FITNESS CLUB SYSTEM

A PROJECT REPORT
For
Project (KCA451)
Session (2023-24)

Submitted by

VIKASH GUPTA
(2200290140177)
SMRITI TIWARI
(2200290140153)
YASH SHRIVASTAVA
(2200290140187)

Submitted in partial fulfillment of the
Requirements for the Degree of

MASTER OF COMPUTER APPLICATION

Under the Supervision of
DR. SANGEETA ARORA
ASSOCIATE PROFFESOR

Submitted to

DEPARTMENT OF COMPUTER APPLICATIONS
KIET Group of Institutions, Ghaziabad
Uttar Pradesh-201206

(JUNE 2024)

CERTIFICATE

Certified that **Vikash Gupta 2200290140177, Smriti Tiwari 2200290140153, Yash Shrivastava 2200290140187** has/ have carried out the project work having "**Fitness Club System**" (**Project-KCA451**) for **Master of Computer Application** from Dr. A.P.J. Abdul Kalam Technical University (AKTU) (formerly UPTU), Lucknow under my supervision. The project report embodies original work, and studies are carried out by the student himself/herself and the contents of the project report do not form the basis for the award of any other degree to the candidate or to anybody else from this or any other University/Institution.

Date:

Vikash Gupta (2200290140177)
Smriti Tiwari (2200290140153)
Yash Shrivastava(2200290140187)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date:

Dr. Sangeeta Arora	**Dr. Arun Tripathi**
Associate Professor	**Head**
Department of Computer Applications	**Department of Computer Applications**
KIET Group of Institutions, Ghaziabad	**KIET Group of Institutions, Ghaziabad**

Fitness Club System
Vikash Gupta
Smriti Tiwari
Yash Shrivastava

ABSTRACT

Our project, the Easy Fitness Club Manager, is a user-friendly software designed to help fitness clubs handle their day-to-day tasks effortlessly. Using Java Swing for a straightforward interface and MySQL for storing data, this system offers features like login, adding new members, updating their info, and keeping track of payments.

The login module ensures secure access to the system, allowing authorized personnel to manage club operations efficiently. Upon authentication, administrators gain access to a dashboard where they can perform various tasks seamlessly.

The member registration module facilitates the addition of new members to the system. Users can input essential details such as name, contact information, and membership type, which are then stored in the MySQL database for easy retrieval and management.

The system also enables administrators to update and delete member details as necessary. Whether it's modifying contact information or updating membership status.

Administrators can track member payments, view outstanding dues, and generate payment reports effortlessly. By integrating with MySQL, all payment transactions are stored securely, ensuring data integrity and reliability.

In summary, Fitness Club System is here to make life easier for fitness clubs. With its user-friendly design and seamless integration with MySQL, managing memberships and payments has never been simpler.

ACKNOWLEDGEMENTS

Success in life is never attained single-handedly. My deepest gratitude goes to my project supervisor, **DR. SANGEETA ARORA** for her guidance, help, and encouragement throughout my project work. Their enlightening ideas, comments, and suggestions.

Words are not enough to express my gratitude to Dr. Arun Kumar Tripathi, Professor and Head, Department of Computer Applications, for his insightful comments and administrative help on various occasions.

Fortunately, I have many understanding friends, who have helped me a lot on many critical conditions.

Finally, my sincere thanks go to my family members and all those who have directly and indirectly provided me with moral support and other kind of help. Without their support, completion of this work would not have been possible in time. They keep my life filled with enjoyment and happiness.

Vikash Gupta
Smriti Tiwari
Yash Shrivastava

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLE

CHAPTER 1

INTRODUCTION

1.1 OVERVIEW

The Easy Fitness Club Manager is an innovative software solution designed to streamline the administrative operations of fitness clubs. The project leverages Java Swing for a user-friendly graphical interface and MySQL for robust data storage, providing a comprehensive system for managing memberships, payments, and member information. This overview outlines the project's key components, objectives, and anticipated outcomes.

Key Features and Functionalities

1. **Secure Login Module:**
   - Ensures only authorized personnel can access the system.
   - Provides a secure authentication mechanism to protect sensitive data.
2. **Member Registration:**
   - Allows administrators to add new members with essential details such as name, contact information, and membership type.
   - Stores member information in a MySQL database for easy retrieval and management.
3. **Member Management:**
   - Enables updating and deleting member details as necessary.
   - Facilitates modifications of contact information and membership status efficiently.
4. **Payment Tracking:**
   - Allows administrators to track member payments and view outstanding dues.
   - Generates detailed payment reports, ensuring financial transparency.
   - Securely stores all transaction data in the MySQL database, maintaining data integrity and reliability.

1.2 OBJECTIVE

The primary goals of the Easy Fitness Club Manager are to enhance operational efficiency and improve member experiences through:

- **Enhancing Accessibility and Convenience:**

  Provides cross-platform access, allowing administrators to manage operations remotely.

- **Promoting Financial Transparency:**

  Accurately tracks transactions and generates financial reports.

- **Encouraging Member Engagement:**

  Utilizes notifications and personalized communication to foster member participation.

- **Supporting Decision-Making and Planning**:

  Offers real-time access to key performance indicators and analytics for informed decision-making.

- **Continuous Improvement and Adaptation:**

  Incorporates feedback mechanisms to prioritize feature enhancements and adapt to industry trends.

- **Seamless Integration:**

  Ensures compatibility with existing systems to minimize disruptions during implementation.

- **Providing Training and Support:**

  Offers comprehensive training and ongoing technical support to maximize system utilization.

1.3 PROJECT FEATURE

The Easy Fitness Club Manager is designed to provide a comprehensive set of features that streamline the administrative operations of fitness clubs. Below is an overview of the key features offered by the system :

- **Secure Login Module**
  - Authentication: Ensures only authorized personnel can access the system through a secure login process.
  - Role-Based Access Control: Different levels of access are granted based on user

roles (e.g., administrators, staff).

- **Member Registration**
  - New Member Enrollment: Allows for easy addition of new members, capturing essential details such as name, contact information, membership type, and start date.
  - Data Storage: All member information is securely stored in the MySQL database for easy retrieval and management.

- **Member Management**
  - Update Member Details: Enables administrators to update member information, including contact details and membership status.
  - Delete Member Records: Provides functionality to remove member records from the system as needed.
  - Membership Renewal: Facilitates the renewal process for existing memberships.

- **Payment Tracking and Management**
  - Payment Recording: Tracks all member payments, including membership fees, classes, and other services.
  - Outstanding Dues: Allows administrators to view and manage outstanding payments.
  - Financial Reporting: Generates detailed payment reports to ensure financial transparency and accuracy.
  - Secure Transactions: All payment data is securely stored and managed within the MySQL database, ensuring data integrity.

- **Dashboard and Reporting**
  - Admin Dashboard: Provides a central dashboard for administrators to access all system functionalities and view key metrics.
  - Real-Time Analytics: Offers real-time data and analytics on membership, payments, and other key performance indicators.
  - Custom Reports: Generates custom reports based on specific criteria, aiding in strategic decision-making and planning.

- **Communication and Engagement**
  - Notifications: Sends automated notifications for payment reminders, membership renewals, and important club announcements.
  - Personalized Communication: Facilitates personalized communication with members through email and messaging features.

- **Data Security and Privacy**
  - Secure Data Storage: Ensures all data is securely stored in the MySQL database, protected against unauthorized access.
  - Compliance: Adheres to data protection regulations to ensure member

information is handled responsibly.

- **User-Friendly Interface**
    - Java Swing Interface: Provides an intuitive and straightforward graphical user interface designed with Java Swing, making it easy for users to navigate and perform tasks.
    - Responsive Design: Ensures the system is accessible across various devices, including desktops, tablets, and mobile phones.

- **Integration and Compatibility**
    - System Integration: Compatible with existing systems and software, facilitating seamless integration and minimizing disruptions during implementation.
    - API Support: Offers API support for integrating with third-party applications and services.

- **Training and Support**
    - Comprehensive Training: Provides detailed training sessions and materials to ensure administrators and staff can effectively use the system.
    - Ongoing Support: Offers continuous technical support to address any operational challenges and ensure smooth system operation.

CHAPTER 2

LITERATURE REVIEW

2.1 DOMAIN SPECIFIC

Literature survey is a very important step in the software development process. Before building any new tool, we need to check the time factor, economy and company strength. When these things are fulfilled, at that point following stages is to figure out which working framework and language can be utilized for building up the device. A lot of help is required for building the tool, internal as well as external. Senior programmers can help and provide this support to the developers from various sources like research papers, books or online websites. Before building the framework, the above thoughts are considered for building up the proposed framework.

2.2 DISCOVERING THE MAGIC OF FITNESS CLUB SYSTEM

The Easy Fitness Club Manager is more than just a software solution; it's a transformative tool that redefines the way fitness clubs operate. By integrating advanced features with an intuitive interface, this system unleashes the magic of efficient club management, enhancing both the administrative experience and member satisfaction. Here's how the Easy Fitness Club Manager brings its magic to life:

1. Effortless Member Management

- Streamlined Registration: The member registration process is quick and seamless, allowing new members to join with ease. Essential details are captured and securely stored, ensuring that all information is readily available for administrative tasks.
- Dynamic Member Profiles: Each member's profile is a hub of information, including personal details, membership type, attendance records, and payment history. This comprehensive view enables administrators to provide personalized service and support.

2. Seamless Payment Handling

- Automated Payment Tracking: The system automates the tracking of payments, ensuring that dues are accurately recorded and monitored. Administrators can easily view outstanding payments and send reminders to members.
- Financial Transparency: Detailed financial reports provide insights into revenue

streams, membership sales, and outstanding dues. This transparency helps in maintaining financial health and making informed decisions.

3. Comprehensive Class and Schedule Management

- Efficient Class Scheduling: Administrators can schedule and manage classes with ease, ensuring that members have access to a variety of sessions. Real-time updates on class availability and waitlists enhance the booking experience.

- Instructor Coordination: The system keeps track of instructor schedules, making it simple to assign and manage instructors for various classes. This ensures that all sessions are well-staffed and run smoothly.

4. Enhanced Member Engagement

- Personalized Communication: The system facilitates personalized communication with members, sending notifications and reminders about upcoming classes, renewals, and club events. This keeps members informed and engaged.

- Community Building: Features like member forums and event notifications foster a sense of community, encouraging members to participate in club activities and interact with each other.

5. Robust Data Security

- Secure Data Storage: All member and transaction data is securely stored in the MySQL database, protected against unauthorized access. This ensures that sensitive information remains confidential and secure.

- Compliance with Regulations: The system is designed to comply with data protection regulations, ensuring that the club's operations adhere to legal standards and best practices.

6. User-Friendly Interface

- Intuitive Design: The Java Swing-based interface is designed to be user-friendly, allowing administrators to navigate the system with ease. Tasks such as member registration, payment tracking, and class scheduling are straightforward and efficient.

- Responsive Accessibility: The system is accessible across multiple platforms, including desktops, tablets, and smartphones, allowing administrators to manage operations remotely and members to engage with the club conveniently.

7. Continuous Improvement and Adaptability

- Feedback Mechanisms: The system includes features for gathering feedback from members and administrators, ensuring that the club can continuously improve its services.

- Scalable and Adaptable: The Easy Fitness Club Manager is designed to grow with the club, easily adapting to new requirements and scaling to accommodate increasing numbers of members and services.

2.3 LITERATURE REVIEW

The development of fitness club management systems reflects a growing demand for streamlined administrative processes and improved member experiences through technological advancements. This review explores key studies and findings in this field, focusing on the role of digital solutions, data security, and technology frameworks.

- In order to ensure efficiency and user-friendly interfaces, **Dinesh et al. (2020)** concentrated on creating a "**Smart Gym Management System**" using primary data approach. The system's goal is to encourage users' physical fitness and health through sports activities and healthy eating.

- Using wearable sensors, **Thakur et al. (2022)** investigated "**Sensor-Based Gym Physical Exercise Recognition**" to track health. They discussed issues with data management and sensor accuracy, focusing on lifestyle promotion and early health issue identification.

- Using a web-based tool for member and inventory management, **Priti et al. (2020)** designed a "**Fitness Studio System**" to handle customers conveniently. Their method was designed to make bodybuilding fitness clubs' operations more efficient.

- The effect of an **"iOS Application with Firebase for Gym Membership Management"** on member satisfaction was examined by **David et al. (2019)**. Their analysis revealed higher member satisfaction and retention along with shorter wait times and better staff-member communication.

- **Ajitesh et al. (2022)** created a "**Fitness Management Website**" that reduced the time needed to complete administrative duties by up to 70%, improving gym operating efficiency. This freed up employees to concentrate on providing high-quality workout experiences.

- The "**Gym Check System**" was created by **Mushel et al. (2016)** to automate administrative duties, offer real-time insights into gym performance, and improve member engagement.

- An "**RFID-Enabled Gym Management System**" was introduced by **Wang et al. (2010)** to automate administrative operations, measure activity, and customize member experiences. The system's ultimate goal was to increase operational effectiveness and member engagement by offering real-time insights for data-driven decisions.

CHAPTER 3

FEASIBILITY STUDY

After doing the project, study and analyzing all the existing or required functionalities of the system, the next task is to do the feasibility study for the project. All projects are feasible-given unlimited resources and in finite time. Feasibility study includes consideration of all the possible ways to provide a solution to the given problem. The proposed solution should satisfy all the user requirements and should be flexible enough so that future changes can be easily done based on the future upcoming requirements. There are three parts in feasibility study

3.1 Operational Feasibility
3.2 Technical Feasibility
3.3 Economic Feasibility
3.4 Behavioral Feasibility

3.1 OPERATIONAL FEASIBILITY

Operational feasibility refers to the assessment of whether a project can be successfully implemented and integrated into the existing operational infrastructure. In the case of this augmented reality project, evaluating its operational feasibility involves considering various factors that impact its practical implementation.

- **Technological Infrastructure:** Assessing the availability and compatibility of the necessary hardware and software components is crucial. This includes computer devices capable of running java applications, as well as reliable internet connectivity to support the seamless integration of database.
- **Cost and Resources:** Conducting a cost-benefit analysis to determine the feasibility of the project within the available budget is crucial. This includes assessing the financial resources required for hardware, software, content creation, maintenance, and ongoing support.
- **Alignment with Club Operations:** The Easy Fitness Club Manager is designed specifically to meet the needs of fitness clubs. Its features, such as member registration, payment tracking, class scheduling, and instructor management, align with the day-to-day operations of these clubs, ensuring a smooth integration into existing workflows.
- **User-Friendly Interface:** The use of Java Swing ensures an intuitive and user-friendly interface, reducing the learning curve for administrators and staff. This ease of use is critical for effective daily operations. The application should be accessible and easy to use for a wide range of users, ensuring a seamless and

engaging user experience.

- **Improved Efficiency:** By automating routine tasks and providing real-time data access, the system enhances operational efficiency, allowing staff to focus on customer service and strategic activities.
- **Member Experience:** Features such as personalized communication, easy class booking, and secure payment handling improve the overall member experience, which is crucial for member retention and satisfaction.
- **Maintenance and Updates**: Considering the feasibility of maintaining and updating the application and its content over time is important. This involves planning for regular maintenance, bug fixes, and potential enhancements to keep the application up-to-date and aligned with evolving technologies and user expectations.

By evaluating these operational feasibility factors, the project team can determine the practicality and viability of implementing the java swing application solution to the gym's operation.

3.2 TECHNICAL FEASIBILITY

Technical feasibility refers to the evaluation of whether the proposed java swing project can be successfully developed, implemented, and integrated from a technical standpoint. Assessing technical feasibility involves considering various factors related to the project's technological aspects. Here are some key considerations:

- **Hardware Requirements:**

  Assessing the technical feasibility involves determining the hardware requirements for the project. This includes identifying the compatible devices that support the java swing application, as well as any additional internet connection.
- **Software Development:**

  Evaluating the technical feasibility of developing the augmented reality application involves assessing the availability of suitable software development tools and frameworks. This includes selecting the appropriate programming languages, libraries, and platforms for creating the application, considering factors such as cross-platform compatibility and support for java swing and MySql database.
- **Technology Stack**

  - **Front-End Development with Java Swing:** Java Swing is used for building the graphical user interface (GUI) of the Easy Fitness Club Manager. Swing is a part of Java Foundation Classes (JFC) that provides a set of 'lightweight' (all-Java language) components that, in turn, provide a wide range of functionality.

Advantages:

1. **Cross-Platform Compatibility:** Java Swing applications are platform-independent. This means the same application can run on Windows, MacOS, and Linux without modification.

2. **Rich Set of Components:** Swing provides a comprehensive set of GUI components such as buttons, tables, and lists, which can be customized as per the application's requirements.

3. **Robust Event Handling:** Swing has a sophisticated event-handling mechanism that makes it easier to create interactive applications.

4. **Ease of Use:** The framework is well-documented and has a large community, making it easier to find support and resources for development and troubleshooting.

Resources: Java Swing benefits from extensive documentation, numerous online tutorials, and a strong community support which can assist in solving development issues and improving the application.

- **Back-End Development with MySQL:** MySQL is chosen for the database management system (DBMS). MySQL is a relational database management system (RDBMS) known for its reliability, scalability, and ease of use.

Advantages:

1. **Performance:** MySQL is designed to handle large volumes of data efficiently. It can manage high-throughput applications, ensuring fast query processing and data manipulation.

2. **Reliability and Security:** MySQL offers robust security features, including SSL support, data encryption, and user management, ensuring the safety of sensitive member information.

3. **Scalability:** MySQL can be scaled vertically and horizontally, making it suitable for growing fitness clubs with increasing data and transaction volumes.

4. **Support and Community:** Like Java Swing, MySQL has extensive documentation and a large, active community, providing ample resources for troubleshooting and optimization.

- **Compatibility**
  1. **Multi-Platform Accessibility:** The Easy Fitness Club Manager is designed to be compatible across various devices and operating systems.
  2. **Desktop Compatibility:** The system is fully functional on all major desktop operating systems, including Windows, MacOS, and Linux. This ensures that administrators and staff can use the system on their preferred devices.
  3. **Mobile and Tablet Compatibility:** The system's interface is responsive and can be accessed via web browsers on tablets and smartphones. This enables administrators to manage club operations on-the-go and allows members to interact with the system from their mobile devices.
  4. **Cross-Browser Support:** The web components of the system are tested across different web browsers such as Chrome, Firefox, Safari, and Edge, ensuring consistent performance and user experience.

- **Scalability**
  1. **Database Scalability:** MySQL's architecture supports both vertical and horizontal scaling.
  2. **Vertical Scaling:** This involves increasing the capacity of a single server (e.g., upgrading the hardware resources like CPU, RAM, and storage) to handle more load.
  3. **Horizontal Scaling:** MySQL can be scaled horizontally by adding more servers to the database cluster. This is particularly useful for distributing the load and managing large volumes of data efficiently.

