

❑ INTRODUCTION

INTRODUCTION

Alpha HR (Human Resource Management System), as the name suggests, is a system that accounts for the human resource/the human capital/asset. This is very nice to count Employees to be an asset to the enterprise. However like all other assets of the Firm, human resource also needs renewal and maintenance.. The proposed system helps us manage our Human capital and lets us embed the HR related data with other modules

Alpha HR as a module is needed in almost all other modules in the enterprise. For example, if we are entering any data related to procurement, we might want to record the person details who wanted it. Similarly for a Project request, we might want to record who the Project manager for the project is. For an Order to be approved, we might need the manager's name of the person who punched the particular order. So HRMS is everywhere progressively when we keep reading about the various functionalities, we will be able to relate the concepts to the real world.

People are employed by companies and organizations throughout the world.

Employees are expected to work and deliver results and in turn help the company move forward. Well being of the employees is a important factor in the performance of employees. Excellent infrastructure, good team work, productive environment, appropriate pay package are some of the factors which lead to employee satisfaction and also increases the productivity of employees.

1.1 EXISTING SYSTEM

In the existing system, employees discuss their problems with immediate superiors over emails or verbally. The process is quite unorganized and is not guaranteed to provide proper results. At times, the employees are just ignored by the superiors and employees continue to remain unproductive.

Few employees do take action and approach the HR department for resolution of their problems.

Overall, the whole process is quite informal and far from acceptable.

1.2 PROBLEM DEFINITION

Employees at times face problem in working with their managers. These problems may be due to the nature of work or can be interpersonal in nature. In all cases, the problems need to be addressed to maintain the spirit of the company.

The problem can be resolved by proper communication between the concerned parties.

1.3 PROPOSED SYSTEM

The solution to the above problems lies in creating an intranet or internet based web application where all employees can raise relevant problems and can expect a solution in a given time frame. The employees can raise the complaint, which is then seen by the manager. The manager needs to provide a satisfactory explanation and provide solutions to the same. Project managers can also login to the system.

The online solution is cost effective and easy to maintain. Hence it is proposed to develop a online system which can take care of the process of complaint resolution in a fair manner.

The proposed solution will be a Web based Application developed in PHP with MySQL as a database. We plan to make use of the WAMP Server as the tool for development.

The candidate system will be a 3 tier based architecture with the following

Front End: The front end will be done in HTML / CSS

Back End: The back end coding will be done in PHP

Database: The data will be stored in MySQL database

1.4 OBJECTIVE

The objective of this project is to develop an official website for an HR processing which does not have domain server name. The system provides necessary online solutions to the employees.

REQUIREMENTS

2. REQUIREMENTS

2.1 HARDWARE REQUIREMENTS:

The hardware requirements for this application are listed below

CPU	Intel Pentium Dual Core / Core 2 Duo / Core i3
Memory (RAM)	1 GB or above
Storage (Hard Disk)	80 GB or above

2.2 SOFTWARE REQUIREMENTS:

The Software requirements for this application are listed below

Operating System	Windows 7 or above
Web design	HTML, CSS
Server Side Scripting	PHP
Client Side Scripting	JavaScript
Database	MySQL

DESIGN

DESIGN

3.1 DATA FLOW DIAGRAM:

A data flow diagram (DFD) is a significant modeling technique for analyzing and constructing information processes. Data flow diagram (DFD) is a graphical representation of the flow of data through an information system. DFD's 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.

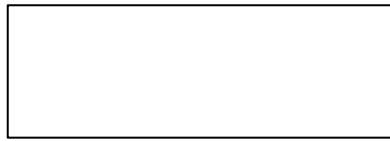
This shows the flow of control through an algorithm, allowing a reader to determine what operations will be performed, in what order, and under what circumstances, but not what kinds of data will be input to and output from the system, nor where the data will come from and go to, nor where the data will 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 modeled. Numerous levels may be required to explain a complicated system.

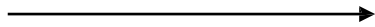
Data flow diagram represents the following objects of the real world through the symbols

- Processes : These are the tasks that occur in the real world. These are represented using rounded rectangles in the DFD
- Data store : The repository of data, which is used to store records, these are represented by an open rectangle.
- External Entity : External Entities are the outside the system, but they either supply input to the system or use the system output. These are represented by a rectangle

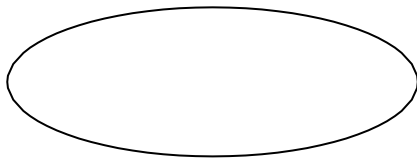
The DFD represents the flow of data in the system. That is within the various modules among the project. The data flows of largest system are divided into many levels to represent the functionality of the module clearly. The convention used in data flow diagram of the system is shown



This represents data sources



This represents data flow

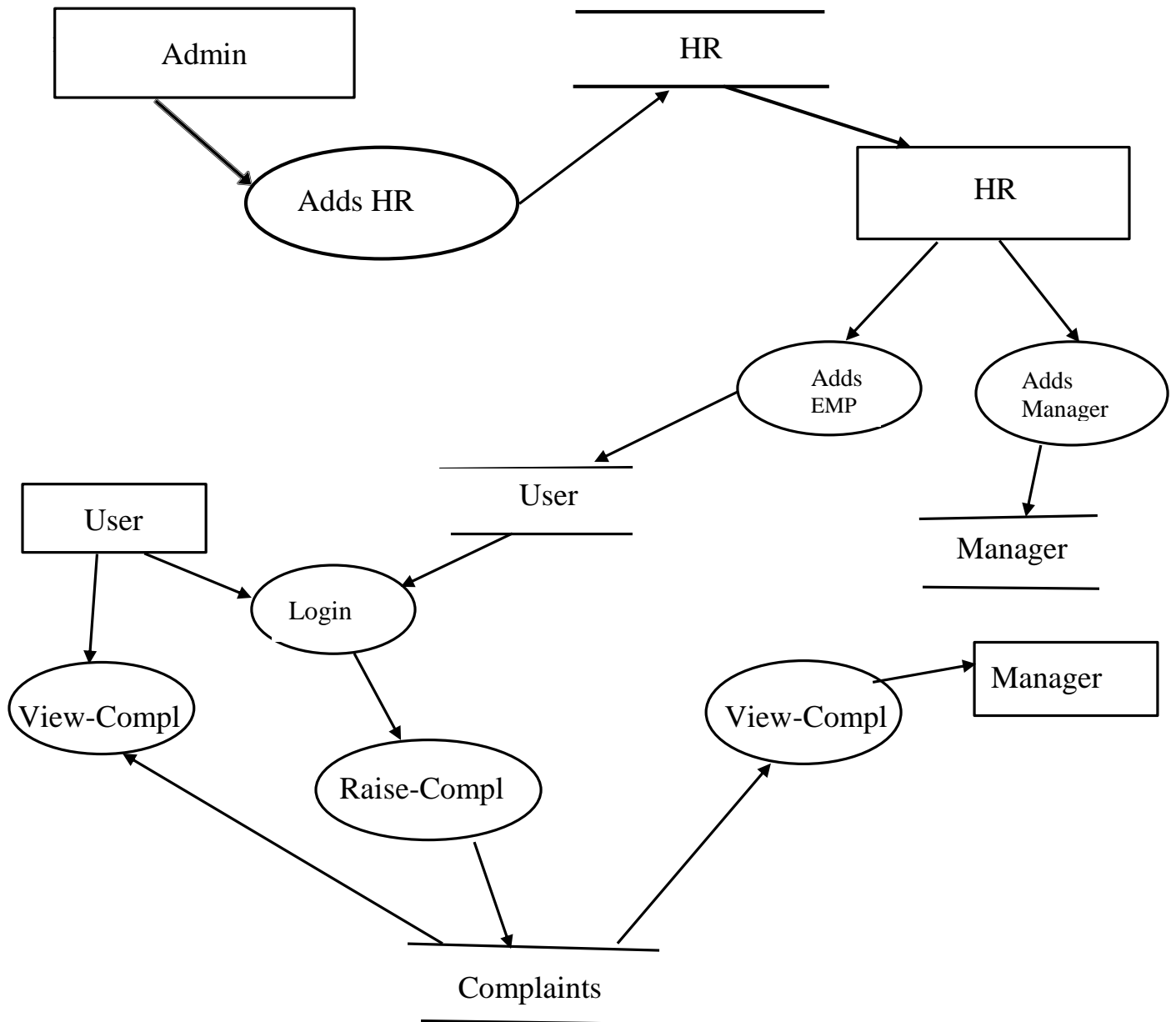


This represents the process



This represents the database

PROPOSED SYSTEM DFD



3.2 Entity Relationship Diagrams

An ER model is an abstract way to describe a database. Describing a database usually starts with a relational database, which stores data in tables.

