Online Attendance System

Project Report submitted in partial fulfillment of the requirement for the degree of

Bachelor of Technology.

in

Computer Science & Engineering

Under the Supervision of

Mr. Amol Vasudeva

By

Shalabh Goel(111292)

to

Jaypee University of Information and Technology

Waknaghat, Solan – 173234, Himachal Pradesh

Certificate

This is to certify that project report entitled “Online Attendance System”, submitted by Shalabh Goel in partial fulfillment for the award of degree of Bachelor of Technology in Computer Science & Engineering to Jaypee University of Information Technology, Waknaghat, Solan has been carried out under my supervision.

This work has not been submitted partially or fully to any other University or Institute for the award of this or any other degree or diploma.

Date:

Mr. Amol Vasudeva

Assistant Professor, CSE/IT Department

Acknowledgement

I would like to express my deep and sincere gratitude and warmly acknowledge the continuous encouragement, invaluable supervision, timely suggestions and inspired guidance offered by my guide Mr. Amol Vasudeva, who was the staunch supporter and motivator of this project. Right from the inception of this project work, my supervisor has guided me till the very end in the true sense of word.

I would also like to extend my gratitude to one and all who are directly involved in the successful completion of this project report.

Date: Shalabh Goel

Table of Content

List of Figures

List of Tables

Abstract

This project is absolutely on the computer-based attendance management system. Computer based attendance system provide efficient means of determining eligibility criteria for students to meet examination requirements. Now, with the advent of the computerized attendance management system, problems associated with the manual processing of student attendance will be alleviated to its minimum level. Hence, computerization of attendance management system aimed at its incorporating a computer based system in processing attendance of the student at promoting speed of operations and accuracy of result. In the overview, this project looks at the existing system of student attendance management and attempt to covert the process from manual to a computerized one in order to reduce the time spent on manual operations to eradicate errors and time consumptions. Security measures have also been installed so as to make sure that no private information like the password is disclosed. The password is encrypted using a cipher scheme before it is sent over the network and also before storing the username and password of a user in database.

CHAPTER 1 (INTRODUCTION)

1.1 BACKGROUND OF THE PROJECT

The world itself has become a global village of technology today, especially with the computer technology. With the advent of new technology, the world is now witnessing an easy way to keep information / data. Hence the introduction of computer is one of the greatest challenges facing man today.

Attendance Management System is software developed for daily student attendance in colleges and institutes. It facilitates to access the attendance information of a particular student in a particular class.

The information is sorted by the operators, which will be provided by the lecturer for a particular class. This system will also help in evaluating attendance eligibility criteria of a student. Since ages, attendance system has remained one of the most important systems for evaluating the working time of students in any college.

In short, this project is used to mark the number of days present/absent in any academic year of students in a college. The software system also helps in evaluating the examination eligibility criteria for a student in the sense that only those students with attendance above 70% (supposedly) are allowed to sit for the semester exams.

And this system will also help in evaluating attendance eligibility criteria of a student. In colleges, attendance is important and mandatory. Nowadays, due to the large number of students, it is efficient to use attendance management system to manage attendance in colleges/institutes.

1.2 AIMS AND OBJECTIVES

This report describes the capabilities that will be provided by the Software application Student Attendance Management System. The purpose of developing this attendance management system is to computerize the traditional way of taking attendance in classes and also manage student information along with their classes and subjects.

Another purpose for developing the software is to generate the reports automatically whenever required,-in between the semester or after the semester.

1.3 PROJECT VISION

The vision of the project is to develop an online attendance website which comes in handy for all the students/faculty of the institute/college/organization. It will provide an easy way of taking attendance reducing large amount of paper work for the faculty. It will also ensure the security of the user so as no one else is able to access the password of other person.

1.4 MOTIVATION

Present market is based on the internet and all the IT firms are focusing themselves on the development of web applications. Using JSP one can build web applications as well web services. So, experienced gained in this project will be useful in future.

1.5 SCOPE OF THE PROJECT

This project involves the design of computer software which is capable of taking attendance record of students for a particular course with the aim/ purpose of determining their eligibility to sit for the exam. Another striking feature about the program is, it has the capability of sending the attendance records in the form of an email to the parents / guardian of the student.

1.6 FEASIBILITY STUDY

1.6.1 ECONOMIC FEASIBILITY:

The system being developed is economic with respect to College or institution's point of view. It is cost effective in the sense that has eliminated the paper work completely. The system is also time effective because the calculations are automated which are made at the end of the month or as per the user requirement. The result obtained contains minimum errors and are highly accurate as the data is required.

1.6.2 TECHNICAL FEASIBILITY:

The technical requirement for the system is economic and it does not use any other additional Hardware or Software.

1.6.3 BEHAVIORAL FEASIBILITY:

The system working is quite easy to use and learn due to its simple but attractive interface. User requires no special training for operating the system.

1.7 INTRODUCTION TO MODULES

Attendance Management System basically has three main modules for proper functioning:

1. Admin.
2. Faculty.
3. Student.

- First module is the admin, which has right for creating space for new batch.(Any entry of new faculty, updating subject if necessary, and sending notice.)
- Second module is handled by the user (sub-admin) which can be a faculty. User has a right of making daily attendance, generating report.
- Third module is the student, who can view his/her attendance which is regularly updated by the faculty.

Attendance can be taken:

- On the basis of Class (Lecture or tutorial)
- On the basis of Subject & Course (Btech/Mtech).
- On the basis of Semester.

1.8 DEFINITION OF TERMS

1. **COMPUTER:** A computer can be finely defined as an automatic electronic machine, that can accept raw fact or data through an input device like the keyboard or mouse: stores, processes the data in the system unit using a particular program and finally supplying the result of information through an output device such as monitor in the form of pictures or texts on the screen called the soft copy or a printer in form of printed paper called the hardcopy.

2. **DATA:** This is a raw material/ facts that have not been processed. Data are also facts about an organization or institution, and its daily transactions.

3. **INFORMATION:** These are data that has been processed and is meaningful to the end user.

4. **COMPUTER-BASED:** It is also referred to as computerized. It is the art of using computer system to carry out task.

5. **ATTENDANCE:** This is an act or fact of attending (being present at) work or institution. Also, attendance is used to define the number of person present at a particular day of an event or an institution.

6. **SYSTEM:** A system is a group of inter-related components working together towards a common goal by accepting input and producing output in an organized transformation process. It is also an organized collection of people; machine and method required to accomplish a set of specific functions.

7. **DATABASE:** Are not merely collections of files, rather the database is a centre source of data meant to be shared by many users for a variety of applications.

CHAPTER 2 (PROJECT WALKTHROUGH)

To develop the online attendance system, **Java Server Pages (JSP)** and **HTML** has been used as front end. A JSP page is a text-based document that describes how to process a request to create a response. JSP are most often used to generate HTML. However, you should not infer from this that JSPs are limited to generating HTML. They can be used to generate any textual output, Wireless Markup Language (WML) to communicate with handheld devices like mobile phones.

Before coming to the actual coding of the program, some important terms need to be defined so that the concept becomes clear. The purpose of this online attendance system is to view a web page that is responsible for receiving input from user and store it in database for current or future reference.

The **request** objects are passed as parameters to the _jspService() method when a client request is made. A request object is of the javax.servlet.http.HttpServletRequest type.
The **response** object is used to carry the response a client after the _jspService() method is executed. The response object is of the javax.servlet.http.HttpServletRespone type.
The **pageContext** object represents the context of the current JSP page. It provides a single API to manage the various scoped attributes.
The **session** object helps access session data and is of the HttpSession type. It is declared if the value of the session attribute in a page directive is true. By default, the value of the session attribute is always true.

2.1 JAVA SERVER PAGES (JSP) OVERVIEW

Architecturally, JSP may be viewed as a high-level abstraction of Java servlets. JSPs are translated into servlets at runtime; each JSP servlet is cached and re-used until the original JSP is modified. JSP can be used independently or as the view component of a server-side model–view–controller design, normally with JavaBeans as the model and Java servlets as the controller.

JSP allows Java code and certain pre-defined actions to be interleaved with static web markup content, with the resulting page being compiled and executed on the server to deliver a document. The compiled pages, as well as any dependent Java libraries, use Java bytecode rather than a native software format. Like any other Java program, they must be executed within a Java virtual machine (JVM) that integrates with the server's host operating system to provide an abstract platform-neutral environment.

JSPs are usually used to deliver HTML and XML documents, but through the use of OutputStream, they can deliver other types of data as well. The Web container creates JSP implicit objects like pageContext, servletContext, session, request & response.

2.1.1 JSP-ARCHITECTURE

The web server needs a JSP engine i.e. a container to process JSP pages. The JSP container is responsible for intercepting requests for JSP pages. This tutorial makes use of Apache which has built-in JSP container to support JSP pages development.

A JSP container works with the Web server to provide the runtime environment and other services a JSP needs. It knows how to understand the special elements that are part of JSPs.

Following diagram shows the position of JSP container and JSP files in a Web Application

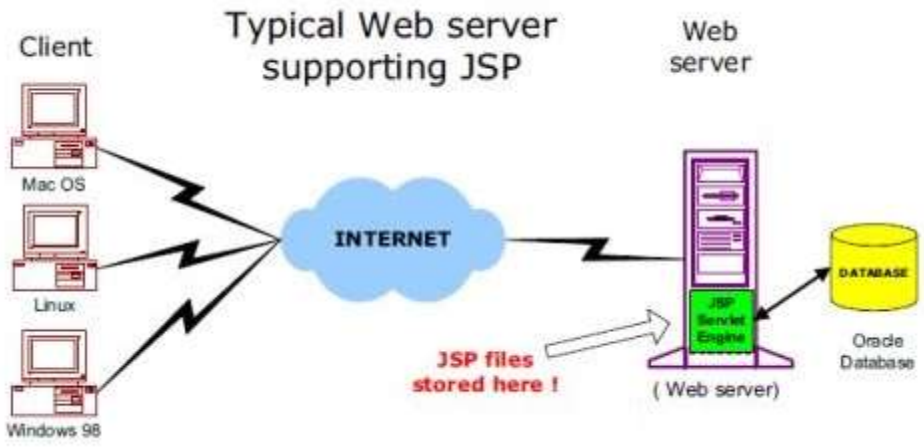

Figure 1: JSP Architecture

2.1.2 TWO-TIER AND THREE-TIER ARCHITECTURE

In the 1980s, the arrival of inexpensive network-connected PCs produced the popular two-tier client-server architecture. In this architecture, there is an application running in the client machine which interacts with the server—most commonly, a database management system (see Figure 2).