- **Application Scalability:**
  1. **Load Balancing:** The system can implement load balancing strategies to distribute the workload evenly across multiple servers, ensuring optimal performance even during peak times.
  2. **Modular Design:** The application is designed in a modular way, allowing for the addition of new features and functionalities without disrupting the existing system.

The technical feasibility of the Easy Fitness Club Manager is well-established, with robust technology choices that ensure reliability, compatibility, scalability, and seamless integration. Java Swing and MySQL provide a solid foundation for developing a feature-rich and user-friendly application that meets the needs of modern fitness clubs. The system's ability to scale and integrate with other tools and platforms further enhances its value, making it a sustainable and future-proof solution for fitness club management.

3.3 ECONOMICAL FEASIBILITY

Establishing Economic feasibility refers to the assessment of whether the proposed augmented reality project is financially viable and justifiable. It involves analyzing the project's costs and potential benefits to determine if the investment in the project is economically feasible. Here are key considerations for evaluating the economic feasibility of the project:

Economic feasibility assesses whether the Easy Fitness Club Manager is financially viable and cost-effective from the perspectives of development, implementation, and ongoing operation. This analysis considers the various costs involved and the anticipated financial benefits.

Development Costs

- **Open-Source Technologies**: The Easy Fitness Club Manager leverages open-source technologies such as Java for front-end development and MySQL for the back-end database. Utilizing these technologies significantly reduces development costs:
  - **No Licensing Fees:** Java and MySQL do not require expensive licensing fees, making them cost-effective choices for software development.
  - **Extensive Documentation and Community Support**: Both technologies have comprehensive documentation and large user communities, which can help reduce development time and costs through readily available resources and community-driven problem solving.
  - **Development Tools:** The use of free and open-source development environments and tools (such as Eclipse for Java development) further reduces costs.

Implementation Costs

- **Hardware and Software Setup:** Initial implementation costs include the setup of necessary hardware and software infrastructure:
  - **Hardware Costs:** This might involve purchasing or upgrading servers, computers, and network equipment. However, given the scalability of the system, initial hardware investments can be kept to a minimum and scaled up as needed.
  - **Software Installation:** Installation and configuration of the Java runtime environment and MySQL database on the servers are straightforward and do not incur additional costs.
- **Training:** Ensuring that staff and administrators are well-trained to use the system is critical:
  - **Training Programs:** Costs include developing training materials and conducting training sessions. These are one-time expenses that yield significant returns in terms of system usability and efficiency.
  - **Technical Support**: Initial technical support might be needed to help staff adapt to the new system. This can be managed through a combination of in-house training sessions and support from the software developers.
- **Data Migration:** Migrating data from legacy systems to the new system involves:
  - **Data Cleaning and Preparation:** Ensuring data accuracy and integrity during

migration.

  - **Migration Tools and Scripts:** Using automated tools and custom scripts to transfer data, reducing manual efforts and errors.
- **Testing and Quality Assurance:** Comprehensive testing (unit, integration, and user acceptance testing) ensures that the system functions correctly, which might incur costs related to temporary test environments and potential rework based on feedback.

Operational Savings

- **Automation of Administrative Tasks:** The system automates many routine administrative tasks, such as member registration, payment processing, and class scheduling:
  - **Labor Cost Reduction:** By reducing the need for manual data entry and management, the system saves on labor costs. Staff can be reallocated to more value-added activities, improving overall productivity.
  - **Efficiency Gains:** Automated notifications and reminders reduce the likelihood of missed payments or appointments, enhancing operational efficiency and member satisfaction.
- **Resource Optimization:** Improved tracking and management of resources, such as scheduling of classes and instructors, lead to better utilization of facilities and human resources.

Return on Investment (ROI)

- **Operational Efficiency:** The system enhances operational efficiency by streamlining processes and reducing errors. This leads to time savings and better use of staff, directly impacting the club's bottom line.
- **Improved Member Retention**: Enhanced customer relationship management (CRM) capabilities and personalized communication contribute to higher member satisfaction and retention rates, which are critical for sustained revenue growth.
- **Financial Management:** The system's ability to generate detailed financial reports and track transactions accurately ensures better financial oversight and planning:
  - **Revenue Growth:** Accurate tracking of payments and dues helps in timely collection of fees, reducing revenue leakage.
  - **Cost Control:** Insightful financial reports help in identifying cost-saving opportunities and areas requiring budget adjustments.

Long-Term Benefits

- **Increased Member Satisfaction and Retention:** By providing a seamless and engaging member experience, the system fosters loyalty and long-term membership:
  - **Personalized Services:** Tailored communications and personalized services enhance member engagement and satisfaction.
  - **Community Building:** Features that facilitate community interactions and events strengthen member loyalty.
- **Scalability and Future-Proofing:** The system's scalable architecture ensures it can grow with the club:
  - **Adaptability:** The system can be easily updated with new features and

functionalities, ensuring it remains relevant as the club's needs evolve.

  - **Sustained Competitive Advantage:** Keeping up with technological advancements and member expectations helps the club maintain a competitive edge in the market.

- **Cost-Effectiveness Over Time:** The initial investment in the system is offset by long-term savings and revenue growth, making the project economically viable:
  - **Reduced Maintenance Costs:** The use of reliable open-source technologies reduces ongoing maintenance and licensing costs.
  - **Enhanced Decision-Making:** Access to real-time data and analytics supports informed decision-making, contributing to the strategic growth of the club.

The economic feasibility of the Easy Fitness Club Manager is well-supported by the use of cost-effective open-source technologies, manageable implementation costs, significant operational savings, and a strong return on investment. The long-term benefits, including increased member satisfaction and retention, further justify the initial investment. Overall, the project promises to deliver substantial financial benefits, making it a sound and economically feasible solution for fitness club management.

3.4 BEHAVIORAL FEASIBILITY

Behavioral feasibility refers to the assessment of whether the proposed augmented reality project is acceptable and practical from a behavioral or human perspective. It involves considering the potential impact on user behavior, attitudes, and acceptance of the project. Behavioral feasibility evaluates whether the stakeholders, including administrators, staff, and members, will accept and effectively use the new Easy Fitness Club Manager system. This involves understanding the readiness and willingness of the users to adopt the new system and ensuring that any resistance is effectively managed. Here are key considerations for evaluating the behavioral feasibility of the project:

- **Stakeholder Acceptance**
  - **Initial Feedback:**
    - **Surveys and Interviews:** Conducting surveys and interviews with potential users, including administrators, staff, and members, has shown a positive reception to the proposed features and functionalities. Feedback mechanisms have been established to continuously gather input and refine the system based on user needs and expectations.
    - **Addressing Pain Points:** The system is designed to resolve common issues faced by fitness club staff, such as inefficient member registration processes, difficulty tracking payments, and managing class schedules. By directly addressing these pain points, the system increases its likelihood of acceptance.
    - **Demonstrations and Pilot Testing:** Initial demonstrations and pilot testing sessions have been conducted to showcase the system's capabilities and gather user feedback. Positive reactions and constructive suggestions have been incorporated into the system design, enhancing its overall acceptance.

- **Training and Support**
  - **Comprehensive Training Programs:**
    - **Initial Training Sessions:** Detailed training sessions are planned for all user groups, including administrators, front-desk staff, and fitness trainers. These sessions will cover all aspects of the system, from basic navigation to advanced functionalities.
    - **Ongoing Training:** Refresher courses and advanced training sessions will be offered periodically to ensure that users remain proficient in using the system and can leverage new features as they are introduced.
    - **User Manuals and Tutorials:** Comprehensive user manuals, video tutorials, and FAQs will be provided to assist users in understanding and navigating the system independently.
  - **Technical Support:**
    - **On-Site Support:** Initially, on-site support will be provided to address any issues that arise during the early stages of system implementation. This hands-on assistance will help in quickly resolving any problems and building user confidence.
- **User Engagement**
  - **User-Friendly Interface:**
    - **Intuitive Design:** The system's interface is designed to be intuitive and easy to navigate, reducing the learning curve for new users. Clear labeling, logical navigation paths, and user-friendly forms enhance overall usability.
    - **Customization Options:** Users can customize their dashboards and views according to their preferences, which helps in making the system more engaging and personalized.
  - **Practical Features:**
    - **Automated Notifications:** Automated reminders and notifications for payments, class schedules, and membership renewals help keep users informed and engaged.
    - **Member Portal:** A dedicated member portal allows users to manage their profiles, view schedules, and make payments conveniently, enhancing their overall experience and engagement with the club.
    - **Community Features:** Features such as discussion forums, event calendars, and social media integration help build a sense of community among members, encouraging regular use and interaction with the system.
- **Change Management**
  - **Structured Change Management Plan:**
    - **Stakeholder Involvement:** Involving key stakeholders in the planning and implementation phases ensures their buy-in and support. Regular updates and consultations with stakeholders help in addressing any concerns and aligning the system with their expectations.
    - **Clear Communication:** Transparent communication about the benefits and changes the new system will bring is crucial. Informative sessions and regular updates help users understand how the system will improve their work processes and member interactions.

- **Phased Implementation:** Implementing the system in phases allows users to adapt gradually. Starting with non-critical functionalities and progressively integrating more complex features helps in managing change effectively.

Behavioral feasibility for the Easy Fitness Club Manager is strong, supported by positive stakeholder feedback, comprehensive training and support programs, user-friendly design, and a well-structured change management plan. By addressing user needs, facilitating ease of use, and ensuring ongoing support, the system is poised for successful adoption and effective utilization by all stakeholders.

The Easy Fitness Club Manager is operationally, technically, economically, and behaviorally feasible. It aligns well with the operational needs of fitness clubs, utilizes reliable and scalable technology, offers a strong ROI, and is likely to be accepted and effectively used by stakeholders. These factors collectively ensure the successful implementation and sustained use of the system, driving efficiency and enhancing member satisfaction in fitness clubs.

CHAPTER 4

REQUIREMENT ANALYSIS

4.1 METHODOLOGIES FOLLOWED

In the development of a java swing project, various methodologies can be employed. One commonly used methodology is the Scrum framework, which is an agile approach to project management. Here's an explanation of how the Scrum methodology can be applied:

4.2 Scrum Methodology:

Description: Scrum is an iterative and incremental framework that promotes collaboration, flexibility, and continuous improvement. It divides the project into time-bound iterations called sprints, typically lasting 2-4 weeks, during which a set of prioritized tasks are completed.

Application, we utilize the Scrum methodology as follows:

- **Product Backlog:** Create a product backlog, which is a prioritized list of features, functionalities, and tasks required for the augmented reality project. The backlog items are typically defined as user stories, representing the needs and expectations of the end-users.
- **Sprint Planning:** At the beginning of each sprint, conduct a sprint planning meeting. During this meeting, the team selects a subset of items from the product backlog to be worked on during the sprint. The team estimates the effort required for each task and determines the sprint goal, which represents the desired outcome of the sprint.
- **Daily Scrum:** Hold daily scrum meetings, also known as daily stand-ups. These brief meetings serve to synchronize the team, discuss progress, and identify any obstacles or issues that need to be addressed. Each team member shares their accomplishments, plans, and potential challenges.
- **Sprint Execution:** Throughout the sprint, the team works on the tasks identified during the sprint planning. The team collaborates closely, with regular communication and coordination, to develop and integrate the augmented reality application's features and functionalities.

- **Sprint Review:** At the end of each sprint, hold a sprint review meeting to showcase the completed work to stakeholders, including the augmented reality application's features, functionalities, and any other deliverables. Collect feedback and review whether the sprint goal was achieved.
- **Sprint Retrospective:** After the sprint review, conduct a sprint retrospective meeting. This retrospective allows the team to reflect on the sprint, discuss what went well and what could be improved, and identify action items for enhancing the development process in subsequent sprints.
- **Iterative Development:** Repeat the sprint cycle by selecting new items from the product backlog and continuing the iterative development process. The team learns from each sprint, adjusts priorities, and continuously improves the augmented reality application based on feedback and evolving requirements.

By applying the Scrum methodology, the project team benefits from improved collaboration, flexibility in adapting to changing requirements, and regular feedback cycles. The iterative nature of Scrum allows for the early and continuous delivery of valuable features, ensuring that the augmented reality project remains aligned with user expectations and generates a high-quality product.

4.3 FUNCTIONAL REQUIREMENT

The functional requirements for the Easy Fitness Club Manager outline the specific functionalities that the system must provide to meet the needs of the stakeholders. These requirements ensure that the system supports all necessary operations for managing a fitness club efficiently.

1. Login and Authentication

- **Secure Login:**
  - Implement a secure login system for administrators, staff, trainers, and members.
  - Use role-based access control to ensure that each user type has appropriate access levels.
  - Support multi-factor authentication for enhanced security.
- **User Roles and Permissions:**
  - Define user roles (e.g., admin, staff, trainer, member) and assign specific permissions to each role.
  - Allow administrators to manage user roles and permissions.

2. Member Registration and Management

- **New Member Registration:**
  - Provide a user-friendly interface for registering new members.
  - Capture essential details, including name, contact information, membership type, payment details, and emergency contact information.
  - Store member information securely in the MySQL database.
- **Member Profile Management:**
  - Allow members and administrators to update contact information and personal details.
  - Enable administrators to update membership status (active, inactive, suspended).

· Provide functionality for uploading and managing member photos.

- **Membership Renewal:**
  · Send automated reminders for upcoming membership renewals via email and SMS.
  · Provide an easy renewal process for members, including online payment options.
  · Track and manage membership expiration dates.

3. Payment Tracking and Management

- **Payment Processing:**

Implement secure payment processing for membership fees, class fees, and other charges.

Support multiple payment methods, including credit/debit cards, bank transfers, and online payment gateways.

- **Payment History:**
  · Maintain detailed records of all payments made by members, including date, amount, payment method, and purpose.
  · Allow members to view their payment history through their profiles.
- **Outstanding Dues:**
  · Track outstanding payments and send automated reminders to members with pending dues.
  · Provide administrators with tools to manage and follow up on overdue payments.

4. Class and Schedule Management

- **Class Scheduling:**
  · Allow administrators and trainers to create and manage class schedules, including class name, description, time, duration, instructor, and location.
  · Provide a calendar view for easy management of class schedules.
- **Personalized Communication:**
  · Allow administrators and trainers to send personalized messages to members.

The functional requirements for the Easy Fitness Club Manager are designed to provide a comprehensive and efficient solution for managing fitness club operations. By addressing key functionalities such as secure login, member registration, payment processing, class scheduling, reporting, and communication, the system ensures that all stakeholder needs are met. This thorough set of functional requirements will guide the development process and help deliver a user-friendly and effective fitness club management system.

4.4 NON-FUNCTIONAL REQUIREMENT

Non-functional requirements, also known as quality attributes or constraints, define the characteristics and constraints of the system beyond its functionality. These requirements describe how the system should perform, rather than what it should do. Non-functional requirements are often related to performance, reliability, security, usability, and other aspects that contribute to the overall system quality. Examples include response time, system availability, data encryption, user interface design, and regulatory compliance.

These requirements ensure that the Easy Fitness Club Manager is not only functional but also reliable, secure, and user-friendly.

1. Performance

- **Response Time:**

  · The system should have a response time of less than 2 seconds for user interactions under normal operating conditions.

  · Critical operations such as login, member registration, and payment processing should be completed within 5 seconds.

- **Throughput:**

  · The system should be able to handle a minimum of 500 concurrent users without performance degradation.

  · Ensure the system can process at least 100 transactions per minute.

- **Availability:**

  · The system should have an uptime of 99.9%, ensuring it is available 24/7 with minimal downtime.

  · Scheduled maintenance should be planned during off-peak hours to minimize impact on users.

2. Scalability

- **Vertical Scalability:**

  · The system should be designed to scale vertically by upgrading the hardware resources, such as adding more CPU and memory to the server.

- **Horizontal Scalability:**

  · The system should support horizontal scalability by adding more servers to distribute the load and improve performance.

  · The database and application should support load balancing to handle increased user and transaction volumes as the fitness club grows.

3. Security

- **Data Encryption:**

  · Store sensitive data, such as passwords and payment information, using strong encryption algorithms (e.g., AES-256).

- **Authentication and Authorization:**

  · Use role-based access control (RBAC) to ensure that users have access only to the functionalities and data they need.

- **Data Protection:**

- Ensure compliance with data protection regulations (e.g., GDPR, CCPA) to protect member privacy and sensitive information.
- Implement regular security audits and vulnerability assessments to identify and mitigate security risks.

4. Usability

- **User Interface:**
  - The interface should be intuitive and easy to navigate, with a clean and modern design.
  - Use consistent design elements and terminology throughout the system to reduce the learning curve.
- **Accessibility:**
  - Ensure the system is accessible to users with disabilities, complying with accessibility standards such as WCAG 2.1.

5. Reliability

- **Error Handling:**
  - Implement robust error-handling mechanisms to provide meaningful error messages and guide users on how to proceed.
  - Ensure the system can gracefully recover from errors without losing data or compromising functionality.
- **Backup and Recovery:**
  - Implement automated backup procedures to ensure data integrity and availability.
  - Ensure that backups are stored securely and can be restored quickly in case of data loss or system failure.

6. Maintainability

- **Code Quality:**
  - Ensure the codebase follows best practices and coding standards to enhance readability and maintainability.
  - Use modular design principles to make it easier to update or replace individual components without affecting the entire system.
- **Documentation:**
  - Provide comprehensive documentation for the system, including user manuals, API documentation, and developer guides.
  - Keep documentation up to date with the latest system changes and updates.

7. Compliance

- **Regulatory Compliance:**
  - Ensure the system complies with relevant industry standards and regulations, such as PCI DSS for payment processing and HIPAA for handling health-related data.
  - Conduct regular compliance audits to verify adherence to these standards.

In a project report, a flowchart can be used to illustrate the various steps involved in the project. For example, a flowchart could be used to show the steps involved in developing a software application, from requirements gathering to testing and deployment. By using a flowchart, project stakeholders can better understand the project workflow and identify areas where improvements can be made.

The non-functional requirements for the Easy Fitness Club Manager ensure that the system is not only functional but also meets high standards of performance, scalability, security, usability, reliability, maintainability, and compliance. These requirements are essential for delivering a robust, efficient, and user-friendly fitness club management system that meets the needs of all stakeholders and supports the club's operations effectively.

