DMSL MINI PROJECT REPORT

ON

ONLINE GYM MANAGEMENT SYSTEM

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE FOR THE
PARTIAL FULFILLMENT FOR THE AWARD OF THE DEGREE

OF

BACHELOR OF ENGINEERING IN
INFORMATION TECHNOLOGY

BY

Ishan Deshmukh	23239
Digvijay Jadhav	23240
Shradha Jadhav	23241
Omkar Jagtap	23242
Vikrant Joshi	23243

Under the guidance of

Mrs. K.Y. Digholkar **Signature**

DEPARTMENT OF INFORMATION TECHNOLOGY PUNE
INSTITUTE OF COMPUTER TECHNOLOGY, PUNE
Sr. No. 27, Trimurti Chowk, Dhankawadi, Pune-43

AY 2024-25

Society for Computer Technology & Research's

PUNE INSTITUTE OF COMPUTER TECHNOLOGY

Sr. No. 27, Trimurti Chowk, Dhanakawadi, Pune-43

DEPARTMENT OF INFORMATION TECHNOLOGY

CERTIFICATE

This is to certify that the DMSL MINI PROJECT report entitled

ONLINE GYM MANAGEMENT SYSTEM

submitted by

Ishan Deshmukh	23239
Digvijay Jadhav	23240
Shradha Jadhav	23241
Omkar Jagtap	23242
Vikrant Joshi	23243

is a bonafide work carried out by them in the second year of Engineering (AY 2024- 25) under the supervision of **Mrs. K.Y. Digholkar** and it is approved for the partial fulfillment of the requirement of Savitribai Phule Pune University for the award of the Degree of Bachelor of Engineering (Information Technology).

Mrs. K.Y. Digholkar Lab Instructor and Mentor	Dr. A. S. Ghotkar Head of the Department	Dr. S. T. Gandhe Principal

Date:
Place: PUNE

.

ABSTRACT

The Online Gym Management System is a database management system developed to simulate and streamline a fitness environment where gym members can be registered, assigned trainers, and enrolled in customized workout or membership plans. The primary objective of this system is to manage data related to gym members, trainers, schedules, and progress tracking using a structured and relational database approach.

This system provides functionalities for storing and retrieving data about gym members, such as their age, fitness goals, health records, enrolled plans, and attendance. Trainers can register and manage their assigned clients, create workout routines, and track performance based on pre-defined metrics and progress logs. The project involves designing and implementing a robust backend using MySQL for efficient data storage, manipulation, and retrieval.

To ensure seamless interaction between users and the database, an intuitive user interface has been designed. The system architecture includes Entity-Relationship modeling, schema design, table normalization, and structured query handling.

Additionally, features such as workout history tracking, trainer feedback, and performance analytics have been integrated to enhance the user experience. The project showcases the real-world application of DBMS concepts in health and fitness management and emphasizes the importance of data consistency, integrity, and scalability.

Overall, the Online Gym Management System combines the discipline of fitness management with the practicality of database systems, offering a creative yet technically sound solution for managing a virtual gym environment.

ACKNOWLEDGEMENT

We take this opportunity to express our heartfelt gratitude to everyone who supported us during the successful completion of our DBMS mini project titled “Online Gym Management System.”

First and foremost, we would like to thank our respected guide, Mrs. K.Y. Digholkar, for their constant guidance, valuable suggestions, and constructive feedback that helped shape this project effectively.

We are thankful to the Department of Information Technology, Pune Institute of Computer Technology (PICT) for providing us with the necessary resources, technical environment, and support for our academic growth.

Our sincere thanks to the DMSL Lab coordinators and staff for their assistance and cooperation throughout the lab sessions.

We also extend our appreciation to our teammates for their active participation, dedication, and collaborative spirit which contributed significantly to the development of this project.

We gratefully acknowledge the use of various open-source resources, including Pokémon data references, which inspired and enriched our project design and implementation.

We would also like to thank our friends and classmates who offered encouragement, tested the system, and shared valuable feedback during development.

Special thanks to our families for their continuous motivation and support during the project duration.

Through this project, we not only deepened our understanding of database systems but also enhanced our teamwork, problem-solving, and software development skills.

This experience will serve as a foundation for our future academic and professional endeavors.

TABLE OF CONTENTS

LISTS OF FIGURES

INTRODUCTION

1.1 Purpose

The purpose of the Online Gym Management System project is to design and implement a database management system that facilitates the management of gym-related activities between users. This system allows users to register as members, choose trainers from a predefined database, and engage in fitness plans based on attributes such as age, fitness goals, workout type, and progress. The primary goal is to demonstrate how DBMS concepts can be practically applied in a fitness-based environment, offering users an interactive platform while showcasing real-world applications of relational databases.

1.2 Scope

The scope of the Online Gym Management System includes:

- Maintaining a database of gym members, each with attributes like ID, Name, Age, Gender, Fitness Goals, Plan Type, and Attendance.
- Managing trainer accounts and their respective assigned members.
- Tracking workout sessions based on predefined routines and updating the database with progress records.
- Recording workout history, performance metrics, and trainer feedback.
- Providing a user-friendly interface for interaction with the database.
- Ensuring data integrity, normalization, and efficient querying using SQL.

