Institute of Technology of Cambodia

Department of Applied Mathematics and Statistics

Project: Gym management system

Lecturers:

Mr. Ngin Kimlong (Course)

Mr. Khean Vesal (TP)

Group :5

Students	IDs
BUN RATNATEPY	e20210320
HAYSAVIN RONGRAVIDWIN	e20211502
CHHIN VISAL	e20210742
HUON SITAHI	e20210954

2025-2026

Acknowledgment

We would like to express our sincere gratitude to **Mr. Khean Vesal**, our TP instructor, for his unwavering support, insightful guidance, and continuous encouragement throughout the development of this project.
His expert advice and constructive feedback played a crucial role in shaping the success of the **BigBoss Gym Management System**.

We are truly grateful for his mentorship, which not only enhanced our technical capabilities but also contributed to our personal and professional growth.

Abstract

The **BigBoss Gym Management System** is a comprehensive full-stack web application designed to streamline and digitize the operations of modern fitness centers. The system integrates a responsive frontend (HTML, CSS, JavaScript), a robust Node.js backend with Express for RESTful API services, a structured MySQL database for persistent data management, and a React-based admin dashboard for gym staff.

A unique component is **GymBot**, an AI-powered chatbot developed using **Java RMI (Remote Method Invocation)**, enabling real-time responses to member inquiries. Key features include member registration, trainer management, payment tracking, workout bookings, feedback collection, and smart chatbot interactions.

This report outlines the system's architecture, implementation phases, database design, frontend-backend integration, and testing results. It demonstrates how full-stack and AI-driven technologies can provide an efficient, scalable, and user-friendly solution for gym management.

Table of Contents

1. Introduction

The fitness industry is rapidly evolving, with technology playing a crucial role in how gyms manage their operations and engage with members. Traditional gym systems often suffer from manual recordkeeping, inefficient communication, and fragmented services. The need for a centralized, user-friendly, and intelligent system has never been more important.

To address these challenges, this project presents the BigBoss Gym Management System which is a comprehensive platform that digitizes and automates gym operations. The system is designed for both gym administrators and members, providing a seamless experience from registration to workout participation. Additionally, the integration of a smart chatbot enhances interactivity and support.

2. Methodology

We adopted a simplified and practical software development process:

- **Planning:** We discussed and identified the key features that our system should have and support.
- **Design:** Structured the system layout, database schema, API endpoints, and chatbot flow.
- **Development:**
    - ➢ Built the frontend first.
    - ➢ Developed backend APIs using Node.js/Express.
    - ➢ Created the admin dashboard using React.js.
    - ➢ Integrated the GymBot chatbot with Java RMI.
    - ➢ Used GitHub for collaborative development and version control.
- **Testing:** Validated functionality via manual and automated tests.
- **Integration:** Linked all components to function as a unified system.
- **Documentation:** Detailed the design, features, and usage for future development.

3. Tools and Frameworks Used

This section outlines the key technologies, tools, and frameworks used throughout the development of the BigBoss Gym Management System. Each component from the user interface to the backend and chatbot that was developed using modern, scalable technologies to ensure performance, security, and maintainability.

Frontend:

- **HTML**: Structured the content and layout of user-facing pages.
- **CSS**: Handled styling and responsive design.
- **JavaScript** : Added interactivity, form handling, and dynamic navigation.

- **Modular CSS**: Maintained consistent theming across pages, including support for dark/light mode.

Backend:

- **Node.js**: Provided the runtime environment for executing backend JavaScript code.
- **Express.js**: Framework for building RESTful APIs, managing routes, and handling middleware.
- **MySQL**: Relational database used to store structured data (members, trainers, payments, etc.).
- **phpMyAdmin**: GUI tool for managing the MySQL database during development.
- **Postman**: API testing tool used to validate endpoints and ensure correct data communication.

Backend Libraries and Middleware:

- **bcrypt**: Hashed passwords for secure user authentication.
- **cors**: Enabled cross-origin requests between frontend and backend.
- **body-parser**: Parsed incoming JSON request bodies.
- **express-validator**: Provided input validation and sanitization.

