PROJECT REPORT ON

GYM MANAGEMENT

Submitted by

SRIRAM R.ANNAN

PRAMEER A.KULKARNI

MARCH 2011

UNDER THE GUIDENCE OF

Mrs.SANDHYA

Mrs. SUNANDA

G.R.PATIL COLLAGE SONARPADA,

DOMBIVLI (EAST)-421203

GYM MANAGEMENT SYSTEM

G.R.PATIL COLLEGE OF ARTS, SCIENCE & COMMERCE

(Affiliated to the University of Mumbai)

Dombivli (E), Mumbai-421204

Gym Management System For Sanjay Health Club

DEVEVLOPED BY

Mr. Sriram R.Annan

Mr. Prameer A.Kulkarni

UNDER THE GUIDANCE OF

Mrs. Sunanda Mulgund

Mrs. Sandhya Pandye

PROJECT SUBMITTED FOR THE PARTIAL FULFILLMENT OF BACHLERS DEGREE OF SCIENCE IN INFORMATION TECHNOLOGY IN THE YEAR 2010-2011

UNIVERSITY OF MUMBAI

Certificate

This is to certify that Mr. SRIRAM R. ANNAN &

Mr. PRAMEER A. KULKARNI

Has satisfactorily completed the project work entitled

“GYM MANAGEMENT”

And

Prepared this project during the academic year 2010-2011

In partial fulfillment for the award of B.Sc IT

Recognized by university of Mumbai

It is further certified that they completed all required phases of the project.

Internal Examiner	External Examiner
Principal	B.sc (it) Coordinator

DECLARATION

We “Mr. Sriram R. Annan & Mr. Prameer A. Kulkarni” of G.R.Patil College Dombivli (e), students of T.Y.B.Sc (IT) (semester VI) hereby declare that we have completed this project on ‘GYM MANAGEMENT’ in the academic year 2010-2011. This information submitted is true original to the best of our knowledge.

Signatures of students

(Sriram R. Annan)

(Prameer A. Kulkarni)

ACKNOWLEDGEMENT

We would like to acknowledge our debt to each & every person associated in this Project Development. The Project Development required huge Commitment from all the individuals involved in it.

We are also indebted to Mrs. Sandhya madam who has guided us throughout the Project Development. We are Thankful for the patience with which she stood by us till the end of our Project. We are very Thankful for her Bounteousness for standing by us in peak movements of the Project Development.

We would also like to acknowledge all the staffs for providing a helping hand to us in times of queries & problems. The Project is a result of the efforts of all the peoples who are associated with the Project directly or indirectly, who helped us to Successfully complete the Project within the specified Time Frame.

We are also very Thankful to Mrs. Sunanda who helped us in the Development of the Project by lending her valuable Support to us.

We would also like to Thanks all the Professors who helped us in developing the Project. Without their

Courage & Support, the Project Development would have been Futile. It was only their building Support & Morale us in attaining the Successful completion of the Project.

We would like to Thanks our colleagues for keeping our Sprits High while preparing the Project. Because of their Diligent & Hard Work, we wouldn't have been able to complete the Project within the given Time Frame.

We are Thankful to each & every people involved with us in this case study, their Encouragement & Support enabled the Project to Materialize & Contributed it to its success. We would like to express our Appreciation to all the people who have contributed to the Successful completion of the Project.

With all Respects & Gratitude, we would like to Thanks to all the people, who have helped in the Development of the Project

By,

Sriram R. Annan

Prameer A. Kulkarni

MAIN REPORT

OF

GYM MANAGEMENT

ANNEXURE

OF

GYM MANAGEMENT

INDEX

<u>INTRODUCTION</u>

1.1 Organization Overview

The Gym Management requires a system that will handle all the necessary and minute details easily and proper database security accordingly to the user. They requires software, which will store data about members, employees, products, payroll, receipts of members etc & all transactions that occur in Gym and lock-up with graphical user interface(GUI).

1.2 Objective of the Project

- The main objective of the project is to design and develop a user friendly system.
- Easy to use and efficient computerized system.
- To develop an accurate and flexible system, it will eliminate data redundancy.
- Computerization can be helpful as means of saving time & money.
- To provide better graphical user interface.
- Less chances of information leakage.
- Provides security to data by using login & password.

1.3 Scope of the Project

- Storing information of members, employees.
- Check validity of information provided by user.
- Storing information of members according to their id.
- Generating reports for different id.

THEORETICAL BACKGROUND

2.1 Introduction to Project

We have done a project on Gym Management and database management and transactions. This system is proposed to be an automate database management & transactions. This stores employee, member, payroll, receipts, and products information. It also provides the facility of search & advanced search for searching the records efficiently & immediately. This system provides data storing & report generation with graphical user interface (GUI).

2.2 System Study

It is always necessary to study and recognize the problems of existing system, which will help in finding out the requirements for the new system. System study helps in finding different alternatives for better solution.

The project study basically deals with different operations and steps involved in generation of examination mark sheets. Ti includes:

1. Data gathering
2. Study of existing system
3. Analyzing problem
4. Studying various documents
5. Feasibility study for further improvements

Following are the steps taken during the initial study:
Initially, we collected all the information, which they wanted to store. Then we studied the working of the current system which is done manually. We noted the limitation of that system which motivated them to have new system.
With the help of these documents we got basic ideas about the system as well as input output of the developed system.

The most important thing is to study system thoroughly.
Here we are studying both existing system and proposed system so that advantages & disadvantages of both the systems can be understood

The first task was identifying how system can be computerized. Some analysis and projections was done regarding changes to be made to the existing system.
The new developed system for **Gym Management** is simple without complexities.

2.3 Existing System

The gym is working manually. The current system is time consuming and also it is very costly, because it involves a lot of paperwork. To manually handle the system was very difficult task. But now-a-days computerization made easy to work.

The following are the reasons why the current system should be computerized:

- To increase efficiency with reduced cost.
- To reduce the burden of paper work.
- To save time management for recording details of each and every member and employee.
- To generate required reports easily.

Limitations of existing system:

- Time consumption:
  As the records are to be manually maintained it consumes a lot of time.

- Paper work:
  Lot of paper work is involved as the records are maintained in the files & registers

- Storage requirements:

  As files and registers are used the storage space requirement is increased.

- Less reliable:
  Use of papers for storing valuable data information is not at all reliable.

- Accuracy:

  As the system is in manual there are lot many chances of human errors. These can cause errors in calculating mechanism or maintaining customer details.

- Difficulty in keeping new records:
  It is difficult for keeping all the new entries of members, their account and transaction details.

2.4 Proposed System

The proposed system is managed by the visual basic 6.0, which are user friendly windows for every user and for maintaining the database Microsoft access is used.

Scope of proposed system:

The system proposed has many advantages.

1. The proposed system is highly secured, because for login the system it requires the username and password which is different for each department therefore providing each department a different view of the customer information.
2. It provides wide range of certain criteria in each window the client is working for better and quicker solution.
3. It maintains report for all criteria and transactions.
4. Manages member information separately for all exercise and employee information separately for considering the requirements of gym.
5. Stores information about regular products.
6. This system can run on any windows operating system.

2.5 SYSTEM ANALYSIS & DESIGN

The way that is followed while carrying on with the development application is as follows

Phase I (defining a problem)

Defining a problem is one of the important activities of the project. The objective is to define precisely the business problem to be solved & thereby determined the scope of the new system. This phase consist of 2 main tasks. The 1st task within this activity is to review the organization needs that originally initiated the project. The 2nd task is to identify, at an abstract or general level, the expected capabilities of the new system. Thus, it helps us to define the goal to be achieved & the boundary of the system. A clear understanding of the problem will help us in building a better system & reduce the risk of project failure. It also specifies the resources that have to be made available to the project.

Three important factors project goal, project bounds & the resource limits are sometimes called the project's term of reference.

Phase II (feasibility study):

The first study aspect is whether the current project is technically feasible i.e. whether the project be carried out with the current equipment, existing software and available personnel. If new technology is required than what is the likelihood that it can be developed?

The second study aspect is whether the project is economically feasible i.e. are there sufficient benefits in creating the system to make the cost acceptable. Are the costs of not creating the system so great that the project must be undertaken?

The third study aspect is whether the project is operationally feasible or not i.e. whether the system will be used if it is developed and implemented? Project is worth developing only if it can meet institutions operating requirements.

The feasibility study proposes one or more conceptual solutions to the problem set for the project. The objective in assessing feasibility is to determine whether a development project has a reasonable chance of success. It helps us to determine the input & output of the system. The following are the criteria that are considered to confirm the project feasibility.

The following feasibility study was undertaken for the proposed system:

Technical feasibility:

At first it’s necessary to check that the proposed system is technically feasible or not & to determine the technology and skill necessary to carry out the project. If they are not available then find out the solution to obtain them. Hardware is already available in the collage.

Economic feasibility:

While considering economic feasibility, it is checked in points like performance, information and outputs from the system. MS Access is available in one package of the windows operating system & does not require additional software cost for the client tools. The cost incurred to develop the system is freeware & does not incur the cost to the project. Backend database technology is a freeware. This justifies economical feasibility of the system.

Social feasibility:

Although generally there is always resistance, initially to any change in the system is aimed at reliving the work load of the users to extent the system is going to facilitate user to perform operations like calculating salary amounts and deductions, generating reports with less possible errors. Thus there is no reason to make system socially unfeasible.

Operational feasibility:

The operational feasibility is obtained by consulting with the system users. Check that proposed solution satisfies the user needs or not. There is no resistance from employee since new system is helpful. The existing system is manual system, while the new system is computerized and extremely user friendly.

Software details of the proposed system:

- **Front End:- Visual Basic 6.0**
- **Back End :- MS Access**

Phase III (System Analysis):

The phase is detailed appraisal of the existing system. This appraisal includes how the system works and what it does. It also includes finding out more detail- what are the problems with the system and what user requires from the system or any new change in the system.

The output of this phase results in detail model of the system. The model describes the system functions & data & system information flow. The phase also contains the detail set of user requirements are used to set objectives for new system.

System study:

It is always necessary to study and recognize the problems of the existing system, which will help in finding out the requirements for new system. System study helps in finding different alternatives for better solution.

The project study basically deals with different operations and steps involved in generation of examination mark sheets. It includes:

1. Data gathering
2. Study of existing system
3. Analyzing problem
4. Studying various documents
5. Feasibility study for further improvements

Following are the steps taken during the initial study:

- Initially, we collected all the information, which they wanted to store.
- Then we studied the working of the current system which is done manually. We noted the limitations of that system which motivated them to have a new system
- Then we analyzed the format the reports generated by the system.

With the help these documents we got basic ideas about the system as well as input & output of the developed system.

GANTT CHART

SR.NO	PHASES	START DATE	DURATION (DAYS)	FINISH DATE	Sign
1	PROJECT SEARCH	8/1/2010	30	8/31/2010	
2	FINALIZE PROJECT	8/11/2010	8	8/19/2010	
3	REQUIREMENT OF PROJECT	8/20/2010	5	8/25/2010	
4	SCHEDULLING THE PROJECT	8/30/2010	18	9/17/2010	
5	GATHER INFORMATION	9/5/2010	9	9/14/2010	
6	BUILT PROTOTYPE	9/13/2010	23	10/14/2010	
7	DATA & PROGRAM MODEL	9/30/2010	7	10/7/2010	
8	CONTEXT LEVEL DFD	10/7/2010	10	10/17/2010	
9	SYSTEM DESIGN	10/12/2010	8	10/20/2010	
10	SYSTEM FLOW CHART, ALL DFD	10/17/2010	30	11/16/2010	
11	FORM & REPORT DESIGINING	11/1/2010	10	11/11/2010	
12	PROJECT CODING	11/10/2010	35	12/21/2010	
13	MODEL TESTING WITH VALIDATION	12/8/2010	8	12/16/2010	
14	SYSTEM INTEGRATION	12/12/2010	4	12/16/2010	
15	SYSTEM TESTING	12/18/2010	8	12/21/2010	
16	COMPLETE DOCUMENTATION	12/20/2010	20	1/14/2011	
17	INSTALL PROGRAM	1/12/2011	3	1/15/2011	

SYSTEM IMPLEMENTATION

4.1 Methodology Adopted

The Spiral Model:

The *spiral model,* originally proposed by Boehm, is evolutionary software process model that couples the iterative nature of prototyping with the controlled and systematic aspects of the linear sequential model. It provides the potential for rapid development of incremental versions of the software. Using the spiral model, software is developed in a series of incremental releases. During early iterations, the incremental release might be a paper model or prototype. During later iterations, increasingly more complete versions of the engineered system are produced. A spiral model is divided into a number of framework activities, also called *task regions.*6 typically, there are between three and six task regions. Figure depicts a spiral model that contains six task regions:

• **Customer communication**—tasks required to establish effective communication between developer and customer.

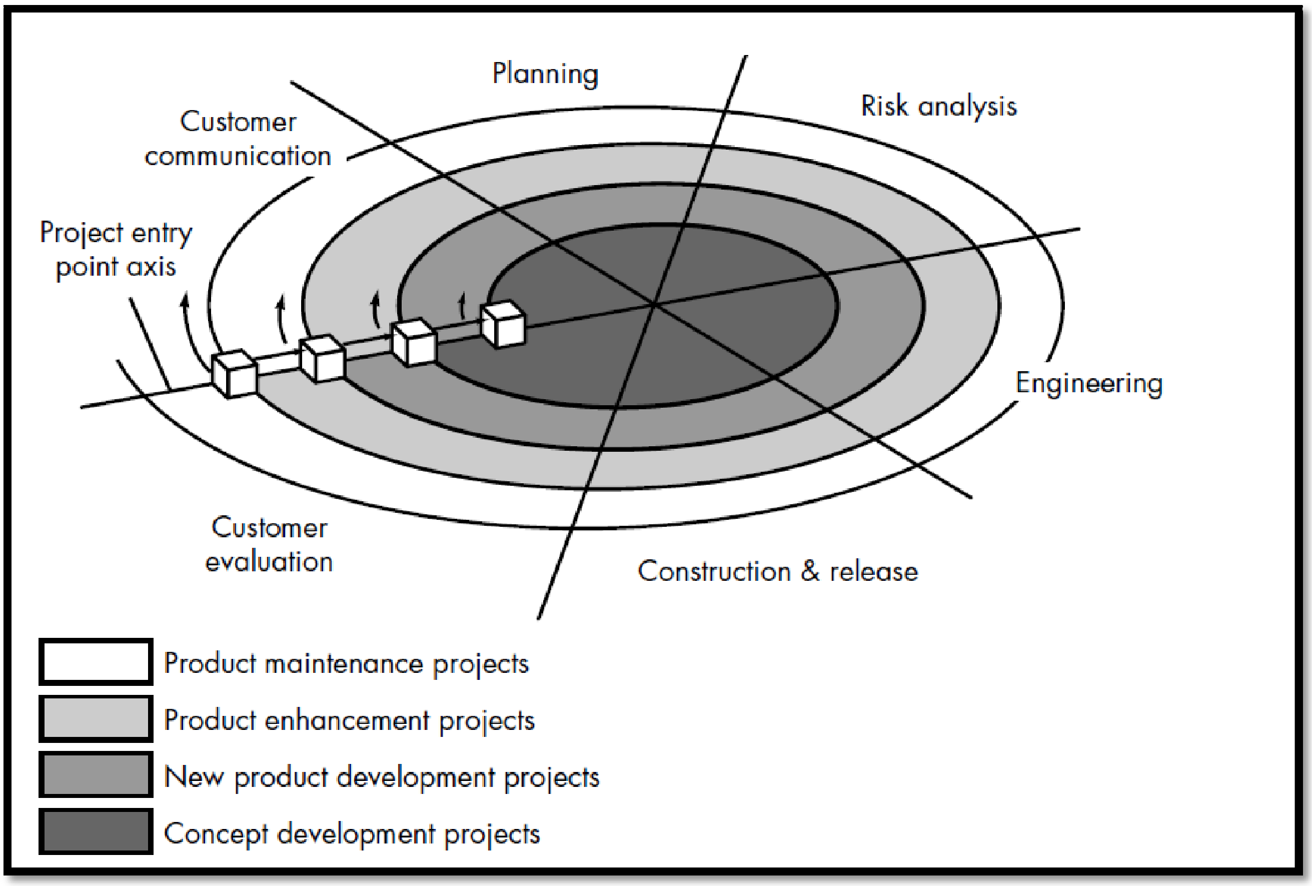
• **Planning**—tasks required to define resources, timelines, and other project related information.

• **Risk analysis**—tasks required to assess both technical and management risks.

• **Engineering**—tasks required to build one or more representations of the application.

• **Construction and release**—tasks required to construct, test, install, and provide user support (e.g., documentation and training).
• **Customer evaluation**—tasks required to obtain customer feedback based on evaluation of the software representations

created during the engineering stage and implemented during the installation stage. Each of the regions is populated by a set of work tasks, called a *task set,* that are adapted to the characteristics of the project to be undertaken. For small projects, the number of work tasks and their formality is low. For larger, more critical projects, each task region contains more work tasks that are defined to achieve a higher level of formality. In all cases, the umbrella activities (e.g., software configuration management and software quality assurance) noted is applied. As this evolutionary process begins, the software engineering team moves around the spiral in a clockwise direction, beginning at the center. The first circuit around the spiral might result in the development of a product specification; subsequent passes around the spiral might be used to develop a prototype and then progressively more sophisticated versions of the software. Each pass through the planning region results in adjustments to the project plan. Cost and schedule are adjusted based on feedback derived from customer evaluation. In addition, the project manager adjusts the planned number of iterations required to complete the software. Unlike classical process models that end when software is delivered, the spiral model can be adapted to apply throughout the life of the computer software. An alternative view of the spiral model can be considered by examining the *project entry point axis,* also shown in Figure. Each cube placed along the axis can be used to represent the starting point for different types of projects.



A "concept development project" starts at the core of the spiral and will continue (multiple iterations occur along the spiral path that bounds the central shaded region) until concept development is complete. If the concept is to be developed into an actual product, the process proceeds through the next cube (new product development project entry point) and a "new development project" is initiated. The new product will evolve through a number of iterations around the spiral, following the path that bounds the region that has somewhat lighter shading than the core. In essence, the spiral, when characterized in this way, remains operative until the software is retired. There are times when the process is dormant, but whenever a change is initiated, the process starts at the appropriate entry point (e.g., product enhancement). The spiral model is a realistic approach to the development of large-scale systems and software.

Because software evolves as the process progresses, the developer and customer better understand and react to risks at each evolutionary level. The spiral model uses prototyping as a risk reduction mechanism but, more important, enables the developer to apply the prototyping approach at any stage in the evolution of the product. It maintains the systematic stepwise approach suggested by the classic life cycle but incorporates it into an iterative framework that more realistically reflects the real world. The spiral model demands a direct consideration of technical risks at all stages of the project and, if properly applied, should reduce risks before they become problematic.

4.2 **SYSTEM REQUIREMENTS**

Hardware and Software Specification:

HARDWARE:

1) *Minimum 5 GB HDD space*
2) *Pentium based processor*
3) *128 MB RAM*
4) *Printer (any)*
5) *Power Supply For Backup*

SOFTWARE**:**

1) *Microsoft Windows 98, 2000, XP*
2) *Microsoft Visual Basic 6.0 Enterprise Edition:*

Visual basic is great! It's an easy, economical and fast application development tool; it's a good prototyping tool and developer's love using it. Like any high-level programming language, Visual Basic lets the programmer write really awful programs, and with Visual Basic, you can screw up more easily and faster than ever! Important business logic can be attached to GUI widgets rather than placed in reusable objects, making it hard to share and reuse code.

3) *Microsoft Access:*

Microsoft Access is a relational database management system from Microsoft, packaged with Microsoft Office Professional, which combines the relational Microsoft Jet Database Engine with a graphical user interface. It can use data stored in Access/Jet. It supports substantial object-oriented (OO) techniques but falls short of being a fully OO development tool.

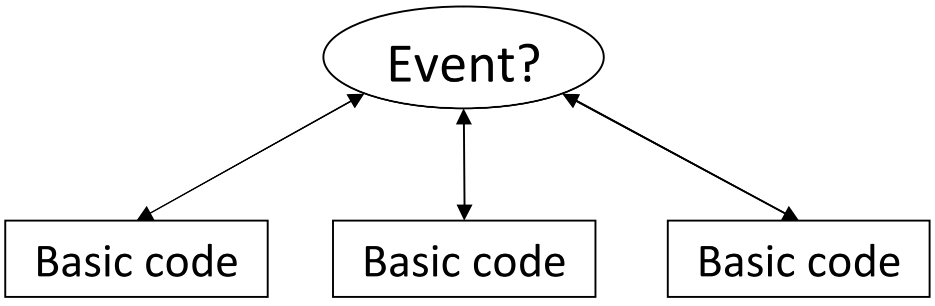
4.3 Technologies used

A) Visual basic 6.0 as the front end:

Here is some discussion about visual basic:

What is visual basic?

- *Visual basic* is a tool that allows you to develop windows (Graphical user interface- GUI) applications. The applications have familiar appearance to the user.
- *Visual basic* is *event driven;* meaning code remains idle until called upon to respond to some event (button pressing, menu selection.. etc). An event processor governs *visual basic.* Nothing happens until an event is detected. Once an event is detected, the code corresponding to that event (event procedure) is executed. Program control is returned the event processor.



Some features of visual basic:

Like the BASIC programming language, Visual Basic was designed to be easily learned and used by beginner programmers. The language not only allows programmers to create simple GUI applications, but can also develop complex applications. Programming in VB is a combination of visually arranging components or controls on a form, specifying attributes and actions of those components, and writing additional lines of code for more functionality. Since default attributes and actions are defined for the components, a simple program can be created without the programmer having to write many lines of code. Performance problems were experienced by earlier versions, but with faster computers and native code compilation this has become less of an issue.

Although programs can be compiled into native code executables from version 5 onwards, they still require the presence of runtime libraries of approximately 1 MB in size. This runtime is included by default in Windows 2000 and later, but for earlier versions of Windows like 95/98/NT it must be distributed together with the executable.

Forms are created using drag-and-drop techniques. A tool is used to place controls (e.g., text boxes, buttons, etc.) on the form (window). Controls have attributes and event handlers associated with them. Default values are provided when the control is created, but may be changed by the programmer. Many attribute values can be modified during run time based on user actions or changes in the environment, providing a dynamic application. For example, code can be inserted into the form resize event handler to reposition a control so that it remains centered on the form, expands to fill up the form, etc. By inserting code into the event handler for a keypress in a text box, the program can automatically translate the case of the text being entered, or even prevent certain characters from being inserted.

Visual Basic can create executables (EXE files), ActiveX controls, or DLL files, but is primarily used to develop Windows applications and to interface database systems. Dialog boxes with less functionality can be used to provide pop-up capabilities. Controls provide the basic functionality of the application, while programmers can insert additional logic within the appropriate event handlers. For example, a drop-down combination box will automatically display its list and allow the user to select any element. An event handler is called when an item is selected, which can then execute additional code created by the programmer to perform some action based on which element was selected, such as populating a related list.

Alternatively, a Visual Basic component can have no user interface, and instead provide ActiveX objects to other programs via Component Object Model (COM). This allows for server-side processing or an add-in module.

The language is garbage collected using reference counting, has a large library of utility objects, and has basic object oriented support. Since the more common components are included in the default project template, the programmer seldom needs to specify additional libraries. Unlike many other programming languages, Visual Basic is generally not case sensitive, although it will transform keywords into a standard case configuration and force the case of variable names to conform to the case of the entry within the symbol table. String comparisons are case sensitive by default, but can be made case insensitive if so desired.

B) <u>MICROSOFT ACCESS as the backend:</u>

Microsoft Access is a database package generally, used to design database applications. Microsoft Access is used in this project for following reasons:

- Microsoft Access able to store large data.
- Its DBMS
- Applying validation is easy in Microsoft Access.
- Creating relationship is not a complex task.
- It provides good graphical interface.
- Microsoft Access can execute any valid SQL query
- It provides all necessary forms of data types.
- Microsoft access has good connectivity with visual basic.

COST AND BENEFIT ANALYSIS

5.1 Cost Estimation

Cost required for the project is to install the software and hardware requirements. Software may include installing Microsoft Access on the system. Cost due to the time taken for completion of the project which can be around 5 months. A Gantt chart given in the beginning helps to understand this in a better way.

5.2 Benefit Analysis

Due to the introduction of this system the cost of handling the system is reduced. The cost mainly includes the charges for registry maintenance, receipt books, files, etc. To reduce the costs the new system was proposed. Positive aspects of the designed system which contributed to the benefit analysis are fast and easy storage of all information. It was also easy to retrieve any required details as fast as possible. There is no need for maintaining receipt books. The new system is very beneficial than because the system is fully automated.

EVENT TABLE

Event:

An occurrence at specific time and place that can be described and is worth remembering is known as Event.

Definition:

Trigger: An occurrence that tells the system that an event has occurred, either the arrival of data needing processing or of a point in time.

Source: An external agent or actor that supplies data to the system.

Activity: Behavior that the system performs when an event occurs.

Response: An output produces by the system that goes to the destination.

Destination: An external agent or the actor that receives data.

Event Table: The table that test event in rows and key pieces of information about each event in columns.

Event	Trigger	Source	Activity	Response	Destination
New Record	New Member For Gym, Tanning Or Both	Employee Or Owner	Make New Member Of Gym	Member Created	System
New Receipt	New Receipts For The Respective Member	Employee Or Owner	Generates Receipts For The Members	Transaction Occurred	Member
Update Member	Update	Owner Or Employee	Update Member Information	Returns Updated Information	System
New Employee	New Employee	Owner	Make New Employee Record	Return Addition Information	System
Payroll	Payroll For Employee	Owner	Issues Salary To Employee	Salary Information	Owner
Inventory	Update Inventory	Owner	Update Inventory Information Of Products	Returns Updated Information	System
Order Products	Order A Product	Owner	Order Products For Gym	Makes An New Order	Owner
Update Schedule	Update Schedule	Owner	Update Employee Schedule For Gym	Schedule Information For Employee	Owner

<u>DETAIL LIFE CYCLE OF THE PROJECT</u>

Phased development process

A development process consists of various phases, each phase ending with a defined output. The main reason for having a phased process is that it breaks the problem of developing software into successfully performing a set of phases, each handling a different concern of software development.

