

**MIS573 System Design and Development**

Under the guidance of Prof. Bengisu Tulu



**Client: Riyasa Studio**

*Prepared by:* Noopura Vaidya, Ajinkya Ghorpade, Vikrant Prayag, Sakshi Rakshe

Table of Contents

[**1. Executive Summary** 5](#_Toc196833947)

[**2. System Request** 6](#_Toc196833948)

[**3. Revised Feasibility Analysis** 7](#_Toc196833949)

[3.1 Introduction 7](#_Toc196833950)

[3.2 Technical Feasibility 7](#_Toc196833951)

[3.2.1 Familiarity with the Application & Technology 7](#_Toc196833952)

[3.2.2 Key Technical Considerations 7](#_Toc196833953)

[3.2.3 System Integration & Compatibility 8](#_Toc196833954)

[3.2.4 Conclusion 8](#_Toc196833955)

[3.3 Economic Feasibility 8](#_Toc196833956)

[3.3.1 Cost-Benefit Analysis (CBA) 8](#_Toc196833957)

[Assumptions: 8](#_Toc196833958)

[Financial Projections: 9](#_Toc196833959)

[3.4 Organizational Feasibility 10](#_Toc196833960)

[3.4.1 Strategic Alignment 10](#_Toc196833961)

[3.4.2 Stakeholder Support 10](#_Toc196833962)

[3.4.3 Resource Availability 10](#_Toc196833963)

[3.5 Overall Feasibility Conclusion 10](#_Toc196833964)

[**4. Use Cases** 10](#_Toc196833965)

[Fig.4.1 Book a Fitness Class use case — casual format. 11](#_Toc196833966)

[Fig.4.2 View Trainer Details use case — casual format. 11](#_Toc196833967)

[Fig.4.3 Request a Room Rental use case — casual format. 11](#_Toc196833968)

[Fig.4.4 Manage Clients and Memberships use case — casual format. 12](#_Toc196833969)

[Fig.4.5 Approve or Reject Room Rental use case — casual format. 12](#_Toc196833970)

[Fig.4.6 View Dashboard Analytics use case — casual format 13](#_Toc196833971)

[Fig.4.7 Manage Class Schedule use case — casual format 13](#_Toc196833972)

[Fig.3.8 Cancel Scheduled Class use case — casual format 14](#_Toc196833973)

[4.2 Revised Functional Requirements Based on Use Cases 14](#_Toc196833974)

[4.2.1 Client Functional Requirements: 14](#_Toc196833975)

[4.2.2 Admin Functional Requirements: 14](#_Toc196833976)

[4.2.3 Trainer Functional Requirements: 14](#_Toc196833977)

[**4.3 Process and Data Models** 15](#_Toc196833978)

[4.3.1 AS-IS Process Model 15](#_Toc196833979)

[4.3.2 Domain Model 16](#_Toc196833980)

[4.3.3 Context Diagram (Top Level DFD) 16](#_Toc196833981)

[4.3.4 Level 0 DFD 17](#_Toc196833982)

[**5. System Architecture Design** 17](#_Toc196833983)

[5.1 System Architecture Overview 17](#_Toc196833984)

[5.2 Architecture Choice Justification 17](#_Toc196833985)

[5.3 How Architecture Fits Organizational Requirements 18](#_Toc196833986)

[5.4 Hardware and Software Environment 19](#_Toc196833987)

[**6. Use Scenarios** 19](#_Toc196833988)

[6.1 Admin Use Scenarios 19](#_Toc196833989)

[6.2 Client Use Scenarios 20](#_Toc196833990)

[6.3 Trainer Use Scenarios 20](#_Toc196833991)

[**7. Interface Structure Design** 21](#_Toc196833992)

[**8. Interface Standards** 22](#_Toc196833993)

[8.1 Interface Metaphor 22](#_Toc196833994)

[8.2 Interface Objects 22](#_Toc196833995)

[8.3 Interface Actions 22](#_Toc196833996)

[8.4 Interface Icons 23](#_Toc196833997)

[8.5 Interface Navigation Structure 23](#_Toc196833998)

[**9. Design Decision Explanation** 23](#_Toc196833999)

[9.1 System Design Strategy 23](#_Toc196834000)

[9.2 Navigation Design 24](#_Toc196834001)

[9.3 Layout Design 24](#_Toc196834002)

[9.4 Input Design 24](#_Toc196834003)

[9.5 Output Design 24](#_Toc196834004)

[9.6 Data Storage Design 25](#_Toc196834005)