This project also emphasizes hands-on learning of ER modeling, schema design, normalization, SQL operations, and implementation of user-driven applications with backend support.

1.3 Definition, Acronym, and Abbreviations

Term/Acronym	Description
DBMS	Database Management System
SQL	Structured Query Language
ER Model	Entity-Relationship Model
UI	User Interface
PICT	Pune Institute of Computer Technology

1.4 References

1. Gym management software features and system overview: https://www.softwareadvice.com/fitness/
2. MySQL Documentation: https://dev.mysql.com/doc/
3. Fundamentals of Database Systems – Elmasri & Navathe
4. W3Schools SQL Tutorial: https://www.w3schools.com/sql/
5. DBMS Lab Manual, PICT

1.5 Developers' Responsibilities: An Overview

The developers of **Online Gym Management System** are responsible for:

- Designing the overall database schema and ER model.
- Implementing tables and relationships with appropriate constraints and normalization.
- Developing a user interface for interaction with the system.
- Writing SQL queries for inserting, updating, retrieving, and deleting data.
- Implementing the logic for workout tracking and trainer assignments based on user data and storing progress records.
- Performing system testing, error handling, and documenting test cases.
- Ensuring smooth functioning of the application while adhering to DBMS principles.
- Maintaining proper documentation for future upgrades or modifications.

GENERAL DESCRIPTION

2.1 Product Function Perspective

The Online Gym Management System functions as an interactive fitness management platform backed by a relational database. The key product functionalities are:

1. **User Registration and Login**
    - Members and trainers can register and log into the system.

2. **Member Database Management**
    - A predefined list of members is stored with attributes: ID, Name, Age, Gender, Fitness Goal, and Membership Plan.

3. **Trainer Assignment**
    - Trainers can be assigned to members based on availability and specialization.

4. **Workout Tracing**
    - Daily workout logs and fitness sessions are recorded and tracked for each member.

5. **Progress Storage**
    - Member progress, feedback, and performance records are stored with timestamps.

6. **User-Friendly Interface**
    - Easy navigation through menus and forms for members and trainers.

7. **Data Retrieval and Querying**
    - Members and admins can retrieve trainer info, workout stats, or attendance details.

8. **Security and Validation**
    - Input checks to prevent SQL injection or invalid data entries.

This system reflects how fitness logic can be structured within a normalized, consistent DBMS framework.

2.2 User Characteristics

The PokeWars system is designed for the following types of users:

1. **Members (End Users)**
    - Basic understanding of fitness goals and workout routines.
    - Familiar with selecting workout plans and tracking progress.
    - May interact via GUI or pre-defined forms and dashboards.
2. **Administrators (Developers/Faculty)**
    - Responsible for adding/removing member or trainer records.
    - Manages database integrity and schema.
    - Monitors the leaderboard and data correctness.

User Traits:

- No prior advanced programming or DBMS experience is required.
- Users should be able to use basic form inputs.
- Designed for DBMS learners and fitness management scenarios.
- The UI focuses on simplicity, readability, and intuitive navigation.
- Assumes users can follow instructions given in popups/help menus.

2.3 General Constraints

Some technical and functional constraints are as follows:

1. **Database Size Limitation**
    - Member and trainer data are limited due to memory and academic scope.
2. **Static Data**
    - Fitness plans and trainer assignments are pre-defined and manually entered.
3. **No Real-Time Tracking**
    - Workout sessions are not tracked live; logs are entered manually post-session.
4. **Simplified Logic**
    - Workout tracking is basic (form-based entry), not integrated with wearables or sensors.
5. **Single-Device Hosting**
    - Hosted locally using MySQL with XAMPP or similar environment.
6. **Interface Limitation**
    - Runs in a basic terminal or minimal GUI, not mobile or web-optimized.
7. **No Multiplayer Over Network**
    - All user actions are processed on a single device or locally.
8. **No Dynamic Data Fetching**
    - No API integrations; no live fitness or health data from external platforms.
9. **Security Scope**
    - Basic input validation only; no password hashing or encryption implemented.
10. **Limited Error Recovery**
    - System does not support rollback of all types of failed actions or incorrect inputs .

2.4 Assumptions and Dependencies

The development of the Online Gym Management System is based on several key assumptions and dependencies that define how the system operates. It is assumed that users have basic familiarity with fitness goals, workout routines, and how gym management systems function. The system also assumes that data entries for gym members, trainers, and workout plans are accurate, consistent, and maintained by an administrator to avoid invalid records.

The project is dependent on a working relational database system, preferably MySQL, and access to a local or hosted environment like XAMPP or phpMyAdmin for database management. It is also assumed that users accessing the system have stable hardware and basic knowledge of how to operate the user interface or input data through forms.

The system relies on predefined workout plans and static member data. Any errors in these definitions could lead to incorrect progress tracking or trainer assignments. Moreover, the database must remain consistent and normalized to prevent anomalies during member registration or trainer assignments.

Another assumption is that the number of gym members and trainers will remain within a manageable limit due to the mini-project scale. The user interface is expected to work on standard browsers and does not support mobile or dynamic web platforms.

Lastly, future extensions such as diet tracking, mobile app integration, and real-time data fetching from wearables are considered out of scope for the current version but are assumed to be possible with additional time and resources.

