R.V. GOVERNMENT ARTS COLLEGE

CHENGALPATTU – 603 001.

(Affiliated to University of Madras)

RECORD NOTE BOOK

Register Number	**:**	____________________
Name	**:**	____________________
Class	**:**	____________________
Subject	**:**	____________________
Semester	**:**	____________________

R.V. GOVERNMENT ARTS COLLEGE

CHENGALPATTU – 603 001.

(Affiliated to University of Madras)

PG Department of Computer Science

Register No :	______________________
Class	**: B.Sc., Computer Science – Third Year**
Subject	**: Practical VIII – Mini project [SE26Q]**

This is to certify that this is a Bonafide Record of Practical work done by ……………………………………, PG Department of Computer Science, Rajeshwari Vedachalam Government Arts College, Chengalpattu – 603 001 during the academic year 2024 - 2025.

Staff In-Charge **Head of the Department**

Submitted for the B.Sc., Degree University Practical Examination held on .... at R.V. Government Arts College, Chengalpattu.

Internal Examiner **External Examiner**

GYM MANAGEMENT SYSTEM

(A SMART AND EFFICIENT GYM MANAGEMENT AND REVENUE TRACKING PLATFORM)

GUIDED BY	:	Dr. A N SWAMYNATHAN
TEAM MEMBERS NAME	:	PRAVEEN S
		DHANUSH P
		KALAIYARASI P
CLASS	:	III-BSC COMPUTER SCIENCE

TABLE OF INDEX

1. ABSTRACT

The **Gym Management System** is a Java-based application designed to streamline the operations of a fitness center. This system provides a user-friendly interface for managing various aspects of a gym, including member registration, trainer assignments, payment tracking, and revenue generation. Built with Java and MySQL, the system ensures efficient data management and secure user authentication.

- **Member Management**: Enables the addition, modification, and removal of gym members.
- **Trainer Management**: Facilitates the assignment and tracking of trainers.
- **Payment Tracking**: Maintains records of membership fees and dues.
- **Revenue Generator**: Provides insights into total income over a specified period.
- **User Authentication**: Secure login system to prevent unauthorized access

The Gym Management System enhances operational efficiency by automating manual tasks and reducing administrative workload. Its integration with a database ensures data consistency and ease of retrieval, making it a robust solution for gym owners and administrators

2. INTRODUCTION

2.1 Overview

The Gym Management System **i**s a Java-based application designed to streamline and automate gym operations. It provides functionalities for member management, trainer assignments, payment tracking, and revenue analysis through an intuitive user interface. Built using Java (Swing) and MySQL**,** the system ensures secure data management and efficient workflow automation

Key features include member registration, trainer management, payment processing, and a revenue generator module for financial analysis. With a user-friendly design and secure authentication system, this project enhances efficiency, reduces manual workload, and improves the overall management of a fitness center.

2.2 Objective

The The Gym Management System aims to automate and streamline gym operations by managing members, trainers, payments, and revenue tracking efficiently. It simplifies member registration, trainer assignments, and payment tracking, ensuring secure data storage and easy access. The system also includes a Revenue Generator module for financial analysis, helping gym owners monitor income and optimize business performance. Through automation, it reduces manual workload, enhances security, and improves overall operational efficiency

3. Software and hardware specification

This project utilizes a combination of server-side and client-side technologies to create a fully functional Food Donation Management System. Below are the key software tools used:

1. Java (Swing for GUI)

The system is developed in Java, a robust, platform-independent language.

Java Swing is used to create the graphical user interface (GUI), making the application user-friendly and interactive.

2. Development Tool – Apache NetBeans IDE 24:)

NetBeans provides an integrated environment for coding, debugging, and testing the application. It simplifies the development process with built-in support for **Java Swing and database connectivity**.

3. Backend – Java JDBC for Database Connectivity:

Java Database Connectivity (JDBC) is used to connect the Java application with the MySQL database.
It enables seamless data fetching, insertion, updating, and deletion operations

4. MySQL

MySQL is used to store and manage all gym-related data, including member records, trainer details, payments, and revenue
It ensures structured data storage, quick retrieval, and easy scalability..

5. External Library – JCalendar (JAR File)

In the Gym Management System, an external library called **JCalendar** is used for handling date-related functions efficiently. **JCalendar** is a **Java-based Swing component** that provides a **graphical date picker** for selecting and managing dates in the application.

6. Operating System Compatibility – Windows / Linux:

The system is cross-platform and can run on **any OS with a Java Runtime Environment (JRE)**..

7. Security – User Authentication for Data Protection:

login system ensures that only authorized users (such as gym admins) can access the system
This prevents unauthorized access and protects sensitive data.

HARDWARE SPECIFICATION

The **Gym Management System** requires a minimum hardware configuration for optimal performance. A **processor** of Intel Core i3 or higher (or AMD equivalent) with a clock speed of at least **2.0 GHz** is recommended. The system should have at least **4GB of RAM**, though **8GB or more** is preferred for smoother execution. A **100GB HDD** is required for data storage, but an **SSD** is recommended for faster access. The system should support a **display resolution of 1366×768** or higher. It is compatible with **Windows 10/11 and Linux (Ubuntu, Fedora)** operating systems. An **internet connection** is necessary if the MySQL database is hosted online. Additionally, standard **peripherals** such as a **keyboard, mouse, and printer** (for receipts) may be required. This setup ensures efficient and seamless management of gym operations.

4. ADVANTAGES & DISADVANTAGES

Advantages:

1. Automation of Operations:

Reduces manual work by managing members, trainers, and payments digitally.

2. Secure Data Management:

MySQL database ensures safe storage and retrieval of member and financial records.

3. Efficient Payment Tracking

Ensures timely collection of fees and tracks pending dues.

4. User-Friendly Interface:

Java Swing-based GUI makes it easy for gym staff to use

Disadvantages:

1. Limited Online Access:

The system is primarily desktop-based, making remote access difficult unless modified for cloud integration

2. Requires Technical Knowledge:

If not properly secured, user data (emails, addresses) could be vulnerable to breaches.

3. Hardware Dependent:

Performance may vary depending on system specifications

4. No Mobile App Support:

Cannot be accessed via smartphones unless developed separately..

5. TABLE STRUCTURE

To support these operations, the system uses a relational database with three primary tables:

1. **Payment** – Stores payments details of all users.

2. **Member** – Records information about a member with their individual types.

3. **Trainer** – Stores details about trainer and their information

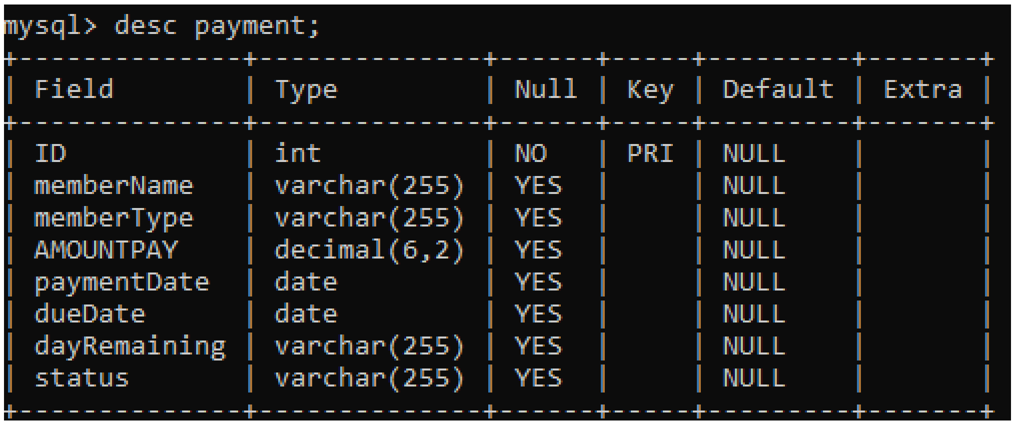
The following section provides a detailed explanation of the table structures used in the system.