Admin Dashboard (React.js):

- **React.js**: JavaScript library for building dynamic user interfaces using components and state.
- **Axios**: Handled asynchronous HTTP requests to backend APIs.
- **Chart.js**: Visualized monthly income data through bar and line graphs.
- **React Modals and Forms**: Enabled editing of member/trainer records with dynamic forms.
- **CSV Export Utility**: Allowed exporting income reports for external use.

GymBot Chatbot (Java RMI)

- **Java SE**: Standard Java development environment for the GymBot chatbot logic.
- **Java RMI (Remote Method Invocation)**: Enabled method calls between the Java chatbot server and backend via ChatBridge.
- **ChatBridge (Node.js)**: A Node.js middleware that forwarded user input to the Java chatbot via **stdin** (standard input) and received replies through **stdout** (standard output). This mechanism enables real-time, decoupled communication between the web interface and the Java RMI-based chatbot

Development and Version Control:

- **Visual Studio Code**: Main editor for development.
- **IntelliJ IDEA**: IDE for Java chatbot development.
- **Git**: Distributed version control system used for tracking code changes.

- **GitHub**: Code hosting and collaboration platform.
- **npm**: Node.js package manager used to install project dependencies.

This diverse stack of technologies enabled the development team to build a modular, secure, and user-centric gym management platform.

4. System Overview

The BigBoss Gym Management System consists of four major components:

- Frontend (HTML, CSS, JS): Provides user-facing pages such as login, registration, contact, workout schedules, member profile, and chatbot interaction.
- Backend (Node.js + Express): Acts as the core API service, handling all requests related to members, trainers, payments, bookings, and chatbot communication.
- Database (MySQL): Stores persistent data for users, trainers, payments, feedback, and workout activities.
- Admin Dashboard (React.js): Allows gym staff to manage member records, track income, view feedback, and monitor trainer performance.
- GymBot Chatbot (Java RMI): A Java-based intelligent assistant that replies to user inquiries and supports gym-related requests.

Each module is developed with modularity and scalability in mind, ensuring that new features can be easily integrated.

5. Database Design

The system utilizes a structured relational database using MySQL, As shown in Figure 1, the database schema includes six core tables, each serving a specific role in supporting gym operations:

- **Members** : Stores user details including full name, contact information, membership type, and login credentials.
- **Trainers** : Holds information about personal trainers such as names, specializations, and available working schedules.
- **Payments** : Records all payment transactions, including the member ID, amount paid, payment method, and transaction date.
- **Feedback**: Captures feedback submitted by members, including ratings and comments related to trainers or services.
- **Bookings** : Logs workout session bookings, linking members to available workout slots or trainer appointments.
- **Contacts**: Stores inquiry or support messages submitted by users via the contact form.

All tables are normalized to reduce data redundancy and improve consistency. Foreign key constraints are used to maintain relationships between entities (e.g., linking bookings and payments to specific members and trainers). This structure ensures data integrity and allows for scalable system expansion.

6. Backend Architecture

The backend is implemented using Node.js and Express.js framework. It provides RESTful APIs for CRUD operations, form submissions, and chatbot request routing. Key API endpoints include:

- POST /api/members : Register new members
- POST /api/login : Authenticate user login
- GET /api/trainers : Retrieve trainer list
- POST /api/feedback: Submit trainer or service feedback
- GET /api/payments: Display payment history
- POST /api/chatbot: Communicate with the Java RMI chatbot

Security measures include password hashing with bcrypt and CORS-enabled headers for frontend-backend communication. Data validation ensures inputs are sanitized before reaching the database.

7. Frontend Development

The frontend is designed using HTML, CSS, and JavaScript to ensure responsiveness and interactivity. Each page serves a specific role:

- index.html: Landing page introducing the gym
- login.html, logout.html: Member authentication
- home.html: Central dashboard with feature navigation
- account.html : Member profile and account updates
- workouts.html: Embedded workout videos and booking form
- trainers.html : Trainer information with feedback option
- contact.html: Inquiry and support contact form
- chatbot.html: Integration with GymBot RMI assistant

Styles are maintained via modular CSS files, ensuring consistent layout and dark/light themes. JavaScript enhances form handling, navigation control, and booking logic.

