

# Chapter 1

## Preamble

### 1.1 Introduction:-

Generally, the notes are circulated on WhatsApp or any kind so it gets exceptionally troublesome to manage the important notes at the time of require. This framework will give a simple approach to share the documents for study purpose. Different users can work simultaneously on the framework. It'll be simple for the instructors to circulate the notes to each and every student. The framework will be utilized by students and instructors in colleges and indeed it can be utilized by schools.

There are numerous students who confront issue in studying at the exam time since either they don't have the notes given by instructors or they must have not been within the college.

This Framework will give a stage to easily access the notes.

The Objective of Notes management system is to provide better facility to the students and instructors to bring out the easy circulation of documents within healthy environment. It'll decrease the manual paperwork, reduced the sharing and distribution time.

## **1.2 Problem Statement:-**

There are numerous students who confront issue in studying at the exam time since either they don't have the notes given by instructors or they must have not been within the college. This Framework will give a stage to easily access the notes. The Objective of Notes management system is to provide better facility to the students and instructors to bring out the easy circulation of documents within healthy environment.

## **1.3 Objectives:-**

The Objective of Notes management system is to provide better facility to the students and instructors to bring out the easy circulation of documents within healthy environment. It'll decrease the manual paperwork, reduced the sharing and distribution time.

## **1.4 Feasibility Study**

The prime focus of the feasibility study is evaluating the practicality about the proposed system keeping in mind a number of factors. The following factors are taken into account before deciding in favour of the new application What we are going to develop in this project is very useful and necessary. As the life changing every day by day, everywhere computer knowledge is most important and internet browsing knowledge as well. This application will be helpful to both staff and the student. Since it provides some knowledge to the people it satisfies the operation feasibility.

### **1.4.1 Economical Feasibility**

This developing application is economically feasible with respect to cost. It also saves time of the users because they can directly access the information from this application without approaching anyone.

#### **1.4.2 Technical Feasibility**

The technical requirement for the system is economic and it does not require any additional hardware and software.

#### **1.4.3 Behavioral Feasibility**

Using this application is quite easier as compared to the present system student can gain knowledge and does not require any special training to work on this application.

## Chapter 2

### System Study

#### 2.1 Existing System:-

Mostly the notes are circulated on WhatsApp or any kind so it gets very difficult to manage the important notes at the time of need. There are many students who face problem in studying at the exam time because either they don't have the notes provided by teachers or they must have not been in the college. This System will provide a platform to easily access the notes.

It has always been the tedious process to borrow the notes from the senior. It is very hard to find the right person who can employ with their notes they have passed. Note circulation system is eradication of this problem. It gives the platform where the person is recommended to the user as per their note's requirement.

#### 2.2 Proposed System:-

This system will provide an easy approach to share the documents for studying purpose. Multiple users can work simultaneously on the system. It will be easy for the teachers to circulate the notes to each and every student. The system will be used by students and teachers in colleges and even it can be used by schools.

The Objective of Class Notes Gallery is to provide better facility to the students and teachers to bring out the easy circulation of documents within healthy environment. It will reduce the manual paperwork, reduced the sharing and distribution time.

### **2.3 Advantages:-**

- This system will provide an easy approach to share the documents for studying purpose.
- It is to provide better facility to the students and teachers to bring out the easy circulation of documents within healthy environment.
- It will be easy for the teachers to circulate the notes to each and every student.
- Time will be saved by this method.
- It will reduce the manual paperwork

# Chapter 3

## Requirement

### 3.1 Hardware Requirements:-

- Processor :- core i3 or Above
- RAM :- 512 mb or 2 More
- HardDisk :- 15 Gb or More.
- Processing Speed :- 2.6 GHz

### 3.2 Software Requirements:-

- Operating system :- windows 7 or above
- Front End :- HTML , JavaScript , CSS
- Server :- Apache Tomcat 5 or above
- IDE :- Adobe Dreamweaver
- Database :- MySQL

### 3.3 Software Requirement Specification:-

#### Functional Requirements:-

##### Scenario:- Staff

- The staff is the only person to add the notes.
- He can also view the notes.
- He can update his profile, If he want to change any detailes.
- He can view the feedback given by the students.
- He can also change is password whenever needed.
- Logout after completion of this work.

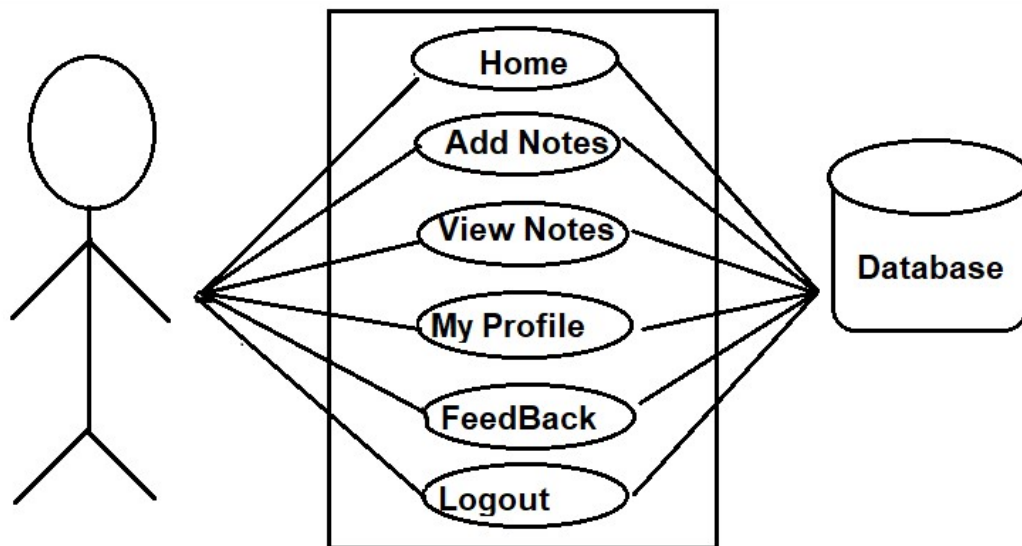


Fig 3.3.1 Use Case Diagrams For STAFF

**Scenario:- Student**

- He can view the notes which are uploaded by the staff.
- He can give the feedback for the notes that are uploaded.
- He can update his profile, If he want to change any detailes.
- He can also change is password whenever needed.
- Logout after completion of this work.

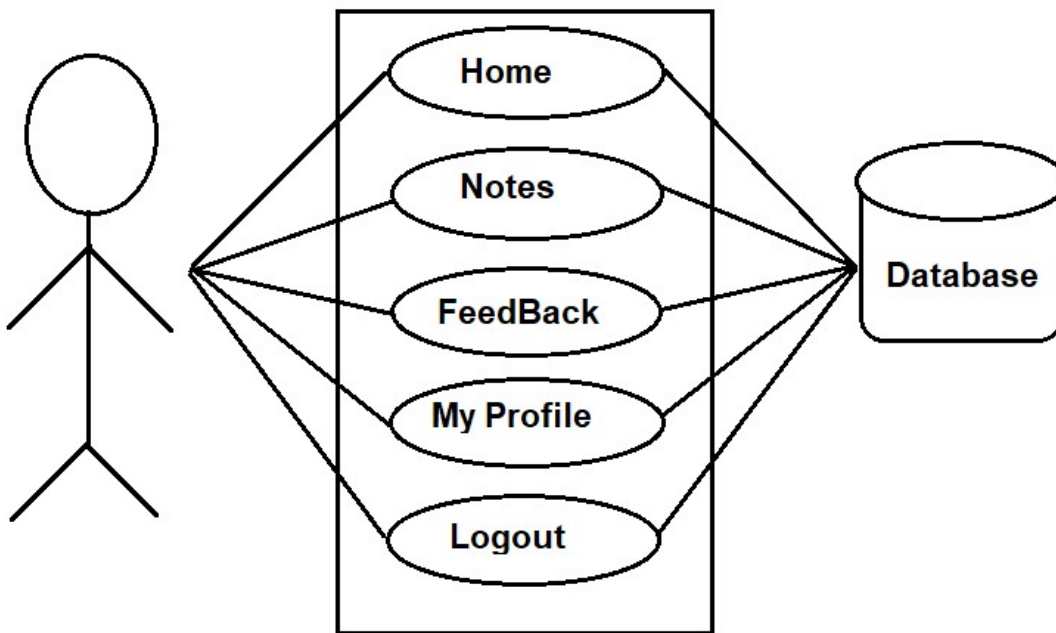


Fig 3.3.2 Use Case Diagrams For STUDENT



### **3.4 Non-Functional Requirements:-**

#### **3.4.1 Performance requirements:-**

1. The overall execution is optimized and user should get convenient GUI with responsive.
2. The user must be able to configure all the parameters which are required to accomplish product.
3. System should be user friendly.

#### **3.4.2 Safety requirements :-**

1. Confidential information should be protected by unauthorized user.
2. The application should be verify the users key before download any files from the cloud.

#### **3.4.3 Security Requirements :-**

1. If the user of the cloud forget the password security question will be asked.
2. If the user's key mismatched the application should provide hints to get back.

#### **3.4.4 Software Quality Attributes :-**

1. Portability: The software is developed in java it can be executed on any platform, for which the JVM is available with minor or no modifications. So it is 90 percent flexible for any platform.
2. Ease of Use: The front end is using HTML, JAVASCRIPT, CSS and it is a desktop utility.
3. Modularity: The whole product is divided into three modules for easy to detect and debug the errors occurred in respected module.

# Chapter 4

## SYSTEM DESIGN

### 4.1 Fundamental Design Concept:-

System design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. System design could be seen as the application of the system theory to product development.

### Modules Details :-

The whole Project is divided into following Modules.

I. Staff

II. Student

### 4.2 Modules Description :-

#### I. Staff :-

He is an super user, who can add the notes ,view the notes. He can see the feedback of the students for his notes.

If any problem in the notes, he can work on that and can get best response by the students.

#### II. Student :-

He can only view/see the notes which are uploaded by the staff,but he cannot add the notes.