Typically, the client application, also known as a fat client, contained some or all of the presentation logic (user interface), the application navigation, the business rules and the database access.

Every time the business rules were modified, the client application had to be changed, tested and redistributed, even when the user interface remained intact. In order to minimize the impact of business logic alteration within client applications, the presentation logic must be separated from the business rules. This separation becomes the fundamental principle in the three-tier architecture.

The two tiers of two-tier architecture is

1. Database (Data tier)
2. Client Application (Client tier)

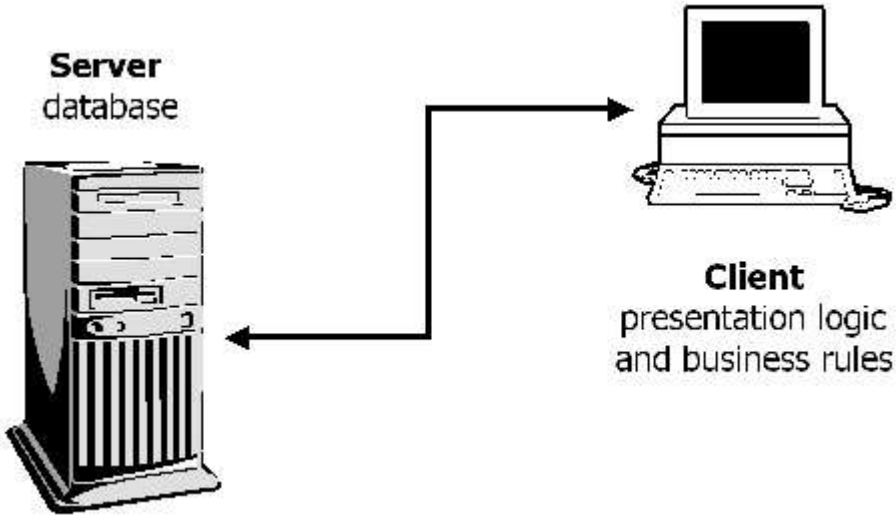

FIGURE 2: TWO-TIER ARCHITECTURE

Advantages:

1. Understanding and maintenances is easier.

Disadvantages:

1. Performance will be reduced when there are more users.

In a three-tier architecture (also known as a multi-tier architecture), there are three or more interacting tiers, each with its own specific responsibilities (see Figure 3):

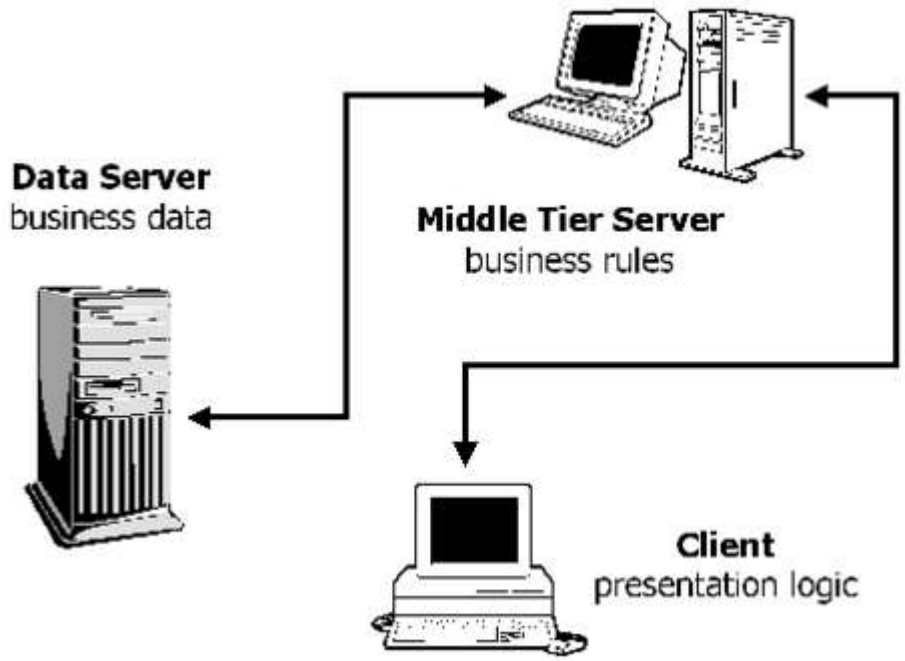

FIGURE 3: THREE-TIER ARCHITECTURE

- **Tier 1**: the client contains the presentation logic, including simple control and user input validation. This application is also known as a thin client.

- **Tier 2**: the middle tier is also known as the application server, which provides the business processes logic and the data access.

- **Tier 3**: the data server provides the business data.

Three tier architecture having three layers. They are

1. Client layer
2. Business layer
3. Data layer

Client layer: Here we design the form using textbox, label etc.

Business layer: It is the intermediate layer which has the functions for client layer and it is used to make communication faster between client and data layer. It provides the business processes logic and the data access.

Data layer: it has the database.

ADVANTAGES OF THREE-TIER ARCHITECTURE:

- It is easier to modify or replace any tier without affecting the other tiers.
- Separating the application and database functionality means better load balancing.
- Adequate security policies can be enforced within the server tiers without hindering the clients.

2.1.3 JSP PROCESSING

As with a normal page, your browser sends an HTTP request to the web server. The web server recognizes that the HTTP request is for a JSP page and forwards it to a JSP engine. This is done by using the URL or JSP page which ends with **.jsp** instead of .html.

The JSP engine loads the JSP page from disk and converts it into servlet content. This conversion is very simple in which all template text is converted to println( ) statements and all JSP elements are converted to Java code that implements the corresponding dynamic behavior of the page. The JSP engine compiles the servlet into an executable class and forwards the original request to a servlet engine.

A part of the web server called the servlet engine loads the Servlet class and executes it. During execution, the servlet produces an output in HTML format, which the servlet engine passes to the web server inside an HTTP response. The web server forwards the HTTP response to your browser in terms of static HTML content.

Finally web browser handles the dynamically generated HTML page inside the HTTP response exactly as if it were a static page.

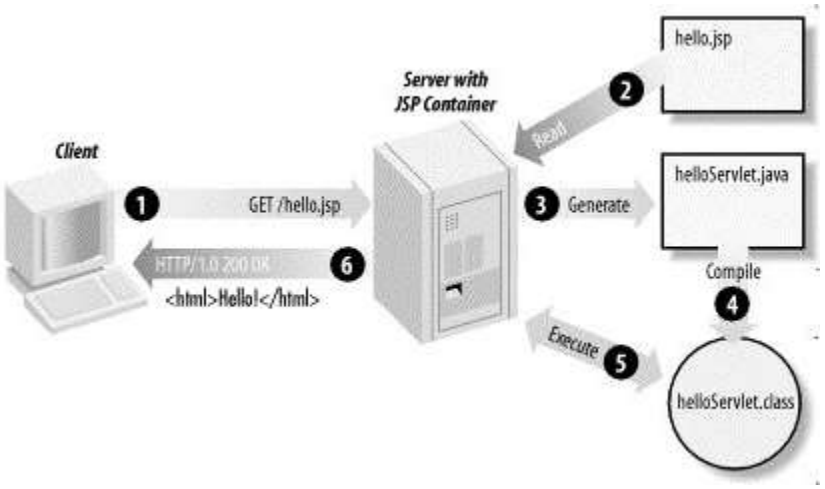

Figure 4: JSP Processing

2.2 JAVA SERVLETS

Java Servlets are programs that run on a Web or Application server and act as a middle layer between requests coming from a Web browser or other HTTP client and databases or applications on the HTTP server. Using Servlets, you can collect input from users through web page forms, present records from a database or another source, and create web pages dynamically.

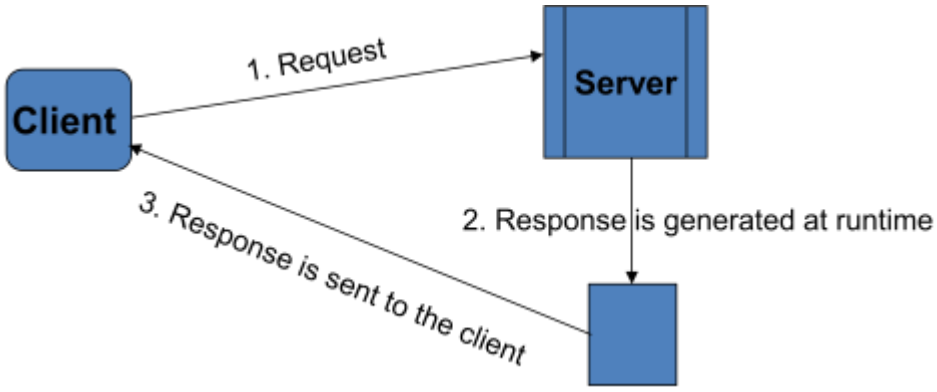

Figure 5: Servlets Architecture

Advantages of Servlets:

- The web container creates threads for handling the multiple requests to the Servlet.
- Threads have many benefits over the processes such as they share a common memory area, lightweight, cost of communication between the threads are low.
- Better performance: because it creates a thread for each request, not the process.
- Portability: because it uses Java Language.
- Secure: because it uses Java Language.
- Robust: Servlets are managed by JVM so no need to worry about memory leaks, garbage collection etc.

2.3 ADVANTAGES OF JSP OVER SERVLETS

Most of the content of a Web page is static and different components need to be placed on the page in a symmetrical way. These static pages are designed by Web designers by using HTML code.