An Entity relationship diagram is a data modeling technique that creates a graphical representation of the entities and the relationship between entities, within an information system.

The three main components of ER 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 an interaction between the entities. In the example above, the customers place 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 a line connecting the entities.

- The cardinality defines the relationship between the entities in the 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.

The three main cardinality relationships are :

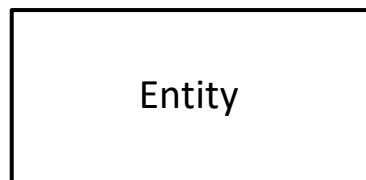
- ❖ One to one expressed as 1:1
- ❖ One to many expressed as 1:M
- ❖ Many to many expressed as M:N

Entity Relationship Notations

Entity- relationship model (ER model) is an abstract way to describe a database. Peter Chen developed ERD in 1976. Since then Charles Bachman and James Martin have added some slight refinements to the basic ERD principles.

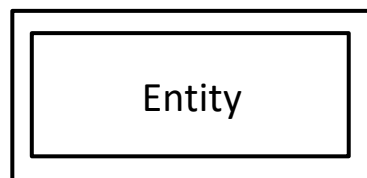
Entity:

An entity is a real-world object or concept which is distinguishable from other objects. It may be something tangible, such as a particular student or building. It may also be somewhat more conceptual, such as CS A-341, or an email address.



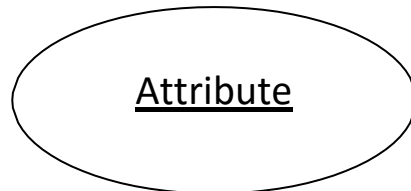
Weak Entity:

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.

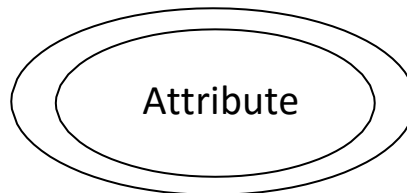


Key Attributes:

A key attribute is the unique, distinguishing characteristic of the entity. For example, an employee's social security number might be the employee's key attribute.

**Multi - value Attribute:**

A multi-value attribute can have more than one value. For example, an employee entity can have multiple skill values.

**Derived Attribute:**

A derived attribute is based on another attribute. For example, an employee's monthly salary is based on the employee's annual salary.

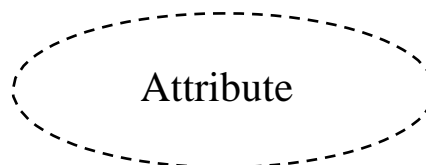
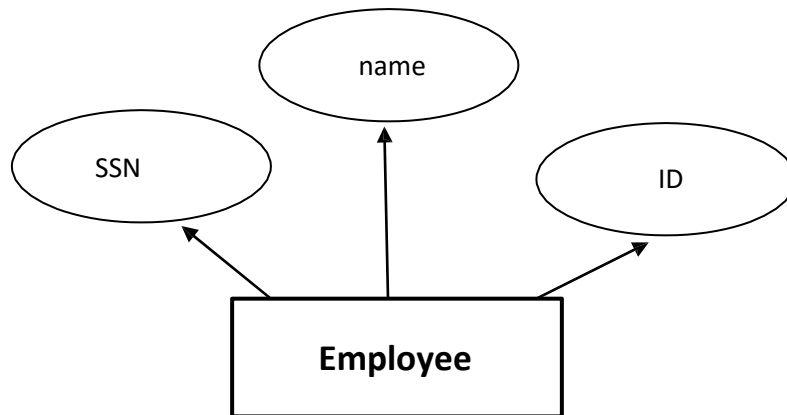


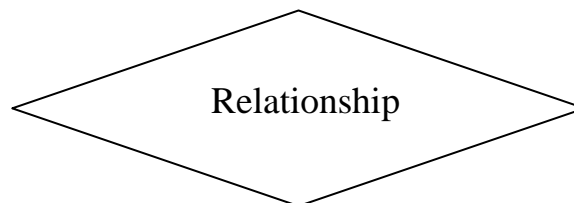
Diagram conventions:

- ☐ An entity set is drawn as a rectangle.
- ☐ Attributes are drawn as ovals.
- ☐ Attributes which belong to the primary key are underlined.



Relationships

Relationships illustrates how two entities share information in the database structure. The relationship must be uniquely identified by the participating entities. A relationship can also have descriptive attributes, to record additional information about the relationship.

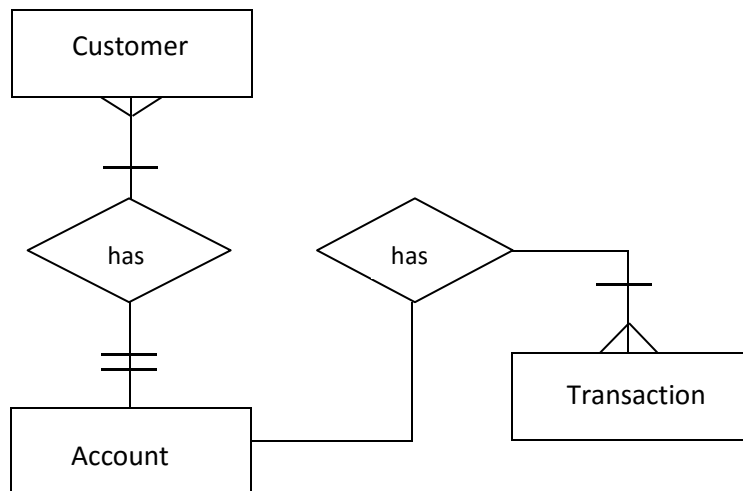


Cardinality

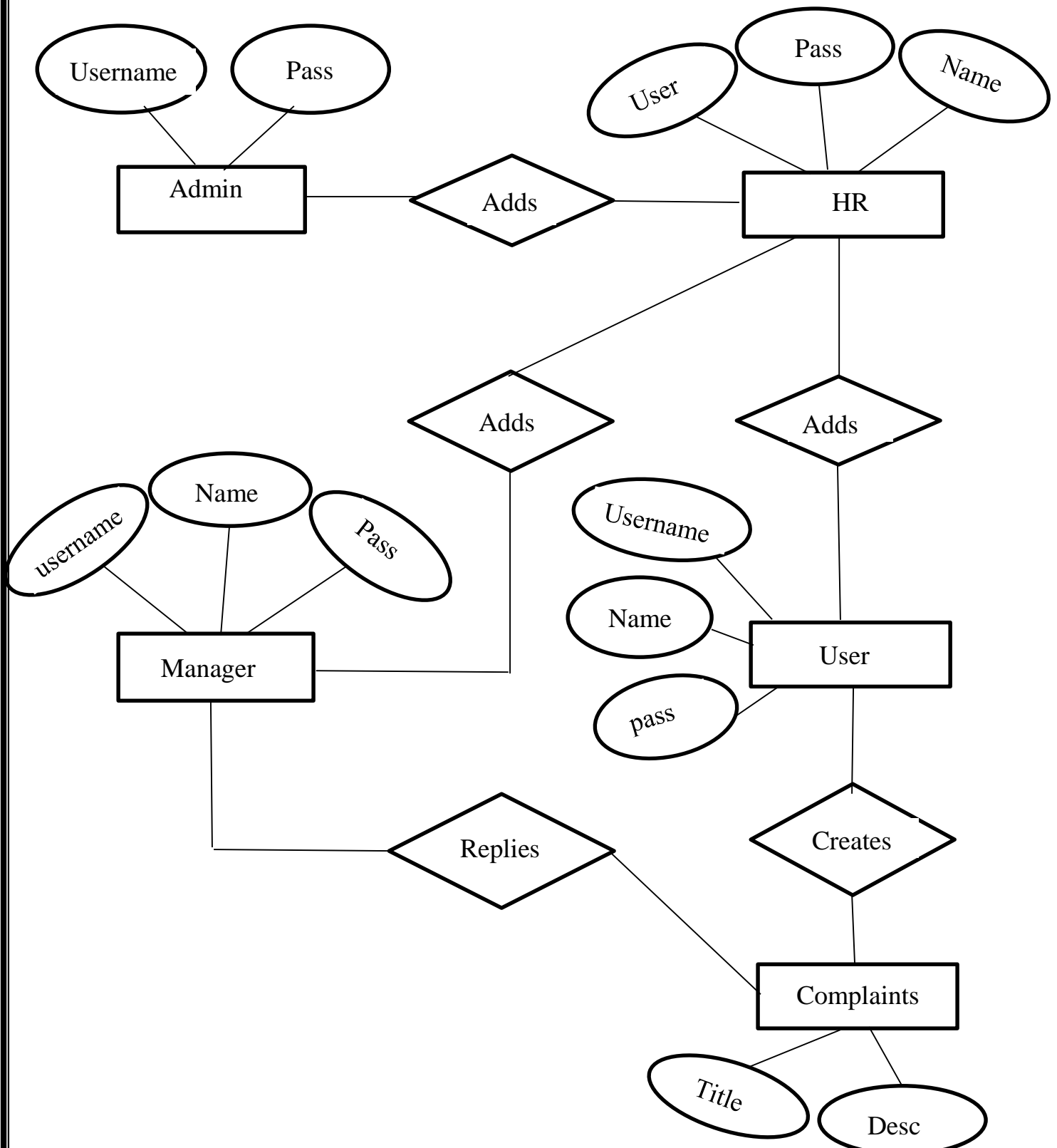
Cardinality specifies how many instances of an entity relate to one instance of another entity.