He can give the feedback to the notes given by the staff,so they can improve in their work if they have issues.

### 4.3 Data dictionary :-

#### Add Notes

Name	Type
<b>id</b> 	int(11)
<b>sname</b>	varchar(40)
<b>s_id</b>	varchar(30)
<b>department</b>	varchar(30)
<b>pdf</b>	varchar(100)
<b>disc</b>	varchar(100)

#### Feedback

Name	Type
<b>id</b> 	int(30)
<b>email</b>	varchar(30)
<b>department</b>	varchar(30)
<b>sub</b>	varchar(30)
<b>topic</b>	varchar(30)
<b>discription</b>	varchar(30)

## Staff registration

Name	Type
<b>name</b>	varchar(30)
<b>staff_id</b> 	varchar(30)
<b>gender</b>	varchar(20)
<b>address</b>	varchar(30)
<b>department</b>	varchar(30)
<b>mobile</b>	varchar(300)
<b>email</b> 	varchar(30)

## Student redistratation

<b>fname</b>	varchar(40)
<b>lname</b>	varchar(40)
<b>dob</b>	varchar(40)
<b>gender</b>	varchar(40)
<b>email</b> 	varchar(40)
<b>mobile</b>	int(40)
<b>degree</b>	varchar(40)
<b>reg</b> 	varchar(40)
<b>address</b>	varchar(40)
<b>pincode</b>	varchar(40)

## Users

Name	Type
id 	int(10)
email 	varchar(30)
password	varchar(30)
utype	varchar(30)

### 4.4 Data flow diagram (DFD) :-

A data flow diagram (DFD) is a significant modelling technique for analysing and constructing information processes. Data-flow diagram (DFD) is a graphical representation of the "flow" of data through an information system. DFDs can also be used for the visualization of data processing (structured design). On a DFD, data items flow from an external data source or an internal data store to an Internal data store or an external data sink, via an internal process. A DFD provides no information about the timing or ordering of processes, or about whether processes will operate in sequence or in parallel. It is therefore quite different from a flowchart

This shows the flow of control through an algorithm, allowing a reader to determine what operations will be performed, in what order, and under what circumstances, but not what kinds of data will be input to and output from the system, nor where the data will come from and go to, nor where the data will be stored (all of which are shown on a DFD). Data-flow diagrams provide the end user with a physical idea of where the data they input ultimately has an effect upon the structure of the whole system from order to dispatch to report. How any system is developed can be determined through a data-flow diagram.

With a data-flow diagram, users are able to visualize how the system will operate, what the system will accomplish, and how the system will be implemented. A designer usually draws a context-level DFD showing the relationship between the entities inside and outside of a system as one single step.

This basic DFD can be then disintegrated to a lower level diagram demonstrating smaller steps exhibiting details of the system that is being modelled. Numerous levels may be required to explain a complicated system. The different versions are Context Diagrams (Level 0), Partitioned Diagrams (single process only -- one level), functionally decomposed, levelled sets of Data Flow Diagrams.

#### **4.4.1 Data Flow Diagrams Symbols:**

A **DFD** usually comprises of four components. These four components can be represented by four simple symbols. These symbols can be explained in detail as follows: External entities (source/destination of data) are represented by squares; Processes (input-processing-output) are represented by rectangles with rounded corners; Data Flows (Physical or electronic data) are represented by arrows; and finally, Data Stores (physical or electronic like XML files) are represented by open-ended rectangles.

##### **Data store**



Or



A data store stores data passively for later access. A data store responds to requests to store and access data. It does not generate any operations.

A data store allows values to be accessed in an order different from the order in which they were generated.

Input flows indicate information or operations that modify the stored data such as adding or deleting elements or changing values. Output flows indicate information retrieved from the store; this information can be an entire value or a component of a value.

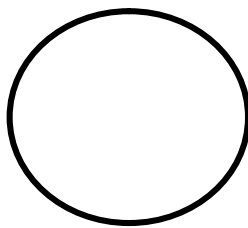
### **Data flow**



A data flow moves data between processes or between processes and data stores. As such, it represents a data value at some point within a computation and an intermediate value within a computation if the flow is internal to the diagram. This value is not changed. The names of input and output flows can indicate their roles in the computation or the type of the value they move.

Data names are preferably nouns. The name of a typical piece of data, the data aspect, is written alongside the arrow.

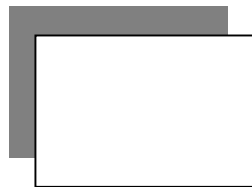
### **Data Process**



A process shows a transformation or manipulation of data flows within the system. The symbol used is rounded rectangle or oval. A descriptive title is placed in the center of the circle. This should be a simple imperative sentence with a specific verb, for example 'maintain customer records' or 'find driver'. E.g.: May be a clerk computing discounts or a combination of manual and electronic activities.

**Source or Sink**

OR



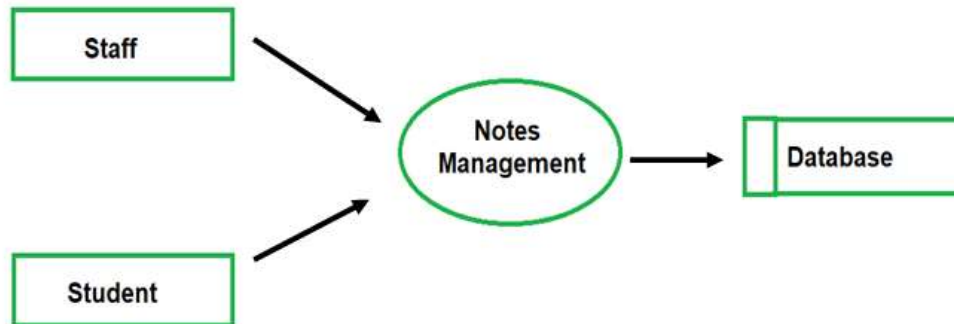
An external entity is a source or destination of a data flow which is outside the area of study. The symbol used is a rectangle or square containing a meaningful and unique identifier.

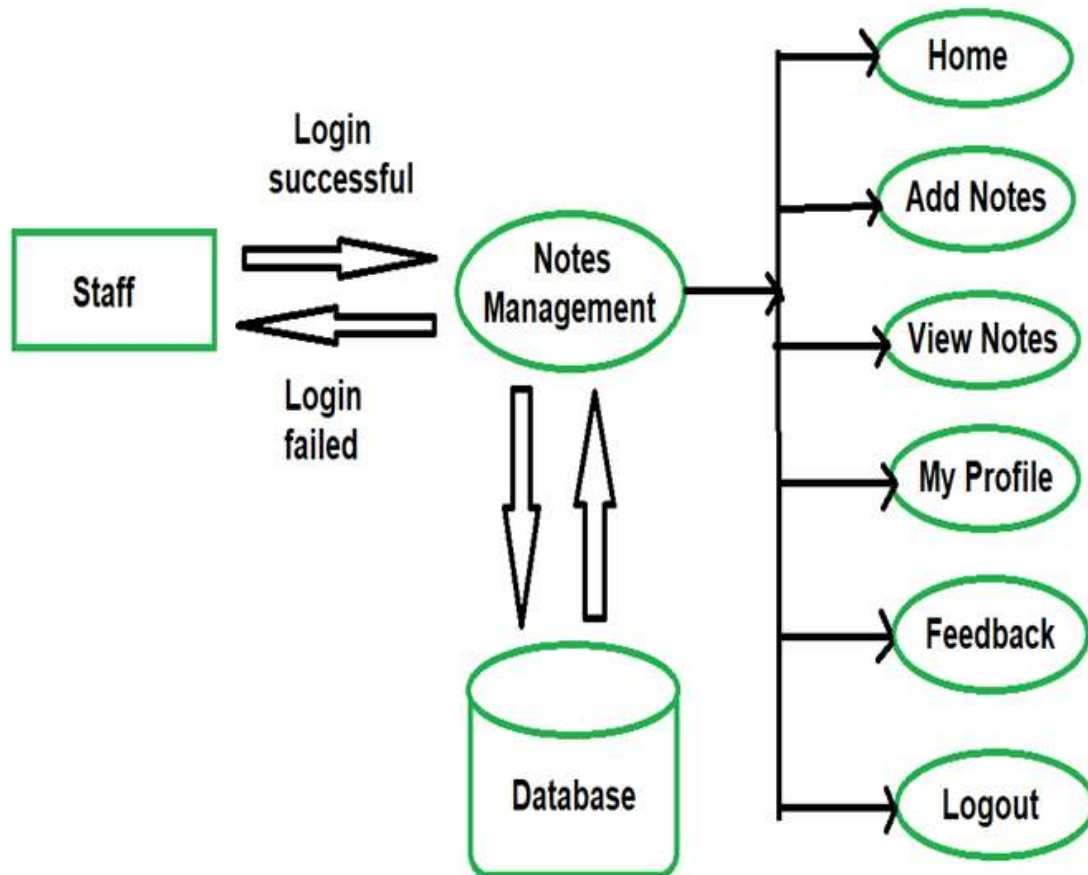
Only those entities which originate or receive data are represented on a business process diagram E.g.: May be one customer or a number of customers with transactions (orders)

