**CHAPTER 1**

**1. INTRODUCTION**

**1.1 OVERVIEW OF THE PROJECT**

Now a day’s social networking sites are used by the people all around the world mainly to share photos and other status information. Most of the social networking sites have only specific features like photo/video sharing, chatting with friends. It does not have features to support students regarding their studies.

To overcome this type of situation Saveetha Social Networking has the features that support the studies of the student like E-Learning to share their study materials in a social networking environment and also contains same features provided in all social networking sites. It also contains Placement information for students.

By integrating the E-Learning and Placement information features into the social networking environment students can keep in touch with their and view the details regarding their placement and studies.

**1.2 LITERATURE SURVEY**

Social media has become a growing phenomenon with many and varied definitions in public and academic use. Most of the social networking site performs sharing of activities, photo, video etc.

In existing many social networking sites it does not have features that support students in their studies and placement.

In Saveetha Social Networking it has regular social networking site features and also features such as E-Learning support student during studies, Placement information support student placement preparation.

**1.3 EXISTING SYSTEM**

There are many social networking sites used by many people it gives common features like chatting, sharing activity & photo, upload photo etc. But it does not have features to support student studies like E-Learning and forum. Also it does not have feature to support student during placement.

**1.4 PROPOSED SYSTEM**

In this proposed system we develop a social networking site using the features

that mainly support students during their academics. By integrating features like E-Learning and Placement information with the social networking site it support student’s studies, can communicate their friends and keep in touch with college events during their academics.

**1.5 OBJECTIVES**

The main objective of the project is to establish a network among the students in saveetha engineering college. All the information can be accessed and shared among the students.

To provide the features that help student in their studies and placement are integrated with this website. They can send and receive messages with other students and can also communicate through chat with other students.

**1.6 ORGANISATION OF THE PROJECT**

The report has been organized in such a way that the flow of the process and modules of the application coincides with the text. The report is split into chapters that the website demands. It also includes the Object Oriented Analysis and Design (OOAD) diagrams associated with the project. The test plan deals with the different test cases the inputs, the actions and their corresponding outputs. Finally, the report is concluded and the future enhancements that can be done to the application are discussed.

**CHAPTER 2**

**REQUIREMENTS SPECIFICATION**

**2.1 INTRODUCTION**

This chapter discusses about the website perspective, functions, user characteristics and the operating environment. Also application specific requirements including the system features and data flow diagram are discussed. In addition performance requirements and software quality attributes are also discussed.

The project is mainly used for making an independent social networking site for the college.

**2.2 OVERALL DESCRIPTION**

In saveetha social networking site the main requirements are computer/laptop with internet connection then students can access it from anywhere.

**2.2.1 PRODUCT PERSPECTIVE**

This project is the college level project and is being implementing under the guidance of college professor. This website will provide features to support student in their studies and pacement. The objective is to develop and implement a website for the students of the saveetha engineering college. Providing the features that help students studies and placement.

**2.2.2 PRODUCT FUNCTION**

In our proposed system we presented the features that provide support to the students studies. It provides some features like E-Learning and Placement information helps during their studies and while preparing for campus interview.

**2.2.3 User Characteristics**

Users who are accessing this website must give their correct username, password and belong to that particular college. The students who are using this website can communicate with their friends and use the features like E-Learning and forum to support their studies.

**2.2.4. Operating Environment**

**Hardware used**

RAM : 1GB and Above

Processor : Intel Core 2 Duo (2 GHz )

Hard Disk : 20 GB

**Software used**

Operating System : Windows XP/Vista/7

Tools : Visual Studio 2010

Language : C# .NET

Database : SQL Server 2008

Documentation Tool : MS Word

**2.2.5 Constraints**

**2.2.5.1 Constraints in Analysis**

* Constraints as operational restrictions
* Constraints integrated in existing model concepts
* Constraints implied by the model structure

**2.2.5.2 Constraints in Design**

* Determination of the involved classes
* Determination of the require classes
* Determination of the involved actions

**2.3 SPECIFIC REQUIREMENTS**

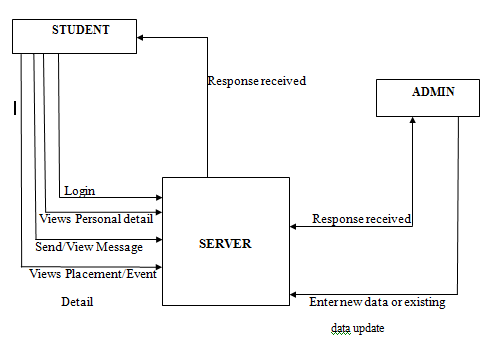
**2.3.1 External Interface Requirements**

**User Interfaces**

* Every conceptual part of the project is reflected using the C# .NET.
* All the contents in the project are implemented using ASP .NET using C#.

**2.3.2 System Architecture**

In this architecture it explains that how the students communicate each other and they receiving information from the administrator.

****

**2.3.3 Performance Requirements**

The website may be satisfy critical, if so, there are issues associated with its integrity level. Performance of the website is also depends on the internet speed. The system performance and system configuration also increase the performance of the website.

**Safety Requirements**

The website requires an active internet connection for using the website. In account of network failure, the website would not provide the proper communication. Hence, an active internet connection is required.

**Security Requirements**

The website is secure and the quality of the website is designed in such a way that it is user friendly and can be easily used by any user.

**2.3.4 Software Quality Attributes**

**Functionality**

Functionality is the required functions available including interoperability and security.

**Reliability**

It is the ability of the website to recover from the failure after detection. The website is prone to fault tolerance and recoverability.

**Reusability**

It is ability of the system to be reused. This website can be reused in various domains.

**Usability**

It is effectiveness of the use of system. The user interaction of the website is developed in such a way that the user can understand easily, learn at faster rate and operate with great level of comfort.

**Efficiency**

The project is having high performance and resource behaviour.

**Portability**

It is the ability of the system to adapt to different environment. This website is platform independent, it can be used in different operating environment and hence the system is operable.

**Integrity**

It is protection of the system from unauthorized access. This website provides a high degree of integrity.

**CHAPTER 3**

**SOFTWARE DESCRIPTION**

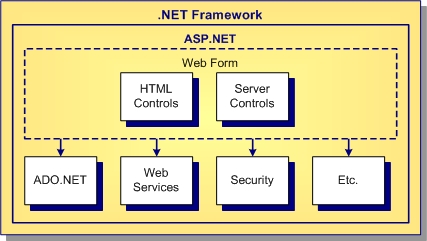
**3.1 FRONT END**

ASP.NET is a set of Web development tools offered by Microsoft. Programs like Visual Studio .NET and Visual Web Developer allow Web developers to create dynamic websites using a visual interface. Of course, programmers can write their own code and [scripts](http://www.techterms.com/definition/script) and incorporate it into ASP.NET websites as well. Though it often seen as a successor to Microsoft's [ASP](http://www.techterms.com/definition/asp) programming technology, ASP.NET also supports Visual Basic.NET, JScript .NET and open-source languages like Python and Perl.

ASP.NET is built on the .NET framework, which provides an application program interface ([API](http://www.techterms.com/definition/api)) for software programmers. The .NET development tools can be used to create applications for both the Windows operating system and the Web. Programs like Visual Studio .NET provide a visual interface for developers to create their applications, which makes .NET a reasonable choice for designing Web-based interfaces as well.

In order for an ASP.NET website to function correctly, it must be published to a Web server that supports ASP.NET applications. Microsoft's Internet Information Services (IIS) Web server is by far the most common platform for ASP.NET websites. While there are some open-source options available for Linux-based systems, these alternatives often provide less than full support for ASP.NET applications.