Designing static pages using Servlets require a good knowledge of Java, but Web designers may not be comfortable with Java programming constructs. In addition, while using Servlets, you have to enclose HTML code in the out.println() method, which disables all IDE support for generating HTML content. Therefore, using JSP is always easier than using Servlets for the same HTML output.

You need to recompile your Servlet for every single change in the source code of the Servlet; whereas in case of JSP, recompilation is not required since JSP are automatically handled by the Web container for any update in their code. Servlets cannot be accessed directly and have to be first mapped in the web.xml file, whereas a JSP page can be accessed directly as a simple HTML page.

Both, Servlets and JSPs are server-side components used to generate dynamic HTML pages; however, JSP is preferred over Servlet as it is developed by using a simple HTML template and automatically handled by the JSP container.

2.4 JSP-SESSION TRACKING

HTTP is a "stateless" protocol which means each time a client retrieves a Web page, the client opens a separate connection to the Web server and the server automatically does not keep any record of previous client request.

Still there are following three ways to maintain session between web client and web server:

1. **COOKIES:** A webserver can assign a unique session ID as a cookie to each web client and for subsequent requests from the client they can be recognized using the received cookie. This may not be an effective way because many time browsers do not support a cookie, so I would not recommend using this procedure to maintain the sessions.

2. **HIDDEN FORM FIELDS:** When the form is submitted, the specified name and value are automatically included in the GET or POST data. Each time when web browser sends request back, then session_id value can be used to keep the track of different web browsers. This could be an effective way of keeping track of the session but clicking on a regular (<A HREF...>) hypertext link does not result in a form submission, so hidden form fields also cannot support general session tracking.

3. **THE SESSION OBJECT:** JSP makes use of servlet provided HttpSession Interface which provides a way to identify a user across more than one page request or visit to a Web site and to store information about that user. By default, JSPs have session tracking enabled and a new HttpSession object is instantiated for each new client automatically. Disabling session tracking requires explicitly turning it off by setting the page directive session attribute to false.

2.5 JSP-JAVABEANS

JavaBeans are reusable software components that separate the business logic from the presentation logic. These are classes that encapsulate many objects into a single object (the bean). It provides the benefits of using separate Java classes instead of embedding large amount of Java code directly in a JSP page.

They are serializable, have a no-argument constructor, and allow access to properties using **getter** and **setter** methods. The name Bean was given to encompass this standard which is meant to create reusable software components for Java.

Advantages of JavaBeans:

- The properties, events, and methods of a bean that are exposed to another application can be controlled.
- A bean may register to receive events from other objects and can generate events that are sent to those other objects.
- Auxiliary software can be provided to help configure a java bean.
- The configuration setting of bean can be saved in a persistent storage and restored.

2.6 HYPERTEXT MARKUP LANGUAGE (HTML)

HTML or HyperText Markup Language is the standard markup language and is used to create Web pages.

HTML is written in the form of HTML elements consisting of *tags* enclosed in angle brackets (like <html>). HTML tags most commonly come in pairs like <h1> and </h1>, although some tags represent empty elements and so are unpaired, for example <img>. The first tag in a pair is the start tag, and the second tag is the end tag (they are also called opening tags and closing tags).

A Web browser can read HTML files and compose them into visible or audible Web pages. The browser does not display the HTML tags and scripts, but uses them to interpret the content of the page. HTML describes the structure of a Website semantically along with cues for presentation, making it a markup language, rather than a programming language.

HTML elements form the building blocks of all Websites. HTML allows images and objects to be embedded and can be used to create interactive forms. It provides a means to create structured documents by denoting structural semantics for text such as headings, paragraphs, lists, links, quotes and other items. It can embed scripts written in languages such as JavaScript which affect the behavior of HTML Web pages.

Web browsers can also refer to Cascading Style Sheets (CSS) to define the look and layout of text and other material. The World Wide Web Consortium (W3C), maintainer of both the HTML and the CSS standards, encourages the use of CSS over explicit presentational HTML

2.7 CASCADING STYLE SHEETS (CSS)

Cascading Style Sheets (CSS) is a style sheet language used for describing the look and formatting of a document written in a markup language. While most often used to change the style of web pages and user interfaces written in HTML. Along with HTML and JavaScript, CSS is a cornerstone technology used by most websites to create visually engaging webpages, user interfaces for web applications, and user interfaces for many mobile applications.

CSS is designed primarily to enable the separation of document content from document presentation, including elements such as the layout, colors, and fonts. This separation can improve content accessibility, provide more flexibility and control in the specification of presentation characteristics, enable multiple HTML pages to share formatting by specifying the relevant CSS in a separate ".css" file, and reduce complexity and repetition in the structural content, such as semantically insignificant tables that were widely used to format pages before consistent CSS rendering was available in all major browsers.

CSS makes it possible to separate presentation instructions from the HTML content in a separate file or style section of the HTML file. For each matching HTML element, it provides a list of formatting instructions. For example, a CSS rule might specify that "all heading 1 elements should be bold," leaving pure semantic HTML markup that asserts "this text is a level 1 heading" without formatting code such as a <BOLD> tag indicating how such text should be displayed.

This separation of formatting and content makes it possible to present the same markup page in different styles for different rendering methods. It can also be used to display the web page differently depending on the screen size or device on which it is being viewed.

CHAPTER 3: SYSTEM DESIGNING

3.1 SOFTWARE ENGINEERING MODEL

The model employed to materialize the Online Attendance System is the "**Modified Waterfall model**".

This method suggests a systematic, sequential approach to software development. This model provides the features of "**Waterfall Model**" as well as the flavor of **"Prototype Model".** As this type of system is already developed, so requirements are very much clear to us. Hence we can use this model.

At the beginning the system will be developed by keeping only employees in mind, so all the activities will be followed in case of any improvements in the system.

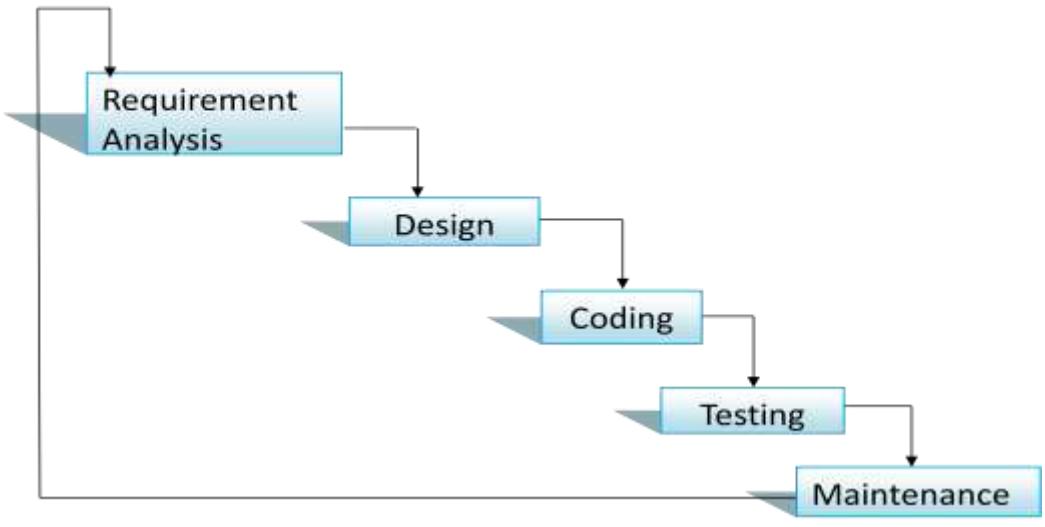

FIGURE 6: WATERFALL MODEL

3.1.2 DEVELOPMENT TOOLS

TECHNOLOGY USED:

LANGUAGE USED:

FRONT END: JSP, HTML, CSS

BACK END: SQLEXPRESS 2008

SYSTEM REQUIREMENTS:

MINIMUM RAM: 512 Mb and above

MINIMUM MEMORY (HARD DISK): 512 Mb and above

PROCESSOR: Intel Pentium 4 and above

OPERATING SYSTEM: Windows XP and above

3.3 SYSTEM REQUIREMENTS SPECIFICATION (S.R.S)

3.3.1 FUNCTIONAL REQUIREMENTS

The functional requirement of the project is defined under three modules. The first module allows the system Administrator(admin) to log into his account and has the privileges to do multiple things some of the include adding new student, modifying student information and modifying student information, deleting of student and sub admin, also there is a provision to change login password.

The second module of the project defines itself in terms of being used by the sub-admin (Lecturers) Lecturer have to enter their login id and Password in system. After that the id is verified and the records of Student of particular semester are displayed on the screen. Lecturer now mark the attendance of student who is present in class, lecturer can also change their password.

The third module of the project allows the students to log into the system and view their current attendance statistics. No other privileges are given to the student.

3.3.2 NON-FUNCTIONAL REQUIREMENTS

- Hardware requirements:

Hardware Interface 1: The system should be embedded in the PC/Laptop.

Hardware Interface 2: 512 Mb hard disk and 512 MB RAM.

- Software Requirements:

Software Interface: Student Attendance management System and Database.

3.4 METHODOLOGY

The web application will be developed in three phases:

- Homepage or Master Page
- Login
- Forms

MASTER MODULE: Master Page will describe the main appearance of the application and it will also act as the homepage of the homepage for the web application. From this page user will be able to go to the pages where he can see the attendance.

- Display pages will be used to display the information like information regarding attendance, profile info, courses selected.

LOGIN MODULE: A login page will be made from which user can login and all the things would be displayed according to the type of user. Login module will check whether the user is valid user or not. For this purpose user will have a valid username and password.

There will be three types of user for this system:

- Admin
- Student
- Faculty

FORMS MODULE: This module consists of the following modules:

- Enter attendance Form
- Student view attendance Form
- Change password Form

3.5 UML DIAGRAMS

3.5.1 USE CASE MODELLING

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases.

The main purpose of a use case diagram is to show what system functions are performed. Roles in the system can be depicted.