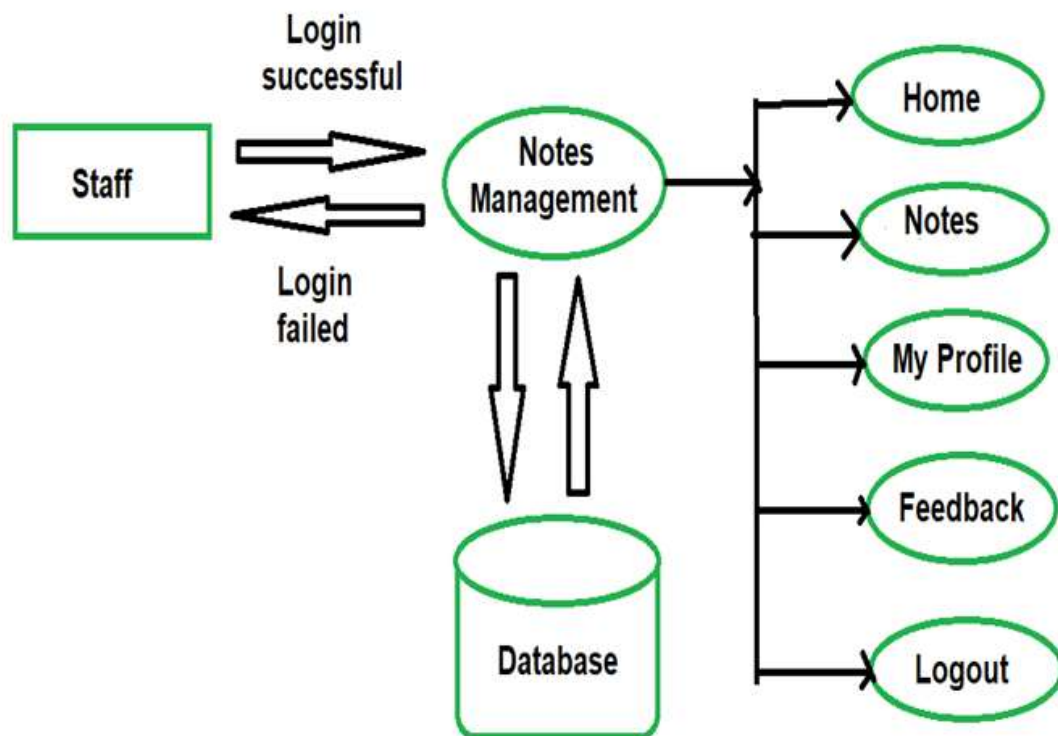


#### 4.5 DFD of Modules:-

##### Level 0 DFD



**LEVEL 1 DFD****Staff :-**

**LEVEL 1 DFD****Student :-**

## 4.6 ER DIAGRAM:-

An entity-relationship diagram is a data modeling technique that creates a graphical representation of the entities, and the relationships between entities, within an information system. An **entity-relationship model (ERM)** is an abstract and conceptual representation of data.

Entity-relationship modeling is a database modeling method, used to produce a type of conceptual schema or semantic data model of a system, often a relational database, and its requirements in a top-down fashion. Diagrams created by this process are called **entity-relationship diagrams, ER diagrams, or ERDs.**

### **The three main components of an E-R Diagram are:**

- The entity is a person, object, place or event for which data is collected. For example, if you consider the information system for a business, entities would include not only customers, but the customers address, and orders as well. The entity is represented by a rectangle and labeled with a singular noun.
- The relationship is the interaction between the entities. In the example above, the customer places an order, so the word “places” defines the relationship between that instance of a customer and the order or orders that they place. A relationship may be represented by a diamond shape, or more simply, by the line connecting the entities. In either case, verbs are used to label the relationships.
- The cardinality defines the relationship between the entities in terms of numbers. An entity may be optional: for example, a sales representative could have no customers or could have one or many customers; or mandatory: for example, there must be at least one product listed in an order. There are several different types of cardinality notations;

crow's foot notation, used here, is a common one. In crow's foot notation, a single bar indicates one, a double bar indicates one and only one (for example, single instance of a product can only be stored in one warehouse), a circle indicates zero, and a crow's foot indicates many.

➤ The three main cardinal relationships are: one-to-one, expressed as 1:1; one-to-many, expressed as 1: M; and many-to-many, expressed as M: N. Entity Relationship Diagram Notations:

**Entity:**

An entity is an object or concept about which you want to store information. An entity is a real-world item or concept that exists on its own. The set of all possible values for an entity, such as all possible customers, is the entity type. In an ER model, we diagram an entity type as a rectangle containing the type name.

**Weak Entity:**

A weak entity is an entity that must be defined by a foreign key relationship with another entity as it cannot be uniquely identified by its own attributes alone.

**Attribute:**

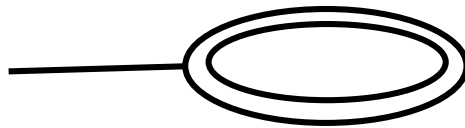
Each entity has attributes, or particular properties that describe the entity. Most of the data in a database consists of values of attributes. The set of all possible values of an attribute is the attribute domain. In an ER model, an attribute name appears in an oval that has a line to the corresponding entity box.

**Key attributes:**

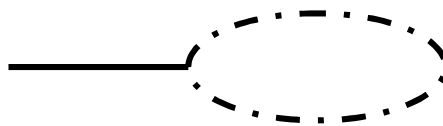
A key attribute is the unique, distinguishing characteristic of the entity. An attribute or set of attributes that uniquely identifies a particular entity is a key. A key attribute in an ER Diagram is represented by an oval that has a line inside it and a line to the corresponding entity box. For example, an employee's social security number might be the employee's key attribute.

**Multi valued attribute:**

A multi valued attribute can have more than one value. We indicate this with a double oval. For example, an employee entity can have multiple skill values.

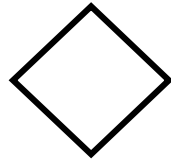
**Derived attribute:**

A derived attribute is based on another attribute. It is denoted by an oval and dotted line within it. For example, an employee's monthly salary is based on the employee's annual salary.

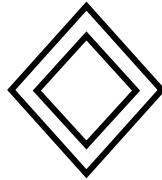


**Relationships:**

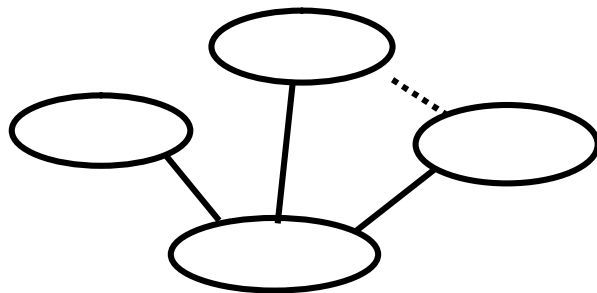
Relationships illustrate how two entities share information in the database structure. An association among entities is called a relationship. An attribute can also be a property of a relationship set. The association among the entities is described as one-to-one, one-to-many, many-to-many. A relationship is indicated by a rhomb

**Identifying relationship:**

Identifying relationship is denoted by double rhombus.

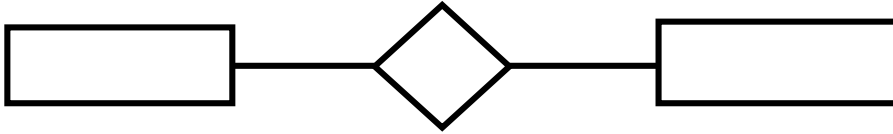
**Composite Attribute :**

A composite attribute has multiple components and each component is atomic or composite. We illustrate this composite nature in the ER model by branching off the component attributes.



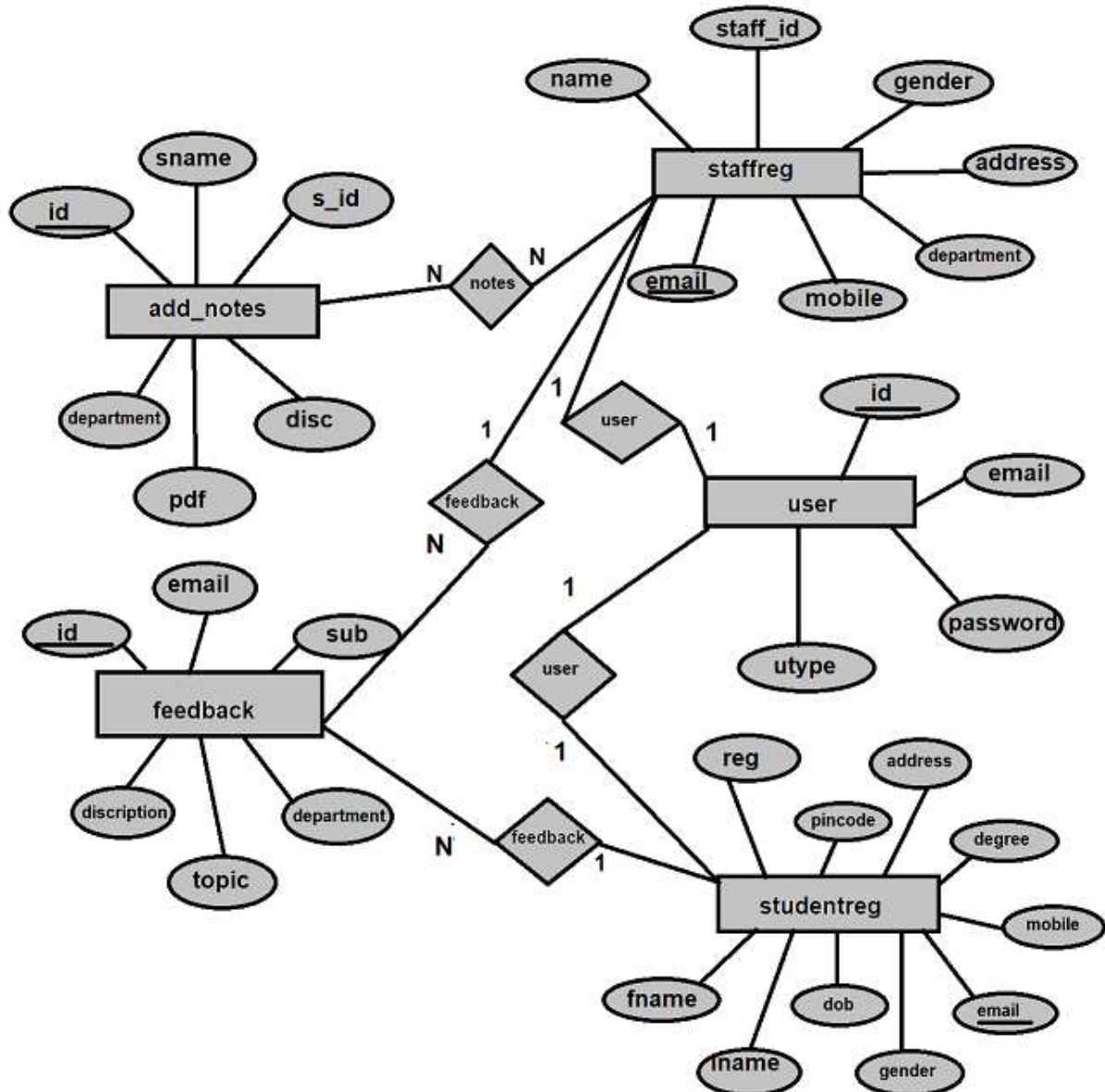
**Total Participation:**

Total participation is represented by a double line.