Ordinarily is also closely linked to cardinality. While cardinality specifies the occurrences of a relationship, ordinarily describes the relationship as either mandatory or optional.

In other words, cardinality specifies the maximum number of relationships and ordinarily specifies the absolute minimum number of relationships.



ER Diagram



3.3 DATA BASE DESIGN

INTRODUCTION

Database design is the process of developing database structures to hold data to cater to user requirements. The final design must satisfy user needs in terms of completeness, integrity, performance and other factors.

The general theme behind the database is handled as an integrated whole, with a minimum of redundancy and improved performance. DBMS is used to manipulate, describe and manage data. A database is a collection of interrelated data store with minimum redundancy to serve many users quickly and efficiently.

The general objective is to make information access easy, quick, inexpensive and flexible for the user. The primary input to the database design process is the organization's statement of requirements. Poor definition of these requirements is a major cause of poor database design, resulting in database of limited scope and utilities which are usable to adapt to changes.

The major step in database design is to identify the entities and relationship that reflects the organization's data naturally. The objective of this step is to specify conceptual structure of the data and is often referred to as data modeling.

The heart of Data Base is DBMS. It manages and controls the database and handles request from the application program in a data manipulation language. To produce the users view, the DBMS take the help of DDL, which describes and identifies data structure

Data structure is refined through a process called normalization. Data are grouped in the simplest way possible; such that alter changes can be made with minimum impact in the data structure. Normalization is a process of simplifying the relationship between data elements in a record. Through normalization, a collection of data in a record structure is replaced by successive record structure they are simplify an more predictable and therefore more manageable.

Tables:

Table for Admin

FIELD	TYPE	NULL	KEY	DEFAULT
Username	VARCHAR(10)	NO	PRIMARY	NULL
Password	VARCHAR(10)	NO		NULL
Name	VARCHAR(30)	NO		NULL
Designation	VARCHAR(20)	NO		NULL
Mobile	VARCHAR(10)	NO		NULL
Email	VARCHAR(100)	NO		NULL

Table for Complaints

FIELD	TYPE	NULL	KEY	DEFAULT
Id	INT(11)	NO	PRIMARY	NULL
Subject	VARCHAR(100)	NO		NULL
Description	VARCHAR(1000)	NO		NULL
Severity	INT(11)	NO		NULL
Username	VARCHAR(20)	NO		NULL
Date	DATE	NO		NULL
Employee type	INT(11)	NO		NULL
Status	INT(11)	NO		NULL
Manager	VARCHAR(20)	NO		NULL
Reply	VARCHAR(1000)	NO		NULL
Hrreply	VARCHAR(1000)	NO		NULL
Ceoreply	VARCHAR(1000)	NO		NULL

Table for HR

FIELD	TYPE	NULL	KEY	DEFAULT
Username	VARCHAR(10)	NO	PRIMARY	NULL
Password	VARCHAR(10)	NO		NULL
Name	VARCHAR(30)	NO		NULL
Designation	VARCHAR(20)	NO		NULL
Mobile	VARCHAR(10)	NO		NULL
Email	VARCHAR(100)	NO		NULL

Table for Employee

FIELD	TYPE	NULL	KEY	DEFAULT
Username	VARCHAR(10)	NO	PRIMARY	NULL
Password	VARCHAR(10)	NO		NULL
Name	VARCHAR(30)	NO		NULL
Designation	VARCHAR(20)	NO		NULL
Mobile	VARCHAR(10)	NO		NULL
Email	VARCHAR(100)	NO		NULL
Manager	VARCHAR(10)	NO		NULL

Table for Chatters

FIELD	TYPE	NULL	KEY	DEFAULT
Name	VARCHAR(10)	NO		NULL
Seen	VARCHAR(20)	NO		NULL




Table for Messages

FIELD	TYPE	NULL	KEY	DEFAULT
Name	VARCHAR(10)	NO		NULL
msg	TEXT	NO	PRIMARY	NULL
posted	VARCHAR(20)	NO		NULL

3.4 FORM DESIGN

FORM TYPES

There are four types classified by what it does in the system. They are:

-  Action forms: To perform some extraction such as storing, modifying, and deleting data
-  Searching memory forms: To perform extraction and display operations on existing historical data
-  Report forms: To generate decision support data from existing records.

We used as output forms, as an input media we used both action and memory forms in combination.

FORM LAYOUT

When form is designed a list prepared of all the items to be included on the form and the maximum space to be reserved. The form user to make sure it has the required details should check the list.

- ☐ Title
- ☐ Data zoning
- ☐ Rules and captions

DESIGN CONSIDERATIONS:

In designing these forms we taken care several attributes that are mentioned below:

- ☐ Identification and wording – Form titles and labels.
- ☐ Maximum readability and use – Legible intelligible uncomplicated and space.
- ☐ Physical factors – Composition, color, layout.
- ☐ Order of data items – Logical sequence, data relation.
- ☐ Ease of data entry – Field positions
- ☐ Size and arrangement - Size, storing, filing and space for signs.
- ☐ Use of instructions – Online help for data entry, status information



WELCOME ADMINISTRATOR

Official Home Page For Administrator

LOGOUT



REGISTRATION FORM

CEO REGISTRATION

HR REGISTRATION



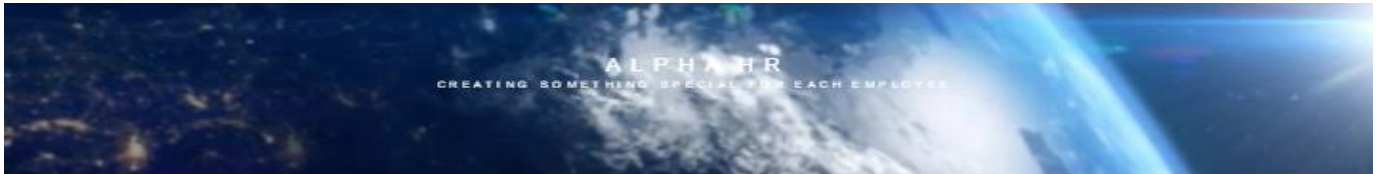
MY PROFILE

EDIT PROFILE

EDIT PASSWORD

© 2018 HRMS. All Rights Reserved. Developed by Jain College.

Designed by [TemplateMonster](#)



ADMINISTRATOR PASSWORD PAGE

Mutify your login password

New Password	<input type="password"/>
Confirm Password	<input type="password"/>

SUBMIT

© 2018 HRMS. All Rights Reserved. Developed by Jain College.