8. Admin Dashboard Features

The admin panel, built with React.js, provides gym staff with secure and efficient access to manage and monitor gym operations. Key features include:

- **Dashboard Summary Cards:** Display key statistics such as the **total number of members**, **total number of trainers**, and the **overall income**.
- **Member Management:** View, search, and filter member lists and profiles.

- **Trainer Management:** Add, update, or remove trainer records.
- **Income Analytics:**
    - A **Membership Monthly Income** section features fully interactive charts, including **bar** and **line graphs**, to visualize income trends.
    - **Filters** allow staff to sort income data by **year** and **membership type**.
    - A **CSV export** function enables staff to download income reports.
    - The **total monthly income** is also displayed prominently for quick insights.
- **Member Editing:** Update member records through dynamic **modals**.
- **Feedback Monitoring:** Review user-submitted feedback and assess trainer performance.

All components are connected to the backend through API calls and update in real time using React state management and lifecycle methods, ensuring a smooth and responsive admin experience.

9. GymBot Chatbot with Java RMI

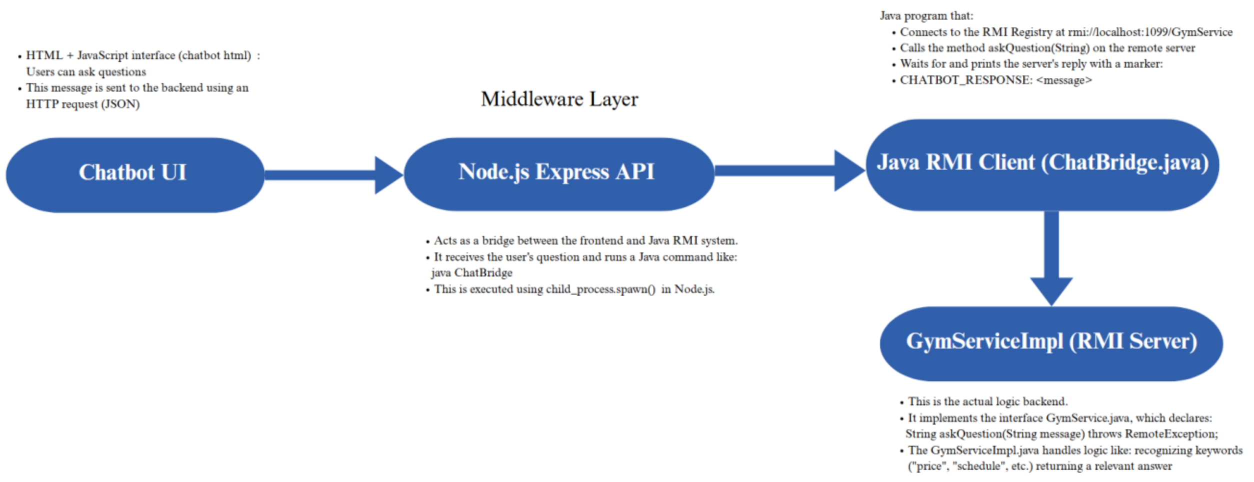
GymBot is a lightweight virtual assistant built using Java RMI (Remote Method Invocation). It allows members to ask gym-related questions and receive real-time answers without staff intervention. Components include:

- GymService (Interface): Defines available methods like getResponse(String input)
- GymServiceImpl (Implementation): Contains logic to process input and return replies
- GymServer: Registers the RMI service
- ChatBridge: Connects Node.js backend to RMI and transfers input/output via stdin/stdout

9.1 GymBot Architecture

This architecture decouples the chatbot from the main server, offering flexibility in deployment. The GymBot system comprises three primary components:

- **Frontend:** HTML, CSS, and JavaScript for user interaction.
- **Backend:** Node.js REST API to handle HTTP requests and interactions.
- **Java RMI Layer:** Specifically designed to process frequently asked questions (FAQs). ChatBridge is included as a critical intermediary because Node.js lacks native support for Java RMI communication. Since Node.js cannot directly invoke Java RMI methods, ChatBridge acts as a translation layer, converting frontend queries into RMI method calls.