## Er diagram



# Chapter 5

## IMPLEMENTATION

### 5.1 Introduction of Language

**HTML: Hyper Text Mark-up Language**, commonly referred to as **HTML**, is the standard mark-up language used to create web pages. Along with CSS and JavaScript, HTML is a cornerstone technology used to create web pages as well as to create user interfaces for mobile and web applications. Web browsers can read HTML files and render them into visible or audible web pages. HTML describes the structure of a website semantically pages. HTML describes the structure of a programming language.

HTML elements form the building blocks of HTML pages. HTML allows images and other objects to be embedded and it 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. HTML elements are delineated by tags, written using angle brackets. Tags such as `<img />` and introduce content into the page directly. Others such as...surrounded provide information about document text and may include other tags as sub-elements. Browsers do not display the HTML tags, but use them to interpret the content of the page.

HTML can embed scripts written in languages such as Java scripts which affect the

behaviour of HTML web pages. HTML mark-up can also refer the browser 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, has encouraged the use of CSS over explicit presentational HTML since 1997.

HTML is a mark-up language that web browsers use to interpret and compose text, images, and other material into visual or audible web pages. Default characteristics for every item of HTML mark-up are defined in the browser, and these characteristics can be altered or enhanced by the web page designer's additional use of CSS. . Many of the text elements are found in the 1988 ISO technical report TR 9537 Techniques for using SGML, which in turn covers the features of early text formatting languages such as that used by the RUNOFF command developed in the early 1960s for the CTSS (Compatible Time-Sharing System) operating system: these formatting commands were derived from the commands used by typesetters to manually format documents. However, the SGML concept of generalized mark-up is based on elements (nested annotated ranges with attributes) rather than merely print effects, with also the separation of structure and mark-up; HTML has been progressively moved in this direction with CSS

It is the mark-up "code" that is read by web browsers like Internet Explorer, Firefox, Safari, etc., either as straight HTML or a newer, XML version of HTML called XHTML. The difference between these versions is beyond the scope of these notes. All the "experts" can't even seem to agree when you can or should use XHTML. Just know that if you use either, any web browser will be able to display your pages for some time to come. I will use both, but try to emphasize

XHTML, because it makes your pages ready for stuff we haven't thought up yet. The .html extension is what turns a regular text file into an HTML file. It only needs to be added at the time you create the file, once it's made simply save the file each time you edit the code. Save your file in a folder where you can easily find it.

The saved code can be opened and edited from within the text editor (make sure that "All Files" is selected from the drop down menu in the editor's dialog box). To open the webpage in your browser simply go to the folder in which it is saved and click on the webpage icon.

## **CSS:**

Stands for “Cascading Style Sheet”. Cascading Style Sheets are used to format the layout of Web pages. They can be used to define text styles, table sizes, and other aspects of Web pages that previously could only be defined in a page’s HTML.

CSS helps Web developers create a uniform look across several pages of Website. Instead of defining the style of each table and each block of text within a page’s HTML, commonly used styles need to be defined only once in a CSS document. Once the style is defined in cascading style sheet, it can be used by any page that references the CSS file. Plus, CSS makes it easy to change styles across several pages at once.

## **MySQL:**

**MySQL** is an open source relational database system(RDBMS); in July 2013, it was

the world's second most widely used RDBMS, and the most widely used open-source client server model RDBMS. It is named after co-founder Michael Widenius's daughter, My. The SQL abbreviation stands for Structured Query Language. . The MySQL development project has made its source code available under the terms of the General Public License as well as under a variety of proprietary agreements. MySQL was owned and sponsored by a single for profit firm, the Swedish company MYSQL AB now owned by Oracle Corporation. For proprietary use, several paid editions are available, and offer additional functionality.

MySQL is a popular choice of database for use in web applications, and is a central component of the widely used LAMP open-source web application software stack (and other "AMP" stacks). LAMP is an acronym for "Linus, Apache, My sql, PERL/PHP/Python". Free software open-source projects that require a full-featured database management system often use MySQL. Applications that use the MySQL database include: TYPO3, MODx, Joomla, Word Press, pHp BB, My BB,

Drupal and other software. MySQL is also used in many high-profile, large-scale websites, including Google (though not for searches), Facebook , Twitter, Flickr and You tube .

On all platforms except Windows, MySQL ships with no GUI tools to administer MySQL databases or manage data contained within the databases. Users may use

the included command line tools or install MY SQL Workbench via a separate download. Many third party GUI tools are also available.

MySQL is written in C and C++ Its SQL parser is written in yacc but it uses a home-

brewed lexical analyser. MySQL works on many system platforms including AIX, BCDi, FreeBSD, HP-UX, ECOM station, i5/os, IRIX, Linux, OS X, Microsoft Windows,

Net BSD, Novell Net Ware, Open BSD, Open Solaris, OS/2 Warp, QNX Oracle Solaris, Symbian, Sun OS, SCO open server, SCO Unix Ware, Sansom and Tru64. A port of MySQL to open VMS also exists.

The MySQL server software itself and the client libraries use dual licensing distribution. They are offered under GPL version 2, beginning from 28 June 2000 (which in 2009 has been extended with a FLOSS License Exception) or to use a proprietary license. Support can be obtained from the official manual. Free support additionally is available in different IRC channels and forums. Oracle offers paid support via its MySQL Enterprise products. They differ in the scope of services and in price. Additionally, a number of third party organizations exist to provide support and services, including MariaDB and Persona.

MySQL has received positive reviews, and reviewers noticed it "performs extremely well in the average case". And that the "developer interfaces are there, and the documentation (not to mention feedback in the real world via Web sites and the like) is very, very good". It has also been tested to be a "fast, stable and true multi-user, multi-threaded MySQL database server"

### **5.1.5 Why You Need XXAMP, MySQL, and PHP?**

PHP is a powerful scripting language that can be run by itself in the command line of any computer with PHP installed. However, PHP alone isn't enough in order to build dynamic web sites. To use PHP on a web site, you need a server that can process PHP scripts. WAMP server allows developers to test PHP scripts locally; this makes it an invaluable piece of your local development environment.

Additionally, dynamic websites are dependent on stored information that can be and easily; this is the main difference between a dynamic site and a static HTML site. However, PHP doesn't provide a simple, efficient way to store data. This is where a relational database management system like MySQL comes into play.

### **5.2 PHP:**

PHP originally stood for “Personal Home Page” and was released as a free, open source project. Over time, the language was reworked to meet the needs of its users. In 1997, PHP was renamed to the current “PHP: Hypertext Preprocessor.” PHP is generally used as a server-side scripting language; it is especially well-suited for creating dynamic web pages and client-side GUI applications. PHP generally runs on a web server, taking PHP code as its input and creating web pages as output. The scripting language features integrated support for interfacing with databases such as MySQL, which makes it a prime candidate for building all manner of web applications, from simple personal web sites to complex enterprise-level applications.

Unlike HTML, which is parsed by a browser when a page loads, PHP is pre-

processed by the machine that serves the document (this machine is referred to as a server). All PHP code contained with the document is processed by the server before the document is sent to the visitor's browser. PHP is a scripted language, which is another great advantage for PHP programmers. PHP can be deployed on most web servers, many operating systems and platforms, and can be used with many relational database management systems. It is available free of charge, and the PHP Group provides the complete source code for users to build, customize and extend for their own use. Many programming languages require that you compile files into machine code before they can be run, which is a time-consuming process. Bypassing the need to compile means you're able to edit and test code much more quickly. Because PHP is a server-side language, running PHP scripts on your local machine requires installing a server on your local machine. PHP is free software released under the PHP License; however it is incompatible with the GNU General Public License (GPL), due to restrictions on the usage of the term PHP. It is a widely-used general-purpose scripting language that is especially suited for web development and can be embedded into HTML. It generally runs on a web server, taking PHP code as its input and creating web pages as output. It can be deployed on most web servers and on almost every operating system and platform free of charge. PHP is installed on more than 20 million websites and 1 million web servers.



**5.2.1 Usage:**

PHP is a general-purpose scripting language that is especially suited for web development. PHP generally runs on a web server, taking PHP code as its input and creating web pages as output. It can also be used for command-line scripting and client-side GUI applications. PHP can be deployed on most web servers, many operating systems and platforms, and can be used with many relational database management systems. It is available free of charge, and the PHP Group provides the complete source code for users to build, customize and extend for their own use.

PHP primarily acts as a filter, taking input from a file or stream containing text and/or PHP instructions and outputs another stream of data; most commonly the output will be HTML. It can automatically detect the language of the user. From PHP 4, the PHP parser compiles input to produce byte code for processing by the Zend Engine, giving improved performance over its interpreter predecessor. Originally designed to create dynamic web pages, PHP's principal focus is server side scripting, and it is similar to other server-side scripting languages that provide dynamic content from a web server to a client, such as Microsoft's Active Server Pages, Sun Microsystems' Java Server Pages, and mod\_perl. PHP has also attracted the development of many frameworks that provide building blocks and a design structure to promote rapid application development (RAD). Some of these include CakePHP, Symphony, Code Igniter, and Zend Framework, offering features similar to other web application frameworks.

The LAMP architecture has become popular in the web industry as a way

deploying web applications. PHP is commonly used as the P in this bundle alongside Linux, Apache and MySQL, although they may also refer to Python or Perl. As of April 2007, over 20 million Internet domains were hosted on servers with PHP installed, and PHP was recorded as the most popular Apache module. Significant websites are written in PHP including the user-facing portion of Facebook, Wikipedia (MediaWiki), and Yahoo!, My Yearbook, Digg, Word press and Tagged. In addition to server-side scripting, PHP can be used to create stand-alone, compiled applications and libraries, it can be used for shell scripting, and the PHP binaries can be called from the command line.