Requirement Analysis:

- Requirements analysis is done in order to understand the problem the software system is to solve. The goal of the requirements activity is to document the requirements in a software requirements specification document.
- There are two major activities in this phase: Problem Understanding or Analysis and Requirement Specification. In problem analysis, the aim is to understand the problem and its context, and the requirements of the new system that is to be developed.
- Once the problem is analyzed and essentials understood, the requirements must be specified in the requirements specification document. The requirements specification document. The requirement document must specify all functional and performance requirements; the formats of inputs and output; and all design constraints that exist due to political, economic, environmental, and security reasons.

Software Design:

- The purpose of the design phase is to plan a solution of the problem specified by the requirements documents. This phase is the first step in moving from the problem domain to the solution domain.
- The design activity often results in three separate outputs:-
  - Architecture Design –

    It focuses on looking at a system as a combination of many different components, and how they interact with each other to produce the desired results.

  - High Level Design –

    It identifies the module that should be built for developing the system and the specifications of these modules.

  - Design Level Design –

    The internal logic of each of the modules is specified.

Coding:

- The goal of the coding phase is to translate the design of the system into code in a given programming language. For a given design, the aim in this phase is to implement the design in the best possible way.
- The coding phase affects both testing and maintenance profoundly. Well-written code can reduce the testing and maintenance effort. The testing and maintenance costs of software are much higher than coding cost. Hence during coding the focus should be developing programs that are easy to read and understand, and not simply on developing programs that are easy to write. Simplicity and clarity should be strived for during the coding phase.

<u>Testing:</u>

- Testing is the major quality control measure used during software development. Its basic function is to detect defects in the software. The goal of testing is to uncover requirement, design, and coding errors in the programs.
- The starting point of testing is **unit testing,** where the different modules or components are tested individually.
- The modules are integrated into the system; **integration testing** is performed, which focuses on testing the interconnection between modules.
- After the system is put together, **system testing** is performed. Here the system is tested against the system requirements to see if all the requirements are met and if the system performs as specified by the requirements.
- Finally the acceptance testing is performed to demonstrate to the client, on real-life data of the client, the operation of the system.
- Then for different test. A test case specification document is produced, which lists all the different test cases, together with the expected outputs.
- The final output of the testing phase is the test report and the error report, or set of such reports. Each test report contains the set of test cases and the result of executing the code with these test cases.

<u>ENTITY RELATIONSHIP DIAGRAMS</u>

ERD:

The entity-relationship (ER) data model allows us to describe the data involved in a real world enterprise in terms of object and their relationships and is widely used to develop an initial database design.

The ER model is important primarily for its role in database design. It provides useful concepts that allow us to move from an informal description of what users want from their database to a more detailed and precise description that can be implemented in a DBMS. The ER model is used in a phase called "Conceptual Database Design". It should be noted that many variations of ER diagrams are in use and no widely accepted standards prevail.

ER modeling is something regarded as a complete approach to design a logical database scheme. This is incorrect because the ER diagram is just an approximate description of data, constructed through a very subjective evaluation of the information collected during requirements analysis.

Entity:

ER modeling is something regarded as a complete approach to design a logical database schema. This is incorrect because the ER diagram is just an approximate description of data, constructed through a very subjective evaluation of the information collected during requirements analysis.

An entity is an object in the real world that is distinguishable from other objects. Examples include the following: The address of the manager of the institution, a Person with unique name etc.

It is often useful to identify a collection of similar entities. Such a collection is called as "Entity set". Note that entity set need not be disjoint.

Attributes:

An entity is described using a set of attributes. All entities in a given entity set have the same attributes; this essentially what we mean by similar. Our choice of attributed reflects the level of detail at which we wish to represent information in crisis.

For e.g. The Admission entity set would use the name, age, and qualification of the students as the attributes. In this case we will store the name, the registry no, the course enrolled of the student and not his/her address or the gender.

Domain:

For each attribute associated with an entity set, we must identify a domain of possible values.

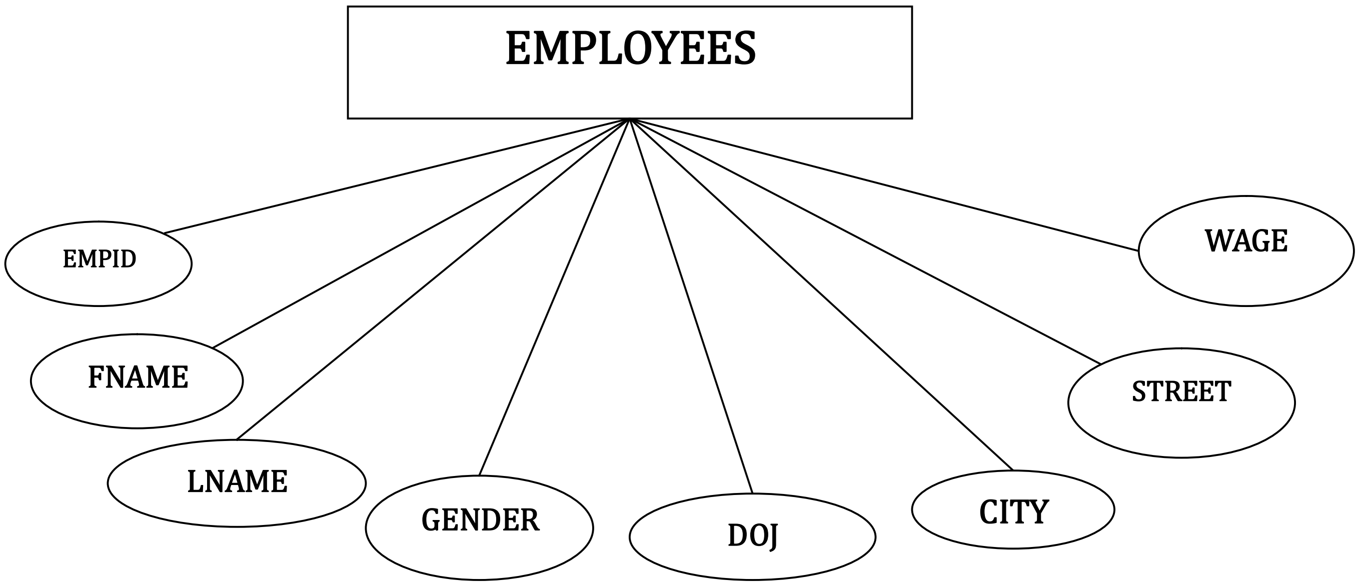
For e.g. the domain associated with the attribute name of the student might be of the set of 20-character string.

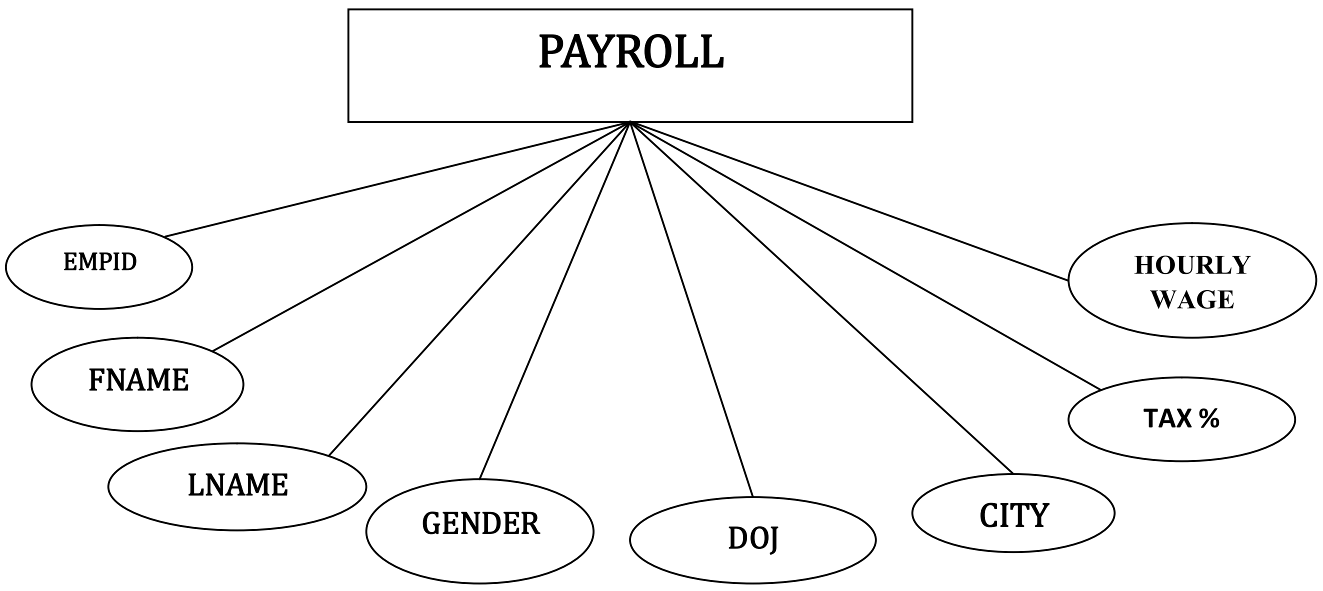
Another example would be the ranking of the students in the institute would be on the scale of 1-6, the associated domain consists of integers 1 through 6.

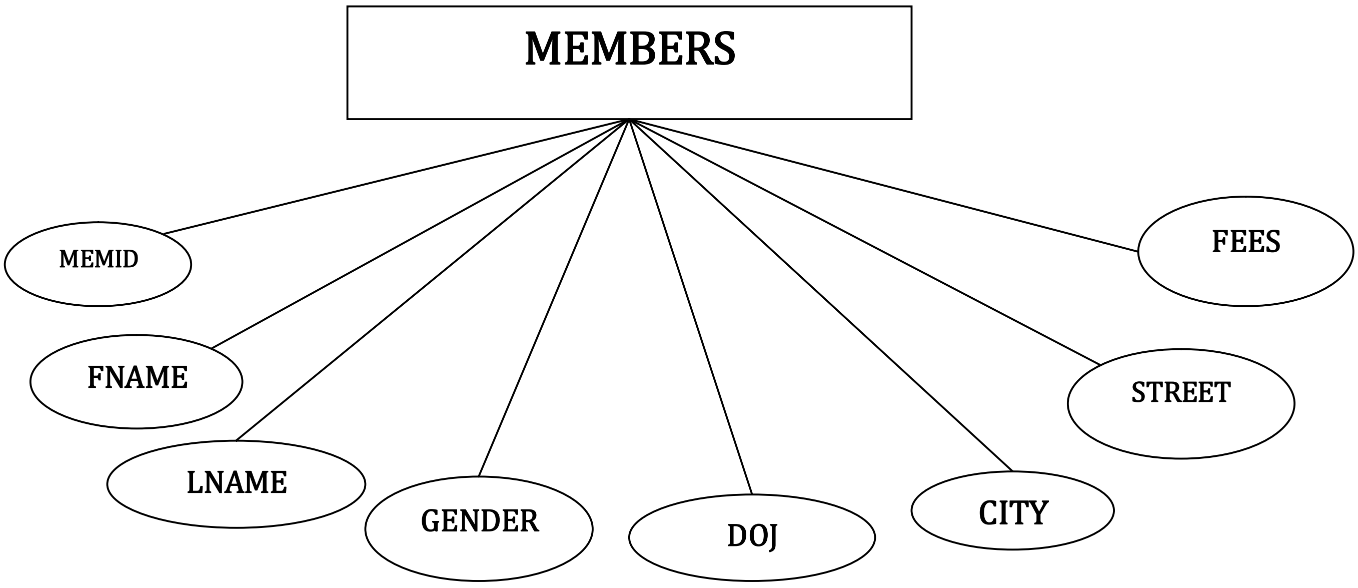
Key:

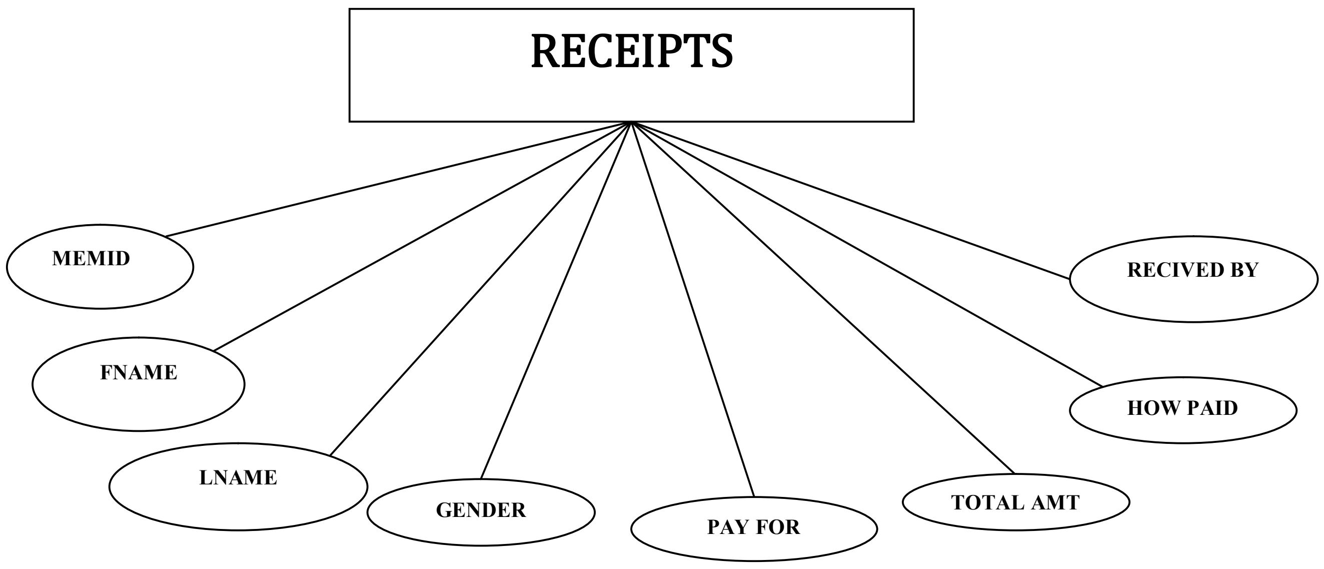
Further, for each entity set we choose a key. A key is a minimal set of attributed whose values uniquely identify an entity in the set. There could be more than one candidate; if so we designate one of them as primary key. For now we will assume that each entity set contains at least one set of attributes that uniquely identify an entity in the entity set; that is the set of attributes contains a key.

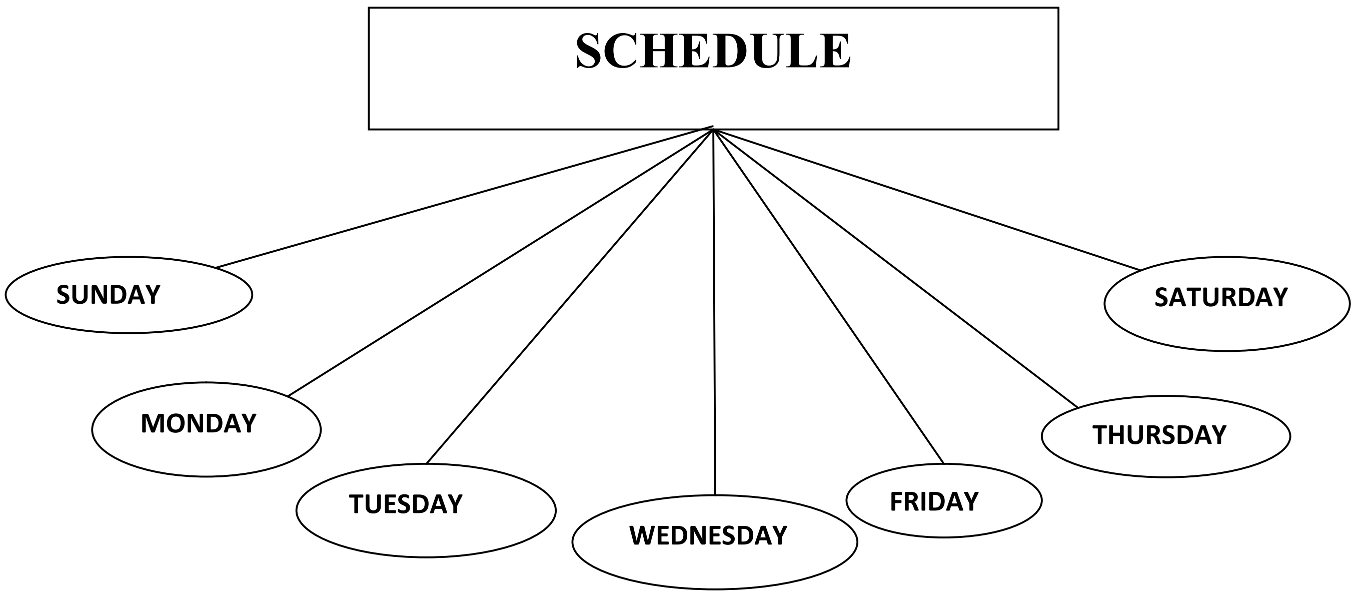
DIAGRAMS

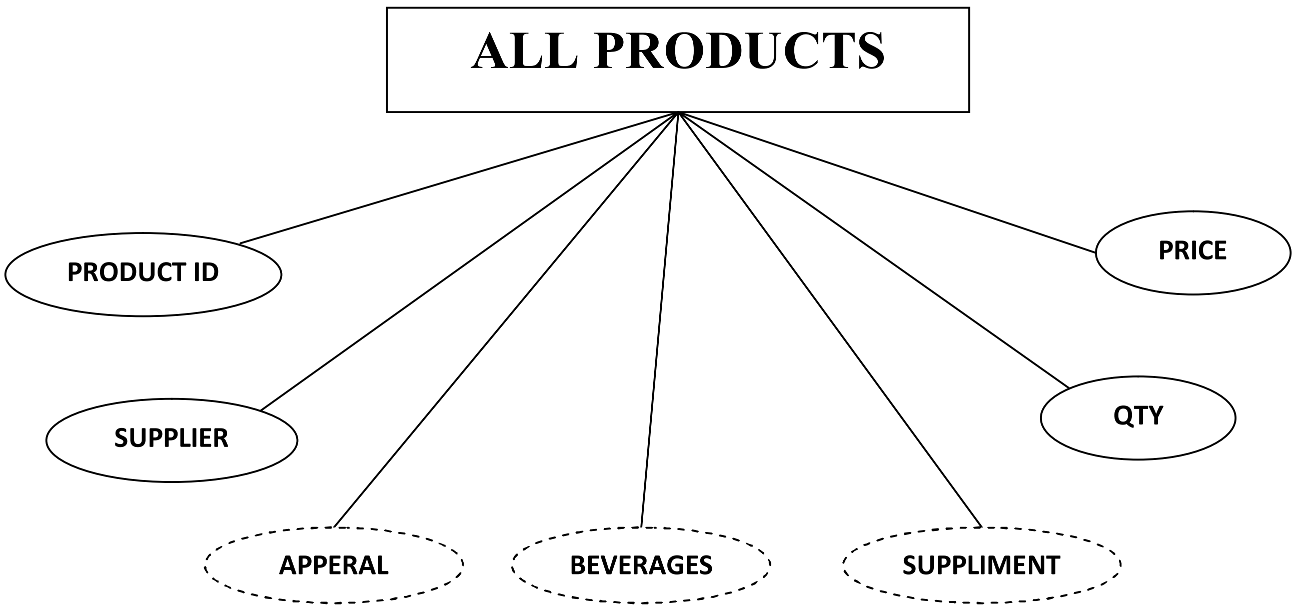


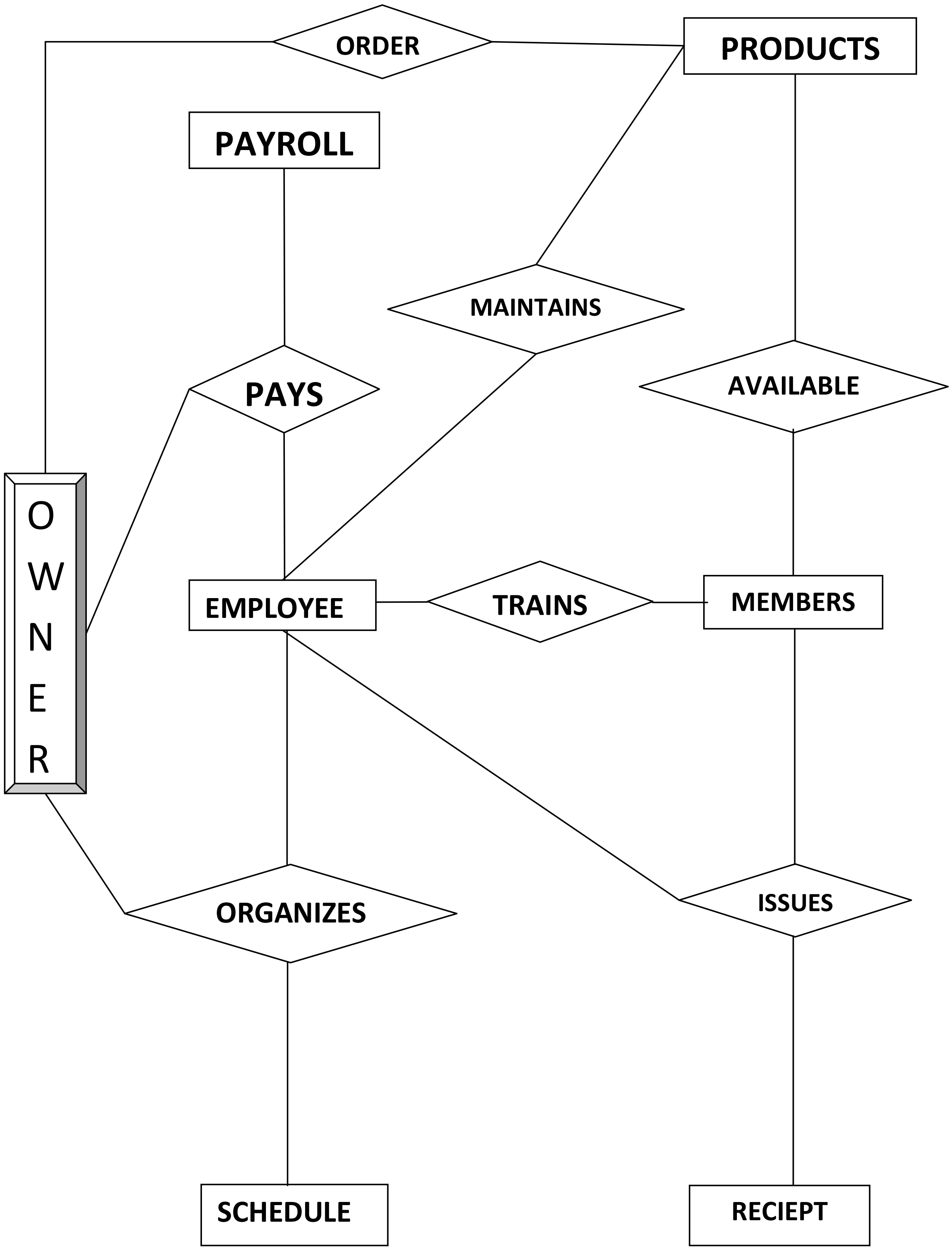

PAYROLL
EMPID
FNAME
LNAME
GENDER
DOJ
CITY
TAX %
HOURLY WAGE


MEMBERS
MEMID
FNAME
LNAME
GENDER
DOJ
CITY
STREET
FEES


RECEIPTS
MEMID
FNAME
LNAME
GENDER
PAY FOR
TOTAL AMT
HOW PAID
RECIVED BY


SCHEDULE
SUNDAY
MONDAY
TUESDAY
WEDNESDAY
FRIDAY
THURSDAY
SATURDAY


ALL PRODUCTS
PRODUCT ID
SUPPLIER
APPERAL
BEVERAGES
SUPPLIMENT
QTY
PRICE


ORDER
PRODUCTS
PAYROLL
MAINTAINS
PAYS
AVAILABLE
OWNER
EMPLOYEE
TRAINS
MEMBERS
ORGANIZES
ISSUES
SCHEDULE
RECIEPT

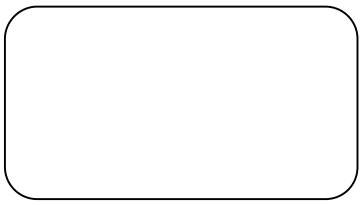
DATA FLOW DIAGRAM

Data Flow Diagram:

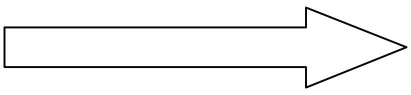
A data flow (DFD) is a graphical system model that shows all of the main requirements for an information system in one datagram: inputs and outputs, processes, and data storage. A DFD describes what data flows rather than how it is processed. Everyone working on a development project can see all aspects of the system working together at once with DFD. That is one reason for its popularity. The DFD is also easy to read because it is graphical model. The DFD is mainly used during problem analysis. End Users, management, and all information systems workers typically can read and interpret the DFD with minimal training.