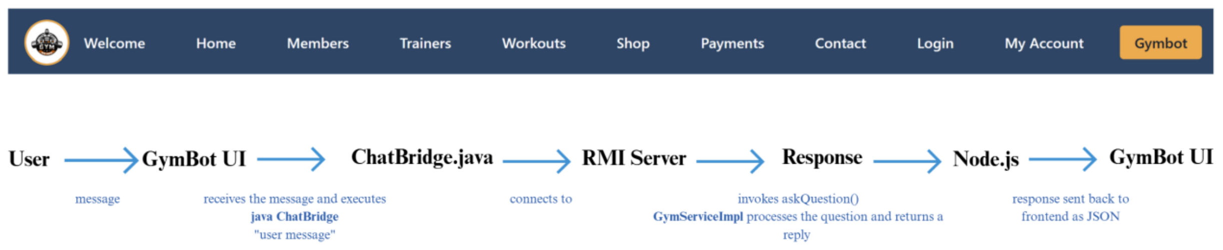

• HTML + JavaScript interface (chatbot html) : Users can ask questions
• This message is sent to the backend using an HTTP request (JSON)
Middleware Layer
Java program that:
• Connects to the RMI Registry at rmi://localhost:1099/GymService
• Calls the method askQuestion(String) on the remote server
• Waits for and prints the server's reply with a marker:
• CHATBOT_RESPONSE: <message>
Chatbot UI
Node.js Express API
Java RMI Client (ChatBridge.java)
• Acts as a bridge between the frontend and Java RMI system.
• It receives the user's question and runs a Java command like: java ChatBridge
• This is executed using child_process.spawn() in Node.js.
GymServiceImpl (RMI Server)
• This is the actual logic backend.
• It implements the interface GymService.java, which declares: String askQuestion(String message) throws RemoteException;
• The GymServiceImpl.java handles logic like: recognizing keywords ("price", "schedule", etc.) returning a relevant answer

9.2 GymBot Message Communication Flow

- User submits an FAQ query via the frontend interface.
- JavaScript captures the query and forwards it via HTTP to the Node.js backend.
- Node.js invokes the Java process (ChatBridge) using RMI.
- ChatBridge (RMI client) communicates the query to GymServer. This intermediate step is necessary because Node.js cannot directly invoke Java RMI methods without a dedicated Java intermediary.
- GymServiceImpl processes the query by matching predefined FAQs and returns the appropriate answer.
- The response is relayed back through Node.js to the frontend, displaying it to the user. The response is sent back to the frontend as a JSON object, ensuring structured data transfer and easy parsing by JavaScript.

This enables smooth and real-time interaction between the user interface and the server logic.

GymBot Full Message WorkFlow


Welcome
Home
Members
Trainers
Workouts
Shop
Payments
Contact
Login
My Account
Gymbot
User
GymBot UI
ChatBridge.java
RMI Server
Response
Node.js
GymBot UI
message
receives the message and executes java ChatBridge "user message"
connects to
invokes askQuestion() GymServiceImpl processes the question and returns a reply
response sent back to frontend as JSON

This workflow represents the actual sequence of communication between the user-facing GymBot UI, the Node.js backend, and the Java RMI server, including the ChatBridge as a vital adapter for remote invocation.

10. Integration and Testing

The system was tested through unit and integration tests:

- All API endpoints were verified using Postman.
- Admin dashboard was tested using manual UI scenarios.
- Chatbot RMI was confirmed via both command-line and UI interface.
- Frontend logic was verified across all pages for responsiveness and error handling.

Users were also surveyed for usability, with positive feedback on system convenience and feature richness.