Designed by [TemplateMonster](#)



CEO REGISTRATION FORM

UserName	<input type="text"/>
Password	<input type="password"/>
Confirm Password	<input type="password"/>
Name	<input type="text"/>
Designation	<input type="text"/>
Mobile	<input type="text"/>
Email	<input type="text"/>

REGISTER

BACK



CHAT FORUM

USERS

anack

anack : yeh

anack : hello

sanjay : hello

sanjay : what u ding?

anack : im working on project

sanjay : hi dharmayakona

anack : nand yeh hi kuniya

LOG OUT



CODING

Introduction to programming language

A programming language is an artificial language designed to communicate instructions to a machine, particularly a computer. Programming languages can be used to create programs that control the behavior of a machine and/or to express algorithms precisely.

The earliest programming languages predate the invention of the computer, and were used to direct the behavior of machines such as Jacquard looms and player pianos. Thousands of different programming languages have been created, mainly in the computer field, with many being created every year. Most programming languages describe computation in an imperative style, i.e., as a sequence of commands, although some languages, such as those that support functional programming or logic programming, use alternative forms of description.

The description of a programming language is usually split into the two components of syntax (form) and semantics (meaning). Some languages are defined by a specification document (for example, the C programming language is specified by an ISO Standard), while other languages, such as Perl 5 and earlier, have a dominant implementation that is used as a reference.

SYNTAX:

A programming language's surface form is known as its syntax. Most programming languages are purely textual; they use sequences of text including words, numbers, and punctuation, much like written natural languages. On the

other hand, there are some programming languages which are more graphical in nature, using visual relationships between symbols to specify a program.

The syntax of a language describes the possible combinations of symbols that form a syntactically correct program. The meaning given to a combination of symbols is handled by semantics (either formal or hard-coded in a reference implementation). Since most languages are textual, this article discusses textual syntax.

SEMANTICS:

The term Semantics refers to the meaning of languages, as opposed to their form (syntax).

STATIC SEMANTIC:

The static semantics defines restrictions on the structure of valid texts that are hard or impossible to express in standard syntactic formalisms. For compiled languages, static semantics essentially include those semantic rules that can be checked at compile time. Examples include checking that every identifier is declared before it is used (in languages that require such declarations) or that the labels on the arms of a case statement are distinct. Many important restrictions of this type, like checking that identifiers are used in the appropriate context (e.g. not adding an integer to a function name), or that subroutine calls have the appropriate number and type of arguments, can be enforced by defining them as rules in a logic called a type system. Other forms of static analyses like data flow analysis may also be part of static semantics. Newer programming languages like Java and C# have definite assignment analysis, a form of data flow analysis, as part of their static semantics.

Dynamic Semantic:

Once data has been specified, the machine must be instructed to perform operations on the data. For example, the semantics may define the strategy by which expressions are evaluated to values, or the manner in which control structures conditionally execute statements. The dynamic semantics (also known as execution semantics) of a language defines how and when the various constructs of a language should produce a program behavior. There are many ways of defining execution semantics. Natural language is often used to specify the execution semantics of languages commonly used in practice. A significant amount of academic research went into formal semantics of programming languages, which allow execution semantics to be specified in a formal manner. Results from this field of research have seen limited application to programming language design and implementation outside academia.

The technology selected for implementing Management System is PHP/MYSQL. Apache is used as the HTTP server. The development was done in a 'windows' environment.

PHP

PHP is a general-purpose scripting language that is especially suited to server-side web development where PHP generally runs on a web server. PHP code is embedded into the HTML source document. Any PHP code in a requested file is executed by the PHP runtime, usually to create dynamic web page content. It can also be used for client-side GUI applications. PHP can be deployed on many web servers and operating systems, and can be used with many relational database management systems (RDBMS).

PHP (recursive acronym for PHP: Hypertext Preprocessor) is a widely-used open source general-purpose scripting language that is especially suited for web development and can be embedded into HTML.

An example:

```
<!DOCTYPE HTML>
<html>
<head>
<title>Example</title>
</head>
<body>
<? Php
echo "Hi, I'm a PHP script!";
?>
</body>
</html>
```

Output: Hi, I'm a PHP script!

FEATURES OF PHP

SERVER-SIDE SCRIPTING: This is the most traditional and main target field for PHP. You need three things to make this work. The PHP parser (CGI or server module), a web server and a web browser. You need to run the web server, with a connected PHP installation. You can access the PHP program output with a web browser, viewing the PHP page through the server. All these can run on your home machine if you are just experimenting with PHP programming.

COMMAND LINE SCRIPTING: You can make a PHP script to run it without any server or browser. You only need the PHP parser to use it this way. This type of usage is ideal for scripts regularly executed using cron (on *nix or Linux) or Task Scheduler (on Windows). These scripts can also be used for simple text processing tasks.

WRITING DESKTOP APPLICATIONS: PHP is probably not the very best language to create a desktop application with a graphical user interface, but if you know PHP very well, and would like to use some advanced PHP features in your client-side applications you can also use PHP-GTK to write such programs. You also have the ability to write cross-platform applications this way. PHP-GTK is an extension to PHP, not available in the main distribution.

PHP can be used on all major operating systems, including Linux, many UNIX variants (including HP-UX, Solaris and OpenBSD), Microsoft Windows, Mac OS X, RISC OS, and probably others. PHP has also support for most of the web servers today. This includes Apache, IIS, and many others. And this includes any web server that can utilize the FastCGI PHP binary, like lighttpd and nginx. PHP works as either a module, or as a CGI processor.

Why we used PHP?

- ☐ PHP runs on different platforms. (Windows, Linux, UNIX, Mac OS X, etc.)
- ☐ PHP is compatible with almost all servers used today. (Apache, IIS, etc.)
- ☐ PHP has support for a wide range of databases.
- ☐ PHP is free.
- ☐ PHP is easy to learn and runs efficiently on the server side.

MYSQL:

MySQL is a relational database management system (RDBMS) that runs as a server providing multi-user access to a number of databases. MySQL is a popular choice of database for use in web applications and is an open source product. The process of setting up a MySQL database varies from host to host, however we will end up with a database name, a user name and a password.

MySQL is currently the most popular open source database server in existence. On top of that, it is very commonly used in conjunction with PHP scripts to create powerful and dynamic server-side applications.

- ☐ MySQL is a database system used on the web
- ☐ MySQL is a database system that runs on a server
- ☐ MySQL is ideal for both small and large applications
- ☐ MySQL is very fast, reliable, and easy to use
- ☐ MySQL supports standard SQL
- ☐ MySQL compiles on a number of platforms
- ☐ MySQL is free to download and use
- ☐ MySQL is developed, distributed, and supported by Oracle Corporation
- ☐ MySQL is named after co-founder Monty Widenius's daughter: My

HTML 5:

HTML5 is a markup language for structuring and presenting content for the World Wide Web and a core technology of the Internet. It is the fifth revision of the HTML standard (created in 1990 and standardized as HTML 4 as of 1997) and as of December 2012, is a W3C Candidate Recommendation. Its core aims have been to improve the language with support for the latest multimedia while keeping it easily readable by humans and consistently understood by computers and devices (web browsers, parsers, etc.). HTML5 is intended to subsume not only HTML 4, but XHTML 1 and DOM Level 2 HTML as well.

Following its immediate predecessors HTML 4.01 and XHTML 1.1, HTML5 is a response to the observation that the HTML and XHTML in common use on the World Wide Web are a mixture of features introduced by various specifications, along with those introduced by software products such as web browsers, those established by common practice, and the many syntax errors in existing web documents. It is also an attempt to define a single markup language that can be written in either HTML or XHTML syntax. It includes detailed processing models to encourage more interoperable implementations it extends, improves and rationalizes the markup available for documents, and introduces markup and application programming interfaces (APIs) for complex web applications. For the same reasons, HTML5 is also a potential candidate for cross-platform mobile applications. Many features of HTML5 have been built with the consideration of being able to run on low-powered devices such as smart phones and tablets.

<audio> and <canvas> elements, as well as the integration of scalable vector graphics (SVG) content (that replaces the uses of generic <object> tags) and MathML for mathematical formulas. These features are designed to make it easy to

include and handle multimedia and graphical content on the web without having to resort to proprietary plugins and APIs. Other new elements, such as <section>, <article>, <header> and <nav>, are designed to enrich the semantic content of documents. New attributes have been introduced for the same purpose, while some elements and attributes have been removed. Some elements, such as <a>, <cite> and <menu> have been changed, redefined or standardized. The APIs and Document Object Model (DOM) are no longer afterthoughts, but are fundamental parts of the HTML5.