**Architecture Diagram**

****

The major components of the ASP.NET architecture are Web Forms, ASP.NET server controls, code-behind logic files, and compiled DLL files. Web Form pages contain HTML elements, text, and server controls. Code-behind files contain application logic for the Forms page. Compiled DLL files render dynamic HTML on the web server.

CodeGear provides tools to simplify ASP.NET development. If you are familiar with rapid application development (RAD) and object oriented programming (OOP) using properties, methods, and events, you will find the ASP.NET model for building Web applications familiar.

**Performance**

ASP.NET aims for performance benefits over other script-based technologies

by compiling the server-side code to one or more [DLL](https://en.wikipedia.org/wiki/Dynamic_link_library) [files](https://en.wikipedia.org/wiki/Computer_file) on the [Web server](https://en.wikipedia.org/wiki/Web_server). This compilation happens automatically the first time a page is requested. This feature provides the ease of development offered by scripting languages with the performance benefits of a compiled binary. However, the compilation might cause a noticeable but short delay to the Web user when the newly edited page is first requested from the Web server, but will not again unless the page requested is updated further.

The ASPX and other resource files are placed in a virtual host on an [Internet Information Services](https://en.wikipedia.org/wiki/Internet_Information_Services) server. The first time a client requests a page, the .NET Framework parses and compiles the files into a .NET assembly and sends the response; subsequent requests are served from the DLL files. By default ASP.NET will compile the entire site in batches of 1000 files upon first request. If the compilation delay is causing problems, the batch size or the compilation strategy may be tweaked.

Developers can also choose to pre-compile their "codebehind" files before deployment, using MS Visual Studio, eliminating the need for [just-in-time compilation](https://en.wikipedia.org/wiki/Just-in-time_compilation) in a production environment. This also eliminates the need of having the source code on the Web server. It also supports pre-compile text.

## Frameworks

It is not essential to use the standard Web forms development model when developing with ASP.NET. Noteworthy frameworks designed for the platform include:

* [Base One Foundation Component Library](https://en.wikipedia.org/wiki/Base_One_Foundation_Component_Library) (BFC) is a [RAD](https://en.wikipedia.org/wiki/Rapid_application_development) framework for building .NET [database](https://en.wikipedia.org/wiki/Database) and [distributed computing](https://en.wikipedia.org/wiki/Distributed_computing) applications.
* [DotNetNuke](https://en.wikipedia.org/wiki/DotNetNuke) is an open-source solution which comprises both a web application framework and a content management system which allows for advanced extensibility through modules, skins, and providers.
* [Castle MonoRail](https://en.wikipedia.org/wiki/MonoRail_(software)), an open-source [MVC](https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93controller) framework with an execution model similar to [Ruby on Rails](https://en.wikipedia.org/wiki/Ruby_on_Rails). The framework is commonly used with [Castle ActiveRecord](https://en.wikipedia.org/wiki/Castle_ActiveRecord), an ORM layer built on [NHibernate](https://en.wikipedia.org/wiki/NHibernate).
* [Spring.NET](http://www.springframework.net), a port of the [Spring framework](https://en.wikipedia.org/wiki/Spring_framework) for Java.
* [Survey Project](https://en.wikipedia.org/w/index.php?title=Survey_Project&action=edit&redlink=1) is an open-source web-based survey and form engine framework written in ASP.NET and C#.
* [Carbon MVVM](http://carbonmvvm.codeplex.com) is an open-source MVVM framework based on ASP.NET

**3.2 FEATURES**

* [**Web Sockets**](http://www.techrepublic.com/blog/programming-and-development/websocket-offers-a-new-approach-to-web-development/5707)

Full support for the Web Sockets [HTML5](http://www.techrepublic.com/blog/webmaster/the-case-for-html5-five-examples/2328) standard is available with ASP.NET 4.5 running on IIS 8.0 via the [SignalR](http://www.asp.net/signalr) library. This allows you to easily add real-time Web functionality to applications.

* **Authentication**

There is now a universal provider (DefaultMembershipProvider) for simplification. In addition, the [OAuth](http://www.techrepublic.com/blog/programming-and-development/oauth-protocol-lets-apps-authenticate-against-services/3888) protocol is embraced.

* **Async programming**

While this feature is not ASP.NET-specific, it is worth noting that C# 5 and Visual Basic 11 provide async support without using multiple threads (via async and await keywords).

* **Web publishing**

This feature has been enhanced, whereas you can compare local and remotes files, publish only selected files, and so forth.

* **Web API**

This API provides the [REST](http://www.techrepublic.com/blog/programming-and-development/poll-do-you-work-with-soap-or-rest-services/6258) approach to building applications — a key difference from the WCF alternative. In addition, the Web API now includes extensive [OData](http://www.techrepublic.com/blog/programming-and-development/odata-standardizes-data-exchange-on-the-web/6124) support, which is another instance of Microsoft embracing open source standards.

* **Friendly URLs**

The popularity of [tinyurl](http://tinyurl.com/) demonstrates the widespread problem of keeping up with long, arcane URLs. This feature is now available with ASP.NET applications via the [FriendlyURLs](http://www.hanselman.com/blog/IntroducingASPNETFriendlyUrlsCleanerURLsEasierRoutingAndMobileViewsForASPNETWebForms.aspx) feature.

* **Mobile**

The explosion of Smartphone, E-Readers and other mobile devices has changed the landscape for Web application development. HTML5 support is supposed to simplify mobile application development. In addition, MVC 4 includes mobile templates, and there are a variety of mobile device emulators that can be used in [Visual Studio 2012](http://www.techrepublic.com/blog/programming-and-development/what-visual-studio-2012-and-net-45-offer-developers/5747).

* [**IIS**](http://www.iis.net/)

This allows you to use new features available in Internet Information Server (IIS) 8.0. Some of these features include prefetching and application initialization like application ping on startup. Also, an express edition of IIS is now available.

* **Master Pages**

A *master page* is a page on which you can define elements that you want to use in other pages on your Web site. Content pages can reference a master page and automatically have those elements provided on the page. This provides a way to give Web sites a uniform appearance.

* **Deprecated Features**

The DB Web controls in the *Borland.Data.Web* namespace have been deprecated. The ASP.NET 2.0 Web controls offer equivalent capability to the DB Web controls.

**CHAPTER 4**

**DETAILED DESIGN AND TEST PLAN**

**4.1 DECOMPOSITION DESCRIPTION**

There are various modules present in our project everything in the project can’t be done at the same time hence we have decomposed our project into some of the modules. Decomposition gives us an easier way to solve any type of problems. It has a power to convert huge things into number of smaller ones. Therefore the work stress will be reduced and much possibility to understand the project very easily.

**4.1.1 Module Decomposition**

Modules in the sense it is the part of the project. Module decomposition is that dividing the work into smaller parts such a way to make our work easier to understand and face the problem that occurs while handling the project work. So based up on the project module description and decomposition had been done.

**4.2 DEPENDENCY DESCRIPTION**

**4.2.1 Inter Module Dependency**

The website is divided into three modules namely:

1. Admin
2. User
3. Chat

User module takes input from the student if the username and password is valid then students can access their personal information, college information, placement information is uploaded by the admin module.

Admin module helps the administrator of the website to upload the college events information, placement information is uploaded to the user module. In chat module student can chat with their classmates and other students studying in the same college. The chat is enabled after the valid login by the student in user module.

**4.2.2 Data Dependency**

The data which are using in out project depends on relational database contents. Each and every data are stored in database in form of tables. So used can store and retrieve data from the database which is stored in the form of tables.

**4.3 DETAILED DESIGN**

In our project modules had been designed based upon its work. The design of our project is that to build a new architecture for the social networking site.