SPECIFIC REQUIREMENTS

3.1 Inputs and Outputs

This section outlines the types of data inputted into the system and the corresponding outputs expected after processing:

Inputs:

1. **Member registration data:** Name, email, and unique ID.
2. **Login credentials:** Trainer ID and password.
3. **Workout plan selection:** IDs or names of selected fitness plans for enrollment.
4. **Trainer assignment:** Selection of a personal trainer based on availability.
5. **Admin inputs:** Trainer profiles, member records, fitness routines and schedule rules.

Outputs:

1. **Confirmation messages** for registration, login, and team creation.
2. **Display of member's assigned workout plan** with schedule and trainer info.
3. **Progress tracking reports** showing completed sessions and status updates.
4. **Session history logs:** date, member involved, trainer assigned, and notes.
5. **Leaderboard:** ranked list of members by workout completion rate or consistency.
6. **Error messages** for invalid inputs or failed operations.

3.2 Functional Requirements

These define the core functionalities that must be supported by the Online Gym Management system:

1. The system must allow members and trainers to register and log in securely.
2. Members should be able to view all available workout plans from the database.
3. The system should enable members to enroll in fitness programs by selecting plans.
4. A progress module should track performance based on metrics like attendance, goals, and trainer feedback.
5. Members should be able to request and get assigned personal trainers from the registered list.
6. The system must store workout session logs along with the timestamp and involved trainers.
7. It should update member progress records in the database after each session.
8. A leaderboard should be generated and updated based on member participation and consistency.
9. Admins must be able to add or update workout plans, trainer profiles, and member data.
10. Queries should be available for retrieving member details, workout plans, and session history.

3.3 Functional Interface Requirements

This section defines the interaction between the user and system components:

- Trainer Interface:
  - Login and registration forms.
  - Workout plan list view with search functionality.
  - Plan selection and enrollment confirmation window.
  - Session tracking and progress view panel.
- Admin Interface:
  - Secure login for admin privileges.
  - Workout plan data management: Add, Update, Delete.
  - Member and trainer data oversight.
  - View leaderboard and session logs.
- Database Interaction:
  - Execution of SQL queries (SELECT, INSERT, UPDATE, DELETE).
  - Proper use of JOINs for linking multiple tables (e.g., Members, Trainers, Sessions).
  - Consistent use of foreign keys for relationship integrity.

3.3 Performance Constraints

The following are limitations in terms of the system's performance:

1. The system must respond to any trainer action within 2-3 seconds.
2. Database queries must not cause system lag or crash.
3. Maximum users supported at a time is around 50 (due to testing constraints).
4. Maximum workout plans in the database should be under 200 to prevent performance degradation.
5. Session logs must be recorded and displayed in under 5 seconds.
6. Queries for leaderboard and history must be optimized using indexes or constraints.
7. No support for concurrent battles due to a single-session design.
8. Basic validations (like empty fields or invalid Pokémon IDs) must be handled in real-time.
9. Data redundancy should be minimized to enhance speed and reduce query time.
10. Use of optimized database schema to avoid unnecessary joins or subqueries.

3.4 Design Constraints

Design constraints are rules or limitations placed on the development process:

1. The system is designed using MySQL as the primary DBMS.
2. User interaction is assumed via a terminal-based or simple GUI (e.g., using frontend).
3. Tables must follow 3rd Normal Form (3NF) for data consistency.
4. No real-time multiplayer or networked components are included.
5. System must be usable in offline environments.
6. Only manual data entry for adding new members or trainers.
7. Interface design is constrained to basic form input and console outputs.
8. No drag-and-drop or graphic animations.
9. Each member can select only a fixed number of plans or sessions (e.g., 3 or 5).
10. All database operations must be ACID compliant.