11. Future Enhancements

To improve the system's functionality, security, and user experience, the following enhancements are proposed for future development:

- **Implement Email Confirmation After Login**
  Add an email verification feature to enhance user authentication and ensure account security.
- **Extend GymBot with Natural Language Processing (NLP)**
  Upgrade the chatbot to support NLP techniques for understanding user intent, enabling more flexible and conversational responses.
- **Deploy the System to Cloud Infrastructure**
  Host the application on scalable cloud platforms such as **Amazon Web Services (AWS)** for better availability, performance, and remote access.
- **Add Push Notifications and Booking Reminders**
  Notify users of upcoming workouts, payments, or system updates to increase engagement and reduce missed appointments.
- **Enhance UI Styling**
  Integrate modern CSS frameworks such as **TailwindCSS** or **Bootstrap 5** to further improve responsiveness, visual consistency, and component reusability.

12. Project Reflections

During the development process, our team faced challenges such as integrating Java RMI with Node.js, which we resolved using a custom ChatBridge component. This experience improved our understanding of cross-language communication and modular architecture. It also taught us the importance of modular design, debugging across platforms, and collaborative problem-solving in real-world software projects.

13. Conclusion

The BigBoss Gym Management System successfully demonstrates the application of full-stack technologies in solving real-world operational issues in fitness centers. From robust backend logic to interactive frontend design and intelligent chatbot support, the system is both functional and extensible. It lays the groundwork for a future-ready digital gym experience that benefits both administrators and members alike.

14. References

Academic and Conceptual References

1. Ian Sommerville, Software Engineering, 10th Edition, Pearson Education, 2015.
2. Oracle, "Java RMI Tutorial," https://docs.oracle.com/javase/tutorial/rmi/
3. Node.js Documentation: https://nodejs.org/docs/latest/api/documentation.html#json-output
4. React.js: https://react.dev/
5. Express.js: https://expressjs.com/
6. MySQL: https://dev.mysql.com/doc/
7. Chart.js: https://www.chartjs.org/