**Module 1: Admin**

This module helps the administrator to upload the information regarding college events, placement information, managing student information and also provides students study materials. Also admin can get the student feedback regarding the website usage and its features.

**Module 2: User**

This module helps students to access their information by using valid username and password. Students can know about the latest information and events happening in the college. Students can upload photos in their profile, can download books from the E-Learning page and can discuss about the topic regarding the studies.

**Module 3: Chat**

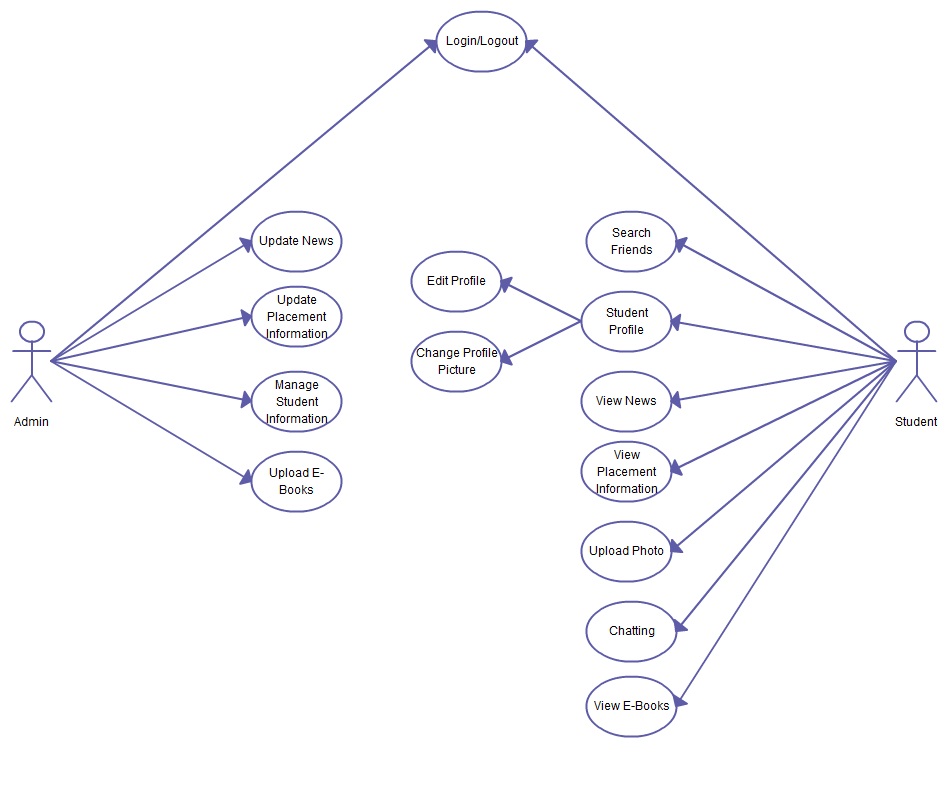
This module helps the students to communicate with their friends studying in the same college. Through this feature students can communicate with each other in a quick way. Student can communicate with the others only if they are in online.

**4.4 OOAD DIAGRAMS**

The typical object oriented analysis and design for the project description can be shown according to the work we have done on the project. The OOAD diagram describes about the data flow within the system. The OOAD diagram consists of the following categories use-case diagram, activity diagram, sequence diagram and collaboration diagram.

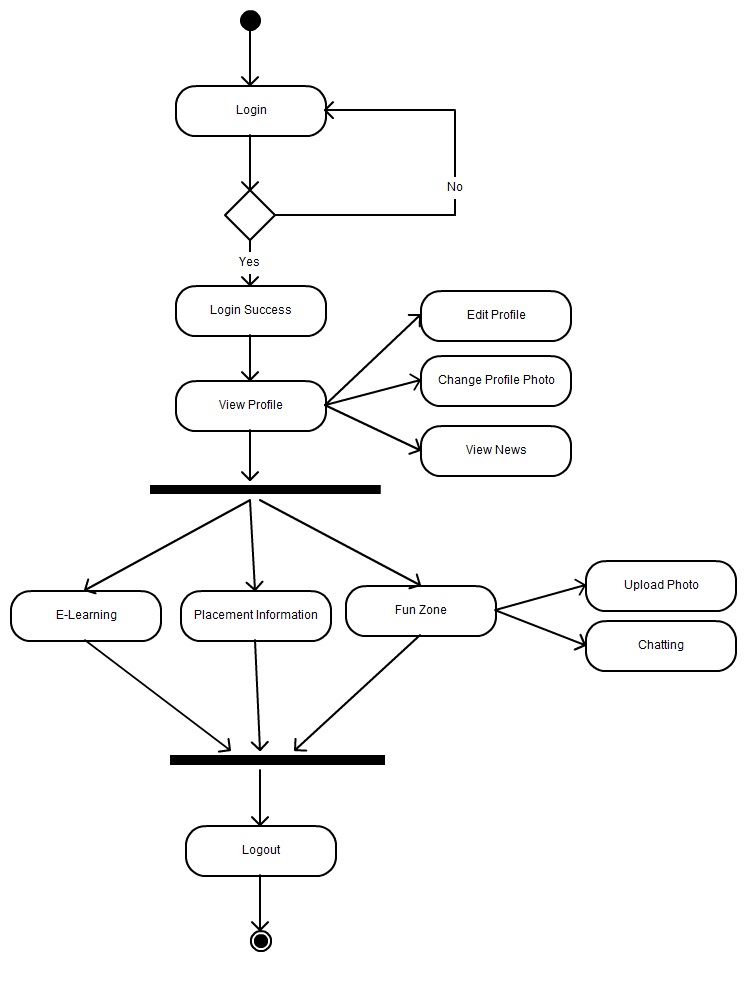
**4.4.1 Use-case Diagram**

A use case diagram is a graph of actor, a set of use case enclosed by a system boundary, a communication associated with actor and the use case and generalization among use case.



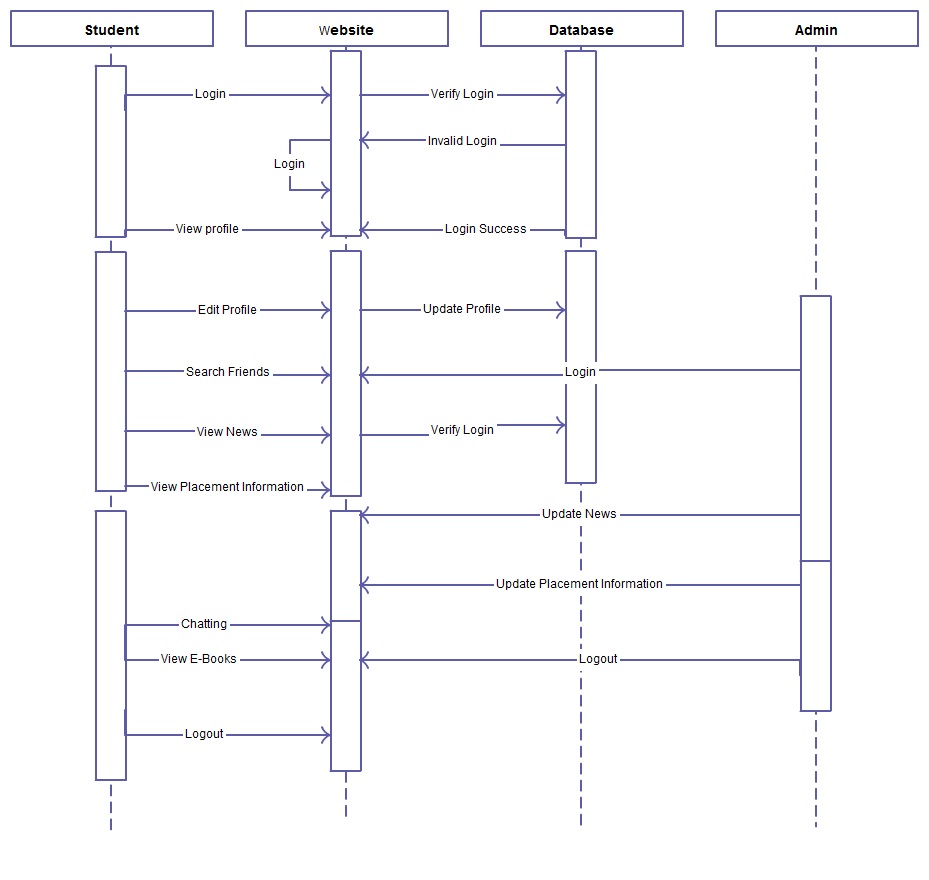
**4.4.2 Activity diagram**

An activity diagram is a variation or special case of a state machine in which the state are the activities representing the performance of the operations and the transactions are triggered by the completion of project.