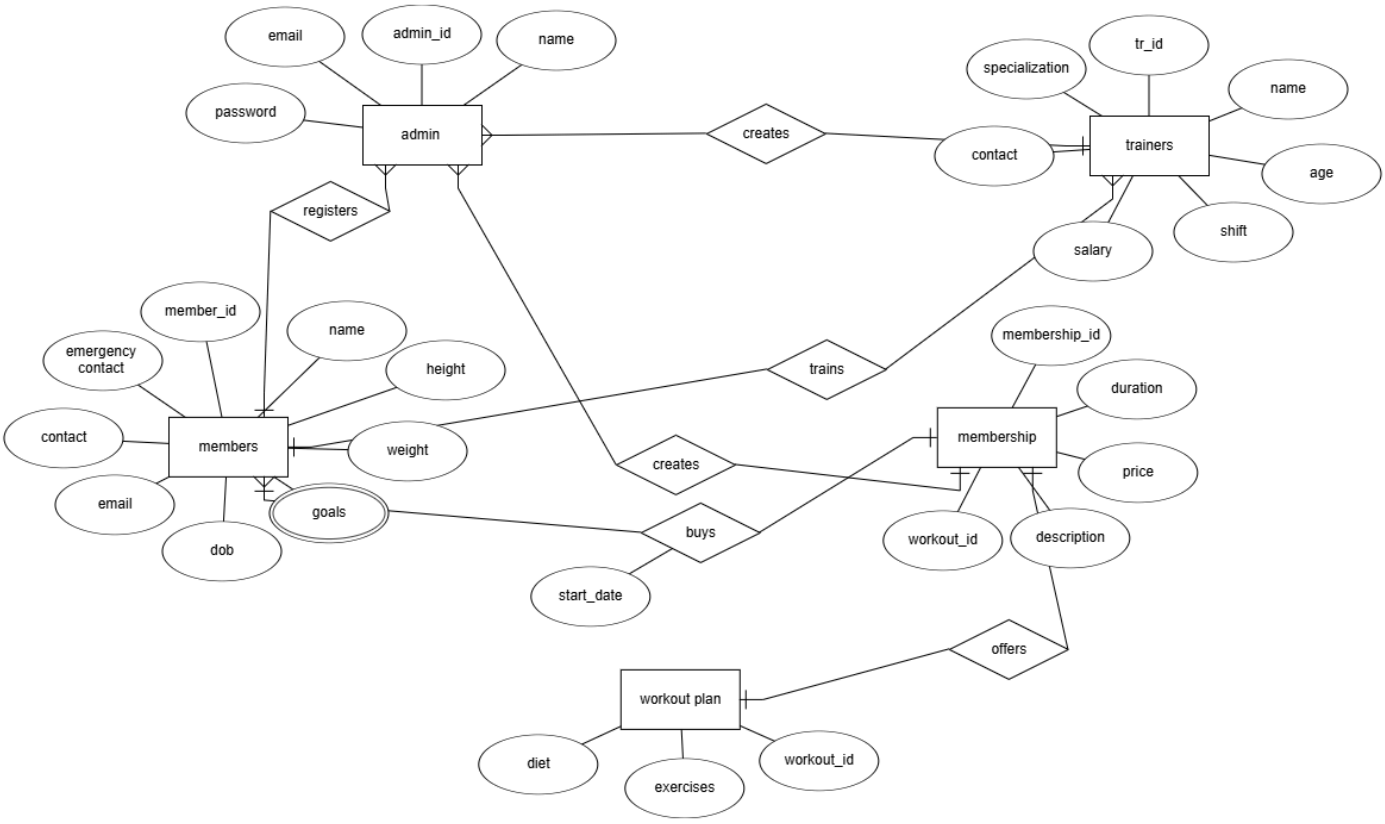
3.5 Acceptance Criteria

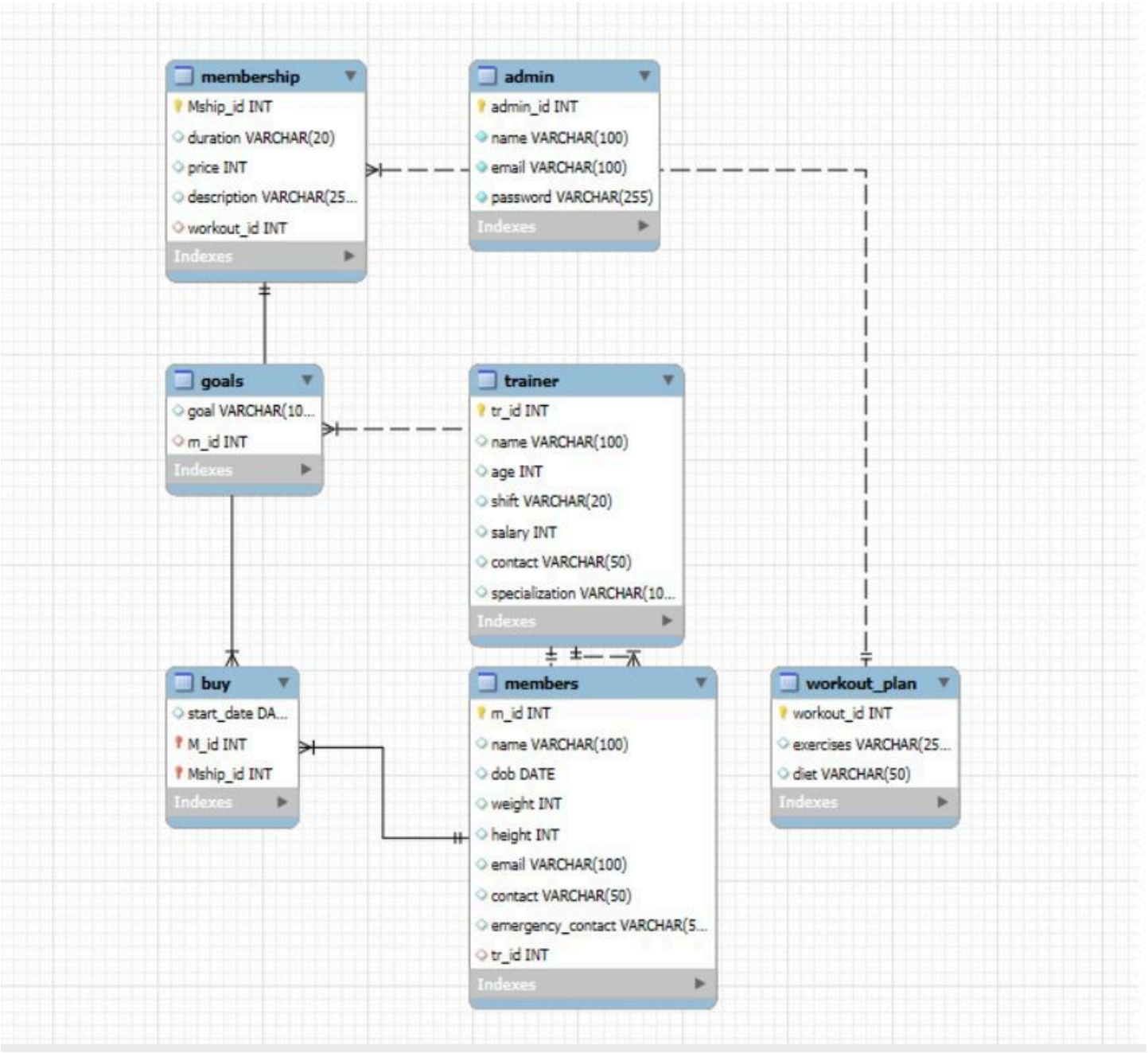
These are the criteria that must be met for the system to be accepted as complete:

1. Members just be able to register, log in, and manage their teams without any error.
2. Plan logic should function correctly and assign sessions based on defined rules.
3. All data (member info, trainer details, session records) must be stored and retrievable.
4. Membership status must update automatically after plan changes.
5. No invalid operations (e.g., registering with same email twice) should be allowed.
6. Queries should produce correct and meaningful results.
7. The system must execute without crashing for a set of defined test cases.
8. System must run on local machines using predefined platforms.
9. The design must match the ER diagram and adhere to constraints.
10. Must pass manual and automated test cases for input validation, plan accuracy, and database integrity.

SYSTEM DESIGN

4.1 ER Diagram




membership
Mship_id INT
duration VARCHAR(20)
price INT
description VARCHAR(25...
workout_id INT
Indexes
admin
admin_id INT
name VARCHAR(100)
email VARCHAR(100)
password VARCHAR(255)
Indexes
goals
goal VARCHAR(10...
m_id INT
Indexes
trainer
tr_id INT
name VARCHAR(100)
age INT
shift VARCHAR(20)
salary INT
contact VARCHAR(50)
specialization VARCHAR(10...
Indexes
buy
start_date DA...
M_id INT
Mship_id INT
Indexes
members
m_id INT
name VARCHAR(100)
dob DATE
weight INT
height INT
email VARCHAR(100)
contact VARCHAR(50)
emergency_contact VARCHAR(5...
tr_id INT
Indexes
workout_plan
workout_id INT
exercises VARCHAR(25...
diet VARCHAR(50)
Indexes

Entities:

1. **Members**
2. **Trainers**
3. **Membership**
4. **Workout plan**

4.2 Schema Description

The database schema consists of five tables:

1. Members (stores details of members)

2.Trainers(stores Trainers details)

3. Membership(stores Membership plans)

4. Workout plan(Stores workout plans made by the trainers)

5. Buy(Stores the date of membership purchased by members

4.3 Table Description

1.Members

Column Name	Data Type	Description
M_id	INT(PK)	Unique id for members
Name	VARCHAR(15)	Name of the member
DOB	DATE	Date of birth of member
Weight	INT	Weight of the member
Height	INT	Height of the member
Age	INT	Age of the member
Goals	VARCHAR(30)	Goals of the member

2.Trainers

Column Name	Data Type	Description
Tr_id	INT(PK)	Unique id for trainers
Name	VARCHAR(15)	Name of trainers
Shift	VARCHAR(15)	Shift of trainers
Age	INT	Age of trainers
Salary	INT	Salary of trainers
M_id	INT(FK)	Member trained by trainer(foreign key from members)

3.Membership

Column Name	Data Type	Description
Mship_id	INT(PK)	Unique id for membership plans
Duration	VARCHAR(20)	Duration of membership in months
Price	INT	Price of the membership plan
Description	VARCHAR(50)	Description of the membership plan

4.Workout Plan

Column Name	Data Type	Description
Workout_id	INT(PK)	Unique id for workout plan
Exercises	VARCHAR(30)	Exercises in the workout plan
Diet	VARCHAR(30)	Diet for the workout plan
Duration	INT	Duration of the workout plan in days
Tr_id	INT(FK)	Id of trainer who created this workout plan
Mship_id	INT(FK)	Id of the membership plan which has this workout plan

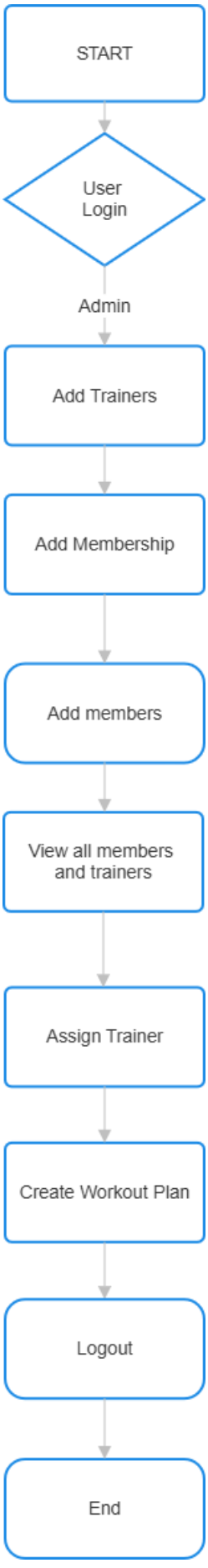
5.Buy

Column Name	Data Type	Description
Start_date	DATE	Start date of the membership plan purchased by member
End_date	DATE	End date of the membership plan purchased by member
Mship_id	INT(FK)	Id of the membership plan purchased by member
M_id	INT(FK)	Member who buyed the membership plan mentioned in mship_id