4.5 SOFTWARE REQUIREMENT

Table 4.1 Software Requirement for Fitness Club System

S. NO.	DESCRIPTION	TYPE
1	Operating System	Window 7
2	Language	Java,Java Swing
3	IDE	NetBeans
4	Database	MySQL

4.6 HARDWARE REQUIREMENT

Table 4.2 Hardware Requirement for Fitness Club System

S. NO.	DESCRIPTION	TYPE
1	Hardware	Intel Core i3
2	Clock Speed	2.37GHz
3	RAM	2GB
4	SSD	128GB

CHAPTER 5

SYSTEM ARCHITECTURE AND DESIGN

The system architecture and design for the Easy Fitness Club Manager provide a blueprint for how the system components interact, the technologies used, and how the overall system is structured to meet both functional and non-functional requirements. The architecture is designed to be modular, scalable, and secure, ensuring a robust solution for managing fitness club operations.

1. System Architecture Overview

The Easy Fitness Club Manager follows a three-tier architecture consisting of:

a. **Presentation Layer:** This layer handles the user interface and user interaction, implemented using Java Swing.

b. **Application Layer:** This layer contains the business logic, processing user requests, and coordinating between the presentation layer and the data layer.

c. **Data Layer:** This layer manages data storage and retrieval, implemented using MySQL.

2. Technology Stack

- **Front-End:** Java Swing for creating a user-friendly graphical interface.
- **Back-End:** Java for business logic and application processing.
- **Database:** MySQL for reliable data storage and management.
- **Integration:** Java JDBC for integrating with external systems and third-party services.

3. Detailed Design

a. Presentation Layer

- **User Interface Components:**
  - **Login Screen:** Secure login form with username, password, and optional MFA.

    - **Dashboard:** Centralized interface for administrators to manage members, payments, schedules, and reports.
    - **Member Management:** Forms and tables for adding, updating, and deleting member information.
    - **Payment Management:** Interfaces for processing payments, viewing payment history, and managing dues.
    - **Reports:** Tools for generating financial, membership, and attendance reports.

b. Application Layer

- **Business Logic Components:**
    - **Authentication Module:** Handles user authentication, role-based access control, and session management.
    - **Member Management Module:** Manages CRUD (Create, Read, Update, Delete) operations for member data.
    - **Payment Processing Module:** Handles payment tracking.
    - **Reporting Module:** Generates various reports and analytics for administrators.
- **Service Layer:**
    - **Database Services:** Provides JDBC for external integration, such as payment CRM systems.

c. Data Layer

- **Database Schema:**

  Tables:
    - **Users:** Stores user information, including roles and authentication details.
    - **Members:** Stores member profiles, contact information, and membership details.
    - **Payments:** Tracks all payment transactions, including amounts, dates, and payment methods.
- **Data Access Objects (DAO):**
    - Implements CRUD operations for each table.
    - Ensures secure and efficient data retrieval and manipulation.

d. Security Design

- **Authentication and Authorization:**

- Implement role-based access control (RBAC) to restrict access based on user roles.
- Support multi-factor authentication (MFA) for enhanced security.

- **Data Protection:**
  - Regularly update and patch the system to protect against vulnerabilities.

4. Integration Design

- **APIs:**
  - Develop JDBC Connection for external integrations and ensure secure data exchange.
  - Document API endpoints and provide usage guidelines for third-party developers.

5. Deployment Architecture

- **Server Infrastructure:**
  - Deploy the application on a reliable cloud platform, to ensure high availability and scalability.
  - Use load balancers to distribute traffic and improve performance.
  - Implement a robust backup and disaster recovery plan to ensure data integrity.

The system architecture and design for the Easy Fitness Club Manager provide a detailed blueprint for building a robust, scalable, and secure fitness club management system. By leveraging a three-tier architecture, well-established technologies, and comprehensive security measures, the system is designed to meet the needs of all stakeholders effectively. The modular design ensures ease of maintenance and scalability, allowing the system to grow with the fitness club and adapt to changing requirements.

5.1 NetBeans and MySQL Integration

Steps for Integration:

1. **Add JDBC Library:** Ensure the MySQL JDBC driver is included in your project.
2. **Establish Database Connection:** Use JDBC to connect to the MySQL database.
3. **Execute SQL Queries:** Perform CRUD operations using SQL statements.
4. **Handle Results:** Process the results returned from the database.
5. **Close Resources:** Properly close all database resources to prevent leaks.

To integrate Java with MySQL through JDBC using the NetBeans IDE, follow these detailed steps:

Step-by-Step Integration in NetBeans IDE

1. Set Up Your Project

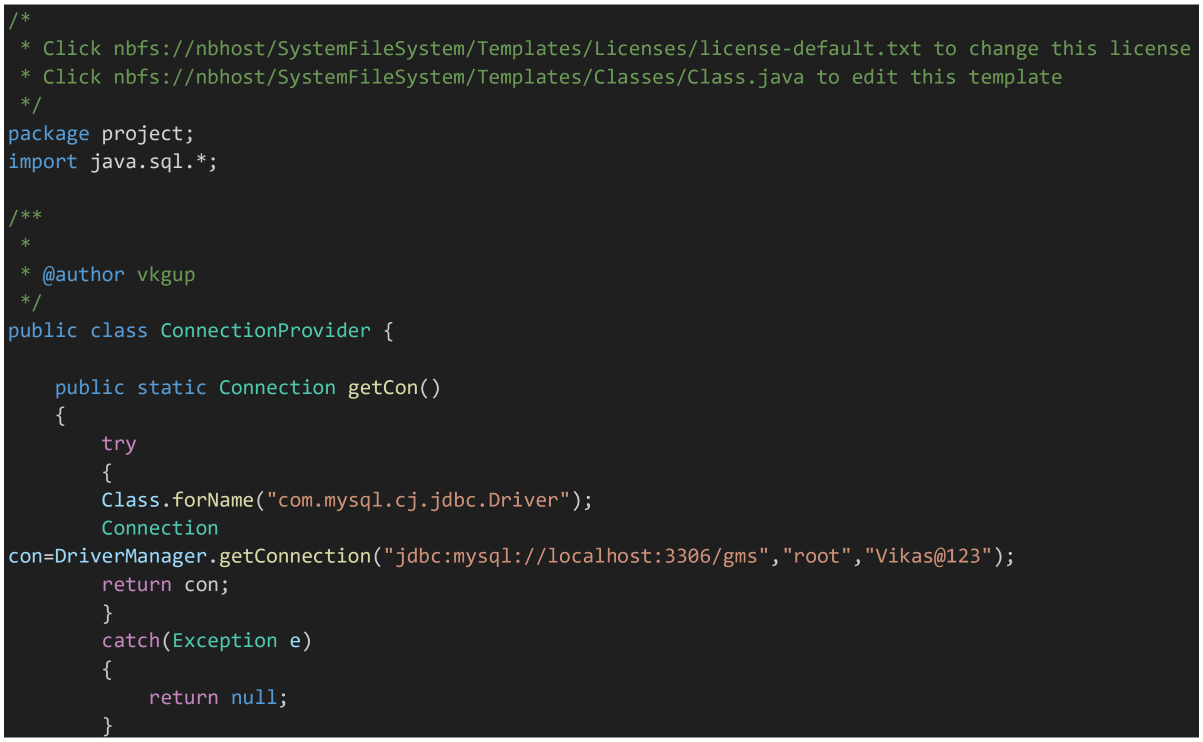
1. **Open NetBeans**: Start the NetBeans IDE and create a new Java project.
   - **File** > **New Project** > **Java** > **Java Application** > **Next** > Enter project name and location > **Finish**.

2. Add MySQL JDBC Driver to Your Project

1. **Download the MySQL JDBC Driver**:
2. **Add the JDBC Driver to Your Project**:
   - Right-click on your project in the Projects pane.
   - Select **Properties**.
   - In the Properties dialog, select **Libraries** from the categories on the left.
   - Click **Add JAR/Folder**.
   - Navigate to and select the **mysql-connector-java.jar** file.
   - Click **Open** to add the JAR file to your project's classpath.

3. Establish Database Connection

1. **Create a Class for Database Connection**:
   - Right-click on your project > **New** > **Java Class**.
   - Name the class **ConnectionProvider**.



```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package project;
import java.sql.*;

/**
 *
 * @author vkgup
 */
public class ConnectionProvider {

    public static Connection getCon()
    {
        try
        {
        Class.forName("com.mysql.cj.jdbc.Driver");
        Connection
con=DriverManager.getConnection("jdbc:mysql://localhost:3306/gms","root","Vikas@123");
        return con;
        }
        catch(Exception e)
        {
            return null;
        }
    }
}
```

3. **Execute SQL Queries**
   Write SQL queries on the project classes to access the Database information:

5.2. ADDING A NEW MEMBER

Adding a new member to the Easy Fitness Club Manager involves a simple and efficient process. Here's a brief overview:

1. **User Interface (UI) Components:**
   - Text Fields: Administrators enter member details such as name, contact information, and membership type.
   - Submit Button: Saves the new member's information.
2. **Data Access Object (DAO) Methods:**
   - Insert Method: Constructs and executes an SQL INSERT statement to add the new member's details to the database.
3. **Service Layer:**
   - Add Member: Validates the input data before calling the DAO's insert method to ensure all information is correct and complete.
4. **Database Operations:**
   - Insert Operation: The MySQL database stores the new member's information using the executed SQL INSERT statement.

Integration Flow

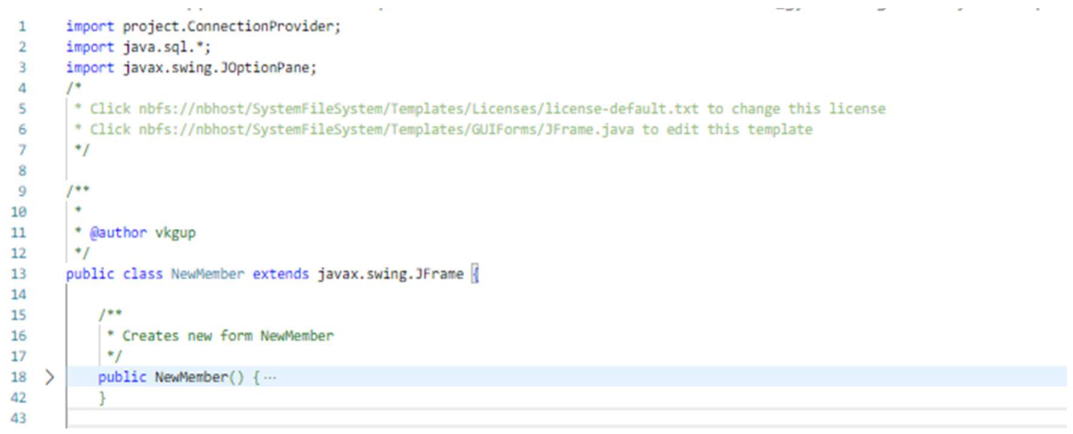
1. Enter Member Details: Administrator inputs member details in the UI.
2. Submit Member Details: Administrator clicks the submit button.
3. Validate and Insert: Service layer validates the details and calls DAO to insert the new member into the database.

Conclusion

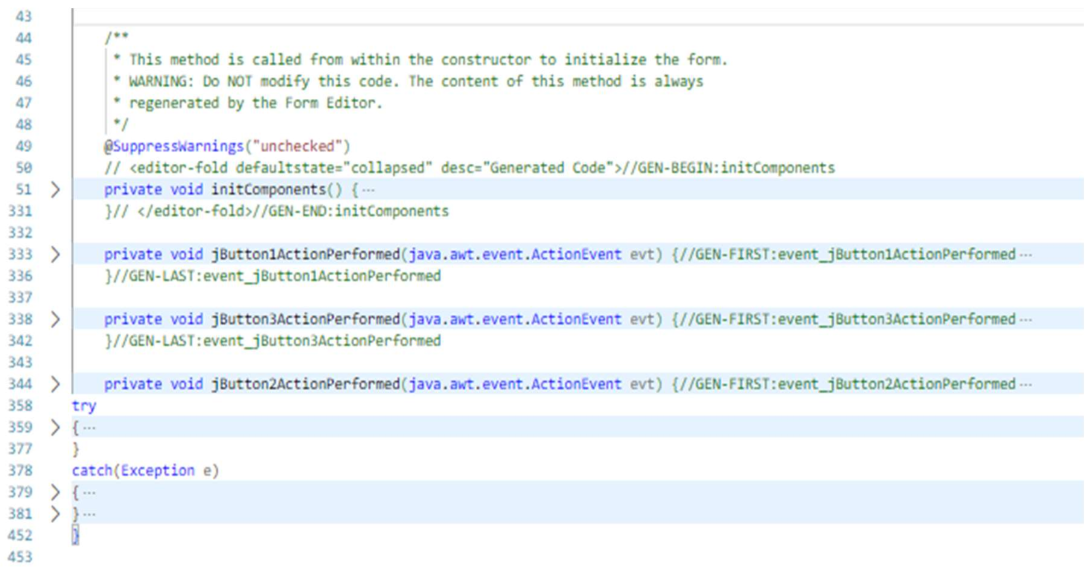
This process ensures that new members are added efficiently and accurately, enhancing the overall management of the fitness club.

Create a AddMember Class:

- Right-click on your project > New > Java Class.
- Name the class **AddMember**.



```
import project.ConnectionProvider;
import java.sql.*;
import javax.swing.JOptionPane;
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template
 */

/**
 *
 * @author vkgup
 */
public class NewMember extends javax.swing.JFrame {

    /**
     * Creates new form NewMember
     */
    public NewMember() {...
    }
```



```
    /**
     * This method is called from within the constructor to initialize the form.
     * WARNING: Do NOT modify this code. The content of this method is always
     * regenerated by the Form Editor.
     */
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
    private void initComponents() {···
    }// </editor-fold>//GEN-END:initComponents

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_jButton1ActionPerformed···
    }//GEN-LAST:event_jButton1ActionPerformed

    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_jButton3ActionPerformed···
    }//GEN-LAST:event_jButton3ActionPerformed

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_jButton2ActionPerformed···
try
{···
}
catch(Exception e)
{···
}···
}
```

Fig. 5.1 AddMember Code

5.3 UPDATE AND DELETE DETAILS

The **UpdateDeleteMember** class is designed to handle the functionality of updating and deleting member records in the fitness club management system. Below is a detailed explanation of the class and its components.

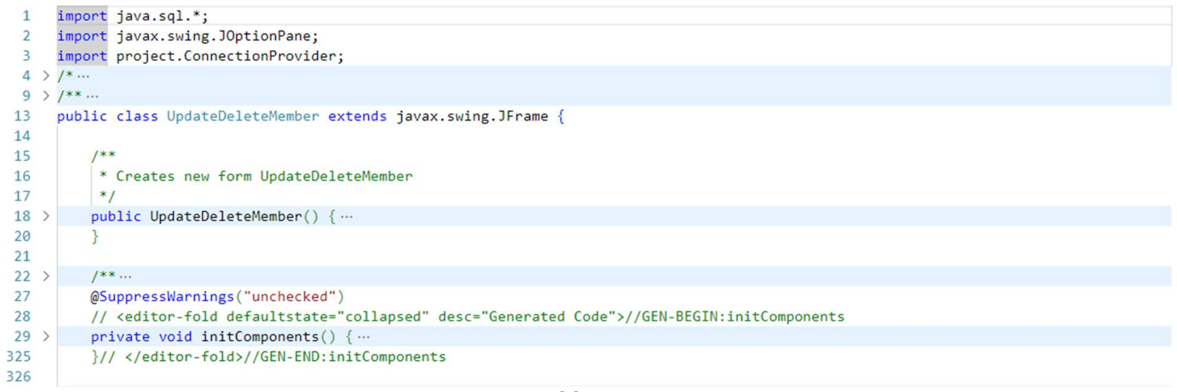
Class Structure

The `UpdateDeleteMember` class typically includes the following key components:

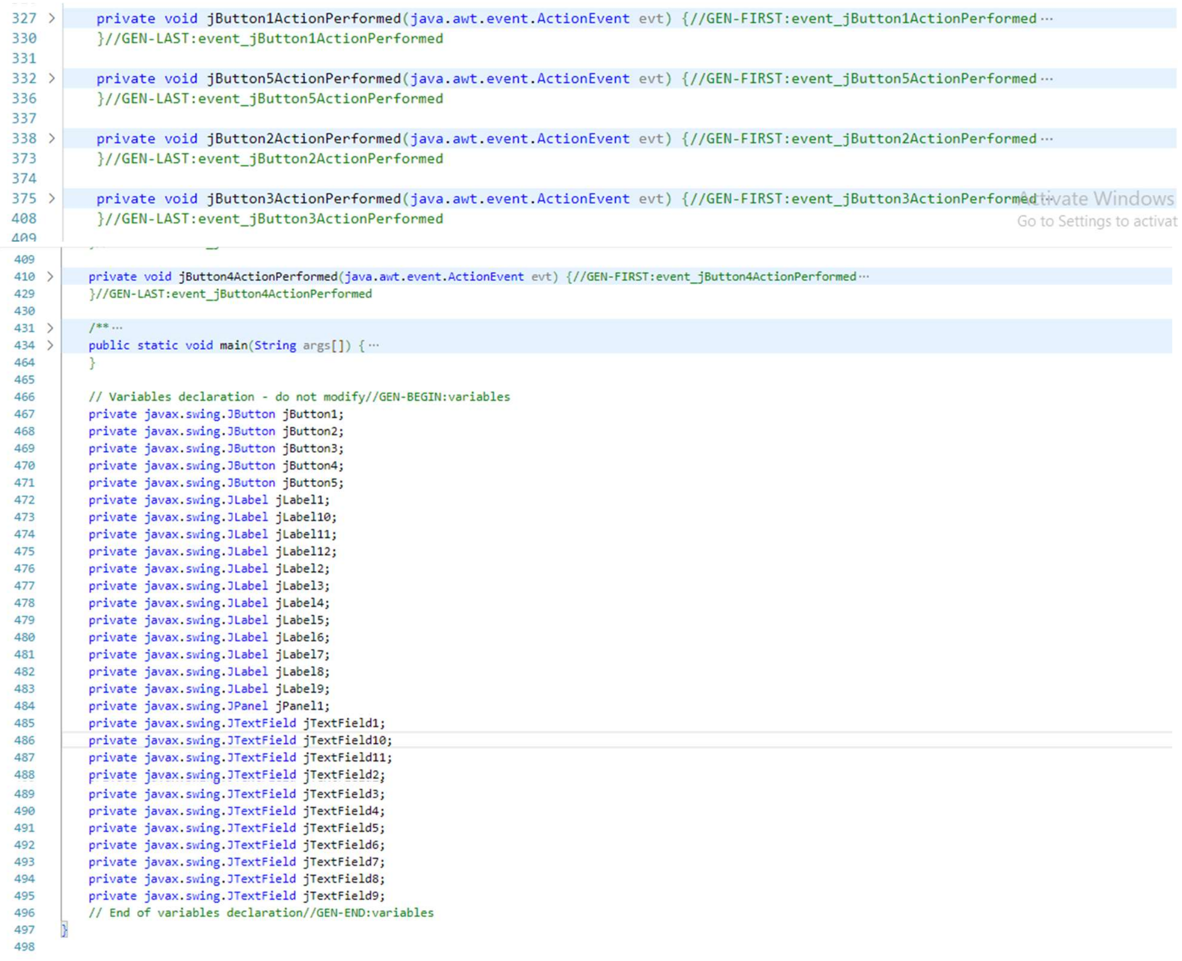
- **Imports:** Importing necessary packages.
- **Connection Setup:** Establishing a connection to the MySQL database.
- **Update Member Method:** Method to update member details.
- **Delete Member Method:** Method to delete a member.
- **Main Method or Integration:** Demonstrating usage or integrating with other parts of the application.