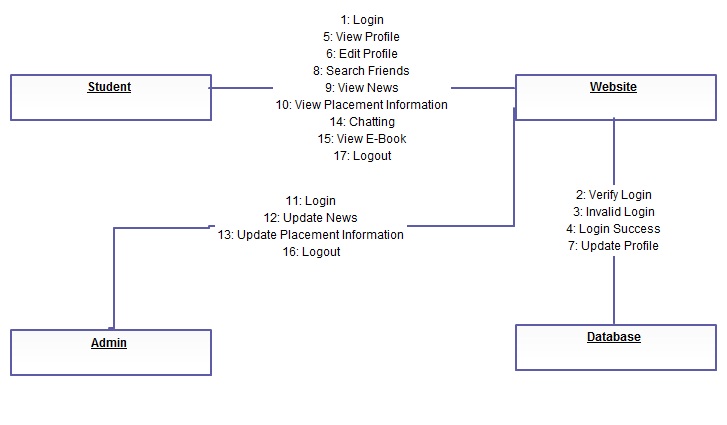
**4.4.3 Sequence Diagram**

A sequence diagram describes the behaviour of the system by viewing the interaction between the system and the environment.



**4.4.4 Collaboration Diagram**

The collaboration diagram represents the collaboration which is a set of objects related in a particular context and interaction which is the set of messages exchanged among the objects within the collaboration to achieve desired outcome.



**4.5 TEST PLAN**

**TESTING:**

The project is tested to verify its correctness and efficiency. The test plan includes following test cases:

|  |  |  |
| --- | --- | --- |
| **Test Case** | **Expected Output** | **Actual Output** |
| Login Session | Proper user should be allowed to access the pages | The user can access the pages when they give the proper details. |
| Profile | In this the user profile must be displayed correctly | The user details is been displayed by fetching the data from the database |
| Chat | The user needs to chat with the other user | The user can chat with the other user but it is not secure enough |
| Photo upload | In this the user should upload the photos | The user can upload the photo |
| Games | In this the user must try to play flash games in this site | The user can play games in flash using the website |
| Discussion Forum | In this the user must be allowed to post there query and respond to the query | The user allowed to post and reply to the query |
| Search | In this the user is allowed to search for user | The user can search based on the details they have. |
| Message | In this the user must send message from one user to the another user | The user can send messages |
| Birthday | In this the birthday wishes is to be send to the users | User can send birthday wishes on their birthday |
| CGPA | In the CGPA must be calculated | The user can calculate their CGPA by providing the details. |
| GPA | GPA must be calculated | The user can calculate the GPA. |

**CHAPTER 5**

**IMPLEMENTATION AND RESULTS**

**5.1 IMPLEMENTATION**

It involves implementation of three phases Admin Module, User Module and Chat Module.

Implementation is the stage of the project when the theoretical design is turned out into working system. This Admin Module phase helps administrator to manage student information, uploads information about college events and placement information.

This User Module phase helps the students to access their information through valid login. Student can view information about college events and placement information.

Chat Module phase helps the students to chat with their friends studying in the same college.

**5.2 RESULT**

We have implemented and evaluated our website as well as the performance and security of the website is evaluated. In the experiment our website considerably takes reduced time to load the page. We have implemented all the features required for a social networking site and features support student studies and placement during academics.

**CHAPTER 6**

**TESTING AND MAINTANENCE**

**6.1 TESTING**

The development of a software system involves a series of production activities where opportunities for injection of human fallibilities are enormous. Errors may begin to occur at the very inception of the process where the objectives may be erroneously or imperfectly specified, as well as errors that occur in later design and development stages.

Software Testing is a critical element of software quality assurance and represents the ultimate review of specification, design and coding. Testing is a set of activities that can be planned in advance and conducted systematically. For this reason a template for software testing – a set of steps into which we can place specific test case design techniques and testing methods-should be designed for the software engineering process.

The following are the testing objectives:

* Testing is a process of executing a program with the intent of finding an error.
* A good test case is one that has a high probability of finding an as yet and discovered error.
* A successful test is one that uncovers an as yet undiscovered error.

If testing is conducted successfully it will uncover errors in the software. As a secondary benefit, testing demonstrates that software functions appear to be working according to specification, that performance requirements appear to have been met. In addition, data collected as testing is conducted provide a good indication of software reliability and some indication of software quality as a whole. Testing cannot show the absence of errors, it can only show that software defects are present.

The following deals with the Test Case Design:

The design of tests for software and other engineered products can be as challenging as the initial design of the design of the product itself. We must design tests that have the highest likelihood of finding the most errors with the minimum amount of the time and effort.

Any engineered product can be tested in one of the two ways:

* Knowing the specified function that a product has been designed to perform, tests can be conducted that demonstrates each function fully operational- BLACK BOX TESTING.
* Knowing the internal workings of a product, tests can be conducted to ensure that “all the gears mesh”, that is, that the internal operations of the product performs according to specifications and all internal components have been adequately exercised- WHITE BOX TESTING.

Although designed to uncover errors, black box tests are used to demonstrate the software functions are operational; that input is properly accepted, and output is correctly produced; that the integrity of the external information is maintained. A black box test examines some aspect of a system with little regard for the internal logical structure of the software.

White box testing of the software is predicted on a close examination of the procedural detail. The “status of the program” may be examined at various points to determine if the expected or asserted status corresponds to the actual status.

The test plan describes the overall strategy for integration. Testing is divided into phases and builds that address specific functional and characteristics of the software. Each of the phases and sub-phases delineates a broad functional category within the software and can generally be related to a specific domain of the program structure. Therefore, program builds are created to correspond to each phase.

A schedule for integration, (i.e.) the start and end dates for each phase is established and availability windows for unit-tested modules are defined.

The Testing Process is as follows:

Software testing is one element of a broader topic that is often referred to as verification and validation.

Verification- “Are we building the product right?”

Validation - “Are we building the right product?”

Verification refers to the set of activities that ensure that software correctly implements a specific function. Validation refers to a different set of activities that the software that has been built is traceable to member requirements.

**UNIT TESTING**

Unit testing focuses verification effort on the smallest unit of the software design the module. The relative complexity of the tests and the errors detected as a result is limited by the constrained scope established for unit testing. The unit test is always white-box oriented, and the step can be conducted in parallel for multiple modules.

The module interface is tested to ensure that information properly flows into and out of the program unit under test. The local data structure is examined to ensure that data stored temporarily maintains its integrity during all steps in an algorithms execution. Boundary conditions are tested to ensure that the module operates properly at the boundaries established to limit or restrict processing. All independent paths through the control structures are exercised to ensure that all statements in a module have been executed at least once. And finally, all error-handling paths are tested.

