1 Semester	Computer Litara A		Mechnics	B	Physics	C	A C Theory	B	Basic
00/as/63	Year 1 Semester	Computer Litara A		Mechnics	B	Physics	C	A C Theory	B	Basic
00/as/64	Year 1 Semester	Computer Litara A		Mechnics	B	Physics	C	A C Theory	D	Basic
00/as/65	Year 1 Semester	Computer Litara A		Mechnics	B	Physics	C	A C Theory	D	Basic
00/as/66	Year 1 Semester	Computer Litara A		Mechnics	B	Physics	C	A C Theory	D	Basic
00/as/67	Year 1 Semester	Computer Litara A		Mechnics	B	Physics	C	A C Theory	A	Basic
00/as/68	Year 1 Semester	Computer Litara A		Mechnics	B	Physics	C	A C Theory	C	Basic
00/as/69	Year 1 Semester	Computer Litara A		Mechnics	B	Physics	C	A C Theory	C	Basic
00/as/70	Year 1 Semester	Computer Litara A		Mechnics	B	Physics	C	A C Theory	C	Basic
00/as/71	Year 1 Semester	Computer Litara A		Mechnics	B	Physics	C	A C Theory	C	Basic

Record: 14 of 31

Datasheet View

Start | Cannot find s... | Thesis - Micro... | database | sab\_db : Data... | pers\_info : Ta... | result : Table | login\_info : Ta... | 5:43 PM

The Students Results Table

Microsoft Access - [login\_info : Table]

File Edit View Insert Format Records Tools Window Help

id number first\_name last\_name designation address e\_mail phone username password outhority

780591674	Vijekumara	Wijesinghe	Student	Beliatta	wdakumara@y	0777454731	wijesinghe	wijesinghe	no
803344264v	Malin	Samaranayake	Student	Malimbe	malin@yahoo.c	343543513	malin	malin	yes
asdf							aa	ss	no

Record: 14 of 3

Datasheet View

Start | Cannot find s... | Thesis - Micro... | database | sab\_db : Data... | pers\_info : Ta... | result : Table | login\_info : T... | 5:44 PM

The Logging Data Table

## Appendix-V

```
<%@LANGUAGE="VBSCRIPT" CODEPAGE="1252"%>
<!--#include file="Connections/sab_conn.asp" -->
<%
Dim rs_serch_food__MMColParam
rs_serch_food__MMColParam = "1"
If (Request.Form("reg_no") <> "") Then
    rs_serch_food__MMColParam = Request.Form("reg_no")
End If
%>
<%
Dim rs_serch_food
Dim rs_serch_food_numRows

Set rs_serch_food = Server.CreateObject("ADODB.Recordset")
rs_serch_food.ActiveConnection = MM_sab_conn_STRING
rs_serch_food.Source = "SELECT * FROM FS WHERE reg_no = " +
Replace(rs_serch_food__MMColParam, "'", "''") + "'"
rs_serch_food.CursorType = 0
rs_serch_food.CursorLocation = 2
rs_serch_food.LockType = 1
rs_serch_food.Open()

rs_serch_food_numRows = 0
%>
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>

<body>
</body>
</html>
<%
rs_serch_food.Close()
Set rs_serch_food = Nothing
%>
```

**National Digitization Project**

***National Science Foundation***

Institute : Sabaragamuwa University of Sri Lanka

1. Place of Scanning : Sabaragamuwa University of Sri Lanka, Belihuloya

2. Date Scanned : 2017-09-25.....

3. Name of Digitizing Company : Sanje (Private) Ltd, No 435/16, Kottawa Rd,  
Hokandara North, Arangala, Hokandara

4. Scanning Officer

Name : B.A.C. Badarwan.....

Signature : .....

Certification of Scanning

*I hereby certify that the scanning of this document was carried out under my supervision, according to the norms and standards of digital scanning accurately, also keeping with the originality of the original document to be accepted in a court of law.*

Certifying Officer

Designation : Librarian.....

Name : T. N. Neighsoorei.....

Signature : .....

Date : 2017-09-25.....

**Mrs. T. N. NEIGHSOOREI**  
(MSc, PhD, ASLA, BA)  
Librarian  
Sabaragamuwa University of Sri Lanka  
P.O. Box 02 Belihuloya, Sri Lanka  
Tel: 094 45 2280045  
Fax: 094 45 2280045

*"This document/publication was digitized under National Digitization Project of the National Science Foundation, Sri Lanka"*