[9.7 Security Design 25](#_Toc196834006)

[9.8 Architecture and Technology Decisions 25](#_Toc196834007)

[9.9 Acquisition Decision 25](#_Toc196834008)

[**10. Testing Plan and Test Data** 26](#_Toc196834009)

[10.1 Test Planning 26](#_Toc196834010)

[10.2 Integration Test Cases (TP-001, TP-002, TP-003, TP-004) 26](#_Toc196834011)

[10.2.1 Test Plan 1 26](#_Toc196834012)

[10.2.2 Test Plan 2 27](#_Toc196834013)

[10.2.3 Test Plan 3 28](#_Toc196834014)

[10.2.4 Test Plan 4 29](#_Toc196834015)

[**11. Recommendations for Implementation** 30](#_Toc196834016)

[11.1 Conversion Strategy 30](#_Toc196834017)

[11.2 Data Conversion 30](#_Toc196834018)

[11.3 Organizational Change Recommendations 31](#_Toc196834019)

[**Appendix A- a brief user manual** 32](#_Toc196834020)

[**Appendix B- original project proposal** 32](#_Toc196834021)

[**Appendix C- our experience as systems analysts** 33](#_Toc196834022)

[**Appendix D – Links** 34](#_Toc196834023)

[**Prototype Link** 34](#_Toc196834024)

## **1. Executive Summary**

Riyasa Studio is a new fitness and wellness center preparing to launch in Pune, India. The studio will offer a variety of physical wellness services such as yoga, Zumba, dance, and strength training. To ensure smooth operations from the very beginning, the studio needed a centralized digital system to manage essential activities such as client registration, class scheduling, trainer coordination, membership tracking, and studio room rentals. Managing all these manually was identified as a major risk to efficiency, accuracy, and customer experience. To address these needs, a custom web application was developed using Mendix — a low-code development platform that enables quick and structured system design.

The Riyasa Studio Management System is a role-based web application that supports three main user groups: **Administrators**, **Trainers**, and **Clients**. Each user has access to specific features based on their role. Administrators have full control over the system, including the ability to add and manage clients and trainers, create and schedule classes, assign trainers to rooms, approve or reject rental requests, and maintain membership and payment records. Trainers can view their assigned classes, check available rooms, and submit rental requests for private sessions or workshops. Clients can view and book classes, access their membership details, and see their payment history and status.

Some of the key features of the system include:

* A class scheduling and booking module with filtering options
* A rental request workflow where trainers can submit detailed room rental requests
* Membership management with support for multiple plan types, renewal logic, and pricing
* Manual payment tracking and payment method input (without live integration)
* Separate dashboards for each role, ensuring a user-friendly and focused experience
* Role-based access control for security and simplicity

The system was designed following core principles from the Systems Design & Engineering process. It includes a domain model that outlines the relationships between key entities such as Clients, Trainers, Classes, Rooms, Memberships, Rentals, and Payments. The project also applied interface design principles to ensure clean navigation and functional clarity. Although some advanced features such as real-time payment processing, social media integration, and marketing automation were originally proposed, they were intentionally removed from the first release to focus on delivering a stable and usable Minimum Viable Product (MVP).

Overall, the Riyasa Studio Management System delivers a professional and scalable solution to support the studio’s operations. It reduces administrative overhead, improves communication among staff, and ensures a better experience for both clients and trainers. The project demonstrates a thoughtful application of system design methodology, practical feature scoping, and effective use of low-code technology to solve real business problems.

## **2. System Request**

|  |  |  |
| --- | --- | --- |
| **Element** | **Description** | **Project Details** |
| Project Sponsor | The person who initiates the project and serves as the primary point of contact on the business side. | Riya and Sajal, Founders of Riyasa Studio |
| Business Need | The business-related reason(s) for initiating the system. | Riyasa Studio needs a centralized system to manage class scheduling, trainer coordination, membership tracking, and room rentals. Manual methods were inefficient and would hinder launch and future operations. |
| Business Requirements | The new or enhanced business capabilities that the system will provide. | The system allows:  - Admins to create and manage classes, rooms, clients, trainers, and memberships  - Trainers to view class schedules and request room rentals  - Clients to browse and book classes, view memberships, and track payments    - Role-based access for different users  - Tracking of rentals, payments, and attendance |
| Business Value | The benefits that the system will create for the organization. | The system will reduce manual workload, improve coordination between trainers and administrators, enhance client experience, and support data-driven management. It enables faster onboarding of clients, prevents booking conflicts, and improves overall studio efficiency |
| Special Issues or Constraints | Issues that pertain to the approval committee’s decision. | The system needed to be ready before studio launch. Payment gateway integration was postponed due to legal/security constraints, but the system was designed to support these additions in future iterations without rework. |

## **3. Revised Feasibility Analysis**

### 3.1 Introduction

This section evaluates the feasibility of implementing the proposed Studio Management System for Riyasa Studio, considering technical, economic, and organizational aspects. The analysis incorporates updated assumptions reflecting Riyasa Studio's recent establishment and the strategic changes adopted to ensure sustainable growth.