Create a UpdateDeleteMember Class:

- Right-click on your project > New > Java Class.
- Name the class **UpdateDeleteMember**.



```
import java.sql.*;
import javax.swing.JOptionPane;
import project.ConnectionProvider;
/*···
/**···
public class UpdateDeleteMember extends javax.swing.JFrame {

    /**
     * Creates new form UpdateDeleteMember
     */
    public UpdateDeleteMember() {···
    }

    /**···
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
    private void initComponents() {···
    }// </editor-fold>//GEN-END:initComponents
```



```
    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_jButton1ActionPerformed ···
    }//GEN-LAST:event_jButton1ActionPerformed

    private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_jButton5ActionPerformed ···
    }//GEN-LAST:event_jButton5ActionPerformed

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_jButton2ActionPerformed ···
    }//GEN-LAST:event_jButton2ActionPerformed

    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_jButton3ActionPerformed ···
    }//GEN-LAST:event_jButton3ActionPerformed

    private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_jButton4ActionPerformed ···
    }//GEN-LAST:event_jButton4ActionPerformed

    /** ···
    public static void main(String args[]) { ···
    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JButton jButton3;
    private javax.swing.JButton jButton4;
    private javax.swing.JButton jButton5;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel10;
    private javax.swing.JLabel jLabel11;
    private javax.swing.JLabel jLabel12;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JLabel jLabel4;
    private javax.swing.JLabel jLabel5;
    private javax.swing.JLabel jLabel6;
    private javax.swing.JLabel jLabel7;
    private javax.swing.JLabel jLabel8;
    private javax.swing.JLabel jLabel9;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JTextField jTextField1;
    private javax.swing.JTextField jTextField10;
    private javax.swing.JTextField jTextField11;
    private javax.swing.JTextField jTextField2;
    private javax.swing.JTextField jTextField3;
    private javax.swing.JTextField jTextField4;
    private javax.swing.JTextField jTextField5;
    private javax.swing.JTextField jTextField6;
    private javax.swing.JTextField jTextField7;
    private javax.swing.JTextField jTextField8;
    private javax.swing.JTextField jTextField9;
    // End of variables declaration//GEN-END:variables
}
```

Fig. 5.2 UpdateDeleteMember Code

5.4 RETRIVE THE LIST OF MEMBERS OF THE GYM

The Easy Fitness Club Manager includes a feature that displays the details of all members on a single screen. This functionality is implemented as follows:

Key Components

1. **User Interface (UI) Components:**
   - Table: Displays all member details such as name, contact information, membership type, and status on a single screen.
2. **Data Access Object (DAO) Methods:**
   - Fetch Method: Executes an SQL SELECT statement to retrieve member records from the Members table in the database.
3. **Service Layer:**
   - Get Members: Calls the DAO's fetch method to retrieve member data, processes it as needed, and sends it to the UI.
4. **Database Operations:**
   - Select Operation: Executes an SQL SELECT statement to fetch member

records from the Members table.

Integration Flow

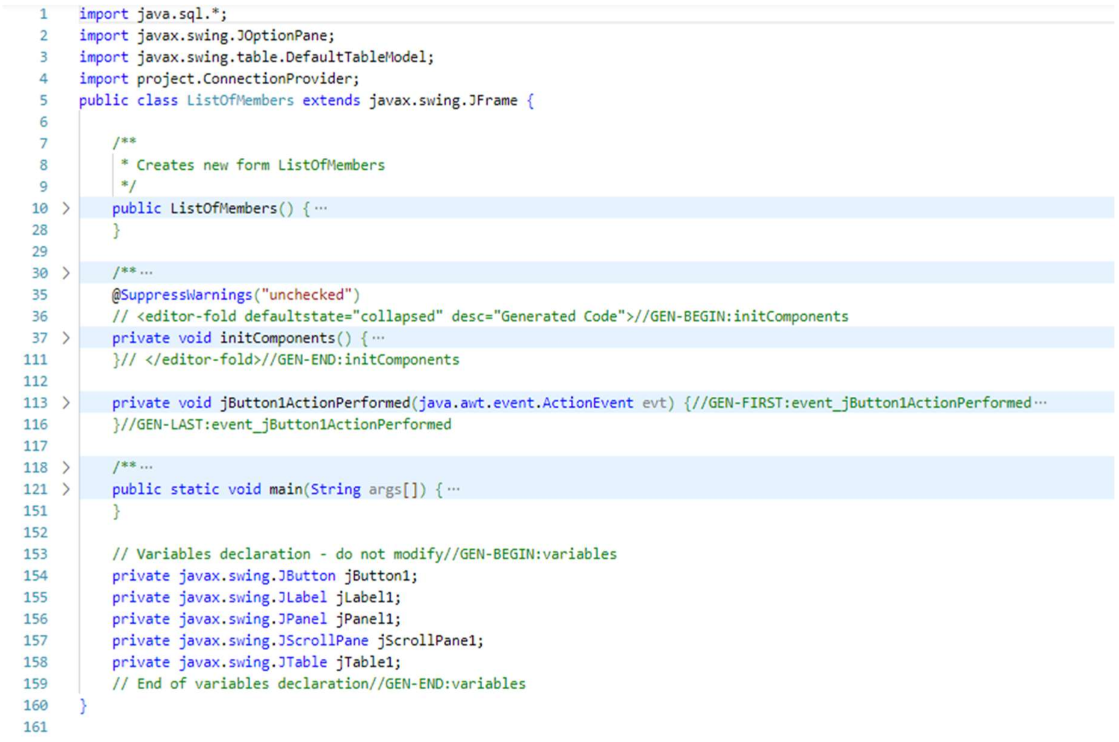
1. **Fetch Member Data:** The UI sends a request to the service layer.
2. **Retrieve and Process Data:** The service layer calls the DAO, retrieves the data, processes it if needed, and sends it back to the UI.
3. **Display Member List:** The UI displays all member details from the Members table in a table format on a single screen.

Conclusion

This feature provides a straightforward way for administrators to view all member information at once. The integration of a user-friendly UI, robust service layer logic, and efficient DAO methods ensures that member details from the Members table are easily accessible and manageable, enhancing the overall administration of the fitness club.

Create a ListOfMembers Class:

- Right-click on your project > New > Java Class.
- Name the class **ListOfMembers.java**.



```java
import java.sql.*;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;
import project.ConnectionProvider;
public class ListOfMembers extends javax.swing.JFrame {

    /**
     * Creates new form ListOfMembers
     */
    public ListOfMembers() {⋯
    }

    /**⋯
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
    private void initComponents() {⋯
    }// </editor-fold>//GEN-END:initComponents

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_jButton1ActionPerformed⋯
    }//GEN-LAST:event_jButton1ActionPerformed

    /**⋯
    public static void main(String args[]) {⋯
    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JButton jButton1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JPanel jPanel1;
    private javax.swing.JScrollPane jScrollPane1;
    private javax.swing.JTable jTable1;
    // End of variables declaration//GEN-END:variables
}
```

Fig. 5.3 ListOfMembers Code

5.5 TRACKING THE PAYMENT AND PAYMENT HISTORY

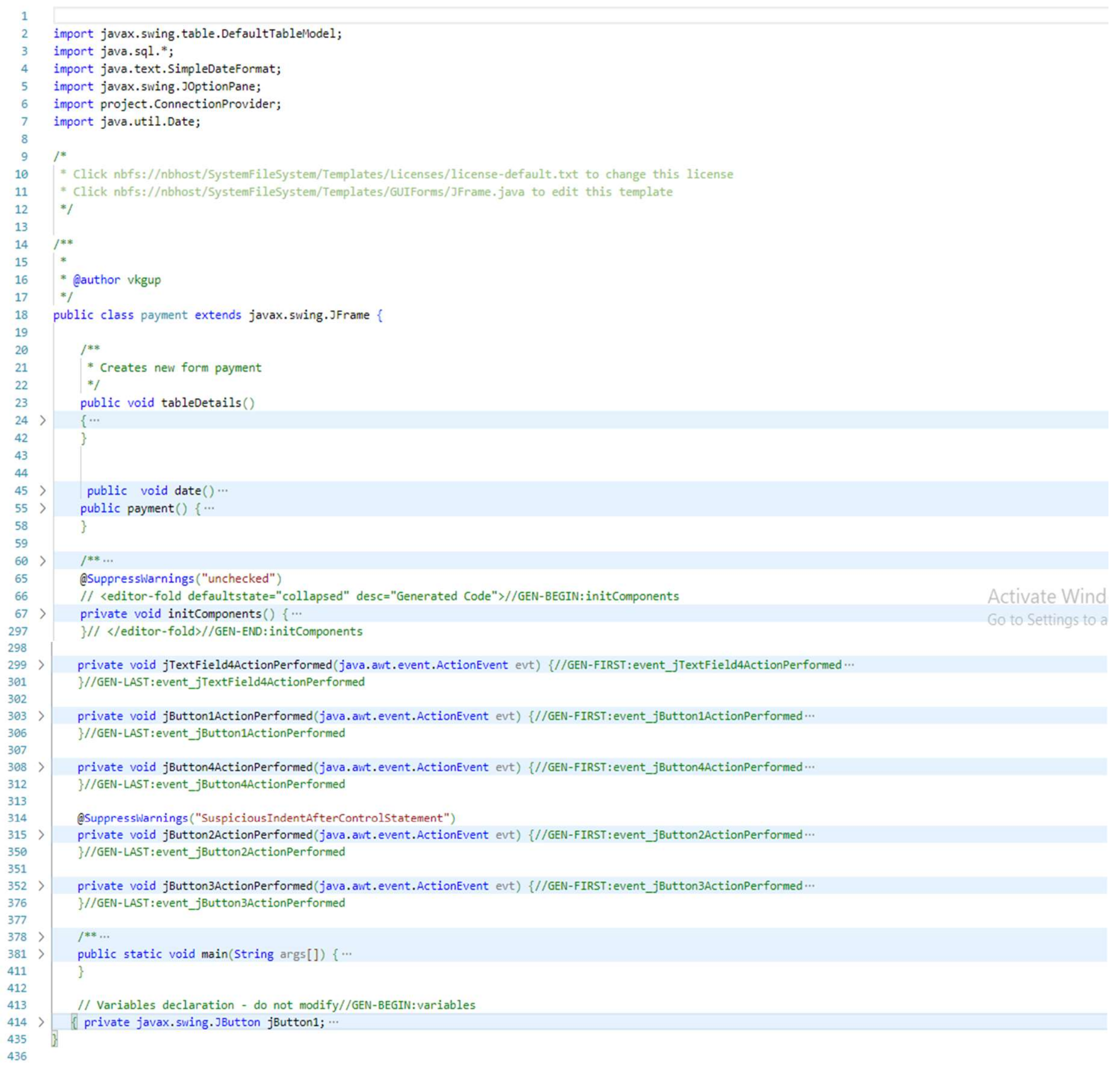
In the Easy Fitness Club Manager, administrators can efficiently track payments received from members and update payment records when payments are received. Here's a brief overview of how this functionality works:

- **Payment Tracking:** Administrators record member payments with details like amount, date, and method.
- **Updating Payments:** When payments are received, administrators mark them as received in the system.
- **Integration:** Payment details are securely stored in the database for real-time updates and accurate financial management.

This efficient process ensures that administrators can easily track and update payment records, facilitating transparent financial management in the club.

Creating a Payment class;

- Right-click on your project > New > Java Class.
- Name the class **Payment.java**.



```

import javax.swing.table.DefaultTableModel;
import java.sql.*;
import java.text.SimpleDateFormat;
import javax.swing.JOptionPane;
import project.ConnectionProvider;
import java.util.Date;

/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template
 */

/**
 *
 * @author vkgup
 */
public class payment extends javax.swing.JFrame {

    /**
     * Creates new form payment
     */
    public void tableDetails()
    {...
    }


     public  void date()...
    public payment() {...
    }

    /**...
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
    private void initComponents() {...
    }// </editor-fold>//GEN-END:initComponents

    private void jTextField4ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_jTextField4ActionPerformed...
    }//GEN-LAST:event_jTextField4ActionPerformed

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_jButton1ActionPerformed...
    }//GEN-LAST:event_jButton1ActionPerformed

    private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_jButton4ActionPerformed...
    }//GEN-LAST:event_jButton4ActionPerformed

    @SuppressWarnings("SuspiciousIndentAfterControlStatement")
    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_jButton2ActionPerformed...
    }//GEN-LAST:event_jButton2ActionPerformed

    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_jButton3ActionPerformed...
    }//GEN-LAST:event_jButton3ActionPerformed

    /**...
    public static void main(String args[]) {...
    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    { private javax.swing.JButton jButton1; ...
}

```

Fig. 5.4 Payment Code

5.6 DASHBOARD SCREEN

The dashboard page in the Easy Fitness Club Manager serves as a central hub for administrators to access key information and perform essential tasks.

- **Central Information Hub:** The dashboard displays critical data such as membership statistics, payment summaries, and upcoming events.
- **Task Management:** Administrators can efficiently manage tasks like adding new members, updating member details, and tracking payments directly from the dashboard.
- **Quick Access:** Essential features like member registration, payment tracking, and event management are easily accessible via intuitive navigation links or interactive widgets.
- **Customizable:** The dashboard may offer customization options, allowing administrators to personalize their view based on their preferences and responsibilities.
- **Real-time Updates:** Information on the dashboard is updated in real-time, ensuring administrators have access to the latest data for informed decision-making.

Overall, the dashboard provides administrators with a comprehensive and user-friendly interface to streamline club management tasks and access vital information efficiently.

Create Home class :

- Right-click on your project > New > Java Class.
- Name the class **Home.java**.

```
import javax.swing.JOptionPane;



public class home extends javax.swing.JFrame {


    public home() {
        initComponents();
    }
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-
BEGIN:initComponents
    private void initComponents() {

        jPanel1 = new javax.swing.JPanel();
        jLabel1 = new javax.swing.JLabel();
        jMenuBar1 = new javax.swing.JMenuBar();
        jMenu1 = new javax.swing.JMenu();
        jMenu2 = new javax.swing.JMenu();
        jMenu3 = new javax.swing.JMenu();
        jMenu4 = new javax.swing.JMenu();
        jMenu5 = new javax.swing.JMenu();
        jMenu6 = new javax.swing.JMenu();

        setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
        setUndecorated(true);
        jPanel1.setBackground(new java.awt.Color(0, 118, 221));

        jLabel1.setFont(new java.awt.Font("Arial", 1, 90)); // NOI18N
        jLabel1.setForeground(new java.awt.Color(255, 255, 255));
        jLabel1.setText("Welcome!");

        javax.swing.GroupLayout jPanel1Layout = new
javax.swing.GroupLayout(jPanel1);
        jPanel1.setLayout(jPanel1Layout);
        jPanel1Layout.setHorizontalGroup(
            jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addGap(448, 448, 448)
                .addComponent(jLabel1)
                .addContainerGap(497, Short.MAX_VALUE))
        );
        jPanel1Layout.setVerticalGroup(
            jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEAD
ING)
            .addGroup(jPanel1Layout.createSequentialGroup()
                .addGap(254, 254, 254)
                .addComponent(jLabel1)
                .addContainerGap(343, Short.MAX_VALUE))
        );

        jMenuBar1.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N

        jMenu1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/images/new
member.png"))); // NOI18N
        jMenu1.setText("New Member");
        jMenu1.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N
        jMenu1.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                jMenu1MouseClicked(evt);
            }
        });
        jMenuBar1.add(jMenu1);

        jMenu2.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/images/update & delete
member.png"))); // NOI18N
        jMenu2.setText("Update & Delete Member");
        jMenu2.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N
        jMenu2.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                jMenu2MouseClicked(evt);
            }
        });
        jMenuBar1.add(jMenu2);

        jMenu3.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/images/list of members.png"))); //
NOI18N
        jMenu3.setText("List Of Memebr");
        jMenu3.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N
        jMenu3.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                jMenu3MouseClicked(evt);
            }
        });
        jMenuBar1.add(jMenu3);

        jMenu4.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/images/payment.png"))); // NOI18N
        jMenu4.setText("Payment");
        jMenu4.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N
        jMenu4.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                jMenu4MouseClicked(evt);
            }
        });
        jMenuBar1.add(jMenu4);

        jMenu5.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/images/logout.png"))); // NOI18N
        jMenu5.setText("Logout");
        jMenu5.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N
        jMenu5.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                jMenu5MouseClicked(evt);
            }
        });
        jMenuBar1.add(jMenu5);

        jMenu6.setIcon(new
javax.swing.ImageIcon(getClass().getResource("/images/exit.png"))); // NOI18N
        jMenu6.setText("Exit");
        jMenu6.setFont(new java.awt.Font("Arial", 1, 14)); // NOI18N
        jMenu6.addMouseListener(new java.awt.event.MouseAdapter() {
            public void mouseClicked(java.awt.event.MouseEvent evt) {
                jMenu6MouseClicked(evt);
            }
        });
        jMenuBar1.add(jMenu6);

        setJMenuBar(jMenuBar1);

        javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setHorizontalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );
        layout.setVerticalGroup(
            layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addComponent(jPanel1, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
        );

        pack();
    }// </editor-fold>//GEN-END:initComponents
    private void jMenu5MouseClicked(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_jMenu5MouseClicked
        // TODO add your handling code here:
        int a=JOptionPane.showConfirmDialog(null,"Do you really want to
Logout","Select",JOptionPane.YES_NO_OPTION);
        if(a==0)
        {
            setVisible(false);
            new login().setVisible(true);
        }
    }//GEN-LAST:event_jMenu5MouseClicked

    private void jMenu6MouseClicked(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_jMenu6MouseClicked
        // TODO add your handling code here:
          int a=JOptionPane.showConfirmDialog(null,"Do you really want to
Exit","Select",JOptionPane.YES_NO_OPTION);
        if(a==0)
        {

            System.exit(0);
        }
    }//GEN-LAST:event_jMenu6MouseClicked

    private void jMenu1MouseClicked(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_jMenu1MouseClicked
        // TODO add your handling code here:
        new NewMember().setVisible(true);
    }//GEN-LAST:event_jMenu1MouseClicked

    private void jMenu2MouseClicked(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_jMenu2MouseClicked
        // TODO add your handling code here:
        new UpdateDeleteMember().setVisible(true);
    }//GEN-LAST:event_jMenu2MouseClicked

    private void jMenu3MouseClicked(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_jMenu3MouseClicked
        // TODO add your handling code here:
        new ListOfMembers().setVisible(true);
    }//GEN-LAST:event_jMenu3MouseClicked

    private void jMenu4MouseClicked(java.awt.event.MouseEvent evt) {//GEN-
FIRST:event_jMenu4MouseClicked
        // TODO add your handling code here:
        new payment().setVisible(true);
    }//GEN-LAST:event_jMenu4MouseClicked

    /**
     * @param args the command line arguments
     */
    public static void main(String args[]) {
        /* Set the Nimbus look and feel */
        //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code
(optional) ">
```

```
        /* If Nimbus (introduced in Java SE 6) is not available, stay with the
default look and feel.
         * For details see
http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
         */
        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(home.class.getName()).log(java.util.l
ogging.Level.SEVERE, null, ex);
        } catch (InstantiationException ex) {
            java.util.logging.Logger.getLogger(home.class.getName()).log(java.util.l
ogging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {
            java.util.logging.Logger.getLogger(home.class.getName()).log(java.util.l
ogging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
            java.util.logging.Logger.getLogger(home.class.getName()).log(java.util.l
ogging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

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

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JLabel jLabel1;
    private javax.swing.JMenu jMenu1;
    private javax.swing.JMenu jMenu2;
    private javax.swing.JMenu jMenu3;
    private javax.swing.JMenu jMenu4;
    private javax.swing.JMenu jMenu5;
    private javax.swing.JMenu jMenu6;
    private javax.swing.JMenuBar jMenuBar1;
    private javax.swing.JPanel jPanel1;
    // End of variables declaration//GEN-END:variables
}
```