USE CASE DESCRIPTION:

USE CASE: Attendance System

ACTORS: Administrator, Faculty, Student

In this diagram there are three actors i.e. Admin who is responsible for adding user record to database, delete, modify the record of existing user. Faculty who will mark attendance which will be stored in database for reference purpose, he/she has the authority to change his/her password. Student who can view his/her current attendance and aggregate percentage attendance.

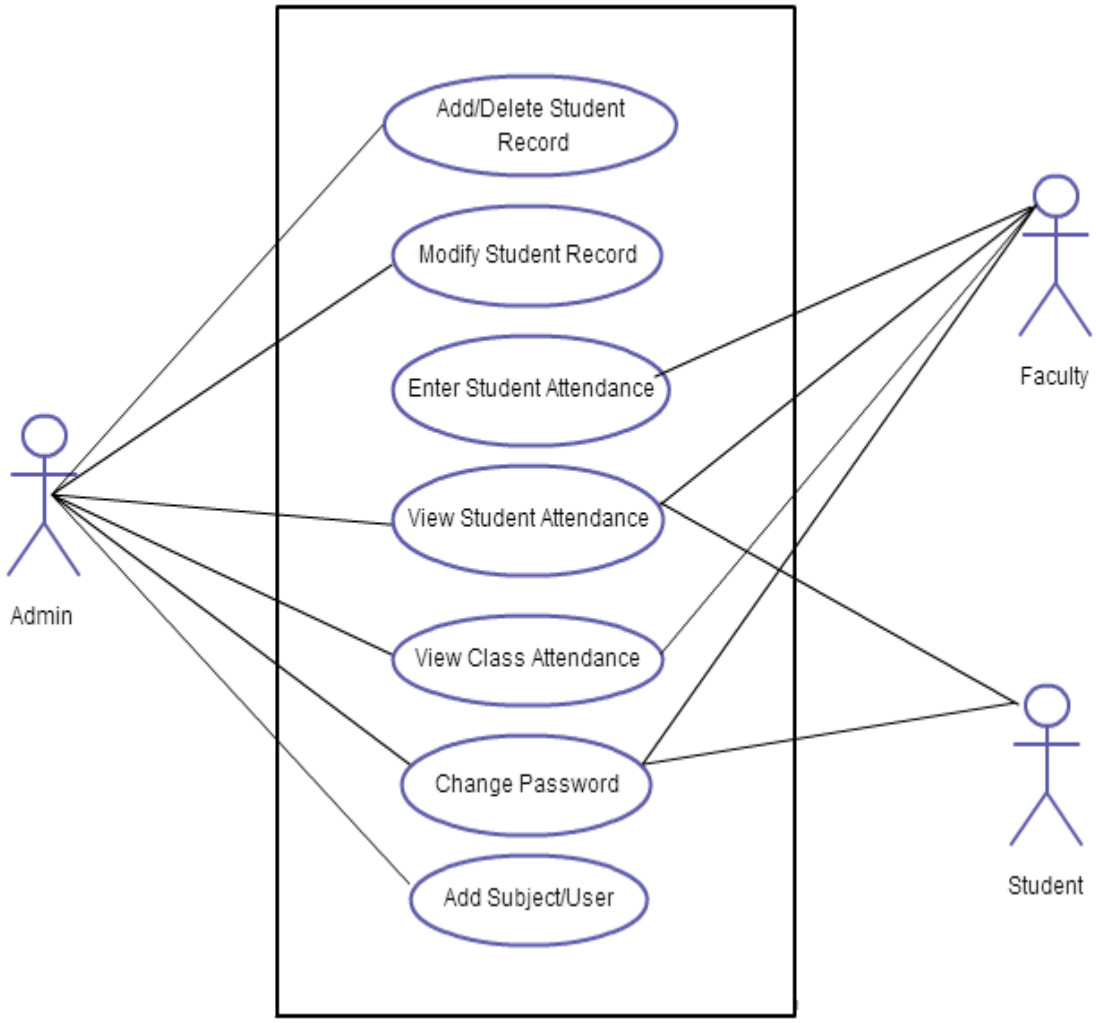

FIGURE 7: USE CASE DIAGRAM

3.5.2 SEQUENCE DIAGRAM

A sequence diagram is an interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. A sequence diagram shows object interactions arranged in time sequence.

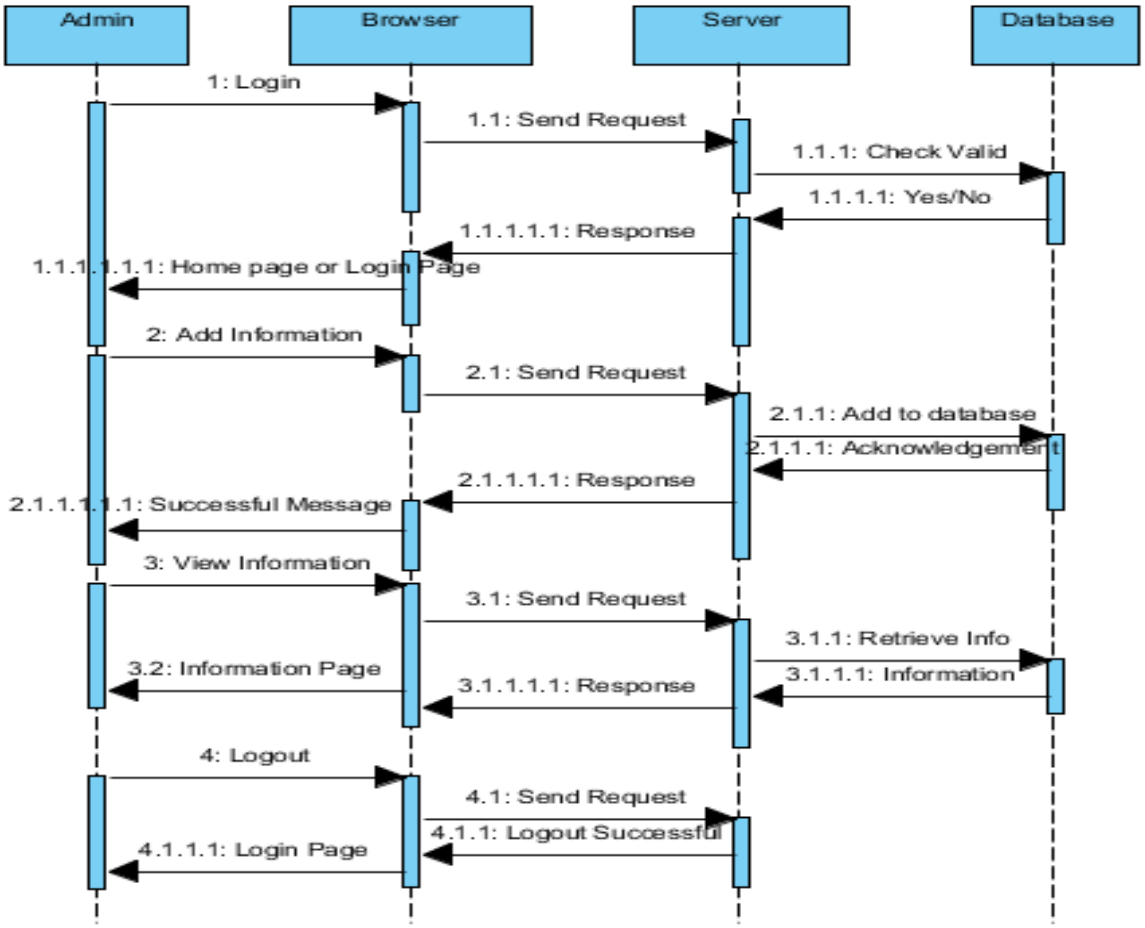

FIGURE 8: SYSTEM SEQUENCE DIAGRAM

3.5.3 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams are intended to model both computational and organizational processes (i.e. workflows). Activity diagrams show the overall flow of control.