DFD SYMBOLS:

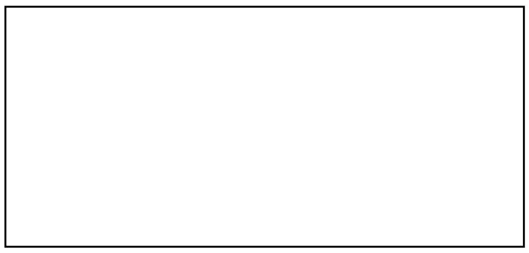
1. Process



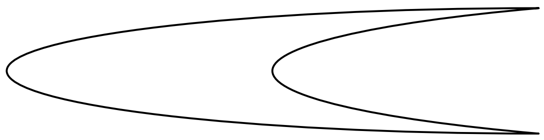
2. Data Flow



3. External Entity

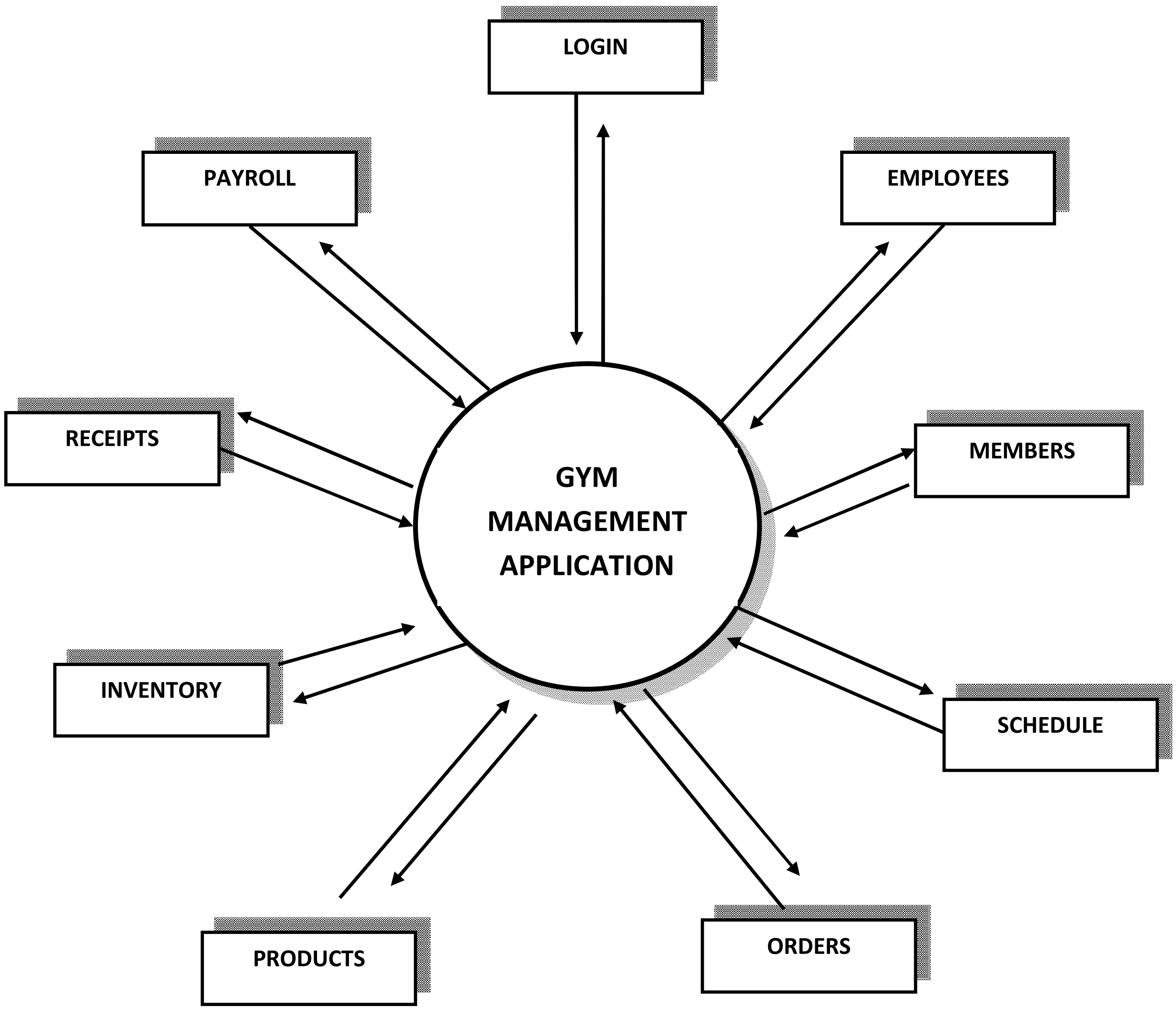


4. Data Store



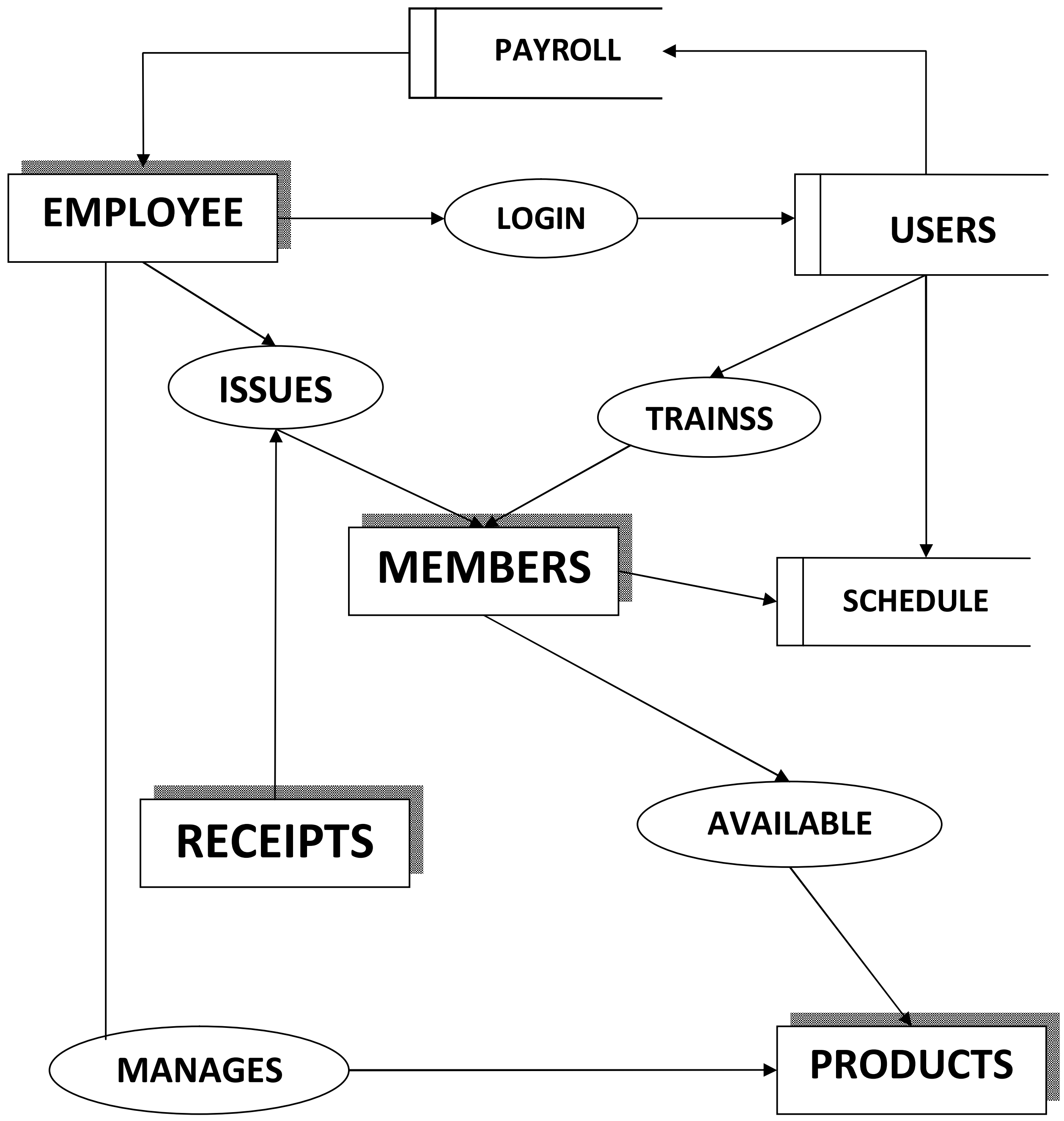
CONTEXT LEVEL DIAGRAM

The context diagram is useful for showing boundaries. The system scope is defined by what is represented within single process and what is represented as an external agent. External agents that supply or receive data from the system are outside of the system scope. Everything else is inside the system scope. Data stores are not usually shown on the context diagram because all of the system's data stores are considered to be within the system scope. The context diagram is simply the highest-level DFD. It is also called as Level 0 DFD. The context diagram provides a good overview of the scope of the system, showing the system in "context" but it does not show any detail about the processing that takes place inside the system.

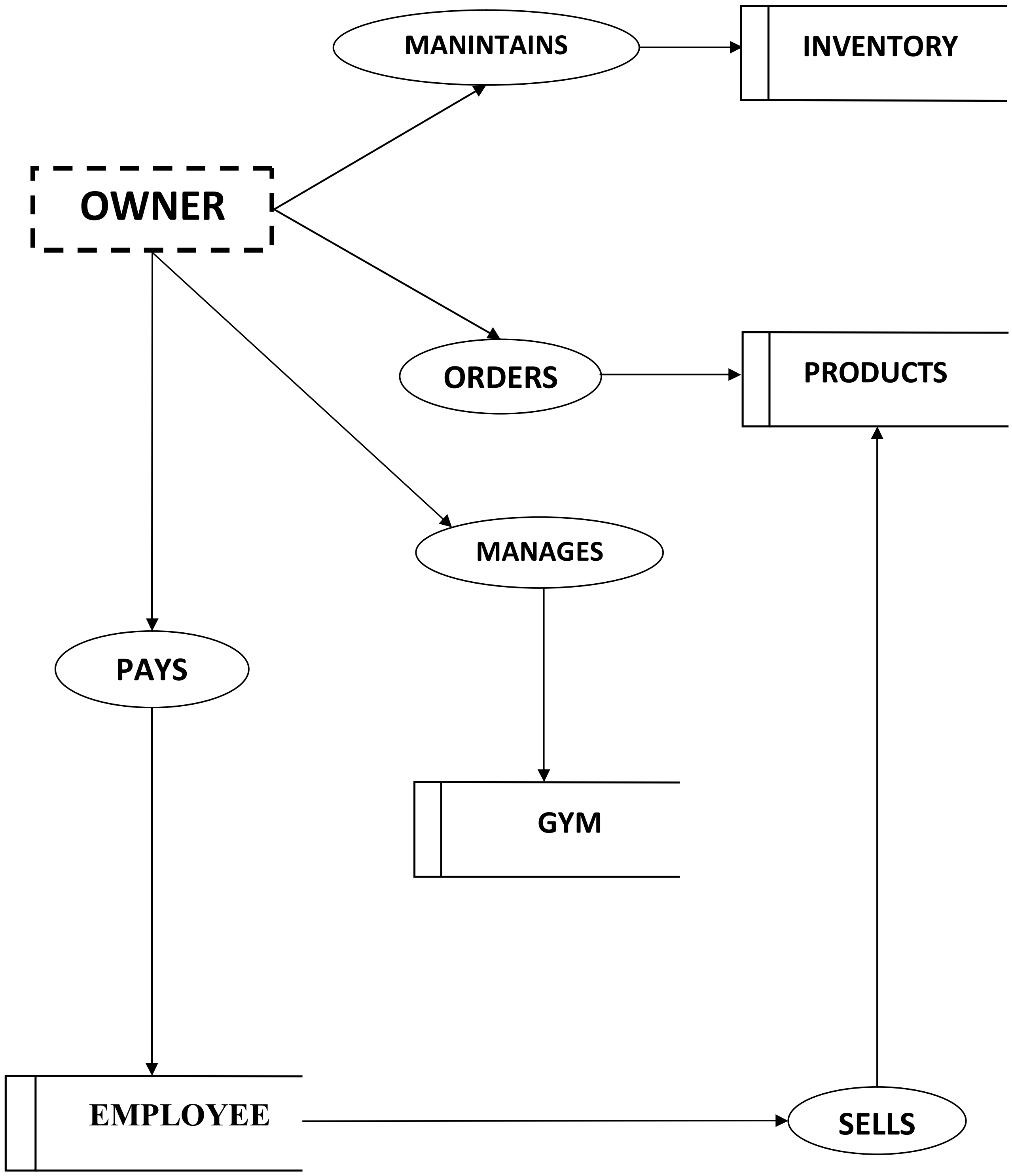

LOGIN
PAYROLL
EMPLOYEES
RECEIPTS
GYM MANAGEMENT APPLICATION
MEMBERS
INVENTORY
SCHEDULE
PRODUCTS
ORDERS

Level 1 DFD:-

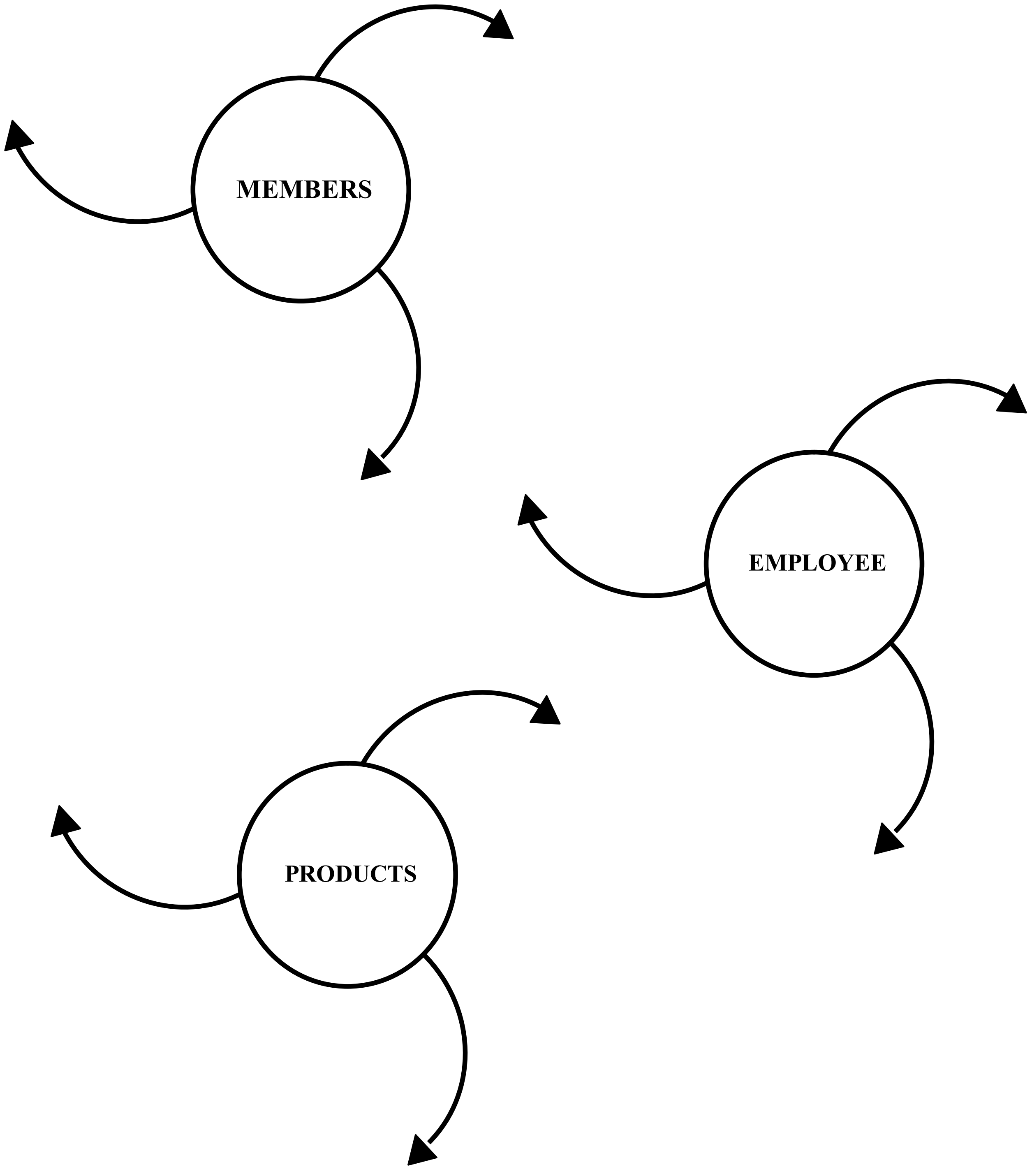
Context diagrams are diagrams where the whole system is represented as a single process. A level 1 DFD notates each of the main sub-processes that together form the complete system. We can think of a level 1 DFD as an “exploded view” of the context diagram. You may also need some downward leveling. That is, the processes identified in the preliminary DFD may not turn out to be primitive processes and may require downward portioning into lower-level DFDs.


PAYROLL
EMPLOYEE
LOGIN
USERS
ISSUES
TRAINSS
MEMBERS
SCHEDULE
RECEIPTS
AVAILABLE
MANAGES
PRODUCTS

LEVEL 2 DFD:


MANINTAINS
INVENTORY
OWNER
ORDERS
PRODUCTS
MANAGES
PAYS
GYM
EMPLOYEE
SELLS

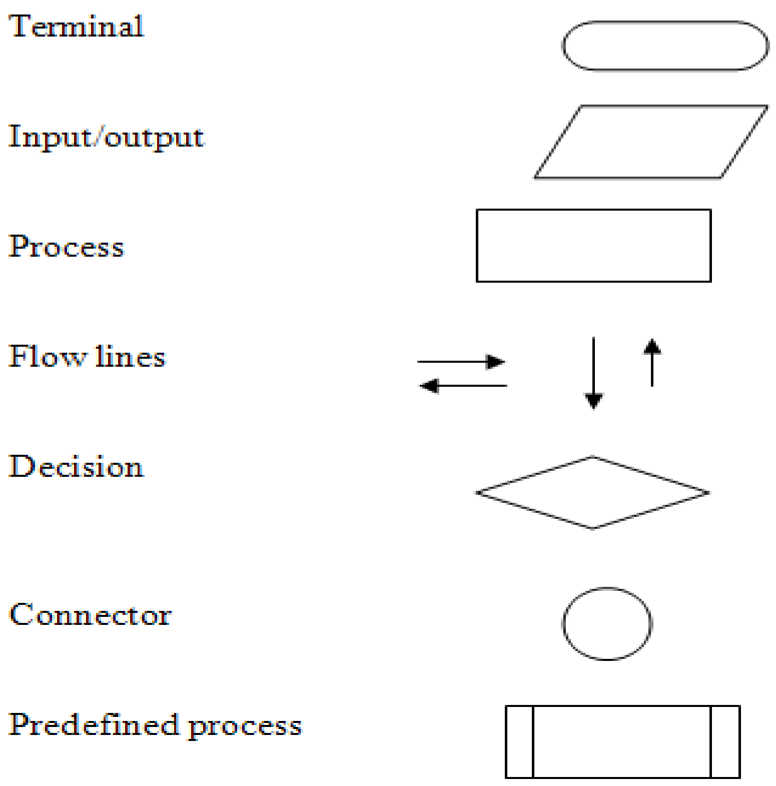
SUB LEVELS

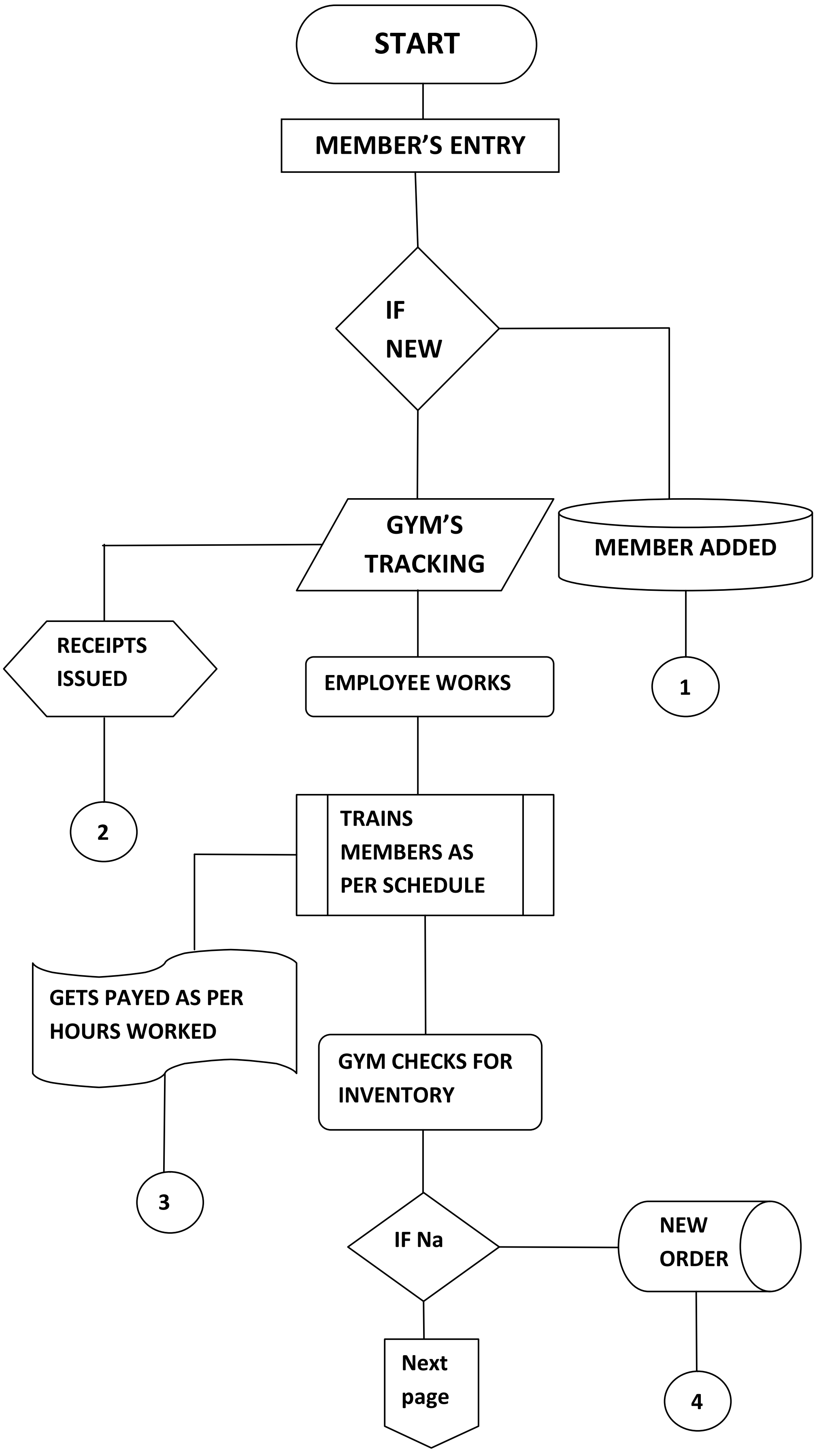


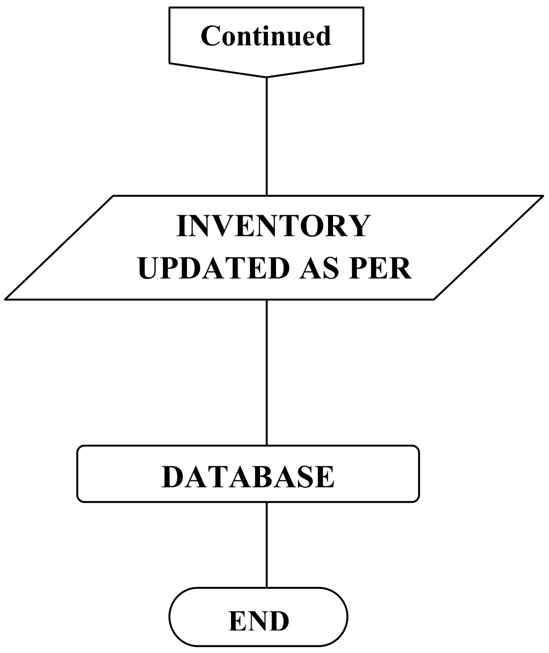
SYSTEM FLOW CHART

Flow chart:

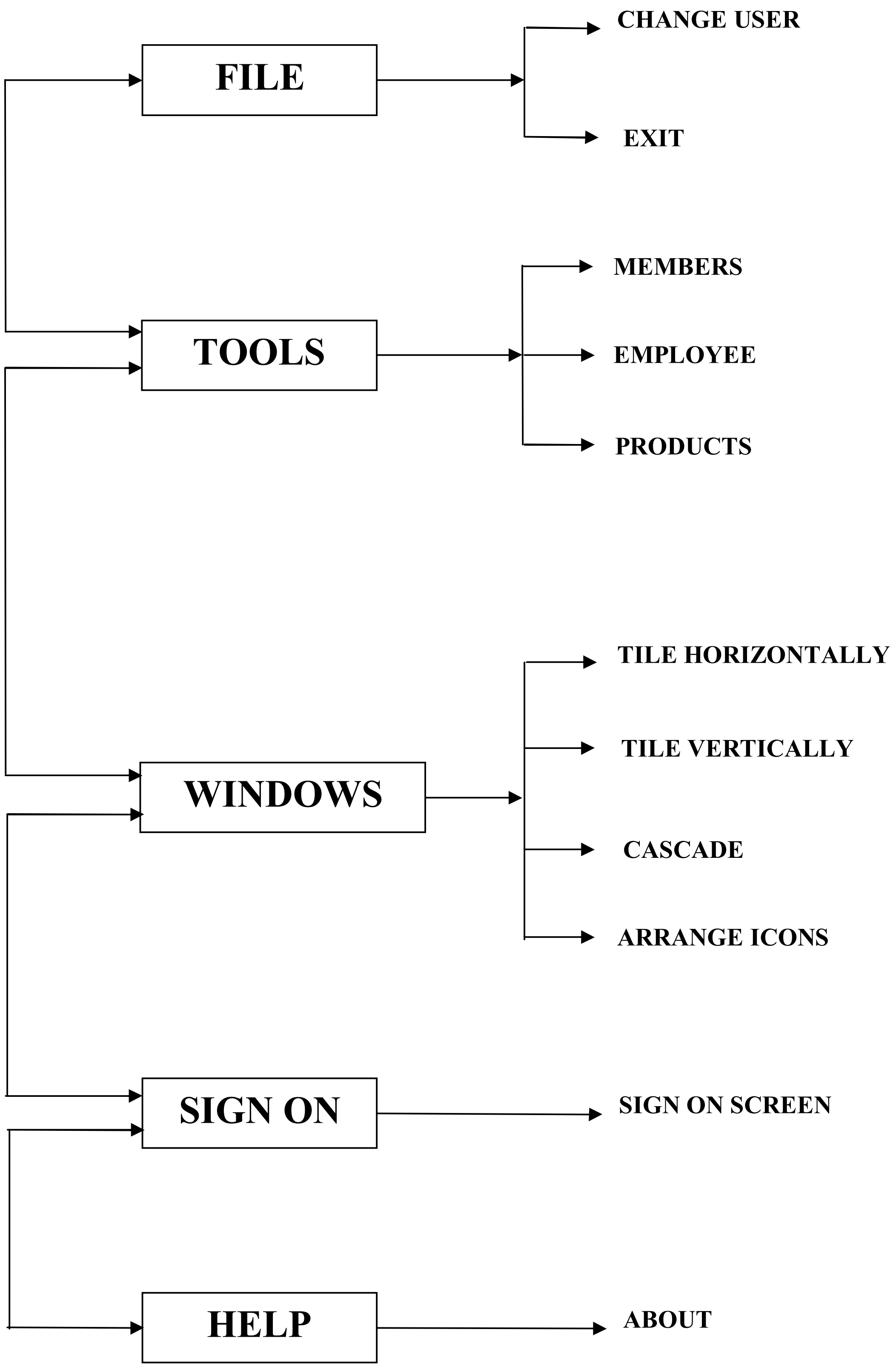
Flow charts are required to understand the system well. With the help of these charts it becomes easy to understand the inputs and outputs of the system which is helpful in later stages of development of the software.