5.7 USE THE DAO IN YOUR MAIN CLASS

The login page in the Easy Fitness Club Manager serves as the entry point for authorized users, typically administrators, to access the system. Here's a brief overview:

- **Authentication:** Users provide their credentials, typically a username and

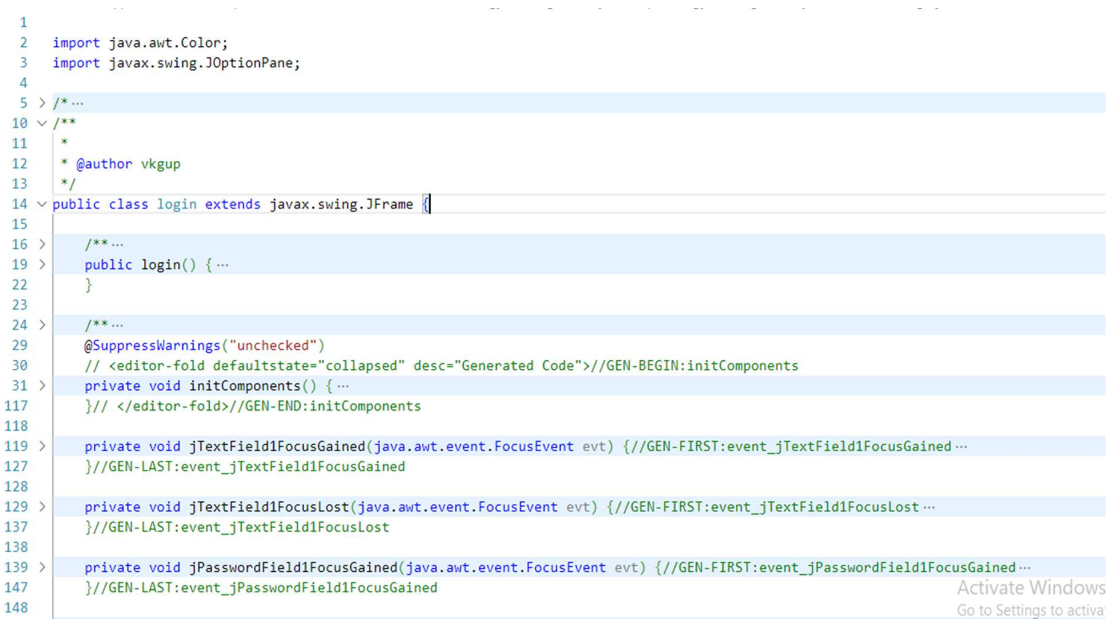
password, to authenticate and gain access to the system.
- **Security:** The login page employs robust security measures to protect against unauthorized access, such as encryption protocols and password hashing.
- **User-Friendly Interface:** The interface is designed to be intuitive and easy to use, allowing users to quickly login and access the system.
- **Error Handling:** The login page includes error handling mechanisms to alert users of incorrect login credentials or other authentication issues.

Overall, the login page ensures secure access to the Easy Fitness Club Manager, providing a seamless entry point for authorized users to manage club operations effectively.

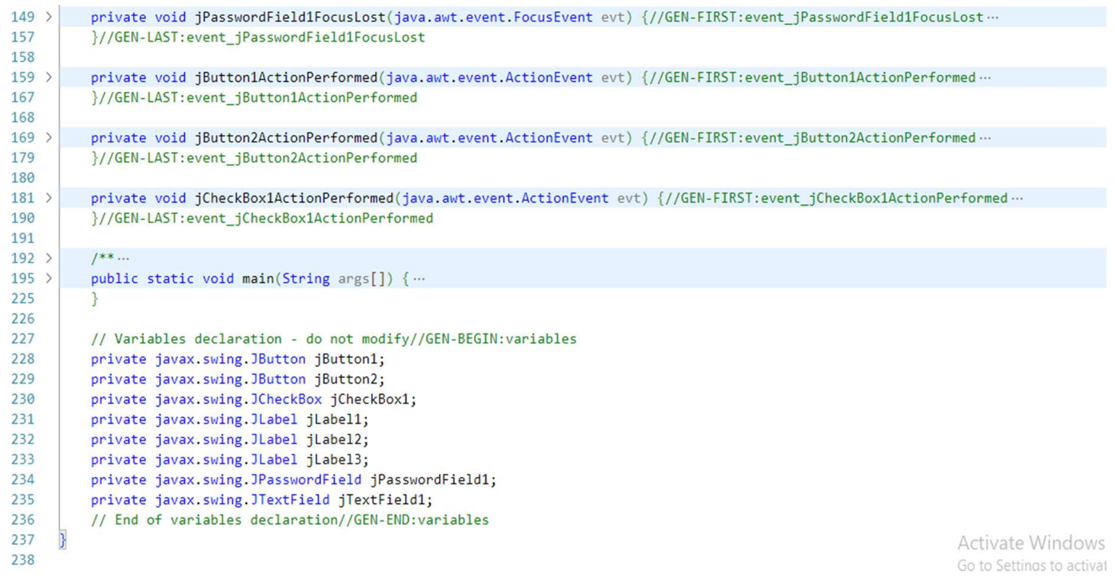
1. **Create a Main Class to Test the DAO:**
   - Right-click on your project > New > Java Main Class.
   - Name the class **Login.java**.

By following these steps, you can successfully integrate Java with MySQL through JDBC using NetBeans IDE. This setup allows you to perform CRUD operations and manage the database effectively, ensuring a robust and scalable solution for your fitness club management system.

Login.java code:-



```

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

/*...
/**
 *
 * @author vkgup
 */
public class login extends javax.swing.JFrame {

    /**...
    public login() {...
    }

    /**...
    @SuppressWarnings("unchecked")
    // <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents
    private void initComponents() {...
    }// </editor-fold>//GEN-END:initComponents

    private void jTextField1FocusGained(java.awt.event.FocusEvent evt) {//GEN-FIRST:event_jTextField1FocusGained...
    }//GEN-LAST:event_jTextField1FocusGained

    private void jTextField1FocusLost(java.awt.event.FocusEvent evt) {//GEN-FIRST:event_jTextField1FocusLost...
    }//GEN-LAST:event_jTextField1FocusLost

    private void jPasswordField1FocusGained(java.awt.event.FocusEvent evt) {//GEN-FIRST:event_jPasswordField1FocusGained...
    }//GEN-LAST:event_jPasswordField1FocusGained

```



```
    private void jPasswordField1FocusLost(java.awt.event.FocusEvent evt) {//GEN-FIRST:event_jPasswordField1FocusLost…
    }//GEN-LAST:event_jPasswordField1FocusLost

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_jButton1ActionPerformed…
    }//GEN-LAST:event_jButton1ActionPerformed

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_jButton2ActionPerformed…
    }//GEN-LAST:event_jButton2ActionPerformed

    private void jCheckBox1ActionPerformed(java.awt.event.ActionEvent evt) {//GEN-FIRST:event_jCheckBox1ActionPerformed…
    }//GEN-LAST:event_jCheckBox1ActionPerformed

    /**…
    public static void main(String args[]) {…
    }

    // Variables declaration - do not modify//GEN-BEGIN:variables
    private javax.swing.JButton jButton1;
    private javax.swing.JButton jButton2;
    private javax.swing.JCheckBox jCheckBox1;
    private javax.swing.JLabel jLabel1;
    private javax.swing.JLabel jLabel2;
    private javax.swing.JLabel jLabel3;
    private javax.swing.JPasswordField jPasswordField1;
    private javax.swing.JTextField jTextField1;
    // End of variables declaration//GEN-END:variables
}
```

Fig. 5.5 Login Code

5.8 Flow Chart Diagram:

The flow chart for the Fitness Club System project visually represents the sequence of operations and interactions between different modules. It typically starts with the user logging into the system, followed by authentication checks. Upon successful login, users can navigate to different functionalities such as adding a new member, updating member details, deleting a member, viewing the list of members, and managing payments. Each of these functionalities involves interactions between the frontend (user interface) and the backend (server and database), ensuring seamless data flow and system operation.

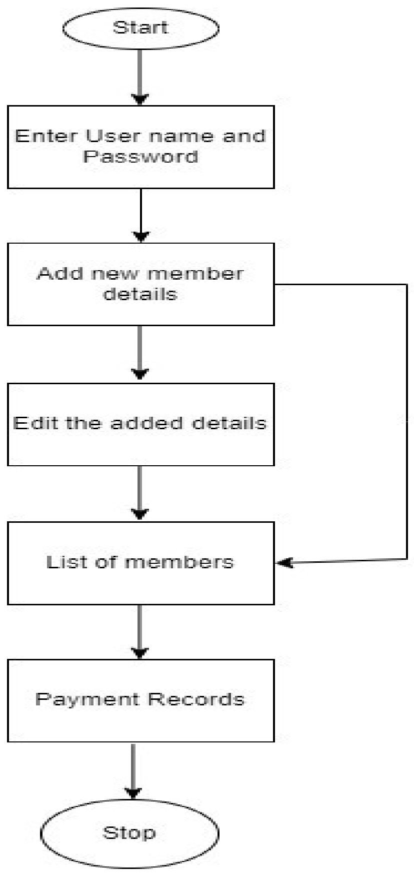


Fig. 5.6: Flow Chart Diagram

5.9 Data flow Diagram

1. Level 0 Data Flow Diagram (DFD):

The Fitness Club System provides a high-level overview of the entire system, illustrating the main processes and the flow of data between them. It depicts the interactions between external entities, such as users, and the core system processes, including user authentication, member management, and payment processing. The diagram highlights the central data repositories, such as the member and payment databases, and shows how data is inputted, processed, and outputted within the system, ensuring a clear understanding of the system's overall functionality and data flow.

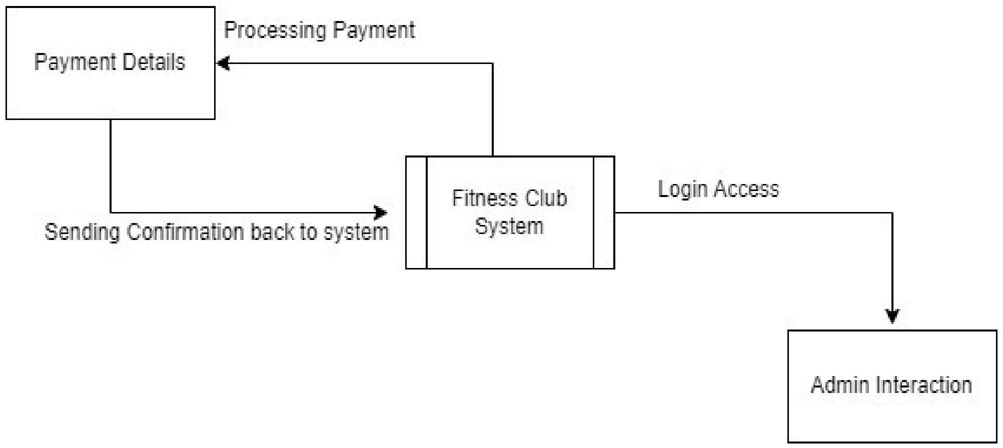


Fig. 5.7: DFD level 0 Diagram

2. **Level 1 Data Flow Diagram (DFD):**
   The Fitness Club System, the focus shifts to the detailed processes and data flows within each main system function identified in the Level 0 DFD. Each process is broken down into sub-processes, illustrating the specific tasks and data transformations involved. For example, within the "Member Management" function, sub-processes may include "Add Member," "Update Member," "Delete Member," and "List Members." Similarly, within the "Payment Processing" function, sub-processes may include "Record Payment," "View Payment History," and "Generate Payment Report." These sub-processes depict the detailed interactions between the user interface, backend server, and database, providing a comprehensive understanding of how data is processed and managed within the system.

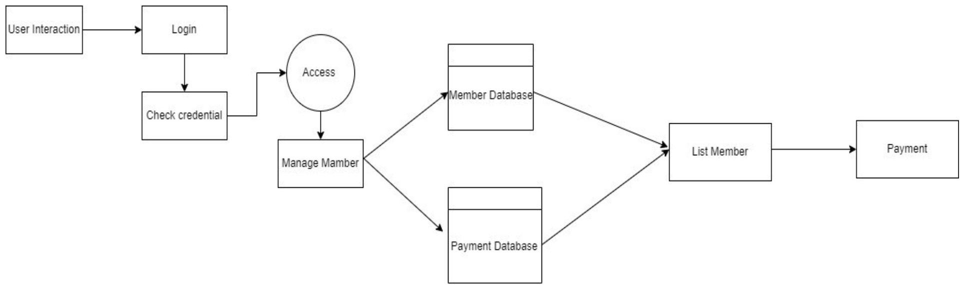


Fig. 5.8: DFD level 1 Diagram

5.10 ER- Diagram:

The Entity-Relationship (ER) Diagram for the Fitness Club System illustrates the relationships between different entities within the system's database. Entities represent the main objects or concepts stored in the database, such as "Member," "Payment," and "User." Relationships between these entities depict how they are connected or associated with each other, such as a "Member" making multiple "Payments" or a "User" having access to various system functionalities. Attributes, such as "Member ID," "Payment Amount," and "Username," provide additional details about each entity, helping to define their characteristics and properties within the database schema.

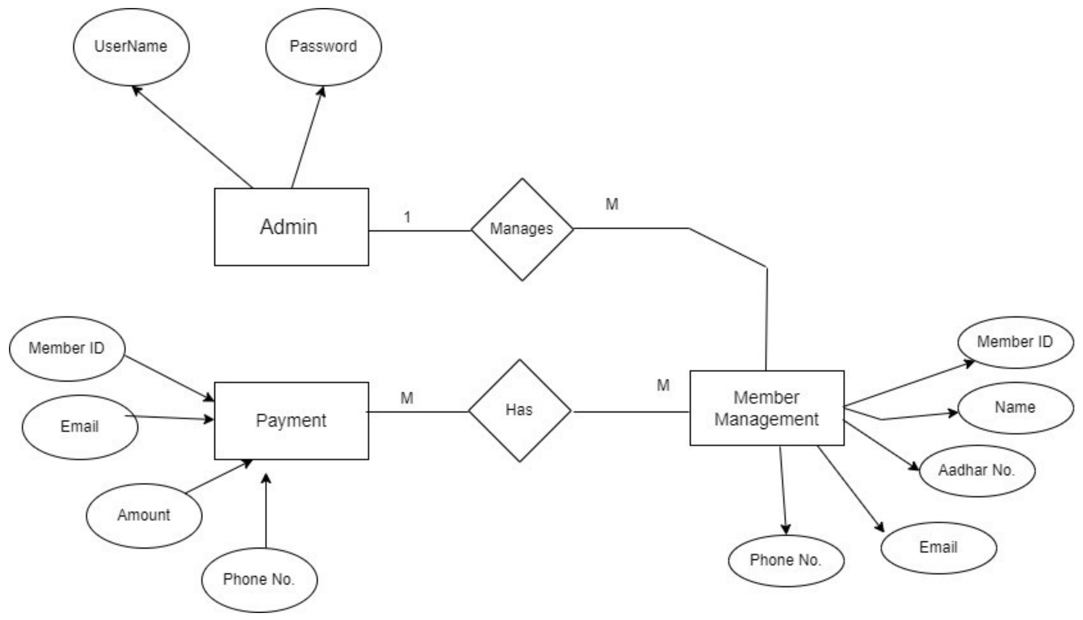


Fig. 5.9: ER- Diagram

5.11 Use Case Diagram:

The use case diagram for the Fitness Club System outlines the various interactions between actors (users) and the system, representing different scenarios or functionalities. Actors include "Admin" and "Member," who interact with the system to perform specific tasks. Use cases depict actions that actors can take within the system, such as "Login," "Add Member," "Update Member," "Delete Member," "Record Payment," and "View Payment History." Each use case describes a specific functionality or feature of the system from the perspective of the actors, providing a clear understanding of how users interact with the system to accomplish their goals.

