Introduction of the Gym Management System:

Our Gym Management Software is a gym and health club membership management system. You can keep records on your members, their memberships, and have quick and easy communication between you and your members. Gym Management also includes a booking system, point of sale, banking, accounting, concessions and has a range of reports that help in the management of your club.

Our Gym Management Software is a complete gym and recreation facility system program which looks after all of your members, memberships and activities. It is designed for gyms, recreation centers, and health clubs.

Our Gym management Software provides lots of functions such data entry of customer, keeping records of all the things about customer's fees, plan, and physical fitness which help to provide good quality of services to customer from Gym managers.

In this proposed system also provide the total information about machinery and data of coaches is also stored in it. Services provided by Gym are also handled by this system.

This system structure is become very simple to understand because of Data Flow Diagram provided by us. Context level Diagram and Some chart are also available in this case study. The demo of using the software such as customer detail form, data base of software is also provided by us.

Objective of the Gym Management System:

- The main objective of the project is to develop software that facilitates the data storage, data maintenance and its retrieval for the gym in an igneous way.
- To store the record of the customers, the staff that has the privileges to access, modify and delete any record and finally the service, gym provides to its customers.
- Also, only the staff has the privilege to access any database and make the required changes, if necessary.
- To develop easy-to-use software which handles the customer-staff relationship in an effective manner.
- To develop a user friendly system that requires minimal user training. Most of features and function are similar to those on any windows platform.

Limitation of Existing System:

- The existing System such as Gym Master is not as much as user (Customer) friendly as compare to our Proposed System.
- The communication with members is not well in Existing System because all the data is handled by Gym manager.
- Customers don't get full accessibility to Gym center and all permissions are allowed only for Gym manager.
- Today's System cannot take effort out of finances and debt collection.
- Today's systems are time taking software and cannot be easily install in Operating System like Linux , Vista , Mc-OS, and Novel. And also need high configuration of PC. Normal PC's cannot install in it.
- Lot of memory space is required for installing existing software.
- Existing Gym management software cannot perform all operation expected by manager such as keeping record of machinery maintenance and service data.

Abstract of the Project Gym Management System:

Gym Management System is an online service that can be setup for your gym to help manage classes, memberships, receive payments (merchant and cash), keep track with detailed statistics, customer management, surveys and it even has an online store so you can sell products to your customers.

- Its simple, it's effective and it's the way customers want their gym!
- Here is our feature list which is continually growing:
- Manage customers
- Manage customer health question forms
- Manage customer surveys
- Screenshot of Customer Options
- Complete site statistics (graphs) membership lists
- Screenshot of User Statistics graphs
- Complete payment statistics with downloadable content to excel and csv file format.
- Export functions; to download customer details to create mailing lists, databases…
- Manage your trainers and class schedules.
- Class management
- Create recurring classes and class types
- Create multiple locations and trainers
- Create plans & passes
- Manage customer barcode/RFID/membership cards for customers.
- Customers can see their own statistics and payment history. Complete Online Store for your products, membership plans & passes.

Some salient features are …

1. Product and Component based
2. Creating & Changing Gyms at ease
3. Query Gym List to any depth
4. Reporting & Charting in more comprehensive way
5. User Accounts to control the access and maintain security
6. Simple Status & Resolutions
7. Multi-level Priorities & Severities.
8. Targets & Milestones for guiding the programmers
9. Attachments & Additional Comments for more information
10. Robust database back-end

Modules & Description

1. **Authenticate User**

The Gym Tracking System first activates the login form. Here the user enters the User name and password and our system starts the authentication process in which the username and password are matched with the existing username and password in the database. If the password matches then it is allowed to the main page else it warns the user for Invalid User name and password.

After successful authentication the system activates menus. The activity log also prepared for failures and security.

2. **Products**

- **List Of Products**

  After successful authentication the user is provided with the list existing products. Here the user can view the details of products and can modify the existing products. This project even provides the facility of adding new projects.

- **Product Versions**

  All the products are maintained in several versions. As it is not possible to complete the whole project in a single version Features required for the product are categorized into several version with dead lines. And the versions are completed according to their dead line dates. Here the user can add new versions to a product or can modify the existing details of version.

- **Product Users**

In order to complete the project each product is allotted with Resources or users. First all the employees with their names and qualifications are stored in the database. Each user is allotted to the product based on their rating,

Qualification and designation. For each user Effective date is stored which specifies the total period a user is valid for that product.

3. **Gym Details**

- **Gym Details**

  In this module the user is provided with the facility for adding Gyms or updating the existing Gyms. As the number of Gyms for a product can be very large this system is provided with efficient filtering. The user can filter the Gyms based on the priority, database, operating system and status. After the user applies filter the list of Gyms are displayed from the database.

- **Gym History**

  Here the Gym history is maintained. All the solutions given for the Gym resolution by various users are stored. As the Gym needs several techniques or methods for resolution it is important to store the history of the Gym.

- **Gym Assignee**

  This displays the list of users for whom the Gym is assigned for resolution. As the Gym need to be resolved for completing the product several user are assigned to find a solution for the Gym. The user can add this Gym to a new user or he can modify the existing user details.

- **Gym Attachments**

  This gives a list of attachments for a particular Gym. The Gym can be of any type it can be a database Gym or a GUI Gym. So while you add a Gym you need to provide with the details of Gym. So the file attachments can be a document, database file or an image file. All then attachments are stored in a location along with the size and type of the file. Here the user can add a new attachment or can change the details of existing files.

4. **Gym Tracking**

- **Track Hierarchy**

All the Gyms saved in the database will have a particular hierarchy. There might be Gyms which can be related to the earlier Gyms saved in the database so our system is provided with a hierarchy. And user can add child nodes in this hierarchy or he can modify the existing values of the nodes. This hierarchy is based on the parent child relation ship between the Gyms.

- **Track Resolution**

This displays a list of all solutions provided by the users allotted to a Gym. This stores the action type and the necessary resolution provided by the user.

- **Track Resources**

This displays list of resources allotted to the project. As the Gyms need to be resolved resources are provided for the Gyms. These Resources will be the resources allotted to the project. The resources are allotted based on the rating of the employee.

5. **View**

- **Product Gym Hierarchy**

  This module is just for displaying the hierarchy for the easy Look of the Gyms. Here the Gyms are displayed in the form of parent child nodes. As it is difficult for the user to look at the vast number of Gyms in the database. And one cannot easily access the relation between the Gyms.

- **Product User Hierarchy**

  This module if for displaying the users allotted to the Gym. The users along with their name and designation are displayed in this module. Even in the allotment of resources there can be hierarchy between the employees depending on their designation. So this module simplifies the hierarchy among the employees.

6. Search

Our system provides with the feature of advanced search technique. Generally Number of Gyms for a project increased tremendously so if we want to know about a particular Gym It takes much amount of time. With the search screen provided one can filter the Gym's base on priority, product, severity, database and type of operating system. He can also list the Gyms between particular time based on the start date and end date. After Searching it displays a list of Gyms. From this list the user can modify the existing Gyms or can add a new Gym.

7. Admin

- **Users**

  All the users of this system are displayed in this module. One can add new user or can update the details of an existing user. Here the password provided by the user is encrypted before saving them to the database for proper security. This module saves the details like address, phone and email.

- **Configuration**

  All the Values that we are using in this system are configurable. Values like status, priority and others can be added dynamically on the screen. Suppose if we limit these fields by hot coding them and if the user wants to add a new value again he has to come to the developer of the product. So In order to avoid this it is provided with the feature of adding values from the screen. And the user can change the status to In Active whenever he wants.

- **Log View**

  In order for the efficient Tracking of the system logs are maintained. As the logs will be in vast it will be a problem for user for checking the database. The Log View module can be searched based on the user and Records between a start date and end date.

8. Logout

In this once the user clicks on Log out First the session variable is killed and then the system is redirected to the login page.

9. Prepare Logs

At all the stages, whenever user performs an operation by clicking a button, automatically the Gym Tracking System logs the activity.

SYSTEM ANALYSIS

INTRODUCTION TO SYSTEM ANALYSIS

System analysis is a process of gathering and interpreting facts, diagnosing problems and the information to recommend improvements on the system. It is a problem solving activity that requires intensive communication between the system users and system developers. System analysis or study is an important phase of any system development process. The system is studied to the minutest detail and analyzed. The system analyst plays the role of the interrogator and dwells deep into the working of the present system. The system is viewed as a whole and the input to the system are identified. The outputs from the organizations are traced to the various processes. System analysis is concerned with becoming aware of the problem, identifying the relevant and decisional variables, analyzing and synthesizing the various factors and determining an optimal or at least a satisfactory solution or program of action. A detailed study of the process must be made by various techniques like interviews, questionnaires etc. The data collected by these sources must be scrutinized to arrive to a conclusion. The conclusion is an understanding of how the system functions. This system is called the existing system. Now the existing system is subjected to close study and problem areas are identified. The designer now functions as a problem solver and tries to sort out the difficulties that the enterprise faces. The solutions are given as proposals. The proposal is then weighed with the existing system analytically and the best one is selected. The proposal is presented to the user for an endorsement by the user. The proposal is reviewed on user request and suitable changes are made. This is loop that ends as soon as the user is satisfied with proposal. Preliminary study is the process of gathering and interpreting facts, using the information for further studies on the system. Preliminary study is problem solving activity that requires intensive communication between the system users and system developers. It does various feasibility studies. In these studies a rough figure of the system activities can be obtained, from which the decision about the strategies to be followed for effective system study and analysis can be taken.

Existing System

In the existing system the exams are done only manually but in proposed system we have to computerize the exams using this application.

- Lack of security of data.
- More man power.
- Time consuming.
- Consumes large volume of pare work.
- Needs manual calculations.
- No direct role for the higher officials

Proposed System

The aim of proposed system is to develop a system of improved facilities. The proposed system can overcome all the limitations of the existing system. The system provides proper security and reduces the manual work.

- Security of data.
- Ensure data accuracy's.
- Proper control of the higher officials.
- Minimize manual data entry.
- Minimum time needed for the various processing.
- Greater efficiency.
- Better service.
- User friendliness and interactive.
- Minimum time required.

FEASIBILITY STUDY

Feasibility study is made to see if the project on completion will serve the purpose of the organization for the amount of work, effort and the time that spend on it. Feasibility study lets the developer foresee the future of the project and the usefulness. A feasibility study of a system proposal is according to its workability, which is the impact on the organization, ability to meet their user needs and effective use of resources. Thus when a new application is proposed it normally goes through a feasibility study .The document provide the feasibility of the project that is being designed and lists various areas that were considered very carefully during the feasibility study of this project such as Technical, Economic and Operational feasibilities. The following are its features:

TECHNICAL FEASIBILITY

The system must be evaluated from the technical point of view first. The assessment of this feasibility must be based on an outline design of the system requirement in the terms of input, output, programs and procedures. Having identified an outline system, the investigation must go on to suggest the type of equipment, required method developing the system, of running the system once it has been designed.

Technical Gyms raised during the investigation are:

- Does the existing technology sufficient for the suggested one?
- Can the system expand if developed?

The project should be developed such that the necessary functions and performance are achieved within the constraints. The project is developed within latest technology. Through the technology may become obsolete after some period of time, due to the fact that never version of same software supports older versions, the system may still be used. So there are minimal constraints involved with this project. The system has been developed using PHP the project is technically feasible for development.

ECONOMIC FEASIBILITY

The developing system must be justified by cost and benefit. Criteria to ensure that effort is concentrated on project, which will give best, return at the earliest. One of the factors, which affect the development of a new system, is the cost it would require.

The following are some of the important financial questions asked during preliminary investigation:

- The costs conduct a full system investigation.
- The cost of the hardware and software.
- The benefits in the form of reduced costs or fewer costly errors.

Since the system is developed as part of project work, there is no manual cost to spend for the proposed system. Also all the resources are already available, it give an indication of the system is economically possible for development.

2.1 PRODUCT PERSPECTIVE

Web traffic can be analyzed by viewing the traffic statistics found in the web server log file, an automatically-generated list of all the pages served. A hit is generated when any file is served. The page itself is considered a file, but images are also files, thus a page with 5 images could generate 6 hits (the 5 images and the page itself). A page view is generated when a visitor requests any page within the web site – a visitor will always generate at least one page view (the main page) but could generate many more.. Web Traffic Analyzer is aimed towards the vendors who want to reach out to the maximum cross-section of customer and common people who can be potential customer. This project envisages bridging the gap between the seller, the retailer and the customer. Web Traffic Analyzer should be user-friendly, 'quick to learn' and reliable software for the above purpose. OSM is intended to be a stand-alone product and should not depend on the availability of other software. It should run on both UNIX and Windows based platform.

2.2 PRODUCT FUNCTIONS

Web traffic is the amount of data sent and received by visitors to a web site. It is a large portion of Internet traffic. This is determined by the number of visitors and the number of pages they visit. Sites monitor the incoming and outgoing traffic to see which parts or pages of their site are popular and if there are any apparent trends, such as one specific page being viewed mostly by people in a particular country. There are many ways to monitor this traffic and the gathered data is used to help structure sites, highlight security problems or indicate a potential lack of bandwidth — not all web traffic is welcome. Some companies offer advertising schemes that, in return for increased web traffic (visitors), pay for screen space on the site. Sites also often aim to increase their web traffic through inclusion on search engines and through Search engine optimization.

2.3 USER CHARACTERISTICS

We have 2 levels of users

- User module: This is a normal level of user who will be very few number of functionality for website
- Administration module: This user is an admin type who has full rights on the system.

2.4 GENERAL CONSTRAINTS

The amount of traffic seen by a web site is a measure of its popularity. By analysing the statistics of visitors it is possible to see shortcomings of the site and look to improve those areas. It is also possible to increase (or, in some cases decrease) the popularity of a site and the number of people that visit it.

2.5 ASSUMPTIONS AND DEPENDENCIES

All the data entered will be correct and up to date. This software package is developed using HTML as front end which is supported by Aapache Server system. MySQL as the back end which is supported by Window 7.

3.1.1 User Interface

- HTML has been used for developing the User Layout for the system
- PHPScript has been used for creating all the validations and client side scripting functionality
- CSS has been used for designing the web pages of the system

3.1.2 HARDWARE INTERFACE:

- Processor : Intel Pentium IV or more
- Ram : 512 MB or more
- Cache : 1 MB
- Hard Disk : 10 GB recommended

3.1.3 Software Interface:

- Client on Internet: Web Browser, Operating System (any)
- Web Server: Operating System (any), Apache 2
- Database: MySQL
- Scripting Language: HTML, PHPScript, JQuery

3.1.4 Communication Protocol

Following protocols are required to be permitted on the server side

- HTTP incoming request

3.2 Functional Requirements

- The system runs of apache server so it is needed that server must have apache server version 2.0 available

- We have used HTML for server side scripting so the current version of HTML must be available on the server
- MySQL database has been used for storing the data of the website
- HTML has been used for creating the layout of the web application
- CSS has been used for creating the designing of the webpages
- PHPScript scripting language has been implemented on the system for performing all of the Client Side Server Validation.

3.4. Classes and Objects of the Project

- Login Class: Used for performing all the operations of the login functionality.
- Page Class: Class for managing all the operations of the page.
- Traffic Class: Class for managing the traffic of the website
- IP Class: It has been used for storing all the IPs which hits the website
- Users Class: Class for managing all the user operations
- Permission Class: This class has been used for managing all the permissions level opeations.

3.5. Non-Functional Requirements

- Performance: System should be able handle multiple users at a time using any of the web browsers.
- Reliability: Database updating should follow transaction processing to avoid data inconsistency.
- Availability: The project will be deployed on a public shared server so it will be available all the time and will be accessible anywhere of the world using internet.
- Security: We have implemented a lot of security mechanism to avoid to hack the system by outer world.