START
MEMBER'S ENTRY
IF NEW
GYM'S TRACKING
MEMBER ADDED
RECEIPTS ISSUED
EMPLOYEE WORKS
1
2
TRAINS MEMBERS AS PER SCHEDULE
GETS PAYED AS PER HOURS WORKED
GYM CHECKS FOR INVENTORY
3
IF Na
NEW ORDER
Next page
4


Continued
INVENTORY UPDATED AS PER
DATABASE
END

MENU TREE


FILE
CHANGE USER
EXIT
TOOLS
MEMBERS
EMPLOYEE
PRODUCTS
WINDOWS
TILE HORIZONTALLY
TILE VERTICALLY
CASCADE
ARRANGE ICONS
SIGN ON
SIGN ON SCREEN
HELP
ABOUT

DATA DICTIONARY

Products		
Fields	**Data type**	**Constraint**
Category	Text	
Product id	Number	Primary key
Description	Text	
Brand	Text	
Supplier	Text	
Case	Number	
Ncase	Number	
Qty	Number	
Case price	Currency	
Sale price	Currency	
Order date	Date/time	
Last inventory	Date/time	

EMPLOYEE		
Fields	**Data Type**	**Constraint**
numemp	number	
employee id	text	primary key
last name	text	
first name	text	
gender	text	
doh	date/time	
street	text	
city	text	
state	text	
zip code	text	
dob	date/time	
phone number	text	
soc	text	
hourly wage	currency	
tax rate	number	

Password		
Fields	**Data Type**	**Constraint**
screen	Text	
login	Text	

PAYROLL		
Fields	**Data Type**	**Constraint**
paynum	auto number	
employee id	text	primary key
last name	text	
first name	text	
hourly wage	currency	
hours worked	number	
date paid	date/time	
gross pay	currency	
tax withheld	currency	
net pay	currency	

SCHEDULE		
Fields	**Data Type**	**Constraint**
sun	text	
mon1	text	
tue1	text	
wed1	text	
thurs1	text	
fri1	text	
sat	text	
mon2	text	
tue2	text	
wed2	text	
thurs2	text	
fri2	text	

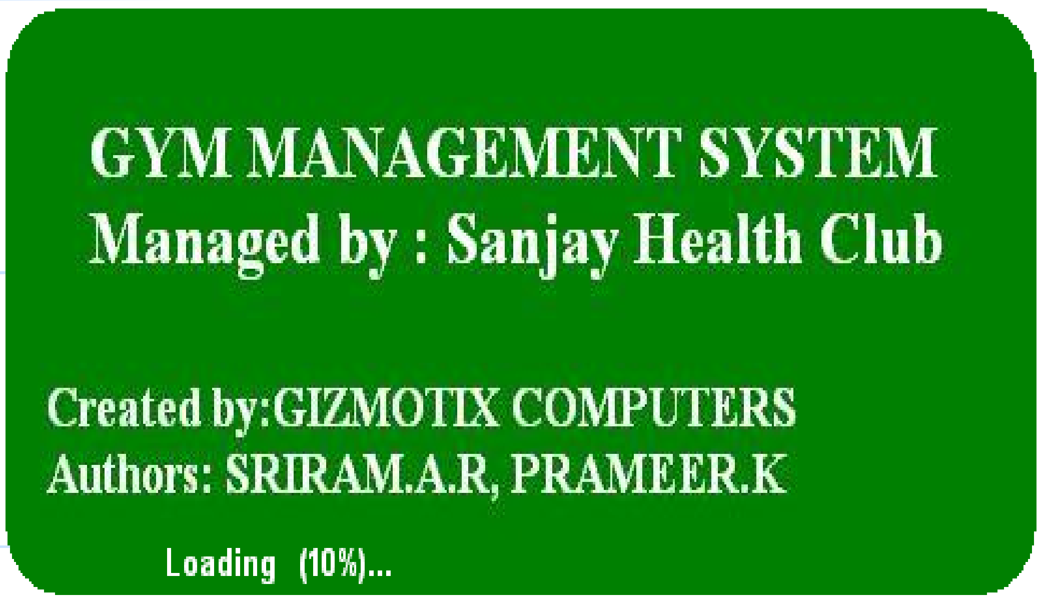
RECIPT		
Fields	**Data Type**	**Constraint**
num	auto number	
date	date/time	
memberid_rec	number	primary key
lfname_rec	text	
gymex_rec	date/time	
tanex_rec	date/time	
old balance	currency	
new charge	currency	
amount	currency	
new balance	currency	
pay for	text	
how paid	text	
check num	number	
prev	date/time	
next	date/time	
rec by	text	

MEMBERS		
Fields	**Data Type**	**Constraint**
member id	autono	primary key
last name	text	
firstname	text	
gender	text	
recommend	text	
membership	text	
gymex	date/time	
gymex type	text	
tanex	date/time	
tanex type	text	
street	text	
city	text	
state	text	
zip	number	
dob	date/time	
phoneno	text	
soc	text	
pay due	date/time	
amount due	currency	
install amount	currency	
balance	currency	
notes	memo	
gexp	yes/no	
texp	yes/no	
od	yes/no	

<u>Programs list (Form list)</u>

1. Splash screen
2. Password
3. MDI
4. Tools
5. Members
6. Receipts
7. All members
8. All receipts
9. Schedule
10. Employees
11. Payroll
12. Products
13. Orders
14. Inventory
15. All products
16. Calendar
17. About
18. Module

Splash screen

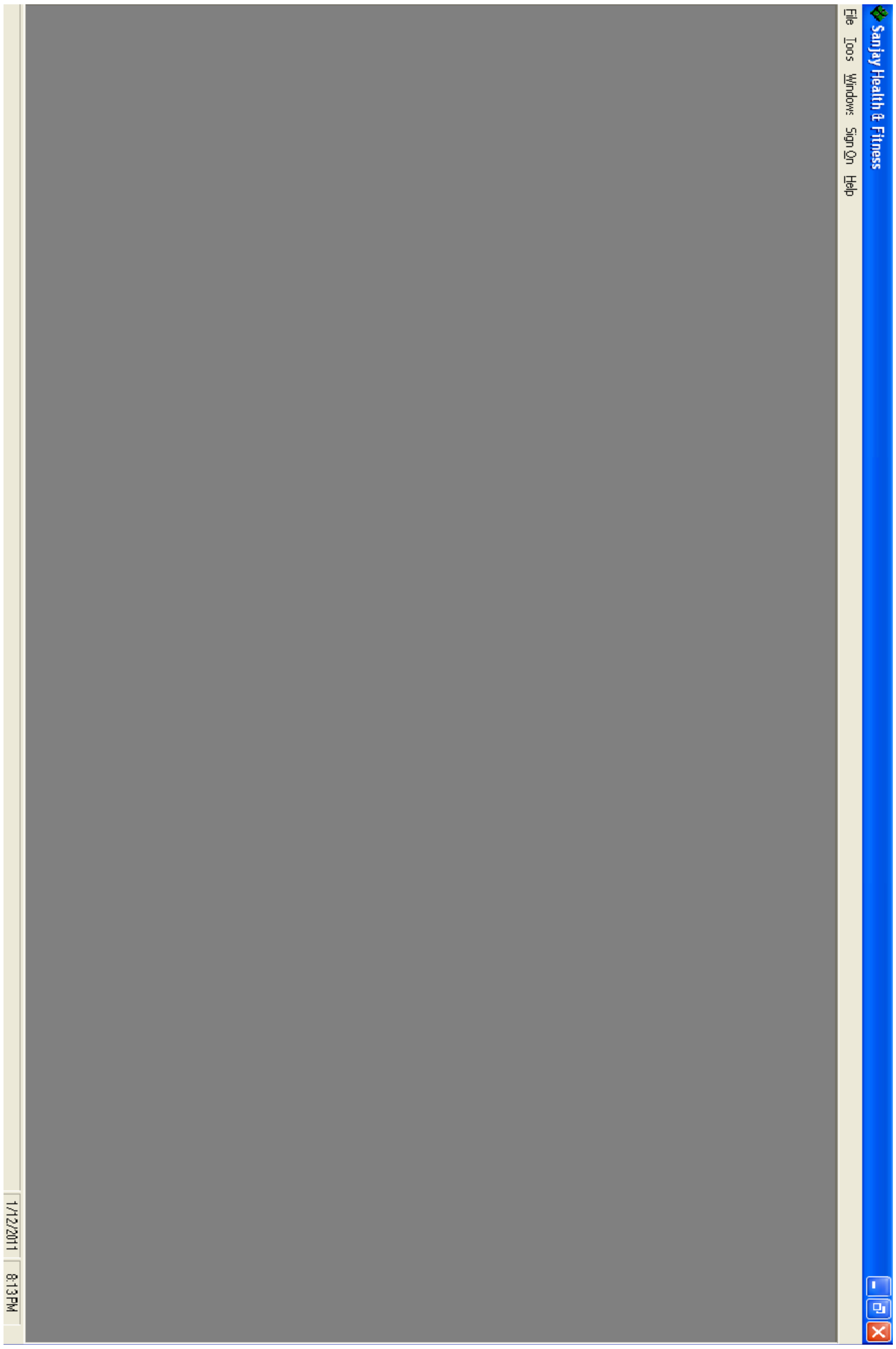


GYM MANAGEMENT SYSTEM
Managed by : Sanjay Health Club

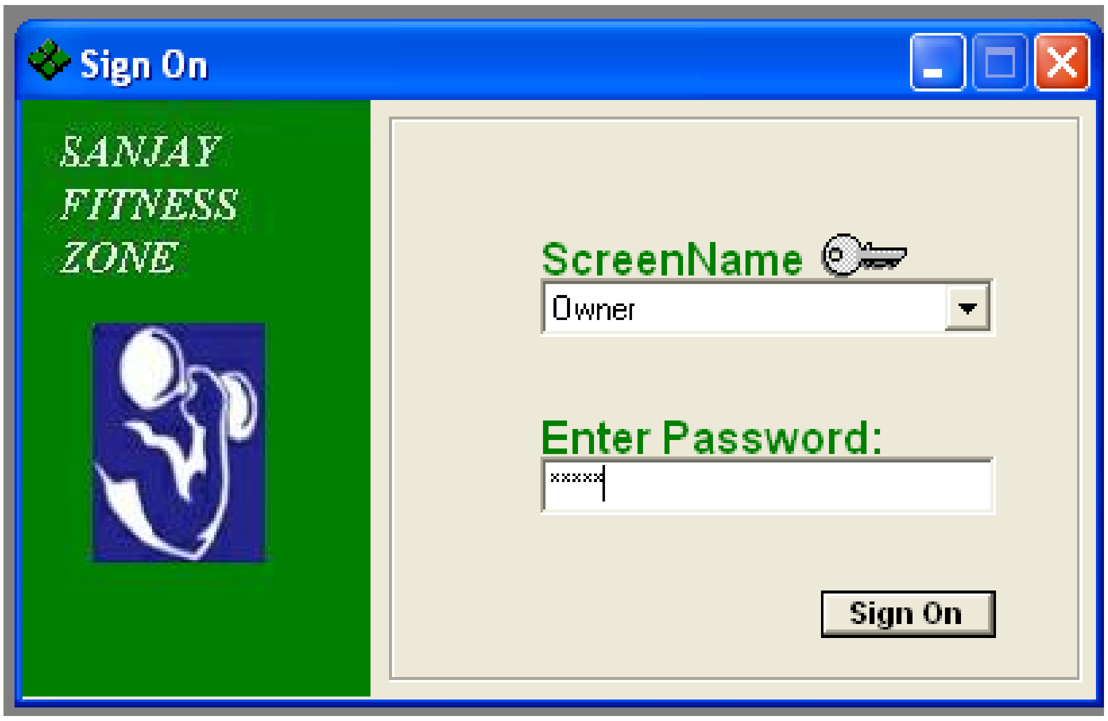
Created by:GIZMOTIX COMPUTERS
Authors: SRIRAM.A.R, PRAMEER.K

Loading (10%)...

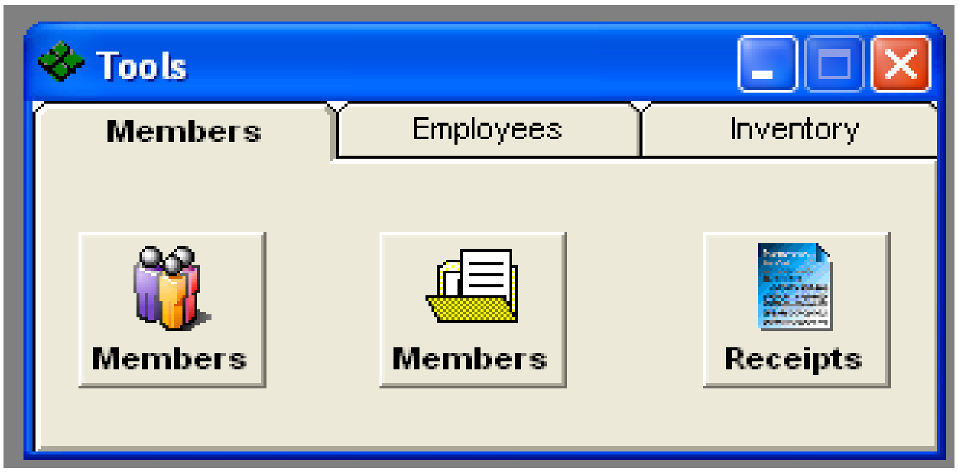
Mdi screen:


Sanjay Health & Fitness
File
Toos
Windows
Sign On
Help
1/12/2011
8:13PM

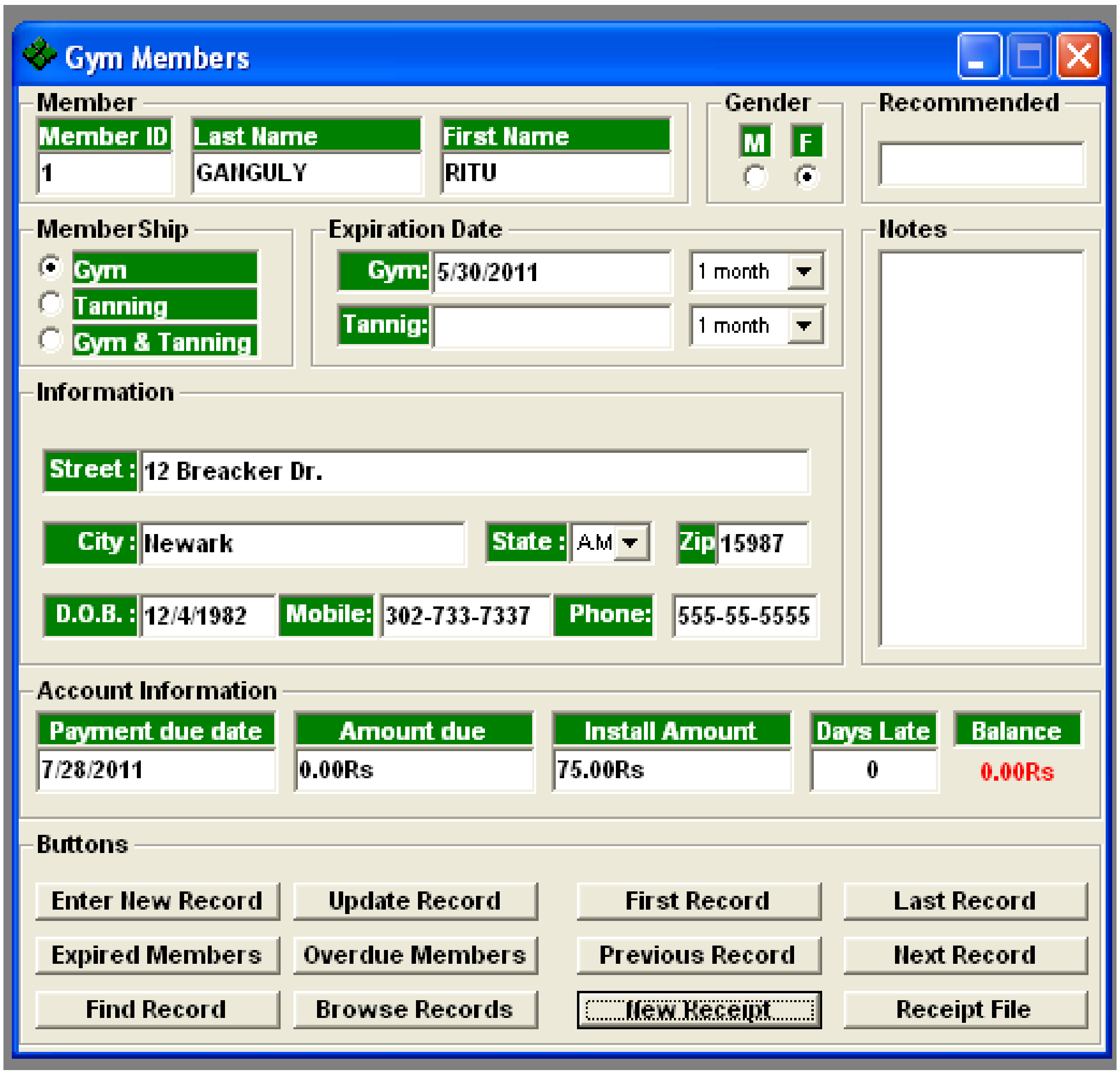
Login screen:



Tools Screen:



Members Screen:



Member

Member ID	Last Name	First Name
1	GANGULY	RITU

Gender: M ○ [✓ F]

Recommended:

MemberShip: [✓ Gym] Tanning; Gym & Tanning

Expiration Date

Gym: 5/30/2011 1 month

Tannig: 1 month

Notes

Information

Street : 12 Breacker Dr.

City : Newark State : AM Zip 15987

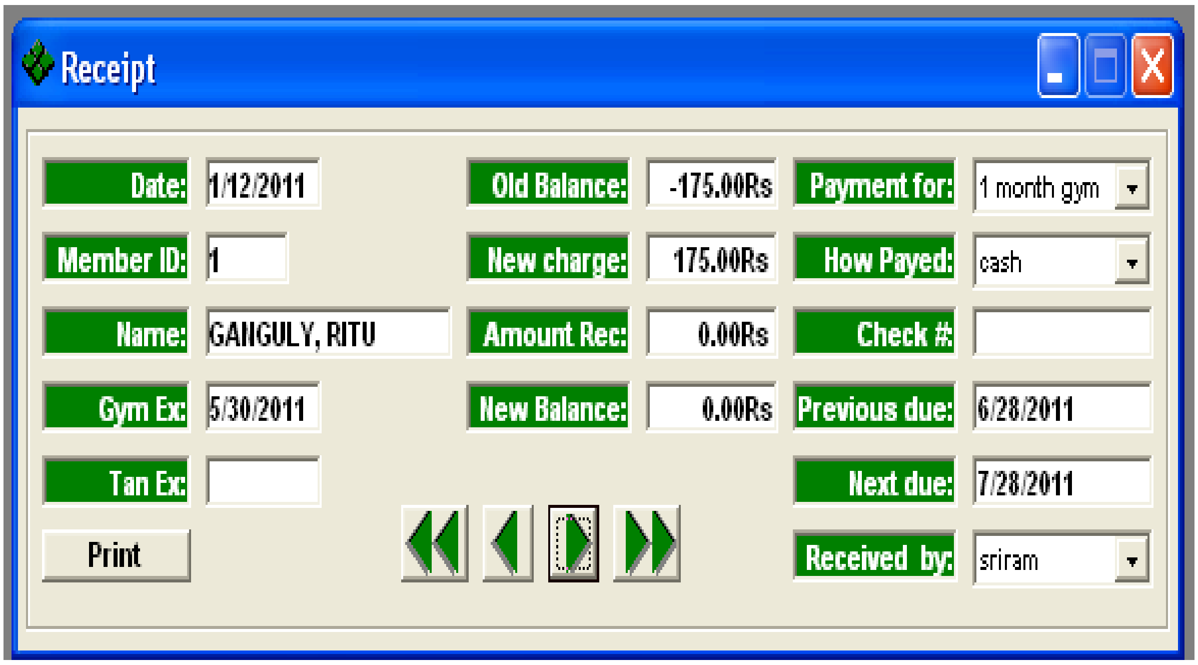
D.O.B. : 12/4/1982 Mobile: 302-733-7337 Phone: 555-55-5555

Account Information

Payment due date	Amount due	Install Amount	Days Late	Balance
7/28/2011	0.00Rs	75.00Rs	0	0.00Rs

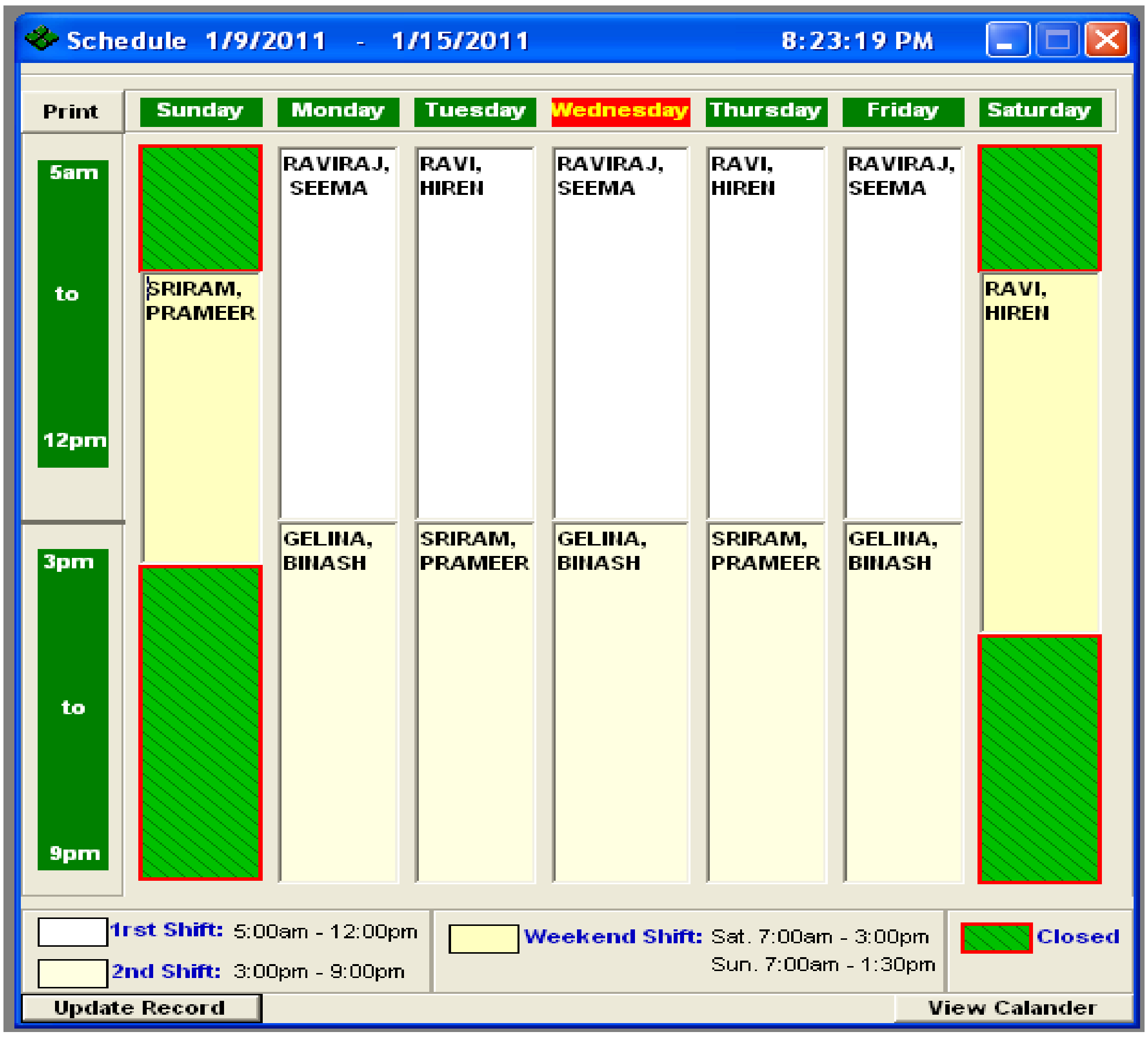
Buttons

Recipts Screen:



Date:	1/12/2011	Old Balance:	-175.00Rs	Payment for:	1 month gym
Member ID:	1	New charge:	175.00Rs	How Payed:	cash
Name:	GANGULY, RITU	Amount Rec:	0.00Rs	Check #:	
Gym Ex:	5/30/2011	New Balance:	0.00Rs	Previous due:	6/28/2011
Tan Ex:				Next due:	7/28/2011
				Received by:	sriram

Schedule Screen:



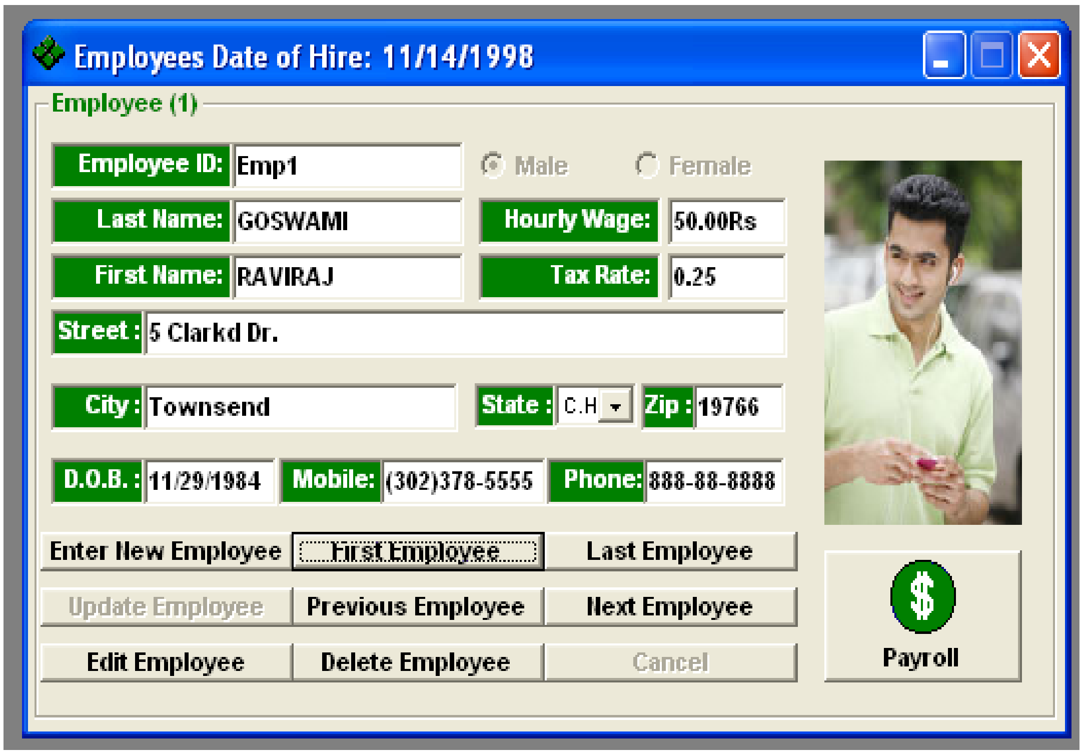
Schedule 1/9/2011 - 1/15/2011 8:23:19 PM

Print	Sunday	Monday	Tuesday	Wednesday	Thursday	Friday	Saturday
5am to 12pm	🟩 SRIRAM, PRAMEER	RAVIRAJ, SEEMA	RAVI, HIREN	RAVIRAJ, SEEMA	RAVI, HIREN	RAVIRAJ, SEEMA	🟩 RAVI, HIREN
3pm to 9pm	🟩	GELINA, BINASH	SRIRAM, PRAMEER	GELINA, BINASH	SRIRAM, PRAMEER	GELINA, BINASH	🟩

□ 1rst Shift: 5:00am - 12:00pm
□ 2nd Shift: 3:00pm - 9:00pm
□ Weekend Shift: Sat. 7:00am - 3:00pm
Sun. 7:00am - 1:30pm
🟩 Closed

Update Record | View Calander

Employee Screen:

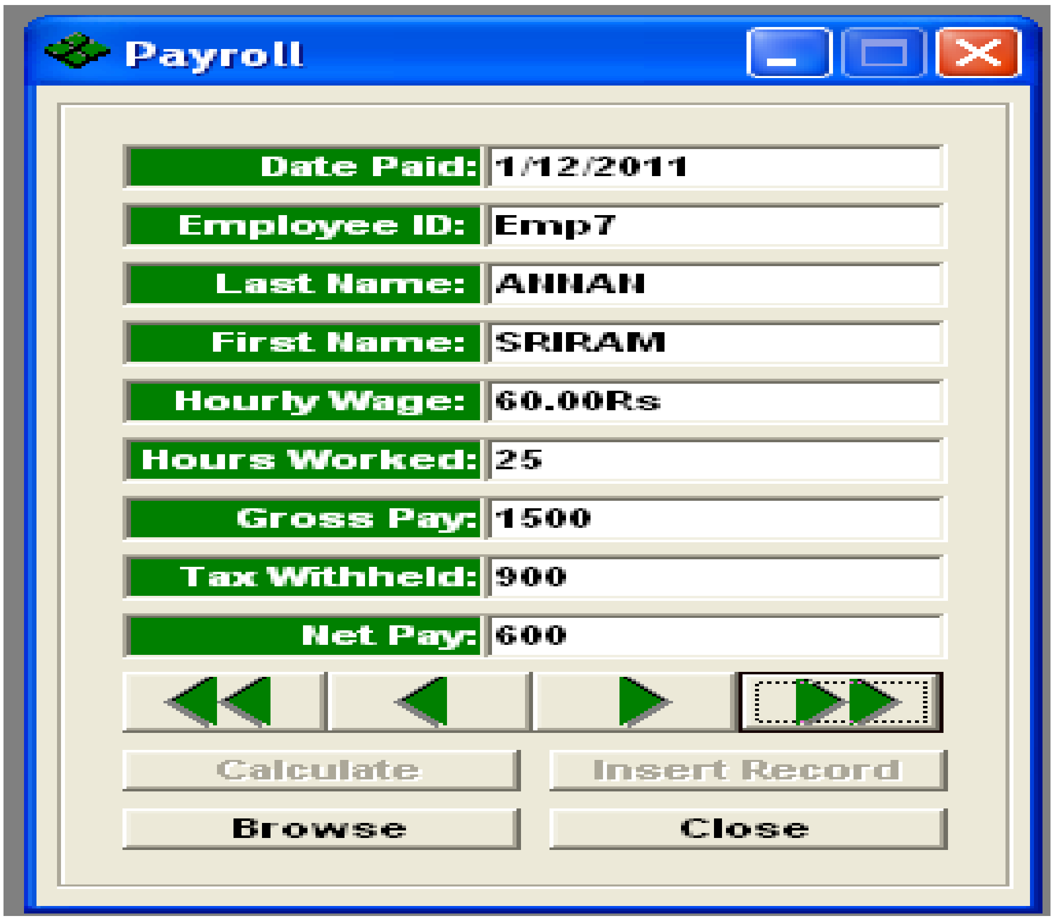


Employees Date of Hire: 11/14/1998

Employee (1)

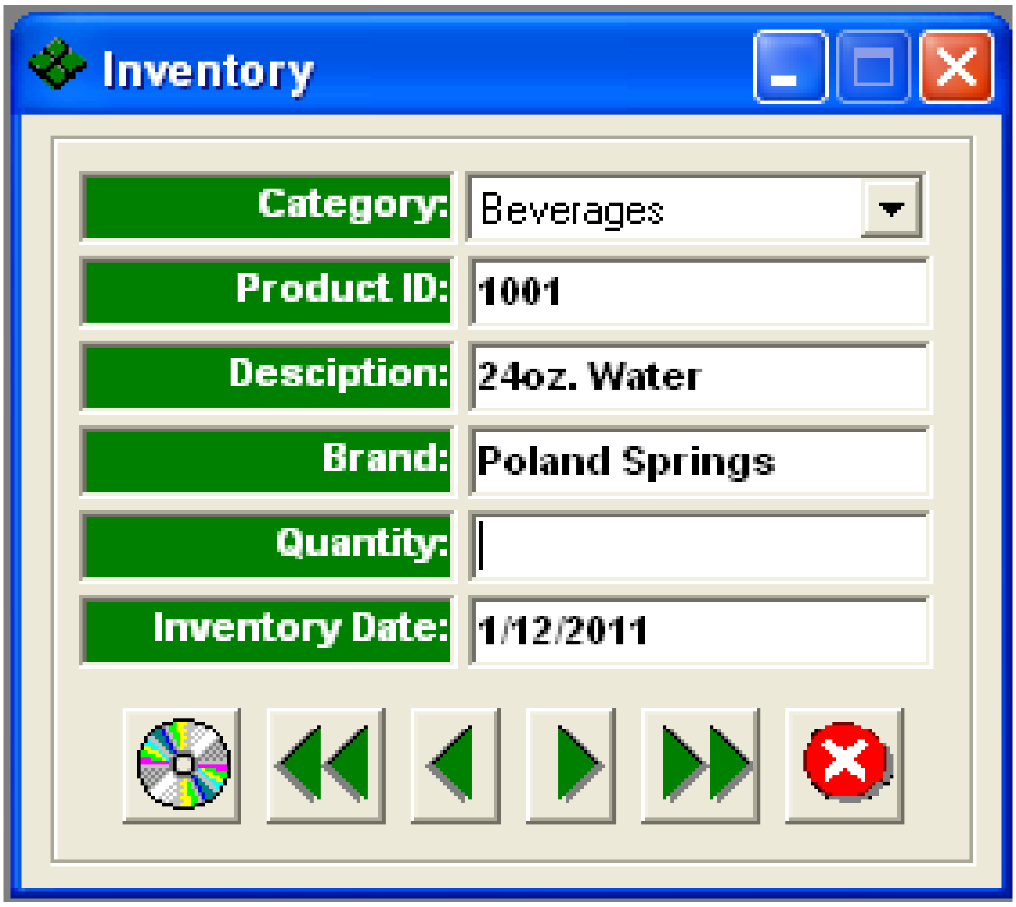
Employee ID:	Emp1	● Male	○ Female
Last Name:	GOSWAMI	Hourly Wage:	50.00Rs
First Name:	RAVIRAJ	Tax Rate:	0.25
Street :	5 Clarkd Dr.		
City :	Townsend	State : C.H	Zip : 19766
D.O.B. :	11/29/1984	Mobile: (302)378-5555	Phone: 888-88-8888

Payroll Screen:



Date Paid:	1/12/2011
Employee ID:	Emp7
Last Name:	ANNAN
First Name:	SRIRAM
Hourly Wage:	60.00Rs
Hours Worked:	25
Gross Pay:	1500
Tax Withheld:	900
Net Pay:	600

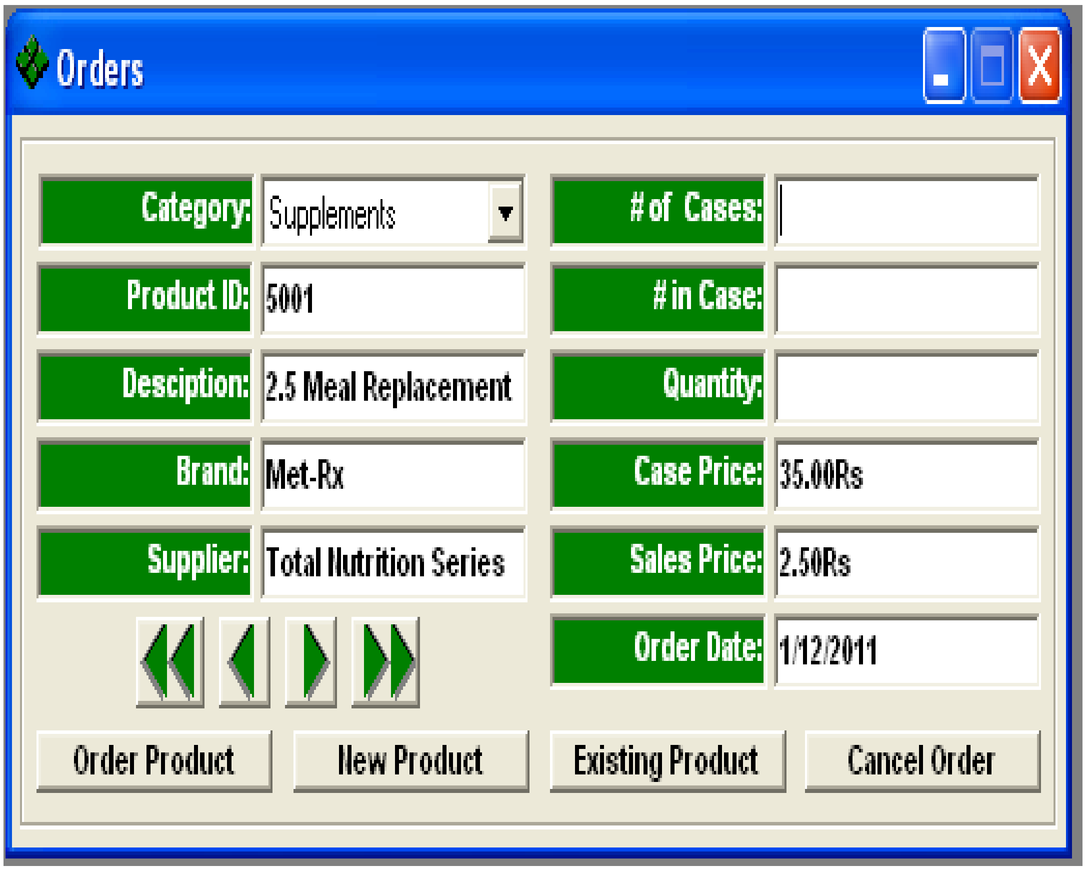
Inventory Screen:



Inventory

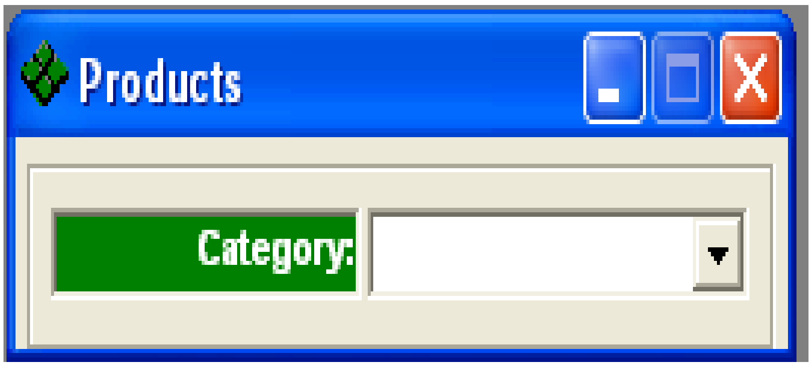
Category:	Beverages
Product ID:	1001
Desciption:	24oz. Water
Brand:	Poland Springs
Quantity:	
Inventory Date:	1/12/2011

Orders Screen:

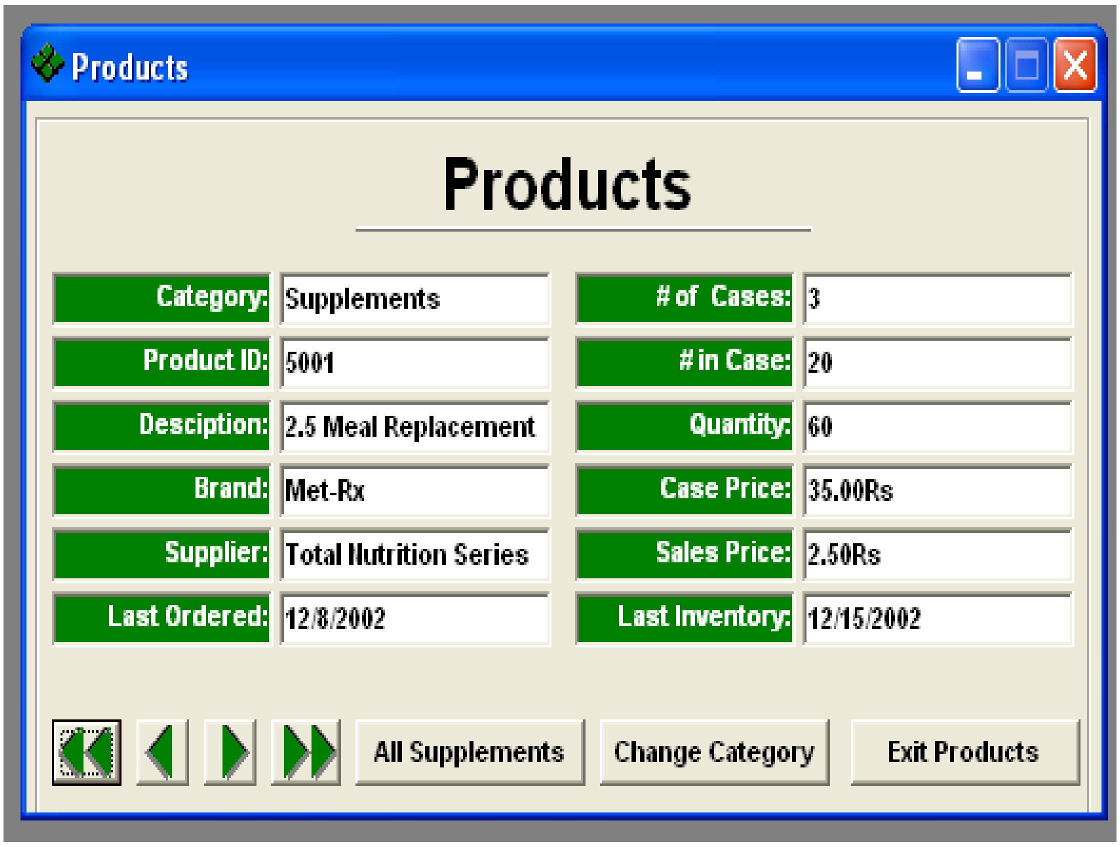


Category:	Supplements	# of Cases:	
Product ID:	5001	# in Case:	
Desciption:	2.5 Meal Replacement	Quantity:	
Brand:	Met-Rx	Case Price:	35.00Rs
Supplier:	Total Nutrition Series	Sales Price:	2.50Rs
		Order Date:	1/12/2011

Products Selection Screen:



Products Screen:

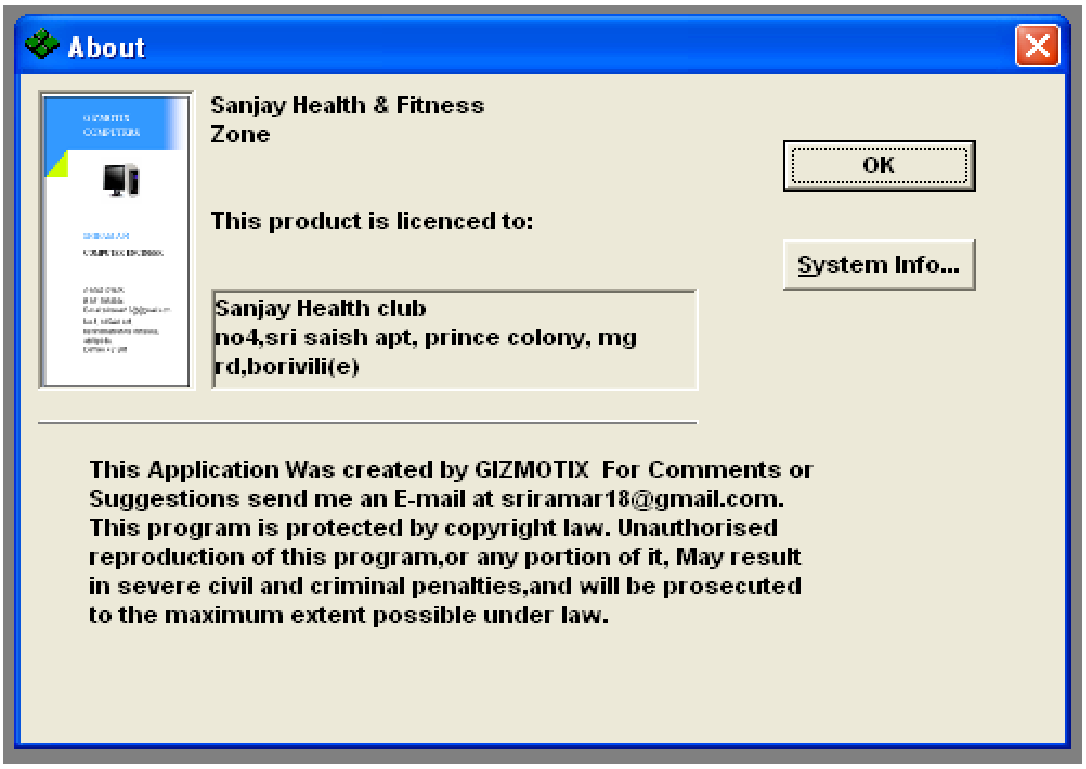


Products

Products

Category:	Supplements	# of Cases:	3
Product ID:	5001	# in Case:	20
Desciption:	2.5 Meal Replacement	Quantity:	60
Brand:	Met-Rx	Case Price:	35.00Rs
Supplier:	Total Nutrition Series	Sales Price:	2.50Rs
Last Ordered:	12/8/2002	Last Inventory:	12/15/2002

About Screen:



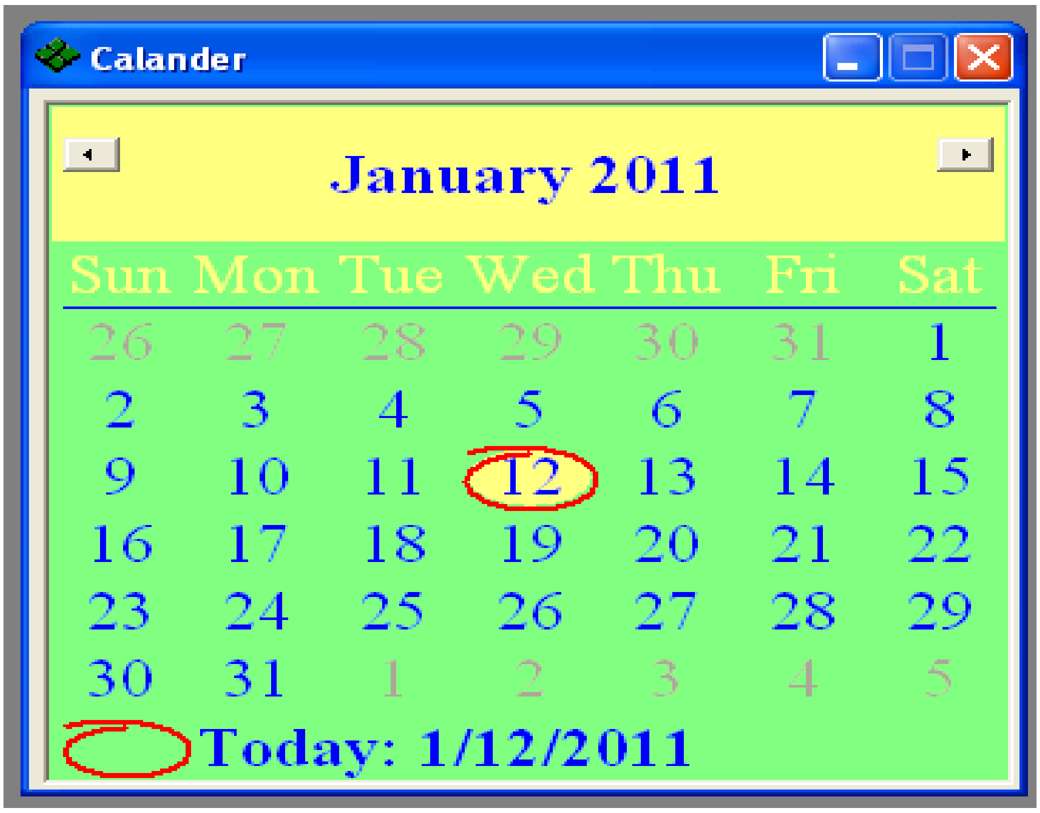
Sanjay Health & Fitness Zone

This product is licenced to:

Sanjay Health club
no4,sri saish apt, prince colony, mg rd,borivili(e)

This Application Was created by GIZMOTIX For Comments or Suggestions send me an E-mail at sriramar18@gmail.com. This program is protected by copyright law. Unauthorised reproduction of this program,or any portion of it, May result in severe civil and criminal penalties,and will be prosecuted to the maximum extent possible under law.

Calander:



Sun	Mon	Tue	Wed	Thu	Fri	Sat
26	27	28	29	30	31	1
2	3	4	5	6	7	8
9	10	11	12	13	14	15
16	17	18	19	20	21	22
23	24	25	26	27	28	29
30	31	1	2	3	4	5

METHODOLOGY USED FOR TESTING

Testing

System testing is designed to uncover the weaknesses that were not found in earlier test. In the testing phase, the program is executed with the explicit intention of finding errors. This includes forced system failures and validation of the system, as its user in the operational environment will implement it. For this purpose test cases are developed.