15. Appendices

Figure 1 : Database Schema

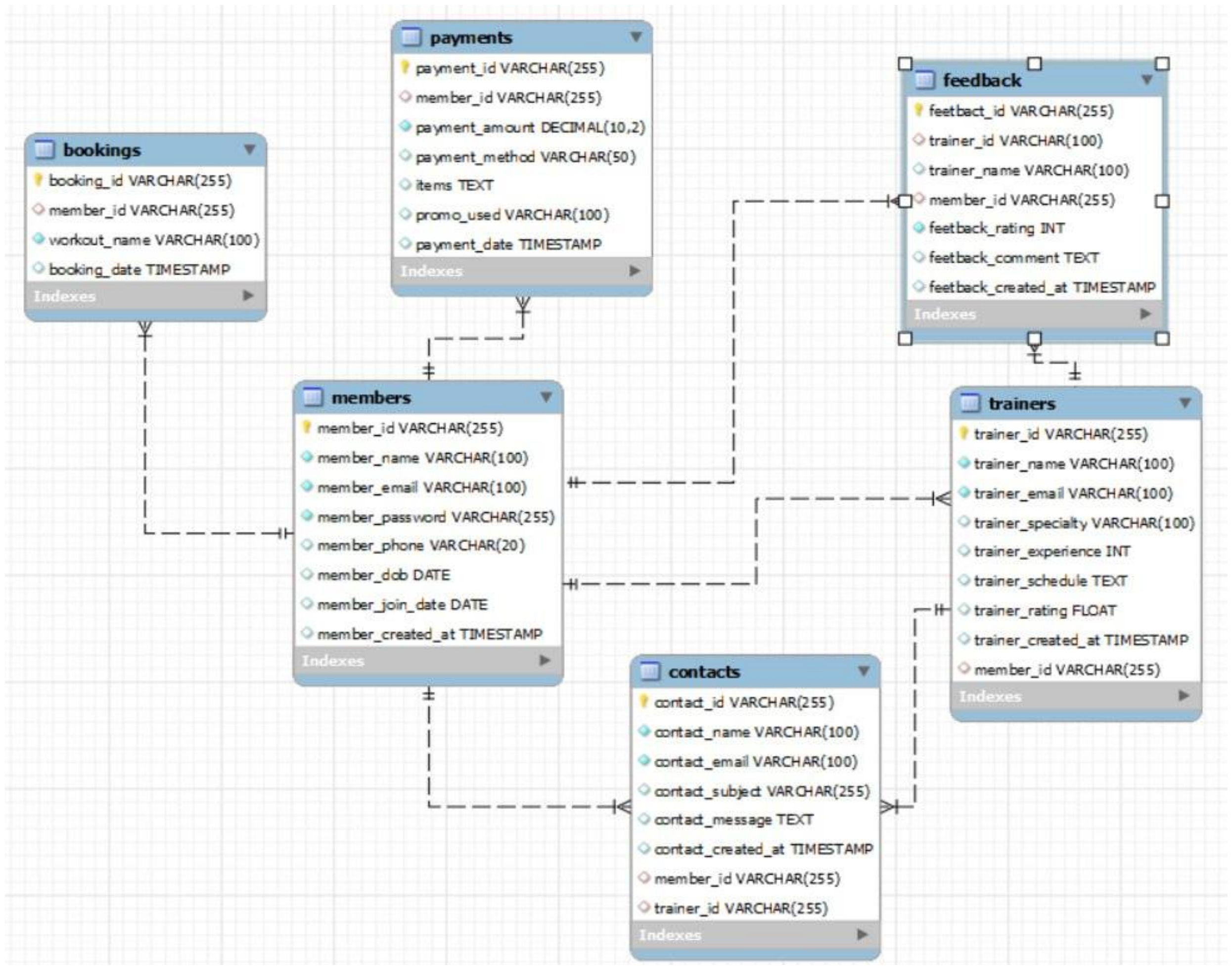
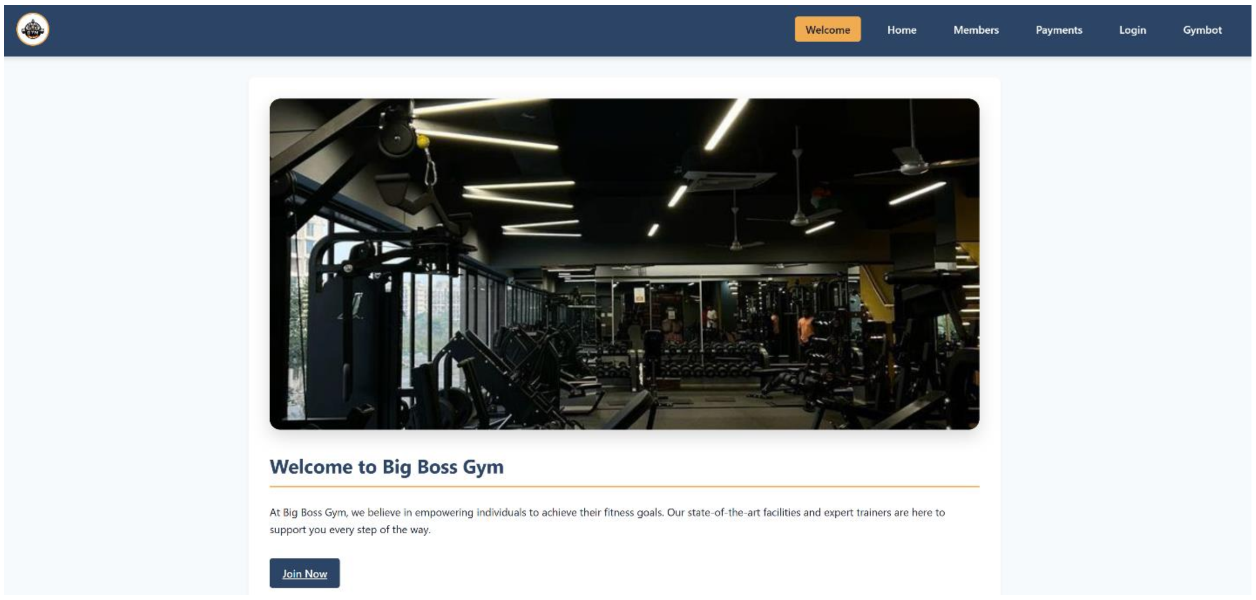