- Maintainability: It is very easy to maintain the system. The system has been developed on HTML so anyone who has the knowledge of HTML, can easily maintain the system
- Portability: Yes this system is portable and we can switch the servers very easily.
- Browser Compatibility: The project being web based required compatibility with at least the popular web browsers. Microsoft Windows XP and above, Linux and Macintosh being the current popular operating system and Microsoft Internet Explorer, Mozilla Firefox, Opera, Safari and Google Chrome being the currently popular web browser.

Operating System / Browsers	Win 2000	WinXP	WinXPSP2	Win Vista	Win 7	Mac OS	Linux
	Modern Browsers						
IE 8.0	N/A	SUPP	SUPP	SUPP	SUPP	N/A	N/A
IE 7.0	N/A	N/A	N/A	N/A		N/A	
IE 6.0	N/A	N/A	N/A	N/A		N/A	
Firefox 3.5	N/A	SUPP	N/A	N/A		N/A	
Opera 9.23	N/A	SUPP	N/A	N/A		N/A	
Safari 9.27	N/A	SUPP	N/A	N/A		SUPP	
	"Legacy" Old Browsers						
IE5.5	N/A	N/A	N/A	N/A		N/A	
Netscape	N/A	N/A	N/A	N/A		N/A	

Security Testing of the Project

Testing is vital for the success of any software. no system design is ever perfect. Testing is also carried in two phases. first phase is during the software engineering that is during the module creation. second phase is after the completion of software. this is system testing which verifies that the whole set of programs hanged together.

White Box Testing:

In this technique, the close examination of the logical parts through the software are tested by cases that exercise species sets of conditions or loops. all logical parts of the software checked once. errors that can be corrected using this technique are typographical errors, logical expressions which should be executed once may be getting executed more than once and error resulting by using wrong controls and loops. When the box testing tests all the independent part within a module a logical decisions on their true and the false side are exercised , all loops and bounds within their operational bounds were exercised and internal data structure to ensure their validity were exercised once.

Black Box Testing:

This method enables the software engineer to device sets of input techniques that fully exercise all functional requirements for a program. black box testing tests the input, the output and the external data. it checks whether the input data is correct and whether we are getting the desired output.

Alpha Testing:

Acceptance testing is also sometimes called alpha testing. Be spoke systems are developed for a single customer. The alpha testing proceeds until the system developer and the customer agree that the provided system is an acceptable implementation of the system requirements.

Beta Testing:

On the other hand, when a system isto be marked as a software product, another process called beta testing is often conducted. During beta testing, a system is delivered among a number of potential users who agree to use it. The customers then report problems to the developers. This provides the product for real use and detects errors which may not have been anticipated by the system developers.

Unit Testing:

Each module is considered independently. it focuses on each unit of software as implemented in the source code. it is white box testing.

Integration Testing:

Integration testing aims at constructing the program structure while at the same constructing tests to uncover errors associated with interfacing the modules. modules are integrated by using the top down approach.

Validation Testing:

Validation testing was performed to ensure that all the functional and performance requirements are met.

System Testing:

It is executing programs to check logical changes made in it with intention of finding errors. a system is tested for online response, volume of transaction, recovery from failure etc. System testing is done to ensure that the system satisfies all the user requirements.

Implementation and Software Specification Testings

Detailed Design of Implementation

This phase of the systems development life cycle refines hardware and software specifications, establishes programming plans, trains users and implements extensive testing procedures, to evaluate design and operating specifications and/or provide the basis for further modification.

Technical Design

This activity builds upon specifications produced during new system design, adding detailed technical specifications and documentation.

Test Specifications and Planning

This activity prepares detailed test specifications for individual modules and programs, job streams, subsystems, and for the system as a whole.

Programming and Testing

This activity encompasses actual development, writing, and testing of program units or modules.

User Training

This activity encompasses writing user procedure manuals, preparation of user training materials, conducting training programs, and testing procedures.

Acceptance Test

A final procedural review to demonstrate a system and secure user approval before a system becomes operational.

Installation Phase

In this phase the new Computerized system is installed, the conversion to new procedures is fully implemented, and the potential of the new system is explored.

System Installation

The process of starting the actual use of a system and training user personnel in its operation.

Review Phase

This phase evaluates the successes and failures during a systems development project, and to measure the results of a new Computerized Transystem in terms of benefits and savings projected at the start of the project.

Development Recap

A review of a project immediately after completion to find successes and potential problems in future work.

Post-Implementation Review

A review, conducted after a new system has been in operation for some time, to evaluate actual system performance against original expectations and projections for cost-benefit improvements. Also identifies maintenance projects to enhance or improve the system.

THE STEPS IN THE SOFTWARE TESTING

The steps involved during Unit testing are as follows:

a. Preparation of the test cases.

b. Preparation of the possible test data with all the validation checks.

c. Complete code review of the module.

d. Actual testing done manually.

e. Modifications done for the errors found during testing.

f. Prepared the test result scripts.

The unit testing done included the testing of the following items:

1. Functionality of the entire module/forms.

2. Validations for user input.

3. Checking of the Coding standards to be maintained during coding.

4. Testing the module with all the possible test data.

5. Testing of the functionality involving all type of calculations etc.

6. Commenting standard in the source files.

After completing the Unit testing of all the modules, the whole system is integrated with all its dependencies in that module. While System Integration, We integrated the modules one by one and tested the system at each step. This helped in reduction of errors at the time of the system testing.

The steps involved during System testing are as follows:

- Integration of all the modules/forms in the system.
- Preparation of the test cases.
- Preparation of the possible test data with all the validation checks.
- Actual testing done manually.

- Recording of all the reproduced errors.
- Modifications done for the errors found during testing.
- Prepared the test result scripts after rectification of the errors.

The System Testing done included the testing of the following items:

1. Functionality of the entire system as a whole.
2. User Interface of the system.
3. Testing the dependent modules together with all the possible test data scripts.
4. Verification and Validation testing.
5. Testing the reports with all its functionality.

After the completion of system testing, the next following phase was the Acceptance Testing. Clients at their end did this and accepted the system with appreciation. Thus, we reached the final phase of the project delivery.

There are other six tests, which fall under special category. They are described below:

- Peak Load Test: It determines whether the system will handle the volume of activities that occur when the system is at the peak of its processing demand. For example, test the system by activating all terminals at the same time.
- Storage Testing: It determines the capacity of the system to store transaction data on a disk or in other files.
- Performance Time Testing: it determines the length of time system used by the system to process transaction data. This test is conducted prior to

implementation to determine how long it takes to get a response to an inquiry, make a backup copy of a file, or send a transmission and get a response.

- Recovery Testing: This testing determines the ability of user to recover data or re-start system after failure. For example, load backup copy of data and resume processing without data or integrity loss.

- Procedure Testing: It determines the clarity of documentation on operation and uses of system by having users do exactly what manuals request. For example, powering down system at the end of week or responding to paper-out light on printer.

- Human Factors Testing: It determines how users will use the system when processing data or preparing reports.

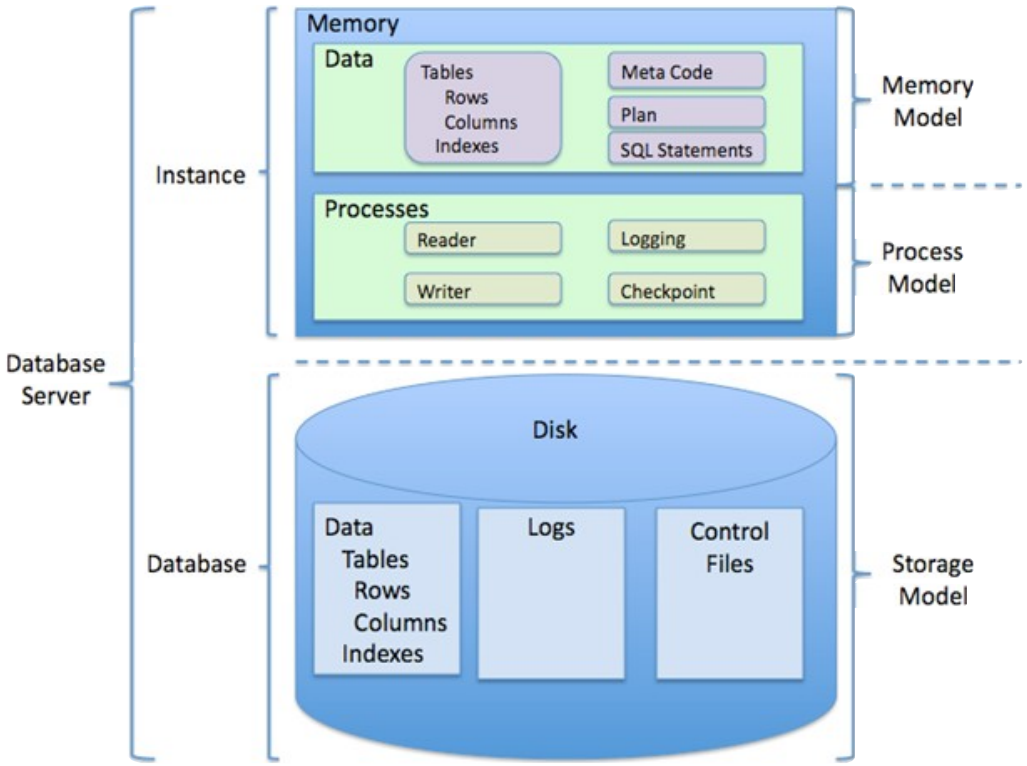
Project Category

Relational Database Management System (RDBMS) : This is an RDBMS based project which is currently using MySQL for all the transaction statements. MySQL is an opensource RDBMS System.

Brief Introduction about RDBSM :

A relational database management system (RDBMS) is a database management system (DBMS) that is based on the relational model as invented by E. F. Codd, of IBM's San Jose Research Laboratory. Many popular databases currently in use are based on the relational database model.

RDBMSs have become a predominant choice for the storage of information in new databases used for financial records, manufacturing and logistical information, personnel data, and much more since the 1980s. Relational databases have often replaced legacy hierarchical databases and network databases because they are easier to understand and use. However, relational databases have been challenged by object databases, which were introduced in an attempt to address the object-relational impedance mismatch in relational database, and XML databases.



Implementation Methodology:

Model View Controller or MVC as it is popularly called, is a software design pattern for developing web applications. A Model View Controller pattern is made up of the following three parts:

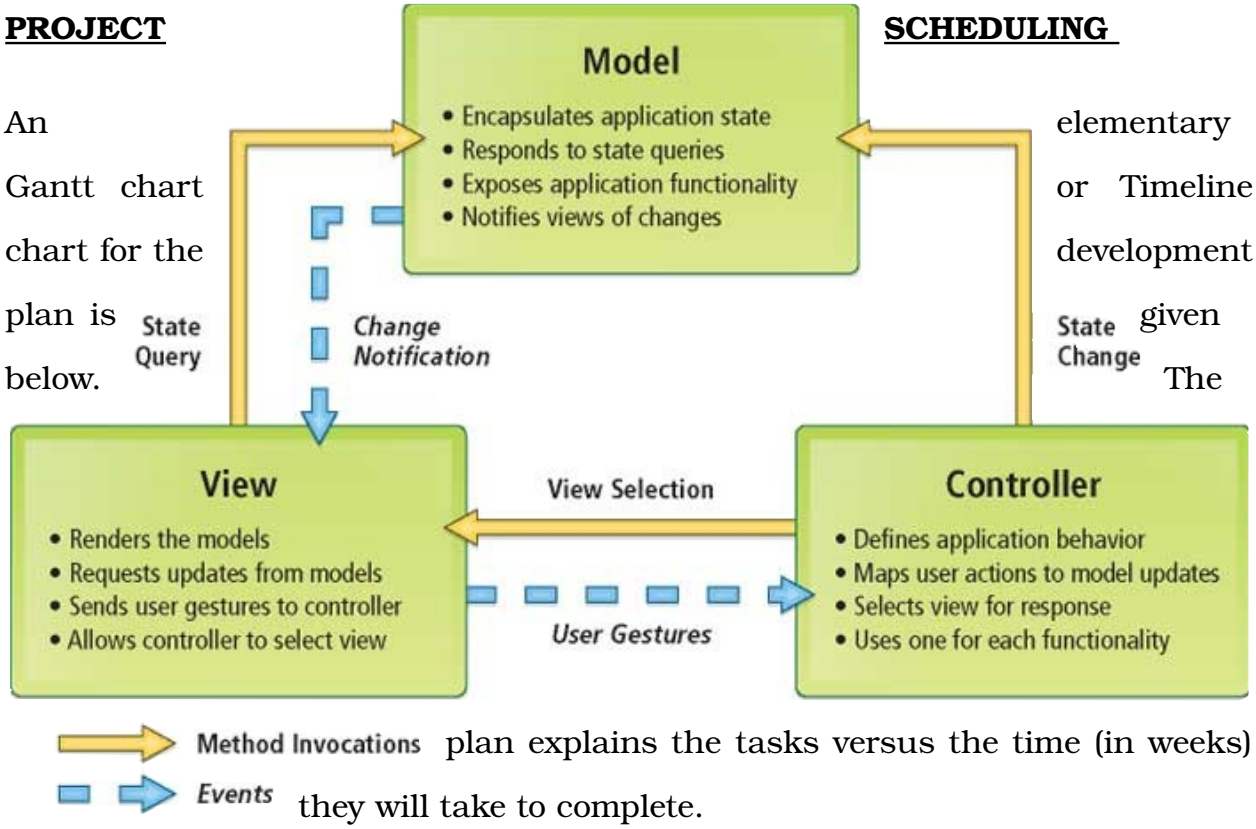
- **Model** - The lowest level of the pattern which is responsible for maintaining data.
- **View** - This is responsible for displaying all or a portion of the data to the user.
- **Controller** - Software Code that controls the interactions between the Model and View.

MVC is popular as it isolates the application logic from the user interface layer and supports separation of concerns. Here the Controller receives all requests for the application and then works with the Model to prepare any data needed by the View. The View then uses the data prepared by the Controller to generate a final presentable response. The MVC abstraction can be graphically represented as follows.

MVC (Model View Controller Flow) Diagram

PROJECT SCHEDULING

An elementary Gantt chart or Timeline chart for the development plan is given below. The plan explains the tasks versus the time (in weeks) they will take to complete.