CSS 3:

In particular, HTML5 adds many new syntactic features. These include the new <video>, Cascading Style Sheets (CSS) is a style sheet language used for describing the presentation semantics (the look and formatting) of a document written in a markup language. It's most common application is to style web pages written in HTML and XHTML, but the language can also be applied to any kind of XML document, including plain XML, SVG and XUL.

CSS is designed primarily to enable the separation of document content (written in HTML or a similar markup language) 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 pages to share formatting, and reduce complexity and repetition in the structural content (such as by allowing for table less web design). CSS can also allow the same markup page to be presented in different styles for different rendering methods, such as on-screen, in print, by

voice (when read out by a speech-based browser or screen reader) and on Braille-based, tactile devices. It can also be used to allow the web page to display differently depending on the screen size or device on which it is being viewed. While the author of a document typically links that document to a CSS style sheet, readers can use a different style sheet, perhaps one on their own computer, to override the one the author has specified.

CSS specifies a priority scheme to determine which style rules apply if more than one rule matches against a particular element. In this so-called cascade, priorities or weights are calculated and assigned to rules, so that the results are predictable.

WAMP:

WAMPs are packages of independently created programs installed on computers that use a Microsoft Windows operating system.

WAMP is an acronym formed from the initials of the operating system Microsoft Windows and the principal components of the package: Apache, MySQL and one of PHP, Perl or Python. Apache is a web server. MySQL is an open-source database. PHP, Perl and Python are scripting languages that can manipulate information held in a database and generate web pages dynamically each time content is requested by a browser. Other programs may also be included in a package, such as phpMyAdmin which provides a graphical user interface for the MySQL database manager.

WAMP stands for Windows, Apache, MYSQL and PHP. WAMP is a small and light Apache distribution containing the most common web development technologies in a single package. Its contents, small size, and portability make it the ideal tool for students developing and testing applications in PHP and MySQL.

4.1 INTERFACE CODING

MANAGER LOGIN PAGE

```
<?php
```

```
// this page is used for manager login
```

```
session_start();
```

```
if (isset($_POST['login']))
```

```
{
```

```
    $username = $_POST['username'];
```

```
    $password = $_POST['password'];
```

```
    $con = mysqli_connect("localhost", "root", "", "hrms");
```

```
    if (mysqli_connect_errno())
```

```
    {
```

```
        $error = "Cannot connect to database server. Try again later.";
```

```
        exit();
```

```
    }
```

```
    $query = "select * from manager where username='$username' and  
password='$password'";
```

```
    $result = mysqli_query($con, $query);
```

```
    $count = mysqli_num_rows($result);
```

```
    if ($count == 1)
```

```
    {
```

```
        // username password combination is correct
```

```
        $_SESSION['manager'] = $username;
```

```
        header('Location: manager-welcome.php');
```

```
    }
```

```
    else
```

```
{  
    // provided username password combination does not exists  
    $error = "Username password combination is wrong";  
}  
mysqli_close($con);
```

```
}
```

```
?>
```

```
<!DOCTYPE html>  
<html lang="en">  
<head>  
    <title>Home</title>
```

```
<?php
```

```
    include('_head.php');
```

```
?>
```

```
</head>
```

```
<body>  
<div class="page">
```

```
<?php
```

```
    include('_header.php');
```

```
?>
```



```

<main>
  <section class="well">
    <div class="container center">
      <h1>Manager Login</h1>
      <hr/>
      <p>For login as Manager. Provide your registered username and password</p>

      <form class="booking-form" method="post">

        <div class="tmInput">
          <input type="text" placeholder="Uername" name="username" required="required"
/>
        </div>

        <div class="tmInput">
          <input type="password" placeholder="Password" name="password"
required="required"/>
        </div>

        <div class="tmInput">
          <input type="submit" name="login" value="Login" class="btn"/>
        </div>

        <?php
          if (isset($error))
          {
            echo $error;
            unset($error);
          }

        ?>

      </form>

    </div>

```

```
</section>
</main>
<?php
    include('_footer.php');
?>
</div>
<script src="js/script.js"></script>
<!-- coded by Diversant -->
</body>
</html>
```

MANAGER REGISTRATION FORM

```
<?php
Session_start();
If (!isset($_SESSION['hr']))
{
    header('location: hr-login.php');
}
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Home</title>
<?php
    include('_head.php');
?>
</head>
<body>
<div class="page">
<?php
    include('_header.php');
?>
<main>
<section class="well">
```

```

<div class="container center">
<h1>Manager Registration Form</h1>
<div class="CSSTableGenerator">
<form method="POST" action="manager-insert.php">
<table border="3">
<tr>
<td></td>
<td>Provide Details For Manager</td>
</tr>
</tr>
<tr>
<td>UserName</td>
<td><input type="text" name="user" required="required" maxlength="10"
Onkeypress="return onlyAplhabets(event)"></td>
</tr>
<tr>
<td>Password</td>
<td><input type="password" name="pass" required="required"
maxlength="10"></td>
</tr>
<tr>
<td>Confirm Password</td>
<td><input type="password" name="cpass" required="required"
maxlength="10"></td>
</tr>
<tr>
<td>Name</td>
<td><input type="text" name="name" required="required"
maxlength="30"></td>
</tr>
<tr>
<td>Designation</td>
<td><input type="text" name="designation" required="required"
maxlength="20"></td>
</tr>

```

```

<tr>
<td>Project Name</td>
<td><input type="text" name="projectname" required="required"
    Maxlength="20"></td>
</tr>
<tr>
<td>Mobile</td>
<td><input type="text" name="mobile" required="required" maxlength="10"
    Onkeypress="return onlyNumber(event)"></td>

<td>Email</td>
<td><input type="email" name="email" required="required" maxlength="100">
</td>
</tr>
<tr>
<td><input id="button" type="submit" name="submit" value="Register"
    Class="btn"></td>
</table>
</from>
</div>
<a href="hr-welcome.php" class="btn">Back</a>
</section>
</main>
<?php
    include('_footer.php');
?>
</div>
<script src="js/script.js"></script>

<!--code by Diversant-->
</body>

</html>

```

Employee Complaint Registration Form

```
<?php
Session-start( );
If (!isset($_SESSION['user-login.php']));
{
    Header('location: user-login.php');
}
?>
<!DOCTYPE html>
<html lang="en">
<head>
    <title>Home</title>
<?php
    Include('_head.php');
?>
</head>

<body>
<div class="page">
<?php
    Include('_header.php');
?>

<main>
<section class="well">
<div class="container center">
<h1>Complaint Form</h1>
<div id="Sign-Up">
<form method="POST" action="ucomplaint-insert.php">
<div class="CSSTableGenerator">
<table>
<tr>
<td>Subject</td>
```

```

<td> <input type="text" name="subject" required="required"></td>
</tr>

<tr>
<td>Description</td>
<td><textarea name="description" rows=10 cols=50></textarea></td>
< /tr>

<td>Severity</td><td>
<select name="severity">
<option value="1"> Low</option>
<option value="2">Medium</option>
<option value="3">High</option>
</select>

</td>
</tr>
</table>
</div>

<input id="button" type="submit" value="Submit" class="btn">
</form>
</div>
</section>
<a href="user-welcome.php" class="btn">Back</a>

</main>

<?php
Include('_footer.php');
?>
</div>
<script src="js/script.js"></script>
<!-- coded by Diversant -->
</body>
</html>

```

TESTING

5.1 TESTING INTRODUCTION

Software 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, but are not limited to, the process of executing a program or application with the intent of finding software bugs (errors or other defects).

Software testing can be stated as the process of validating and verifying that a computer program/application/product:

- Meets the requirements that guided its design and development,
- Works as expected,
- Can be implemented with the same characteristics,
- And satisfies the needs of stakeholders.

Software testing, depending on the testing method employed, can be implemented at any time in the development process. Traditionally most of the test effort occurs after the requirements have been defined and the coding process has been completed, but in the agile approaches most of the test effort is on-going. As such, the methodology of the test is governed by the chosen software development methodology.

Different software development models will focus the test effort at different points in the development process. Newer development models, such as Agile, often employ test-driven development and place an increased portion of the testing in the hands of the developer, before it reaches a formal team of testers. In a more traditional model, most of the test execution occurs after the requirements have been defined and the coding process has been completed.