### 3.2 Technical Feasibility

The technical feasibility assesses whether the Riyasa Studio Management System can be successfully developed, deployed, and maintained using Mendix, while focusing on efficient management of bookings, trainer assignments, client memberships, and studio rentals. Integration with payment gateways and marketing automation tools is reserved for potential future phases, based on business growth.

### 3.2.1 Familiarity with the Application & Technology

**Business Familiarity:** Riyasa Studio is a newly launched fitness business, and its owners have limited prior experience with digital fitness management systems. Therefore, the system needed to be intuitive, requiring minimal training for end-users including receptionists, trainers, and studio administrators.

**Development Technology:** Mendix, a low-code development platform, was selected to facilitate rapid, flexible application development. The platform provides built-in database management, security controls, and user interface components, reducing the need for deep technical expertise.

**Risk Level:** Low – Mitigation: User onboarding sessions and utilization of Mendix's pre-built modules to minimize development complexity and speed up adoption.

### 3.2.2 Key Technical Considerations

|  |  |  |
| --- | --- | --- |
| **Factor** | **Risk Level** | **Mitigation Strategy** |
| Database Management | Low | Utilize Mendix’s domain models for efficient and automated database structuring. |
| Hosting & Security | Low | Deploy on Mendix Cloud with built-in encryption and authentication best practices. |
| Scalability | Low | Mendix’s scalable infrastructure supports future expansion beyond the initial user base of 50–100 clients and trainers. |
| Future Payment Integration (Optional) | Medium (future scope) | Design a modular system architecture to easily integrate third-party payment gateways when needed. |
| Notifications and Marketing | Low | External services (existing email/SMS systems and digital marketing managed by staff) remain separate, avoiding unnecessary complexity. |

### 3.2.3 System Integration & Compatibility

As Riyasa Studio was newly established, no legacy systems existed, simplifying system deployment.  
 The current system scope includes centralized management of:

* Client bookings and membership handling
* Trainer assignments and scheduling
* Room rental and studio management workflows

At this stage, no direct integration with payment gateways or social media automation tools is implemented.  
 However, the system architecture is designed to allow easy future expansion, including integration with:

* Payment gateways (e.g., Razorpay, Stripe)
* Notification services (e.g., automated email/SMS APIs)
* Marketing CRM systems if needed

**Risk Level:** Low – Mitigation: Modular design ensures future integrations can be implemented without disrupting current operations.

### 3.2.4 Conclusion

**Overall Technical Feasibility:** Highly Feasible  
The selection of Mendix, combined with a clean, modular design approach, ensures that the Riyasa Studio Management System is technically sound, scalable for future needs, and maintainable with limited technical resources.

**Development Timeline:** 4–6 weeks using Mendix’s Agile development approach.

### 3.3 Economic Feasibility

### 3.3.1 Cost-Benefit Analysis (CBA)

An updated CBA was conducted to assess the financial viability of the system over a five-year period. The analysis incorporates revised assumptions based on Riyasa Studio's early-stage operations and strategic adjustments.

#### Assumptions:

* **Revenue Growth:** Anticipated annual growth of 15%–20% in memberships and studio rentals.
* **Premium Services:** Introduction of premium memberships and private rental offerings from the second operational year (2026).
* **Cost Management:** Phased hiring strategy and delayed investment in non-essential equipment to control operational expenses.
* **Discount Rate:** A 10% discount rate applied to account for the time value of money.
* **Inflation:** Operational costs projected to increase by 5% annually.

#### Financial Projections:

|  |  |  |  |
| --- | --- | --- | --- |
| **Year** | **Total Benefits (₹)** | **Total Costs (₹)** | **Net Cash Flow (₹)** |
| 2025 | 4,50,000 | 2,37,000 | 2,13,000 |
| 2026 | 6,00,000 | 6,30,000 | -30,000 |
| 2027 | 7,00,000 | 9,22,220 | -2,22,220 |
| 2028 | 8,00,000 | 6,66,526 | 1,33,474 |
| 2029 | 9,00,000 | 6,88,379 | 2,11,621 |

**Key Metrics:**

* **Net Present Value (NPV):** ₹2,43,615
* **Return on Investment (ROI):** 310%
* **Break-even Point:** Early 2028 (approximately 2.5 years post-launch)

**Conclusion:** The economic analysis indicates strong financial viability, with a positive NPV and a break-even point within a reasonable timeframe.