Because a module is not a stand-alone program, a driver and/or stub software must be developed for each unit test. In most applications a driver is nothing more than a “main program” that accepts test case data to the module to be tested, and prints the relevant results. Stubs serve to replace modules interface, may do minimal data manipulation, prints verification of entry, and returns.

Unit testing for our project was conducted in full. The basic modules were further subdivided into sub-modules and unit testing was conducted on each such module. The modules were tested for boundary conditions, independence paths, error handling paths and presentation of exact for the given input.

**INTEGRATION TESTING**

A neophyte in the software world might ask a seemingly legitimate question one all modules have been unit tested: If they all work individually, why do you doubt that they will work when we put them together? The problem, of course, is “putting them together “-interfacing.

Data can be lost across the interface; one module can have an inadvertent, adverse affect on another sub-function when combined, may not produce desired major functions, etc. Integration testing is a systematic technique tests to uncover errors associated with interfacing.

Top-Down Integration is an incremental approach to the construction of program structure. Modules are integrated by moving downward through the control hierarchy, beginning with the main control module. Modules subordinate to the main control module are incorporated into the structure in either a depth-first manner.

Bottom-Up Integration testing, as its name itself implies, begins construction and testing with the atomic modules. Because modules are integrated from the bottom up, processing required for modules subordinate to a give level is always available and the need for stubs is eliminated.

Integration testing for this project was carried out using the Bottom-up integration-testing model. The atomic modules were first tested unit-wise for output. Then the modules were integration with the index page of the website to test if the session is being tracked with consistency.

**SYSTEM TESTING**

System testing is actually a series of different tests whose primary purpose is to fully exercise the computer-based system. Although each test has a different purpose, all work should verify that all system elements have been properly integration and perform allocated functions.

* Recovery Testing is a system test that forces the software to fail in a variety of ways and verifies that recovery is properly performed
* Security Testing attempts to verify that protection mechanism built into a system will, in fact, project it from improper penetration
* Stress Testing is designed to confront programs with abnormal situation
* Performance Testing is designed to test the run-time

Performance of software within the context of integration

System

**6.2 MAINTENANCE**

In the proposed architecture, enterprises declared the purpose for which the data are collected, who can receive them, the length of time the data can be retained, and the authorized users who can access them. Hippocratic databases also created a privacy authorization table shared by all customers, but it does not allow distinguishing which particular method is used for fulfilling a service. Moreover, enterprises are able to provide their services in different ways, and each different method may require different data. Depending on the different kinds of methods, customers should provide different personal information. Asking for all personal information for different service methods as compulsory would clearly violate the principle of minimal disclosure.

**CHAPTER 7**

**CONCLUSION AND FUTUREWORK**

**7.1 CONCLUSION**

This project is designed to meet requirements of a social networking site. It has developed in ASP .NET using C# keeping in mind the specification of the website. For designing the website we have developed simple architecture diagram. The website takes less time to load in any system with different browser. The performance and security of the website are verified that it will give better security and performance under any instance. Features that are required for the social networking site and feature that will support students in their studies and placement during their academics.

**7.2 FUTURE WORK**

We are planning some additional features to the website. The website would be better than now. Enhance the E-Learning feature by adding video tutorial through which learning become easier for the students. Add placement training options so students can train for their placement test process.

**APPENDIX A**

**SAMPLE CODING**

**Login.aspx**

<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Login.aspx.cs" Inherits="Login" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">

<head runat="server"><title>Saveetha Social Networking Login Page</title>

</head>

<body background="Images/loginbackground.jpg">

<form id="form1" runat="server">

<div style="color: #00CCFF">

<asp:Image ID="Image1" runat="server"

style="z-index: 1; left: -44px; top: 0px; position: absolute" />

<asp:Label ID="Label2" runat="server" ForeColor="#FFFFCC"

style="z-index: 1; left: 810px; top: 341px; position: absolute"

Text="Password"></asp:Label>

<asp:Label ID="Label1" runat="server" ForeColor="#FFFFCC"

style="z-index: 1; left: 809px; top: 307px; position: absolute"

Text="User Name"></asp:Label>

<asp:LinkButton ID="LinkButton1" runat="server" Font-Size="Small"

ForeColor="White"

style="z-index: 1; left: 785px; top: 391px; position: absolute"

onclick="LinkButton1\_Click">Forgot your password? </asp:LinkButton>

<asp:TextBox ID="TextBox1" runat="server"

style="z-index: 1; left: 904px; top: 305px; position: absolute; width: 150px;"></asp:TextBox>

<asp:TextBox ID="TextBox2" runat="server"

style="z-index: 1; left: 903px; top: 344px; position: absolute; height: 17px; width: 151px;"

TextMode="Password"></asp:TextBox>

<asp:ImageButton ID="ImageButton1" runat="server"

ImageUrl="Images/loginbutton.jpg"

style="z-index: 1; left: 986px; top: 389px; position: absolute"

onclick="ImageButton1\_Click" />

<asp:Image ID="Image2" runat="server" ImageUrl="Images/memberlogin.jpg"

style="z-index: 1; left: 792px; top: 258px; position: absolute" />

<asp:Label ID="Label8" runat="server" Font-Bold="True" ForeColor="White"

style="z-index: 1; left: 809px; top: 368px; position: absolute; height: 18px; width: 271px"

Text="User name or Password does not match"></asp:Label>

</div>

</form>

</body>

</html>

**Login.aspx.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Configuration;

using System.Data.SqlClient;

public partial class Login : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

Label8.Visible = false;

}

protected void ImageButton1\_Click(object sender, ImageClickEventArgs e)

{

AppSettingsReader asr = new AppSettingsReader();

string conn = asr.GetValue("ConnectionString", typeof(System.String)).ToString();

try

{

SqlConnection sc = new SqlConnection(conn);

sc.Open();

string cmdstr = "SELECT username,password,role FROM login where username='" + TextBox1.Text +"' and password='" + TextBox2.Text + "'";

SqlCommand cmd = new SqlCommand(cmdstr, sc);

SqlDataReader rd = cmd.ExecuteReader();

if (rd.Read())

{

Session["UserID"] = Convert.ToString(rd[0]);

Session["RoleId"] = Convert.ToString(rd[2]);

Response.Redirect("MyAccount.aspx");

}

else

{

Label8.Visible = true;

}

sc.Close();

}

catch (Exception ex)

{

Console.WriteLine(ex.ToString());

}

}

protected void LinkButton1\_Click(object sender, EventArgs e)

{

Response.Redirect("Forgotpassword.aspx");

}

}

**Profile.aspx**

<%@ Page Title="Saveetha Social Networking" Language="C#" MasterPageFile="~/Master.master" AutoEventWireup="true" CodeFile="Profile.aspx.cs" Inherits="MyProfile" %>

<asp:Content ID="Content1" ContentPlaceHolderID="head" Runat="Server">

</asp:Content>

<asp:Content ID="Content2" ContentPlaceHolderID="ContentPlaceHolder1" Runat="Server">

<p>

<asp:Label ID="Label36" runat="server" Font-Bold="False" ForeColor="#0066CC"

style="z-index: 4; left: 155px; top: 351px; position: absolute; width: 109px; height: 33px"

Text="Label"></asp:Label>

<asp:Label ID="Label37" runat="server" Font-Bold="False" ForeColor="#0066CC"

style="z-index: 4; left: 153px; top: 384px; position: absolute; width: 145px; height: 19px"

Text="Label"></asp:Label>

<asp:LinkButton ID="LinkButton8" runat="server" Font-Size="Medium"