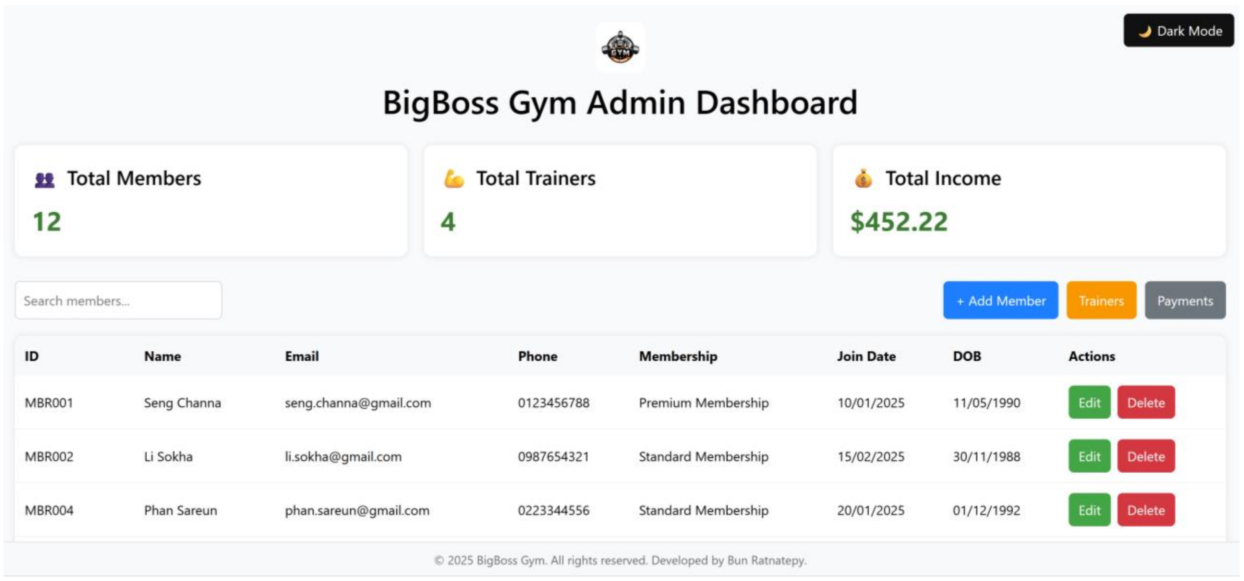
Figure 2 : Web UI



Welcome to Big Boss Gym

At Big Boss Gym, we believe in empowering individuals to achieve their fitness goals. Our state-of-the-art facilities and expert trainers are here to support you every step of the way.

Figure 3 : Admin Dashboard



BigBoss Gym Admin Dashboard

Total Members 12

Total Trainers 4

Total Income $452.22

ID	Name	Email	Phone	Membership	Join Date	DOB	Actions
MBR001	Seng Channa	seng.channa@gmail.com	0123456788	Premium Membership	10/01/2025	11/05/1990	Edit Delete
MBR002	Li Sokha	li.sokha@gmail.com	0987654321	Standard Membership	15/02/2025	30/11/1988	Edit Delete
MBR004	Phan Sareun	phan.sareun@gmail.com	0223344556	Standard Membership	20/01/2025	01/12/1992	Edit Delete

© 2025 BigBoss Gym. All rights reserved. Developed by Bun Ratnatepy.