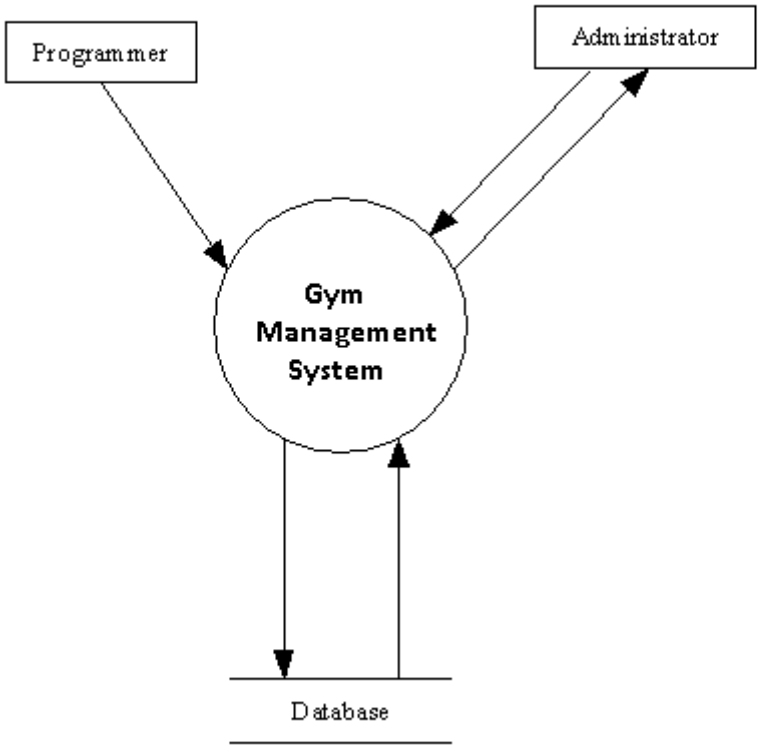


	January				February				March			
Requirement Gathering												
Analysis												
Design												
Coding												
Testing												
Implement												
	W 1	W 2	W 3	W 4	W 1	W 2	W 3	W 4	W 1	W 2	W 3	W4

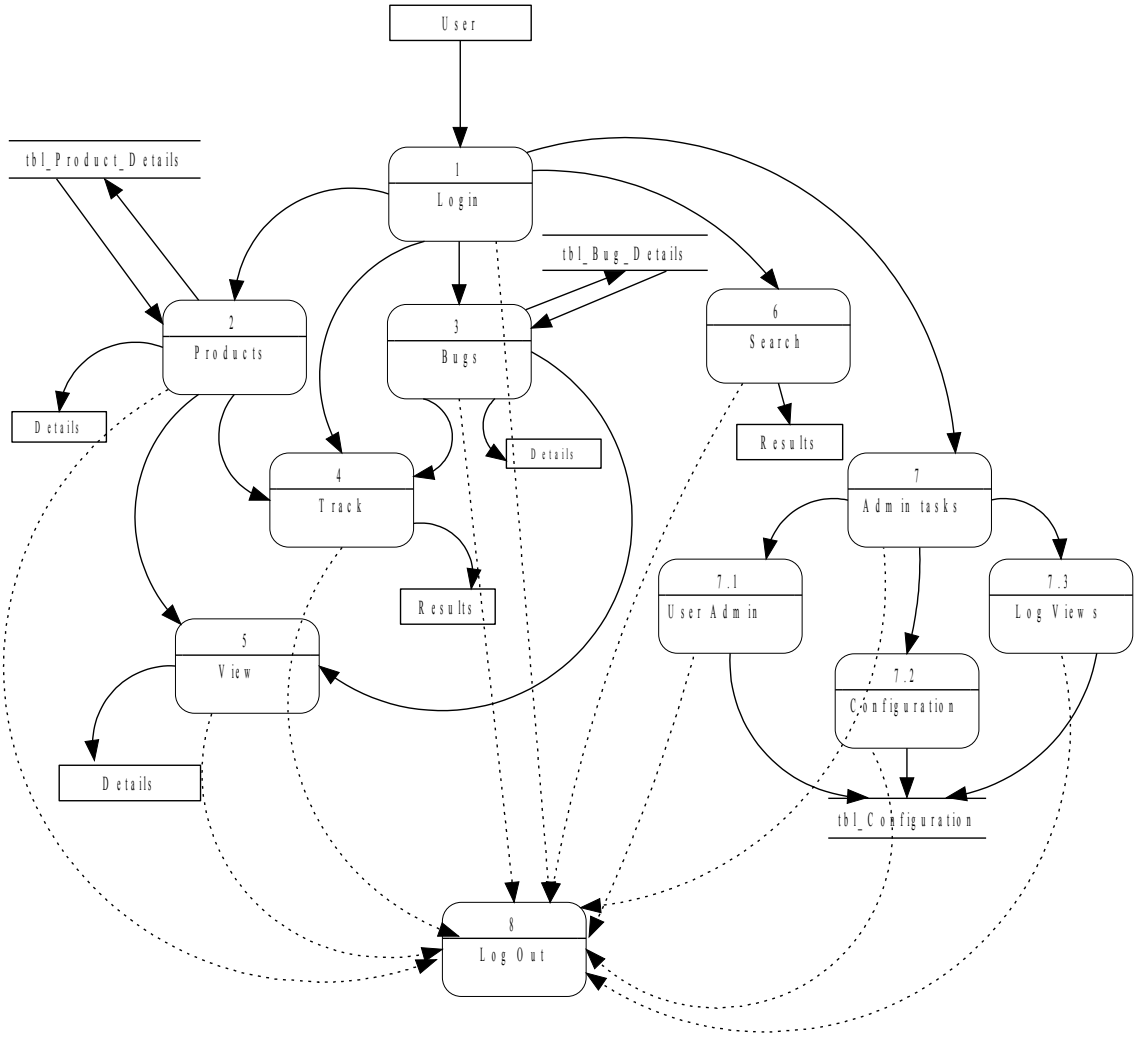
W*i*'s are weeks of the months, for i =1, 2, 3, 4

DATA FLOW DIAGRAMS

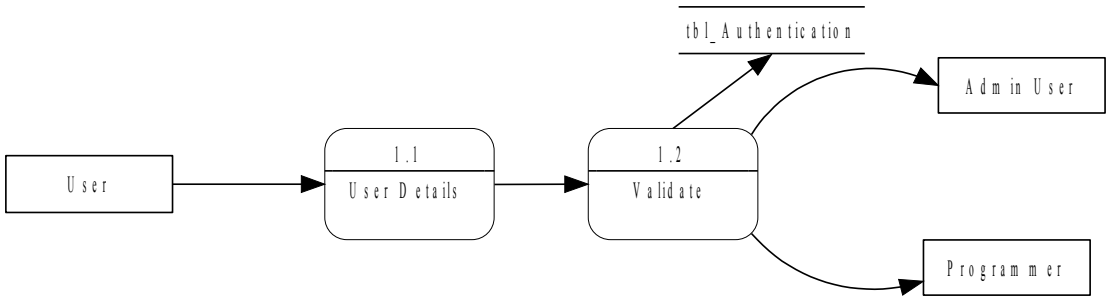
BTS - TOP LEVEL DIAGRAM



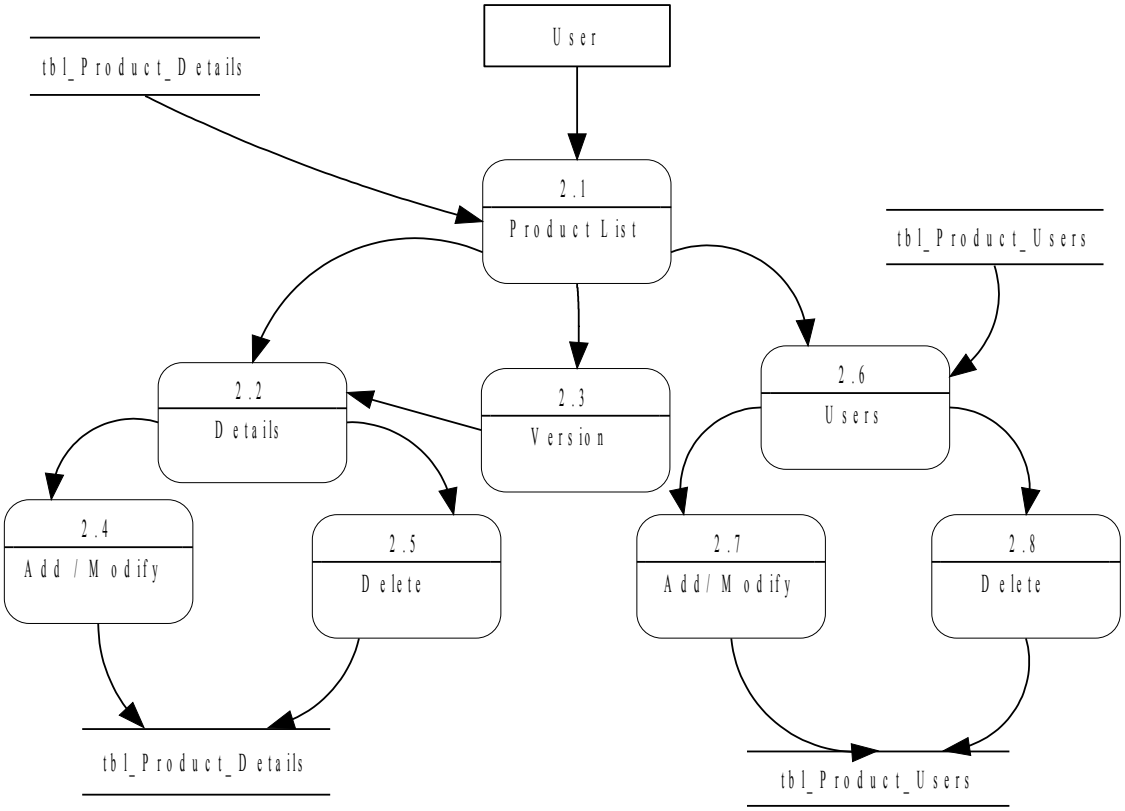
BTS - TOP LEVEL DIAGRAM



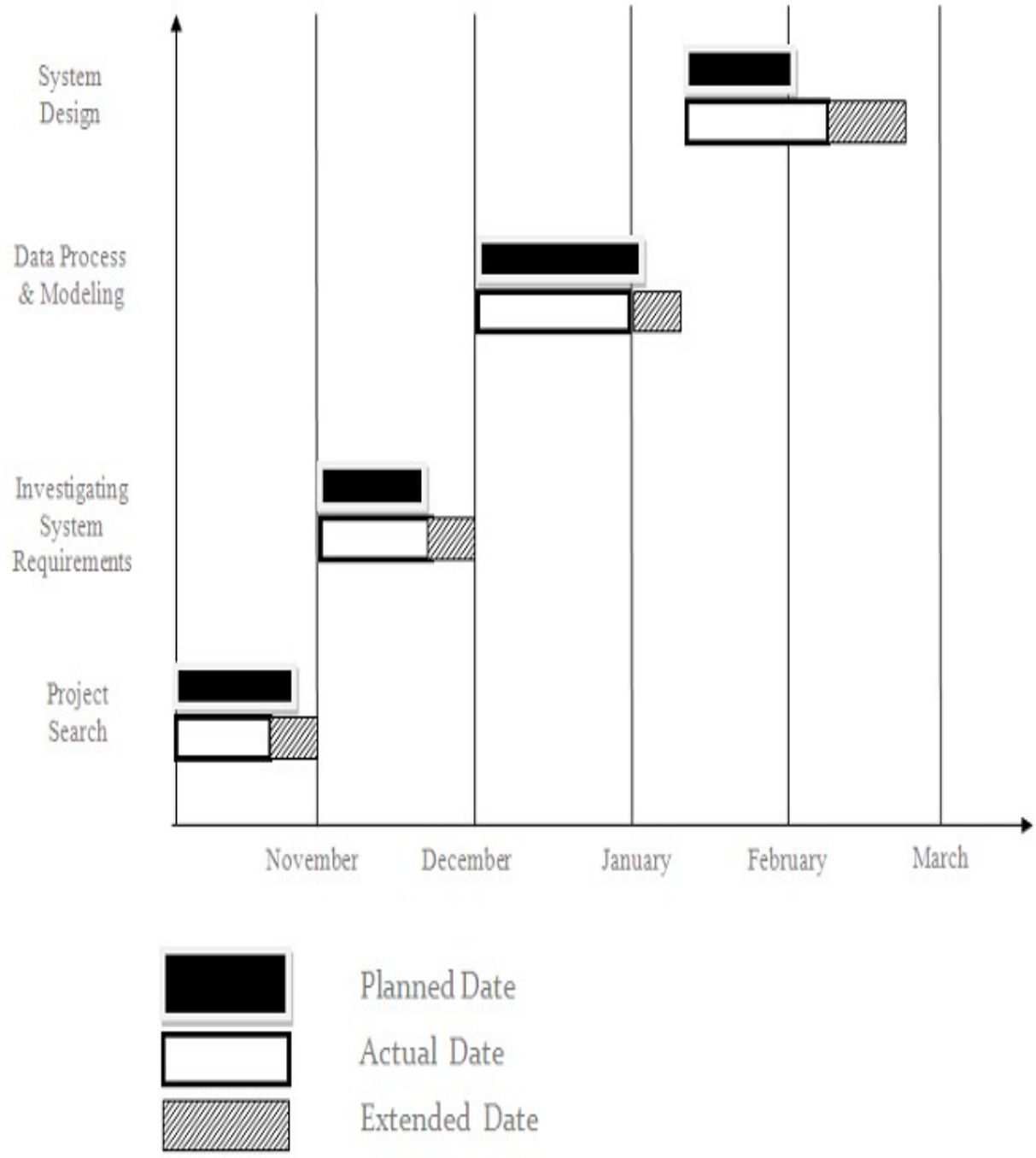
LOW LEVEL DIAGRAM - LOGIN



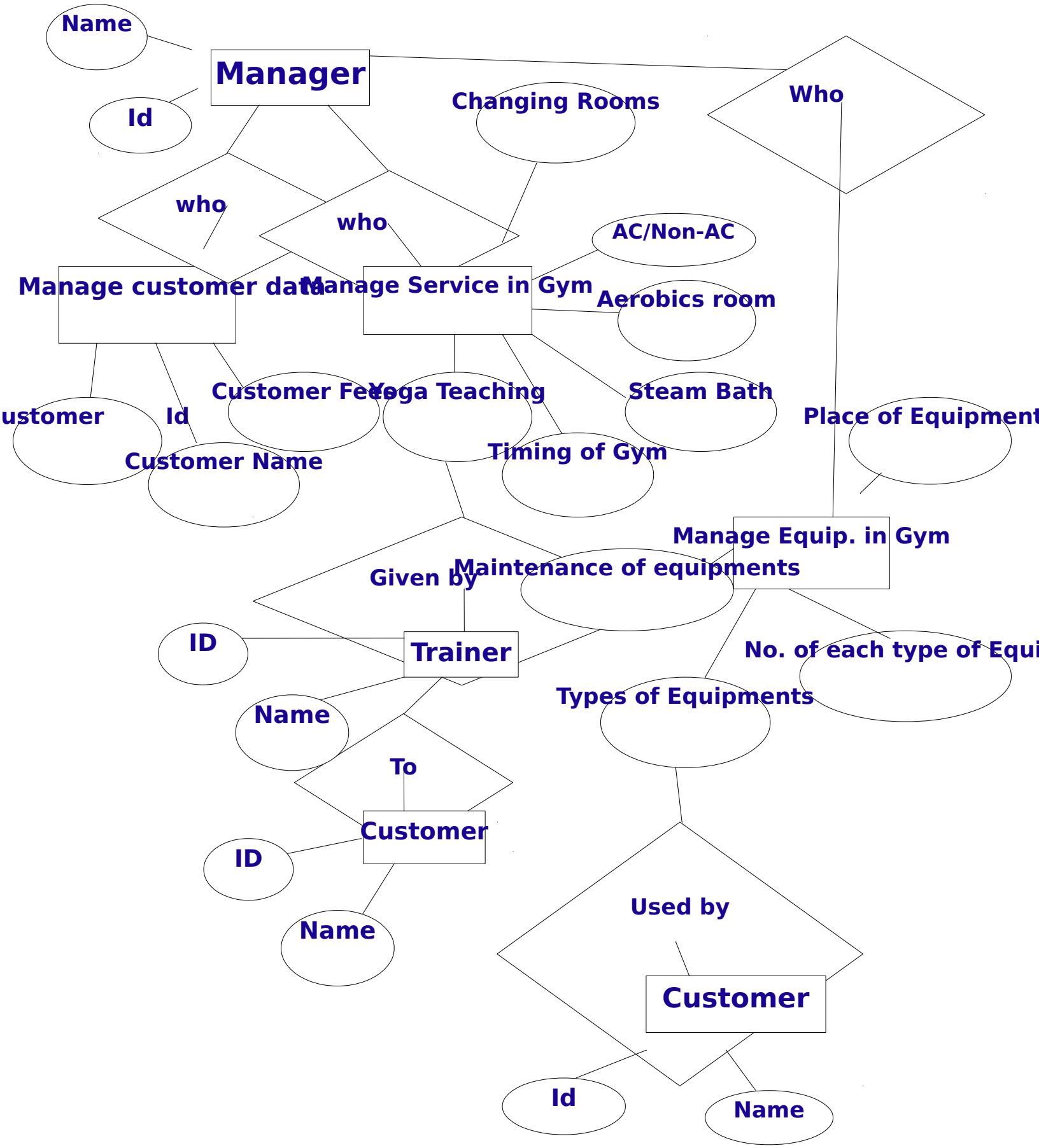
LOW LEVEL DIAGRAM - PRODUCTS



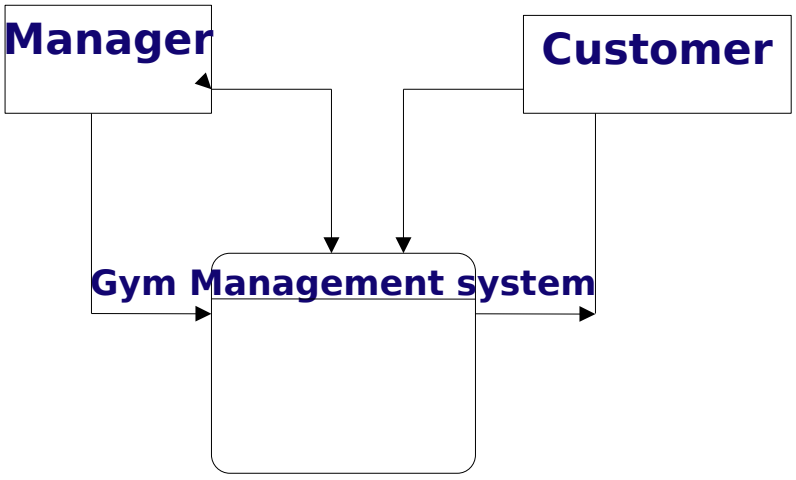
Gant Chart for Development:

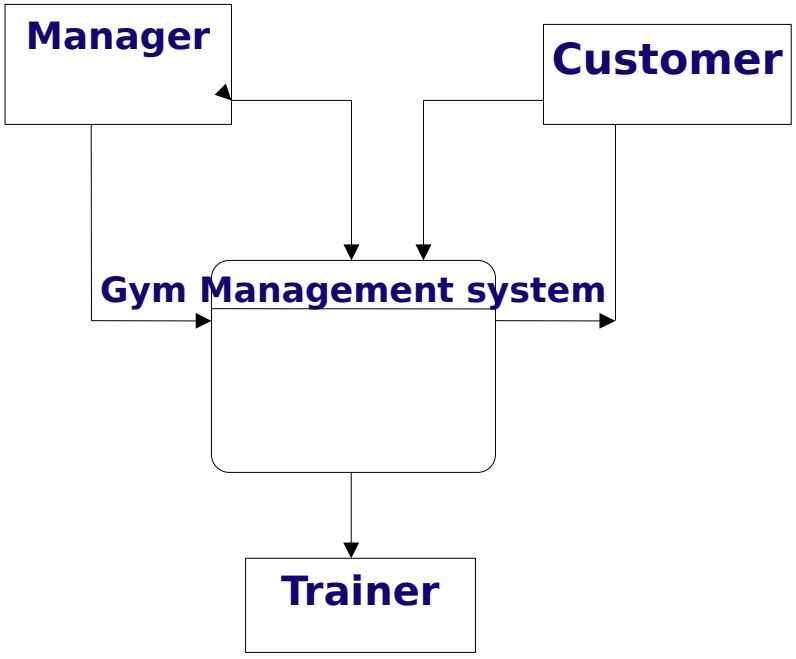


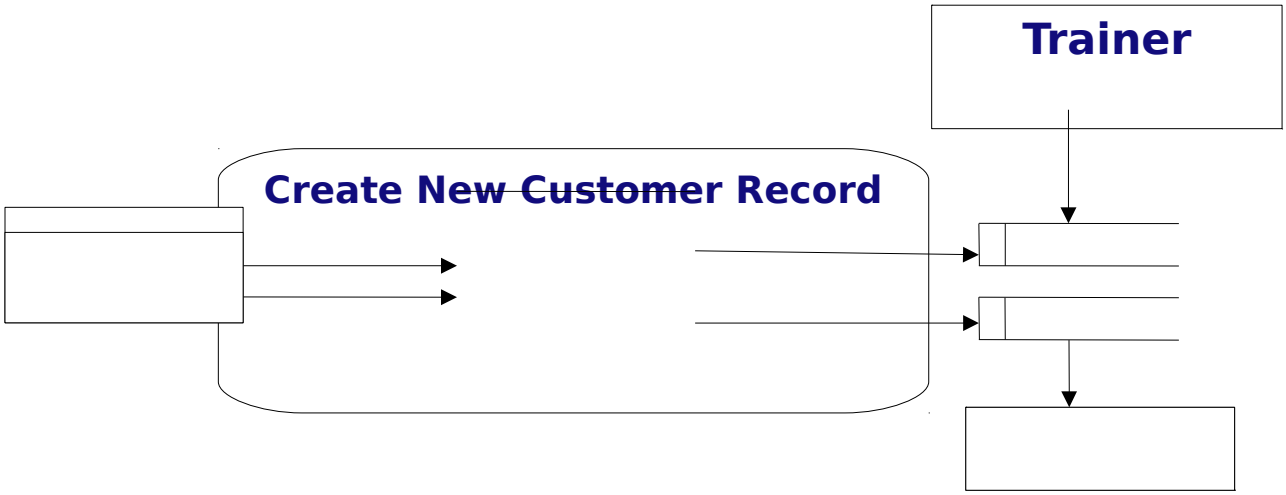
ER Diagram:

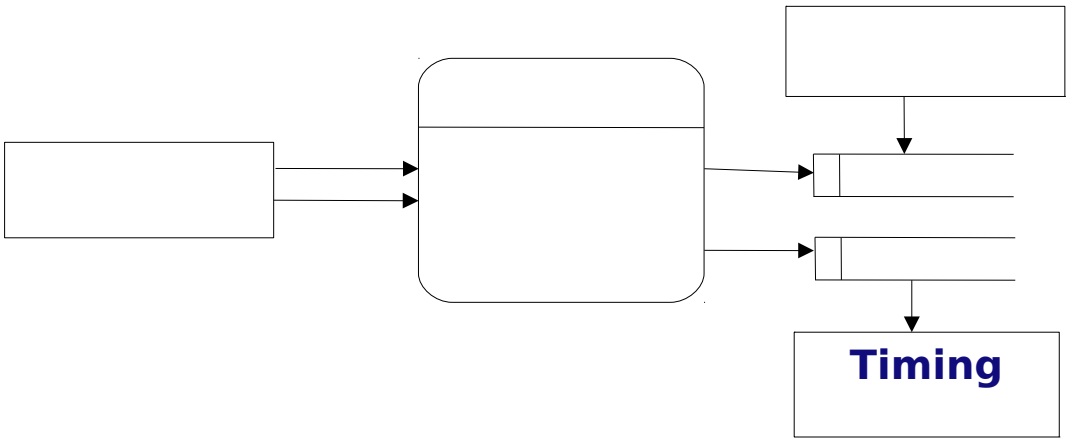

Name
Manager
Id
Changing Rooms
Who
who
who
AC/Non-AC
Manage customer data
Manage Service in Gym
Aerobics room
ustomer
Id
Customer Fees
Yoga Teaching
Steam Bath
Place of Equipment
Customer Name
Timing of Gym
Manage Equip. in Gym
Given by
Maintenance of equipments
ID
Trainer
No. of each type of Equi
Types of Equipments
Name
To
Customer
ID
Name
Used by
Customer
Id
Name

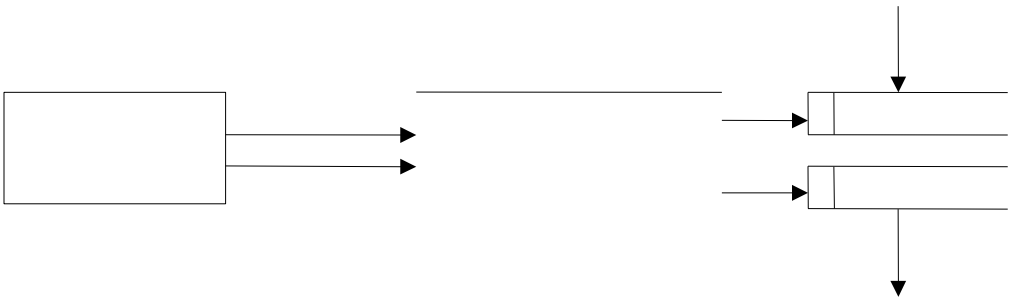
Dataflow Diagrams of the Gym Management System:

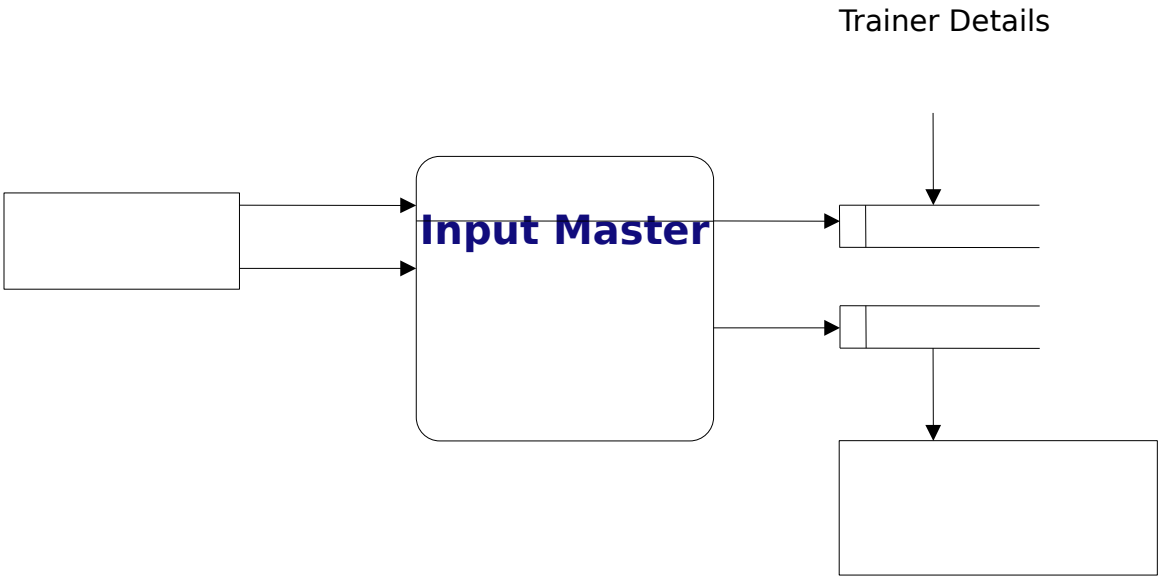


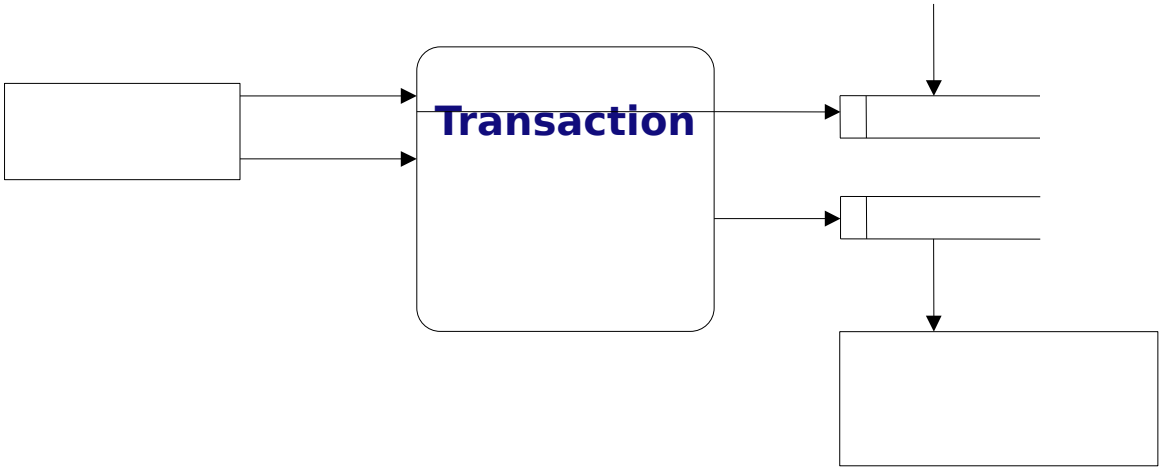



Trainer
Create New Customer Record

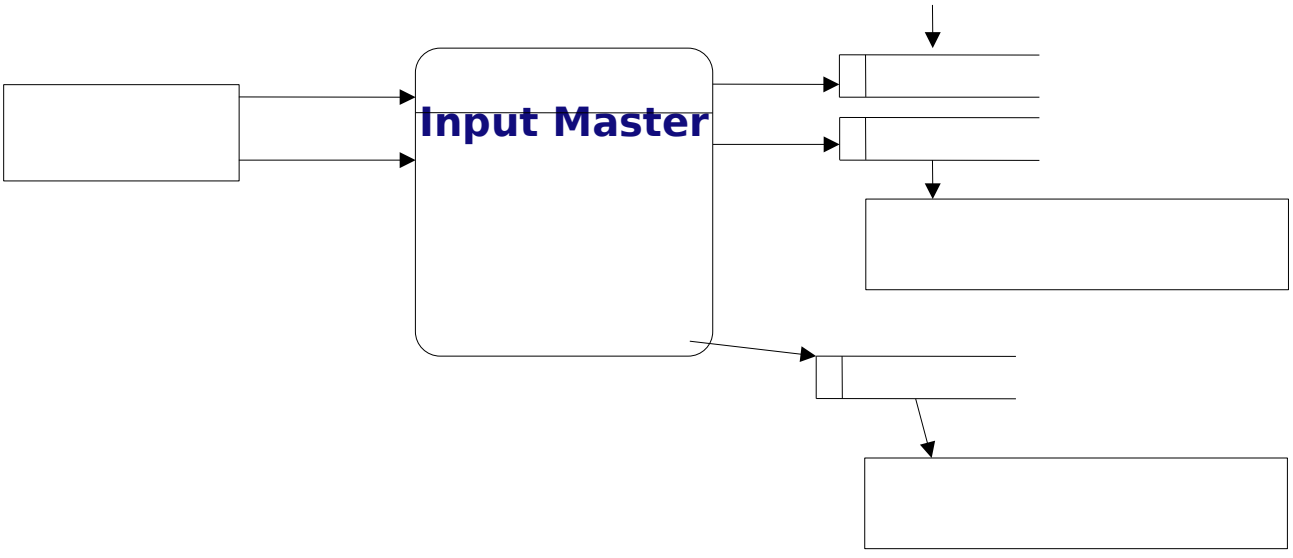

Timing


Absence/present


Trainer Details
Input Master

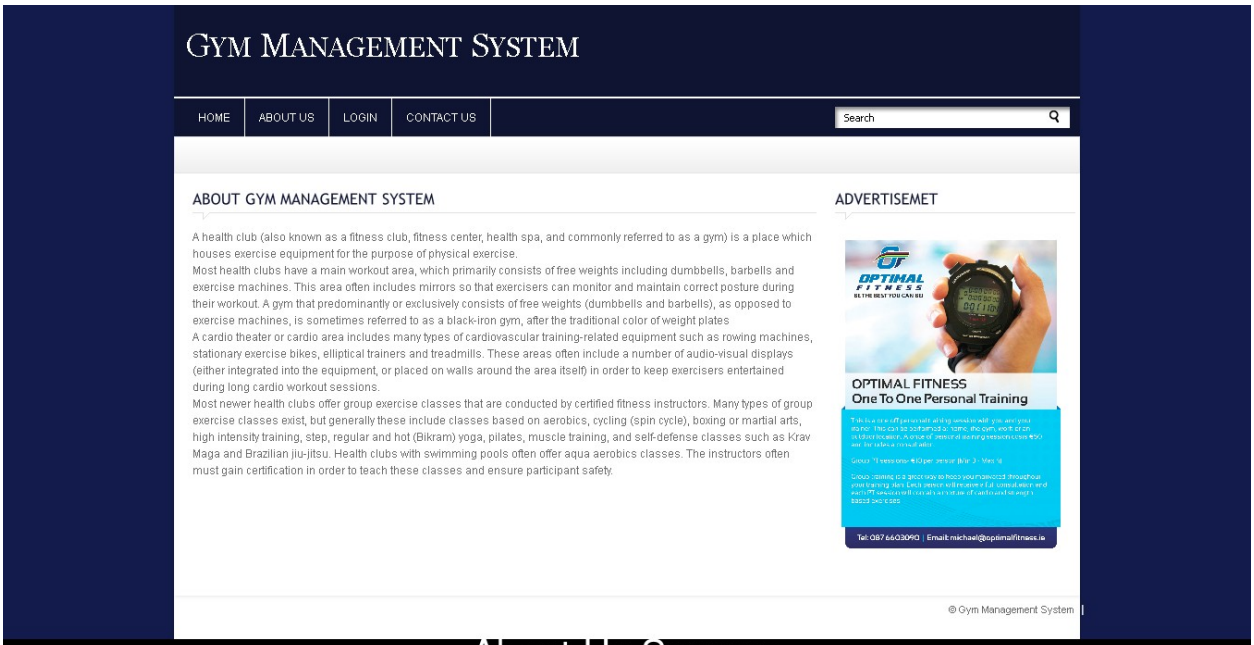


Trainer Details

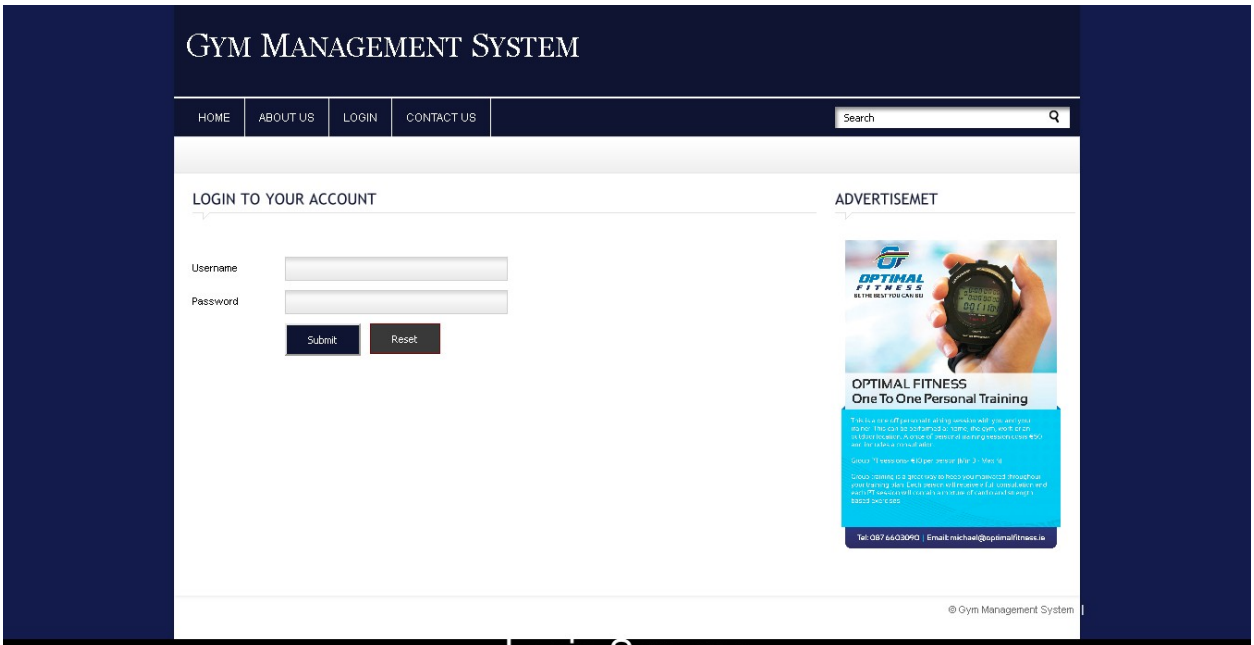


Home Page Screens

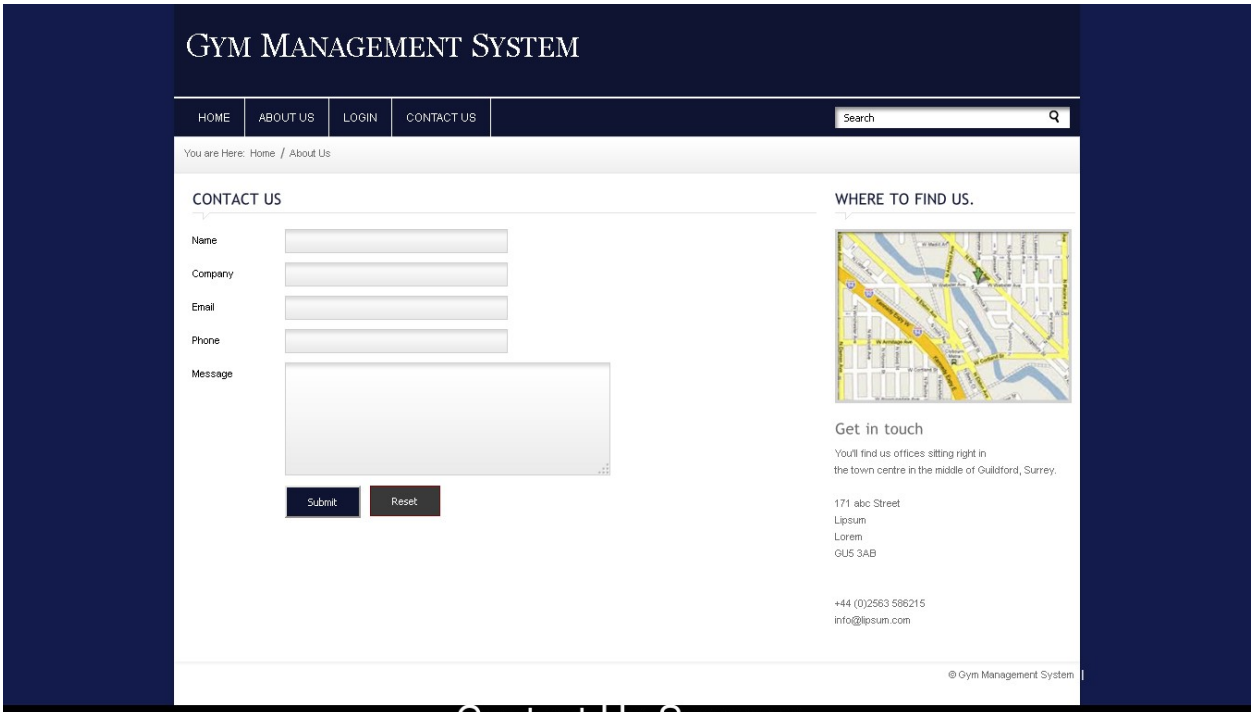
Slider Page Screens



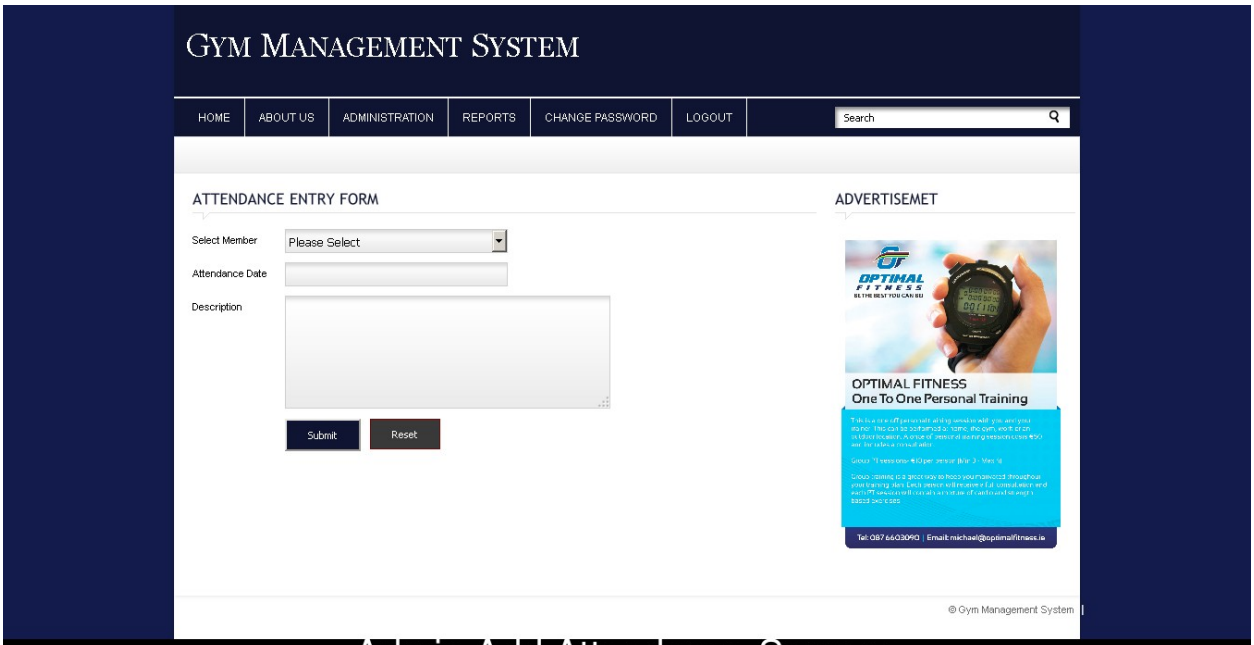
About Us Screens



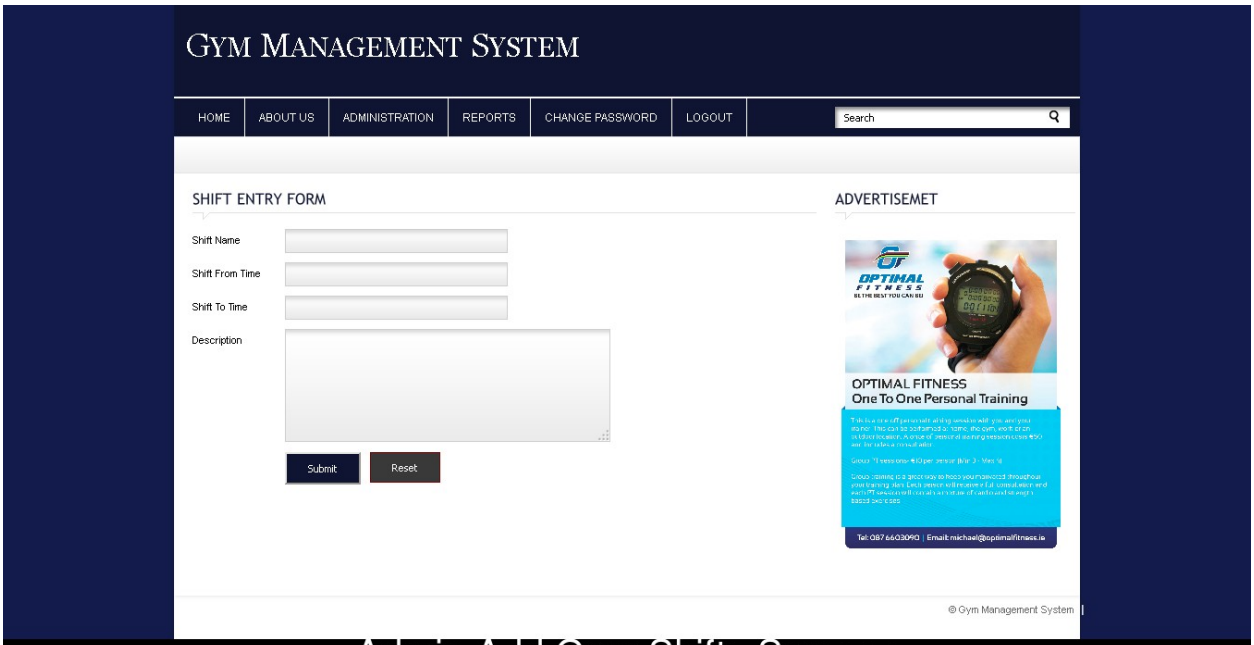
Login Screens



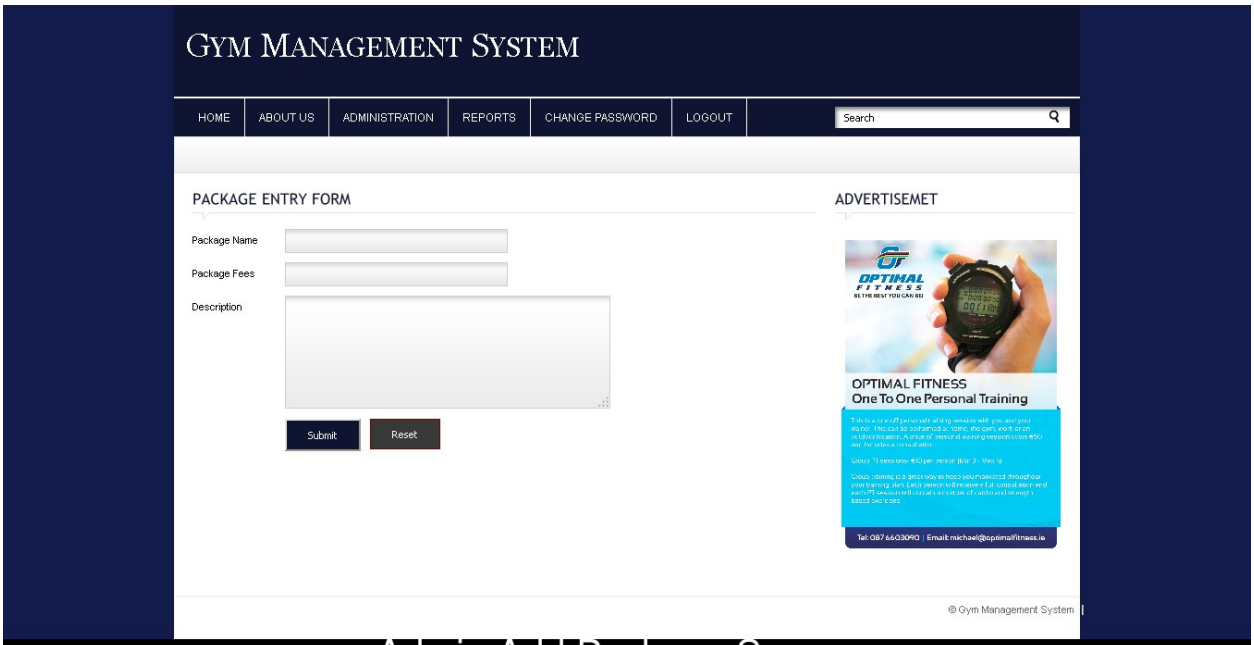
Contact Us Screens



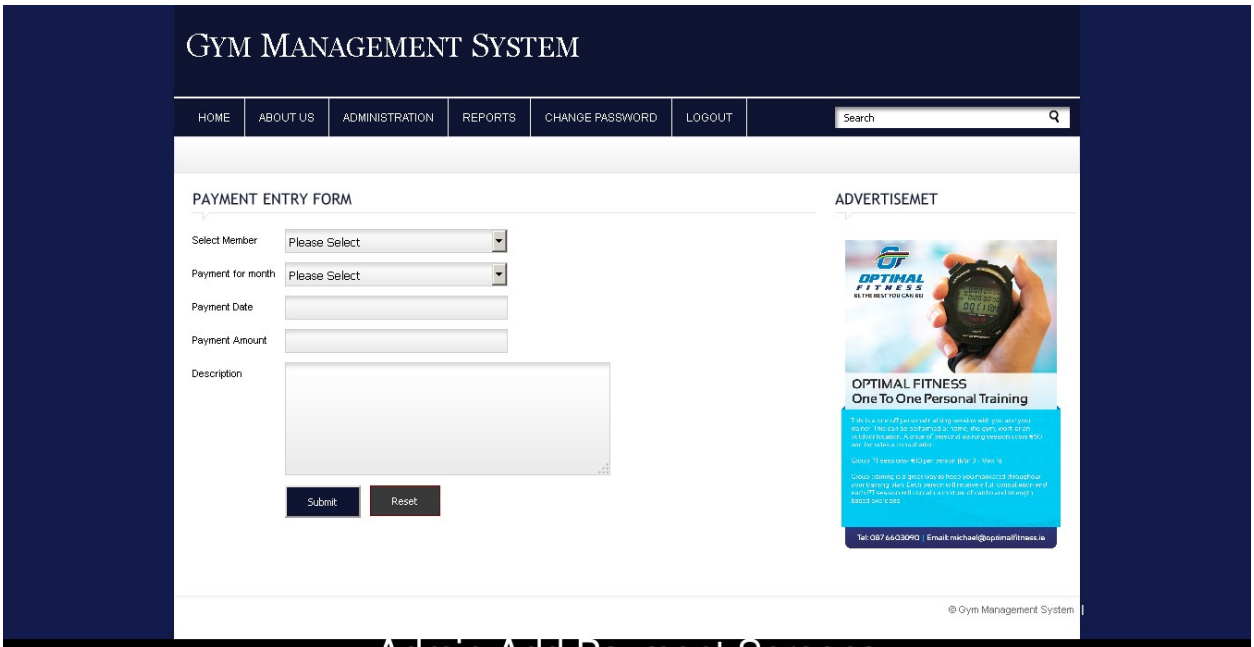
Admin Add Attendance Screens



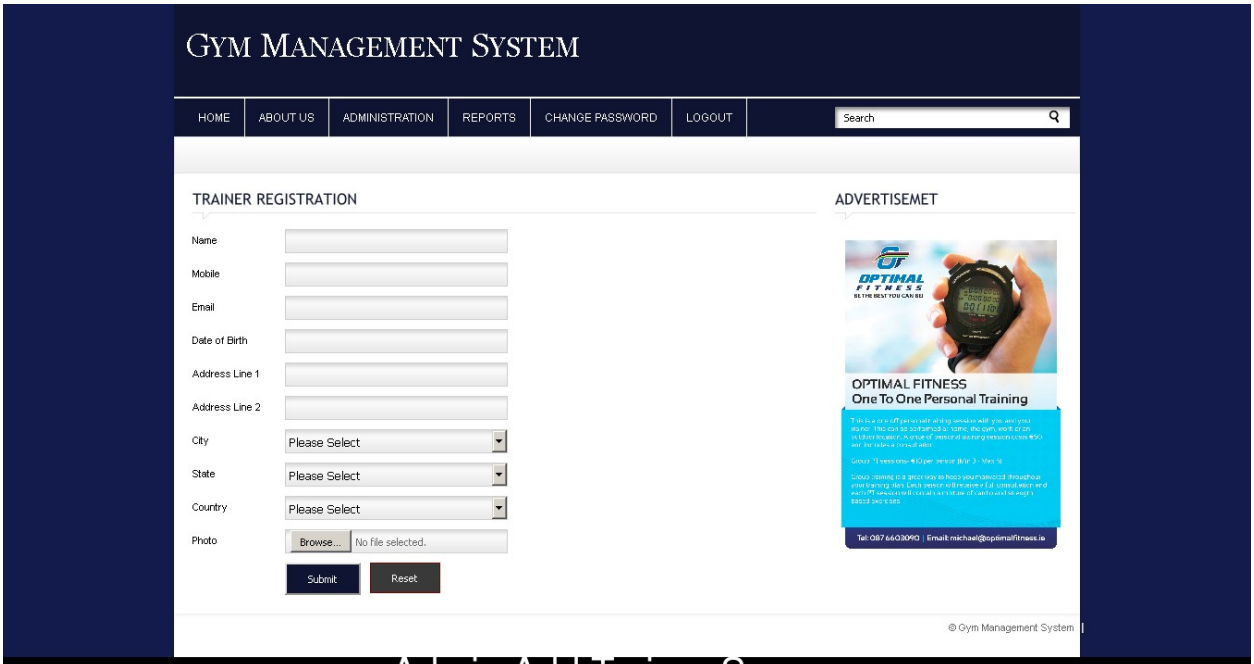
Admin Add Gym Shifts Screens



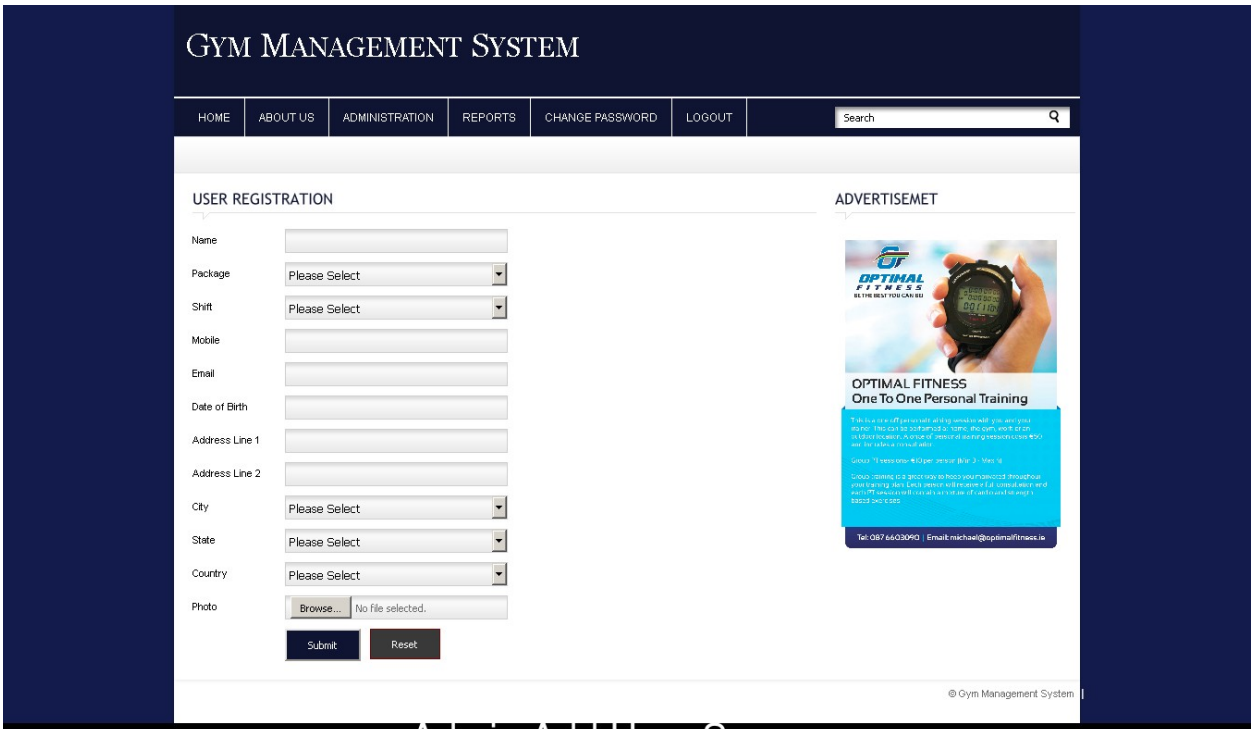
Admin Add Package Screens



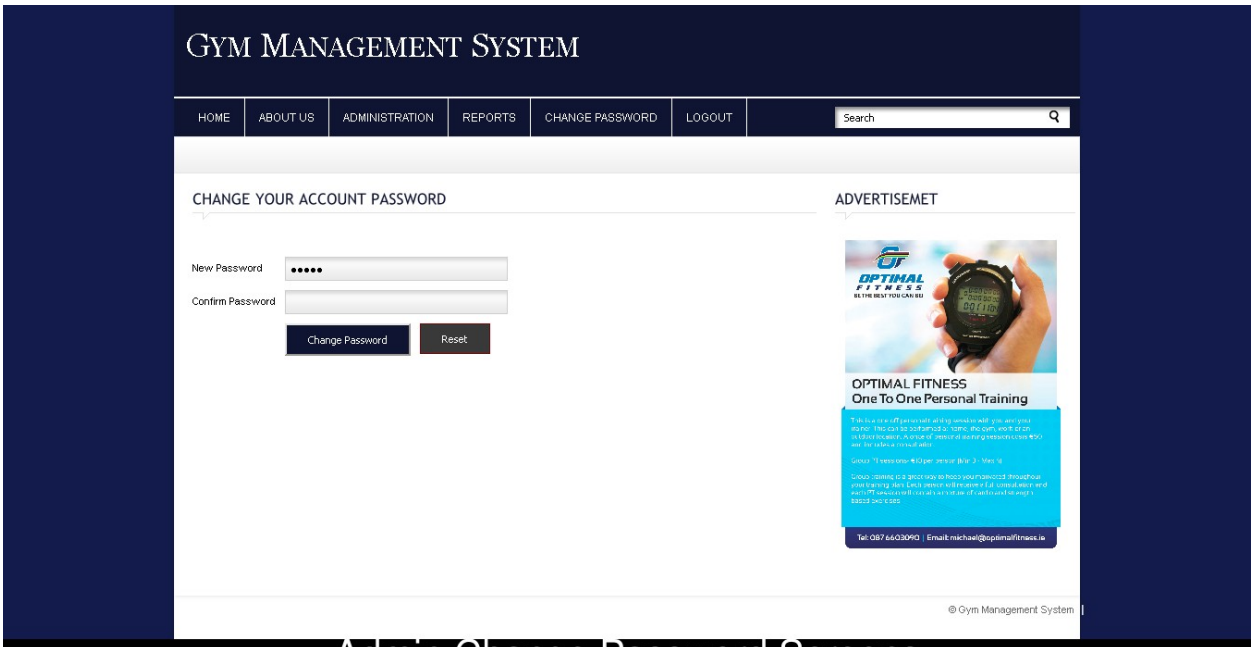
Admin Add Payment Screens



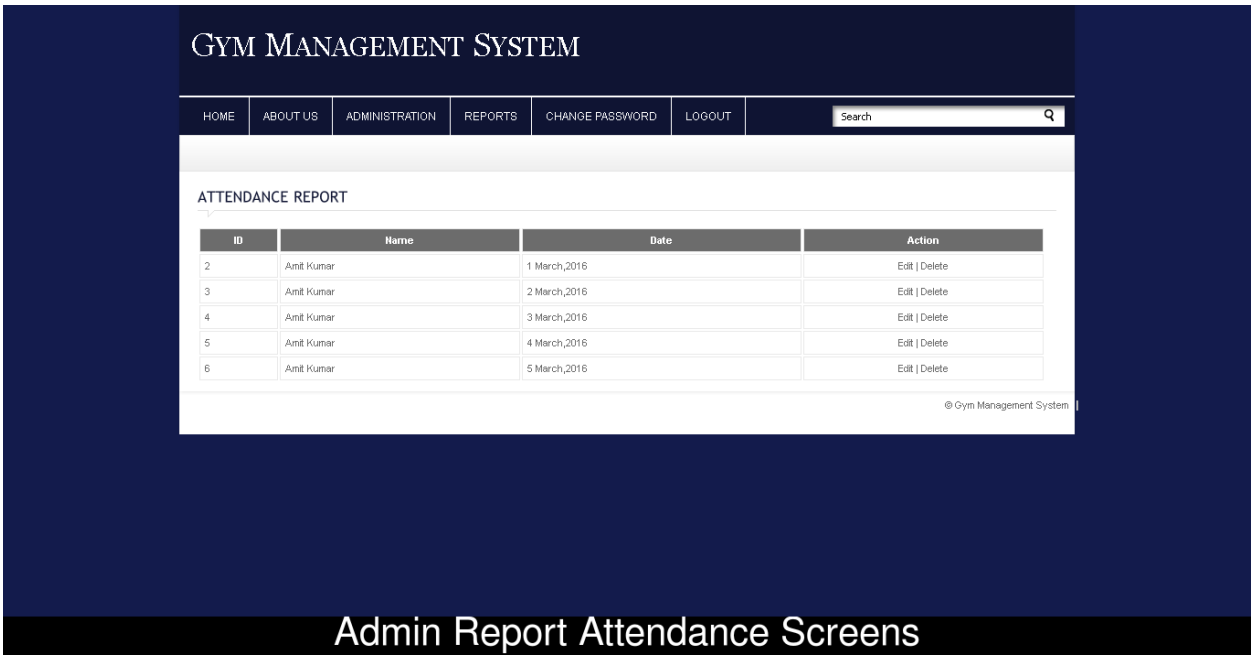
Admin Add Trainer Screens



Admin Add User Screens

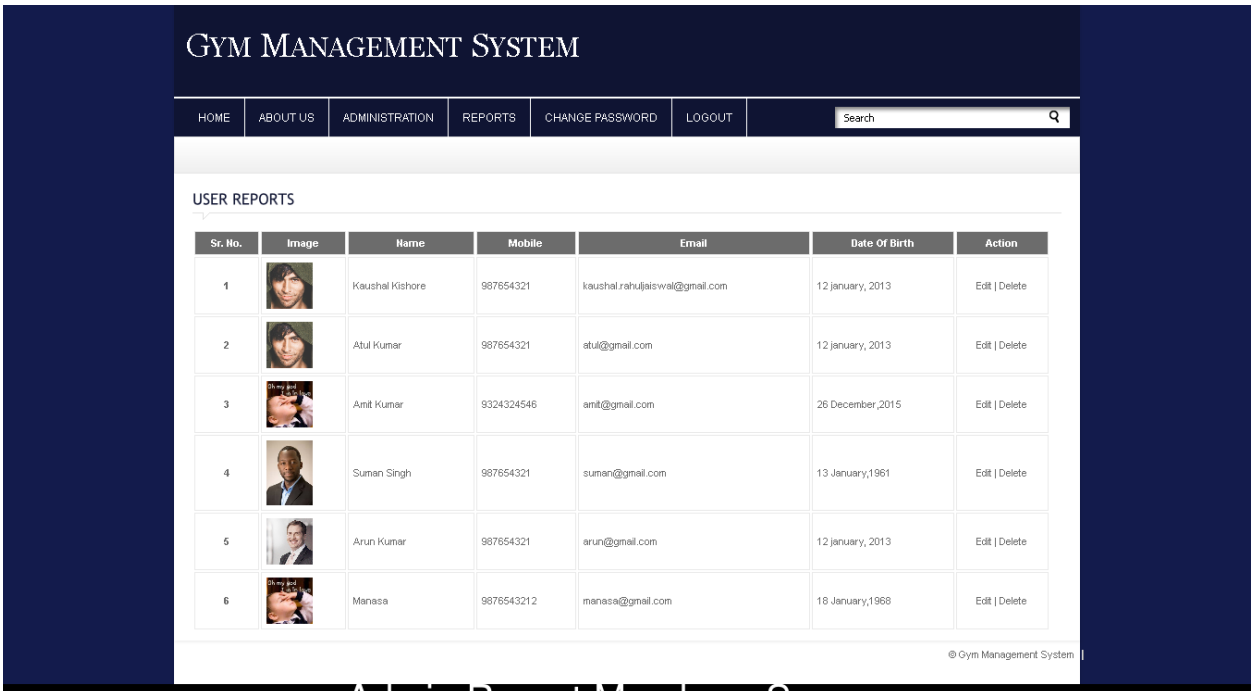


Admin Change Password Screens

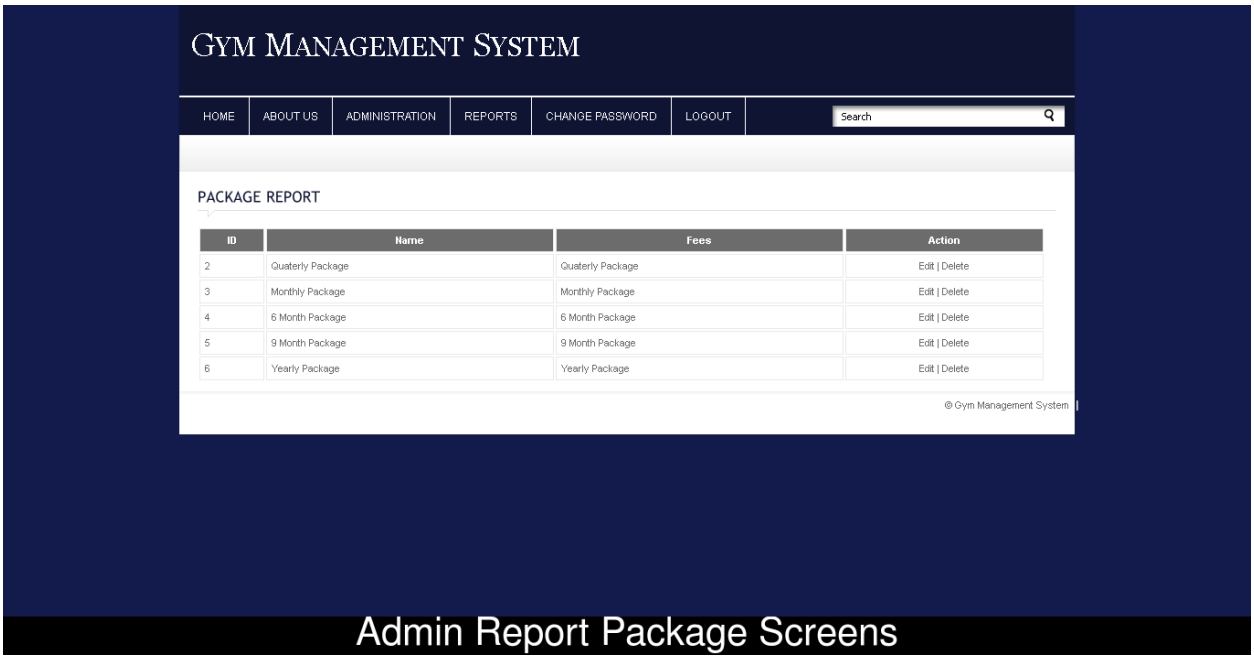


Admin Report Attendance Screens

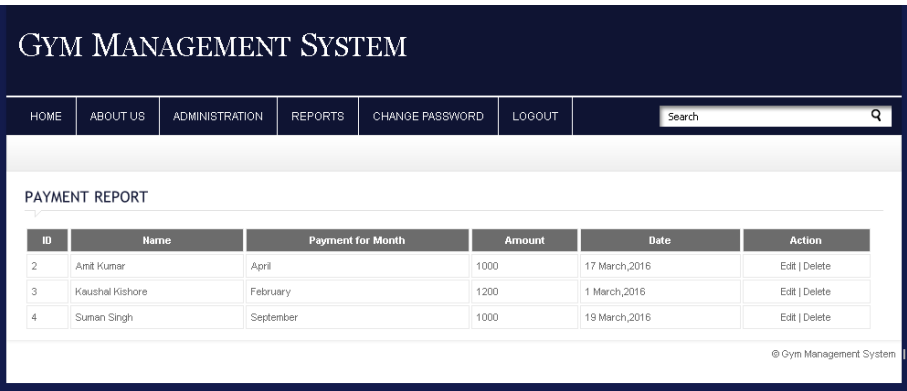
Admin Report Gym Shifts Screens



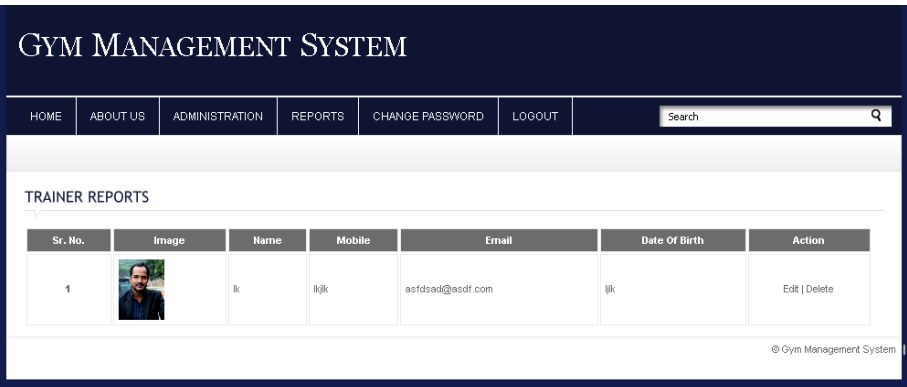
Admin Report Members Screens



Admin Report Package Screens



Admin Report Payment Screens



Admin Report Trainers Screens

Code of the page About.php:

```
<?php
include_once("includes/header.php");
if($_REQUEST[car_id])
{
$SQL="SELECT * FROM car WHERE car_id = $_REQUEST[car_id]";
$rs=mysql_query($SQL) or die(mysql_error());
$data=mysql_fetch_assoc($rs);
}
?>
<div class="crumb">
   </div>
   <div class="clear"></div>
<div id="content_sec">
<div class="col1">
        <div class="contact" style="font-size:14px;">
<h4 class="heading colr">About Gym Management System</h4>
<div style="font-size:12px;">
        <p>
        A health club (also known as a fitness club, fitness center, health spa, and commonly referred to as a gym) is a place which houses exercise equipment for the purpose of physical exercise.
        </p>
        <p>
        Most health clubs have a main workout area, which primarily consists of free weights including dumbbells, barbells and exercise machines. This area often includes mirrors so that exercisers can monitor and maintain correct posture during their workout. A gym that predominantly or exclusively consists of free weights
```

(dumbbells and barbells), as opposed to exercise machines, is sometimes referred to as a black-iron gym, after the traditional color of weight plates

A cardio theater or cardio area includes many types of cardiovascular training-related equipment such as rowing machines, stationary exercise bikes, elliptical trainers and treadmills. These areas often include a number of audio-visual displays (either integrated into the equipment, or placed on walls around the area itself) in order to keep exercisers entertained during long cardio workout sessions.

Most newer health clubs offer group exercise classes that are conducted by certified fitness instructors. Many types of group exercise classes exist, but generally these include classes based on aerobics, cycling (spin cycle), boxing or martial arts, high intensity training, step, regular and hot (Bikram) yoga, pilates, muscle training, and self-defense classes such as Krav Maga and Brazilian jiu-jitsu. Health clubs with swimming pools often offer aqua aerobics classes. The instructors often must gain certification in order to teach these classes and ensure participant safety.

<?php include_once("includes/sidebar.php"); ?>