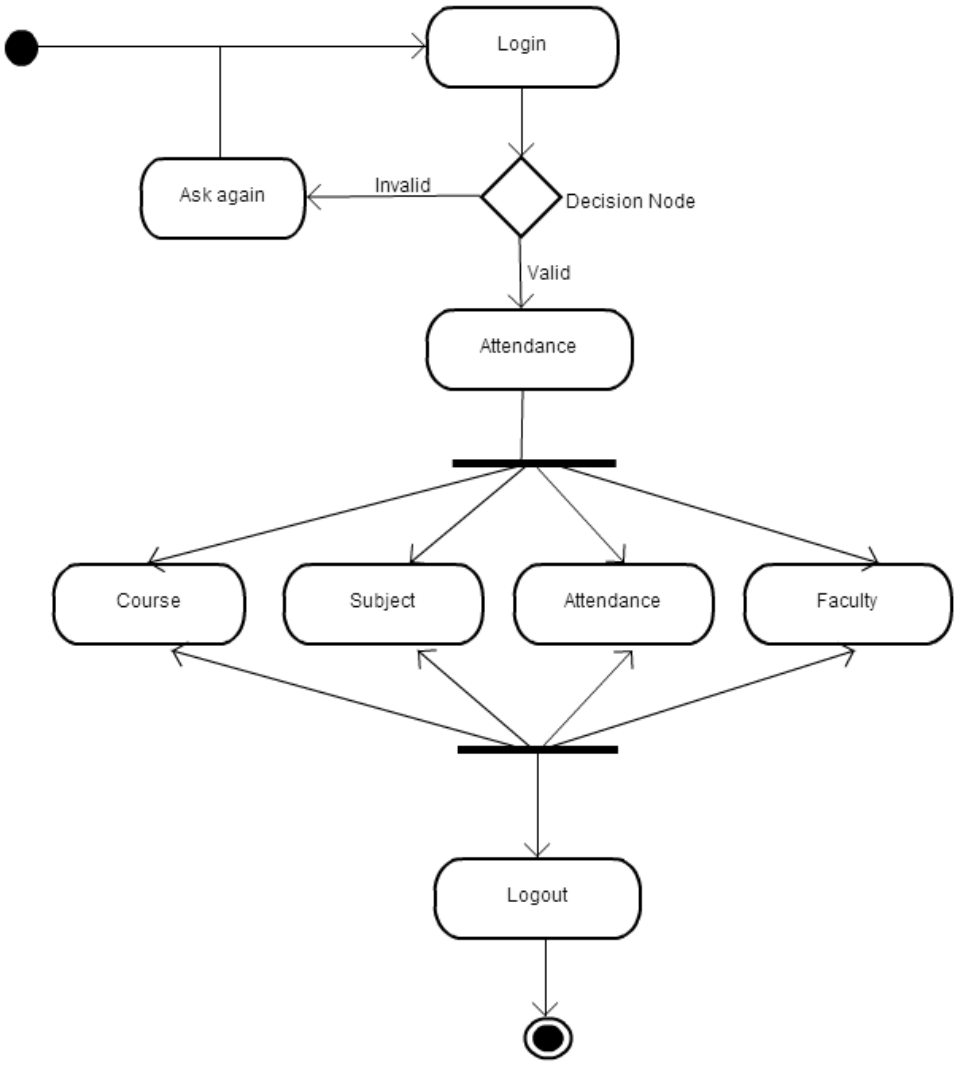

FIGURE 9: ACTIVITY DIAGRAM - 1

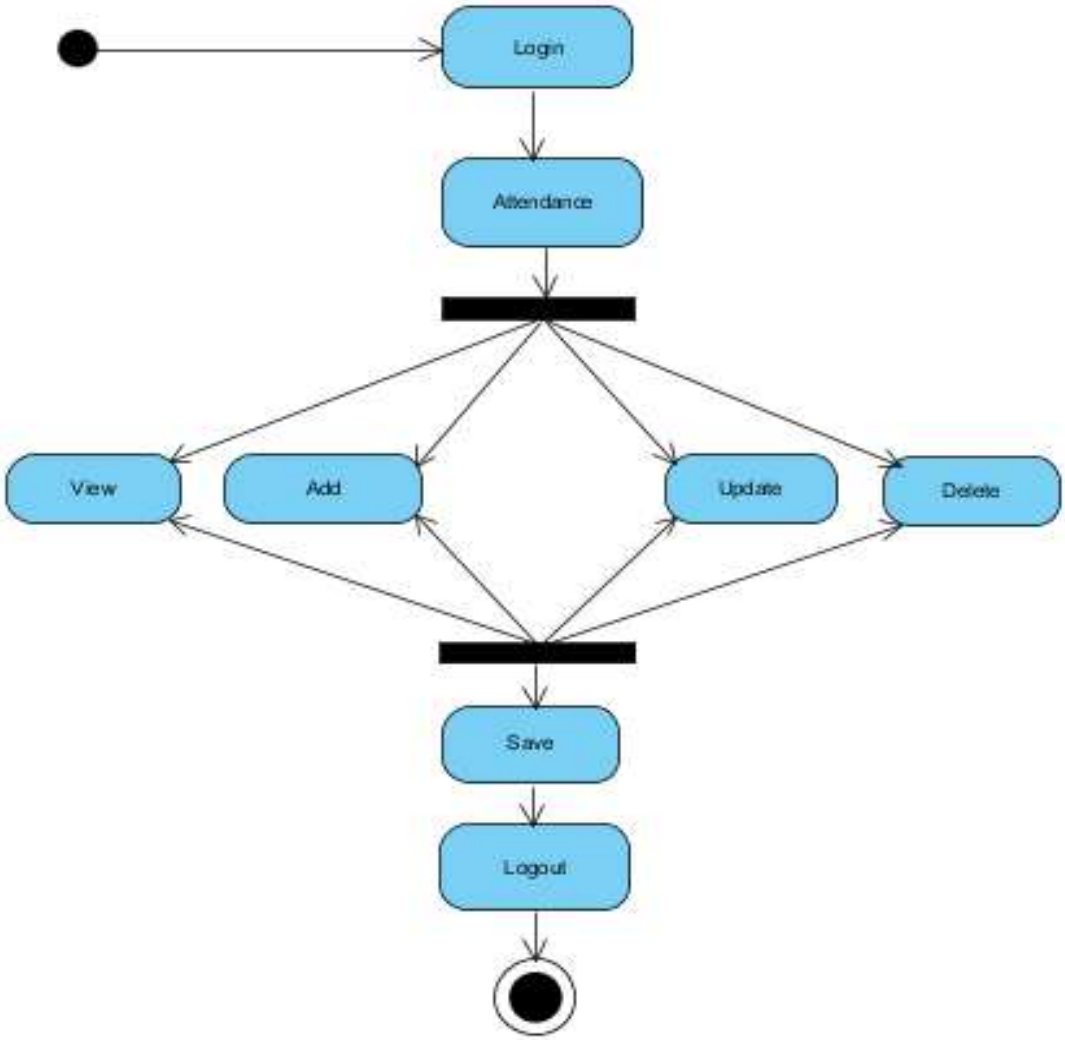

FIGURE 10: ACTIVITY DIAGRAM – 2

3.5.4 DATA FLOW DIAGRAM

This diagram depicts the flow of data from one point of system to another. A DFD shows what kind of information will be input to and output from the system, where the data will come from and go to, and where the data will be stored. It does not show information about the timing of processes, or information about whether processes will operate in sequence or in parallel

LEVEL '0' DFD OR CONTEXT DIAGRAM:

FIGURE 11: CONTEXT DIAGRAM

LEVEL 1 DFD:

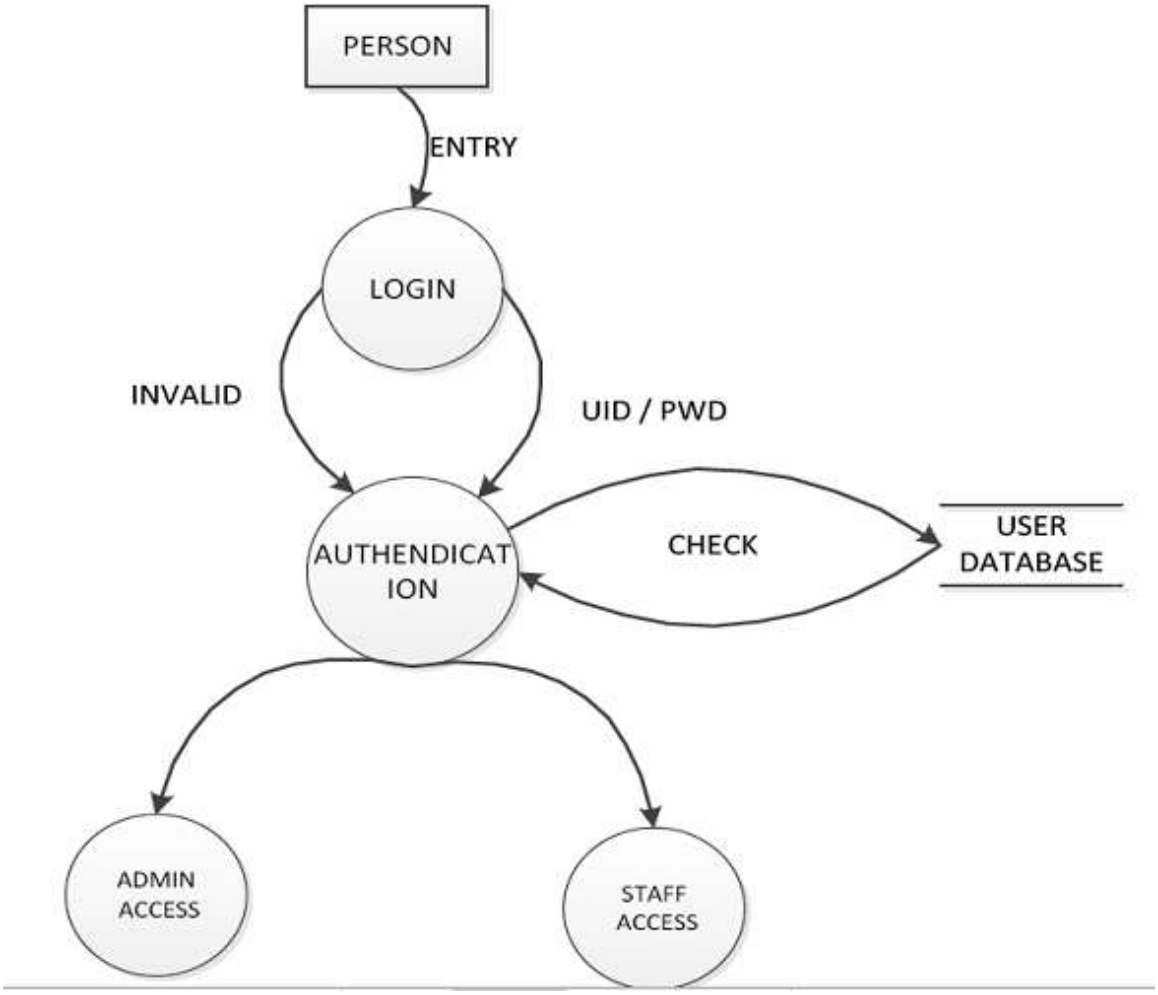

FIGURE 12: LEVEL 1 DFD

STAFF

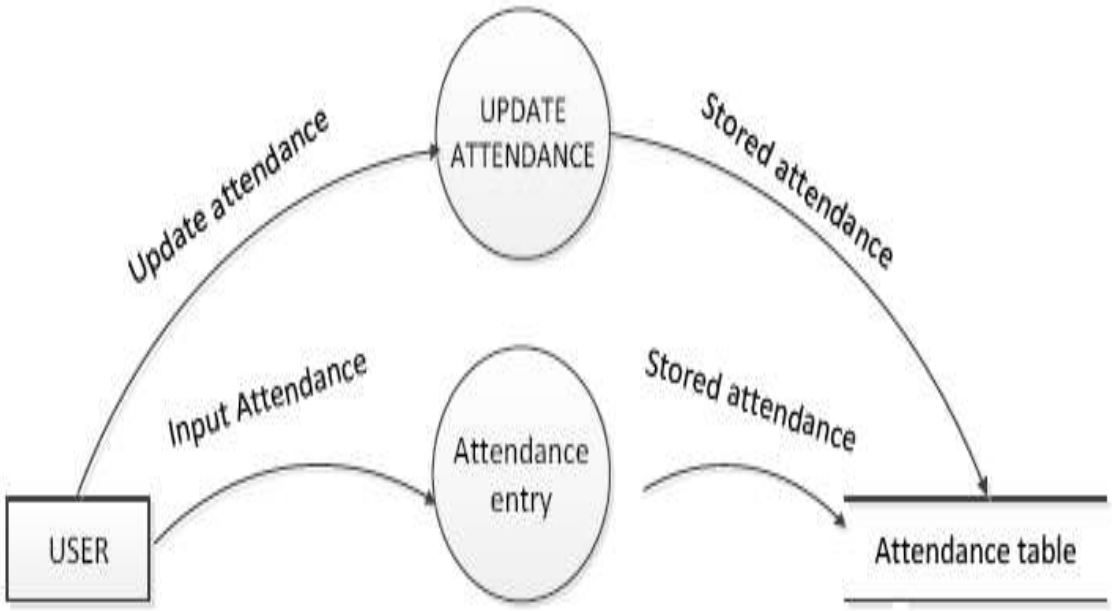

FIGURE 13: LEVEL 2 DFD

3.5.5 DATABASE DESIGN

3.5.5.1 LOGIN TABLE

- To create login details for the user.