1. Payment Table

Purpose:

The payment table stores registration information of payment date and so on of all users in the system

Table Structure:



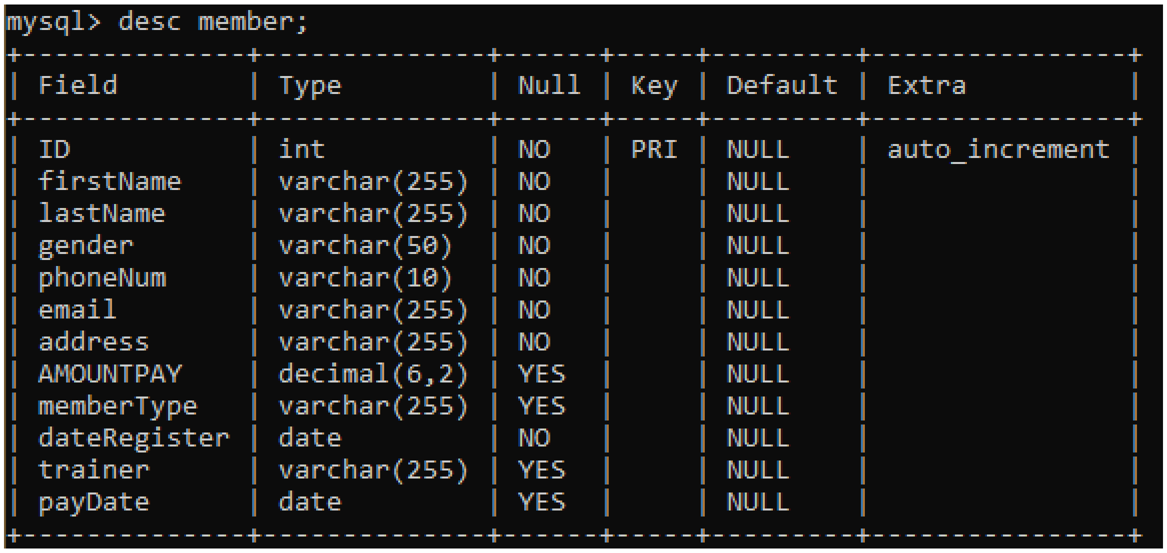
```
mysql> desc payment;
+--------------+--------------+------+-----+---------+-------+
| Field        | Type         | Null | Key | Default | Extra |
+--------------+--------------+------+-----+---------+-------+
| ID           | int          | NO   | PRI | NULL    |       |
| memberName   | varchar(255) | YES  |     | NULL    |       |
| memberType   | varchar(255) | YES  |     | NULL    |       |
| AMOUNTPAY    | decimal(6,2) | YES  |     | NULL    |       |
| paymentDate  | date         | YES  |     | NULL    |       |
| dueDate      | date         | YES  |     | NULL    |       |
| dayRemaining | varchar(255) | YES  |     | NULL    |       |
| status       | varchar(255) | YES  |     | NULL    |       |
+--------------+--------------+------+-----+---------+-------+
```

2. Member Table

Purpose:

The Member table used to store the information of the members with respective member type and registration date.

Table Structure:



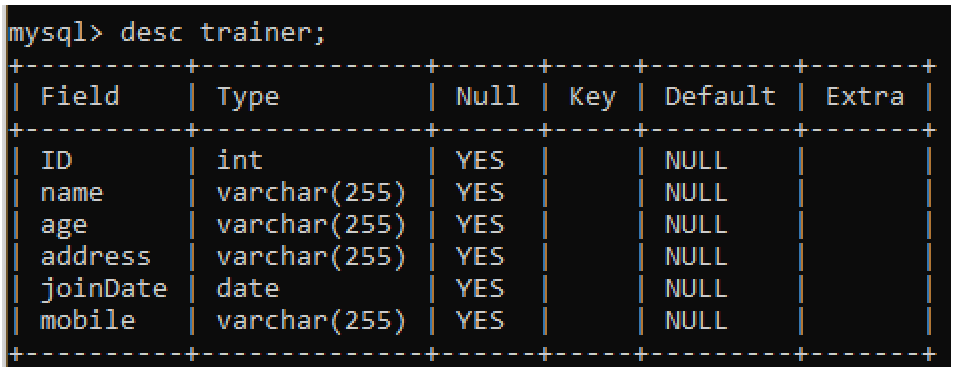
```
mysql> desc member;
+--------------+--------------+------+-----+---------+----------------+
| Field        | Type         | Null | Key | Default | Extra          |
+--------------+--------------+------+-----+---------+----------------+
| ID           | int          | NO   | PRI | NULL    | auto_increment |
| firstName    | varchar(255) | NO   |     | NULL    |                |
| lastName     | varchar(255) | NO   |     | NULL    |                |
| gender       | varchar(50)  | NO   |     | NULL    |                |
| phoneNum     | varchar(10)  | NO   |     | NULL    |                |
| email        | varchar(255) | NO   |     | NULL    |                |
| address      | varchar(255) | NO   |     | NULL    |                |
| AMOUNTPAY    | decimal(6,2) | YES  |     | NULL    |                |
| memberType   | varchar(255) | YES  |     | NULL    |                |
| dateRegister | date         | NO   |     | NULL    |                |
| trainer      | varchar(255) | YES  |     | NULL    |                |
| payDate      | date         | YES  |     | NULL    |                |
+--------------+--------------+------+-----+---------+----------------+
```

3. Trainer Table

Purpose:

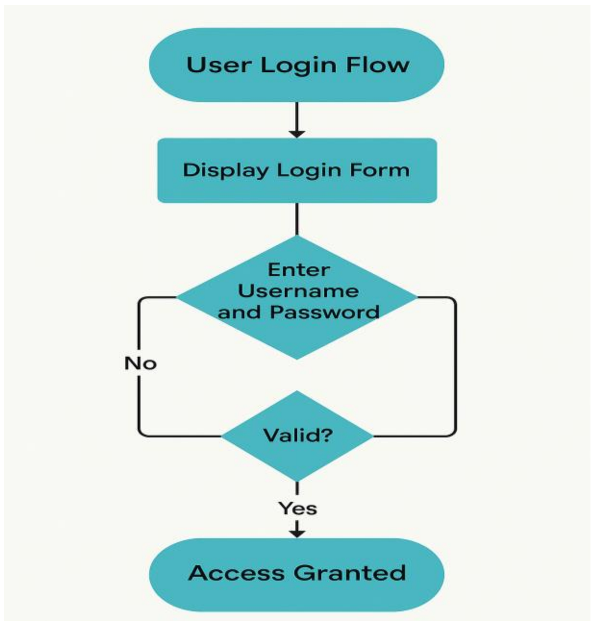
Stores details of **gym trainer**. Helps in **assigning trainers** to members based on expertise.

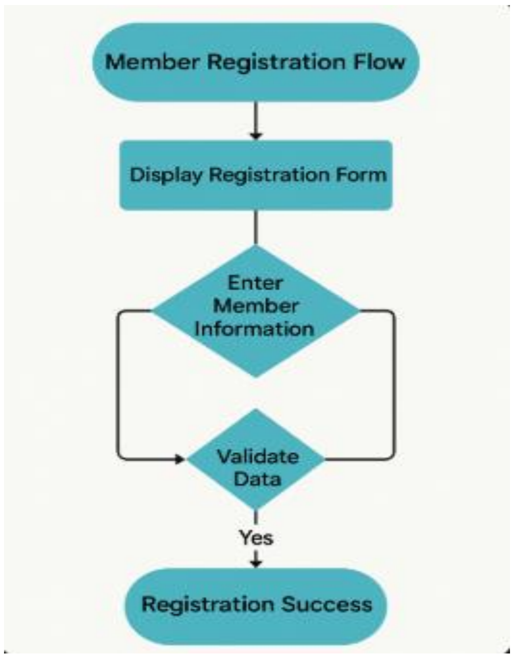
Table Structure:

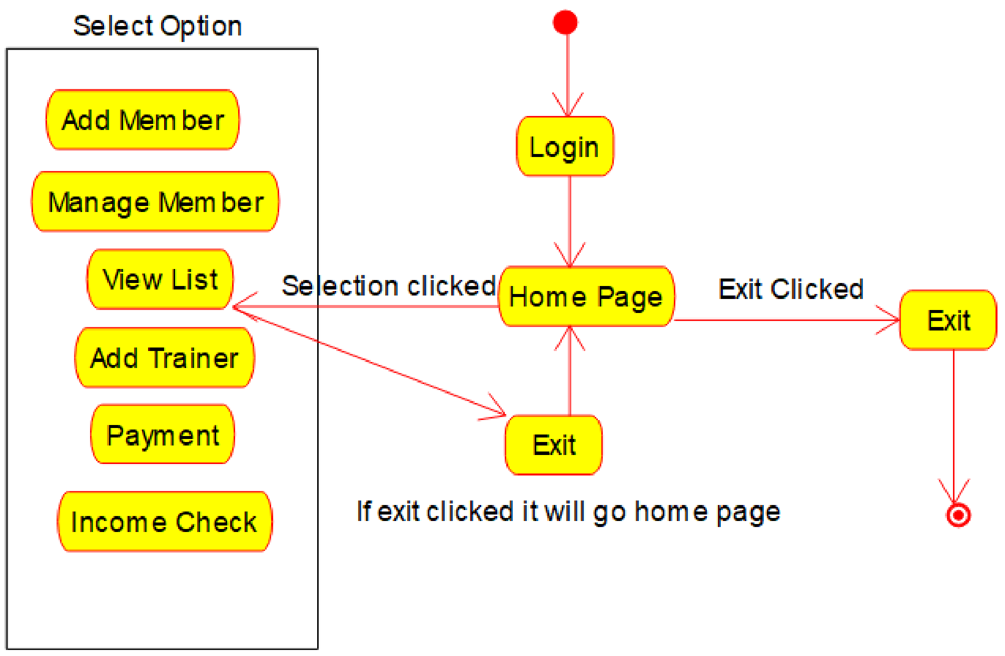


```
mysql> desc trainer;
+----------+--------------+------+-----+---------+-------+
| Field    | Type         | Null | Key | Default | Extra |
+----------+--------------+------+-----+---------+-------+
| ID       | int          | YES  |     | NULL    |       |
| name     | varchar(255) | YES  |     | NULL    |       |
| age      | varchar(255) | YES  |     | NULL    |       |
| address  | varchar(255) | YES  |     | NULL    |       |
| joinDate | date         | YES  |     | NULL    |       |
| mobile   | varchar(255) | YES  |     | NULL    |       |
+----------+--------------+------+-----+---------+-------+
```