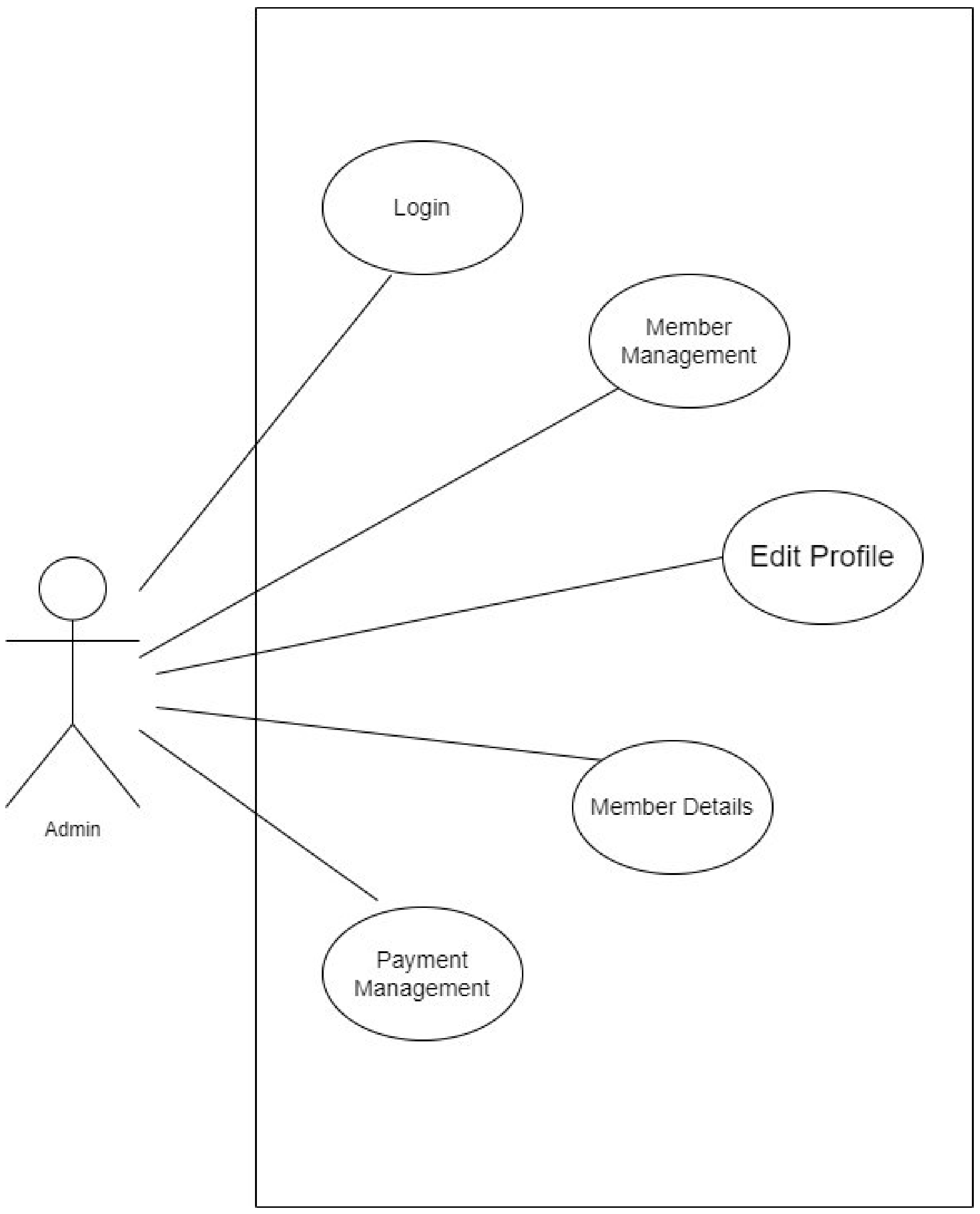


Fig. 5.10: Use Case Diagram

CHAPTER 6

IMPLEMENTATION AND RESULT

The Fitness Club System is designed to streamline fitness club operations, including member management and payment processing. The project structure is well-organized, with distinct directories for the backend and frontend, facilitating maintainability and scalability. The backend, built with Node.java and Express.java, handles server-side logic, database operations, and API routes. The frontend, developed using React, manages the client-side interface and interactions. Integration between the frontend and backend is seamless, achieved through RESTful APIs and state management with React hooks. Member and payment data are managed efficiently, with CRUD operations facilitated by API endpoints, and error handling implemented both on the frontend and backend.

6.1 PROJECT STRUCTURE AND ORGANIZATION

The project structure and organization play a crucial role in ensuring the smooth development and maintenance of the Fitness Club System. This topic focuses on establishing a well-defined structure and organization that promotes clarity, efficiency, and collaboration among team members. It encompasses the directory structure, file organization, naming conventions, and version control practices followed throughout the project.

- **Directory Structure:** The directory structure outlines the hierarchical organization of project files and folders. It provides a logical and organized framework for storing and accessing project assets, scripts, libraries, and resources. The structure is intuitive, reflecting the different components and modules of the Fitness Club System. Common directories include "backend" for server-side logic and API routes, "frontend" for client-side code and assets, "models" for database schemas, "controllers" for handling requests, and "routes" for defining API endpoints. Additionally, there are directories for utilities, configuration files, and documentation.

- **File Organization:** File organization focuses on how files are named, categorized, and grouped within the project directories. Consistent naming conventions that are descriptive and easy to understand are essential. Meaningful names are given to scripts, components, models, and other project assets to facilitate efficient file management and future maintenance. Organizing files into relevant sub-directories based on their functionality or purpose further enhances the project's structure and organization. For example, "memberRoutes.java" in the "routes" directory, "Member.java" in the "models" directory, and "MemberForm.java" in the "components" directory.

- **Naming Conventions:** Naming conventions provide a set of rules and guidelines for naming project elements consistently. This includes naming variables, functions, classes, components, and other elements. Consistent naming conventions improve code readability and facilitate collaboration and understanding among team members. Following industry-standard naming conventions or establishing a custom convention specific to the project ensures clarity. For instance, using camelCase for variables and functions, PascalCase for classes and components, and meaningful names for directories and files.
- **Version control:** Version control is crucial for managing and tracking changes made to project files over time. It ensures that different team members can work concurrently on the project without conflicts. Employing a version control system, such as Git, allows for easy collaboration, rollback to previous versions, and proper documentation of changes. Establishing a clear workflow for committing, branching, merging, and resolving conflicts is essential to maintain a reliable and organized version history. This includes creating feature branches, regular commits with descriptive messages, and pull requests for code reviews.

By emphasizing a well-defined project structure and organization, the Fitness Club System benefits from improved code maintainability, efficient file management, easy collaboration, and effective communication. It lays the foundation for a streamlined development process and sets the stage for successful project execution and delivery.

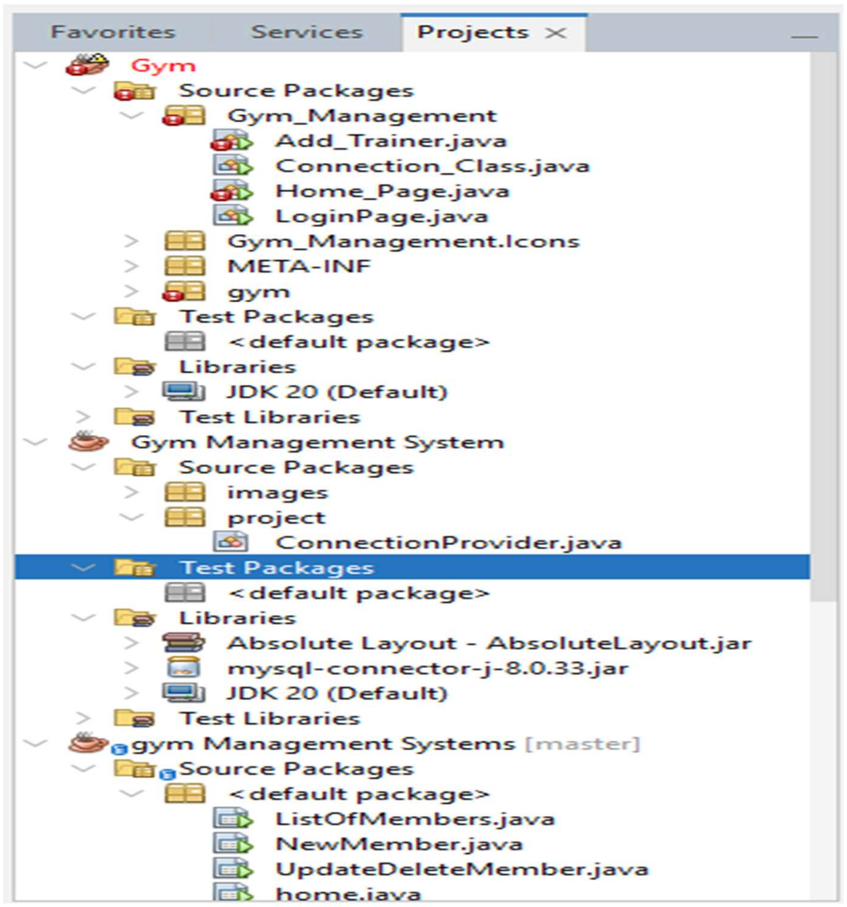

```
Favorites    Services    Projects ×
Gym
  Source Packages
    Gym_Management
      Add_Trainer.java
      Connection_Class.java
      Home_Page.java
      LoginPage.java
    Gym_Management.Icons
    META-INF
    gym
  Test Packages
    <default package>
  Libraries
    JDK 20 (Default)
  Test Libraries
Gym Management System
  Source Packages
    images
    project
      ConnectionProvider.java
  Test Packages
    <default package>
  Libraries
    Absolute Layout - AbsoluteLayout.jar
    mysql-connector-j-8.0.33.jar
    JDK 20 (Default)
  Test Libraries
gym Management Systems [master]
  Source Packages
    <default package>
      ListOfMembers.java
      NewMember.java
      UpdateDeleteMember.java
      home.java
```

Fig 6.1. Project Directory

6.2 INTEGRATION AND CONTENT MANAGEMENT

The implementation of the Fitness Club System involved the seamless integration of the backend and frontend components, along with the creation and management of content related to member and payment data. The integration process encompassed setting up the development environment, configuring necessary packages and dependencies, and establishing project settings. Within this integrated environment, the focus shifted to content creation and management, where member information, payment records, and other assets were developed and managed efficiently.

- **Integration:** The backend, built using Node.java and Express.java, handles server-side logic, database operations, and API routes. The frontend, developed using React, manages the client-side interface and interactions. The frontend communicates with the backend through RESTful APIs, ensuring seamless data flow between the two. State management is handled using React's useState and useEffect hooks, facilitating the dynamic updating of the user interface based on real-time data.

- **Content management:** Content management in the Fitness Club System involves handling member data, payment records, and other related information. CRUD operations (Create, Read, Update, Delete) are implemented for managing this data efficiently. Member information and payment details are stored in a MongoDB database, with Mongoose used for schema definition and interaction.

Techniques such as components, state management, and modular code structure are employed to efficiently manage and manipulate the content. Components like MemberForm, MemberList, and PaymentForm are designed to handle different aspects of the system, ensuring a modular and maintainable codebase. These components interact with the backend through API endpoints, facilitating efficient data management.

By integrating the backend and frontend seamlessly and managing content effectively, the Fitness Club System delivers a robust and user-friendly experience. The system is capable of handling member and payment data efficiently, ensuring accurate record-keeping and real-time updates. This integration and content management approach forms the foundation for a reliable and efficient Fitness Club System.

6.3 ACTIONS AND CLASSES

Scripts play a crucial role in the development of the Fitness Club System, enabling the implementation of various functionalities and interactions within the application. Here are some key scripts that contribute to the success of the Fitness Club System:

- **authController.java:** Handles user authentication, including login and session management.
- **memberController.java:** Manages CRUD operations for member information, allowing the addition, updating, deletion, and listing of members.
- **paymentController.java:** Manages payment records, including adding new payments and retrieving payment histories.
- **authMiddleware.java:** Ensures that routes requiring authentication are protected, allowing only authorized users to access certain endpoints.
- **memberRoutes.java:** Defines API endpoints related to member management, routing requests to the appropriate controller functions.
- **paymentRoutes.java:** Defines API endpoints related to payment management, ensuring proper routing of payment-related requests.
- **LoginForm.java:** Manages the login form on the frontend, handling user input and form submission to authenticate users.
- **MemberForm.java:** Handles the frontend form for adding and updating member information, managing user input and form submission.
- **PaymentForm.java:** Manages the frontend form for recording new payments, handling input and submission to update the payment records.

In summary, scripts empower the Fitness Club System by implementing essential functionalities, interactions, and behaviors. They handle user authentication and input, manage member and payment data, facilitate form submissions, and ensure secure access to various parts of the application. By leveraging the power of scripting, the system can provide a robust and efficient user experience, managing the application's logic and state seamlessly.

6.4 TESTING

In the Fitness Club System project, various types of tests are performed to ensure the functionality, performance, and user experience of the application. Here, we focus on two key types of tests conducted to ensure the system's reliability: Unit Testing and End-to-End Testing.

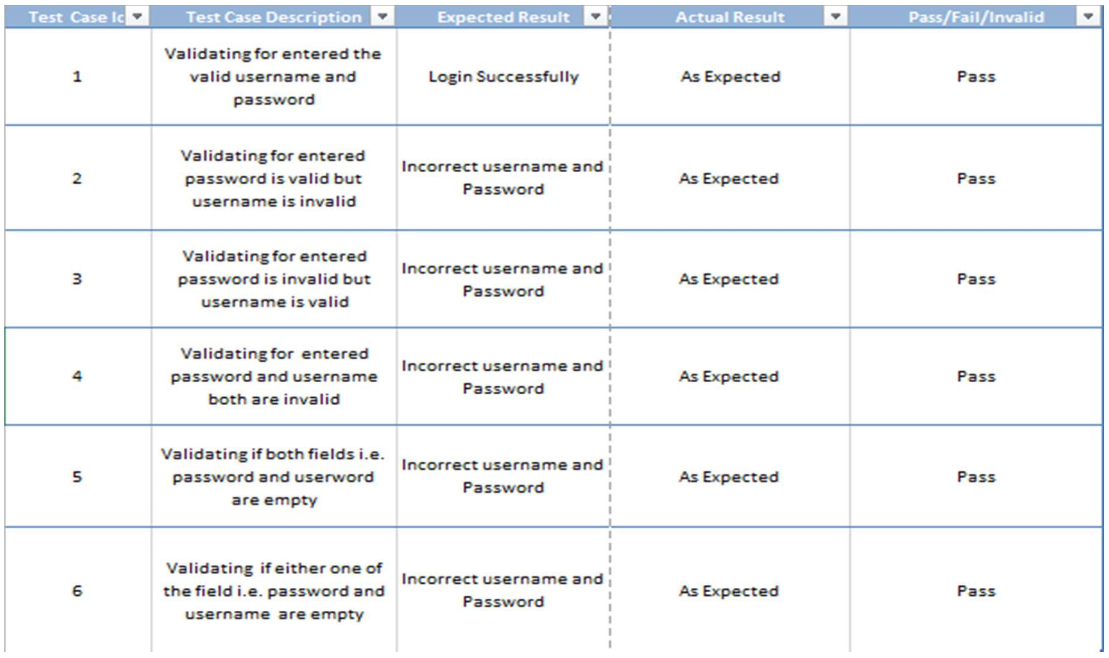
- **Unit Testing:** Unit testing is a critical component of the Fitness Club System project, focusing on evaluating the accuracy and reliability of individual components and functions. This test ensures that each part of the application performs as expected in isolation. For instance, unit tests are written for the authentication module to verify that login and session management functions correctly. Similarly, unit tests are conducted for member and payment management functions to ensure they handle CRUD operations accurately. By conducting unit tests, any potential issues or bugs within specific functions can be identified early in the development process, allowing the team to make necessary adjustments and improvements. A successful outcome of these tests ensures that each component of the system operates reliably and as intended.

- **End-to-End Testing:** End-to-end testing evaluates the entire workflow of the Fitness Club System, simulating real user interactions to ensure that the system functions correctly from start to finish. This includes testing key user interactions such as logging in, adding new members, updating member information, recording payments, and viewing member and payment lists. The application's response to these interactions is tested to verify that they trigger the expected actions or behaviors within the system. The user interface elements, including forms, buttons, and menus, are evaluated for visual clarity, readability, and consistency. The responsiveness and fluidity of the UI elements are also assessed to ensure smooth transitions and user-friendly interaction flow. The objective is to create an intuitive and seamless user experience where users can efficiently navigate through the system and access relevant information without any confusion or frustration.

By conducting Unit Testing and End-to-End Testing, any usability concerns or issues with the user interaction flow or UI design can be identified and addressed. A successful outcome of these tests ensures that users can effortlessly engage with the Fitness Club System, intuitively interact with its features, and find the UI elements visually appealing and responsive, leading to a positive and efficient user experience.