FIELDS	DATATYPE	CONSTRAINTS	DESCRIPTION
Tablename	Varchar(20)	Primarykey	Stored number of tables from login

TABLE 1: LOGIN DETAILS

3.5.5.2 TEACHER/STAFF TABLE

- To create a table for username and password details of teachers.

FIELDS	DATATYPE	CONSTRAINTS	DESCRIPTION
Scode	Varchar(20)	Primarykey	Defines separate subject code id
Sname	Varchar(15)	NotNull	Subject name (ex:algo)
f_name	Varchar(15)	NotNull	Teacher's first Name
l_name	Varchar(15)	NotNull	Teacher's last Name
Pass	Varchar(20)	NotNull	Teacher login Password

TABLE 2: TEACHER'S DETAILS

3.5.5.3 STUDENT TABLE

- To create a table for student's personal details.

FIELDS	DATATYPE	CONSTRAINTS	DESCRIPTION
Userid	Varchar(6)	Auto_increment Primarykey	Student roll number
f_name	Varchar(20)	NotNull	Student's first name
l_name	Varchar(20)	NotNull	Student's last name
Email	Varchar(15)	NotNull	Student's email id
Pass	Varchar(20)	NotNull	Student's password for login
Date	Date	NotNull	Enter day by day attendance
Sub	Varchar(15)	NotNull	Particular subject

TABLE 3: STUDENT'S DETAILS

3.5.6 INPUT DESIGN

Input design is part of overall system design that requires special attention designing input data is to make the data entered easy and free from **errors**. The input forms are designed using the controls available in JSP framework. Validation is made for each and every data that is entered.

Input design is the process of converting the user originated inputs to a computer based format. A system user interacting through a workstation must be able to tell the system whether to accept the input to produce reports. The collection of input data is considered to be most expensive part of the system design. Since the input has to be planned in such a manner so as to get relevant information, extreme care is taken to obtain pertinent information.

This project first will entered to the input of allocation forms it will be created on student details form and subject entry form, time table form .it will helps to calculate subject wise attendance system. Next one if u want any verification on your data's also available in details show forms. Attendance to be entered single subject wise is available in this attendance system project.

3.5.7 OUTPUT DESIGN

Output design this application "**Student Attendance management system"** generally refers to the results and information that are generated by the system for many end-users; output is the main reason for developing the system and the basis on which they evaluate the usefulness of the application.

The output is designed in such a way that it is attractive, convenient and informative. Forms are designed with various features, which make the console output more pleasing. As the outputs are the most important sources of information to the users, better design should improve the system's relationships with us and also will help in decision making. Form design elaborates the way output is presented and the layout available for capturing information.

One of the most important factors of the system is the output it produces. This system refers to the results and information generated. Basically the output from a computer system is used to communicate the result of processing to the user.

Attendance management system to show the report subject wise attendance maintaining by staffs. These forms will show weekly report and consolidate report generated date, batch, and class wise to our end user.

3.5.8 DESIGN CONSTRAINTS

This section will design limitations that apply to the system being developed and will be adhered to give during the development phase of the project

Programming language:

The server program that is written in Java using the Standard Edition (SE v1.5) development kit on the computer.

Databases:

Microsoft MySQL Server 2008 are used as the database management system of the system. The database is stored on the server.

Development tool:

The MyEclipse Professional IDE and the Net Beans IDE 6.0 should be used as the primary development tools to build. On the server and the client programs

Standards:

The server program can run under Windows XP, Windows 7, Windows 8 operating systems. The server requires that the Java 1.5 (or higher) Runtime Environment installed on your computer. The computer hardware specifications must meet the minimum requirements of 2.3 GHz CPU speed, 2.0GB of RAM, and at least 500Mb of hard disk space for database storage.

Legal restrictions:

The only authority the installation of the system must approve the dean of educational services (or an equivalent distribution).

3.5.9 OTHER DATABASE REQUIREMENTS:

- The system includes three databases: student, class, and instructor
- The student database contains student information, such as name, address and identification number containing student.
- The class database contains information about classes' class title, department code, and of course include number of students. The class databases are grouped by department code and course name.

3.5.10 CHARACTERISTICS OF THE PROPOSED SYSTEM

- **User Friendly:** The proposed system is user friendly because the retrieval and storing of data is fast and data is maintained efficiently. Moreover the graphical user interface is provided in the proposed system, which provides user to deal with the system very easily.

- **Reports are easily generated:** reports can be easily generated in the proposed system so user can generate the report as per the requirement (monthly) or in the middle of the session. User can give the notice to the students so he/she becomes regular in class.

- **Very less paper work:** The proposed system requires very less paper work. All the data is feted into the computer immediately and reports can be generated through computers. Moreover work becomes very easy because there is no need to keep data on papers.

- **Computer operator control**: Computer operator control will be there so no chance of errors. Moreover storing and retrieving of information is easy. So work can be done speedily and in time.

3.5.11APPLICATION USER INTERFACE

As the System has got its interface created, the following descript the run through of the application. Figure 14 depicts the first window that user gets when he runs the system.

Figure 14: Home Page

This window shows that user has two options i.e. “LOGIN” and “CHANGE PASSWORD”. Once the user selects Login option by entering username and password, he/she is redirected to another window depending on the type of user i.e. whether the user is faculty or a student different window would show up.

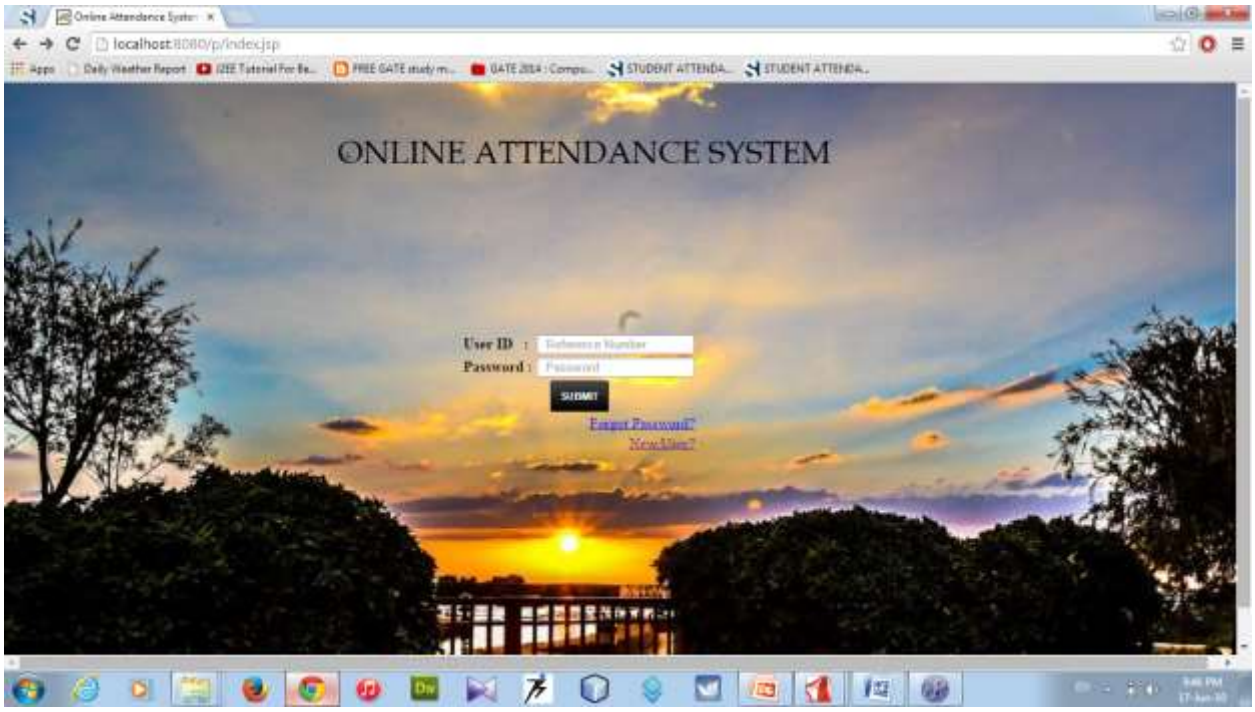

FIGURE 15: Login Window

Figure 12 shows the attendance page, where the faculty can mark the attendance of the student on the basis of the course, semester, and batch of the student. The attendance will be present for all the students initially and so the faculty would only have to mark absent.