6. FLOW DIAGRAM







7.Source code

#Establish the Database connection

```
package database;
import java.sql.*;
import java.lang.*;

public class ConnectionProvider {
    public static Connection getConnection(){
        String url = "jdbc:mysql://localhost:3306/gmsdb_dump";
        String username = "root";
        String password = "Dhanush1@";

        try{
            Class.forName("com.mysql.cj.jdbc.Driver");
            Connection connnection = DriverManager.getConnection(url,username,password);
            return connnection;
        }catch(ClassNotFoundException | SQLException e){
            return null;
        }
    }
}
```

#login page

```
package com.mycompany.gymmanagementsystem;
import javax.swing.*;
public class login extends javax.swing.JFrame {
    public login() {
        initComponents();
        incorrectusernameView.setVisible(false);
        incorrectPassView.setVisible(false);
    }
    private void initComponents() {

        loginView = new javax.swing.JLabel();
        incorrectPassView = new javax.swing.JLabel();
        userNameTxtField = new javax.swing.JTextField();
        passwordField = new javax.swing.JPasswordField();
        loginBtn = new javax.swing.JButton();
        showpasswordCheckBox = new javax.swing.JCheckBox();
        exitBtn = new javax.swing.JButton();
        incorrectusernameView = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setTitle("Log In ");
        setLocation(new java.awt.Point(0, 0));
        getContentPane().setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());
        loginView.setBackground(new java.awt.Color(204, 204, 204));
        loginView.setFont(new java.awt.Font("Tahoma", 1, 48)); // NOI18N
        loginView.setText("LOG IN");
        incorrectPassView.setBackground(new java.awt.Color(255, 255, 255));
        incorrectPassView.setFont(new java.awt.Font("Tahoma", 1, 14));
        incorrectPassView.setForeground(new java.awt.Color(255, 0, 0));
        incorrectPassView.setIcon(new(getClass().getResource("/icons/incorrecticon.png")));
        incorrectPassView.setText("Incorrect Password");
        userNameTxtField.setFont(new java.awt.Font("Tahoma", 1, 14)); // NOI18N
        userNameTxtField.setText("Enter username!");
        userNameTxtField.addFocusListener(new java.awt.event.FocusAdapter() {

    public void focusGained(java.awt.event.FocusEvent evt) {
                userNameTxtFieldFocusGained(evt);
            }
     public void focusLost(java.awt.event.FocusEvent evt) {
                userNameTxtFieldFocusLost(evt);
            }
        });
        userNameTxtField.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                userNameTxtFieldActionPerformed(evt);
            }
        });
        getContentPane().add(userNameTxtField, new
    org.netbeans.lib.awtextra.AbsoluteConstraints(560, 180, 260, 31));

        passwordField.setFont(new java.awt.Font("Tahoma", 1, 14)); // NOI18N
        passwordField.setText("Enter password!");
        passwordField.addFocusListener(new java.awt.event.FocusAdapter() {
            public void focusGained(java.awt.event.FocusEvent evt) {
                passwordFieldFocusGained(evt);
            }
            public void focusLost(java.awt.event.FocusEvent evt) {
                passwordFieldFocusLost(evt);
            }
        });
        passwordField.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                passwordFieldActionPerformed(evt);
            }
        });
        getContentPane().add(passwordField, new
org.netbeans.lib.awtextra.AbsoluteConstraints(560, 260, 260, 31));


        loginBtn.setFont(new java.awt.Font("Tahoma", 1, 14)); // NOI18N
        loginBtn.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/icons/loginicon.png"))); // NOI18N
        loginBtn.setText("Login");
        loginBtn.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                loginBtnActionPerformed(evt);
            }
        });
        getContentPane().add(loginBtn, new org.netbeans.lib.awtextra.AbsoluteConstraints(550,
370, -1, -1));
```

```
        showpasswordCheckBox.setFont(new java.awt.Font("Tahoma", 1, 14)); // NOI18N
        showpasswordCheckBox.setText("Show passwords");
        showpasswordCheckBox.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                showpasswordCheckBoxActionPerformed(evt);
            }
        });
        getContentPane().add(showpasswordCheckBox, new
org.netbeans.lib.awtextra.AbsoluteConstraints(570, 320, -1, -1));


        exitBtn.setFont(new java.awt.Font("Tahoma", 1, 14));
        exitBtn.setIcon(new (getClass().getResource("/icons/exiticon.png")));
        exitBtn.setText("Exit");
        exitBtn.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                exitBtnActionPerformed(evt);
            }
        });
        incorrectusernameView.setBackground(new java.awt.Color(255, 255, 255));
        incorrectusernameView.setFont(new java.awt.Font("Tahoma", 1, 14)); // NOI18N
        incorrectusernameView.setForeground(new java.awt.Color(255, 0, 0));
        incorrectusernameView.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/icons/incorrecticon.png"))); // NOI18N
        incorrectusernameView.setText("Incorrect Username");


        jLabel2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/image.jpg")));
        setSize(new java.awt.Dimension(937, 469));
        setLocationRelativeTo(null);
```

```
    }// </editor-fold>

    private void showpasswordCheckBoxActionPerformed(java.awt.event.ActionEvent evt) {
        if (showpasswordCheckBox.isSelected()){
            passwordField.setEchoChar((char)0);
        }else{
            passwordField.setEchoChar('*');
        }
    }

    private void loginBtnActionPerformed(java.awt.event.ActionEvent evt) {
        if (userNameTxtField.getText().equals("Enter username!") &&
passwordField.getText().equals("Enter password!")){
            JOptionPane.showMessageDialog(null,"Required username and
password!","Error",JOptionPane.PLAIN_MESSAGE);
        }else{
            if(userNameTxtField.getText().equals("admin") &&
passwordField.getText().equals("admin")){
                dispose();
                //setVisible(false);
                new home().setVisible(true);
            }else if (!userNameTxtField.getText().equals("admin") &&
passwordField.getText().equals("admin")){
                incorrectPassView.setVisible(false);
                incorrectusernameView.setVisible(true);
            }else if (userNameTxtField.getText().equals("admin") &&
!passwordField.getText().equals("admin")){
                incorrectusernameView.setVisible(false);
                incorrectPassView.setVisible(true);
            }else{
                incorrectusernameView.setVisible(true);
                incorrectPassView.setVisible(true);
            }
        }
    }
    private void exitBtnActionPerformed(java.awt.event.ActionEvent evt) {
        int exit = JOptionPane.showConfirmDialog(null, "Do you want to
exit!","Select",JOptionPane.YES_NO_CANCEL_OPTION);
        if (exit == 0){
            System.exit(0);
        }
    }
    private void userNameTxtFieldFocusGained(java.awt.event.FocusEvent evt) {

        if (userNameTxtField.getText().equals("Enter username!")){
            userNameTxtField.setText("");
        }
    }
    private void userNameTxtFieldFocusLost(java.awt.event.FocusEvent evt) {

        if (userNameTxtField.getText().equals("")){
            userNameTxtField.setText("Enter username!");
        }
    }
    private void passwordFieldFocusGained(java.awt.event.FocusEvent evt) {
```

```
    if (passwordField.getText().equals("Enter password!")){
        passwordField.setText("");
    }
}
private void passwordFieldFocusLost(java.awt.event.FocusEvent evt) {

    if (passwordField.getText().equals("")){
        passwordField.setText("Enter password!");
    }
}

private void userNameTxtFieldActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void passwordFieldActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}
public static void main(String args[]) {

    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new login().setVisible(true);
        }
    });
}
```

```
    // Variables declaration - do not modify
    private javax.swing.JButton exitBtn;
    private javax.swing.JLabel incorrectPassView;
    private javax.swing.JLabel incorrectusernameView;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JButton loginBtn;
    private javax.swing.JLabel loginView;
    private javax.swing.JPasswordField passwordField;
    private javax.swing.JCheckBox showpasswordCheckBox;
    private javax.swing.JTextField userNameTxtField;
}
```

#Code to design Home page

```
package com.mycompany.gymmanagementsystem;

import java.awt.Color;
import javax.swing.JButton;
import javax.swing.JOptionPane;


public class home extends javax.swing.JFrame {


    public home() {
        initComponents();

    }

  private void logOutMouseClicked(java.awt.event.MouseEvent evt) {
        int logout = JOptionPane.showConfirmDialog(null, "Log out?", "Select",
JOptionPane.YES_NO_CANCEL_OPTION);
```

```
if (logout == 0) {
            setVisible(false);
            new login().setVisible(true);
        }

    }

    private void addMemberMouseClicked(java.awt.event.MouseEvent evt) {
        setVisible(false);
        new newMember().setVisible(true);
    }

    private void editMemberMouseClicked(java.awt.event.MouseEvent evt) {
        setVisible(false);
        new editMember().setVisible(true);
    }

    private void trainersMouseClicked(java.awt.event.MouseEvent evt) {
        setVisible(false);
        new trainer().setVisible(true);
    }

    private void paymentsMouseClicked(java.awt.event.MouseEvent evt) {
        setVisible(false);
        new memberList().setVisible(true);
    }

    private void payments1MouseClicked(java.awt.event.MouseEvent evt) {
        setVisible(false);
        new payment().setVisible(true);
    }

    private void revenueBtnActionPerformed(java.awt.event.ActionEvent evt) {

        new RevenueGenerator().setVisible(true);
    }


    public static void main(String args[]) {

        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new home().setVisible(true);
            }
        });
    }
```