4.4 Relationships

From Table	To Table	Type Relationship	Description
MEMBERS.mem_id	MEMBERSHIP.Mship_id	Many-to-One	A member can buy one membership, but many members can have the same plan.
MEMBERS.mem_id	TRAINERS.Tr_id	Many-to-One	A member is trained by one trainer, but a trainer can train many members.
TRAINERS.Tr_id	WORKOUT_PLAN.Workout_id	One-to-Many	A trainer can create multiple workout plans.
MEMBERSHIP.Mship_id	WORKOUT_PLAN.Workout_id	One-to-Many	A membership can include multiple workout plans.

4.5 System Flow Diagram/ Activity Diagram



4.3 User Interface Design

1. Login Page

- Input fields for:
    - **Username**
    - **Password**
- Login button
- Toggle switch to select login type:
    - **Admin**

2. Member Dashboard

- Welcome header with Member’s name
- Sidebar navigation menu:
    - Home
    - My Membership
    - Workout Plan
    - Assigned Trainer
    - Profile
- Membership section showing:
    - Type, Start Date, End Date, Status
    - Buy/Renew Membership button
- Workout Plan view with:
    - Duration, Diet, Exercises
- Assigned Trainer details:
    - Name, Shift, Contact Info
- Logout button

3. Trainer Dashboard

- Trainer profile header with Trainer’s name
- Sidebar menu:
    - Dashboard

    - My Members
    - Create Workout Plan
    - Schedule
- Members List with:
    - Member Name, Membership Type, Progress Overview
- Workout Plan Creator:
    - Select Member
    - Input Duration, Exercises, Diet
    - Save Plan button
- Logout button

4. Admin Dashboard

- Admin profile header with Admin name
- Sidebar menu:
    - Dashboard
    - Manage Members
    - Manage Trainers
    - Membership Plans
    - Workout Plans
- Panels:
    - Member & Trainer Records (Add/Edit/Delete)
    - Membership Management (Add/Edit Pricing, Duration)
    - Workout Plan Logs
- Logout button

5. Membership Page

- Dropdown to select Membership Plan
- Displays:
    - Description, Duration, Price, Offers
- Buy Membership button
- View past membership history

6. Workout Plan Page

- Workout summary view:
    - Exercises, Diet, Duration
- Option to view different plans (if applicable)
- Progress Tracker (optional enhancement)

4.3 Error Messages / Alerts Design

To ensure a smooth and intuitive user experience, the system provides clear and informative alerts in response to incorrect actions or invalid data inputs.

Login Errors

- "Invalid Username or Password!" – Displayed when login credentials are incorrect.
- "User not found!" – Shown if the entered username is not registered in the system.
- "Please select a login type (Member/Trainer/Admin)." – Triggered when login type is left unselected.

Membership Errors

- "No active membership found!" – If a member attempts to access features without a valid membership.
- "Membership already active!" – When a member tries to repurchase an already active membership.
- "Membership plan not available!" – When a selected plan no longer exists or is disabled.

Workout Plan Errors

- "No workout plan assigned yet." – Displayed if the trainer hasn't assigned a plan to the member.
- "Workout plan already exists for this member." – If a trainer tries to create multiple plans for the same member without removing the previous one.

Form Validation Alerts

- "Please fill in all required fields." – Triggered when mandatory form inputs are left empty.
- "Invalid input! Please enter valid data." – Displayed when user input does not match expected format (e.g., age, weight).
- "Date format incorrect. Please use DD-MM-YYYY." – For incorrectly formatted date entries.

General Alerts

- "Action successful!" – For successful operations like login, form submission, updates, etc.
- "Something went wrong. Please try again later." – Generic message for system or server errors.

4.4 Test Case Design

Test cases ensure the system functions as expected.

TC ID	Module	Test Case Description	Input	Expected Output
TC003	Login	Invalid login	Username: member001, Password: wrongpass	Error alert: "Incorrect username or password."
TC004	Login	Empty fields	Username: "", Password: ""	Alert: "Username and password are required."
TC005	Member Management	View gym members (Admin)	-	Member list displayed with details
TC006	Trainer Management	Add new trainer	Name, Phone, Email, Specialization	Trainer successfully added
TC007	Trainer Management	Assign trainer to member	Member ID: 102, Trainer ID: 5	Assignment successful message
TC008	Membership Management	Create new membership	Member ID: 102, Plan: 3 months	Membership created successfully
TC009	Membership Management	Create duplicate membership	Member ID: 102 (already has active membership)	Alert: "Membership already active."
TC010	Member View	Member views assigned trainer	Member ID: 102	Assigned trainer details shown
TC011	Trainer View	Trainer views assigned members	Trainer ID: 5	List of assigned members displayed
TC012	Dashboard Navigation	Admin navigates between dashboard sections	Click on "Manage Members", "Add Trainer", etc.	Corresponding section content displayed
TC014	Logout	Admin logout	Click Logout button	Redirected to login page