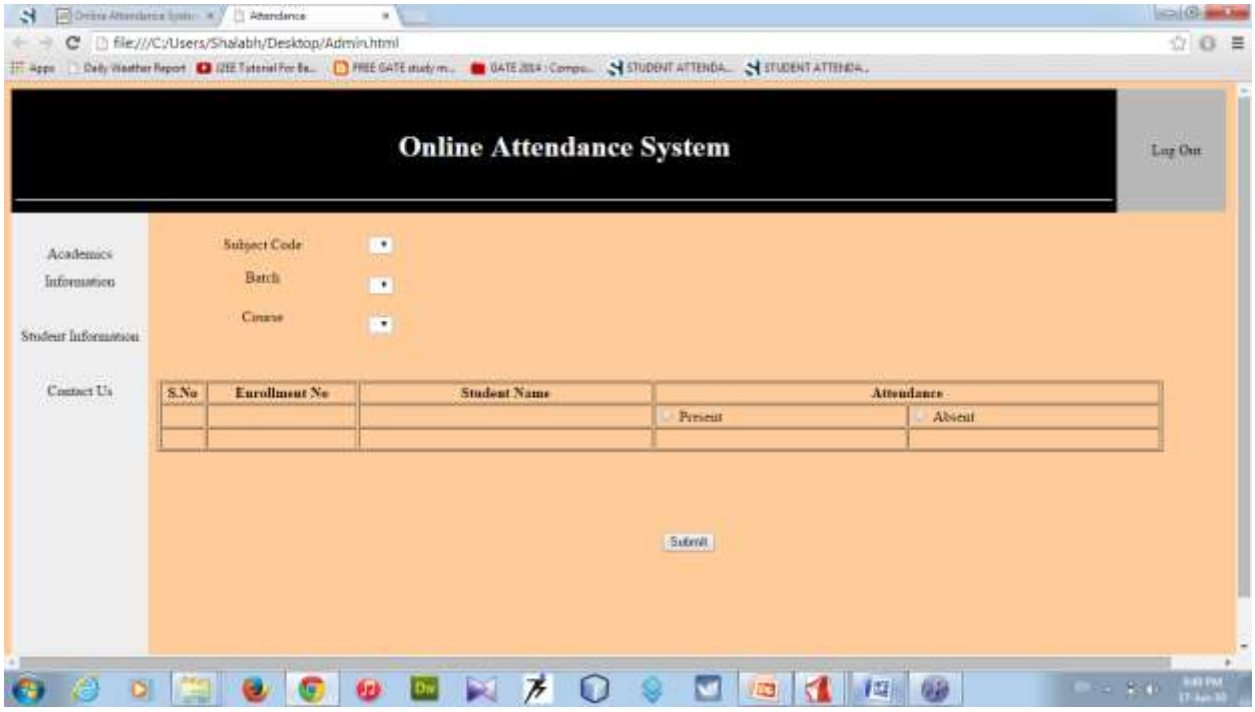

FIGURE 16: ATTENDANCE PAGE

Figure 13 shows the student page where the student can see his/her attendance percentage according to the subjects and courses selected. This page shows attendance for lecture, tutorial and lecture & tutorial combined.

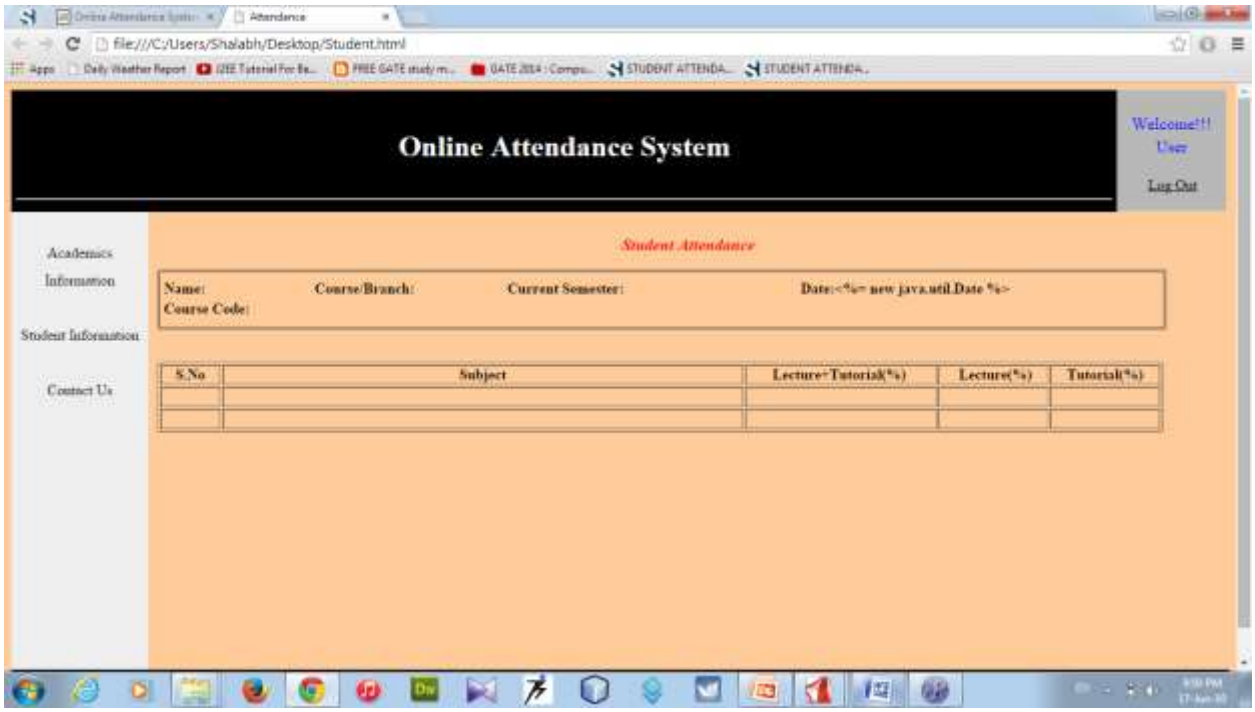

FIGURE 17: STUDENT PAGE

CHAPTER 4: CONCLUSION AND FUTURE SCOPE

4.1 SUMMARY

From all discussions or ideas that have been considered in this write-up about a student attendance management system, it can be seen that the project is automated attendance system. The system is designed using any preferred or assigned programming software: MyEclipse.

When the faculty enters a data of a student attendance, as results are being accessed, the automated attendance gives a percent in the continuous assessment of the student.

4.2 FUTURE SCOPE

An additional feature of sending e-mail to the parents of the students will be added, so as if the attendance of a student is below specified criteria an e-mail is sent to them regarding information about their child.

4.3 CONCLUSION

The Attendance Management System is developed using JSP. The system has reached a steady state where user is allowed to login. The new system\design it will improve on the overall performance on the management of student's attendance, thus the system solves the problem that it was intended to solve.

4.4 BIBLIOGRAPHY:

LINKS:

- http://msdn.microsoft.com/en-us/library/ms973829.aspx
- http://www.umsl.edu/~sauterv/analysis/488_f01_papers/quillin.htm
- http://www.tutorialspoint.com/uml/
- http://www.w3schools.com
- http://www.stackoverflow.com
- https://www.1keydata.com/sql/sql.html
- https://www.sqlcourse.com
- https://www.sql-tutorial.net
- https://www.codecademy.com/en/tracks/web
- https://cs.au.dk/~amoeller/papers/servlets/paper.pdf
- https://www.oracle.com/technetwork/articles/java/servlets-jsp-140445.html

TEXTBOOKS:

- OReilly Learning, "PHP MySQL and JavaScript", Jun 2009.
- The complete reference HTML & CSS Fifth Edition.
- Learn JavaScript ™ In a Weekend.
- Java Programming by UDIT AGARWAL.
- Software Engineering by IAN SOMMERVILLE
- Software Engineering by PRESSMAN
- UML user guide

4.5 APPENDIX:

index.jsp

```
<%@ page language="java" import="java.util.*" pageEncoding="ISO-8859-1"%>


<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>Online Attendance System</title>
        <meta http-equiv="pragma" content="no-cache">
        <meta http-equiv="cache-control" content="no-cache">
        <meta http-equiv="expires" content="0">
        <meta http-equiv="keywords" content="keyword1,keyword2,keyword3">
        <meta http-equiv="description" content="This is my page">
        <link href="./css/online.css" rel="stylesheet" type="text/css">
  </head>

  <body>
<%
String stud_id=request.getParameter("studid");%>
<%
if(stud_id!=null)
{
%>
<script>
alert("Registration Done!!");
</script>
<%
```

```
}
%>
  <div class="main">
  <div class="d1">
  <form action="login2.jsp" method="post">
  <table align="center" border=0 style="">
  <tr>
  <td><b>User ID    :  </b></td>
  <td><strong><input type="text" placeholder="  Reference  Number" name="id"></strong></td>
  </tr>
  <tr>
  <td><b>Password  :  </b></td>
  <td><input type="password" name="pwd" placeholder="  Password"></td>
  </tr>
  <tr><br>
  <td colspan="2" align="center"> <input class="button" type="submit" value="SUBMIT"></td>
  </tr>
  <tr>
  <td colspan="2" align="right"><a href="new_user_teach.jsp">New User?</a></td>
  </tr>
  <tr>
  <td colspan="2" align="right"><a href="forget_pwd.jsp">Forgot Password?</a></td>
  </tr>
  </table>
  </form>
  <%
  if(session.getAttribute("reply")!=null)
  {
  if(session.getAttribute("reply").equals("1"))
```

```
 {
 out.println("<span style='color:red'><b>Invalid User ID or Password !!</b></span>");
 }
 }
 %>
 </div>

<br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><
br><br><br><br><br><br><br><br><br><br><br><br><br>

 <hr size=2 color="black">

 <center><b>&copy; Shalabh Goel</b></center>

 </div>
 </body>
</html>>
```