#Login form to add member

```
package com.mycompany.gymmanagementsystem;
import com.mysql.cj.xdevapi.Result;
import javax.swing.*;
import database.ConnectionProvider;
import java.awt.Color;
import java.sql.*;
import java.util.regex.Pattern;

public class newMember extends javax.swing.JFrame {


    public newMember() {
        initComponents();
        trainer();
        fnEmpty.setVisible(false);
        lnEmpty.setVisible(false);
        addressEmpty.setVisible(false);
        phoneNumsEmpty.setVisible(false);
        trainerList.setVisible(false);

            int id = 1;
        try{
            String str1 = String.valueOf(id);
            memberID.setText("00"+str1);

            Connection connection = ConnectionProvider.getConnection();

            Statement memberSt = connection.createStatement();
            String sqlMember = "SELECT COUNT(id) FROM member";
            ResultSet result = memberSt.executeQuery(sqlMember);

            while(result.next()){
                id = result.getInt(1);
                id = id +1;
                String str = String.valueOf(id);
                memberID.setText("00"+str);
            }
        }catch(Exception e){
            JOptionPane.showMessageDialog(this, "praveen");
        }
    }
    private void saveBtnActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String emailRegex = "^(.+)@(.+)$";
        String fn = firstName.getText().toString();
        String ln = lastName.getText().toString();
        String phonenums = phoneNums.getText().toString();
        String em = email.getText().toString();
        String addr = address.getText().toString();


        if(fn.isEmpty()){
            fnEmpty.setVisible(true);
        }else{
            fnEmpty.setVisible(false);
        }

        if(ln.isEmpty()){
            lnEmpty.setVisible(true);
        }else{
            lnEmpty.setVisible(false);
        }

        if(phonenums.isEmpty()){
            phoneNumsEmpty.setVisible(true);
        }else{
            phoneNumsEmpty.setVisible(false);
        }

        if(em.isEmpty()){
            emailEmpty.setText("Email is required!");
        }else if ( !((Pattern.compile(emailRegex)).matcher(em).matches())){
            emailEmpty.setText("Email is not valid!");
        }
        else{
            emailEmpty.setText("");
        }

        if(addr.isEmpty()){
            addressEmpty.setVisible(true);
        }else{
            addressEmpty.setVisible(false);
        }

        String id = memberID.getText().toString();
        String memType = (String)memberType.getSelectedItem();
        String gen = (String)gender.getSelectedItem();
        String amount = amountPay.getText().toString();
        String trainer;
        if (memType.equals("Basic")){
            trainer = "none";
        }else{
            trainer = (String)trainerList.getSelectedItem();
        }
        java.util.Date date = jDateChooser2.getDate();
        java.sql.Date dateRegister = new java.sql.Date(date.getTime());

        try{

            Connection connection = ConnectionProvider.getConnection();
            String sql = "INSERT INTO member VALUES (?,?,?,?,?,?,?,?,?,?,?,?)";


            PreparedStatement statement = connection.prepareStatement(sql);
            statement.setString(1, id);
            statement.setString(2, fn);
            statement.setString(3, ln);
            statement.setString(4, gen);
            statement.setString(5, phonenums);
            statement.setString(6, em);
            statement.setString(7, addr);
            statement.setString(8, amount);
            statement.setString(9, memType);
            statement.setDate(10, dateRegister);
            statement.setString(11, trainer);
            statement.setString(12,null);
            statement.executeUpdate();



            JOptionPane.showMessageDialog(null, "Save member successfully!");
            setVisible(false);
```

```
            new newMember().setVisible(true);

        }catch(Exception e){
            JOptionPane.showMessageDialog(null, "Error to save
information","Error",JOptionPane.PLAIN_MESSAGE);
            e.printStackTrace();
        }

        int searchid = 0;

        try{
            searchid++;
            String sqlMember = "SELECT * FROM member where id='"+id+"'";
            Connection connection = ConnectionProvider.getConnection();
            Statement memberSt = connection.createStatement();
            ResultSet result = memberSt.executeQuery(sqlMember);

            while (result.next()){

                searchid = 1;
                String payID = result.getString(1);
                String payName = result.getString(2) + " " + result.getString(3);
                String payType = result.getString(9);
                Double payAmount = result.getDouble(8);

                String payDate = result.getString(12);


                PreparedStatement paymentSt = connection.prepareStatement("INSERT INTO
payment (ID, memberName, memberType, amountPay) VALUES (?,?,?,?)");
                paymentSt.setString(1, payID);
                paymentSt.setString(2, payName);
                paymentSt.setString(3, payType);
                paymentSt.setDouble(4, payAmount);

                paymentSt.executeUpdate();
            }
            if(searchid == 0){
                JOptionPane.showMessageDialog(null, "Member ID does not exist!");
            }
        }catch(Exception e){
            JOptionPane.showMessageDialog(null, e);
        }
```

```
    }

    private void trainer(){
        try{

            Connection connection = ConnectionProvider.getConnection();
            String sql = "SELECT DISTINCT name FROM trainer";

            PreparedStatement statement = connection.prepareStatement(sql);
            ResultSet result = statement.executeQuery();
            trainerList.removeAllItems();

            while (result.next()){
                trainerList.addItem(result.getString("name"));
            }

        }catch(Exception e){

            e.printStackTrace();
        }
    }
    private void resetBtnActionPerformed(java.awt.event.ActionEvent evt) {
        firstName.setText("");
        lastName.setText("");
        phoneNums.setText("");
        address.setText("");
        amountPay.setText("9.99");
        email.setText("");
        memberType.setSelectedIndex(0);
    }

    private void exitBtnActionPerformed(java.awt.event.ActionEvent evt) {
        int exit = JOptionPane.showConfirmDialog(null, "Exit to 
Dashboard?","Select",JOptionPane.YES_NO_CANCEL_OPTION);
        if (exit == 0){
            setVisible(false);
            new home().setVisible(true);
        }
    }

    private void memberTypeItemStateChanged(java.awt.event.ItemEvent evt) {
        String memType = (String)memberType.getSelectedItem();
        if (memType.equals("Basic")){
            amountPay.setText("599");
        }else if (memType.equals("Plus")){
            amountPay.setText("999");
        }else if (memType.equals("Premium")){
            amountPay.setText("1299");
        }
    }

    private void memberTypeActionPerformed(java.awt.event.ActionEvent evt) {
        String memType = (String)memberType.getSelectedItem();
        if (memType.equals("Basic")){
            trainerList.setVisible(false);
        }else{
            trainerList.setVisible(true);
        }
    }

    private void amountPayActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
    }

    public static void main(String args[]) {

        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(newMember.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(newMember.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);
        } catch (IllegalAccessException ex) {
```

```
java.util.logging.Logger.getLogger(newMember.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(newMember.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new newMember().setVisible(true);
            }
        });
    }
```

#Dashboard To Manage Members

```
package com.mycompany.gymmanagementsystem;

import database.ConnectionProvider;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.Statement;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;


public class memberList extends javax.swing.JFrame {

    public memberList() {
        initComponents();
        DefaultTableModel model  =(DefaultTableModel)jTable1.getModel();
        String id =null;
        String memberName=null;
        String trainerName=null;
        String memberType=null;
        double amountPay=0.0;
        String paymentDate = null;
        String dueDate = null;
        String dayRemaining = null;
        String status = null;

        try{
            int checkid=0;

            String sqlMember = "SELECT member.ID, member.firstName, member.lastName, 
member.trainer, member.memberType, "
                    + "member.amountPay, payment.paymentDate, payment.dueDate, 
payment.dayRemaining, payment.status "
                    + "from gmsdb_dump.member, gmsdb_dump.payment WHERE member.ID = 
payment.ID;";

            Connection connection = ConnectionProvider.getConnection();
            Statement memberSt = connection.createStatement();
            ResultSet result= memberSt.executeQuery(sqlMember);

            while (result.next()){
                checkid = 1;
                id = result.getString(1);
                memberName = result.getString(2) + " " + result.getString(3);
                trainerName = result.getString(4);
                memberType = result.getString(5);
                amountPay = result.getDouble(6);
                paymentDate = result.getString(7);
                dueDate = result.getString(8);
                dayRemaining = result.getString(9);
                status = result.getString(10);

                model.addRow(new Object[]{id,memberName,trainerName,memberType,amountPay, 
paymentDate, dueDate, dayRemaining, status});
            }
        }catch(Exception e){
            JOptionPane.showMessageDialog(null, e);
        }
    }
    private void exitBtnActionPerformed(java.awt.event.ActionEvent evt) {
        int exit = JOptionPane.showConfirmDialog(null, "Exit to 
Dashboard?","Select",JOptionPane.YES_NO_CANCEL_OPTION);
        if (exit == 0){
            setVisible(false);
            new home().setVisible(true);
        }
    }


    public static void main(String args[]) {

        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(memberList.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(memberList.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(memberList.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(memberList.class.getName()).log(java.util.logging.Level.SE
VERE, null, ex);
        }
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new memberList().setVisible(true);
            }
        });
    }
```