### **5.2.2 Speed Optimization:**

As with many scripting languages, PHP scripts are normally kept as human-readable source code, even on production web servers. In this case, PHP scripts will be compiled at runtime by the PHP engine, which increases their execution time. PHP scripts are able to be compiled before runtime using PHP compilers as with other programming languages such as C (the language PHP and its extensions are written in). Code optimizers aim to reduce the computational complexity of the compiled code by reducing its size and making other changes that can reduce the execution time with the overall goal of improving performance. The nature of the PHP compiler is such that there are often opportunities for code optimization, and an example of a code optimizer is the Zend Optimizer PHP extension.

Another approach for reducing overhead for high load PHP servers is using PHP

accelerators. These can offer significant performance gains by caching the compiled form of a PHP scriin shared memory to avoid the overhead of parsing and compiling the code every time the script runs.

## 5.3 SOURCE CODE

### Database Connection:-

```
<?php
$server="localhost";
$username="root";
$password="";
$databasename="notes";
$link=mysqli_connect($server,$username,$password,$databasename);
//mysql_connect();
if($link===false){
die("ERROR:could not connect.".mysqli_connect_error());
}
```

**LOG\_CHECK :-**

```
<?php
session_start();
require_once('config.php');
$username=$_POST['email'];
$password=$_POST['password'];
$_SESSION['user']=$username;
$ss="select * from users where email='$username' ";
$rs=mysqli_query($link,$ss);
$row=mysqli_fetch_array($rs);
if(empty($row))
{
?>
<script>
alert("Invalid Username");
document.location="login.html";
</script>
<?php
}
else
{
$pass=$row['password'];
$utype=$row['utype'];
if($pass==$password)
{
if($utype=="staff")
{
//header("location :admin_home.html")
```

```
?>

<script>
alert("You have Logged in as Staff");
document.location="staff_home.html";
</script>
<?php
}
if($utype=="student")
{
?>
<script>
alert("You have Logged in as student");
document.location="student_home.html";
</script>
<?php
}
}
else
{
?>
<script>
alert("Invalid Password");
document.location="login.html";
</script>
<?php
}
}??>
```

**Home Page:-**

```

<html lang="en">
<head>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-
scale=1.0">
<title>notes management</title>
<link rel="stylesheet" href="bootstrap/bootstrap.css.css">
<link rel="stylesheet" href="bootstrap/responsive.css">
<link rel="stylesheet" href="bootstrap/style.css">
<link rel="stylesheet" href="bootstrap/font-awesome.min.css">
<link rel="stylesheet" href="css\home.css">
<style>
body{
background-image: url("img/notelimg.jpg");
width: 100%;
height: 100%;
background-size: cover;
background-repeat: no-repeat;
}
</style>
</head>
<body>
<nav class="logo navbar-expand-md navbar-dark">
<nav>
<button class="navbar-toggler" type="button" data-
toggle="collapse" data-target="#collapsibleNavbar">
<span class="navbar-toggler-icon" ></span>

```

```

</button>

<div class="collapse navbar-collapse" id="collapsibleNavbar">
  <ul class="navbar-nav">
    <li class="logo">
    <li><a class="nav active" href="home.html">Home</a></li>
    <li> <a class="nav" data-toggle="dropdown" href="">Register</a>
    <div class="dropdown-menu">
      <a class="dropdown-item bg-warning text-white "
      href="staff.html" ><h3>Staff</h3></a>
      <div class="dropdown-divider "></div>
      <a class="dropdown-item bg-warning text-white"
      href="student.html" bg="dark"><h3>Student</h3></a>
    </div></li>
    <li><a class="nav" href="login.html">Login</a></li>
      <li><a class="nav" href="notes.html">Notes</a></li>
      <li><a class="nav" href="about.html">About</a></li>
    </ul>
  </div>
</nav>
</nav>

<!-- ALL JS FILES -->
<script src="bootstrap/jquery-3.2.1.min.js"></script>
<script src="bootstrap/popper.min.js"></script>
<script src="bootstrap/bootstrap.min.js"></script>
<!-- ALL PLUGINS -->
<script src="bootstrap/jquery.superslides.min.js"></script>
<script src="bootstrap/images-loded.min.js"></script>
<script src="bootstrap/isotope.min.js"></script>
<script src="bootstrap/baguetteBox.min.js"></script>
<script src="bootstrap/form-validator.min.js"></script>

```



```
<script src="bootstrap/contact-form-script.js"></script>  
<script src="bootstrap/custom.js"></script>  
</body>  
</html>
```

**Add Notes :-**

```
<html lang="en">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

<meta name="viewport" content="width=device-width, initial-
scale=1.0">

<title>notes management</title>

<link rel="stylesheet" href="bootstrap/bootstrap.css.css">
<link rel="stylesheet" href="bootstrap/classic.css">
<link rel="stylesheet" href="bootstrap/responsive.css">
<link rel="stylesheet" href="bootstrap/style.css">
<link rel="stylesheet" href="bootstrap/font-awesome.min.css">
<link rel="stylesheet" href="css/home.css">

<style>
body
{
background-image: url("img/sl2.jpeg");
background-size: cover;
border: 5px;
}
img{
max-width:100%;
max-height:100%;
}
.R1 {
color:black;
font-size:40px;
```

```

font-family:serif;
}
</style>
</head>
<body>
<nav class="logo navbar-expand-md navbar-dark">
<nav> <button class="navbar-toggler type="button" data-
toggle="collapse"
data-target="#collapsibleNavbar">
<span class="navbar-toggler-icon" ></span>
</button>
<div class="collapse navbar-collapse" id="collapsibleNavbar">
<ul class="navbar-nav">
<li class="logo">
<li><a class="nav" href="staff_home.html">Home</a></li>
<li><a class="active" href="staff_add.php">Add Notes</a></li>
<li><a class="nav" href="view_notes.php">View Notes</a></li>
<li><a class="nav" href="profile.php">My Profile</a></li>
<li><a class="nav" href="staff_feedback.php">FeedBack</a></li>
<li><a class="nav" href="logout.php">Logout</a></li>
</ul>
</nav>
<div class="container">
<br/><br/>
<center>
<h3>Add Notes</h3>
<table cellpadding="30" cellspacing="50">
<form action="staff_insertnotes.php" method="POST"
enctype="multipart/form-data">

```

[illegible]

```
<script src="bootstrap/popper.min.js"></script>
<script src="bootstrap/bootstrap.min.js"></script>
<!-- ALL PLUGINS -->
<script src="bootstrap/jquery.superslides.min.js"></script>
<script src="bootstrap/images-loded.min.js"></script>
<script src="bootstrap/isotope.min.js"></script>
<script src="bootstrap/baguetteBox.min.js"></script>
<script src="bootstrap/form-validator.min.js"></script>
<script src="bootstrap/contact-form-script.js"></script>
<script src="bootstrap/custom.js"></script>
</body>
</html>
```

**View Notes:-**

```
<html lang="en">
<head>
<meta charset="UTF-8">
<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-
scale=1.0">
<title>Student Notes</title>
<link rel="stylesheet" href="css/bootstrap1.css/animate.css">
<link rel="stylesheet" href="css/bootstrap1.css/baguetteBox.min.css">
<link rel="stylesheet" href="css/bootstrap1.css/bootstrap.css.css">
<link rel="stylesheet" href="css/bootstrap1.css/classic.css">
<link rel="stylesheet" href="css/bootstrap1.css/classic.date.css">
<link rel="stylesheet" href="css/bootstrap1.css/classic.time.css">
<link rel="stylesheet" href="css/bootstrap1.css/custom.css">
<link rel="stylesheet" href="css/bootstrap1.css/font-
awesome.min.css">
<link rel="stylesheet" href="css/bootstrap1.css/mycss.css">
<link rel="stylesheet" href="css/bootstrap1.css/responsive.css">
<link rel="stylesheet" href="css/bootstrap1.css/style.css">
<link rel="stylesheet" href="css/bootstrap1.css/superslides.css">
<link rel="stylesheet" href="css\home.css">
<link rel="stylesheet" href="css\notes.css">
<style>
```

```

Body
{
  background-image: url("img/va8.jpg");
  background-size: cover;
  border: 5px;
  font-family: Ink Free;
}

.R1{
color:black;
font-size:40px;
font-family:serif;
}

lable{
width: 100px;
display: inline-block;
}

</style>
</head>
<body>
<nav class="logo navbar-expand-md navbar-dark">
<nav>
<button class="navbar-toggler" type="button" data-
toggle="collapse" data-target="#collapsibleNavbar">
<span class="navbar-toggler-icon" ></span>
</button>
<div class="collapse navbar-collapse" id="collapsibleNavbar">
<ul class="navbar-nav">
<li class="logo">
<li><a class="nav " href="student_home.html">Home</a></li>

```

```

<li><a                                class="nav"                                active"
href="student_notes.php">Notes</a></li>

<li><a                                class="nav"                                href="student_feedback.php">Feed
Back</a></li>

<li><a                                class="nav"                                href="student_profile.php">My
Profile</a></li>

    <li><a class="nav" href="logout.php">Logout</a></li>
</ul>
</nav>

<div class="container">
<center><h3>My Notes</h3>

<table align=center border=border cellpadding="30"
cellspacing="50">

<tr>
<th>Sl.no</th>
<th>Subject Name</th>
<th>Department</th>
<th>Files</th>
<th>Discription</th>
<th>VIEW</th>
</tr>

<?php
require_once('config.php');
session_start();
$id=$_SESSION['user'];
$i=1;

$query1 = "SELECT * FROM studentreg where email='$id' ";
$result1 = mysqli_query($link,$query1);
$row1 = mysqli_fetch_array($result1);
$dept=$row1['degree'];
$query = "SELECT * FROM add_notes where department='$dept'";