System should not be tested as a single, monolithic unit. The testing process should therefore proceed in stages where testing is carried out incrementally in conjunction with system implementation. Errors in program components may come to light at a later stage of testing process. The process is therefore an iterative one with information being fed back from later stages to earlier parts of the process.

5.2 TESTING METHODS

There are many approaches to software testing. Reviews, walkthroughs, or inspections are referred to as static testing, whereas actually executing programmed code with a given set of test cases is referred to as dynamic testing. Static testing can be omitted, and unfortunately in practice often is. Dynamic testing takes place when the program itself is used. Dynamic testing may begin before the program is 100% complete in order to test particular sections of code and are applied to discrete functions or modules. Typical techniques for this are either using stubs/drivers or execution from a debugger environment.

Static testing involves verification whereas dynamic testing involves validation. Together they help improve software quality.

THE BOX APPROACH

Software testing methods are traditionally divided into white- and black-box testing. These two approaches are used to describe the point of view that a test engineer takes when designing test cases.

WHITE-BOX TESTING

White-box testing (also known as clear box testing, glass box testing, and transparent box testing and structural testing) tests internal structures or workings of a program, as opposed to the functionality exposed to the end-user. 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). This test allows the tester to

- ☐ Check whether all independent paths within a module have been exercised at least once.
- ☐ Exercise all the logical decisions on their false sides.
- ☐ Execute all loops and their boundaries and within their bounds.
- ☐ Exercise the internal data structure to ensure their validity.
- ☐ Ensure whether all the possible validity checks and validity look ups have been provided to validate data entry.

Techniques used in white-box testing include:

- ☐ API testing (application programming interface) - testing of the application using public and private APIs
- ☐ Code coverage - creating tests to satisfy some criteria of code coverage (e.g., the test designer can create tests to cause all statements in the program to be executed at least once)
- ☐ Fault injection methods - intentionally introducing faults to gauge the efficacy of testing strategies.

BLACK BOX TESTING

It is case design method used on the functional requirements of the software. It will help a software engineer to derive sets of input conditions that will exercise all the functional requirements of the program. Black box testing attempts to find errors in the following categories.

- ☐ Incorrect or missing functions
- ☐ Interface errors
- ☐ Errors in external database access
- ☐ Performance errors
- ☐ Initialization and termination errors

One advantage of the black box technique is that no programming knowledge is required. Whatever biases the programmers may have had, the tester likely has a different set and may emphasize different areas of functionality. On the other hand, black-box testing has been said to be "like a walk in a dark labyrinth without a flashlight. Because they do not examine the source code, there are situations when

a tester writes many test cases to check something that could have been tested by only one test case, or leaves some parts of the program untested.

This method of test can be applied to all levels of software testing: unit, integration, system and acceptance. It typically comprises most if not all testing at higher levels, but can also dominate unit testing as well.

TESTING STRATEGIES:

A test strategy is an outline that describes the testing approach of the software development cycle. It is created to inform project managers, testers, and developers about some key issues of the testing process. This includes the testing objective, methods of testing new functions, total time and resources required for the project, and the testing environment.

The test strategy describes the test level to be performed. There are primarily three levels of testing: unit testing, integration testing, and system testing. In most software development organizations, the developers are responsible for unit testing. Individual testers or test teams are responsible for integration and system testing.

UNIT TESTING:

Unit testing, also known as component testing refers to tests that verify the functionality of a specific section of code, usually at the function level. In an object-oriented environment, this is usually at the class level, and the minimal unit tests include the constructors and destructors.

These types of tests are usually written by developers as they work on code (white-box style), to ensure that the specific function is working as expected. One function might have multiple tests, to catch corner cases or other branches in the code. Unit testing alone cannot verify the functionality of a piece of software, but rather is used to assure that the building blocks the software uses work independently of each other.

INTEGRATION TESTING

Integration testing is any type of software testing that seeks to verify the interfaces between components against a software design. Software components may be integrated in an iterative way or all together ("big bang"). Normally the former is considered a better practice since it allows interface issues to be localized more quickly and fixed.

Integration testing works to expose defects in the interfaces and interaction between integrated components (modules). Progressively larger groups of tested software components corresponding to elements of the architectural design are integrated and tested until the software works as a system.

SYSTEM TESTING

System testing is actually a series of different tests whose primary purpose is to fully exercise the computer based system. The following were the types which were carried out for the system.

VALIDATION TESTING

The validation testing can be defined in many way but simple definition is that, validation succeeds when the software function in a manner that can be reasonably expected by the end.

5.3 TEST CASE

A test case in software engineering is a set of conditions or variables under which a tester will determine whether an application or software system is working correctly or not. The mechanism for determining whether a software program or system has passed or failed such a test is known as a test oracle. In some settings, an oracle could be a requirement or use case, while in others it could be a heuristic. It may take many test cases to determine that a software program or system is considered sufficiently scrutinized to be released. Test cases are often referred to as test scripts, particularly when written. Written test cases are usually collected into test suites.

Formal Test Cases

A formal written test-case is characterized by a known input and by an expected output, which is worked out before the test is executed. The known input should test a precondition and the expected output should test a post condition.

Informal Test Cases

In scenario testing, hypothetical stories are used to help the tester think through a complex problem or system. These scenarios are usually not written down in any detail. They can be as simple as a diagram for a testing environment or they could be a description written in prose. The ideal scenario test is a story that is motivating, credible, complex, and easy to evaluate. They are usually different from test cases in that test cases are single steps while scenarios cover a number of steps of the key.

Sl. No.	Condition	Action
1a.	Username is correct, password is wrong. Login button is clicked.	An error message is displayed as "Username password combination is wrong".
1b.	Username is wrong, password is correct. Login button is clicked.	An error message is displayed as "Username password combination is wrong".
1c.	Username and password both are wrong. Login button is clicked.	An error message is displayed as "Username password combination is wrong".
2.	Username or password or both are kept blank. Login button is clicked.	A warning is issued by the browser "Please fill out this field"
3.	Username and password are correct. Login button is clicked.	The home page of the administrator (admin-welcome.php) is opened in the browser.