#Payment page

```
package com.mycompany.gymmanagementsystem;

import database.ConnectionProvider;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.Statement;
import java.text.SimpleDateFormat;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;


public class payment extends javax.swing.JFrame {


    private void initTable(){
        DefaultTableModel model  =(DefaultTableModel)jTable1.getModel();

        ID.setText("");
        name.setText("");
        type.setText("");
        pay.setText("");
        date.setText("");
        due.setText("");
        dRemaining.setText("");
        status.setText("");

        String id = null;
        String memberName=null;
        String memberType=null;
        String paymentDate = null;
        String due = null;
        double amountPay=0.0;
        String strStatus = null;
        String dayRemaining = null;


        try{
            int checkid = 0;

            Connection connection = ConnectionProvider.getConnection();
            Statement paymentSt=connection.createStatement();
            String sqlPayment = "SELECT * FROM payment";
            ResultSet result=paymentSt.executeQuery(sqlPayment);

            while (result.next()){
              checkid = 1;
              id = result.getString(1);
              memberName = result.getString(2);
              memberType = result.getString(3);
              amountPay = result.getDouble(4);
              paymentDate = result.getString(5);
              due = result.getString(6);
              dayRemaining = result.getString(7);
              strStatus = result.getString(8);
              int dayDiff = 0;

              if (due == null){
                due = "null";
              }else{
                java.util.Date dDue = new SimpleDateFormat("yyyy-MM-dd").parse(due);
                java.util.Date currDate = new java.util.Date();

                long diff = dDue.getTime() - currDate.getTime();
                long diffDays = diff / (24*60*60*1000) + 1;
                dayDiff = (int)diffDays;
              }
              if(dayDiff<=0)
              {
                strStatus="Not Paid";
                dayRemaining = "null";

              }

              if (paymentDate == null){
                strStatus = "Not Paid";
                paymentDate = "null";
                due = "null";
                dayRemaining = "null";
              }


              model.addRow(new
      Object[]{id,memberName,memberType,amountPay,paymentDate, due, dayDiff + " Days",
      strStatus });
```

```
            sqlPayment = "UPDATE payment SET dayRemaining=?, status=?  WHERE id=?";

            Connection connection1 = ConnectionProvider.getConnection();
            PreparedStatement paymentStUpdate=connection.prepareStatement(sqlPayment);
            paymentStUpdate.setString(1,String.valueOf(dayDiff) + " Days");
            paymentStUpdate.setString(2,strStatus);
            paymentStUpdate.setString(3,id);


            paymentStUpdate.executeUpdate();

        }

    }catch(Exception e){
        JOptionPane.showMessageDialog(null, e);
        e.printStackTrace();
    }
}

public payment() {
    initComponents();

    java.util.Date getToday = new java.util.Date();
    String strToday = new SimpleDateFormat("yyyy-MM-dd").format(getToday);
    today.setText(strToday);

    initTable();

}
private void searchBtnMouseClicked(java.awt.event.MouseEvent evt) {
    int searchid = 0;
    String id = searchField.getText();
    DefaultTableModel model  =(DefaultTableModel)jTable1.getModel();
    try{
        String sqlPayment = "SELECT * FROM payment WHERE id='"+id+"'";
        Connection connection = ConnectionProvider.getConnection();
        Statement paymentSt = connection.createStatement();
        ResultSet result = paymentSt.executeQuery(sqlPayment);

        while (result.next()){
            searchid = 1;
            model.setRowCount(0);

            ID.setText("00"+result.getString(1));
            name.setText(result.getString(2));
```

```
                type.setText(result.getString(3));
                pay.setText(result.getString(4));

                String pDate = result.getString(5);
                if (pDate == null){
                    pDate = "null";
                }
                String pDue = result.getString(6);
                if(pDue == null){
                    pDue = "null";
                }
                String pDRemaining = result.getString(7);
                if(pDRemaining == null){
                    pDRemaining = "null";
                }
                String pStatus = result.getString(8);
                if (pStatus == null){
                    pStatus = "null";
                }

                date.setText(pDate);
                due.setText(pDue);
                dRemaining.setText(pDRemaining);
                status.setText(pStatus);

                model.addRow(new Object[]{result.getString(1), result.getString(2),
result.getString(3), result.getString(4), pDate, pDue, pDRemaining, pStatus});
            }
            if(searchid == 0){
                JOptionPane.showMessageDialog(null, "Member ID does not exist!");
            }
        }catch(Exception e){
            JOptionPane.showMessageDialog(null, e);
        }

    }

    private void resetBtnMouseClicked(java.awt.event.MouseEvent evt) {
        DefaultTableModel model  =(DefaultTableModel)jTable1.getModel();
        model.setRowCount(0);
        ID.setText("");
        name.setText("");
        type.setText("");
        pay.setText("");
        date.setText("");
        due.setText("");
        dRemaining.setText("");
        status.setText("");

        try{
            String sqlPayment = "SELECT * FROM payment";
            Connection connection = ConnectionProvider.getConnection();
            Statement paymentSt = connection.createStatement();
            ResultSet result= paymentSt.executeQuery(sqlPayment);

            while (result.next()){
                String pDate = result.getString(5);
                if (pDate == null){
                    pDate = "null";
                }
                String pDue = result.getString(6);
                if(pDue == null){
                    pDue = "null";
                }
                String pDRemaining = result.getString(7);
                if(pDRemaining == null){
                    pDRemaining = "null";
                }
                String pStatus = result.getString(8);
                if (pStatus == null){
                    pStatus = "null";
                }
                model.addRow(new Object[]{result.getString(1), result.getString(2),
result.getString(3), result.getString(4), pDate, pDue, pDRemaining , pStatus});
            }
        }catch(Exception e){
            JOptionPane.showMessageDialog(null, e);
        }
    }

    private void payBtnMouseClicked(java.awt.event.MouseEvent evt) {
        String id = ID.getText();
        String payDate = today.getText();

        int newMonth = Integer.parseInt(payDate.substring(6,7)) + 1;
        String dueDate = payDate.substring(0,4) + "-0" + Integer.toString(newMonth) + "-" +
payDate.substring(8,10);

        String status = "Paid";
        try{

            java.util.Date dDue = new SimpleDateFormat("yyyy-MM-dd").parse(dueDate);
            java.util.Date currDate = new java.util.Date();
            int dayDiff = 0;
            long diff = dDue.getTime() - currDate.getTime();
            long diffDays = diff / (24*60*60*1000) + 1;
            dayDiff = (int)diffDays;

            String daysRemaining = String.valueOf(dayDiff);

            String sqlMember = "UPDATE member SET payDate=? WHERE id=?;";
            String sqlPayment = "UPDATE payment SET paymentDate=?, dueDate=?,
dayRemaining=?, status=? WHERE id=?";

            Connection connection = ConnectionProvider.getConnection();

            PreparedStatement memberSt=connection.prepareStatement(sqlMember);
            PreparedStatement paymentSt=connection.prepareStatement(sqlPayment);

            memberSt.setString(1,payDate);
            memberSt.setString(2,id);

            paymentSt.setString(1,payDate);
            paymentSt.setString(2,dueDate);
            paymentSt.setString(3, daysRemaining + " Days");
            paymentSt.setString(4,status);
            paymentSt.setString(5, id);

            memberSt.executeUpdate();
            paymentSt.executeUpdate();

            JOptionPane.showMessageDialog(null, "Successfully Paid!");
            setVisible(false);
            new payment().setVisible(true);

        }catch(Exception e){
            JOptionPane.showMessageDialog(null, e);
            e.printStackTrace();
        }
    }

    private void exitBtnActionPerformed(java.awt.event.ActionEvent evt) {
        int exit = JOptionPane.showConfirmDialog(null, "Exit to
Dashboard?","Select",JOptionPane.YES_NO_CANCEL_OPTION);
```

```
        if (exit == 0){
            setVisible(false);
            new home().setVisible(true);
        }
    }
    public static void main(String args[]) {

        try {
            for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
                if ("Nimbus".equals(info.getName())) {
                    javax.swing.UIManager.setLookAndFeel(info.getClassName());
                    break;
                }
            }
        } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(payment.class.getName()).log(java.util.logging.Level.SEVE
RE, null, ex);
        } catch (InstantiationException ex) {

java.util.logging.Logger.getLogger(payment.class.getName()).log(java.util.logging.Level.SEVE
RE, null, ex);
        } catch (IllegalAccessException ex) {

java.util.logging.Logger.getLogger(payment.class.getName()).log(java.util.logging.Level.SEVE
RE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {

java.util.logging.Logger.getLogger(payment.class.getName()).log(java.util.logging.Level.SEVE
RE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new payment().setVisible(true);
            }
        });
    }
```