6.5 Test Cases

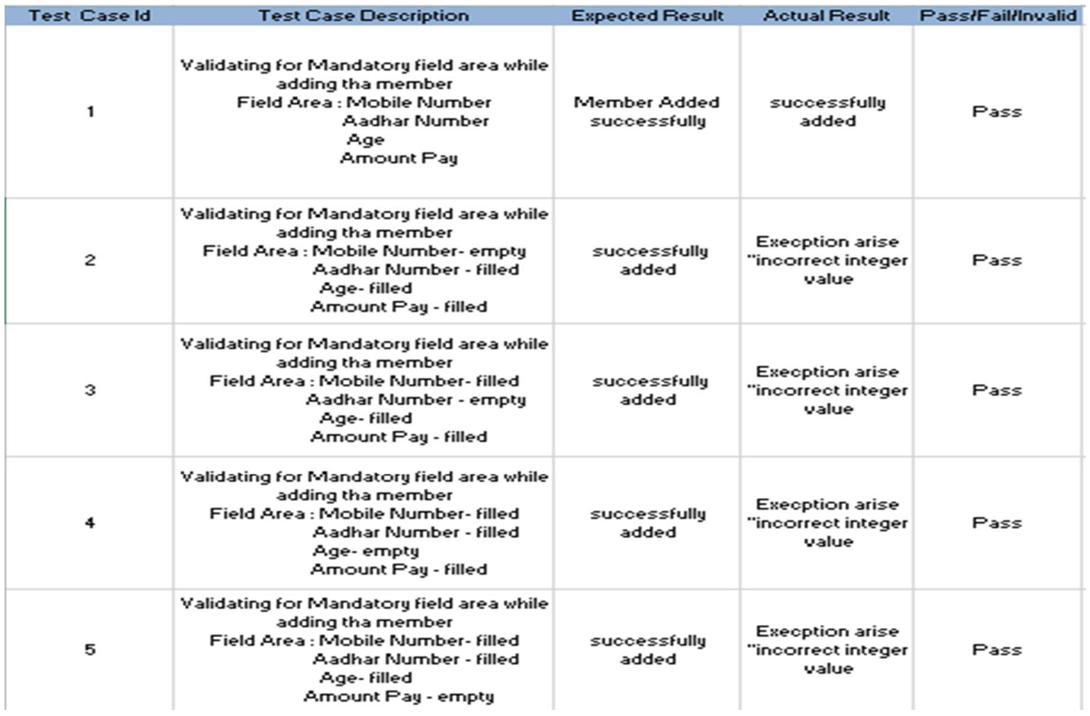
- **Test Case for Login Module –**



Test Case Ic	Test Case Description	Expected Result	Actual Result	Pass/Fail/Invalid
1	Validating for entered the valid username and password	Login Successfully	As Expected	Pass
2	Validating for entered password is valid but username is invalid	Incorrect username and Password	As Expected	Pass
3	Validating for entered password is invalid but username is valid	Incorrect username and Password	As Expected	Pass
4	Validating for entered password and username both are invalid	Incorrect username and Password	As Expected	Pass
5	Validating if both fields i.e. password and userword are empty	Incorrect username and Password	As Expected	Pass
6	Validating if either one of the field i.e. password and username are empty	Incorrect username and Password	As Expected	Pass

Table 6.1 Login

- **Test Case for Add Member Module** –



Test Case Id	Test Case Description	Expected Result	Actual Result	Pass/Fail/Invalid
1	Validating for Mandatory field area while adding tha member Field Area : Mobile Number Aadhar Number Age Amount Pay	Member Added successfully	successfully added	Pass
2	Validating for Mandatory field area while adding tha member Field Area : Mobile Number- empty Aadhar Number - filled Age- filled Amount Pay - filled	successfully added	Exeception arise "incorrect integer value	Pass
3	Validating for Mandatory field area while adding tha member Field Area : Mobile Number- filled Aadhar Number - empty Age- filled Amount Pay - filled	successfully added	Exeception arise "incorrect integer value	Pass
4	Validating for Mandatory field area while adding tha member Field Area : Mobile Number- filled Aadhar Number - filled Age- empty Amount Pay - filled	successfully added	Exeception arise "incorrect integer value	Pass
5	Validating for Mandatory field area while adding tha member Field Area : Mobile Number- filled Aadhar Number - filled Age- filled Amount Pay - empty	successfully added	Exeception arise "incorrect integer value	Pass

Table 6.2: Add Member

- **Test Case for Update And Delete Module –**

Test Case Id	Test Case Description	Expected Result	Actual Result	Pass/Fail/Invalid
1	Validating for entered the valid username and password	Login Successfully	As Expected	Pass
2	Validating for entered password is valid but username is invalid	Incorrect username and Password	As Expected	Pass
3	Validating for entered password is invalid but username is valid	Incorrect username and Password	As Expected	Pass
4	Validating for entered password and username both are invalid	Incorrect username and Password	As Expected	Pass
5	Validating if both fields i.e. password and userword are empty	Incorrect username and Password	As Expected	Pass
6	Validating if either one of the field i.e. password and username are empty	Incorrect username and Password	As Expected	Pass

Table 6.3 Update and Delete

- **Test Case for Payment Module –**

Test Case Id	Test Case Description	Expected Result	Actual Result	Pass/Fail/Invalid
1	Validating for entered the valid username and password	Login Successfully	As Expected	Pass
2	Validating for entered password is valid but username is invalid	Incorrect username and Password	As Expected	Pass
3	Validating for entered password is invalid but username is valid	Incorrect username and Password	As Expected	Pass
4	Validating for entered password and username both are invalid	Incorrect username and Password	As Expected	Pass
5	Validating if both fields i.e. password and userword are empty	Incorrect username and Password	As Expected	Pass
6	Validating if either one of the field i.e. password and username are empty	Incorrect username and Password	As Expected	Pass

Table 6.4 Payment

CHAPTER 7

PROJECT DIAGRAMS AND TEST CASES

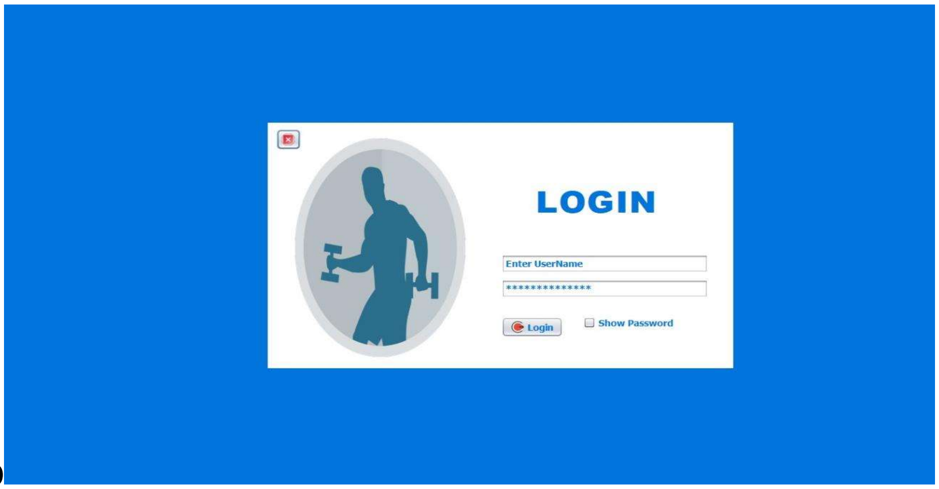
7.1 Project Screenshots:

- **Admin Login Page:** The User Login Page provides a secure gateway for existing users to access their accounts. Users are required to input their Username and password to log in successfully.

  Functionality: The Login Page module is crucial for securing the fitness club system by managing user authentication. User authentication is the process of verifying the identity of a user before granting access to the system. This module ensures that only authorized individuals can enter the system and access its features and data.

  Features:

  User Input: Prompts for username and password.
  Validation: Checks the credentials against the stored data in the database.
  Access Control: Grants or denies access based on user roles (e.g., administrator, member).
  Security: Includes password recovery options and locks accounts after multiple failed logins attempts to prevent unauthorized access.



0

Fig. 7.1 Login Page

- **HOME PAGE:** The home page serves as the entry point to the platform, presenting users with three key modules: Login, Register, and Admin. Its clean and intuitive design ensures ease of navigation for users of all levels.

  Functionality:

  The Home Page module serves as the central hub for users once they have logged into the fitness club system. It provides quick access to various features and modules, offering an overview of key information and facilitating navigation within the system. The Home Page aims to provide a user-friendly interface that enhances the user experience by displaying relevant information and options tailored to the user's role.

  Features:

  Dashboard Overview:

  Summary Panels: Displays summarized information such as total number of members, upcoming payments, recent activities, and system notifications.
  Quick Access Links: Provides links to commonly used features such as adding a new member, viewing the member list, and processing payments.

  User-Specific Information:

  Personalized Welcome Message: Greets the user with a personalized message based on their login credentials.
  User Profile Access: Offers a quick link to the user's profile where they can view and update their personal information.

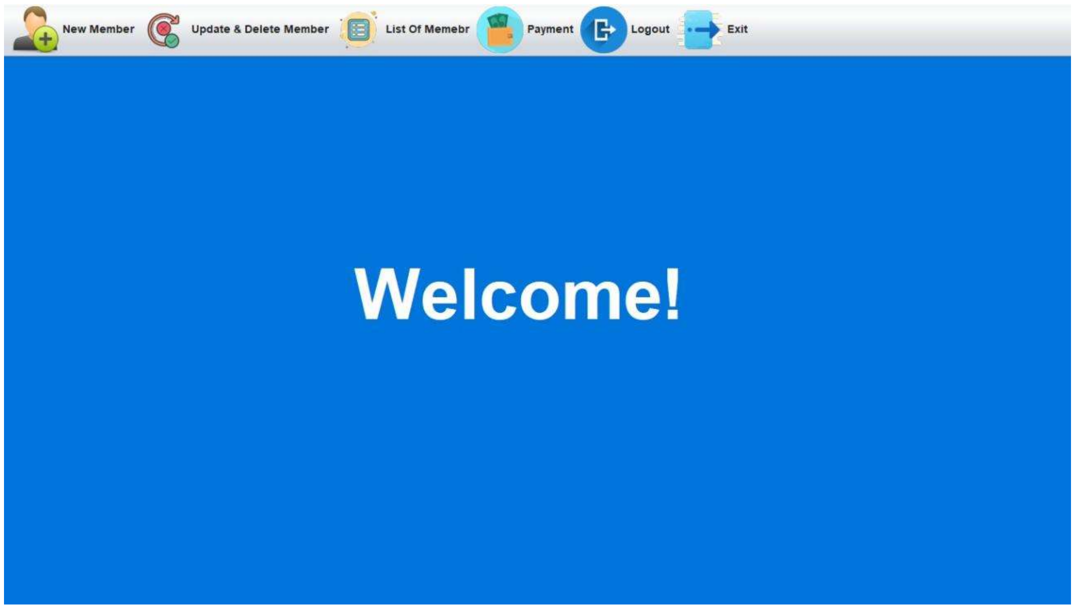


Fig. 7.2 Home Page

- **Add Member Details**: In this module, administrators can input and store detailed information about new members joining the fitness club. It includes fields such as name, contact information, membership type, and any relevant medical history, facilitating efficient management and organization of member data.

Functionality: Facilitates the addition of new member information into the system. The "Add Member Details" module is a critical component of the fitness club system, designed to handle the process of registering new members. This module streamlines the entry of comprehensive member information into the system, ensuring that all necessary details are accurately captured and stored.

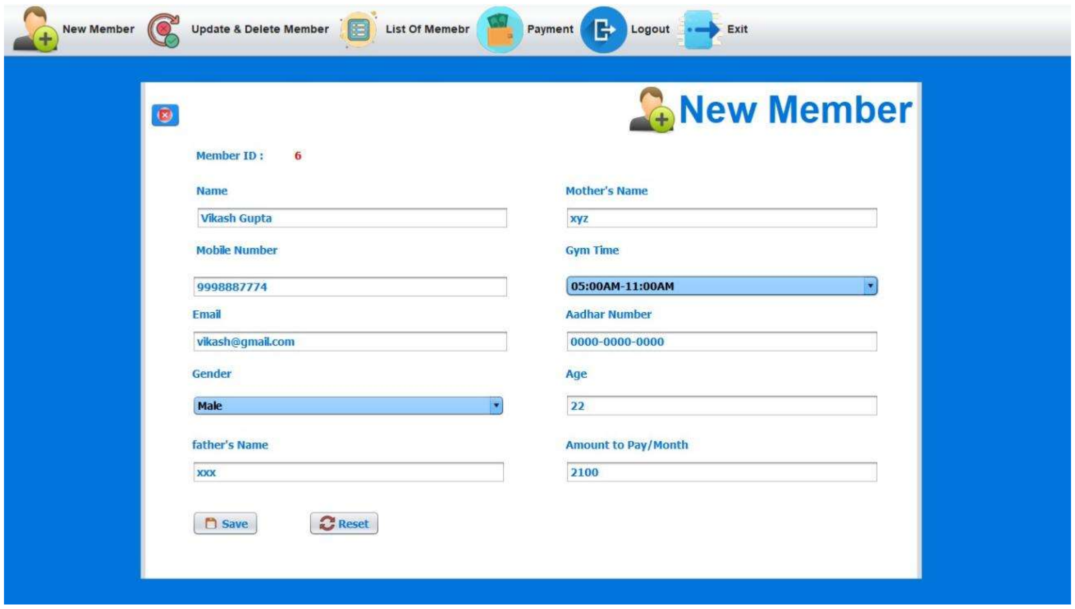
Features:

Data Entry Form: Provides fields for entering details such as name, contact information, date of birth, membership type, and medical history.

Input Validation: Ensures that the data entered is valid and complete before submission.

Database Integration: Saves the member details to the database for future reference and management.

User Feedback: Provides confirmation or error messages based on the success of the data entry process.

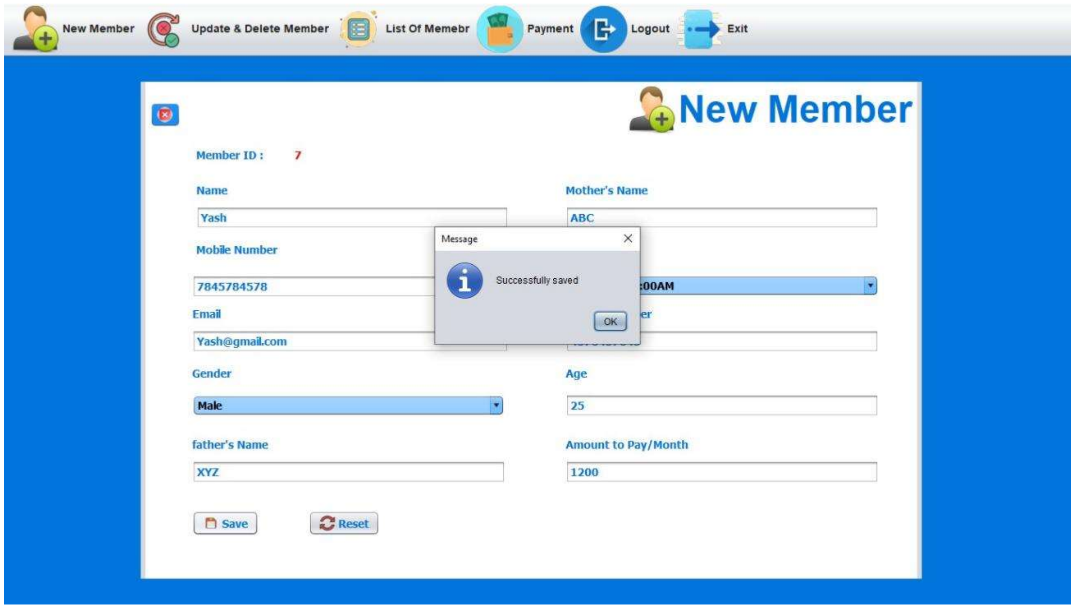


Member ID : 6

Name: Vikash Gupta	Mother's Name: xyz
Mobile Number: 9998887774	Gym Time: 05:00AM-11:00AM
Email: vikash@gmail.com	Aadhar Number: 0000-0000-0000
Gender: Male	Age: 22
father's Name: xxx	Amount to Pay/Month: 2100

Fig. 7.3 Add Member Details

Member register successfully.



Member ID : 7

Name: Yash

Mother's Name: ABC

Mobile Number: 7845784578

:00AM

Email: Yash@gmail.com

er

Gender: Male

Age: 25

father's Name: XYZ

Amount to Pay/Month: 1200

Message

Successfully saved

Fig. 7.4 Add Member Successfully

- **Update and Delete Module**: This module enables administrators to modify existing member details or remove outdated or inactive member records from the system. It ensures data accuracy and relevance by allowing for the seamless update and maintenance of member information as needed.

  Functionality: Allows administrators to modify or remove existing member records. It is a vital component of the fitness club system, designed to provide administrators with the tools necessary to maintain accurate and up-to-date member records. This module enables administrators to efficiently modify existing member information or remove member records from the system as needed.

  Features:

  Search Function: Enables searching for specific members by various criteria such as name or membership ID.

  Edit Function: Allows updating of member details, ensuring that the most current information is stored.

  Delete Function: Provides the ability to remove member records that are no longer needed.

  Confirmation Dialogs: Prompts for confirmation before making permanent changes, reducing the risk of accidental deletions.

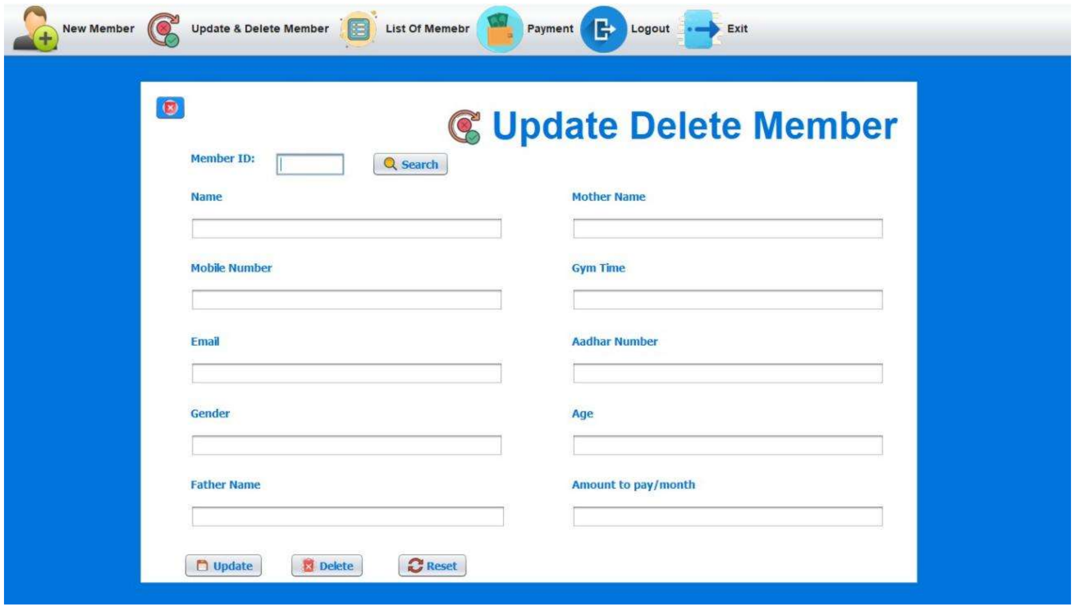
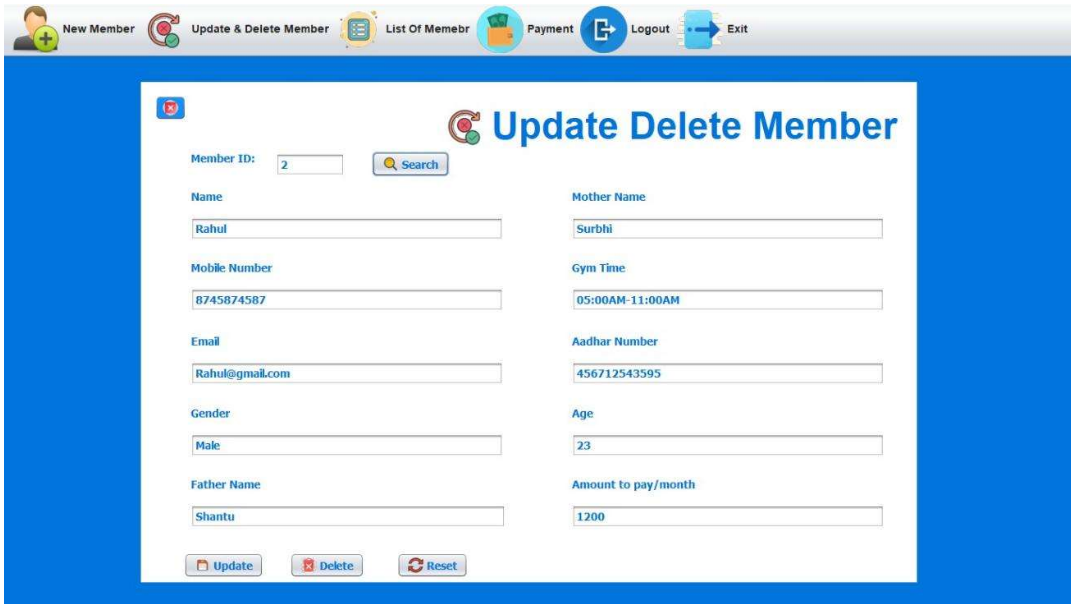


Fig. 7.5 Update Delete Member Details

Member details are edited by admin by just click update or delete button.