TEST CASE RESULTS

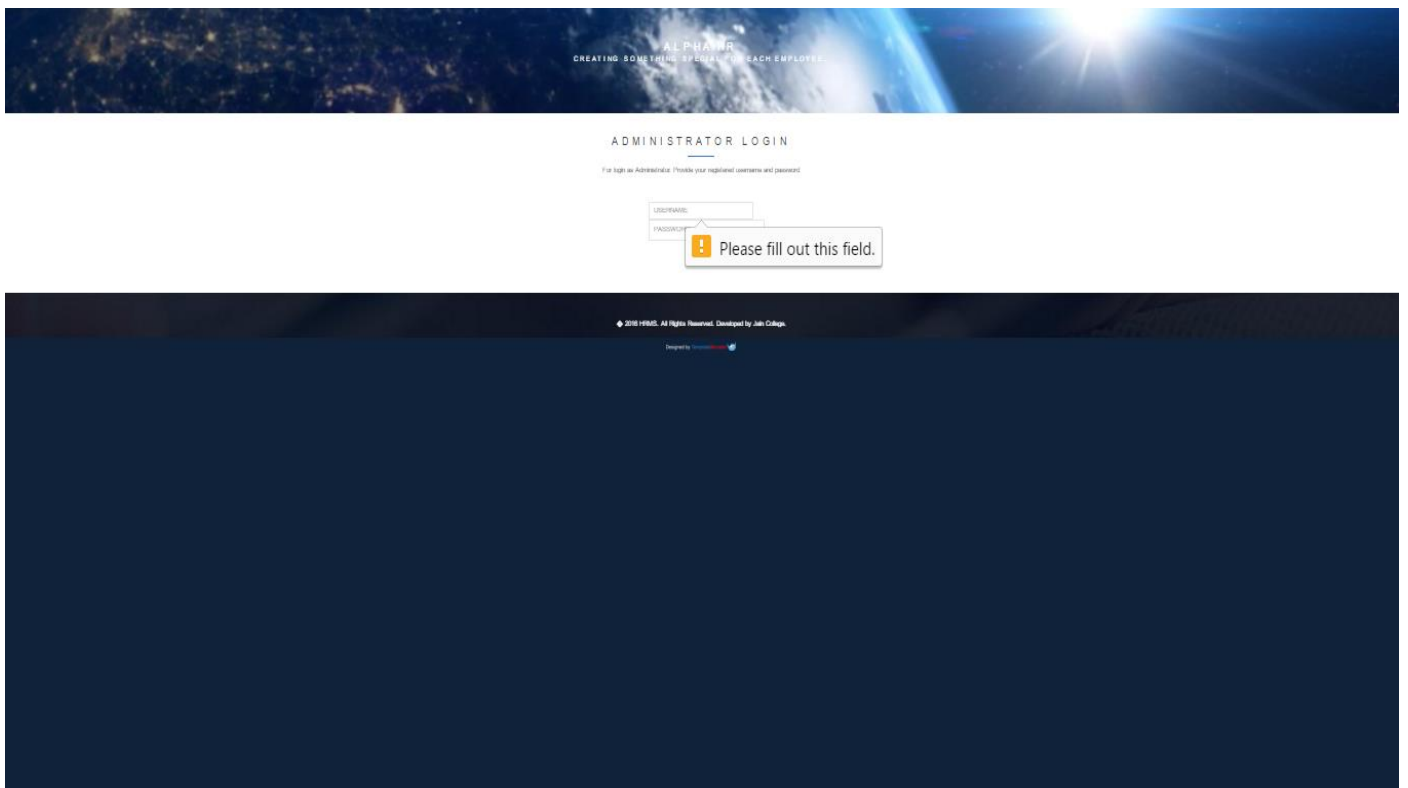
Test Case: 1

Module : Administrator

Input Name : Empty

Input Password: Empty

Observed Output : Please fill out this field”\



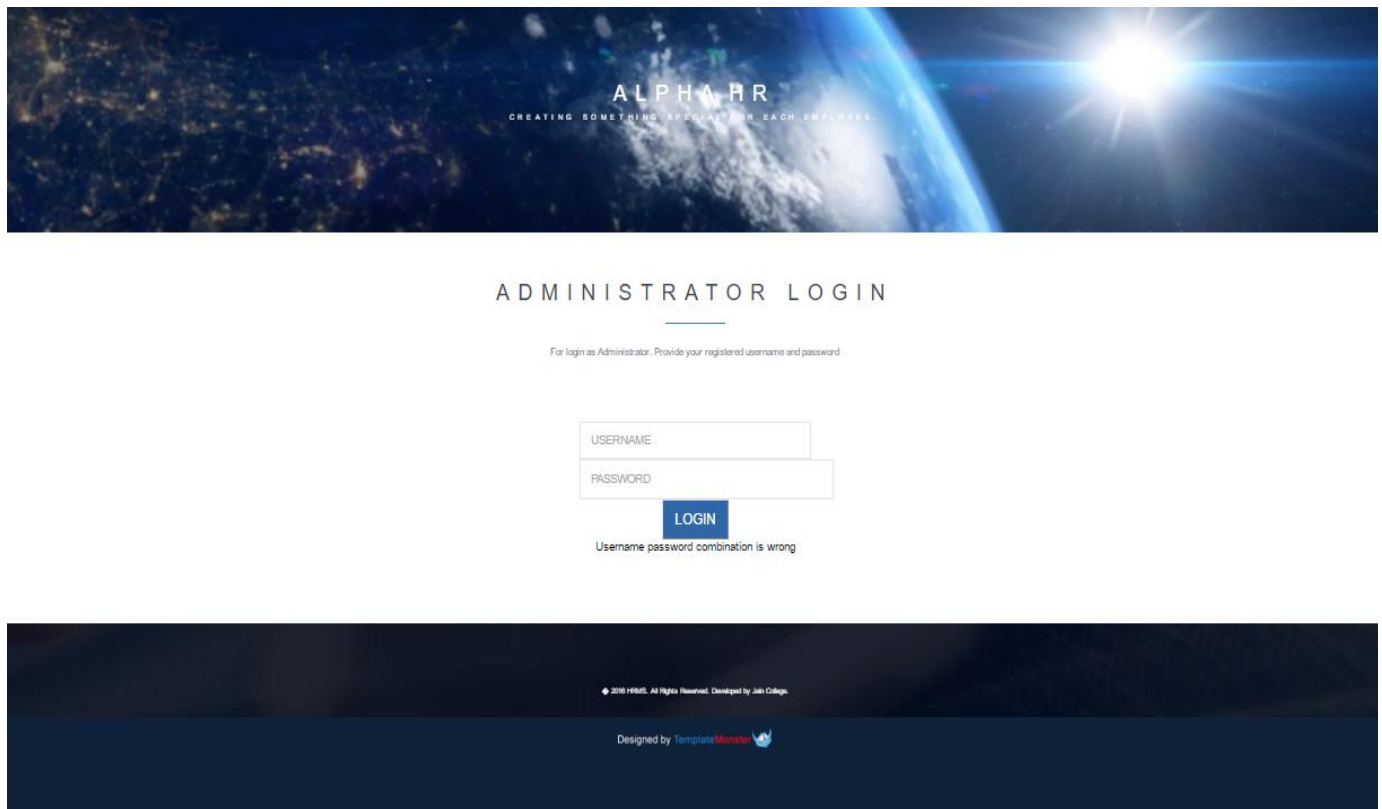
Test Case: 2

Module : Administrator

Input Name : Admin name is correct

Input Password: password is wrong

Observed Output : “ User name password combination is wrong



ALPHA HR
CREATING SOMETHING SPECIAL FOR EACH EMPLOYEE.

ADMINISTRATOR LOGIN

For login as Administrator, Provide your registered username and password.

USERNAME

PASSWORD

LOGIN

Username password combination is wrong

© 2018 HRMS. All Rights Reserved. Developed by Jain College.

Designed by [TemplateMonster](#)

Test Case: 3

Module : Admin Profile Update

Input Email : Missed out @

Observed Output : “ Please include an ‘@’ in the email address. ‘Address’ is missing an ‘@’ ”



UPDATE PROFILE

Use this page to update your profile.

	Update your profile
Name	<input type="text" value="James"/>
Designation	<input type="text" value="Admin"/>
E-mail	<input type="text" value="admingmail.com"/>
Mobile	<input type="text" value="98876"/>

UPDATE

BACK

! Please include an '@' in the email address. 'admingmail.com' is missing an '@'.

© 2016 HRMS. All Rights Reserved. Developed by Jain College.

Designed by [TemplateMonster](#)

Test Case: 4

Module : Admin Profile Update

Input Number : Enter less then 10 digit

Observed Output : “ Please lengthen this text to 10 characters or more(your currently using 1 character) ”

ALPHA HR
CREATING SOMETHING SPECIAL FOR EACH EMPLOYEE.

UPDATE PROFILE
Use this page to update your profile.

Update your profile	
Name	<input type="text" value="James"/>
Designation	<input type="text" value="Admin"/>
E-mail	<input type="text" value="james@gmail.com"/>
Mobile	<input type="text" value="1"/>