A screenshot of a spreadsheet

AI-generated content may be incorrect.

*Note: All financial figures are in Indian Rupees (₹) and are based on projected estimates.*

### 3.4 Organizational Feasibility

### 3.4.1 Strategic Alignment

The implementation of the Studio Management System aligns with Riyasa Studio's strategic objectives to streamline operations and enhance customer experience from the outset.

### 3.4.2 Stakeholder Support

Key stakeholders, including the studio owners and staff, have expressed support for the system, recognizing its potential to improve efficiency and service quality.

### 3.4.3 Resource Availability

Given the studio's startup status, resources are limited. However, the choice of a low-code platform like Mendix reduces the need for extensive technical expertise, making the project manageable within existing constraints.

**Conclusion:** Organizational readiness is sufficient to support the successful implementation and adoption of the proposed system.

### 3.5 Overall Feasibility Conclusion

|  |  |  |
| --- | --- | --- |
| **Feasibility Factor** | **Conclusion** | **Key Points** |
| Technical Feasibility | Highly Feasible | Mendix platform, scalable, low risk |
| Economic Feasibility | Highly Favorable | NPV ₹2,43,615, ROI +310%, break-even ~2.5 years |
| Organizational Feasibility | Highly Favorable | Strong stakeholder support, phased scaling |

## **4. Use Cases**

|  |  |  |
| --- | --- | --- |
| **Use Case Name:** Book a Fitness Class | ID: UC-1 | Priority: High |
| **Actor:** Client | | |
| **Description:** The client searches for available classes, views details, and reserves a class slot | | |
| **Trigger:** Client wants to book a class | | |
| **Type:** External | | |
| **Preconditions:**   1. Client must be logged into the system 2. Class list must be available | | |
| **Normal Course:**   1. Client logs into their account 2. Client navigates to Class Schedule 3. Client searches or filters available classes 4. Client selects a class to view details 5. Client clicks “Reserve Class” button 6. System confirms booking and updates the My Bookings page | | |
| **Postconditions:**   1. The class is reserved 2. Confirmation is shown to client 3. Booking appears in client’s "My Bookings" page | | |

Fig.4.1 Book a Fitness Class use case — casual format.

|  |  |  |
| --- | --- | --- |
| **Use Case Name:** View Trainer Details | ID: UC-2 | Priority: High |
| **Actor:** Client | | |
| **Description:** The client views detailed information about trainers to help choose the right class | | |
| **Trigger:** Client wants to browse trainers before booking a class | | |
| **Type:** External | | |
| **Preconditions:**   1. Client must be logged into the system 2. Trainer information must exist in the system | | |
| **Normal Course:**   1. Client logs into their account 2. Client navigates to the Classes section 3. Client views trainer information linked to classes (Name, Bio, Expertise, Upcoming Classes) | | |
| **Postconditions:**   1. Trainer details are displayed 2. Client can make informed class booking decisions | | |

#### Fig.4.2 View Trainer Details use case — casual format.

|  |  |  |
| --- | --- | --- |
| **Use Case Name:** Request a Room Rental | ID: UC-3 | Priority: High |
| **Actor:** Trainer | | |
| **Description:** The trainer fills out and submits a room rental request form specifying timing, purpose, and other details. | | |
| **Trigger:** Trainer wants to reserve a room for a private session or event. | | |
| **Type:** External | | |
| **Preconditions:**   1. Trainer must be logged into the system 2. Room rental feature must be enabled | | |
| **Normal Course:**   1. Trainer logs into the system 2. Trainer navigates to the Rooms section 3. Trainer views available rooms 4. Trainer selects "Request Rental" 5. Trainer fills rental form (Date, Time, Purpose, Equipment, Expected Attendees) 6. Trainer submits rental request 7. System saves the rental request for admin review | | |
| **Postconditions:**   1. Rental request is submitted and stored with “Pending” status 2. Admin notified of new rental request. | | |