login.jsp

```
<jsp:directive.page import="java.sql.*"/>
<jsp:scriptlet>
   String userid = request.getParameter("userid");
   String pwd = request.getParameter("pwd");
   Class.forName("com.mysql.jdbc.Driver");
   Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/online_att",
         "root", "shalabh");
   Statement st = con.createStatement();
   ResultSet rs;
   rs = st.executeQuery("select * from members where uname='" +userid+ "' and pass='"
+pwd+ "'");
   if (rs.next()) {
      session.setAttribute("userid", userid);
      //out.println("welcome " + userid);
      //out.println("<a href='logout.jsp'>Log out</a>");
      response.sendRedirect("success.jsp");

   } else {
      out.println("Invalid password <a href='stud_login.jsp'>try again</a>");

   }
</jsp:scriptlet>
```

login2.jsp

```
<%@ page import ="java.sql.*" %>
<%
   String userid = request.getParameter("userid");
   String pwd = request.getParameter("pwd");
   Class.forName("com.mysql.jdbc.Driver");
   Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/online_att",
         "root", "shalabh");
   Statement st = con.createStatement();
   ResultSet rs;
   rs = st.executeQuery("select * from members where uname='" +userid+ "' and pass='"
+pwd+ "'");
   if (rs.next()) {
      session.setAttribute("userid", userid);
      //out.println("welcome " + userid);
      //out.println("<a href='logout.jsp'>Log out</a>");
   response.sendRedirect("success1.jsp");

   } else {
      out.println("Invalid password <a href='Teacher_login.jsp'>try again</a>");

   }
%>
```

new_user.jsp

```
<%@ page language="java" import="java.util.*,java.sql.*,conn.*" pageEncoding="ISO-8859-1"%>


<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>

    <title>New User</title>

        <meta http-equiv="pragma" content="no-cache">
```

```
        <meta http-equiv="cache-control" content="no-cache">
        <meta http-equiv="expires" content="0">
        <meta http-equiv="keywords" content="keyword1,keyword2,keyword3">
        <meta http-equiv="description" content="This is my page">
        <!--
        <link rel="stylesheet" type="text/css" href="styles.css">
        -->
<link href="./css/online1.css" rel="stylesheet" type="text/css">
 </head>
 <body>
  <div class="main">
  <center><h1>ONLINE ATTENDANCE SYSTEM</h1></center><br>
  <h2>Registration Form</h2><br>
  <div class="d2">
  <form action="register.jsp" method="post">
  <fieldset><br>
  <legend><b>Personal details :</b><br></legend>
  <table width="78%"  border="0">
<tr>
        <td>First Name </td>
        <td><input  type="text"  placeholder="  First  Name"  name="fname"
size="50"></td></tr>
        <tr><td>Last Name  </td>
        <td><input  type="text"  placeholder="  Last  Name"  name="lname"
size="50"></td>
</tr>
<tr>
        <td>Year </td>
        <td><input type="text" placeholder="  YYYY" name="year" size="50"></td></tr>
        <tr><td>Batch </td>
        <td><input type="text" placeholder="  Batch" name="batch" size="50"></td>
```

```
</tr>

<tr>
        <td>Enrollment No : </td>
        <td><input type="text" placeholder="   Enrollment  No" name="studid"
size="50"></td></tr>
        <tr><td>Password </td>
        <td><input type="password" placeholder="  Login Password" name="passwd"
size="50"></td>


</tr>
</table>
</fieldset>



</table>
</fieldset>
 <table width="80%" border="0">
 <tr>
<td    colspan=4    align="center"><input    type="submit"    class="button"
value="Register"></td>
</tr>

 </table>
  </form>
  </div>

<br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><
br><br><br><br><br>
<hr size=2 color="black">
```

```
<center><b>&copy; 2014 Shalabh Goel. All Rights Reserved.</b></center>


  </div>

 </body>
</html>
```

registeration.jsp

```
<jsp:directive.page import="java.sql.*"/>

<jsp:scriptlet>
   String user = request.getParameter("uname");
   String pwd = request.getParameter("pass");
   String fname = request.getParameter("fname");
   String lname = request.getParameter("lname");
   String email = request.getParameter("email");
   Class.forName("com.mysql.jdbc.Driver");
   Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/online_att",
```

```
            "root", "shalabh");
    Statement st = con.createStatement();
    //ResultSet rs;
    int i = st.executeUpdate("insert into members(first_name, last_name, email, uname,
pass, regdate) values ('" + fname + "','" + lname + "','" + email + "','" + user + "','" + pwd
+ "', CURDATE())");
    if (i > 0) {
        //session.setAttribute("userid", user);
        response.sendRedirect("welcome.jsp");
       // out.print("Registration Successfull!"+"<a href='index.jsp'>Go to Login</a>");
    } else {
        response.sendRedirect("index.jsp");
    }
</jsp:scriptlet>
```

registeration2.jsp

```
<%@ page import ="java.sql.*" %>

<%
   String user = request.getParameter("uname");
   String pwd = request.getParameter("pass");
   String fname = request.getParameter("fname");
   String lname = request.getParameter("lname");
   String email = request.getParameter("email");
   Class.forName("com.mysql.jdbc.Driver");
   Connection con =
DriverManager.getConnection("jdbc:mysql://localhost:3306/online_att",
         "root", "shalabh");
   Statement st = con.createStatement();
   //ResultSet rs;
   int i = st.executeUpdate("insert into teacher(first_name, last_name, email, uname,
pass) values ('" + fname + "','" + lname + "','" + email + "','" + user + "','" + pwd + "' ");
   if (i > 0) {
      //session.setAttribute("userid", user);
      response.sendRedirect("welcome.jsp");
     // out.print("Registration Successfull!"+"<a href='index.jsp'>Go to Login</a>");
   } else {
      response.sendRedirect("index.jsp");
   }
%>
```

stud_login.jsp

```
<%@ page language="java" import="java.util.*" pageEncoding="ISO-8859-1"%>


<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>Online Attendance System</title>
        <meta http-equiv="pragma" content="no-cache">
        <meta http-equiv="cache-control" content="no-cache">
        <meta http-equiv="expires" content="0">
        <meta http-equiv="keywords" content="keyword1,keyword2,keyword3">
        <meta http-equiv="description" content="This is my page">
        <link href="./css/online.css" rel="stylesheet" type="text/css">
```

```
 </head>
 <body>
<%
String stud_id=request.getParameter("studid");%>
<%
if(stud_id!=null)
{
%>
<script>
alert("Registration Done!!");
</script>
<%
}
%>
 <div class="main">
 <div class="d1">
 <form action="login.jsp" method="post">
 <table align="center" border=1 style="">
 <tr>
 <td></td></tr>
 <tr>
 <td><b>User I          :        </b></td>
 <td><strong><input        type="text"              name="userid"        placeholder="Enter
Username"></strong></td>
 </tr>
 <tr>
 <td><b>Password    :        </b></td>
 <td><input type="password" name="pwd" placeholder="Enter Password" ></td>
 </tr>
 <tr><br>
```

```
 <td colspan="2" align="center"> <input class="button" type="submit"
value="SUBMIT"></td>
 </tr>
 <tr>
 <td colspan="2" align="right"><a href="Student_login.jsp">New User?</a></td>
 </tr>
 <tr>
 <td colspan="2" align="right"><a href="index.jsp">HOME</a></td>
 </tr>
 </table>
 </form>

 </div>

<br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><
br><br><br><br><br><br><br><br><br><br><br><br><br>

 <hr size=2 color="black">

 <center><b>&copy; Shalabh Goel</b></center>

 </div>
 </body>
</html>>
```

Teacher_login.jsp

```
<%@ page language="java" import="java.util.*" pageEncoding="ISO-8859-1"%>


<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
    <title>Online Attendance System</title>
        <meta http-equiv="pragma" content="no-cache">
        <meta http-equiv="cache-control" content="no-cache">
        <meta http-equiv="expires" content="0">
        <meta http-equiv="keywords" content="keyword1,keyword2,keyword3">
        <meta http-equiv="description" content="This is my page">
        <link href="./css/online.css" rel="stylesheet" type="text/css">
  </head>
```

```
 <body>
<%
String stud_id=request.getParameter("studid");%>
<%
if(stud_id!=null)
{
%>
<script>
alert("Registration Done!!");
</script>
<%
}
%>
 <div class="main">
 <div class="d1">
 <form action="login2.jsp" method="post">
 <table align="center" border=0 style="">
 <tr>
 <td><b>User ID   : </b></td>
 <td><strong><input type="text" placeholder=" Reference Number" name="id"></strong></td>
 </tr>
 <tr>
 <td><b>Password : </b></td>
 <td><input type="password" name="pwd" placeholder=" Password"></td>
 </tr>
 <tr><br>
 <td colspan="2" align="center"> <input class="button" type="submit" value="SUBMIT"></td>
 </tr>
```

```
<tr>
<td colspan="2" align="right"><a href="new_user_teach.jsp">New User?</a></td>
</tr>
<tr>
<td colspan="2" align="right"><a href="forget_pwd.jsp">Forgot Password?</a></td>
</tr>
</table>
</form>
<%
if(session.getAttribute("reply")!=null)
{
if(session.getAttribute("reply").equals("1"))
{
out.println("<span style='color:red'><b>Invalid User ID or Password !!</b></span>");
}
}
%>
</div>

<br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><br><
br><br><br><br><br><br><br><br><br><br><br><br><br>

<hr size=2 color="black">

<center><b>&copy; Shalabh Goel</b></center>

</div>
</body>
</html>>
```

success.jsp

```
<%
   if ((session.getAttribute("userid") == null) || (session.getAttribute("userid") == "")) {
%>
You are not logged in<br/>
<a href="stud_login.jsp">Please Login</a>
<%} else {
%>
<jsp:forward page="student_page.jsp">
<jsp:param name="userid" value='<%= session.getAttribute("userid")%>'/>
</jsp:forward>

<a href='logout.jsp'>Log out</a>
<%
   }
%>
```

success1.jsp

```
<%
   if ((session.getAttribute("userid") == null) || (session.getAttribute("userid") == "")) {
%>
You are not logged in<br/>
<a href="stud_login.jsp">Please Login</a>
<%} else {
%>
<jsp:forward page="attendence.jsp">
<jsp:param name="userid" value='<%= session.getAttribute("userid")%>'/>
</jsp:forward>

<a href='logout.jsp'>Log out</a>
<%
   }
%>
```