<?php include_once("includes/footer.php"); ?>

Code of the Page Attandance.php

```
<?php
	include_once("../includes/db_connect.php");
	include_once("../includes/functions.php");
	if($_REQUEST[act]=="save_attendance")
	{
save_attendance();
exit;
	}
	if($_REQUEST[act]=="delete_attendance")
	{
delete_attendance();
exit;
	}

	###Code for save attendance#####
	function save_attendance()
	{
$R=$_REQUEST;
if($R[attendance_id])
{
	$statement = "UPDATE `attendance` SET";
	$cond = "WHERE `attendance_id` = '$R[attendance_id]'";
	$msg = "Data Updated Successfully.";
}
else
```

```
{
        $statement = "INSERT INTO `attendance` SET";
        $cond = "";
        $msg="Data saved successfully.";
}
$SQL=   $statement."
`attendance_user_id` = '$R[attendance_user_id]',
`attendance_date` = '$R[attendance_date]',
`attendance_description` = '$R[attendance_description]'".
 $cond;
$rs = mysql_query($SQL) or die(mysql_error());
header("Location:../attendance-report.php?msg=$msg");
        }
#########Function for delete attendance##########3
function delete_attendance()
{
        /////////Delete the record//////////
        $SQL="DELETE FROM attendance WHERE attendance_id =
$_REQUEST[attendance_id]";
        mysql_query($SQL) or die(mysql_error());
        header("Location:../attendance-report.php?msg=Deleted Successfully.");
}
?>
```

Code of the Page Login.php:

```
<?php
	session_start();


	include_once("../includes/db_connect.php");
	if($_REQUEST[act]=="check_login")
	{
check_login();
	}
	if($_REQUEST[act]=="logout")
	{
logout();
	}
	if($_REQUEST[act] == "change_password")
	{
change_password();
	}
####Function check user#######
function check_login()
{
	$user_user=$_REQUEST[user_user];
	$user_password=$_REQUEST[user_password];
	$SQL="SELECT * FROM user WHERE user_username = '$user_user' AND
user_password = '$user_password'";
	$rs = mysql_query($SQL) or die(mysql_error());
	if(mysql_num_rows($rs))
	{
```

```
$_SESSION[login]=1;

$_SESSION['user_details'] = mysql_fetch_assoc($rs);

if($_SESSION['user_details']['user_level_id'] == 2) {

	$SQL="SELECT * FROM branch_type WHERE bt_hod = '".
$_SESSION['user_details']['user_id']."'";

	$rs = mysql_query($SQL) or die(mysql_error());

	$_SESSION['branch_details'] = mysql_fetch_assoc($rs);

}

header("Location:../index.php");

	}

	else

	{

header("Location:../login.php?msg=Invalid User and Password.");

	}

}

####Function logout####

function logout()

{

	$_SESSION[login]=0;

	$_SESSION['user_details'] = 0;

	header("Location:../login.php?msg=Logout Successfullly.");

}

#####Function for changing the password ####

function change_password() {

	$R = $_REQUEST;

	if($R['user_confirm_password'] != $R['user_new_password']) {
```

```
header("Location:../change-password.php?msg=Your new passsword and confirm password does not match!!!");

exit;

	}

	$SQL = "UPDATE `user` SET user_password = '$R[user_new_password]' WHERE `user_id` = ".$_SESSION['user_details']['user_id'];

	$rs = mysql_query($SQL) or die(mysql_error());

	header("Location:../change-password.php?msg=Your Password Changed Successfully !!!");

	print $SQL;

	die;

}

?>
```

Code of the Page Package.php

```
<?php
include_once("../includes/db_connect.php");
include_once("../includes/functions.php");
if($_REQUEST[act]=="save_package")
{
save_package();
exit;
}
        if($_REQUEST[act]=="delete_package")
        {
delete_package();
exit;
        }
        if($_REQUEST[act]=="update_package_status")
        {
update_package_status();
exit;
        }

        ###Code for save package#####
        function save_package()
        {
$R=$_REQUEST;
if($R[package_id])
{