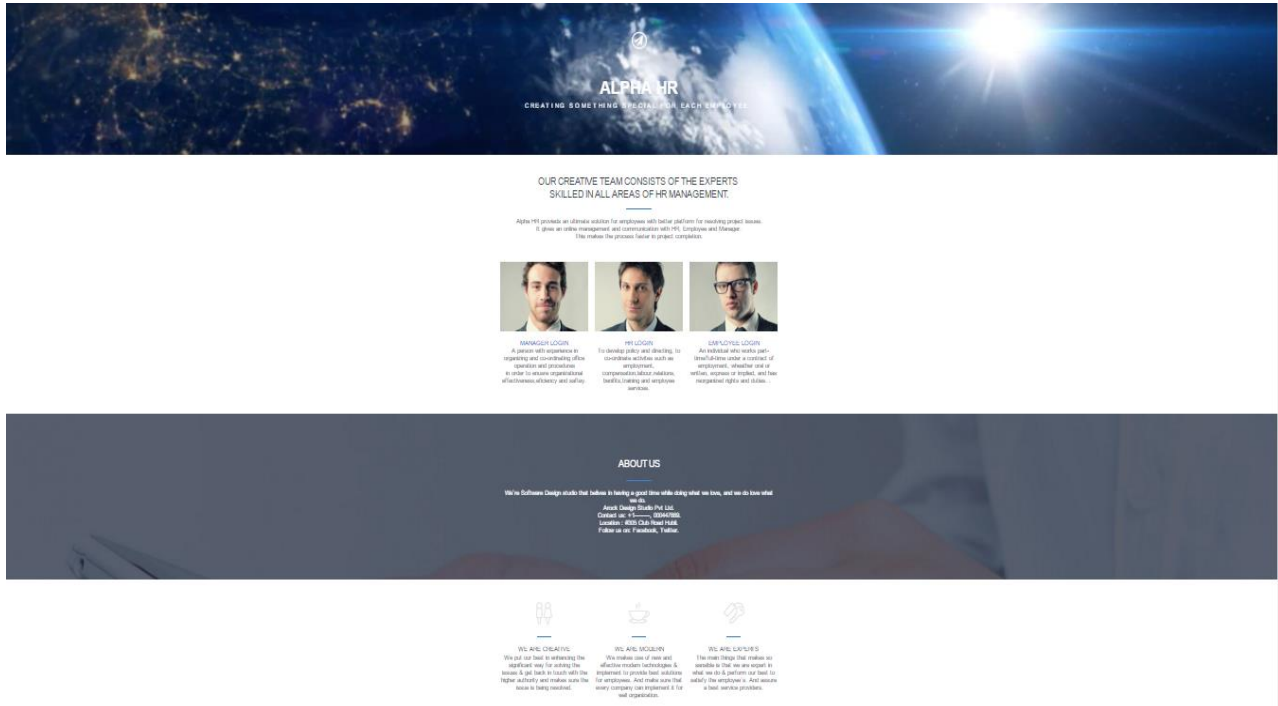
UPDATE BACK

Please lengthen this text to 10 characters or more (you are currently using 1 characters).

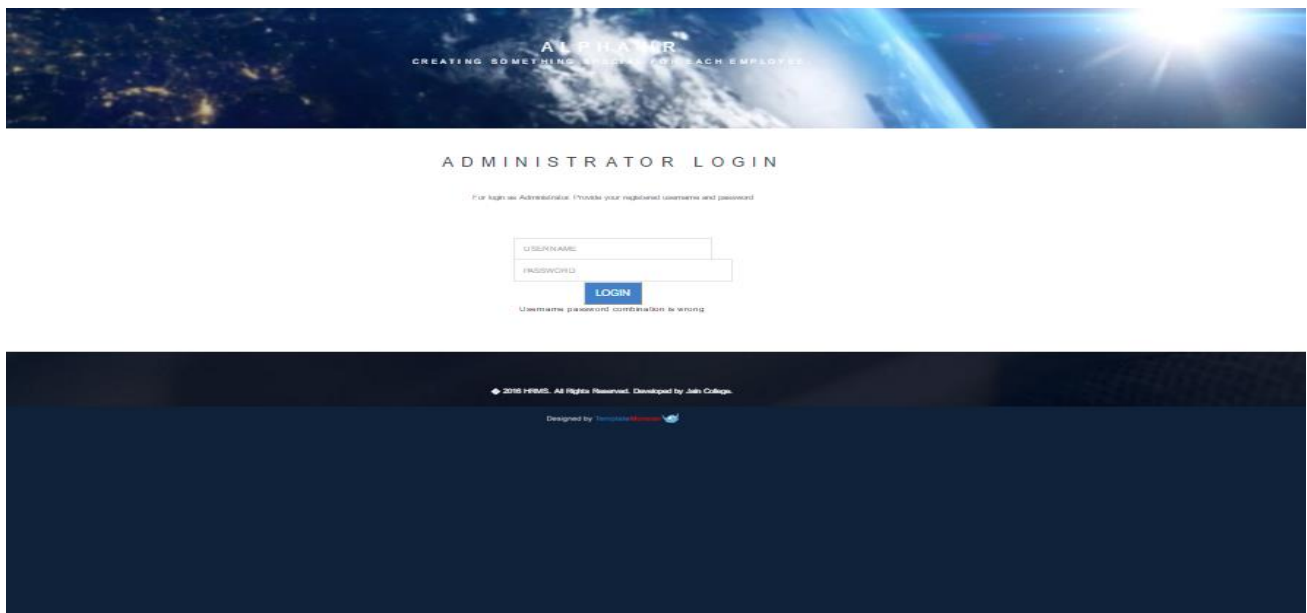
© 2018 HRMS. All Rights Reserved. Developed by Jain College.
Designed by [TemplateMonster](#)

SCREEN SHOTS

HOME PAGE



ADMINISTRATOR LOGIN



VIEW ISSUE PAGE



VIEW ISSUES							
#	Subject	Description	Date	Severity	Comments	Status	Reply
1	Unsatified with Manager	I am not satified with Manager Reply: "	2015-03-12	1	amok	Issue has been resolved	
7	I'm not going to work	my don't want to work with Manager Reply: "My beta have meeting on 5/19/20"	2015-03-23	3	ambush	Issue has been resolved it got updated	active the issue immediately
14	not working	philosophy to the staff Manager Reply: "Hello Hello how are you?"	2015-04-10	1	ambush	Issue has been resolved it did have meeting on 20th	
15	I about	Relationships around in which with Manager Reply: "	2015-04-11	1	ambush	New complaint. Waiting for manager reply	
16	Unsatified with Manager	unable to work with manager because the work has try to use for coordination Manager Reply: "I have taken many notes but those are not complete how through"	2015-03-11	1	warjay	Issue has been resolved it has meeting on 1st may and we can solve that...	
17	hello	hello Manager Reply: "hey hi"	2015-04-14	1	warjay	Manager reply received. Waiting for showing results	Reply
18	about	greeting Manager Reply: "	2015-04-14	1	warjay	New complaint. Waiting for manager reply	
21	hello	hello Manager Reply: "	2015-04-14	1	warjay	New complaint. Waiting for manager reply	

© 2015 ALPHA HR. All Rights Reserved. Developed by Jain College.

REPLY FORUM



ComplaintId	17
Reply	<div></div>

[SUBMIT](#) [BACK](#)

© 2015 ALPHA HR. All Rights Reserved. Developed by Jain College.

Designed by [TemplateMonster](#)

CHAT FORUM



CHAT FORUM

USERS
SANGAY



EMPLOYEE WELCOME PAGE



WELCOME EMPLOYEE

Official Home Page For Employees




MY PROFILE



ISSUE BOX



HR HOME PAGE




ALPHA HR
CREATING SOMETHING SPECIAL FOR EACH EMPLOYEE

WELCOME HR

Official Home Page Of HR


LOGOUT



REGISTRATION FORM

MANAGER REGISTRATION


EMPLOYEE REGISTRATION



MY PROFILE

EDIT PROFILE

EDIT PASSWORD




ISSUE BOX

VIEW ISSUE'S

© 2016 HRMS. All Rights Reserved. Developed by Jain College.

Designed by [TemplateMonster](#)

EDIT PROFILE



ALPHA HR
CREATING SOMETHING SPECIAL FOR EACH EMPLOYEE

YOUR PROFILE

Your current profile:

Name	Akash
Designation	Senior Manager
Mobile	998876655
E-mail	akash@gmail.com

UPDATE PROFILE BACK

© 2016 HRMS. All Rights Reserved. Developed by Jain College.

EDIT PASSWORD



New Password	<input type="password"/>
Confirm Password	<input type="password"/>

SUBMIT

BACK

© 2016 HRMS. All Rights Reserved. Developed by Jain College.

Designed by [TemplateMonster](#)

RASIE ISSUE



COMPLAINT FORM

Name	<input type="text"/>
Description	<div><div></div></div>
Severity	<input type="text"/>

SUBMIT

© 2016 HRMS. All Rights Reserved. Developed by Jain College.

Designed by [TemplateMonster](#)

❑ CONCLUSION

CONCLUSION

The current version of the software has selected few features for implementation, as per the scope of this project. This project will go a long way in helping the companies to streamline the management of employees and will lead to happy employees who will work for the betterment of the company. The process will also be quick and easy to maintain.

FUTURE ENCHANCEMENT

FUTURE ENCHANCEMENT

- The status of the complaints will popped as SMS alerts.
- The chat forum will enables even Manager and HR to send messages and also have conversation with employees.
- The chat forum enables voice calling and video calling with team members.

❑ REFERENCES AND BIBLIOGRAHY

WEBSITE

www.google.com

www.w3schools.com

www.stackoverflow.com

BOOKS

Web development by Robert Sabesta

Notes provided in the Institute