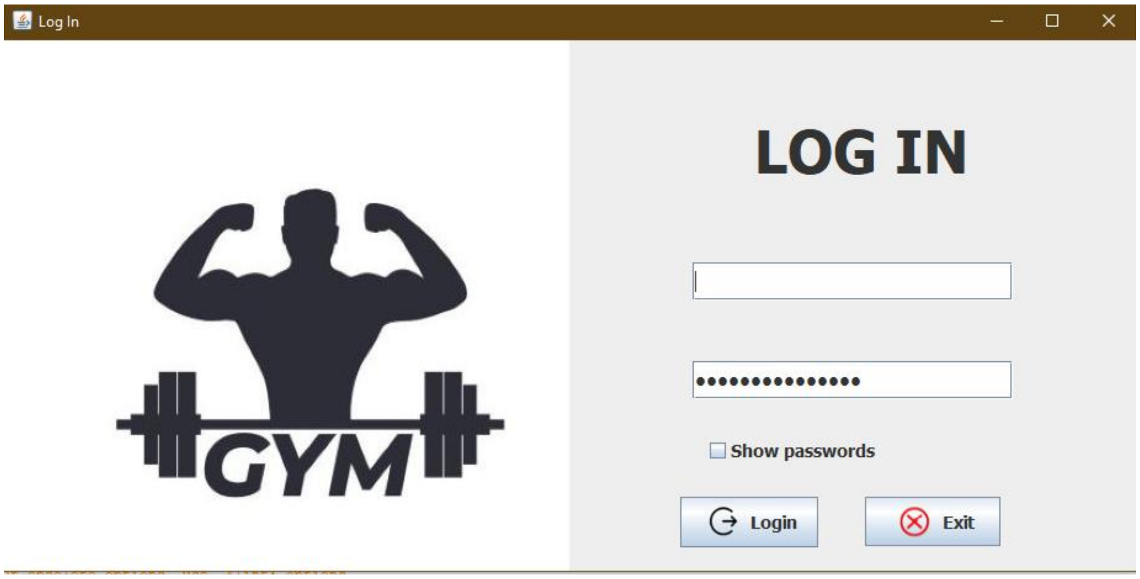
#Calculation Page

```
package com.mycompany.gymmanagementsystem;


import database.ConnectionProvider;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.text.SimpleDateFormat;
import javax.swing.JOptionPane;



public class RevenueGenerator extends javax.swing.JFrame {

   public RevenueGenerator() {
      initComponents();
      lblRevenue.setVisible(false);
   }
   private void btnGenerateActionPerformed(java.awt.event.ActionEvent evt) {

      SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");
      String fromDate = sdf.format(jDateChooser2.getDate());
      String toDate = sdf.format(jDateChooser3.getDate());

      try {
         Connection connection = ConnectionProvider.getConnection();
         String sql = "SELECT SUM(amountPay) FROM payment WHERE paymentDate
BETWEEN ? AND ?";
         PreparedStatement ps = connection.prepareStatement(sql);
         ps.setString(1, fromDate);
         ps.setString(2, toDate);
```

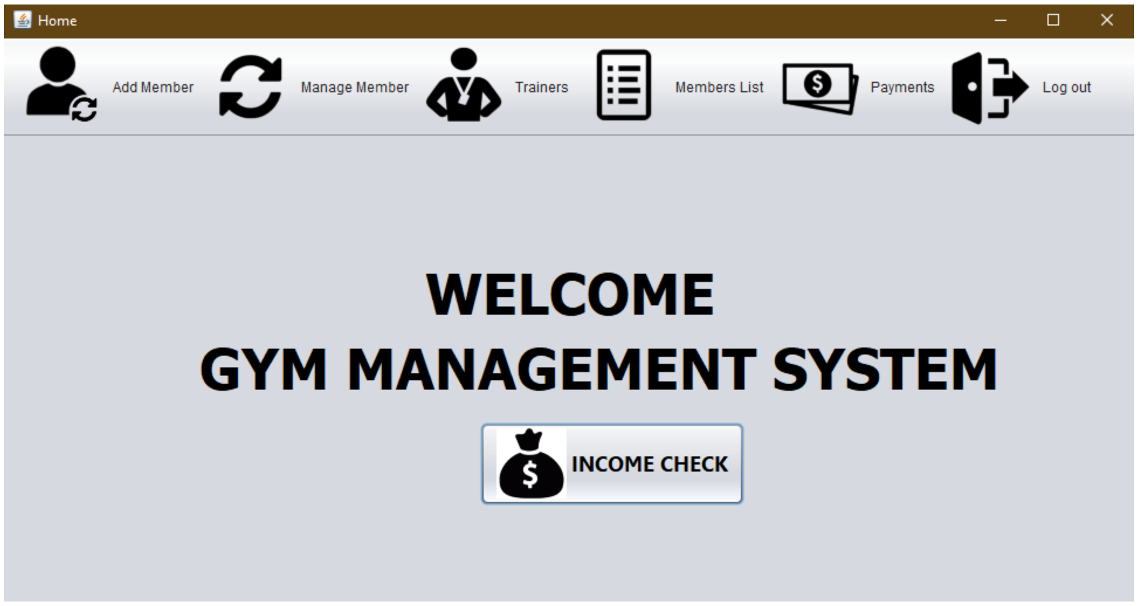
```
            ResultSet rs = ps.executeQuery();

            if (rs.next()) {
                double totalRevenue = rs.getDouble(1);
                lblRevenue.setText("Total Revenue: ₹" + totalRevenue);
                lblRevenue.setVisible(true);
            } else {
                lblRevenue.setText("No records found.");
                lblRevenue.setVisible(true);
            }
        } catch (Exception e) {
            JOptionPane.showMessageDialog(this, "Error: " + e.getMessage());
        }

    }


public static void main(String args[]) {

        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new RevenueGenerator().setVisible(true);
            }
        });
    }
```