```

```
        $statement = "UPDATE `package` SET";
        $cond = "WHERE `package_id` = '$R[package_id]'";
        $msg = "Data Updated Successfully.";
}
else
{
        $statement = "INSERT INTO `package` SET";
        $cond = "";
        $msg="Data saved successfully.";
}
$SQL=   $statement."
`package_title` = '$R[package_title]',
`package_fees` = '$R[package_fees]',
`package_description` = '$R[package_description]'".
 $cond;
$rs = mysql_query($SQL) or die(mysql_error());
header("Location:../package-report.php?msg=$msg");
        }
#########Function for delete package##########3
function delete_package()
{
        /////////Delete the record//////////
        $SQL="DELETE FROM package WHERE package_id =
$_REQUEST[package_id]";
        mysql_query($SQL) or die(mysql_error());
        header("Location:../package-report.php?msg=Deleted Successfully.");
}
```

?>

Code of the Page Payment.php

```
<?php
	include_once("../includes/db_connect.php");
	include_once("../includes/functions.php");
	if($_REQUEST[act]=="save_payment")
	{
save_payment();
exit;
	}
	if($_REQUEST[act]=="delete_payment")
	{
delete_payment();
exit;
	}
	if($_REQUEST[act]=="update_payment_status")
	{
update_payment_status();
exit;
	}


	###Code for save payment#####
	function save_payment()
	{
$R=$_REQUEST;
if($R[payment_id])
{
```

```
        $statement = "UPDATE `payment` SET";
        $cond = "WHERE `payment_id` = '$R[payment_id]'";
        $msg = "Data Updated Successfully.";
}
else
{
        $statement = "INSERT INTO `payment` SET";
        $cond = "";
        $msg="Data saved successfully.";
}
$SQL=   $statement."
`payment_user_id` = '$R[payment_user_id]',
`payment_for_month` = '$R[payment_for_month]',
`payment_date` = '$R[payment_date]',
`payment_amount` = '$R[payment_amount]',
`payment_description` = '$R[payment_description]'".
 $cond;
$rs = mysql_query($SQL) or die(mysql_error());
header("Location:../payment-report.php?msg=$msg");
        }
#########Function for delete payment##########3
function delete_payment()
{
        /////////Delete the record//////////
        $SQL="DELETE FROM payment WHERE payment_id =
$_REQUEST[payment_id]";
        mysql_query($SQL) or die(mysql_error());
```

```
        header("Location:../payment-report.php?msg=Deleted Successfully.");
}
?>
```

Code of the Page Shift.php

```
<?php
	include_once("../includes/db_connect.php");
	include_once("../includes/functions.php");
	if($_REQUEST[act]=="save_shift")
	{
save_shift();
exit;
	}
	if($_REQUEST[act]=="delete_shift")
	{
delete_shift();
exit;
	}
	if($_REQUEST[act]=="update_shift_status")
	{
update_shift_status();
exit;
	}


	###Code for save shift#####
	function save_shift()
	{
$R=$_REQUEST;
if($R[shift_id])
{
```

```
        $statement = "UPDATE `shift` SET";
        $cond = "WHERE `shift_id` = '$R[shift_id]'";
        $msg = "Data Updated Successfully.";
}
else
{
        $statement = "INSERT INTO `shift` SET";
        $cond = "";
        $msg="Data saved successfully.";
}
$SQL=   $statement."
`shift_title` = '$R[shift_title]',
`shift_from_time` = '$R[shift_from_time]',
`shift_to_time` = '$R[shift_to_time]',
`shift_description` = '$R[shift_description]'".
 $cond;
$rs = mysql_query($SQL) or die(mysql_error());
header("Location:../shift-report.php?msg=$msg");
        }
#########Function for delete shift##########3
function delete_shift()
{
        /////////Delete the record//////////
        $SQL="DELETE FROM shift WHERE shift_id = $_REQUEST[shift_id]";
        mysql_query($SQL) or die(mysql_error());
        header("Location:../shift-report.php?msg=Deleted Successfully.");
```

```
}
?>
```