```



```

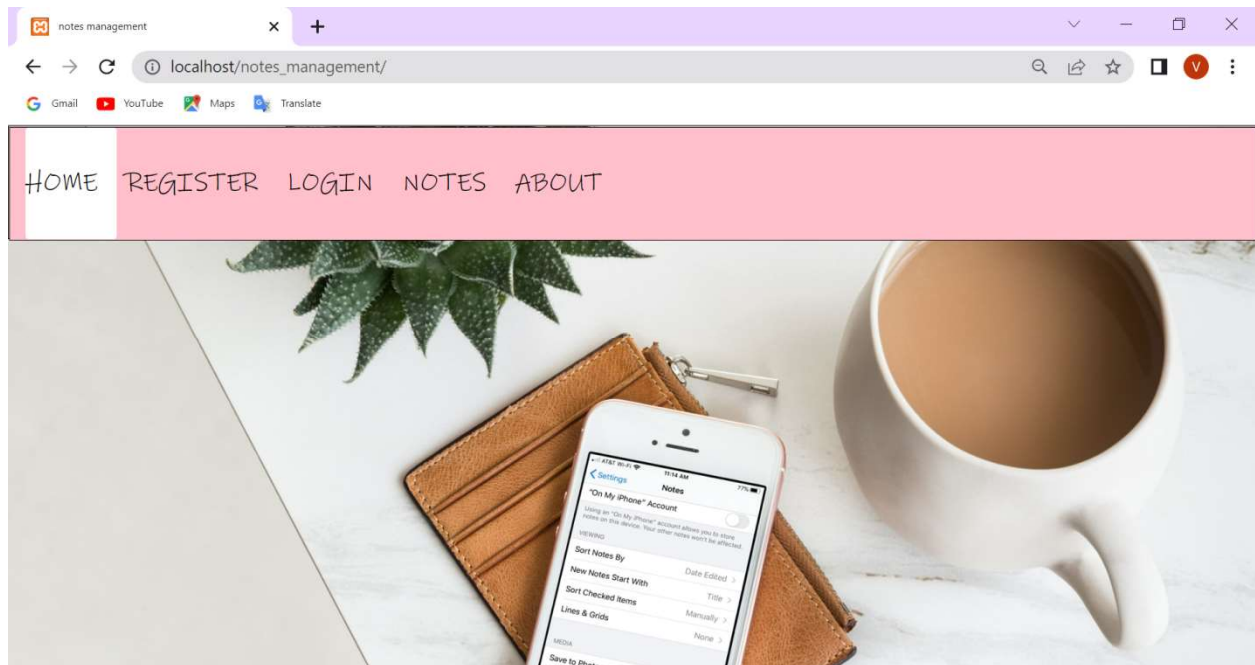
$result = mysqli_query($link,$query);
while($row = mysqli_fetch_array($result))
{
?>
<tr>
<td><?php echo $i; ?>.</td>
<td><?php echo $row['sname']; ?></td>
<td><?php echo $row['department']; ?></td>
<td><?php echo $row['pdf']; ?></td>
<td><?php echo $row['disc']; ?></td>
<td><a href="staff_viewpdf.php?id=?php echo $row['id'];
?>">View</a></td>
</tr>
<?php
$i++;
}
?>
</table>
</center>
<!-- ALL JS FILES -->
<script src="bootstrap/jquery-3.2.1.min.js"></script>
<script src="bootstrap/popper.min.js"></script>
<script src="bootstrap/bootstrap.min.js"></script>
<!-- ALL PLUGINS -->
<script src="bootstrap/jquery.superslides.min.js"></script>
<script src="bootstrap/images-loded.min.js"></script>
<script src="bootstrap/isotope.min.js"></script>
<script src="bootstrap/baguetteBox.min.js"></script>
<script src="bootstrap/form-validator.min.js"></script>
<script src="bootstrap/contact-form-script.js"></script>

```

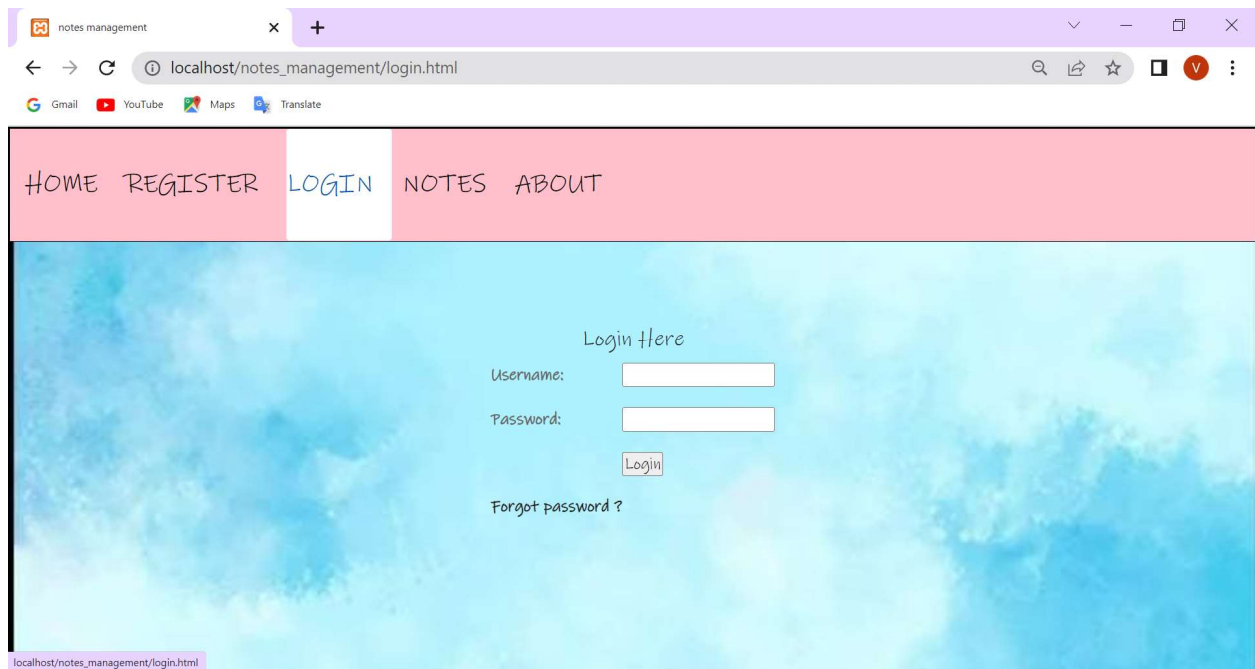
```
<script src="bootstrap/custom.js"></script>  
</body>  
</html>
```