Member ID:	2
Name	Rahul
Mother Name	Surbhi
Mobile Number	8745874587
Gym Time	05:00AM-11:00AM
Email	Rahul@gmail.com
Aadhar Number	456712543595
Gender	Male
Age	23
Father Name	Shantu
Amount to pay/month	1200

Fig. 7.6 Edit Member Successfully

- **List of Members Module:** With this module, administrators can view a comprehensive list of all registered members within the fitness club system. It provides an overview of member demographics, membership status, and other relevant details, aiding in membership management and communication.

  Functionality: Provides an overview of all registered members and their details.
  This module is a crucial component of the fitness club system, designed to give administrators a comprehensive view of all registered members and their relevant details. This module facilitates efficient member management by presenting organized

and accessible information about the membership base.

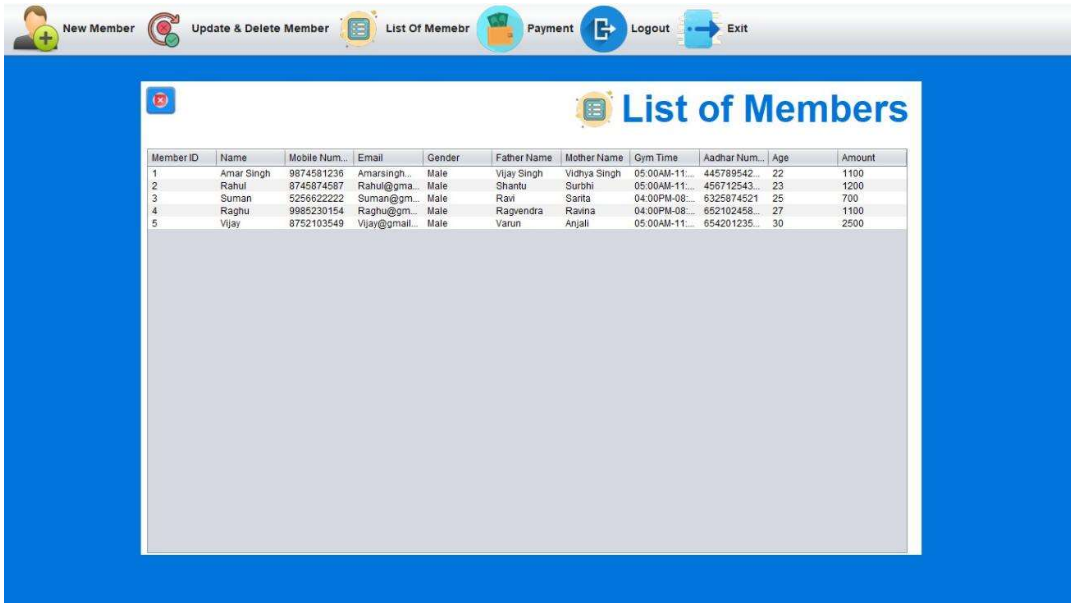
Features:

Member Listing: Displays a list of members with key information such as names, membership types, and contact details.

Search and Filter: Allows filtering and sorting of the list to quickly find specific members or groups of members.

Detail View: Offers the ability to view detailed information about a specific member.

Export Options: Enables exporting of the member list for reporting or administrative purposes.



Member ID	Name	Mobile Num...	Email	Gender	Father Name	Mother Name	Gym Time	Aadhar Num...	Age	Amount
1	Amar Singh	9874581236	Amarsingh...	Male	Vijay Singh	Vidhya Singh	05:00AM-11:...	445789542...	22	1100
2	Rahul	8745874587	Rahul@gma...	Male	Shantu	Surbhi	05:00AM-11:...	456712543...	23	1200
3	Suman	5256622222	Suman@gm...	Male	Ravi	Sarita	04:00PM-08:...	6325874521	25	700
4	Raghu	9985230154	Raghu@gm...	Male	Ragvendra	Ravina	04:00PM-08:...	652102458...	27	1100
5	Vijay	8752103549	Vijay@gmail...	Male	Varun	Anjali	05:00AM-11:...	654201235...	30	2500

Fig. 7.7 List of members

- **Payment Module:** The payment module facilitates the management of member subscriptions, fees, and transactions. It allows administrators to record and track payments made by members, generate invoices or receipts, and manage payment schedules, ensuring financial transparency and accountability within the fitness club.

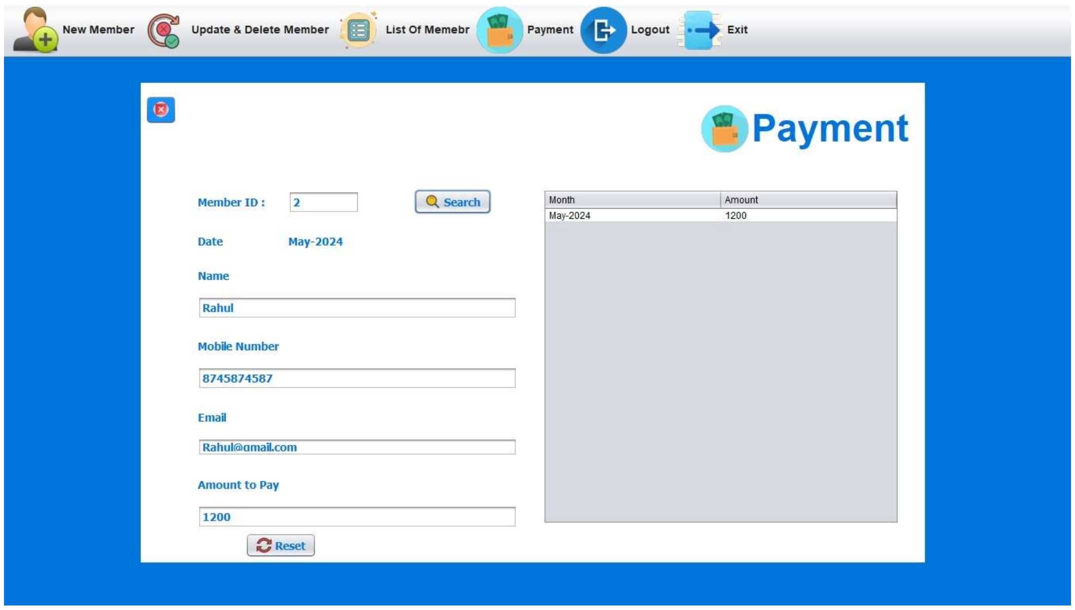
  Functionality: Manages financial transactions related to membership fees and other charges. It is a critical component of the fitness club system, designed to handle all financial aspects related to member transactions. This module ensures that membership fees and other charges are efficiently processed, recorded, and tracked, providing both administrators and members with a clear view of financial activities.

  Features:

  Payment Recording: Allows recording of payments made by members.

  Transaction History: Maintains a history of all payments, including dates, amounts, and payment methods.

Invoice Generation: Creates invoices and receipts for member payments.
Billing Management: Sends reminders for upcoming payments and manages recurring billing cycles.
Integration with Payment Gateways: Supports online payment processing through integration with various payment gateways.

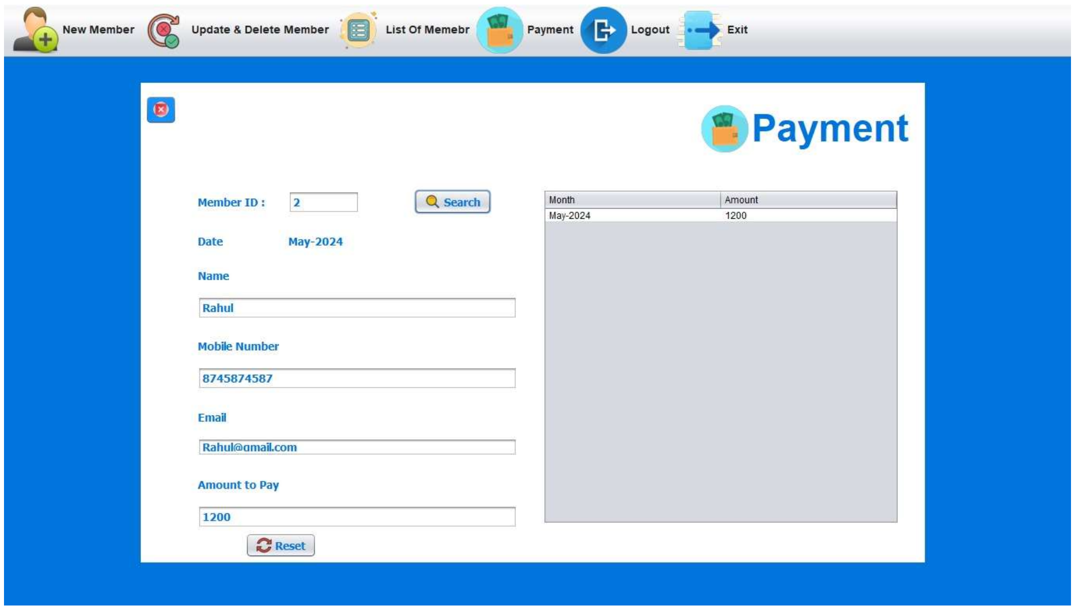


Member ID : 2
Date May-2024
Name
Rahul
Mobile Number
8745874587
Email
Rahul@gmail.com
Amount to Pay
1200

Month	Amount
May-2024	1200

Fig. 7.8 Payment module

Add payment details.

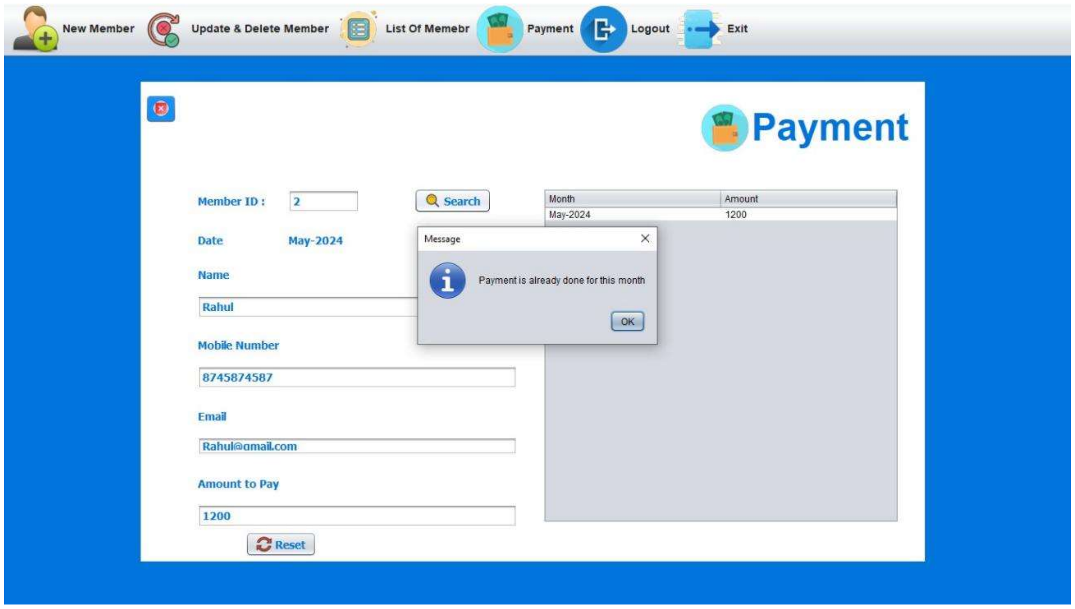


Member ID : 2
Date May-2024
Name
Rahul
Mobile Number
8745874587
Email
Rahul@gmail.com
Amount to Pay
1200

Month	Amount
May-2024	1200

Fig. 7.9 Payment Details

Payment made successfully.



Member ID : 2

Date May-2024

Name

Rahul

Mobile Number

8745874587

Email

Rahul@gmail.com

Amount to Pay

1200

Month	Amount
May-2024	1200

Message

Payment is already done for this month

Fig. 7.10 Payment Done Successfully

- **Logout Module:** This module provides a secure way for users to log out of the fitness club system, terminating their current session and ensuring data privacy and security. It helps prevent unauthorized access to sensitive information by requiring users to authenticate themselves again before accessing system functionalities.

  Functionality: Ensures secure termination of user sessions. It is an essential part of the fitness club system, designed to handle all financial transactions related to member subscriptions, membership fees, and any additional charges. This module ensures that the financial aspects of the fitness club operations are managed efficiently and accurately.

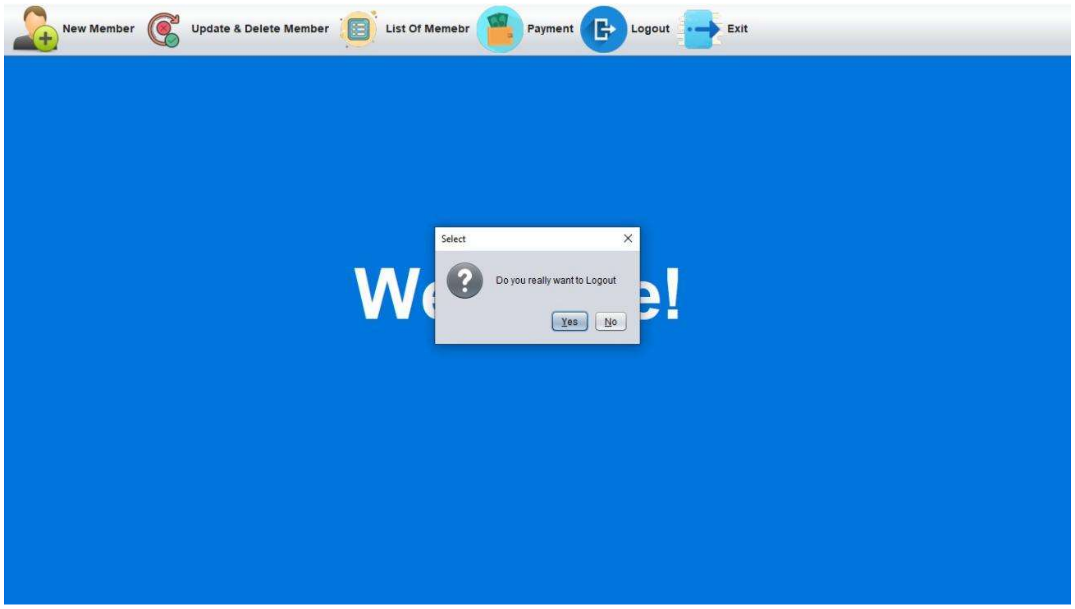
  Features:

  Session Clearing: Clears all session data upon logout to prevent unauthorized access.

  Redirection: Redirects users to the login page after they log out.

  Security: Ensures that no sensitive data remains accessible once the user has logged out.

  User Feedback: Provides confirmation that the user has successfully logged out.



Do you really want to Logout

Fig. 7.11 Logout

CHAPTER 8

CONCLUSION AND FUTURE SCOPE

Conclusion:

In conclusion, the development of the Fitness Club System represents a significant step forward in modernizing and streamlining the management of fitness clubs. By leveraging technology, this system offers a comprehensive solution for handling member information, payment processing, and administrative tasks efficiently.

Through the Fitness Club System, club administrators can easily manage member records, track payments, and ensure smooth operations. Members benefit from streamlined registration processes, transparent payment tracking, and access to up-to-date information about club activities and services.

Moreover, the system's user-friendly interface and accessibility via web and mobile devices ensure widespread adoption and ease of use for both administrators and members. By centralizing club management functions and automating repetitive tasks, the Fitness Club System enables clubs to operate more effectively and focus on providing excellent services to their members.

In essence, the Fitness Club System represents a significant advancement in the management of fitness clubs, offering a modern and efficient solution for administrators and members alike. As the fitness industry continues to evolve, initiatives like this have the potential to revolutionize the way fitness clubs operate, enhancing the overall experience for both staff and members.

Future Scope:

In brief, the future scope for the Fitness Club System includes:

1. Advancing features such as workout tracking, class scheduling, and personal training management.
2. Integrating with wearable devices and fitness tracking apps for enhanced member engagement.
3. Expanding payment options and loyalty programs for member retention.
4. Implementing data analytics and reporting tools for insights into member behavior and club performance.
5. Enhancing communication channels for better member engagement and community building.
6. Incorporating virtual reality (VR) experiences for immersive fitness sessions and training simulations.

BIBLIOGRAPHY

[1] Satyam Kumar, Kirti Verma, Prashant Singh, Anand Nayyar, "A Study on Role of Java Swing and AWT in Developing Desktop Based Applications," *International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT)*, Volume 6, Issue 5, September-October 2020, [Link](https://www.ijsrcseit.com/paper/CSEIT206531.pdf).

[2] Sanjana Ray, Shruti Sharma, Vijay Khare, "Swing and AWT for Enhanced GUI Based Java Applications," *International Journal of Computer Applications*, Volume 180, Issue 36, January 2018, [Link](https://www.ijcaonline.org/archives/volume180/number36/rah-2018-180-3616.pdf).

[3] Rajat Sharma, Akshay Sethi, "Enhancing GUI Based Java Applications Using AWT and Swing," *International Journal of Engineering Science and Computing*, Volume 7, Issue 6, June 2017, [Link](https://www.ijesc.org/upload/04895b057e93c5481b6cdddc1fa2a58e.Enhancing%20GUI%20based%20Java%20Applications%20using%20AWT%20and%20Swing.pdf).

[4] Basant Dewangan, Madhu Nigam, "Implementation of Library Management System Using Java Swing," *International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT)*, Volume 5, Issue 5, September-October 2020, [Link](https://www.ijsrcseit.com/paper/CSEIT205643.pdf).

[5] Neha Sahu, Rohit Banthia, "A Study on Development of Blood Bank Management System using Java Swing," *International Journal of Computer Applications*, Volume 173, Issue 5, February 2020, [Link](https://www.ijcaonline.org/archives/volume173/number5/sahu-2020-ijca-920013.pdf).

[6] Ishan Agrawal, Mohit Makhija, "Automated Voting System using Java Swing," *International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT)*, Volume 5, Issue 5, September-October 2020, [Link](https://www.ijsrcseit.com/paper/CSEIT205550.pdf).

APPLICATIONS

- www.google.com
- www.stackoverflow.com
- www.w3school.com
- www.udemy.com
- www.gitHub.com
- www.youtube.com
- https://www.scribd.com/