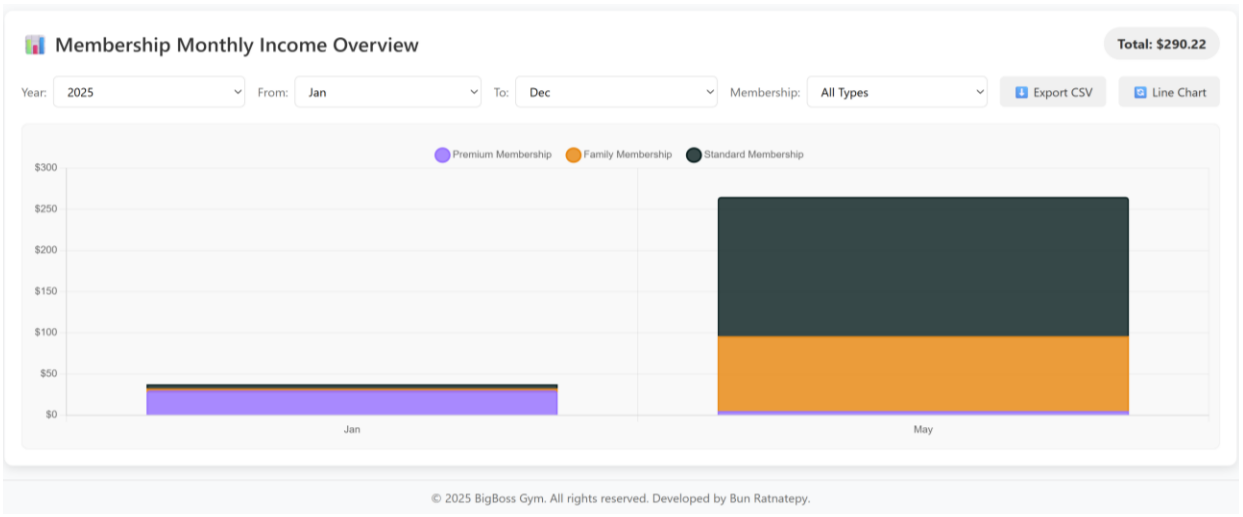
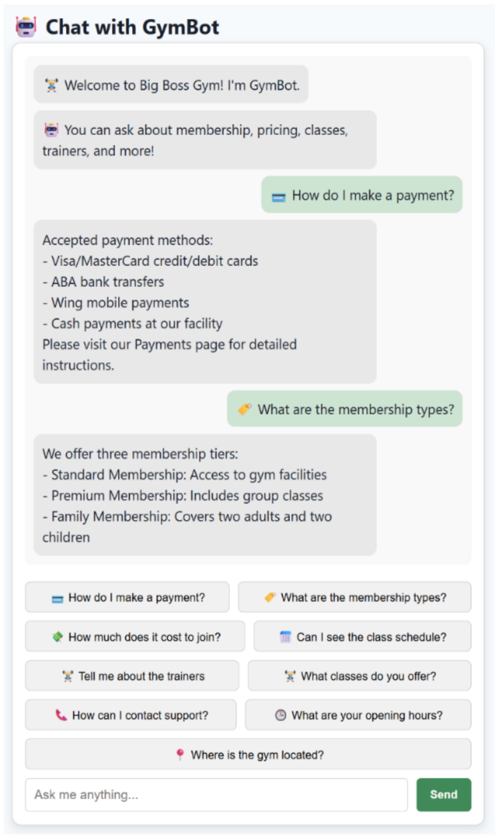


Figure 4 : Chatbot UI



Chat with GymBot

Welcome to Big Boss Gym! I'm GymBot.

You can ask about membership, pricing, classes, trainers, and more!

How do I make a payment?

Accepted payment methods:
- Visa/MasterCard credit/debit cards
- ABA bank transfers
- Wing mobile payments
- Cash payments at our facility

Please visit our Payments page for detailed instructions.

What are the membership types?

We offer three membership tiers:
- Standard Membership: Access to gym facilities
- Premium Membership: Includes group classes
- Family Membership: Covers two adults and two children

How do I make a payment?
What are the membership types?
How much does it cost to join?
Can I see the class schedule?
Tell me about the trainers
What classes do you offer?
How can I contact support?
What are your opening hours?
Where is the gym located?
Ask me anything...
Send

The full source code of the Gym Management System is available on GitHub:
https://github.com/Ratnatepy/Gym_Management_System