When a new system replaces the old one, such as in the present case, the organization can extract data from the old system to test them on the new. Such data usually exist in sufficient volume to provide sample listings and they can create a realistic environment that ensures eventual system success. Regardless of the source of test data, the programmers and analyst will eventually conduct four different types of tests.

Integration Testing

The integration is the next important concept that highlights in the testing scenario. Integration testing can be performed in different strategies. One of them is the Big Bang testing in which one could first test all of a system's modules separately and then whole systems at once. But here we proceed abruptly from the module testing and the integration testing disappears. Another alternative is the Incremental Testing.

With the Incremental testing there are many advantages. We can start the integration as soon as reasonable subsets of modules have been developed. It is easier to localize errors incrementally. The partial aggressions of modules often constitute important subsystems that can have autonomy with these testing. The need for stubs and drivers can be reduced.

There are two approaches to the Incremental Testing. They include Bottom-up and Top-down aggregations. The former means starting aggregation and testing from leaves of the module charts. The latter means starting from the top-level modules and substitute for higher-level modules. In our project we have used the top-down approach of incremental testing.

Top-down integration is an incremental approach to the construction of programs structure. Modules are integrated by moving downward through the control hierarchy, beginning with the main control module that is the basic connectivity module in our project. Test is done on each module.

The top down integration strategy verifies major control or decision points. In the beginning of the integration phase dummy frames were selected as stubs to ensure that the data flow occurred through the correct hierarchical structure. Later the actual module replaces these stubs.

System Testing

The system testing deals with the process of testing the system as a whole. This is done after the integration process. Moving through each module from top to bottom tests the entire system. The verification and validation process are then carried out. The errors that occur the testing phase are eliminated and a well functioning system is developed.

Test case design focuses on a set of techniques, which meets all testing objectives, which are mentioned below.

1. Testing is a process of executing a program with the intent of finding an error.
2. A successful test is one that uncovers an as yet undiscovered error.

Testing demonstrates that software functions work according to specifications. In addition data collected from testing provides a good indication of software reliability and some indication of software quality as a whole.

Testing results in the deduction in the numbers of errors. Critical modules are tested as early as possible .The following tests have been carried out after developing the system.

Various Testing Methods

Unit testing focuses verification efforts on the smallest unit of the software design, the module. This is also known as Module Testing. The modules are tested separately. This testing is carried out during programming stage itself.

Validation Testing

Validation testing can be defined in many ways but a simple definition is that validation succeeds when the software functions in a manner that can be reasonably expected by the users .After validation test has been conducted one of the two possible conditions exists

1. The function or the performance characteristics confirm to specification and are accepted.

2. A deviation from specification is uncovered and a deficiency list is created.

Output Testing

After performing the validation testing the next step is output testing of the proposed system since no system is useful if it does not produce the required output in the specific format. The outputs generated or displayed by the system under consideration are tested by asking the users about the formats required by them.

User Acceptance Testing

User acceptance of a system is a key factor for the success of any system. The system under consideration is tested for user acceptance by constantly keeping in touch with prospective system users at the time development and making changes whenever required.

Quality Assurance Methodologies

Quality assurance is a planned and systematic of all actions necessary to provide adequate confidence that the item or product confirms to established technical requirements. The purpose of software quality assurance group is to provide assurances that the procedures, tools and techniques used during product development and modification and adequate to provide desired level of confidence in the work products.

Often, software quality assurance personnel are organizationally distinct from software development group. Preparation of a Software Quality Assurance

Plan for each software products is primary responsibility of software quality assurance group.

Quality assurances personnel are sometimes are charge of arrangements for walkthroughs, inspections and major milestones reviews. In addition, quality assurance personnel often conduct the project post mortem, write project legacy document and provide long term retention of the project records.

Typically the quality assurance group will work with the development group to derive Source Code Test Plan. A test plan for the source code specifies the objectives of testing; the test plan for source code specifies the objectives of testing, the test completion criteria, the system integration plan, and methods to be used on particular test inputs expected outcomes.

There are four types of tests that the source code must satisfy: function tests, performance tests, stress test and structural test.

Functional test cases specify typical operating conditions, typical input values and typical expected values. Function tests are also tests that are performed on the inside and just beyond the functional boundaries. Examples of functional test include testing a real-valued square route routine with small positive numbers, zero and negative numbers; or testing a matrix version of the inversion routine on a one-by-one matrix and a singular matrix.

Performance tests are also designed to verify response time under varying loads, percent execution time spent in various segments of the program, throughput, primary and secondary memory utilization and traffic rates on the data channels and communication links.

Stress tests are designed in such a way that to overload a system in various ways. Examples of stress tests include attempting to sign on more than the maximum number allowed terminal, processing more than the allowed number of identifiers or static levels or disconnecting a communication link.

Structure test are concerned with examining of the internal processing logic of the software system. The particular routines called and the logic paths traversed through the routines are object of interest.

System verification and validation

System verification and validation is done to check the quality of the software in simulated and live environment. A number of different transactions are used to perform verification. Validation is the process of demonstrating that the implemented software does satisfy the system requirements. One aspect of software validation is to statistically analyze the program without resorting to actual execution. The system validation done in such-a-way that the system response time will not cause any hardship to the user.

White Box Testing

White box testing is a test case design method that uses the control structure of the procedural design to derive test cases. Using white box testing methods, we can derive test cases that

- Guarantee that all independent paths within a module have been exercised at least once.

- Exercise all logical decisions on their true and false sides
- Execute all loops at their boundaries and within their operational bounds
- Exercise internal data structures to ensure their validity.

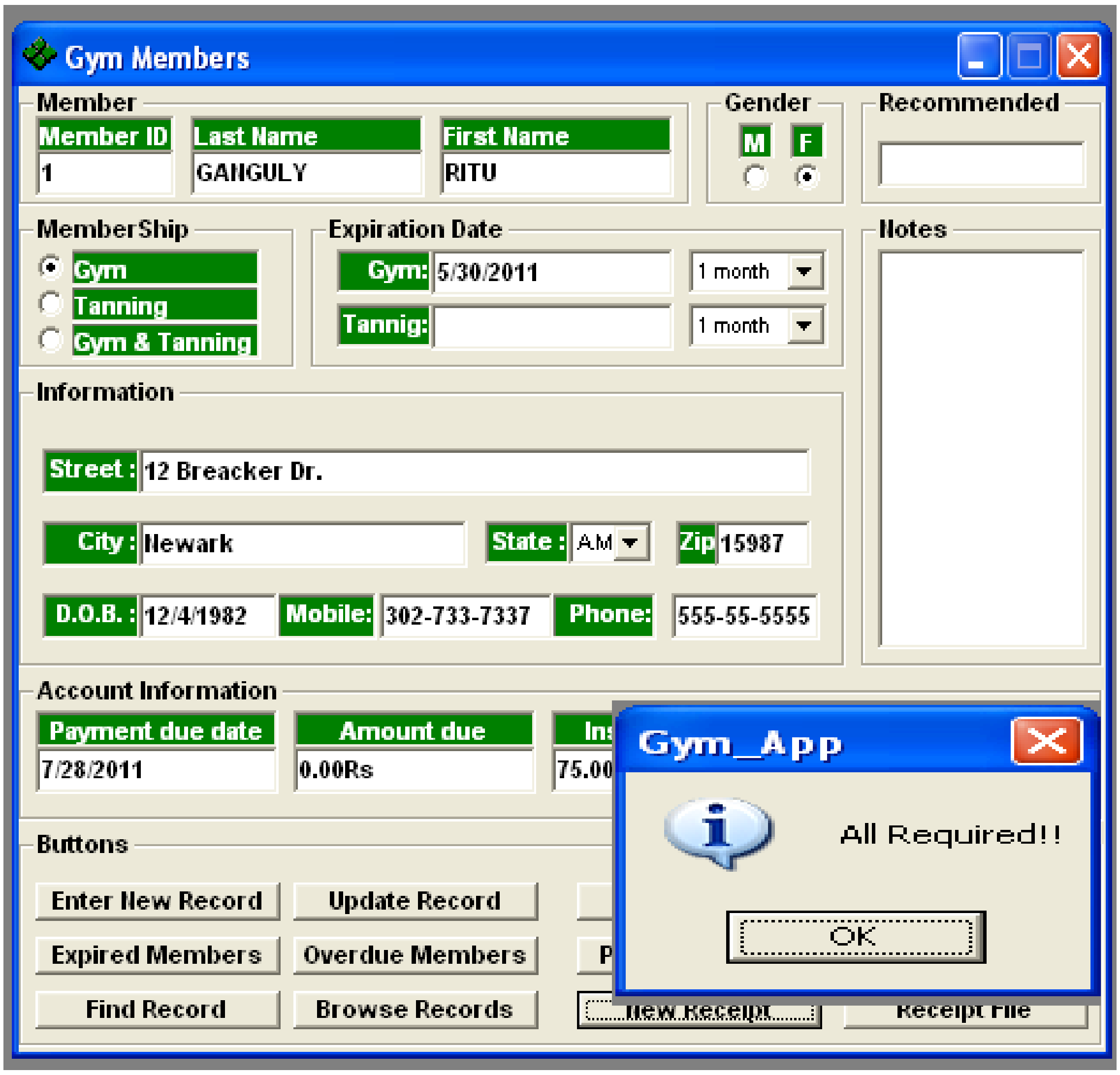
Black Box Testing

Black box testing methods focus on the functional requirements if the software. That is, black box testing enables us to derive sets of input conditions that will fully exercise all functional requirements of the program.

Black box testing attempts to find errors in following categories:

- Incorrect or missing functions
- Interface errors
- Errors in data structures or external database access
- Performance errors

Initialization and termination errors



Gym Members

Member

Member ID	Last Name	First Name
1	GANGULY	RITU

Gender: M ( ) F (●)

Recommended

MemberShip: (●) Gym ( ) Tanning ( ) Gym & Tanning

Expiration Date

Gym: 5/30/2011 — 1 month

Tannig: — 1 month

Notes

Information

Street : 12 Breacker Dr.

City : Newark State : AM Zip 15987

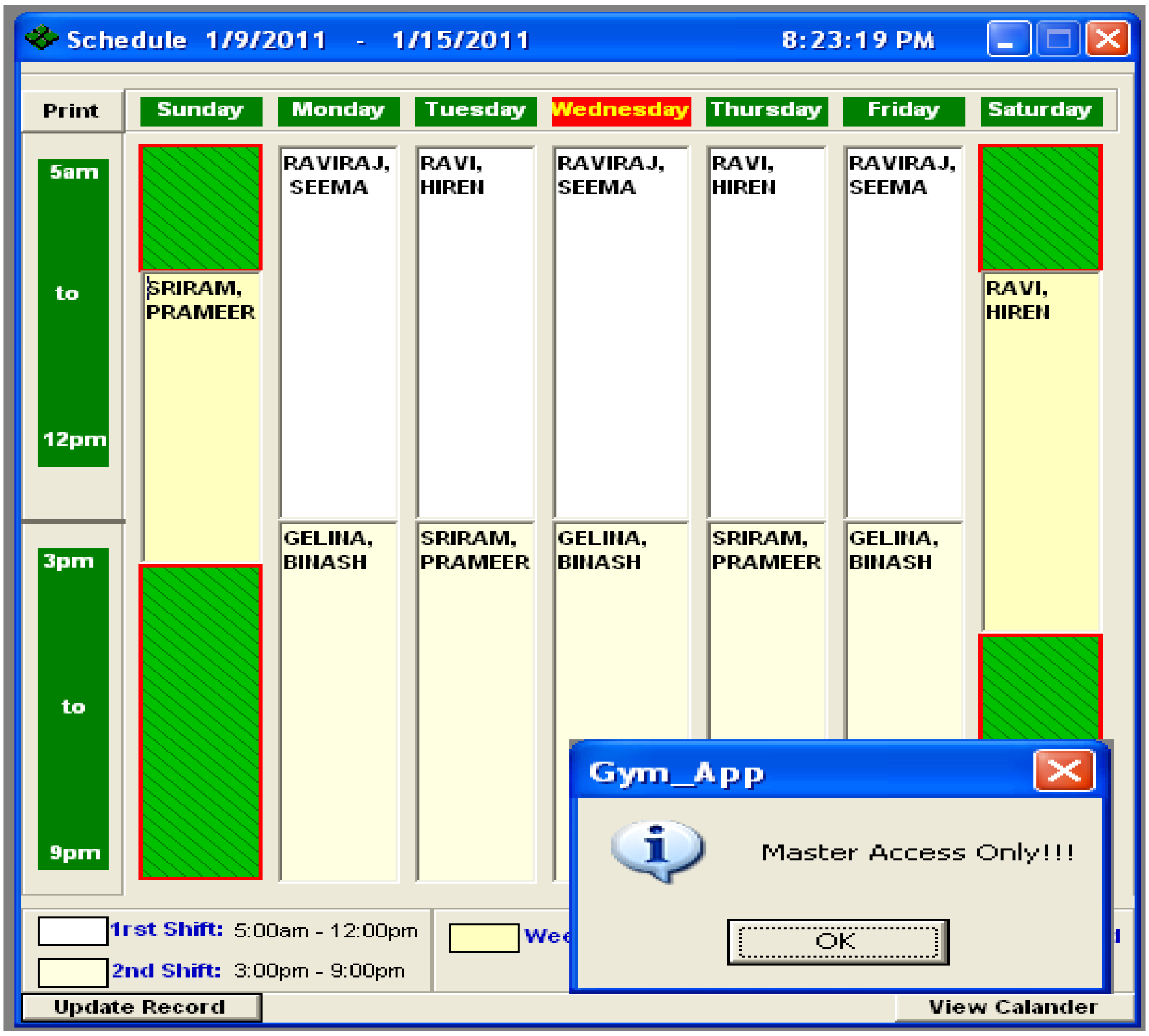
D.O.B. : 12/4/1982 Mobile: 302-733-7337 Phone: 555-55-5555

Account Information

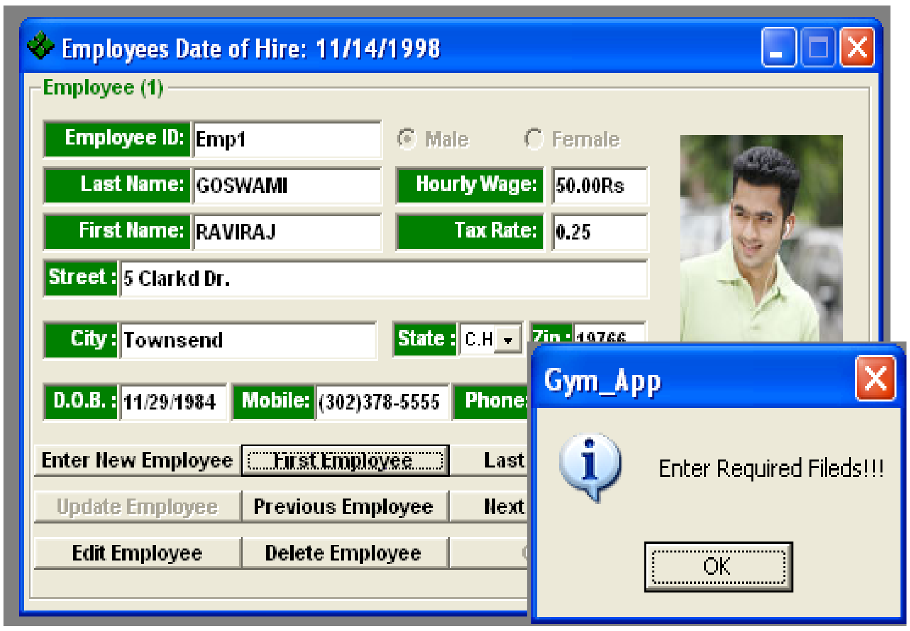
Payment due date	Amount due	Ins
7/28/2011	0.00Rs	75.00

Gym_App

All Required!!



Master Access Only!!!



Employees Date of Hire: 11/14/1998

Employee (1)

Employee ID: Emp1

Male Female

Last Name: GOSWAMI

Hourly Wage: 50.00Rs

First Name: RAVIRAJ

Tax Rate: 0.25

Street : 5 Clarkd Dr.

City : Townsend

State : C.H

Zip : 19766

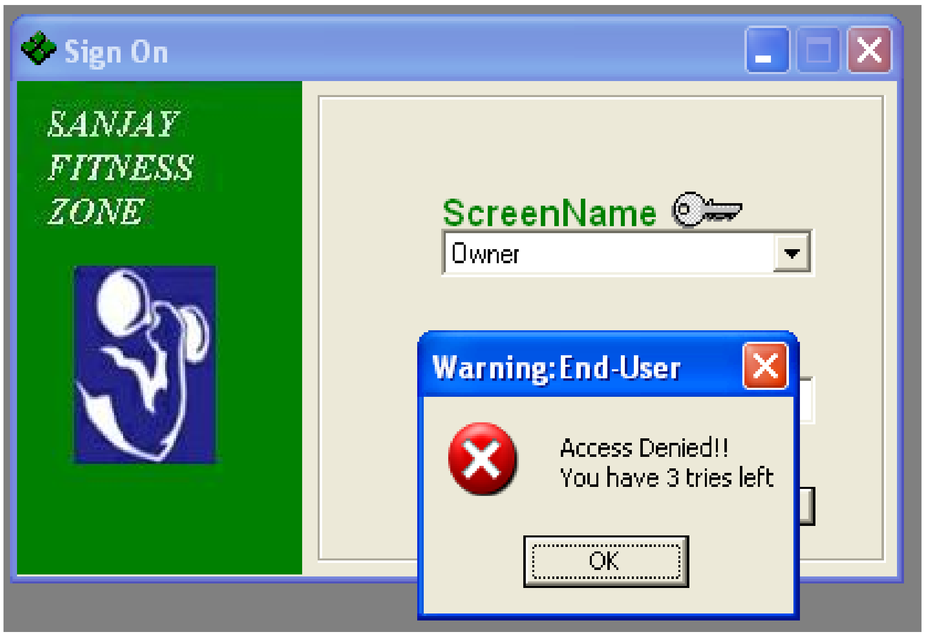
D.O.B. : 11/29/1984

Mobile: (302)378-5555

Phone:

Gym_App

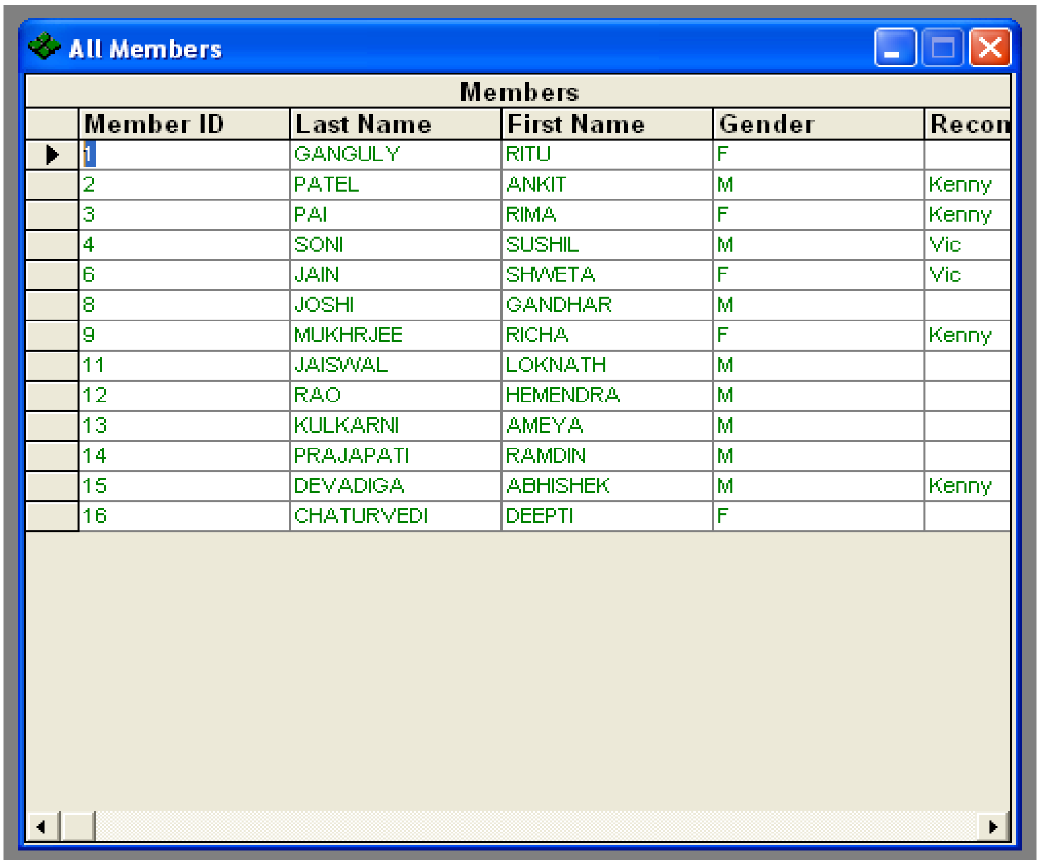
Enter Required Fileds!!!



Access Denied!!
You have 3 tries left

REPORTS

Members Report:

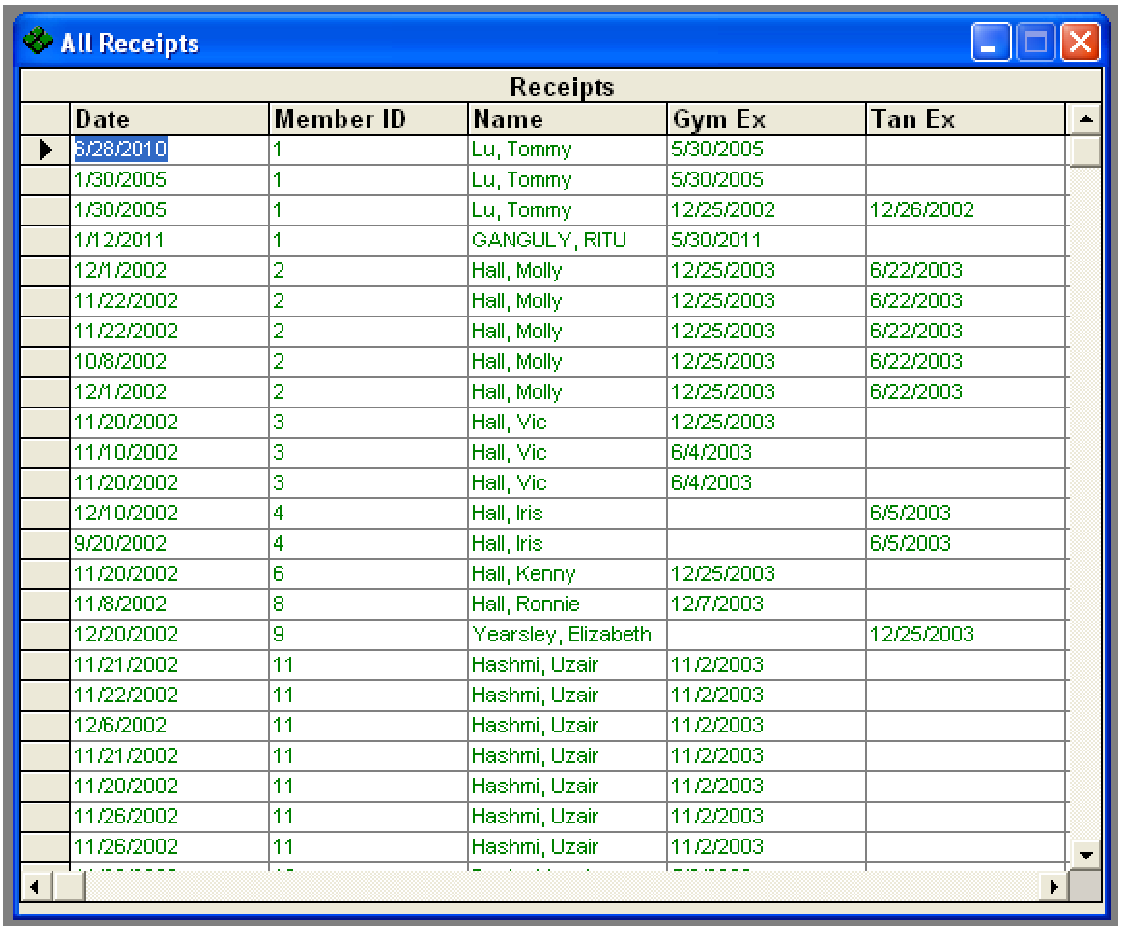


All Members

Members

	Member ID	Last Name	First Name	Gender	Recon
▶	1	GANGULY	RITU	F	
	2	PATEL	ANKIT	M	Kenny
	3	PAI	RIMA	F	Kenny
	4	SONI	SUSHIL	M	Vic
	6	JAIN	SHWETA	F	Vic
	8	JOSHI	GANDHAR	M	
	9	MUKHRJEE	RICHA	F	Kenny
	11	JAISWAL	LOKNATH	M	
	12	RAO	HEMENDRA	M	
	13	KULKARNI	AMEYA	M	
	14	PRAJAPATI	RAMDIN	M	
	15	DEVADIGA	ABHISHEK	M	Kenny
	16	CHATURVEDI	DEEPTI	F	