Code of the page trainer.php

```
<?php
	session_start();
	include_once("../includes/db_connect.php");
	include_once("../includes/functions.php");
	if($_REQUEST[act]=="save_trainer")
	{
save_trainer();
exit;
	}
	if($_REQUEST[act]=="delete_trainer")
	{
delete_trainer();
exit;
	}
	if($_REQUEST[act]=="get_report")
	{
get_report();
exit;
	}
	###Code for save trainer#####
	function save_trainer()
	{
$R=$_REQUEST;
//////////////////////////////////////
$image_name = $_FILES[trainer_image][name];
```

```
$location = $_FILES[trainer_image][tmp_name];
if($image_name!="")
{
	move_uploaded_file($location,"../uploads/".$image_name);
}
else
{
	$image_name = $R[avail_image];
}
//die;
if($R[trainer_id])
{
	$statement = "UPDATE `trainer` SET";
	$cond = "WHERE `trainer_id` = '$R[trainer_id]'";
	$msg = "Data Updated Successfully.";
	$condQuery = "";
}
else
{
	$statement = "INSERT INTO `trainer` SET";
	$cond = "";
	$msg="Data saved successfully.";
}
$SQL=   $statement."
`trainer_name` = '$R[trainer_name]',
`trainer_add1` = '$R[trainer_add1]',
```

```
`trainer_add2` = '$R[trainer_add2]',

`trainer_city` = '$R[trainer_city]',

`trainer_state` = '$R[trainer_state]',

`trainer_country` = '$R[trainer_country]',

`trainer_email` = '$R[trainer_email]',

`trainer_mobile` = '$R[trainer_mobile]',

`trainer_gender` = '$R[trainer_gender]',

`trainer_dob` = '$R[trainer_dob]',

`trainer_image` = '$image_name'".

 $cond;

$rs = mysql_query($SQL) or die(mysql_error());


//// Creating Trainer Leaves /////

if($R[trainer_id] == "") {

        $id = mysql_insert_id();

}


if($_SESSION['login']!=1)

{

        header("Location:../login.php?msg=You are registered successfully. Login with
your credential !!!");

        exit;

}

else if($_SESSION['trainer_details']['trainer_level_id'] == 3) {

        header("Location:../trainer.php?trainer_id=".$_SESSION['trainer_details']
['trainer_id']."&msg=Your account updated successfully !!!");

        exit;
```

```
}
header("Location:../trainer-report.php?msg=$msg");
	}
#########Function for delete trainer##########3
function delete_trainer()
{
	$SQL="SELECT * FROM trainer WHERE trainer_id = $_REQUEST[trainer_id]";
	$rs=mysql_query($SQL);
	$data=mysql_fetch_assoc($rs);

	/////////Delete the record//////////
	$SQL="DELETE FROM trainer WHERE trainer_id = $_REQUEST[trainer_id]";
	mysql_query($SQL) or die(mysql_error());

	//////////Delete the image//////////
	if($data[trainer_image])
	{
unlink("../uploads/".$data[trainer_image]);
	}
	header("Location:../trainer-report.php?msg=Deleted Successfully.");
}
?>
```

Code of the page User.php:

```
<?php
	session_start();
```

```
	include_once("../includes/db_connect.php");
	include_once("../includes/functions.php");
	if($_REQUEST[act]=="save_user")
	{
save_user();
exit;
	}
	if($_REQUEST[act]=="delete_user")
	{
delete_user();
exit;
	}
	if($_REQUEST[act]=="get_report")
	{
get_report();
exit;
	}
	###Code for save user#####
	function save_user()
	{
$R=$_REQUEST;
///Checking Username Exits or not ////
/*SQL="SELECT * FROM user WHERE user_username =
$_REQUEST[user_username]";
$rs=mysql_query($SQL);
$data=mysql_fetch_assoc($rs);
if($data['user_username'] && $R['user_id'] == "") {
```

```
	header("Location:../user.php?msg=Username Already Exits. Kindly choose another....");
	return;
}*/
////////////////////////////////////
$image_name = $_FILES[user_image][name];
$location = $_FILES[user_image][tmp_name];
if($image_name!="")
{
	move_uploaded_file($location,"../uploads/".$image_name);
}
else
{
	$image_name = $R[avail_image];
}
//die;
if($R[user_id])
{
	$statement = "UPDATE `user` SET";
	$cond = "WHERE `user_id` = '$R[user_id]'";
	$msg = "Data Updated Successfully.";
	$condQuery = "";
}
else
{
	$statement = "INSERT INTO `user` SET";
	$cond = "";
```

```
        $msg="Data saved successfully.";
        create_user_leaves();
}
$SQL=   $statement."
`user_level_id` = '3',
`user_name` = '$R[user_name]',
`user_shift_id` = '$R[user_shift_id]',
`user_package_id` = '$R[user_package_id]',
`user_add1` = '$R[user_add1]',
`user_add2` = '$R[user_add2]',
`user_city` = '$R[user_city]',
`user_state` = '$R[user_state]',
`user_country` = '$R[user_country]',
`user_email` = '$R[user_email]',
`user_mobile` = '$R[user_mobile]',
`user_gender` = '$R[user_gender]',
`user_dob` = '$R[user_dob]',
`user_image` = '$image_name'".
 $cond;
$rs = mysql_query($SQL) or die(mysql_error());

//// Creating User Leaves /////
if($R[user_id] == "") {
        $id = mysql_insert_id();
}
```

```
if($_SESSION['login']!=1)
{
	header("Location:../login.php?msg=You are registered successfully. Login with your credential !!!");
	exit;
}
else if($_SESSION['user_details']['user_level_id'] == 3) {
	header("Location:../user.php?user_id=".$_SESSION['user_details']['user_id']."&msg=Your account updated successfully !!!");
	exit;
}
header("Location:../user-report.php?msg=$msg");
	}
#########Function for delete user##########3
function delete_user()
{
	$SQL="SELECT * FROM user WHERE user_id = $_REQUEST[user_id]";
	$rs=mysql_query($SQL);
	$data=mysql_fetch_assoc($rs);

	/////////Delete the record//////////
	$SQL="DELETE FROM user WHERE user_id = $_REQUEST[user_id]";
	mysql_query($SQL) or die(mysql_error());

	//////////Delete the image//////////
	if($data[user_image])
	{
```

```
unlink("../uploads/".$data[user_image]);
        }
        header("Location:../user-report.php?msg=Deleted Successfully.");
}
?>
```

Code of the page Package Report.php

```
<?php
	include_once("includes/header.php");
	include_once("includes/db_connect.php");
	$SQL="SELECT * FROM `package`";
	$rs=mysql_query($SQL) or die(mysql_error());
?>
<script>
function delete_package(package_id)
{
	if(confirm("Do you want to delete the package?"))
	{
this.document.frm_package.package_id.value=package_id;
this.document.frm_package.act.value="delete_package";
this.document.frm_package.submit();
	}
}
</script>
	<div class="crumb">
   </div>
   <div class="clear"></div>
	<div id="content_sec">
<div class="col1" style="width:100%">
<div class="contact">
	<h4 class="heading colr">Package Report</h4>
	<?php
```

```
        if($_REQUEST['msg']) {
        ?>
<div class="msg"><?=$_REQUEST['msg']?></div>
        <?php
        }
        ?>
        <form name="frm_package" action="lib/package.php" method="post">
<div class="static">
        <table style="width:100%">
          <tbody>
          <tr class="tablehead bold">
            <td scope="col">ID</td>
<td scope="col">Name</td>
<td scope="col">Fees</td>
<td scope="col">Action</td>
          </tr>
        <?php
        $sr_no=1;
        while($data = mysql_fetch_assoc($rs))
        {
        ?>
          <tr>
<td><?=$data[package_id]?></td>
<td><?=$data[package_title]?></td>
<td><?=$data[package_title]?></td>
<td style="text-align:center">
```

```
        <a href="package.php?package_id=<?php echo $data[package_id] ?
>">Edit</a> | <a href="PHPscript:delete_package(<?=$data[package_id]?
>)">Delete</a> </td>

</td>

          </tr>

        <?php } ?>

        </tbody>

        </table>

</div>

<input type="hidden" name="act" />

<input type="hidden" name="package_id" />

        </form>

</div>

</div>

        </div>

<?php include_once("includes/footer.php"); ?>
```

Code of the page Attandance Report.php

```
<?php
	include_once("includes/header.php");
	include_once("includes/db_connect.php");
	$SQL="SELECT * FROM `attendance`,`user` WHERE attendance_user_id = user_id";
	$rs=mysql_query($SQL) or die(mysql_error());
?>
<script>
function delete_attendance(attendance_id)
{
	if(confirm("Do you want to delete the attendance?"))
	{
this.document.frm_attendance.attendance_id.value=attendance_id;
this.document.frm_attendance.act.value="delete_attendance";
this.document.frm_attendance.submit();
	}
}
</script>
	<div class="crumb">
   </div>
   <div class="clear"></div>
	<div id="content_sec">
<div class="col1" style="width:100%">
<div class="contact">
	<h4 class="heading colr">Attendance Report</h4>
	<?php
```

```
        if($_REQUEST['msg']) {
        ?>
<div class="msg"><?=$_REQUEST['msg']?></div>
        <?php
        }
        ?>
        <form name="frm_attendance" action="lib/attendance.php"
method="post">
<div class="static">
        <table style="width:100%">
          <tbody>
          <tr class="tablehead bold">
            <td scope="col">ID</td>
<td scope="col">Name</td>
<td scope="col">Date</td>
<td scope="col">Action</td>
          </tr>
        <?php
        $sr_no=1;
        while($data = mysql_fetch_assoc($rs))
        {
        ?>
          <tr>
<td><?=$data[attendance_id]?></td>
<td><?=$data[user_name]?></td>
<td><?=$data[attendance_date]?></td>
<td style="text-align:center">
```

```
        <a href="attendance.php?attendance_id=<?php echo
$data[attendance_id] ?>">Edit</a> | <a href="PHPscript:delete_attendance(<?
=$data[attendance_id]?>)">Delete</a> </td>

</td>

          </tr>

        <?php } ?>

        </tbody>

        </table>

</div>

<input type="hidden" name="act" />

<input type="hidden" name="attendance_id" />

        </form>

</div>

</div>

        </div>

<?php include_once("includes/footer.php"); ?>
```

Code of the page change Password.php

```
<?php include_once("includes/header.php"); ?>

        <div class="crumb">

    </div>

    <div class="clear"></div>

        <div id="content_sec">

<div class="col1">

        <div class="contact">

        <h4 class="heading colr">Change Your Account Password</h4>

        <div class='msg'><?=$_REQUEST['msg']?></div>

        <form action="lib/login.php" method="post" name="frm_car">

<ul class="forms">

        <li class="txt">New Password</li>

        <li class="inputfield"><input name="user_new_password" type="password"
class="bar" required /></li>

</ul>

<ul class="forms">

        <li class="txt">Confirm Password</li>

        <li class="inputfield"><input name="user_confirm_password"
type="password" class="bar" required /></li>

</ul>

<div class="clear"></div>

<ul class="forms">

        <li class="txt"> </li>

        <li class="textfield"><input type="submit" value="Change Password"
class="simplebtn"></li>

        <li class="textfield"><input type="reset" value="Reset"
class="resetbtn"></li>
```

```
</ul>
<input type="hidden" name="act" value="change_password">
        </form>
        </div>
</div>
<div class="col2">
        <?php include_once("includes/sidebar.php"); ?>
</div>
        </div>
<?php include_once("includes/footer.php"); ?>
```

Code of the page Payment-report.php

```
<?php
	include_once("includes/header.php");
	include_once("includes/db_connect.php");
	$SQL="SELECT * FROM `payment`,`user`,`month` WHERE
payment_for_month = month_id AND payment_user_id = user_id";
	$rs=mysql_query($SQL) or die(mysql_error());
?>
<script>
function delete_payment(payment_id)
{
	if(confirm("Do you want to delete the payment?"))
	{
this.document.frm_payment.payment_id.value=payment_id;
this.document.frm_payment.act.value="delete_payment";
this.document.frm_payment.submit();
	}
}
</script>
	<div class="crumb">
   </div>
   <div class="clear"></div>
	<div id="content_sec">
<div class="col1" style="width:100%">
<div class="contact">
	<h4 class="heading colr">Payment Report</h4>
	<?php
```

```
        if($_REQUEST['msg']) {
        ?>
<div class="msg"><?=$_REQUEST['msg']?></div>
        <?php
        }
        ?>
        <form name="frm_payment" action="lib/payment.php" method="post">
<div class="static">
        <table style="width:100%">
          <tbody>
          <tr class="tablehead bold">
            <td scope="col">ID</td>
<td scope="col">Name</td>
<td scope="col">Payment for Month</td>
<td scope="col">Amount</td>
<td scope="col">Date</td>
<td scope="col">Action</td>
          </tr>
        <?php
        $sr_no=1;
        while($data = mysql_fetch_assoc($rs))
        {
        ?>
          <tr>
<td><?=$data[payment_id]?></td>
<td><?=$data[user_name]?></td>
```

```
<td><?=$data[month_name]?></td>

<td><?=$data[payment_amount]?></td>

<td><?=$data[payment_date]?></td>

<td style="text-align:center">

        <a href="payment.php?payment_id=<?php echo $data[payment_id] ?
>">Edit</a> | <a href="PHPscript:delete_payment(<?=$data[payment_id]?
>)">Delete</a> </td>

</td>

          </tr>

        <?php } ?>

        </tbody>

        </table>

</div>

<input type="hidden" name="act" />

<input type="hidden" name="payment_id" />

        </form>

</div>

</div>

        </div>

<?php include_once("includes/footer.php"); ?>
```

Identification of Need

The old manual system was suffering from a series of drawbacks. Since whole of the system was to be maintained with hands the process of keeping, maintaining and retrieving the information was very tedious and lengthy. The records were never used to be in a systematic order. there used to be lots of difficulties in associating any particular transaction with a particular context. If any information was to be found it was required to go through the different registers, documents there would never exist anything like report generation. There would always be unnecessary consumption of time while entering records and retrieving records. One more problem was that it was very difficult to find errors while entering the records. Once the records were entered it was very difficult to update these records. In present, work done in the electricity board is performed manually which is a great headache for the department .The reason behind it is that there is lot of information to be maintained and have to be kept in mind while running the business .For this reason we have provided features Present system is partially automated (computerized), actually existing system is quite laborious as one has to enter same information at three different places.

Following points should be well considered:

- Documents and reports that must be provided by the new system: there can also be few reports, which can help management in decision-making and cost controlling, but since these reports do not get required attention, such kind of reports and information were also identified and given required attention.
- Details of the information needed for each document and report.
- The required frequency and distribution for each document.
- Probable sources of information for each document and report.
- With the implementation of computerized system, the task of keeping records in an organized manner will be solved. The greatest of all is the retrieval

of information, which will be at the click of the mouse. So the proposed system helps in saving the time in different operations and making information flow easy giving valuable reports.

DATA DICTIONARY:

This is normally represented as the data about data. It is also termed as metadata some times which gives the data about the data stored in the database. It defines each data term encountered during the analysis and design of a new system. Data elements can describe files or the processes.

Following are some major symbols used in the data dictionary

- = equivalent to
- + and
- [] either/ or
- () Optional entry

Following are some rules, which defines the construction of data dictionary entries:

1. Words should be defined to understand for what they need and not the variable need by which they may be described in the program .
2. Each word must be unique. We cannot have two definition of the same client.
3. Aliases or synonyms are allowed when two or more enters shows the same meaning. For example a vendor number may also be called as customer number.
4. A self-defining word should not be decomposed. It means that the reduction of any information in to subpart should be done only if it is really required that is it is not easy to understand directly.

Data dictionary includes information such as the number of records in file, the frequency a process will run, security factor like pass word which user must enter to get excess to the information.

References and Bibliography:

- http://www.bluedart.com/
- http://www.wampserver.com/en/
- http://www.HTML.net/
- http://www.tutorialspoint.com/mysql/
- httpd.apache.org/docs/2.0/misc/tutorials.html