**GYM MEMBERSHIP MANAGEMENT SYSTEM**

**Introduction:**  
The Gym Membership Management System is designed to help gyms efficiently manage member details, membership plans, and attendance records. By organizing member information digitally, the system reduces manual work, minimizes errors, and ensures data consistency. Using Java, JDBC, and exception handling, it supports tasks like adding, updating, and retrieving member data while preventing duplicate entries through custom exceptions. This solution provides a streamlined, reliable approach to handling gym operations, enhancing both data accuracy and administrative efficiency.

**Objective:**

* Create a gym membership management application to add, update, and

track members’ attendance and membership plans.

• Implement inheritance for different membership types, collections for

tracking attendance, and JDBC for storing member information.

**Platform Specification:**

**HARDWARE:**

INTEL CORE 13,15,17 OR MAC,MINIMUM 4GB RAM

**SOFTWARE:**

JAVA,SQL,JDBC,JIRA

**Functional requirements:**

**1. Member Management:**

* Allow adding a new gym member, capturing details like name, contact information, address, membership type, and start and expiry dates.
* Enable updating existing member information, such as contact details or membership status.
* Fetch and display a member’s information using their unique ID or other identifiers.

**2. Membership Management:**

* **Manage Membership Plans:** Define different types of memberships, including their costs, durations, and specific benefits.
* **Check Membership Validity:** Automatically validate if a member’s membership has expired based on the membership expiry date.

**3. Attendance Tracking:**

* **Log Daily Attendance:** Allow staff to log each member’s attendance daily and maintain a record of attendance history.
* **Retrieve Attendance Records:** Enable retrieving a member’s attendance records over a specific period for monitoring and analysis.

**4.** **Payments and Transactions:**

* **Record Membership Payments:** Track payments made by members for their memberships, including amount, payment date, and method

**5. Error Handling:**

* **Invalid Input Handling:** Validate input data for each field (e.g., email format, phone number length) and prompt appropriate error messages for invalid data.
* **Exception Management:** Handle JDBC and file I/O exceptions to ensure smooth application functioning and data integrity.

**6. File Management:**

* **Data Storage in Database:** Use a database (via JDBC) to persistently store member details, attendance records, membership plans, and payments.

**Schema diagram:**

**Members:** Stores information about gym members.

* **Columns:**
  + **member\_id: INT (PK, AUTO\_INCREMENT) -** Unique ID for each membercustomer\_id: INT
  + **first\_name: VARCHAR(50) -** First name of the member
  + **last\_name: VARCHAR(50) -** Last name of the member
  + **email: VARCHAR(100) -** Email address of the member (Unique)
  + **phone: VARCHAR(15)** - Contact number of the member
  + **address: VARCHAR(255) -** Address of the member
  + **membership\_type: VARCHAR(50) -** Type of membership (e.g., Regular, Premium)
  + **join\_date: DATE -** Date when the membership started
  + **membership\_expiry: DATE -** Date when the membershipexpires

**MembershipPlans**: Defines various membership plans available.

* **Columns:**
  + **plan\_id: INT (PK, AUTO\_INCREMENT) -** Unique ID for each plan
  + **plan\_name: VARCHAR(50) -** Name of the membership plan
  + **duration\_months: INT -** Duration of the plan in months
  + **monthly\_fee: DECIMAL(10, 2) -** Cost of the plan
  + **features: TEXT -** Description of the plan features.

**Attendance:** Tracks daily attendance for each member.

* **Columns:**
  + **attendance\_id: INT (PK, AUTO\_INCREMENT) -** Unique ID for each attendance record
  + **member\_id: INT (FK) -** Reference to the member\_id in Members table
  + **attendance\_date: DATE -** Date of attendance
  + **check\_in\_time: TIME -**  Check-in time of the member for the session.
  + **check\_out\_time: TIME -** Check-out time of the member for the session.
  + **Foreign Key:** member\_id references Members(member\_id)

**Payments:** Records membership payments for members.

* **Columns:**
  + **payment\_id: INT (PK, AUTO\_INCREMENT) -** Unique ID for each payment record
  + **member\_id: INT (FK) -** Reference to the member\_id in Members table
  + **payment\_amount: DECIMAL(10, 2**) - Amount paid by the member
  + **payment\_date:** TIMESTAMP (default: CURRENT\_TIMESTAMP) - Date and time of payment
  + **payment\_method: VARCHAR(50) -** Method of payment (e.g., Credit Card, Cash)
  + **payment\_status: VARCHAR(50)-** Paid or Pending
  + **Foreign Key:** member\_id references Members(member\_id)

**MembershipType:** Defines types of memberships.

 **Columns**:

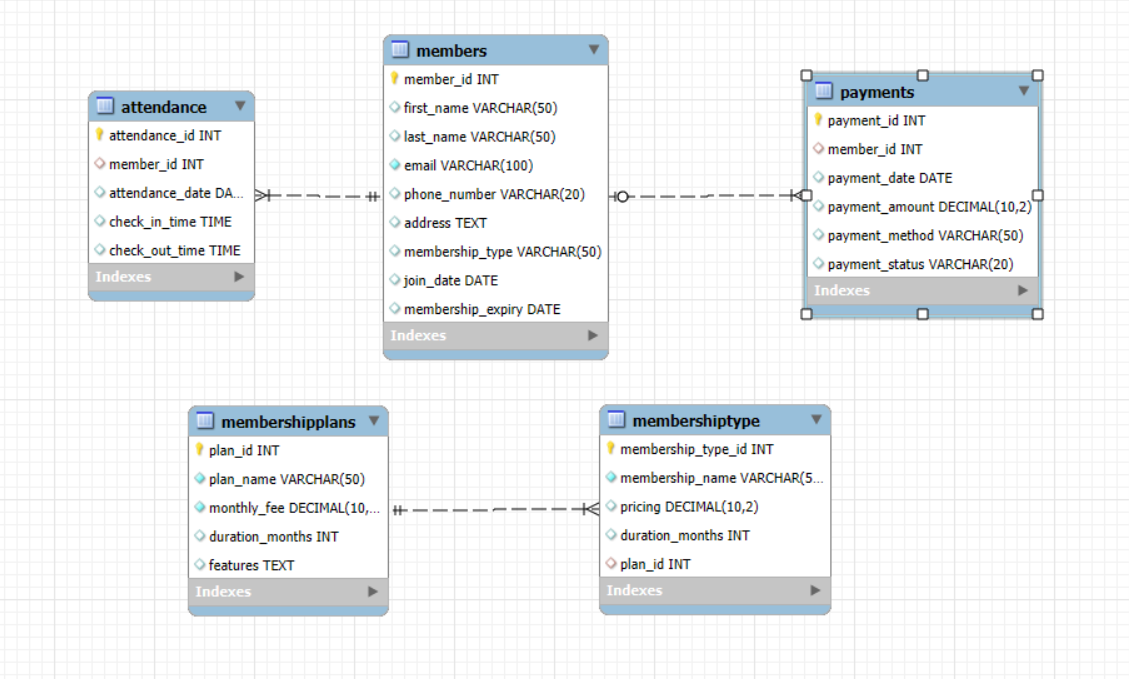
* **membership\_type\_id(INT (Primary Key):** Unique identifier for each membership type
* **membership\_name(varchar(50):** Name of the membership type (e.g.,

"Basic", "Premium").

* **Pricing(Decimal(10,2) :** Pricing for the membership type.
* **duration\_months(int) :** Duration in months for the membership.
* **plan\_id( int foreign key) :** References plan\_id in the

MembershipPlans table.

**SCHEMA DIAGRAM:**



**ER Diagram:**