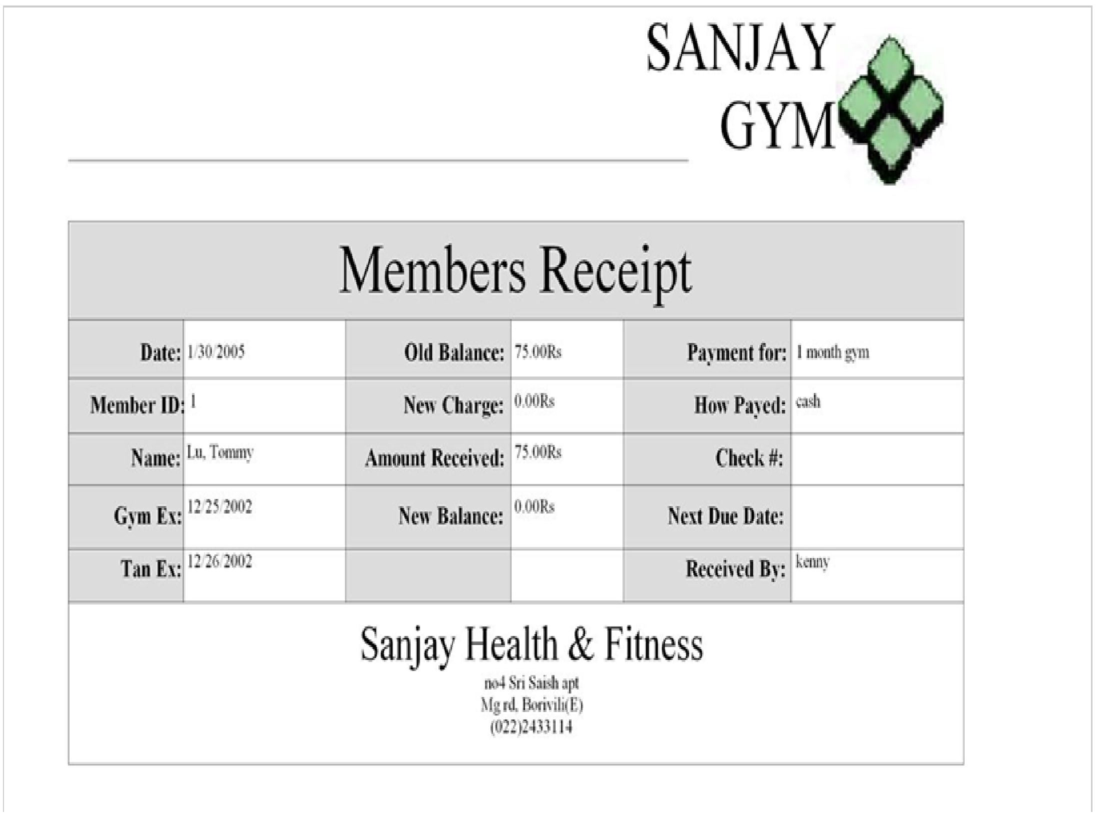
Recipts Report:



Receipts

Date	Member ID	Name	Gym Ex	Tan Ex
6/28/2010	1	Lu, Tommy	5/30/2005	
1/30/2005	1	Lu, Tommy	5/30/2005	
1/30/2005	1	Lu, Tommy	12/25/2002	12/26/2002
1/12/2011	1	GANGULY, RITU	5/30/2011	
12/1/2002	2	Hall, Molly	12/25/2003	6/22/2003
11/22/2002	2	Hall, Molly	12/25/2003	6/22/2003
11/22/2002	2	Hall, Molly	12/25/2003	6/22/2003
10/8/2002	2	Hall, Molly	12/25/2003	6/22/2003
12/1/2002	2	Hall, Molly	12/25/2003	6/22/2003
11/20/2002	3	Hall, Vic	12/25/2003	
11/10/2002	3	Hall, Vic	6/4/2003	
11/20/2002	3	Hall, Vic	6/4/2003	
12/10/2002	4	Hall, Iris		6/5/2003
9/20/2002	4	Hall, Iris		6/5/2003
11/20/2002	6	Hall, Kenny	12/25/2003	
11/8/2002	8	Hall, Ronnie	12/7/2003	
12/20/2002	9	Yearsley, Elizabeth		12/25/2003
11/21/2002	11	Hashmi, Uzair	11/2/2003	
11/22/2002	11	Hashmi, Uzair	11/2/2003	
12/6/2002	11	Hashmi, Uzair	11/2/2003	
11/21/2002	11	Hashmi, Uzair	11/2/2003	
11/20/2002	11	Hashmi, Uzair	11/2/2003	
11/26/2002	11	Hashmi, Uzair	11/2/2003	
11/26/2002	11	Hashmi, Uzair	11/2/2003	

Recipt:



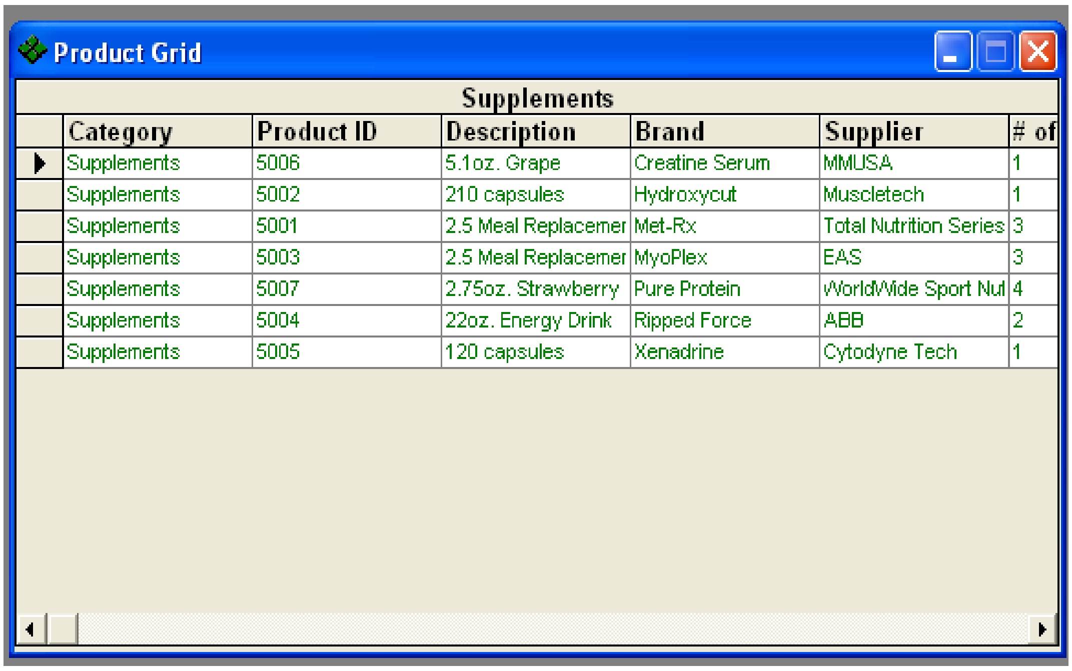
SANJAY GYM

Members Receipt

Date:	1/30/2005	**Old Balance:**	75.00Rs	**Payment for:**	1 month gym
Member ID:	1	**New Charge:**	0.00Rs	**How Payed:**	cash
Name:	Lu, Tommy	**Amount Received:**	75.00Rs	**Check #:**	
Gym Ex:	12/25/2002	**New Balance:**	0.00Rs	**Next Due Date:**	
Tan Ex:	12/26/2002			**Received By:**	kenny

Sanjay Health & Fitness
no4 Sri Saish apt
Mg rd, Borivili(E)
(022)2433114

Products report:



Supplements

	Category	Product ID	Description	Brand	Supplier	# of
▶	Supplements	5006	5.1oz. Grape	Creatine Serum	MMUSA	1
	Supplements	5002	210 capsules	Hydroxycut	Muscletech	1
	Supplements	5001	2.5 Meal Replacemer	Met-Rx	Total Nutrition Series	3
	Supplements	5003	2.5 Meal Replacemer	MyoPlex	EAS	3
	Supplements	5007	2.75oz. Strawberry	Pure Protein	WorldWide Sport Nut	4
	Supplements	5004	22oz. Energy Drink	Ripped Force	ABB	2
	Supplements	5005	120 capsules	Xenadrine	Cytodyne Tech	1

Coding

I. Splash Screen

```
Option Explicit
Dim mintCount As Integer, mintPause As Integer

Private Sub Form_Load()
Dim X(2) As pointapi
Dim lRegion As Long
Dim lRegion1 As Long
Dim lRegion2 As Long
Dim lResult As Long

   Screen.MousePointer = vbHourglass
   frmSplash.Width = 500 * Screen.TwipsPerPixelX
   frmSplash.Height = 500 * Screen.TwipsPerPixelY

   lRegion = CreatePolygonRgn(X(0), 3, alternate)
   lRegion1 = CreatePolygonRgn(X(0), 3, alternate)
   lRegion2 = CreateRoundRectRgn(0, 0, 480, 213, 50, 50)
   lResult = CombineRgn(lRegion, lRegion1, lRegion2, rgn_or)
   DeleteObject lRegion1
   DeleteObject lRegion2
   lResult = SetWindowRgn(frmSplash.hWnd, lRegion, True)
End Sub

Private Sub Form_Unload(Cancel As Integer)
   Screen.MousePointer = vbDefault
End Sub

Private Sub tmrCount_Timer()
   mintPause = mintPause + 1

   If mintCount < 50 Then
      mintCount = mintCount + 1
      lblCount.Caption = "(" & mintCount & "%)..."
       frmSplash.Refresh
```

```
    ElseIf mintCount < 100 Then
        mintCount = mintCount + 2
        lblCount.Caption = "(" & mintCount & "%)..."
        frmSplash.Refresh
    End If

    If mintPause = 101 Then
        lblCount.Caption = "App..."
        lblInform.Caption = "Starting"
    ElseIf mintPause > 150 Then
        Unload Me
        frmPassword.Show
        mdiDtcc.Show
    End If
End Sub
```

II. Password

```
Option Explicit
Dim mintctr As Integer
Dim mrstLogin As Recordset
Dim pdbEnter As Database

Private Sub cboName_LostFocus()
   txtPassword.Text = ""
End Sub

Private Sub cmdOn_Click()
Dim flag As Boolean
Dim xText
flag = False

   If txtPassword.Text = "" Then
      MsgBox "Please Enter Password", vbOKOnly + vbCritical, _
      "Warning:End-User"
      txtPassword.SetFocus
      flag = True
   End If

   If cboName.ListIndex = 0 Then
      If cboName = mrstLogin![fldScreen] And txtPassword =
mrstLogin![fldPass] Then
         mdiDtcc.tbrChoices.Visible = True
         mdiDtcc.mnuMembers = True
         mdiDtcc.mnuEmp = True
         mdiDtcc.mnuInv = True
         mdiDtcc.mnuChUser.Enabled = True
         Unload Me
         mdiDtcc.mnuOn.Visible = False
         flag = True
         gblnPriv = True
         mdiDtcc.ToolCenter
         frmTools.Show
         frmTools.stb1.Tab = 0
```

```
        End If
    Else
        mrstLogin.MoveNext
        If cboName = mrstLogin![fldScreen] And txtPassword =
mrstLogin![fldPass] Then
            mdiDtcc.tbrChoices.Visible = True
            mdiDtcc.mnuChUser.Enabled = True
            mdiDtcc.mnuMembers = True
            mdiDtcc.mnuEmp = True
            mdiDtcc.mnuInv = True
            Unload Me
            mdiDtcc.mnuOn.Visible = False
            flag = True
            gblnPriv = False
        End If
        mrstLogin.MoveFirst
    End If

    If flag = False Then
        mintctr = mintctr + 1
        If mintctr = 4 Then
            End
        Else
            xText = "You have" + Str(4 - mintctr) + " tries left"
            If mintctr = 3 Then
                xText = "This is your last chance!!"
            End If
            MsgBox "Access Denied!!" & vbCrLf & _
            xText, vbOKOnly + vbCritical, "Warning:End-User"
            txtPassword.Text = ""
        End If
    End If
End Sub

Private Sub Form_Load()
    Set pdbEnter = OpenDatabase(App.Path & "\GymMembers.mdb")
    Set mrstLogin = pdbEnter.OpenRecordset("tblPass")
    mdiDtcc.tbrChoices.Visible = False
    mdiDtcc.mnuScreen.Enabled = False
    mdiDtcc.mnuChUser.Enabled = False
    mdiDtcc.mnuMembers = False
    mdiDtcc.mnuEmp = False
    mdiDtcc.mnuInv = False
    cboName = mrstLogin![fldScreen]
    mintctr = 0
End Sub

Private Sub Form_Unload(Cancel As Integer)
    mdiDtcc.mnuScreen.Enabled = True
End Sub
```

III. MDI

```
Option Explicit
Private Sub MDIForm_Load()
   frmPassword.Top = mdiDtcc.ScaleHeight / 1.5
   frmPassword.Left = mdiDtcc.ScaleWidth / 3.3
End Sub

Private Sub mnuAbout_Click()
   frmAbout.Top = mdiDtcc.ScaleHeight / 5
   frmAbout.Left = mdiDtcc.ScaleWidth / 4
   frmAbout.Show
End Sub

Private Sub mnuCas_Click()
   mdiDtcc.Arrange vbCascade
End Sub

Private Sub mnuChUser_Click()
   Dim pstrUser As String
   pstrUser = MsgBox("Change User?", vbYesNo + vbQuestion)

   If pstrUser = vbYes Then
      CloseForms
      mnuOn.Visible = True
      frmPassword.Show
      frmPassword.Top = mdiDtcc.ScaleHeight / 4
      frmPassword.Left = mdiDtcc.ScaleWidth / 4
   End If
End Sub

Private Sub mnuEmp_Click()
    ToolCenter
    frmTools.Show
    frmTools.stb1.Tab = 1
End Sub

Private Sub mnuExit_Click()
    Unload Me
End Sub

Private Sub mnuIcons_Click()
    mdiDtcc.Arrange vbArrangeIcons
End Sub

Private Sub mnuInv_Click()
    ToolCenter
    frmTools.Show
    frmTools.stb1.Tab = 2
End Sub

Private Sub mnuMembers_Click()
    ToolCenter
    frmTools.Show
    frmTools.stb1.Tab = 0
End Sub

Private Sub mnuScreen_Click()
```

```
    frmPassword.Show
    frmPassword.Top = mdiDtcc.ScaleHeight / 4
    frmPassword.Left = mdiDtcc.ScaleWidth / 5
End Sub

Private Sub mnuThor_Click()
    mdiDtcc.Arrange vbTileHorizontal
End Sub

Private Sub mnuTvert_Click()
    mdiDtcc.Arrange vbTileVertical
End Sub

Private Sub tbrChoices_ButtonClick(ByVal Button As
MSComctlLib.Button)
    Select Case Button.Key
        Case "Members"
            ToolCenter
            frmTools.Show
            frmTools.stb1.Tab = 0
        Case "Employees"
            ToolCenter
            frmTools.Show
            frmTools.stb1.Tab = 1
        Case "Inventory"
            ToolCenter
            frmTools.Show
            frmTools.stb1.Tab = 2
    End Select
End Sub

Public Sub CloseForms()
    Unload frmAbout
    Unload frmAllPro
    Unload frmCal
    Unload frmGym
    Unload frmInventory
    Unload frmNewEmp
    Unload frmOrders
    Unload frmPayroll
    Unload frmProducts
    Unload frmReceipt
    Unload frmSchedule
    Unload frmTools
End Sub

Public Sub ToolCenter()
    If frmTools.WindowState <> vbMinimized Then
        frmTools.Top = mdiDtcc.ScaleHeight / 4
        frmTools.Left = mdiDtcc.ScaleWidth / 3
    End If
```

IV. <u>Members</u>

```
Option Explicit
Dim flag As Integer
Dim mblnBrow As Boolean
Dim mintClear As Integer
Dim mintFind As Integer
Dim mblnCheck As Boolean
Dim mdatGExp As Date
Dim mdatTExp As Date
Dim mdatOD As Date
Dim mintOD As Integer
Dim mblnExpOD As Boolean

Private Sub cmdBrowse_Click()
   If mblnBrow = True Then Form_Load
End Sub

Private Sub cmdExp_Click()
   If mblnExpOD = True Then Form_Load
   mrstGym.MoveFirst
   Do Until mrstGym.EOF
      If mrstGym!fldGExp = True Or mrstGym!fldTExp = True Then
         mblnBrow = True
         Set mrstGym = pdbMembers.OpenRecordset("SELECT * FROM
tblMembers WHERE fldGExp = true  or fldTExp=true ORDER BY
fldMemberID")
         ShowRecord
         mintFind = 0
         mblnExpOD = True
      Exit Sub
      Else
         mrstGym.MoveNext
      End If
   Loop
   MsgBox "No Expired Members!!!", vbOKOnly + vbInformation
```

```
    Form_Load
End Sub

Private Sub cmdFind_Click()
    mintClear = 1
    mintFind = 1
    txtId.SetFocus
    ClearRecord
    Set mrstGym = pdbMembers.OpenRecordset("SELECT * FROM
tblMembers ORDER BY fldMemberID")
End Sub

Private Sub cmdOD_Click()
    If mblnExpOD = True Then Form_Load
     mrstGym.MoveFirst
    Do Until mrstGym.EOF
        If mrstGym!fldOD = True Then
            mblnBrow = True
            Set mrstGym = pdbMembers.OpenRecordset("SELECT * FROM
tblMembers WHERE fldOD = true   ORDER BY fldMemberID")
            ShowRecord
            mintFind = 0
            mblnExpOD = True
        Exit Sub
        Else
            mrstGym.MoveNext
        End If
    Loop
    MsgBox "No Overdue Members!!!", vbOKOnly + vbInformation
```

```
   Form_Load
End Sub

Private Sub cmdRecFile_Click()
   gblnRec = True
   If frmReceipt.WindowState <> vbMinimized Then
      frmReceipt.Top = mdiDtcc.ScaleHeight / 5
      frmReceipt.Left = mdiDtcc.ScaleWidth / 8
   End If
   frmReceipt.Show
End Sub

Private Sub cmdUpdate_Click()
   Checktxt
   If mblnCheck = False Then
      If txtPayduedate = IsDate And (txtAmountdue And textinstall) =
iscurrency Then
      WriteRecord
      mrstGym.Update
      Else
      MsgBox "Enter valid Data!!", vbOKOnly + vbInformation
      End If
   Else
      MsgBox "All Required!!", vbOKOnly + vbInformation
      mblnCheck = False
   End If
   If flag = False Then ShowRecord
End Sub
```

```
Private Sub cmdEnter_Click()
    mrstGym.AddNew
    ClearRecord
    txtLastName.SetFocus
    flag = 1
  End Sub

  Private Sub cmdFirst_Click()
    mrstGym.MoveFirst
    ShowRecord
  End Sub

  Private Sub cmdLast_Click()
    mrstGym.MoveLast
    ShowRecord
  End Sub

  Private Sub cmdNewReceipt_Click()
    gblnRec = False
    If frmReceipt.WindowState <> vbMinimized Then
      frmReceipt.Top = mdiDtcc.ScaleHeight / 5
      frmReceipt.Left = mdiDtcc.ScaleWidth / 8
    End If
    frmReceipt.Show
  End Sub

  Private Sub mnuExit_Click()
    Unload Me
```

```
End Sub

Private Sub cmdNext_Click()
    mrstGym.MoveNext
    If mrstGym.EOF Then mrstGym.MoveLast
    ShowRecord
    'MsgBox "This is Last Record...", vbInformation
End Sub

Private Sub cmdPrev_Click()
    mrstGym.MovePrevious
    If mrstGym.BOF Then mrstGym.MoveFirst
    ShowRecord
    'MsgBox "This is First Record...", vbInformation
 End Sub

Private Sub Form_Load()
    Set pdbMembers = OpenDatabase(App.Path & "\GymMembers.mdb")
    Set mrstGym = pdbMembers.OpenRecordset("SELECT * FROM
tblMembers ORDER BY fldMemberID")
    mblnBrow = False
    cmdFirst_Click
End Sub

Public Sub ShowRecord()
    With mrstGym
        txtId = !fldMemberID
        txtLastName = !fldLastName
        txtFirstName = !fldFirstName
        If !fldGender = "M" Then optM = True
        If !fldGender = "F" Then optF = True
        If !fldRecommend <> "" Then txtRecommed = !fldRecommend Else
txtRecommed = ""
        If !fldMemberShip = "Gym" Then optGym = True
        If !fldMemberShip = "Tanning" Then optTanning = True
        If !fldMemberShip = "Gym & Tanning" Then optGym_Tanning = True
        If !fldGymEx <> "" Then
          mdatGExp = !fldGymEx
          If mdatGExp < Date Then
            .Edit
            !fldGExp = True
            .Update
            lblGE.Visible = True
            txtEx_gym.Width = 975
            txtEx_gym.ForeColor = vbRed
            txtEx_gym = !fldGymEx
          Else
            .Edit
            !fldGExp = False
            .Update
            lblGE.Visible = False
            txtEx_gym.Width = 1695
            txtEx_gym.ForeColor = vbBlack
            txtEx_gym = !fldGymEx
          End If
        Else
          .Edit
          !fldGExp = False
```

```
            .Update
            lblGE.Visible = False
            txtEx_gym.Width = 1695
            txtEx_gym = ""
        End If
        If !fldGymExType <> "" Then cboGym_date = !fldGymExType Else
cboGym_date.ListIndex = -1
        If !fldTanEx <> "" Then
            mdatTExp = !fldTanEx
            If mdatTExp < Date Then
                .Edit
                !fldTExp = True
                .Update
                lblTE.Visible = True
                txtEx_tan.Width = 975
                txtEx_tan.ForeColor = vbRed
                txtEx_tan = !fldTanEx
            Else
                .Edit
                !fldTExp = False
                .Update
                lblTE.Visible = False
                txtEx_tan.Width = 1695
                txtEx_tan.ForeColor = vbBlack
                txtEx_tan = !fldTanEx
            End If
        Else
            .Edit
            !fldTExp = False
            .Update
            lblTE.Visible = False
            txtEx_tan.Width = 1695
            txtEx_tan = ""
        End If

        If !fldTanExType <> "" Then cboTan_date = !fldTanExType Else
cboTan_date.ListIndex = -1
        txtStreet = !fldStreet
        txtCity = !fldCity
        cboState = !fldState
        txtZip = !fldZip
        txtDOB = !fldDOB
        txtPhone = !fldPhoneNumber
        txtSS = !fldSoc
        If !fldPayDue <> "" Then
            txtPayduedate = !fldPayDue
            mdatOD = !fldPayDue
            mintOD = Date - mdatOD
            If mintOD > 0 Then
                txtLate = mintOD
                .Edit
                !fldOD = True
                .Update
            Else
                txtLate = "0"
                .Edit
                !fldOD = False
                .Update
            End If
        Else
            txtPayduedate = ""
            txtLate = "0"
            .Edit
            !fldOD = False
            .Update
        End If

        txtAmountdue = Format(!fldAmountDue, "Currency")
        txtInstall = Format(!fldInstallAmount, "Currency")
        lblBalance = Format(!fldBalance, "Currency")
        If !fldNotes <> "" Then txtNotes = !fldNotes Else txtNotes = ""
    End With
End Sub

Public Sub ClearRecord()
    If mintClear = 1 Then
        txtId = ""
        mintClear = 0
    Else
        txtId = mrstGym!fldMemberID
    End If
    txtLastName = ""
    txtFirstName = ""
    optM.Value = False
    optM.TabStop = True
    optF.Value = False
    txtRecommed = ""
    optGym.Value = False
    optGym.TabStop = True
    optTanning.Value = False
    optGym_Tanning.Value = False
    txtEx_gym = ""
    cboGym_date.ListIndex = -1
    txtEx_tan = ""
    cboTan_date.ListIndex = -1
    txtStreet = ""
    txtCity = ""
    cboState.ListIndex = -1
    txtZip = ""
    txtDOB = ""
    txtPhone = ""
    txtSS = ""
    txtPayduedate = ""
    txtAmountdue = ""
    txtInstall = ""
    lblBalance = Format(0, "Currency")
    txtNotes = ""
    lblGE.Visible = False
    txtEx_gym.Width = 1695
    lblTE.Visible = False
    txtEx_tan.Width = 1695
    txtLate = "0"
End Sub

Public Sub WriteRecord()
    With mrstGym
```

```
        If flag = 1 Then
           !fldMemberID = txtId
           flag = 0
        Else
           .Edit
        End If
        !fldLastName = txtLastName
        !fldFirstName = txtFirstName
        If optM = True Then !fldGender = "M"
        If optF = True Then !fldGender = "F"
        If txtRecommed <> "" Then !fldRecommend = txtRecommed Else
!fldRecommend = ""
        If optGym = True Then !fldMemberShip = "Gym"
        If optTanning = True Then !fldMemberShip = "Tanning"
        If optGym_Tanning = True Then !fldMemberShip = "Gym & Tanning"
        If txtEx_gym <> "" Then !fldGymEx = txtEx_gym Else !fldGymEx =
Null
        If cboGym_date <> "" Then !fldGymExType = cboGym_date
        If txtEx_tan <> "" Then !fldTanEx = txtEx_tan Else !fldTanEx = Null
        If cboTan_date <> "" Then !fldTanExType = cboTan_date
        !fldStreet = txtStreet
        !fldCity = txtCity
        !fldState = cboState
        !fldZip = txtZip
        !fldDOB = txtDOB
        !fldPhoneNumber = txtPhone
        If txtSS <> "" Then !fldSoc = txtSS
        If txtPayduedate <> "" Then !fldPayDue = txtPayduedate Else
!fldPayDue = Null
```

```
        If txtAmountdue <> "" Then !fldAmountDue = txtAmountdue Else
!fldAmountDue = "0"
        If txtInstall <> "" Then !fldInstallAmount = txtInstall Else
!fldInstallAmount = "0"
        !fldBalance = lblBalance
        If txtNotes <> "" Then !fldNotes = txtNotes Else !fldNotes = ""
    End With
End Sub

Private Sub txtId_KeyPress(KeyAscii As Integer)
    If mintFind = 1 Then
        If KeyAscii = 13 Then
            mrstGym.MoveFirst
            Do Until mrstGym.EOF
                If txtId = mrstGym!fldMemberID Then
                    ShowRecord
                    mintFind = 0
                Exit Sub
                Else
                    mrstGym.MoveNext
                End If
            Loop
            MsgBox "Member doesn't exist", vbOKOnly + vbInformation,
"Wrong Entry"
            txtId.SelStart = 0
            txtId.SelLength = Len(txtId)
        End If
    End If
End Sub
```

```
Private Sub txtLastName_KeyPress(KeyAscii As Integer)
    If mintFind = 1 Then
        If KeyAscii = 13 Then
            mrstGym.MoveFirst
            Do Until mrstGym.EOF
                If txtLastName = mrstGym!fldLastName Then
                    mblnBrow = True
                    Set mrstGym = pdbMembers.OpenRecordset("SELECT *
FROM tblMembers WHERE fldLastName = '" & txtLastName & "' ORDER
BY fldMemberID")
                    ShowRecord
                    mintFind = 0
                Exit Sub
                Else
                    mrstGym.MoveNext
                End If
            Loop
            MsgBox "Member doesn't exist", vbOKOnly + vbInformation,
"Wrong Entry"
            txtLastName.SelStart = 0
            txtLastName.SelLength = Len(txtLastName)
        End If
    End If
End Sub

Public Sub Checktxt()
    If txtId = "" Or txtLastName = "" Or txtFirstName = "" _
    Or txtStreet = "" Or txtCity = "" Or cboState.ListIndex = -1 _     Or txtZip
= "" Or txtDOB = "" Or txtPhone = "" Or txtSS = "" Then
        mblnCheck = True
    End If
End Sub
```