System Implementation

5.1 Hardware and Software Platform Description

Hardware Requirements:

1. Processor: Intel Core i5 or higher (or AMD equivalent)
2. RAM: Minimum 8GB (Recommended: 16GB for optimal performance)
3. Storage: At least 500GB HDD (Recommended: SSD for better performance)
4. Graphics: Integrated GPU (Dedicated GPU recommended for better visuals)
5. Display: Minimum resolution of 1366×768 (Recommended: Full HD 1920×1080)

Software Requirements:

1. Operating System: Windows 10/11, Linux (Ubuntu 20.04+), or macOS
2. Database: MySQL (Version 8.0 recommended)
3. Frontend: React.js
4. Version Control: Git and GitHub
5. API Development: RESTful API for managing members, trainers, and sessions
6. Testing Framework: Postman for backend testing

5.2 Tools Used

Development Tools:

1. Node.js and Express.js: Framework for backend logic and API creation
2. MySQL: Database management for storing gym-related data
3. GitHub / Git: Version control for source code tracking
4. Postman: API testing and debugging

Design and Documentation Tools:

1. ERDplus: For ER diagrams and system design and schema design.

5.3 System Verification and Testing (Test Case Execution)

The application was tested manually using the designed test cases:

• Each module (Login, Book Issue, Return, Fine) was tested individually.

• Edge cases (e.g., invalid inputs, no book availability) were verified.

• All test cases passed during final testing, and the system behaved as expected under typical load conditions.

5.4 Future Work / Extension

The *Online Gym Management* system can be further enhanced with:

1. **Real-Time Trainer Sessions** – Enable live interaction between trainers and members using WebSockets.
2. **AI Fitness Advisor** – Integrate an AI-based assistant to suggest workouts and diets based on user goals.
3. **Mobile App Development** – Develop an Android & iOS app using Flutter for on-the-go gym access.
4. **Wearable Integration** – Sync data from fitness trackers (e.g., Fitbit, Apple Watch) for activity monitoring.

5.5 Conclusion

The Online Gym Management System successfully implements a fitness management platform with features like member registration, workout tracing, trainer assignments and performance monitoring. The system is built using React.js., Node.js, Express.js and MySQL ensuring scalability and efficiency. Testing through manual validation and user feedback guarantees system robustness. Future enhancements like diet tracking, mobile app integration and real time trainer can further enrich the experience.

References

1.Gym management software features and system overview: https://www.softwareadvice.com/fitness/

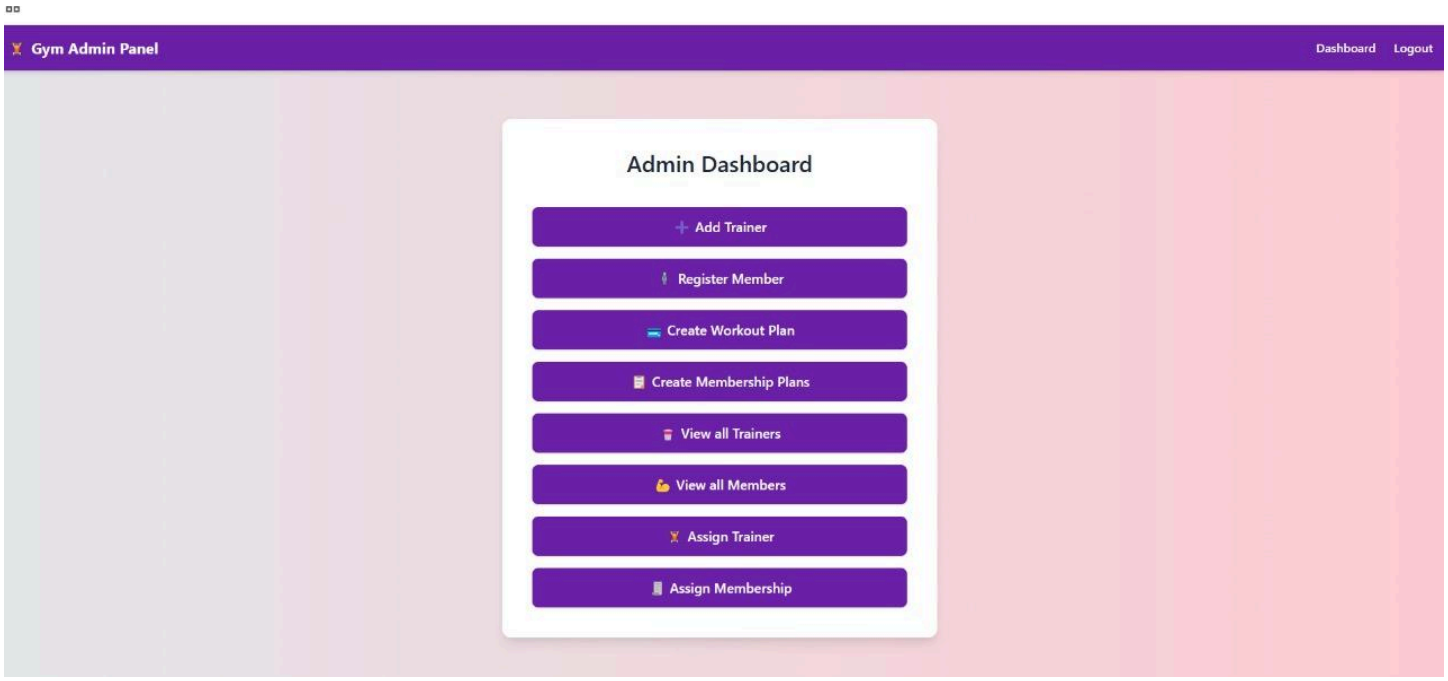
2. MySQL Documentation: https://dev.mysql.com/doc/