ForeColor="#006699"

style="z-index: 2; left: 178px; top: 573px; position: absolute; width: 117px"

PostBackUrl="~/Birthday.aspx" Font-Names="Bell MT">BirthDay Cards</asp:LinkButton>

<asp:LinkButton ID="LinkButton7" runat="server" Font-Size="Medium"

ForeColor="#006699"

style="z-index: 2; left: 178px; top: 547px; position: absolute; width: 100px"

PostBackUrl="~/Friends.aspx" Font-Names="Bell MT">Friends</asp:LinkButton>

<asp:LinkButton ID="LinkButton6" runat="server" Font-Size="Medium"

ForeColor="#006699"

style="z-index: 2; left: 178px; top: 521px; position: absolute; width: 100px"

PostBackUrl="~/Classmates.aspx" Font-Names="Bell MT">ClassBuddies</asp:LinkButton>

<asp:Image ID="Image3" runat="server" ImageUrl="~/Images/defaultimg.jpg"

style="z-index: 4; left: 158px; top: 223px; position: absolute; height: 127px; width: 129px" />

<br />

<asp:Image ID="Image4" runat="server" ImageUrl="~/images/myprofile.jpg"

style="z-index: 1; left: 136px; top: 418px; position: absolute; height: 312px"

Width="173px" />

<asp:LinkButton ID="LinkButton4" runat="server" Font-Size="Medium"

ForeColor="#006699"

style="z-index: 2; left: 177px; top: 599px; position: absolute; width: 90px; height: 19px"

PostBackUrl="~/UserSettings.aspx" Font-Names="Bell MT">Settings</asp:LinkButton>

<asp:LinkButton ID="LinkButton2" runat="server" Font-Size="Medium"

ForeColor="#006699"

style="z-index: 2; left: 179px; top: 496px; position: absolute; width: 90px; height: 19px"

Font-Names="Bell MT" PostBackUrl="~/Messages.aspx">Messages</asp:LinkButton>

<asp:LinkButton ID="LinkButton5" runat="server" Font-Size="Medium"

ForeColor="#006699"

style="z-index: 2; left: 180px; top: 470px; position: absolute; width: 90px"

PostBackUrl="~/MyAccount.aspx" Font-Names="Bell MT">My Account</asp:LinkButton>

<asp:Image ID="Image6" runat="server" ImageUrl="~/images/imgbg.jpg"

style="z-index: 2; left: 142px; top: 217px; position: absolute; width: 162px; height: 199px" />

</asp:Content>

<asp:Content ID="Content3" ContentPlaceHolderID="ContentPlaceHolder2" Runat="Server">

<p>

<br />

<asp:Image ID="Image1" runat="server" ImageUrl="~/images/profile\_main.jpg"

style="z-index: 1; left: 316px; top: 220px; position: absolute; height: 338px; width: 654px" />

<asp:Label ID="Label35" runat="server"

style="z-index: 1; left: 345px; top: 466px; position: absolute; width: 371px; height: 86px;"

Text="About Me:" Font-Names="Bell MT" ForeColor="#006699"></asp:Label>

<asp:Label ID="Label10" runat="server"

style="z-index: 1; left: 345px; top: 392px; position: absolute; width: 141px"

Text="Label" Font-Names="Bell MT" ForeColor="#006699"></asp:Label>

<asp:Label ID="Label6" runat="server"

style="z-index: 1; left: 345px; top: 361px; position: absolute; width: 141px"

Text="Label" Font-Names="Bell MT" ForeColor="#006699"></asp:Label>

<asp:Label ID="Label5" runat="server"

style="z-index: 1; left: 345px; top: 329px; position: absolute; width: 141px"

Text="Label" Font-Names="Bell MT" ForeColor="#006699"></asp:Label>

<asp:Label ID="Label4" runat="server"

style="z-index: 1; left: 345px; top: 307px; position: absolute; width: 141px"

Text="Label" Font-Names="Bell MT" ForeColor="#006699"></asp:Label>

<asp:Label ID="Label3" runat="server"

style="z-index: 1; left: 345px; top: 274px; position: absolute; width: 141px"

Text="Label" Font-Names="Bell MT" ForeColor="#006699"></asp:Label>

<asp:Button ID="Button1" runat="server" BackColor="#FDB349"

BorderColor="#FFB451" Font-Bold="True" ForeColor="Black"

style="z-index: 2; top: 307px; left: 745px; position: absolute; height: 26px; width: 93px"

Text="Edit Profile" PostBackUrl="~/ProfileEdit.aspx" />

<asp:Label ID="Label2" runat="server" Font-Bold="False" Font-Size="Medium"

Font-Underline="True" ForeColor="Black"

style="z-index: 2; left: 751px; top: 247px; position: absolute; height: 22px; width: 195px"

Text="Label" Font-Names="Bell MT"></asp:Label>

<asp:Label ID="Label9" runat="server" Font-Bold="True" ForeColor="#0099CC"

style="z-index: 2; left: 767px; top: 437px; position: absolute; height: 33px; width: 159px"

Text="Label"></asp:Label>

<asp:Label ID="Label8" runat="server" Font-Bold="True" ForeColor="#0099CC"

style="z-index: 2; left: 767px; top: 389px; position: absolute; height: 33px; width: 159px"

Text="Label"></asp:Label>

<asp:Label ID="Label7" runat="server" Font-Bold="True" ForeColor="#0099CC"

style="z-index: 2; left: 767px; top: 358px; position: absolute; height: 33px; width: 159px"

Text="Label"></asp:Label>

<asp:Label ID="Label1" runat="server" Font-Bold="True" ForeColor="#0099CC"

style="z-index: 2; left: 344px; top: 248px; position: absolute; height: 20px; width: 382px"

Text="Label"></asp:Label>

<asp:Label ID="Label30" runat="server" Font-Bold="True" Font-Names="Bell MT"

Font-Size="Medium" ForeColor="#006699"

style="z-index: 1; left: 425px; top: 783px; position: absolute; height: 19px"

Text="Dream Company"></asp:Label>

<asp:Label ID="Label28" runat="server" Font-Bold="True" Font-Names="Bell MT"

Font-Size="Medium" ForeColor="#006699"

style="z-index: 1; left: 425px; top: 936px; position: absolute; height: 19px"

Text="Hobbies"></asp:Label>

<asp:Label ID="Label27" runat="server" Font-Bold="True" Font-Names="Bell MT"

Font-Size="Medium" ForeColor="#006699"

style="z-index: 1; left: 425px; top: 898px; position: absolute; height: 19px"

Text="Pets"></asp:Label>

<asp:Label ID="Label23" runat="server" Font-Bold="True" Font-Names="Bell MT"

Font-Size="Medium" ForeColor="#006699"

style="z-index: 1; left: 425px; top: 822px; position: absolute; height: 19px"

Text="Mobile Phone"></asp:Label>

<asp:Label ID="Label22" runat="server" Font-Bold="True" Font-Names="Bell MT"

Font-Size="Medium" ForeColor="#006699"

style="z-index: 1; left: 425px; top: 859px; position: absolute; height: 19px"

Text="Home Phone"></asp:Label>

<asp:Label ID="Label20" runat="server" Font-Bold="True" Font-Names="Bell MT"

Font-Size="Medium" ForeColor="#006699"

style="z-index: 1; left: 425px; top: 738px; position: absolute; height: 19px"

Text="Email"></asp:Label>

<asp:Label ID="Label18" runat="server" Font-Bold="True" Font-Names="Bell MT"

Font-Size="Medium" ForeColor="#006699"

style="z-index: 1; left: 425px; top: 700px; position: absolute; height: 19px"