V. Employees

```
Option Explicit
   Dim pdbEmp As Database
Dim mrstEmp As Recordset
Dim mblnflag As Boolean
Dim mntNum As Integer
Dim mstrId As String
Dim mblnCheck As Boolean
Dim mblnEdit As Boolean
Dim mntDel As Integer

Private Sub cmdCancel_Click()
   mblnflag = False
   mrstEmp.MoveFirst
   ShowRecord
   mblnEdit = True
   EditState
End Sub

Private Sub cmdDelete_Click()
   mntDel = MsgBox("Are you sure you want to DELETE" & vbLf &
txtLastName & ", " & txtFirstName, vbYesNo + vbCritical)
   If mntDel = vbYes Then
      mrstEmp.Delete
      mrstEmp.MoveFirst
      ShowRecord
   End If
End Sub
```

```
Private Sub cmdEdit_Click()
    cmdCancel.Enabled = True
    txtLastName.SetFocus
    mblnEdit = False
    EditState
End Sub

Private Sub cmdEnter_Click()
    cmdCancel.Enabled = True
    mrstEmp.MoveLast
    mntNum = (mrstEmp!fldNumEmp + 1)
    mstrId = "Emp" & Mid(mrstEmp!fldEmployeeID, 4) + 1
    mrstEmp.AddNew
    ClearRecord
    txtLastName.SetFocus
    mblnflag = True
    mblnEdit = False
    EditState
End Sub

Private Sub cmdFirst_Click()
    mrstEmp.MoveFirst
    ShowRecord
End Sub

Private Sub cmdLast_Click()
    mrstEmp.MoveLast
    ShowRecord
```

```
End Sub

Private Sub cmdNext_Click()
    mrstEmp.MoveNext
    If mrstEmp.EOF Then mrstEmp.MoveLast
    ShowRecord
    'MsgBox "This is Last Record...", vbInformation
End Sub

Private Sub cmdPayroll_Click()
    If frmPayroll.WindowState <> vbMinimized Then
        frmPayroll.Top = mdiDtcc.ScaleHeight / 8
        frmPayroll.Left = mdiDtcc.ScaleWidth / 4
    End If
    frmPayroll.Show
End Sub

Private Sub cmdPrev_Click()
    mrstEmp.MovePrevious
    If mrstEmp.BOF Then mrstEmp.MoveFirst
    ShowRecord
    'MsgBox "This is First Record...", vbInformation
End Sub

Private Sub cmdUpdate_Click()
    Checktxt
    If mblnCheck = False Then
        WriteRecord
        mrstEmp.Update
        mblnEdit = True
        EditState
    Else
        MsgBox "All Required!!", vbOKOnly + vbInformation
        mblnCheck = False
        txtEmployeeID.SetFocus
    End If
    If mblnflag = False Then ShowRecord
End Sub

Private Sub Form_Load()
    Set pdbEmp = OpenDatabase(App.Path & "\GymMembers.mdb")
    Set mrstEmp = pdbEmp.OpenRecordset("SELECT * FROM
tblEmployees ORDER BY fldNumEmp")
    mrstEmp.MoveFirst
    ShowRecord
End Sub

Public Sub ShowRecord()
    With mrstEmp
        frmNewEmp.Caption = "Employees Date of Hire: " & !fldDOH
        fra1 = "Employee (" & !fldNumEmp & ")"
        txtEmployeeID = !fldEmployeeID
        txtLastName = !fldLastName
        txtFirstName = !fldFirstName
        If !fldGender = "M" Then optM = True
        If !fldGender = "F" Then optF = True
        txtStreet = !fldStreet
        txtCity = !fldCity
        cboState = !fldState
        txtZip = !fldZipCode
        txtDOB = !fldDOB
        txtPhone = !fldPhoneNumber
        txtSS = !fldSoc
        txtHourlyWage = Format(!fldHourlyWage, "Currency")
        txtTax = !fldTaxRate
        gsngPer = !fldTaxRate
    End With
    LoadEmpPicture
End Sub

Public Sub ClearRecord()
    frmNewEmp.Caption = "Employees Date of Hire: " & Date
    fra1 = "Employee (" & mntNum & ")"
    txtEmployeeID = mstrId
    txtLastName = ""
    txtFirstName = ""
    optM.Value = False
    optM.TabStop = True
    optF.Value = False
    txtStreet = ""
    txtCity = ""
    cboState.ListIndex = -1
    txtZip = ""
    txtDOB = ""
    txtPhone = ""
    txtSS = ""
    txtHourlyWage = Format(0, "Currency")
    txtTax = ""
    imgEmp.Picture = LoadPicture(App.Path & "\Images\NoPic.jpg")
End Sub

Public Sub WriteRecord()
    With mrstEmp
        If mblnflag = True Then
            !fldDOH = Date
            !fldEmployeeID = txtEmployeeID
            !fldNumEmp = mntNum
            mblnflag = False
        Else
            .Edit
        End If
        !fldLastName = txtLastName
        !fldFirstName = txtFirstName
        If optM = True Then !fldGender = "M"
        If optF = True Then !fldGender = "F"
        !fldStreet = txtStreet
        !fldCity = txtCity
        !fldState = cboState
        !fldZipCode = txtZip
        !fldDOB = txtDOB
        !fldPhoneNumber = txtPhone
        !fldSoc = txtSS
        !fldHourlyWage = txtHourlyWage
        !fldTaxRate = txtTax
    End With
End Sub
```

```
Private Sub txtHourlyWage_GotFocus()
   txtHourlyWage.SelStart = 0
   txtHourlyWage.SelLength = Len(txtHourlyWage)
End Sub

Public Sub Checktxt()
   If txtEmployeeID = "" Or txtLastName = "" Or txtFirstName = "" _
   Or txtHourlyWage = "" Or txtTax = "" Or txtStreet = "" Or _
   txtCity = "" Or cboState.ListIndex = -1 Or txtZip = "" Or _
   txtDOB = "" Or txtPhone = "" Or txtSS = "" Then
      mblnCheck = True
   End If
End Sub

Public Sub EditState()
   If mblnEdit = True Then
      optM.Enabled = False
      optF.Enabled = False
      txtLastName.Locked = True
      txtFirstName.Locked = True
      txtHourlyWage.Locked = True
      txtTax.Locked = True
      txtStreet.Locked = True
      txtCity.Locked = True
      cboState.Locked = True
      txtZip.Locked = True
      txtDOB.Locked = True
      txtPhone.Locked = True
```

```
    txtSS.Locked = True
    txtHourlyWage.Locked = True
    txtTax.Locked = True
    cmdUpdate.Enabled = False
    cmdEdit.Enabled = True
    cmdFirst.Enabled = True
    cmdLast.Enabled = True
    cmdNext.Enabled = True
    cmdPrev.Enabled = True
    cmdDelete.Enabled = True
    cmdEnter.Enabled = True
    cmdCancel.Enabled = False
Else
    optM.Enabled = True
    optF.Enabled = True
    txtLastName.Locked = False
    txtFirstName.Locked = False
    txtHourlyWage.Locked = False
    txtTax.Locked = False
    txtStreet.Locked = False
    txtCity.Locked = False
    cboState.Locked = False
    txtZip.Locked = False
    txtDOB.Locked = False
    txtPhone.Locked = False
    txtSS.Locked = False
    txtHourlyWage.Locked = False
    txtTax.Locked = False
    cmdUpdate.Enabled = True
```

```
        cmdEdit.Enabled = False
        cmdFirst.Enabled = False
        cmdLast.Enabled = False
        cmdNext.Enabled = False
        cmdPrev.Enabled = False
        cmdDelete.Enabled = False
        cmdEnter.Enabled = False
    End If
End Sub

Public Sub LoadEmpPicture()
    Dim pstrPic As String
    pstrPic = txtEmployeeID
    On Error Resume Next
    imgEmp.Picture = LoadPicture(App.Path & "\Images\" & pstrPic &
".jpg")
    If Err Then
         imgEmp.Picture = LoadPicture(App.Path & "\Images\NoPic.jpg")
    End If
End Sub
```

VI. Payroll

```
Option Explicit
Dim pdbPay As Database
Dim mrstPay As Recordset
Dim gblnPeriod As Boolean

Private Sub cmdAdd_Click()
   If txtHourWorked = vbNullString Then
      MsgBox "Enter Hours Worked", vbOKOnly + vbInformation
      txtHourWorked.SetFocus
   ElseIf txtNet = vbNullString Then
      MsgBox "Must Calculate", vbOKOnly + vbInformation
      cmdCalculate.SetFocus
   Else
      WriteRecord
      mrstPay.Update
      MsgBox "Payroll was added", vbOKOnly
      Unload Me
   End If
End Sub

Private Sub cmdBrowse_Click()
   Dim Message, Title, MyValue, pstrUpper As String

   Message = "Enter employee ID to view Employee Records." _
   & vbNewLine & vbTab & vbTab & "     -or-" & vbNewLine & _
   "Enter ( All ) to view all Employee Records."
   Title = "Browse Records"
```

```
        On Error Resume Next
    mrstPay.MoveFirst
    MyValue = InputBox(Message, Title)
    pstrUpper = UCase(MyValue)
    If pstrUpper = "ALL" Then
        Set mrstPay = pdbPay.OpenRecordset("SELECT * FROM tblPayroll
ORDER BY fldEmployeeID")
        On Error Resume Next
        txtHourWorked.Locked = True
        cmdCalculate.Enabled = False
        cmdAdd.Enabled = False
        cmdFirst.Enabled = True
        cmdLast.Enabled = True
        cmdPrevious.Enabled = True
        cmdNext.Enabled = True
        ShowRecord
    ElseIf MyValue <> "" Then
        Set mrstPay = pdbPay.OpenRecordset("SELECT * FROM tblPayroll
WHERE fldEmployeeID = '" & MyValue & "' ORDER BY fldPayNum")
        On Error Resume Next
        txtHourWorked.Locked = True
        cmdCalculate.Enabled = False
        cmdAdd.Enabled = False
        cmdFirst.Enabled = True
        cmdLast.Enabled = True
        cmdPrevious.Enabled = True
        cmdNext.Enabled = True
        ShowRecord
    End If
    If Err And MyValue <> "" Then
       MsgBox "Employee does not exist.", vbOKOnly
       cmdFirst.Enabled = False
       cmdLast.Enabled = False
       cmdPrevious.Enabled = False
       cmdNext.Enabled = False
   End If
End Sub

Private Sub cmdCalculate_Click()
   Dim psngGross As Single
   Dim psngTax As Single
   Dim psngNet As Single
    On Error Resume Next
   gblnPeriod = False
    If txtHourWorked = vbNullString Then
       MsgBox "Enter Hours Worked", vbOKOnly + vbInformation
       txtHourWorked.SetFocus
   Else
       psngGross = (txtHourlyWage * txtHourWorked)
       txtGross = Format(psngGross, "Currency")
       psngTax = (gsngPer * psngGross)
       txtTax = Format(psngTax, "Currency")
       psngNet = psngGross - psngTax
       txtNet = Format(psngNet, "Currency")
   End If
 End Sub

Private Sub cmdClose_Click()
```

```
    Unload Me
End Sub

Private Sub cmdFirst_Click()
    mrstPay.MoveFirst
    ShowRecord
End Sub

Private Sub cmdLast_Click()
    mrstPay.MoveLast
    ShowRecord
End Sub

Private Sub cmdNext_Click()
    mrstPay.MoveNext
    If mrstPay.EOF Then mrstPay.MoveLast
    ShowRecord
    'MsgBox "This is Last Record...", vbInformation
End Sub

Private Sub cmdPrevious_Click()
    mrstPay.MovePrevious
    If mrstPay.BOF Then mrstPay.MoveFirst
    ShowRecord
    'MsgBox "This is First Record...", vbInformation
End Sub

Private Sub Form_Load()
    Set pdbPay = OpenDatabase(App.Path & "\GymMembers.mdb")
    Set mrstPay = pdbPay.OpenRecordset("tblPayroll")

    txtEmployeeID = frmNewEmp.txtEmployeeID
    txtLastName = frmNewEmp.txtLastName
    txtFirstName = frmNewEmp.txtFirstName
    txtHourlyWage = frmNewEmp.txtHourlyWage
    txtDatePaid = Date
    cmdFirst.Enabled = False
    cmdLast.Enabled = False
    cmdPrevious.Enabled = False
    cmdNext.Enabled = False
End Sub

Public Sub ShowRecord()
    With mrstPay
        txtEmployeeID = !fldEmployeeID
        txtLastName = !fldLastName
        txtFirstName = !fldFirstName
        txtHourlyWage = Format(!fldHourlyWage, "Currency")
        txtHourWorked = !fldHoursWorked
        txtDatePaid = !fldDatePaid
        txtGross = !fldGrossPay
        txtTax = !fldTaxWithheld
        txtNet = !fldNetPay
    End With
End Sub

Public Sub WriteRecord()
    mrstPay.AddNew
    With mrstPay
        !fldEmployeeID = txtEmployeeID
        !fldLastName = txtLastName
        !fldFirstName = txtFirstName
        !fldHourlyWage = txtHourlyWage
        !fldHoursWorked = txtHourWorked
        !fldDatePaid = txtDatePaid
        !fldGrossPay = txtGross
        !fldTaxWithheld = txtTax
        !fldNetPay = txtNet
    End With
End Sub

Private Sub Form_Unload(Cancel As Integer)
    gblnPeriod = False
End Sub

Private Sub txtHourWorked_KeyPress(KeyAscii As Integer)
     If KeyAscii < Asc("0") Or KeyAscii > Asc("9") Then
        If KeyAscii = 13 Then
            KeyAscii = 0
            SendKeys vbTab
        ElseIf KeyAscii = Asc(".") And gblnPeriod = False Then
            gblnPeriod = True
        ElseIf KeyAscii = Asc(vbBack) Then
            'fine...
        Else
            KeyAscii = 0
        End If
    End If
End Sub

Private Sub txtHourWorked_LostFocus()
    If txtHourWorked = "." Then
        txtHourWorked = ""
        gblnPeriod = False
    ElseIf Right(txtHourWorked, 1) = "." Then
        txtHourWorked = Format(txtHourWorked, "")
        gblnPeriod = False
    End If
End Sub
```

VII. Products

```
Option Explicit
Dim mdbPro As Database
Dim mrstPro As Recordset
Dim mstrCat As String
Private Sub cboCat_Click()
   mstrCat = cboCat.Text
   Set mdbPro = OpenDatabase(App.Path & "\GymMembers.mdb")
   Set mrstPro = mdbPro.OpenRecordset("SELECT * FROM tblAllProducts
WHERE fldCategory = '" & mstrCat & "' ORDER BY fldProduct_ID")
   mrstPro.MoveFirst
   ShowRecord
   lblCat.Visible = False
   cboCat.Visible = False
   lin1(0).Visible = True
   lin1(1).Visible = True
   lblProduct.Visible = True
   Me.Height = 4485
   Me.Width = 7740
   fraBorder.Height = 4035
   fraBorder.Width = 7515
   cmdView.Caption = "All " & cboCat.Text
End Sub
Private Sub cmdChange_Click()
   lblCat.Visible = True
   cboCat.Visible = True
   lin1(0).Visible = False
   lin1(1).Visible = False
```

```
    lblProduct.Visible = False
    Me.Height = 1200
    Me.Width = 4000
    fraBorder.Height = 735
    fraBorder.Width = 3795
End Sub

Private Sub cmdExit_Click()
    Unload Me
End Sub

Private Sub cmdFirst_Click()
    mrstPro.MoveFirst
    ShowRecord
End Sub

Private Sub cmdLast_Click()
    mrstPro.MoveLast
    ShowRecord
End Sub

Private Sub cmdNext_Click()
    mrstPro.MoveNext
    If mrstPro.EOF Then mrstPro.MoveLast
    ShowRecord
    'MsgBox "This is Last Record...", vbInformation
End Sub

Private Sub cmdPrevious_Click()
```

```
    mrstPro.MovePrevious
    If mrstPro.BOF Then mrstPro.MoveFirst
    ShowRecord
    'MsgBox "This is First Record...", vbInformation
End Sub

Public Sub ShowRecord()
    With mrstPro
        txtCategory = !fldCategory
        txtPid = !fldProduct_ID
        txtxDesr = !fldDescription
        txtBrand = !fldBrand
        txtSup = !fldSupplier
        txtODate = !fldOrder_date
        If !fldLast_Inventory <> "" Then txtIDate = !fldLast_Inventory Else
txtIDate = ""
        txtCase = !fldCase
        txtNCase = !fldNCase
        txtQuantity = !fldQuantity
        txtCasePrice = Format(!fldCasePrice, "CURRENCY")
        txtPrice = Format(!fldSalePrice, "CURRENCY")
    End With
End Sub

Private Sub cmdView_Click()
    gstrProName = cboCat
    gstrAllProduct = "SELECT * FROM tblAllProducts WHERE fldCategory
= '" & cboCat & "' ORDER BY fldBrand"
    If frmAllPro.WindowState <> vbMinimized Then
```

```
        frmAllPro.Top = mdiDtcc.ScaleHeight / 30
        frmAllPro.Left = mdiDtcc.ScaleWidth / 10
    End If
    frmAllPro.Show
End Sub

Private Sub Form_Load()
    lin1(0).Visible = False
    lin1(1).Visible = False
    lblProduct.Visible = False
    Me.Height = 1200
    Me.Width = 4000
    fraBorder.Height = 735
    fraBorder.Width = 3795
End Sub
```

VIII. Inventory

```
Option Explicit
Dim mdbInv As Database
Dim mrstInv As Recordset
Dim mstrCat As String
Private Sub cboCat_Click()
    mstrCat = cboCat.Text
    Set mdbInv = OpenDatabase(App.Path & "\GymMembers.mdb")
    Set mrstInv = mdbInv.OpenRecordset("SELECT * FROM tblAllProducts
WHERE fldCategory = '" & mstrCat & "' ORDER BY fldProduct_ID")
    mrstInv.MoveFirst
    ShowRecord
    txtQuantity.SetFocus
    cmdUpdate.Enabled = True
    cmdFirst.Enabled = True
    cmdPrevious.Enabled = True
    cmdNext.Enabled = True
    cmdLast.Enabled = True
End Sub

Public Sub ShowRecord()
    With mrstInv
        txtPid = !fldProduct_ID
        txtxDesr = !fldDescription
        txtBrand = !fldBrand
    End With
End Sub
```

```
Private Sub cboCat_LostFocus()
    If cboCat.ListIndex = -1 Then cboCat.SetFocus
End Sub

Private Sub cmdCancel_Click()
    Unload Me
End Sub

Private Sub cmdFirst_Click()
    mrstInv.MoveFirst
    ShowRecord
    txtQuantity.SetFocus
End Sub

Private Sub cmdLast_Click()
    mrstInv.MoveLast
    ShowRecord
    txtQuantity.SetFocus
End Sub

Private Sub cmdNext_Click()
    mrstInv.MoveNext
    If mrstInv.EOF Then mrstInv.MoveLast
    ShowRecord
    'MsgBox "This is Last Record...", vbInformation
    txtQuantity.SetFocus
End Sub

Private Sub cmdPrevious_Click()
```

```
    mrstInv.MovePrevious
    If mrstInv.BOF Then mrstInv.MoveFirst
    ShowRecord
    'MsgBox "This is First Record...", vbInformation
    txtQuantity.SetFocus
End Sub

Private Sub cmdUpdate_Click()
    If txtQuantity <> "" Then
        With mrstInv
            .Edit
            !fldQuantity = txtQuantity
            !fldLast_Inventory = txtDate
            .Update
            MsgBox "Inventory Updated", vbOKOnly + vbInformation
            Unload Me
        End With
    Else
        MsgBox "Quantity Required!!!", vbOKOnly + vbInformation
        txtQuantity.SetFocus
    End If
End Sub
Private Sub Form_Load()
    txtDate = Date
End Sub
Private Sub txtQuantity_KeyPress(KeyAscii As Integer)
    If KeyAscii < Asc("0") Or KeyAscii > Asc("9") Then
        If KeyAscii = 13 Then
            KeyAscii = 0
            SendKeys vbTab
        ElseIf KeyAscii = Asc(vbBack) Then
             'fine...
        Else
            KeyAscii = 0
        End If
    End If
End Sub
```

IX. Module

```
Public gblnRec As Boolean
Public gblnPriv As Boolean
Public gblnCK As Boolean
Public gsngPer As Single
Public gstrAllProduct As String
Public gstrAllRec As String
Public gstrAllMem As String
Public gstrProName As String
Public pdbMembers As Database
Public mrstGym As Recordset
Global Const winding = 2
Global Const alternate = 1
Global Const rgn_or = 2

Type pointapi
  X As Long
  Y As Long
End Type

Declare Function CreatePolygonRgn Lib "gdi32" (lpPoint As pointapi,
ByVal nCount As Long, ByVal nPolyfillMode As Long) As Long
Declare Function CreateRoundRectRgn Lib "gdi32" (ByVal x1 As Long,
ByVal y1 As Long, ByVal x2 As Long, ByVal y2 As Long, ByVal x3 As
Long, ByVal y3 As Long) As Long
Declare Function CombineRgn Lib "gdi32" (ByVal hDestRgn As Long,
ByVal hSrcRgn1 As Long, ByVal hSrcRgn2 As Long, ByVal
nCombineMode As Long) As Long
```

```
Declare Function SetWindowRgn Lib "user32" (ByVal hWnd As Long,
ByVal hRgn As Long, ByVal bRedraw As Boolean) As Long
Declare Function DeleteObject Lib "gdi32" (ByVal hObject As Long) As
Long
```

System Maintanence & Evalution

The maintenance avtivity consist of following tasks:

1. Backup
2. Digonastic
3. Integrity changes
4. Recovery
5. Design changes
6. Performance tuning

These features ensure the availability of the databases round the clock as the database maintenance is possible online when the system is in use. RDBMS allows an online maintenance, rapid recovery and software based fault tolerance. The rapid recovery features allows the system adminiatrator to provide 'time' to go back for the recovery of the data if the system fails due to power or network crash. Based on theis time, system automatically goes back and collects all the changes and writes to disk.

CONCLUSION

The "**GYM MANAGEMENT SYSTEM**" is successfully designed and developed to fulfilling the necessary requirements, as identified in the requirements analysis phase, such as the system is very much user friendly, form level validation and field level validation are performing very efficiently.

The new computerized system was found to be much faster and reliable and user friendly then the existing system, the system has been designed and developed step by step and tested successfully. It eliminates the human error that are likely to creep in the kind of working in which a bulk quantity of data and calculations as to be processed.

The system results in quick retrieval of information that is very vital for the progress any organization. Cost is minimized in case of stationary. Burden of manual work is reduced as whenever transaction takes place, there is a no need to record it in many places manually.

SCOPE FOR FURTHER DEVELOPMENT

The software has been developed in such a way that it can accept modifications and further changes. The software is very user friendly and future any changes can be done easily.

Software restructuring is carried out. Software restructuring modifies source code in an effort to make it amenable to future changes. In general, restructuring does not modify the overall program architecture. It tends to focus on the design details of individual modules and on local data structure defined within modules.

Every system should allow scope for further development or enhancement. The system can be adapted for any further development. The system is so flexible to allow any modification need for the further functioning of programs.

Since the objectives may be brought broad in future, the system can be easily modified accordingly, as the system has been modularized. The future expansion can be done in a concise manner in order to improve the efficiently of the system.

LIST OF ABBEREVATIONS

OBJECTS		**Prefixes**
Form		frm
Text Box		Txt
Combo box		Cmb
Frame		Fra
Command Button		Cmd
Label		Lbl
Option Button		Opt
Date picker		Dtb
Menu		mnu

BIBLIOGRAPHY

List of References: -

- Visual Basic Black Book by Stephen Holzner
- Programming In Visual Basic 6.0 by Julia Bradley & Anita Millspaugh.
- System Analysis and Design in Changing World by Satzinger.
- Software Engineering – A Practitioners Approach 7/e, by Roger S. Pressman.
- System Analysis and Design.
- Mastering VB 6.0

Web Refrence

- www.wikipedia.org
- www.vbcode.com
- www.codeguru.com/vb
- www.vb.com
- www.oop.com