3. Fundamentals of Database Systems – Elmasri & Navathe

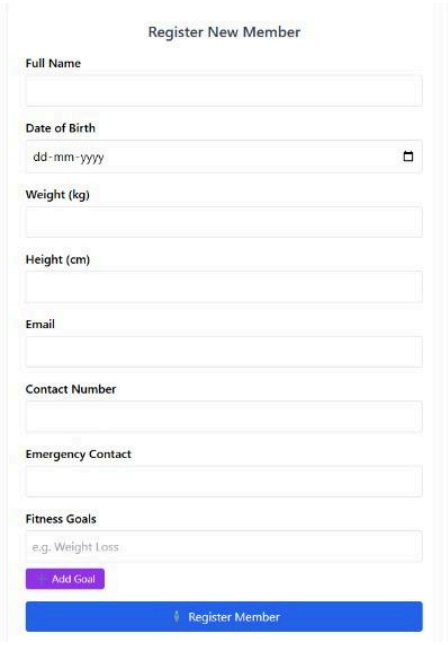
4. W3Schools SQL Tutorial: https://www.w3schools.com/sql/

5.DBMS Lab Manual, PICT

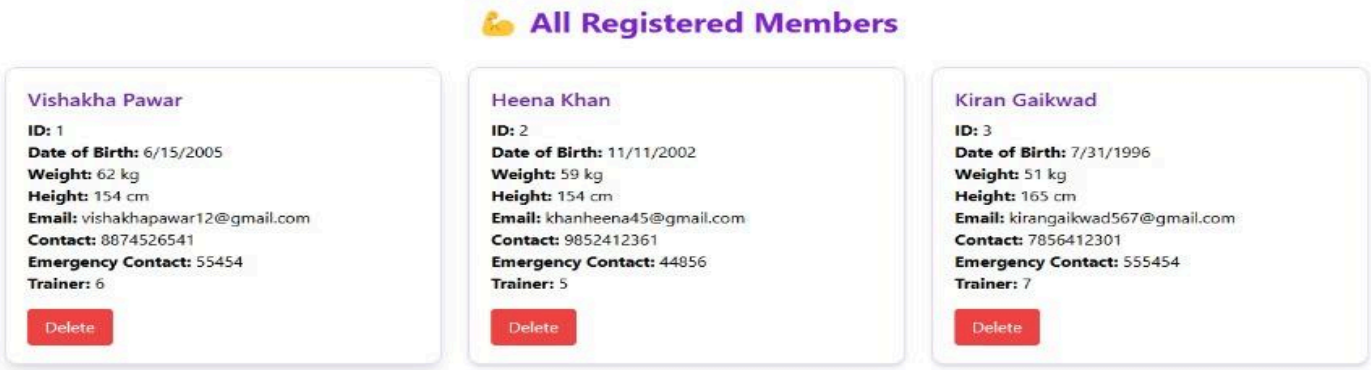
LIST OF FIGURES



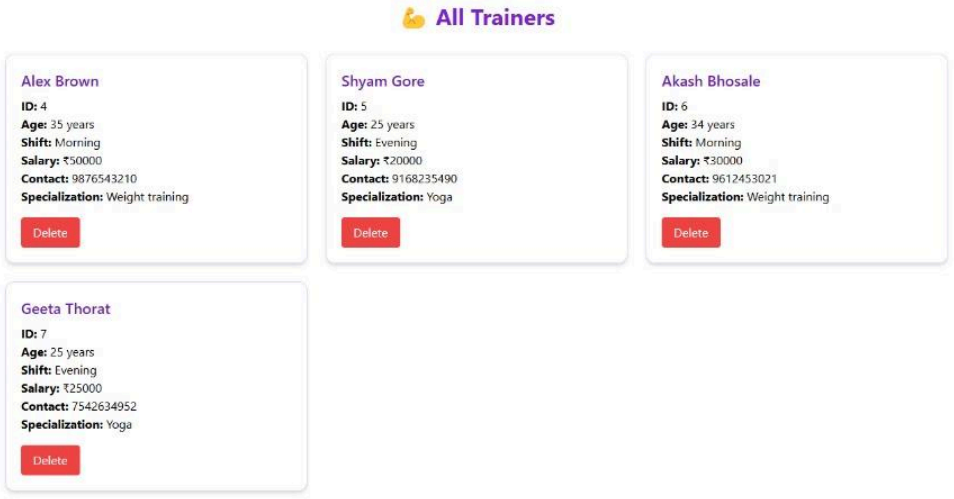
Dashboard Page



Register Member Page



View Members Page



View Trainers Page