Text="Sports"></asp:Label>

<asp:Label ID="Label16" runat="server" Font-Bold="True" Font-Names="Bell MT"

Font-Size="Medium" ForeColor="#006699"

style="z-index: 1; left: 425px; top: 662px; position: absolute; height: 19px"

Text="Age"></asp:Label>

<asp:Label ID="Label14" runat="server" Font-Bold="True" Font-Names="Bell MT"

Font-Size="Medium" ForeColor="#006699"

style="z-index: 1; left: 425px; top: 625px; position: absolute; height: 19px"

Text="Here For"></asp:Label>

<asp:Label ID="Label26" runat="server" Font-Bold="False" Font-Names="Bell MT"

Font-Size="Medium" ForeColor="#006699"

style="z-index: 1; left: 625px; top: 823px; position: absolute; height: 19px"

Text="Label"></asp:Label>

<asp:Label ID="Label25" runat="server" Font-Bold="False" Font-Names="Bell MT"

Font-Size="Medium" ForeColor="#006699"

style="z-index: 1; left: 625px; top: 858px; position: absolute; height: 19px"

Text="Label"></asp:Label>

<asp:Label ID="Label34" runat="server" Font-Bold="False" Font-Names="Bell MT"

Font-Size="Medium" ForeColor="#006699"

style="z-index: 1; left: 625px; top: 935px; position: absolute; height: 19px"

Text="Label"></asp:Label>

<asp:Label ID="Label33" runat="server" Font-Bold="False" Font-Names="Bell MT"

Font-Size="Medium" ForeColor="#006699"

style="z-index: 1; left: 625px; top: 897px; position: absolute; height: 19px"

Text="Label"></asp:Label>

<asp:Label ID="Label32" runat="server" Font-Bold="False" Font-Names="Bell MT"

Font-Size="Medium" ForeColor="#006699"

style="z-index: 1; left: 625px; top: 783px; position: absolute; height: 19px"

Text="Label"></asp:Label>

<asp:Label ID="Label24" runat="server" Font-Bold="False" Font-Names="Bell MT"

Font-Size="Medium" ForeColor="#006699"

style="z-index: 1; left: 625px; top: 737px; position: absolute; height: 19px"

Text="Label"></asp:Label>

<asp:Label ID="Label19" runat="server" Font-Bold="False" Font-Names="Bell MT"

Font-Size="Medium" ForeColor="#006699"

style="z-index: 1; left: 625px; top: 699px; position: absolute; height: 19px"

Text="Label"></asp:Label>

<asp:Label ID="Label17" runat="server" Font-Bold="False" Font-Names="Bell MT"

Font-Size="Medium" ForeColor="#006699"

style="z-index: 1; left: 625px; top: 661px; position: absolute; height: 19px"

Text="Label"></asp:Label>

<asp:Label ID="Label15" runat="server" Font-Bold="False" Font-Names="Bell MT"

Font-Size="Medium" ForeColor="#006699"

style="z-index: 1; left: 625px; top: 623px; position: absolute; height: 19px; width: 374px"

Text="Friends, Classmates, Gangs,Social Networking,Carrier"></asp:Label>

<asp:Label ID="Label12" runat="server" Font-Bold="False" Font-Names="Bell MT"

Font-Size="Medium" ForeColor="#006699"

style="z-index: 1; left: 625px; top: 587px; position: absolute; height: 19px"

Text="Label"></asp:Label>

<asp:Label ID="Label11" runat="server" Font-Bold="True" Font-Names="Bell MT"

Font-Size="Medium" ForeColor="#006699"

style="z-index: 1; left: 425px; top: 587px; position: absolute; height: 19px"

Text="Birthday"></asp:Label>

</asp:Content>

**Profile.aspx.cs**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Web;

using System.Web.UI;

using System.Web.UI.WebControls;

using System.Configuration;

using System.Data.SqlClient;

public partial class MyProfile : System.Web.UI.Page

{

string user;

protected void Page\_Load(object sender, EventArgs e)

{

//Label35.Visible = false;

//FileUpload1.Visible = false;

//Button2.Visible = false;

object us = Session["UserID"];

if (Session["UserID"] == null || Session["RoleId"] == null)

Response.Redirect("Login.aspx");

else

{

user = Session["UserID"].ToString();

String role = Session["RoleId"].ToString();

AppSettingsReader asr = new AppSettingsReader();

string conn = asr.GetValue("ConnectionString", typeof(System.String)).ToString();

try

{

SqlConnection sc = new SqlConnection(conn);

sc.Open();

string cmdstr = "SELECT fname,lname,college,address,city,state,zip,country,university,yjoin,yleave,DATENAME(month, dob) AS month, YEAR(dob) AS year, DAY(dob) AS Expr1, DATEDIFF(yy, dob, GETDATE()) AS age,email,hphone,cphone FROM register where username='" + user + "'";

string cmdstr1 = "SELECT pets,hobbies,dreamcompany,sports,aboutme FROM userdetails where username='" + user + "'";

string cmdstr2 = "SELECT photo FROM userdetails where username='" + user + "'";

SqlCommand cmd = new SqlCommand(cmdstr, sc);

SqlCommand cmd1 = new SqlCommand(cmdstr1, sc);

SqlCommand cmd2 = new SqlCommand(cmdstr2, sc);

SqlDataReader rd = cmd.ExecuteReader();

if (rd.Read())

{

Label36.Text = rd.GetString(0) + " " + rd.GetString(1);

Label37.Text = "College " + " " + rd.GetString(2);

Label1.Text = rd.GetString(0) + " " + rd.GetString(1);

Label2.Text = rd.GetString(0) + " " + rd.GetString(1);

Label8.Text = "College " + " " + rd.GetString(2);

Label3.Text = rd.GetString(3);

Label4.Text = rd.GetString(4);

Label5.Text = rd.GetString(5);

Label6.Text = rd.GetString(6);

Label10.Text = rd.GetString(7);

Label9.Text = "University " + rd.GetString(8);

Label7.Text = "Batch " + rd.GetString(9) + " - " + rd.GetString(10);

Label12.Text = rd[11] + " " + rd[13];

Label17.Text = rd[14].ToString();

Label24.Text = rd.GetString(15);

Label25.Text = rd.GetString(16);

Label26.Text = rd.GetString(17);

}

else

{

}

rd.Close();

SqlDataReader rd1 = cmd1.ExecuteReader();

if (rd1.Read())

{

Label19.Text = rd1.GetString(3);

Label32.Text = rd1.GetString(2);

Label33.Text = rd1.GetString(0);

Label34.Text = rd1.GetString(1);

Label35.Text = "About me:"+" "+rd1.GetString(4);

}

else

{

}

rd1.Close();

SqlDataReader rd2 = cmd2.ExecuteReader();

if (rd2.Read())

{

Image3.ImageUrl = rd2.GetString(0);

}

else

{

//Image2.ImageUrl = "C:/Inetpub/wwwroot/Classmates/images/photoimg.jpg";

}

rd2.Close();

sc.Close();

}

catch (Exception ex)

{

Response.Write(ex.Message);