8. OUTPUT

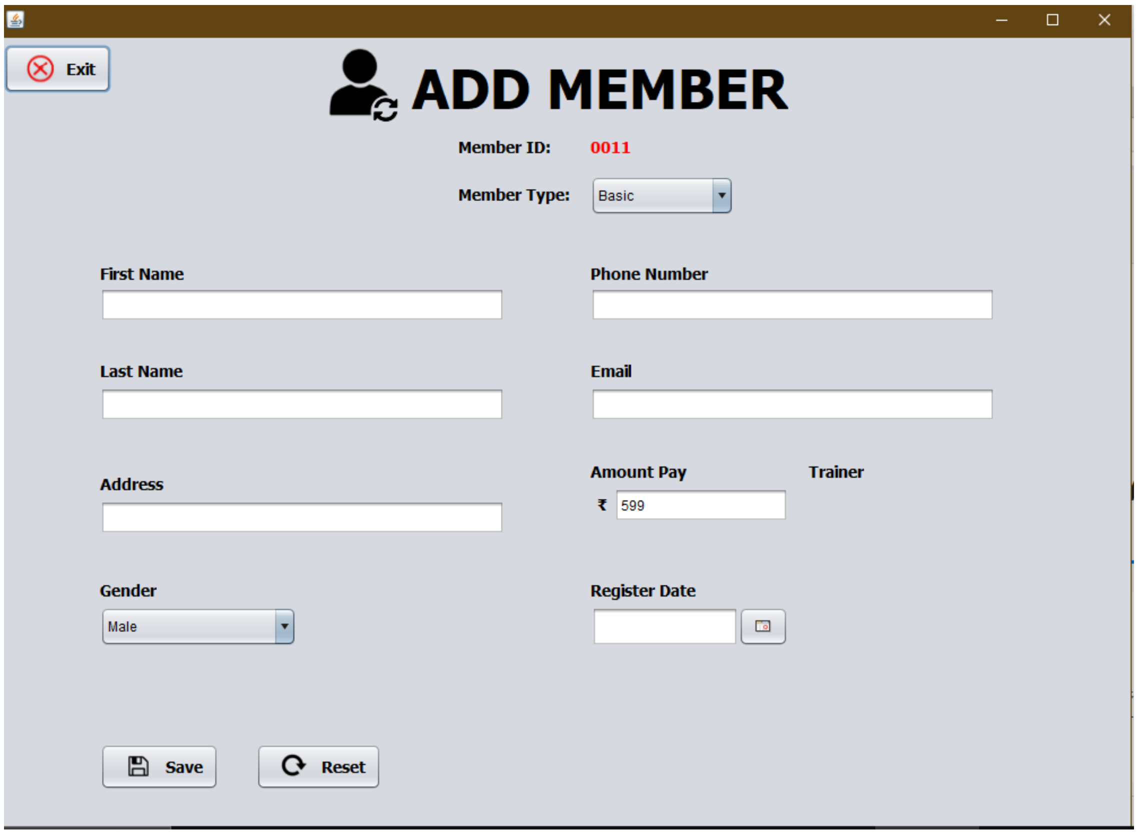
1. Welcoming Login page



2. Home page for quick navigation.



3. This image shows new members registration form

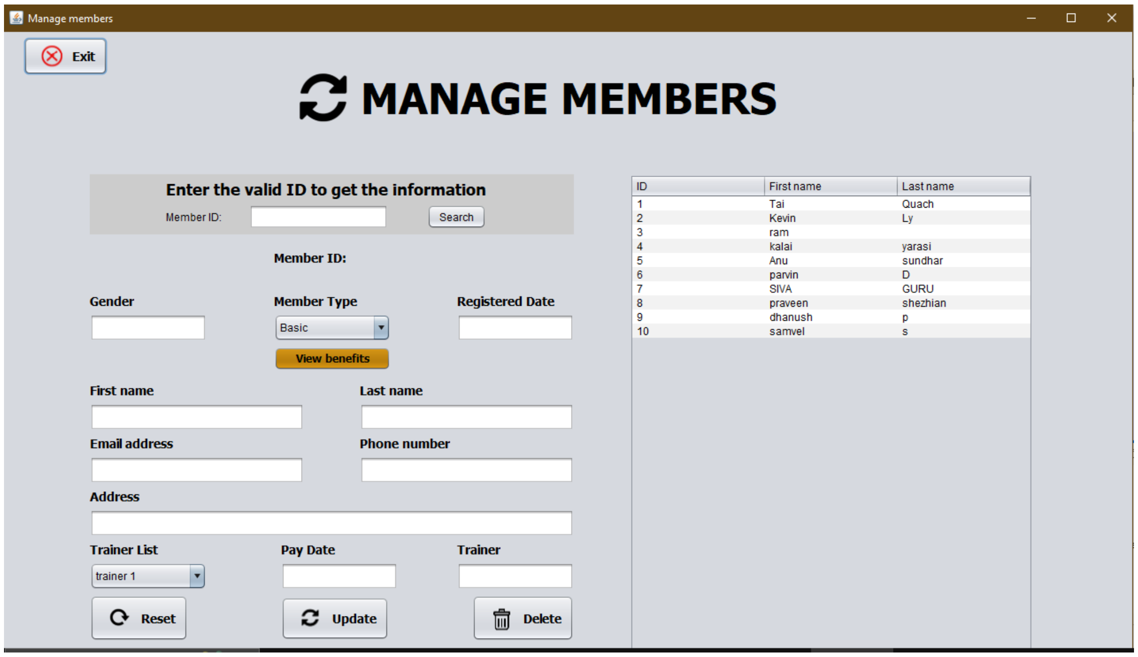


Member ID: 0011

Member Type: Basic

First Name	Phone Number
Last Name	Email
Address	Amount Pay: ₹ 599
	Trainer
Gender: Male	Register Date

4. This page to modify member records.



Enter the valid ID to get the information

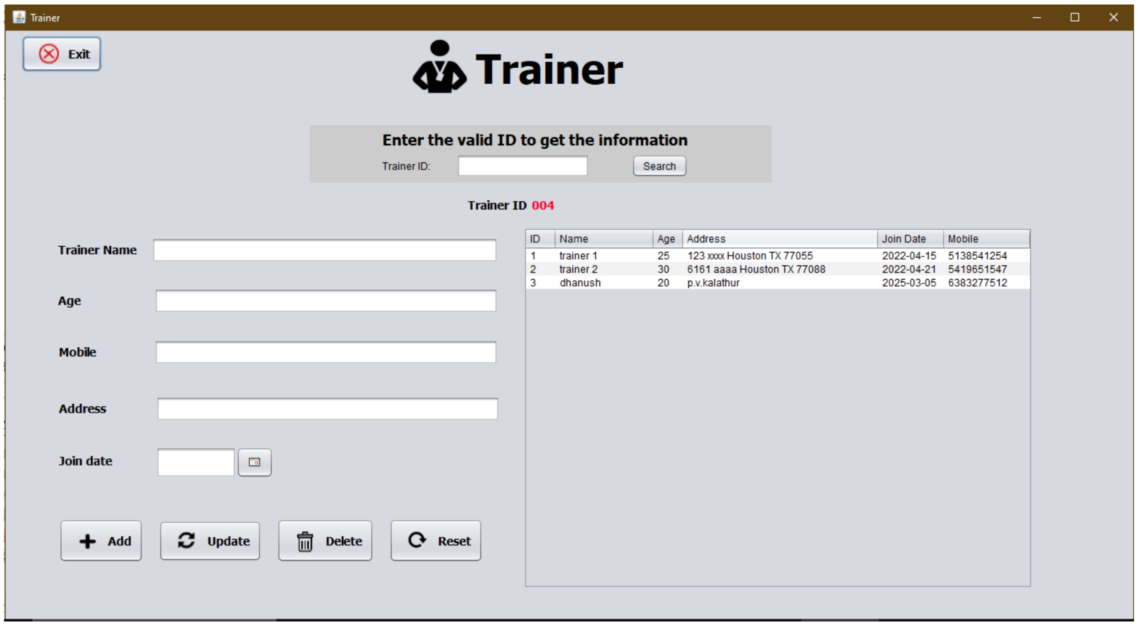
Member ID:

Member ID:

Gender	Member Type: Basic	Registered Date
First name	Last name	
Email address	Phone number	
Address		
Trainer List: trainer 1	Pay Date	Trainer

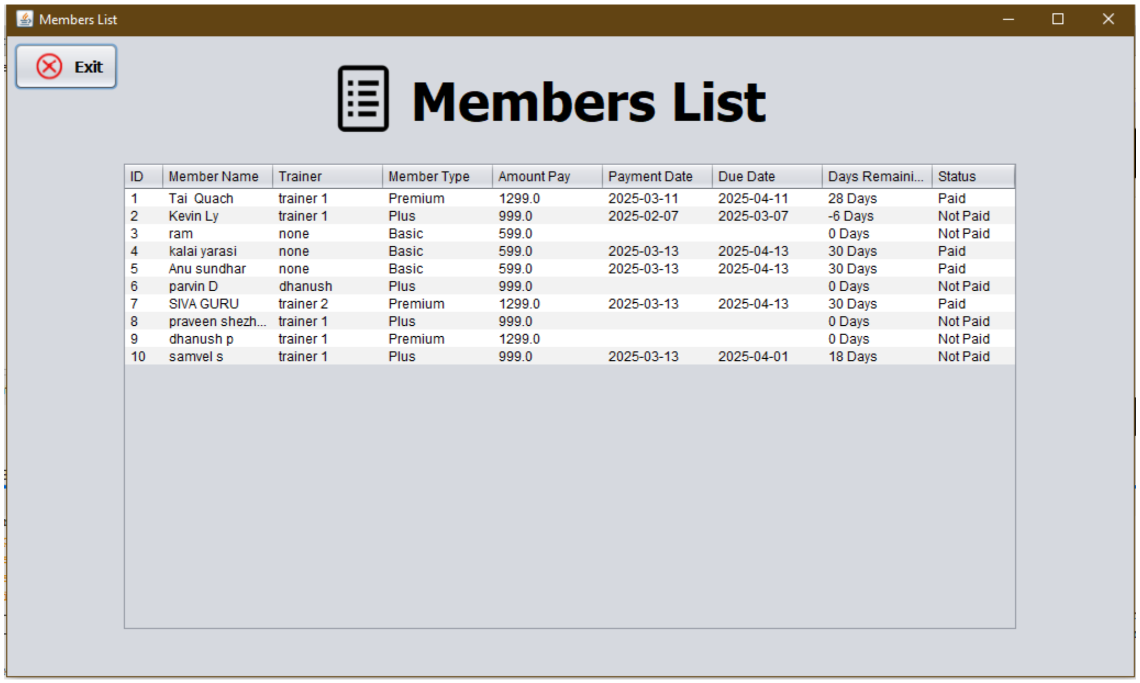
ID	First name	Last name
1	Tai	Quach
2	Kevin	Ly
3	ram	
4	kalai	yarasi
5	Anu	sundhar
6	parvin	D
7	SIVA	GURU
8	praveen	shezhian
9	dhanush	p
10	samvel	s

5. New trainer are added through this page.



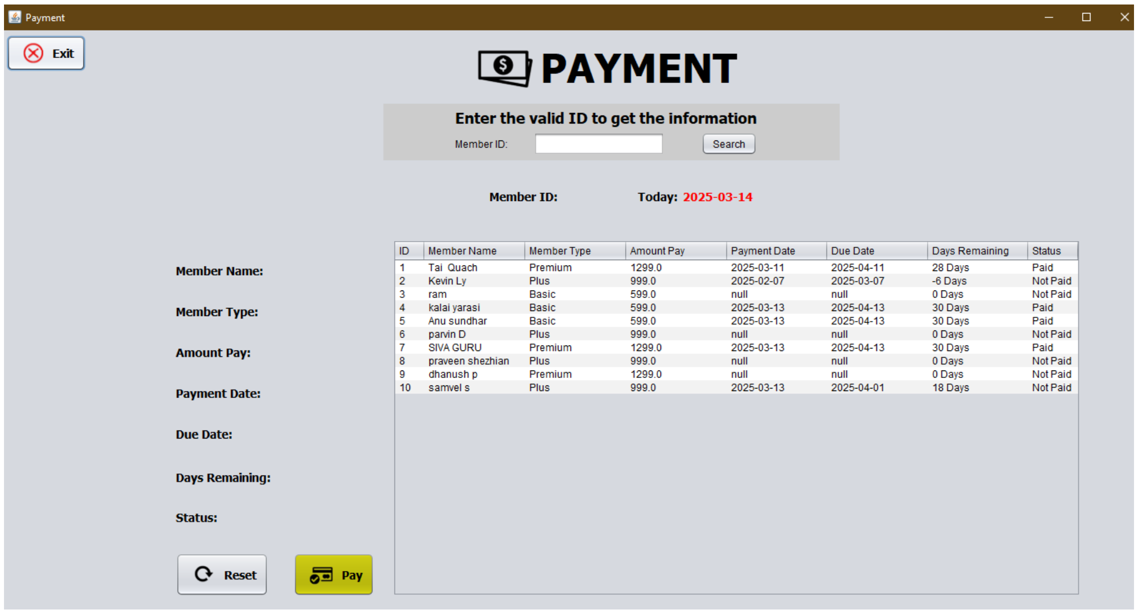
ID	Name	Age	Address	Join Date	Mobile
1	trainer 1	25	123 xxxx Houston TX 77055	2022-04-15	5138541254
2	trainer 2	30	6161 aaaa Houston TX 77088	2022-04-21	5419651547
3	dhanush	20	p.v.kalathur	2025-03-05	6383277512