## 5.3 SCREEN SHOTS:-

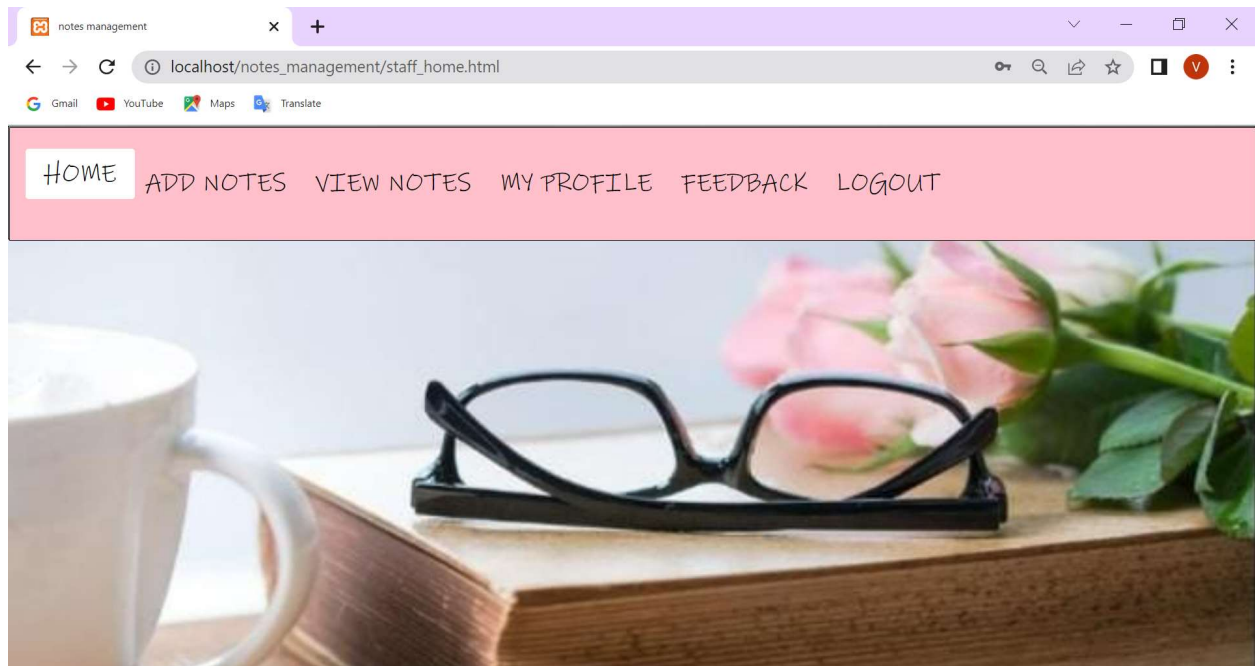
### Homepage:-



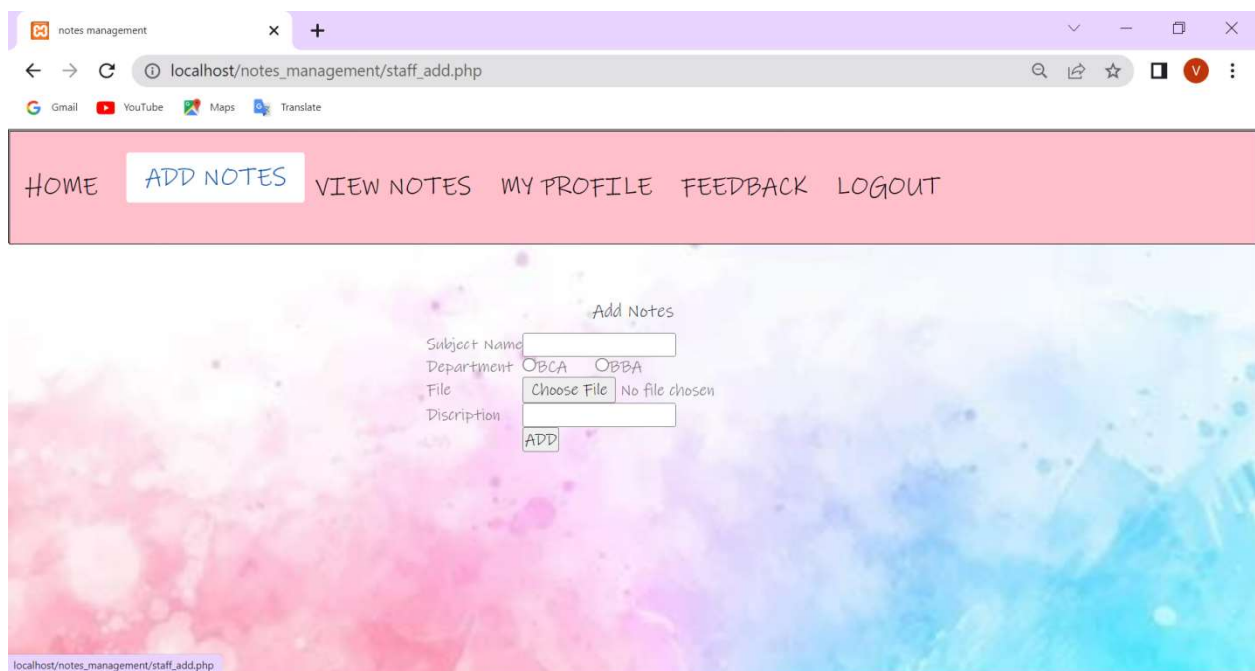
### Login:-



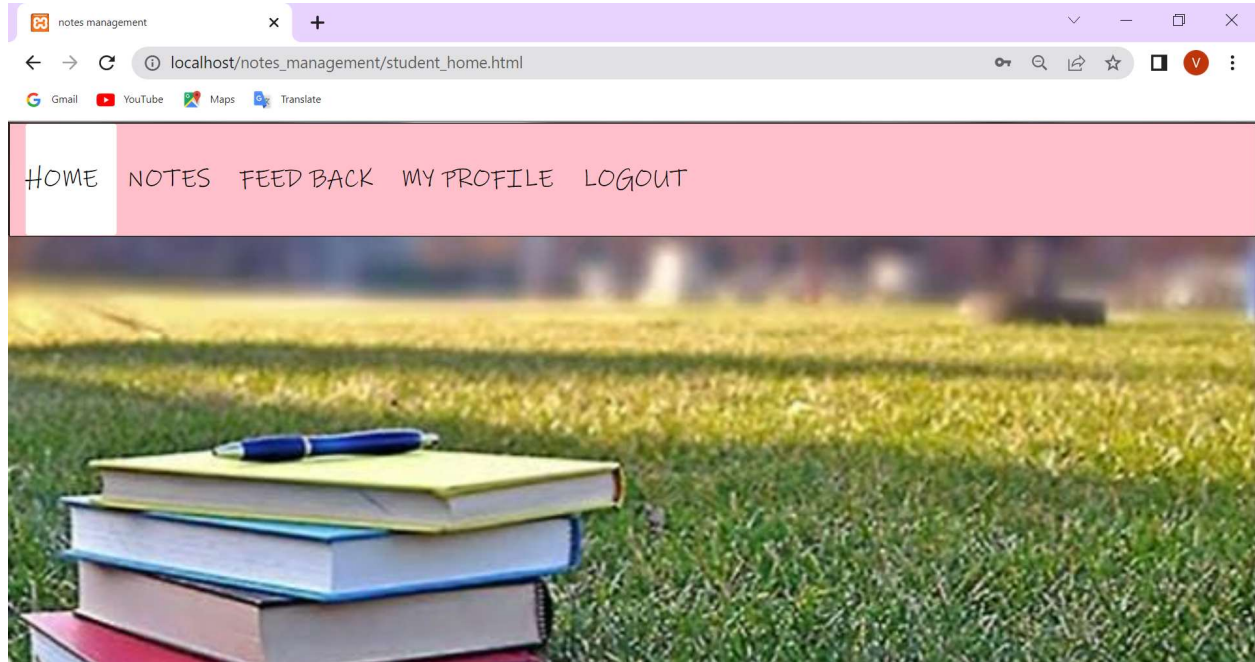
## Staff home



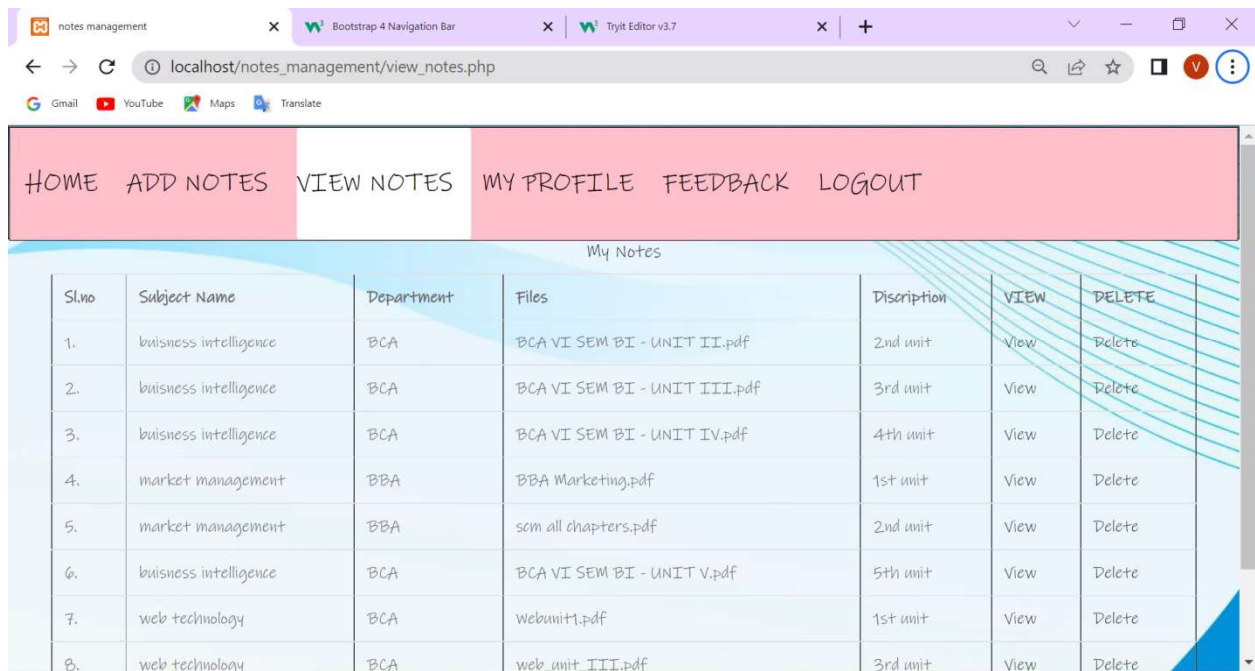
## Add notes



## Student home



## View notes



## Feedback

The screenshot shows a web browser window with the address bar displaying `localhost/notes_management/Student_feedback.php`. The navigation bar at the top contains links: HOME, NOTES, FEED BACK (active), MY PROFILE, and LOGOUT. The main content area has a light blue background with a faint image of a person. It features a form titled "FEED BACK" with the following fields:

- Department:
- Subject:
- Topic:
- Discription:
- 

Below the form, there is another "FEED BACK" title and a table with the following data:

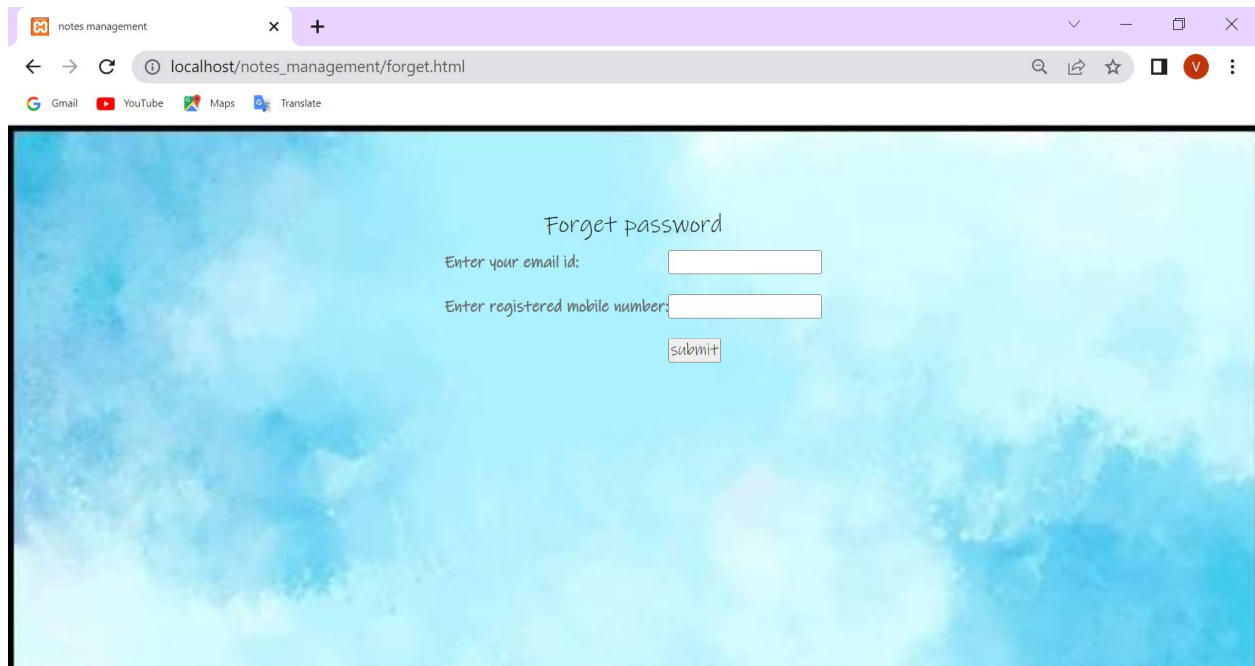
S.No	Department	Subject	Topic	Discription	Delete
1	bca	web technology	Building Forms from d	not understood	<input type="button" value="delete"/>