}

}

}

}

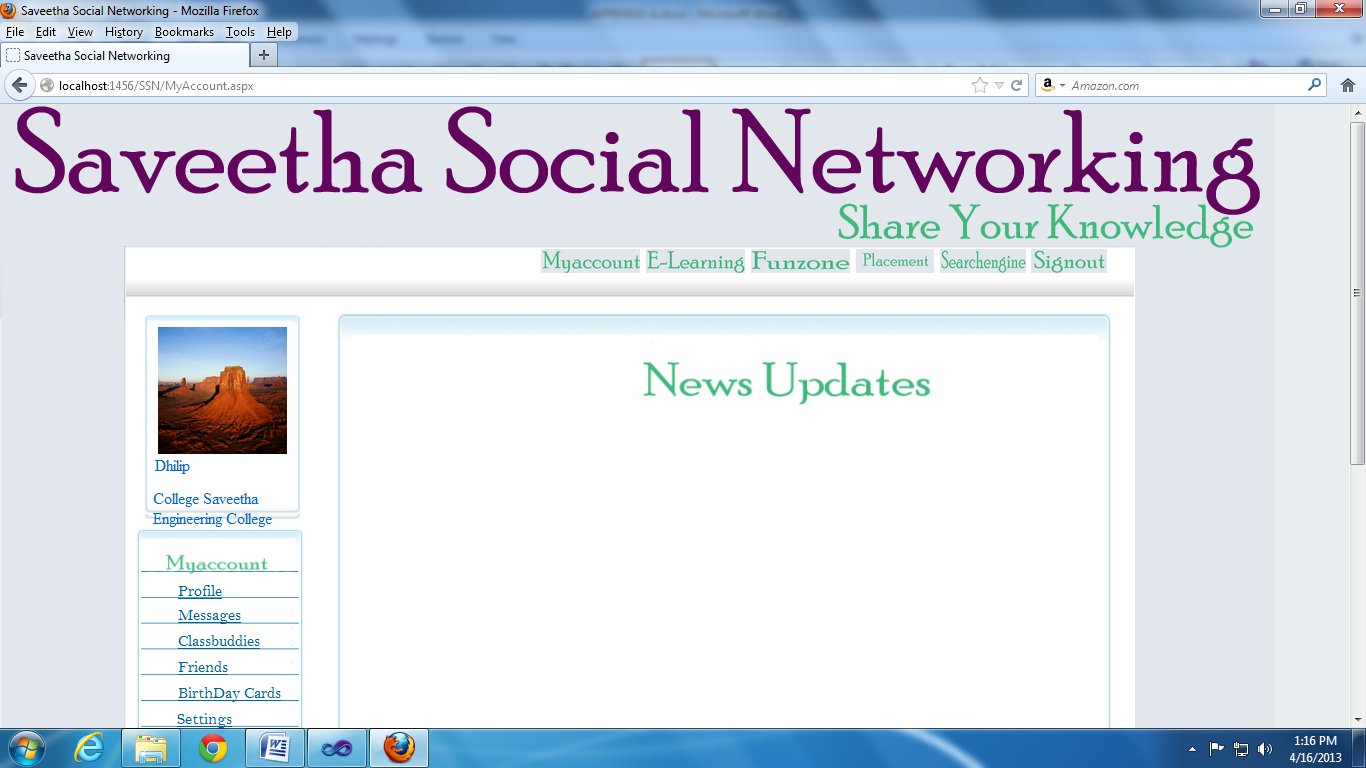
**APPENDIX B**

**OUTPUT SCREENSHOTS**

**Login Page**



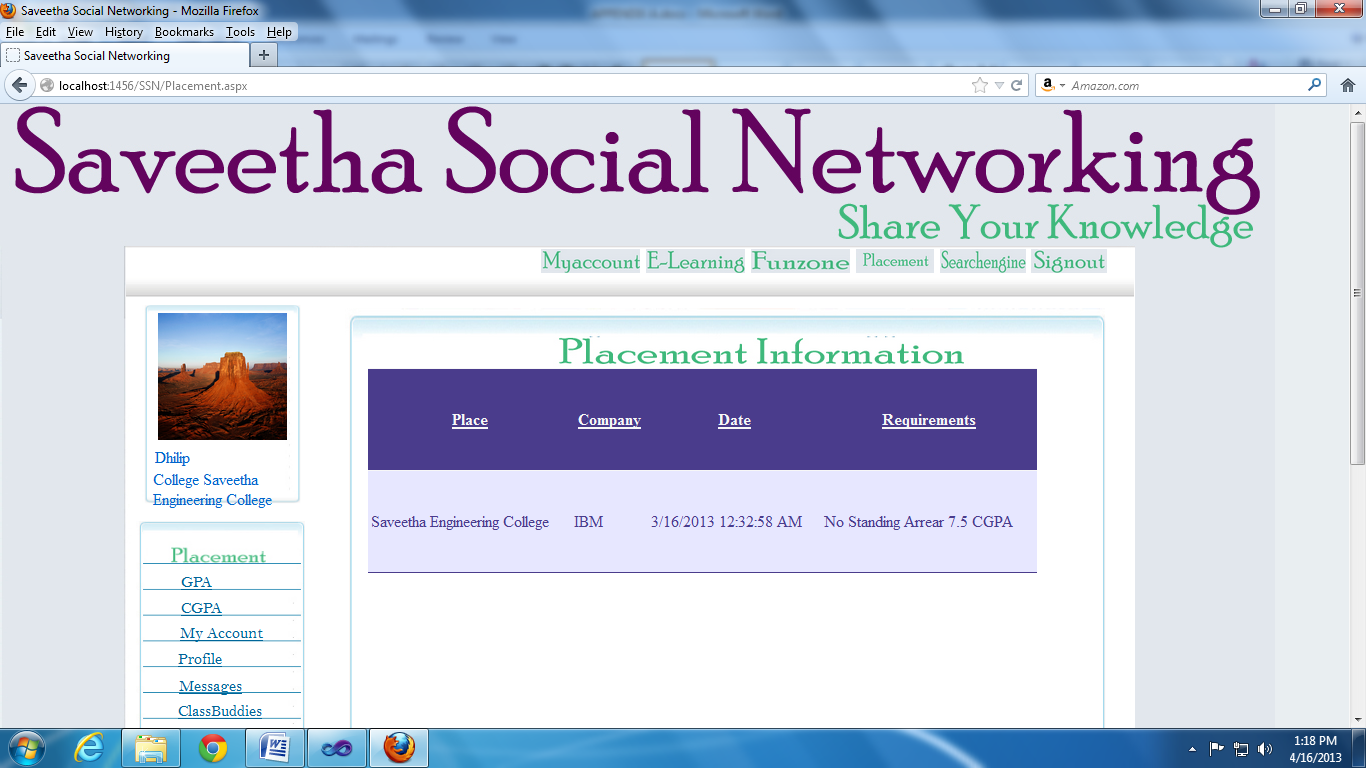
**My Account**



**Profile**



**Placement**



**Message**



**Inbox**



**REFERENCE**

[1] Ellison, Nicole B.; Steinfield, Charles; Lampe, Cliff (2007). ["The benefits of Facebook "friends": Exploring the relationship between college students' use of online social networks and social capital"](http://jcmc.indiana.edu/vol12/issue4/ellison.html)

[2] MacDonald, Matthew; Szpuszta, Mario (2005). *Pro ASP.NET 2.0 in C# 2005* (1st edition Ed.).

[3] Anne Boehm: *Murachs ASP.NET 3.5 Web Programming with VB 2008*, July 21, 2008, Mike Murach and Associates

[4] Israel B. Ocbina: *Mastering VB.NET and C#. 7th Edition. October 22, 2004. by Cyberocbina Cafê. A .NET Developers Edition*.

[5] Stephen Walther: *ASP.NET 3.5 Unleashed*, December 28, 2007, Sams Publishing

[6] Selly, Dominic; Andrew Troelsen, Tom Barnaby (2005). [*Expert ASP.NET 2.0 Advanced Application Design: Advanced Application Design*](http://books.google.co.in/books?id=RCVoZfzs6hwC&pg=PA191&lpg=PA191l&source=web&ots=dWcSWieXbi&sig=a6pm4m70IMmwJx5iqTLqenPDciQ&hl=en#PPA192,M1)