6. Member history is listed here.



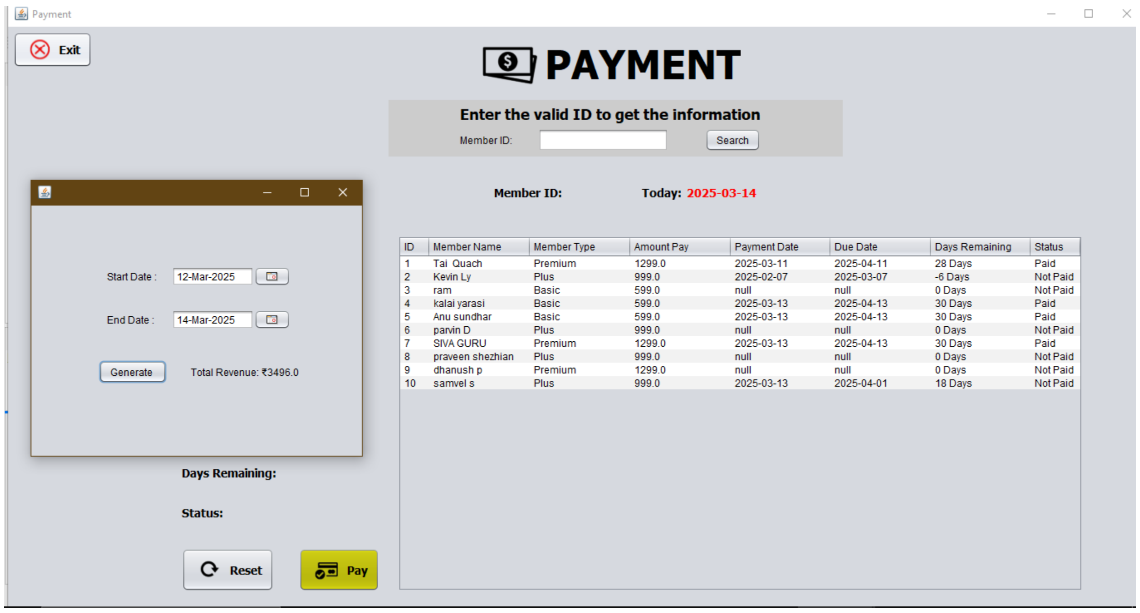
ID	Member Name	Trainer	Member Type	Amount Pay	Payment Date	Due Date	Days Remaini...	Status
1	Tai Quach	trainer 1	Premium	1299.0	2025-03-11	2025-04-11	28 Days	Paid
2	Kevin Ly	trainer 1	Plus	999.0	2025-02-07	2025-03-07	-6 Days	Not Paid
3	ram	none	Basic	599.0			0 Days	Not Paid
4	kalai yarasi	none	Basic	599.0	2025-03-13	2025-04-13	30 Days	Paid
5	Anu sundhar	none	Basic	599.0	2025-03-13	2025-04-13	30 Days	Paid
6	parvin D	dhanush	Plus	999.0			0 Days	Not Paid
7	SIVA GURU	trainer 2	Premium	1299.0	2025-03-13	2025-04-13	30 Days	Paid
8	praveen shezh...	trainer 1	Plus	999.0			0 Days	Not Paid
9	dhanush p	trainer 1	Premium	1299.0			0 Days	Not Paid
10	samvel s	trainer 1	Plus	999.0	2025-03-13	2025-04-01	18 Days	Not Paid

7. Make payment for the members



ID	Member Name	Member Type	Amount Pay	Payment Date	Due Date	Days Remaining	Status
1	Tai Quach	Premium	1299.0	2025-03-11	2025-04-11	28 Days	Paid
2	Kevin Ly	Plus	999.0	2025-02-07	2025-03-07	-6 Days	Not Paid
3	ram	Basic	599.0	null	null	0 Days	Not Paid
4	kalai yarasi	Basic	599.0	2025-03-13	2025-04-13	30 Days	Paid
5	Anu sundhar	Basic	599.0	2025-03-13	2025-04-13	30 Days	Paid
6	parvin D	Plus	999.0	null	null	0 Days	Not Paid
7	SIVA GURU	Premium	1299.0	2025-03-13	2025-04-13	30 Days	Paid
8	praveen shezhian	Plus	999.0	null	null	0 Days	Not Paid
9	dhanush p	Premium	1299.0	null	null	0 Days	Not Paid
10	samvel s	Plus	999.0	2025-03-13	2025-04-01	18 Days	Not Paid

8. Calculating revenue generation.



Start Date : 12-Mar-2025

End Date : 14-Mar-2025

Total Revenue: ₹3496.0

ID	Member Name	Member Type	Amount Pay	Payment Date	Due Date	Days Remaining	Status
1	Tai Quach	Premium	1299.0	2025-03-11	2025-04-11	28 Days	Paid
2	Kevin Ly	Plus	999.0	2025-02-07	2025-03-07	-6 Days	Not Paid
3	ram	Basic	599.0	null	null	0 Days	Not Paid
4	kalai yarasi	Basic	599.0	2025-03-13	2025-04-13	30 Days	Paid
5	Anu sundhar	Basic	599.0	2025-03-13	2025-04-13	30 Days	Paid
6	parvin D	Plus	999.0	null	null	0 Days	Not Paid
7	SIVA GURU	Premium	1299.0	2025-03-13	2025-04-13	30 Days	Paid
8	praveen shezhian	Plus	999.0	null	null	0 Days	Not Paid
9	dhanush p	Premium	1299.0	null	null	0 Days	Not Paid
10	samvel s	Plus	999.0	2025-03-13	2025-04-01	18 Days	Not Paid

FUTURE ENHANCEMENTS

1. **Revenue Generator Enhancements:**
   - Monthly and yearly revenue trends.
   - Membership retention vs. dropout rates.
   - Expense tracking for gym maintenance & staff salaries.

2. **Advanced Membership & Subscription Plans:**
   - Introduce auto-renewal options for memberships.
   - Allow members to pause or upgrade their subscription plans.

3. **Mobile App Development**
   - Member registration & payment.
   - Workout tracking & trainer communication.
   - Push notifications for reminders & offers.

4. **Attendance & Check-in System:**
   - Implement RFID or QR-based check-ins for members.
   - Track trainer availability & member attendance automatically.

5. **Automated Billing & Online Payment Integration:**
   - Add UPI, credit card, and PayPal payment gateways for hassle-free payments
   - Generate e-receipts and send payment confirmation emails.

6. **Trainer Performance Monitoring:**
   - Track trainer sessions, member feedback, and performance ratings.
   - Generate trainer salary reports based on attendance & sessions completed.

CONCLUSION

The Gym Management System is a comprehensive and efficient solution designed to automate and streamline gym operations. By integrating modules for member registration, trainer management, payment tracking, and revenue generation, this system enhances operational efficiency, reduces manual workload, and improves overall management.

With features like secure login authentication, automated payment tracking, and member-trainer mapping, the system ensures a seamless experience for both gym members and administrators. Additionally, the Revenue Generator module provides valuable financial insights, helping gym owners make data-driven decisions to maximize profitability.

Future enhancements, such as mobile app integration, AI-driven workout recommendations, online payments, and biometric check-ins, can further improve the system's scalability and usability. Implementing cloud-based storage and wearable device integration will make the system more accessible and tech-driven.

In conclusion, the Gym Management System serves as a reliable, scalable, and user-friendly platform, ensuring efficient gym administration and enhanced member satisfaction. By incorporating advanced features in future iterations, this system can evolve into a complete digital fitness ecosystem, catering to modern gym management needs.

BIBLIOGRAPHY

1. Apache NetBeans IDE 24. Available at: https://netbeans.apache.org

- Used for develop Graphical user interface For building and managing the Java-based application.

2. MySQL. Available at: https://www.mysql.com/

- Used as the Integrated Development Environment (IDE) to develop and manage project

files**.**

3. GeeksforGeeks: Available at: https://www.geeksforgeeks.org

- Referenced for java swing

4. W3Schools. Available at: https://www.w3schools.com/sql

- Referenced for SQL queries and database design

5. Book. "Herbert Schildt " Available at: Library

- Referenced Herbert Schildt – *Java: The Complete Reference*, for better quick information

6. Java Available at: https://www.java.com/en/download/manual.jsp

7. JCalendar (jar) Available at: https://toedter.com/jcalendar/

- Used for calendar date picking functionality