## Profile

The screenshot shows a web browser window with the address bar displaying `localhost/notes_management/profile.php`. The navigation bar at the top contains links: HOME, ADD NOTES, VIEW NOTES, MY PROFILE (active), FEEDBACK, and LOGOUT. The main content area has a light orange background with a faint image of a person. It features a form titled "MY PROFILE" with the following fields:

- Name:
- Staff id:
- Address:
- Email id:
- Mobile:
-

### Forgot password



The screenshot shows a web browser window with the title 'notes management'. The address bar displays 'localhost/notes\_management/forget.html'. The page content features a 'Forgot password' form on a light blue background. The form includes two input fields: 'Enter your email id:' and 'Enter registered mobile number:'. Below these fields is a 'submit' button.

notes management

localhost/notes\_management/forget.html

Forgot password

Enter your email id:

Enter registered mobile number:

# CHAPTER 6

## TESTING AND RESULTS

### 6.1 Introduction

Testing is an investigation conducted to provide stakeholders with information about the quality of the product or service under test. Software testing can also provide an objective, independent view of the software to allow the business to appreciate and understand the risks of software implementation. Test techniques include the process of executing a program or application with the intent of finding software bugs (errors or other defects), and verifying that the software product is fit for use.

Software testing involves the execution of a software component or system component to evaluate one or more properties of interest. In general, these properties indicate the extent to which the component or system under test:

- meets the requirements that guided its design and development,
- responds correctly to all kinds of inputs,
- performs its functions within an acceptable time,
- is sufficiently usable,
- can be installed and run in its intended environments, and
- achieves the general result its stake holders desire.

Errors are relevant only after the system is in use. A system that has been put to



use without testing can be disastrous in terms of the application. The importance of software testing and its implications cannot be overemphasized. Software testing is a critical element of Software Quality Assurance and represents the ultimate review of the specifications, design and coding.

## 6.2 White box Testing:

**White-box testing** (also known as clear box testing, glass box testing, transparent box testing, and structural testing) is a method of testing software that tests internal structures or workings of an application, as opposed to its functionality (i.e. black-box testing). In white-box testing an internal perspective of the system, as well as programming skills, are used to design test cases. The tester chooses inputs to exercise paths through the code and determine the appropriate outputs. This is analogous to testing nodes in a circuit, e.g. in-circuit testing (ICT). White-box testing can be applied at the unit, integration and system levels of the software testing process. Although traditional testers tended to think of white-box testing as being done at the unit level, it is used for integration and system testing more frequently today. It can test paths within a unit, paths between units during integration, and between subsystems during a system-level test. Though this method of test design can uncover many errors or problems, it has the potential to miss unimplemented parts of the specification or missing requirements.

### 6.3 Black Box Testing:

**Black Box Testing**, also known as Behavioral Testing, is a software testing method

in which the internal structure/ design/ implementation of the item being tested is not known to the tester. These tests can be functional or non-functional, though usually functional. This method is named so because the software program, in the eyes of the tester, is like a black box; inside which one cannot see. This method attempts to find errors in the following categories:

- Incorrect or missing functions
- Interface errors
- Errors in data structures or external database access
- Behaviour or performance errors
- Initialization and termination errors

### 6.4 TESTING OBJECTIVES:

The main objective of testing is to uncover a host of errors, systematically and with minimum effort and time. Stating formally, we can say, Testing is a process of executing a program with the intent of finding an error.

A successful test is one that uncovers an as yet undiscovered error.

A good test case is one that has a high probability of finding error, if it exists.

But there is one thing that testing cannot do (just to quote a very famous sentence) “Testing cannot show the absence of defects, it can only show that software defects are presents.”

As the test results are gathered and evaluated they begin to give a qualitative indication of the reliability of the software. If severe errors are detected, the overall quality of the software is a natural suspect. If, on the other hand, all the errors, which are encountered, are easily modifiable, then one of the two conclusions can be made:

The tests are inadequate to detect possibly present errors.

The software more or less confirms to the quality and reliable standards.

## 6.5 LEVELS OF TESTING

In order to uncover the errors present in different phases we have the concept of levels of testing. The basic levels of testing are

Client Needs	Acceptance Testing
Requirements	System Testing
Design	Integration Testing
Code	Unit Testing

## 6.6 Unit Testing:

**Unit Testing** is a level of software testing where individual units/ components of a software are tested. The purpose is to validate that each unit of the software performs as designed. A unit is the smallest testable part of software. It usually has one or a few inputs and usually a single output. In procedural programming a unit may be an individual program, function, procedure, etc. In object-oriented

programming, the smallest unit is a method, which may belong to a base/ super class, abstract class or derived/ child class. (Some treat a module of an application as a unit. This is to be discouraged as there will probably be many individual units within that module.) Unit testing frameworks, drivers, stubs, and mock/ fake objects are used to assist in unit testing.

Unit testing focuses verification effort on the smallest unit of software i.e. the module. Using the detailed design and the process specifications testing is done to uncover errors within the boundary of the module. All modules must be successful in the unit test before the start of the integration testing begins.

In this project “Communication Media” each service can be thought of a module. There are so many modules like fault registration, mailing, chatting. Each module has been tested by giving different sets of inputs (giving wrong ID and password etc)when developing the module as well as finishing the development so that each module works without any error. The inputs are validated when accepting from the user.

## **6.7 Integration Testing:**

**Integration Testing** is a level of software testing where individual units are combined and tested as a group. The purpose of this level of testing is to expose faults in the interaction between integrated units. Test drivers and test stubs are used to assist in Integration Testing. Testing performed to expose defects in the interfaces and in the interactions between integrated components or systems. In the testing the programs are constructed and tested in small segments.

After the unit testing we have to perform integration testing. The goal here is to see if modules can be integrated properly, the emphasis being on testing interfaces between modules. This testing activity can be considered as testing the design and hence the emphasis on testing module interactions

In this project 'Communication Media', the main system is formed by integrating all the modules. When integrating all the modules I have checked whether the integration effects working of any of the services by giving different combinations of inputs with which the two services run perfectly before Integration.

## **6.8 SYSTEM TESTING**

Here the entire software system is tested. The reference document for this process is the requirements document, and the goal is to see if software meets its requirements.

Here entire 'Communication Media' has been tested against requirements of project and it is checked whether all requirements of project have been satisfied or not.

## **6.9 ACCEPTANCE TESTING**

Acceptance Test is performed with realistic data of the client to demonstrate that the software is working satisfactorily. Testing here is focused on external behavior

of the system; the internal logic of program is not emphasized. In this project ‘XXXXXXXXXXXXX’ I have collected some data and tested whether project is working correctly or not.

Test cases should be selected so that the largest number of attributes of an equivalence class is exercised at once. The testing phase is an important part of software development. It is the process of finding errors and missing operations and also a complete verification to determine whether the objectives are met and the user requirements are satisfied.

## **6.10 TEST PLAN**

This document describes the plan for testing the course scheduling software. All major testing activities are specified here.

Test Units:

In this project we will perform two levels of testing Unit Testing and System Testing the basic units to be tested are:

Fault Register module

Login module

Admin module

User module

Features to be tested:

All the functional features specified in the requirements document will be tested.  
No testing is done for the performance requirements like response time.

**Approach for Testing:**

For unit testing structural testing based on the branch coverage criteria will be used. The goal is to achieve branch coverage of more than 95%. System testing will be largely functional in nature.

**Test Deliverables:**

The following documents are required.

Unit Test Report for each Unit

Test case specification for system testing

Test report for system testing

The test case specification for system testing has to be submitted for review before system testing commences

**6.11 Schedule and Personnel Allocation:**

The entire testing is performed for a period of 10 days. Test case specifications for the system testing are produced while unit testing is going on. The number of persons allocated for my project is one

## 6.12 TEST CASE REPORT

Here we specify all the test cases that are used for system testing. The different conditions that need to be tested along with the test cases used for testing those conditions and the expected outputs are given. The goal is to test the different functional requirements, as specified in the requirements document. Test cases have been selected for both valid and invalid inputs.

## 6.13 SAMPLE TEST CASE REPORT

S.No	Test case	Condition being checked	Expected output	Actual Output
1	Valid User	validity for Login	Successful Login	Successfully Logged in
2	Invalid User	Validity for Login	Login should fail	Failed to login



## 7. Conclusion:

It has been a matter of immense delight, respect and challenge to have this opportunity to take up this venture and complete it effectively. It was a pleasant experience working with the professors. This will be helpful when we are going work in industry & educational field where we can put all these it in our practice. While creating this framework Notes management system, I have learnt a part about the working of system. Amid the development process, I have got it the concept of planning and building a system. Whereas working on my framework I have utilized all the information which was instructed us and all that produces this project complete.

## 8. Future Enhancement:

The project has a exceptionally tremendous scope in future. The project can be implemented on intranet in future. Project can be upgraded in near future as and when necessity for the same arises, because it is exceptionally adaptable in terms of development. With the proposed software of notes management system will be ready to use by any association hence run the complete work in a much superior,

precise and mistake freeway. The following are long term scope for the project.

- 1- Only a specific association individual can use this system.
- 2- Special ID of students and instructors will be generated by the system.
- 3- Mailing framework will be done with the message alert system.

## Chapter 9

# REFERENCES AND BIBLIOGRAPHY

### REFERENCES AND BIBLIOGRAPHY

- [www.w3schools.com](http://www.w3schools.com)
- [www.tutorialpoints.com](http://www.tutorialpoints.com)
- [www.greenforgeeks.org](http://www.greenforgeeks.org)
- [www.stackoverflow.com](http://www.stackoverflow.com)