#### Fig.4.3 Request a Room Rental use case — casual format.

|  |  |  |
| --- | --- | --- |
| **Use Case Name:** Manage Clients and Memberships | ID: UC-4 | Priority: High |
| **Actor:** Administrator | | |
| **Description:** Admin creates, edits, and manages client profiles and their membership details | | |
| **Trigger:** Admin wants to update client information or assign memberships | | |
| **Type:** External | | |
| **Preconditions:**   1. Admin must be logged into the system 2. Client management module must be accessible | | |
| **Normal Course:**   1. Admin logs into their dashboard 2. Admin navigates to Clients section 3. Admin views list of clients 4. Admin selects a client to edit or creates a new client 5. Admin updates fields like Name, Contact Info, Status, Membership Details 6. Changes are saved and updated in the system | | |
| **Postconditions:** Client profile and membership data are updated successfully | | |

#### Fig.4.4 Manage Clients and Memberships use case — casual format.

|  |  |  |
| --- | --- | --- |
| **Use Case Name:** Approve or Reject Room Rental | ID: UC-5 | Priority: High |
| **Actor:** Administrator | | |
| **Description:** Admin reviews rental requests submitted by trainers and either approves or rejects them t | | |
| **Trigger:** Admin notices a pending room rental request | | |
| **Type:** External | | |
| **Preconditions:**   1. Admin must be logged into the system 2. Rental requests must exist in “Pending” status | | |
| **Normal Course:**   1. Admin logs into the dashboard. 2. Admin navigates to Rental Requests. 3. Admin views pending requests. 4. Admin opens the request details. 5. Admin either approves (marks as Approved) or rejects (marks as Rejected, adds reason). 6. System updates the status and notifies the trainer. | | |
| **Postconditions:** Rental request is either Approved or Rejected with a reason logged | | |

#### Fig.4.5 Approve or Reject Room Rental use case — casual format.

|  |  |  |
| --- | --- | --- |
| **Use Case Name:** View Dashboard Analytics | ID: UC-6 | Priority: High |
| **Actor:** Administrator | | |
| **Description:** Admin views overall studio analytics including active classes, total users, pending rentals, and recent activity. | | |
| **Trigger:** Admin logs into dashboard. | | |
| **Type:** External | | |
| **Preconditions:**   1. Client must be logged into the system 2. Dashboard module must be operational | | |
| **Normal Course:**   1. Admin logs into the system 2. Dashboard displays key studio metrics:   - Number of active classes  - Total registered users  - Pending rental requests  - Recent class enrollments | | |
| **Postconditions:** Admin sees updated studio overview | | |

#### Fig.4.6 View Dashboard Analytics use case — casual format

|  |  |  |
| --- | --- | --- |
| **Use Case Name:** Manage Class Schedule | ID: UC-7 | Priority: High |
| **Actor:** Administrator | | |
| **Description:** Admin creates, edits, and manages fitness classes, including assigning trainers, rooms, schedules, and setting class status | | |
| **Trigger:** Admin wants to add a new class or modify an existing class. | | |
| **Type:** External | | |
| **Preconditions:**   1. Admin must be logged into the system 2. Classes module must be active | | |
| **Normal Course:**   1. Admin logs into their dashboard 2. Admin navigates to the Classes section. 3. Admin clicks "Add Class" or selects an existing class to edit. 4. Admin fills in or updates Class Name, Description, Trainer, Room, Schedule, and Status. 5. Admin saves the new or updated class. 6. System updates class listings and schedules. | | |
| **Postconditions:**   1. New or updated class is visible to clients and trainers. 2. Class schedule is reflected in booking options. | | |

#### Fig.4.7 Manage Class Schedule use case — casual format

|  |  |  |
| --- | --- | --- |
| **Use Case Name:** Cancel Scheduled Class | ID: UC-8 | Priority: High |
| **Actor:** Trainer | | |
| **Description:** Trainer cancels a class they are scheduled to teach, typically due to unavailability or low enrollment | | |
| **Trigger:** Trainer realizes they cannot attend a scheduled class and needs to cancel | | |
| **Type:** External | | |
| **Preconditions:**   1. Trainer must be logged into the system 2. Class must already exist and be assigned to the trainer | | |
| **Normal Course:**   1. Trainer logs into their account 2. Trainer navigates to My Classes 3. Trainer selects the class to cancel 4. Trainer clicks "Cancel Class" button 5. System updates class status to "Cancelled" and notifies clients (future enhancement) | | |
| **Postconditions:**   1. Class is marked as Cancelled in the system 2. It is no longer available for client booking | | |

#### Fig.3.8 Cancel Scheduled Class use case — casual format

### 4.2 Revised Functional Requirements Based on Use Cases

##### 4.2.1 Client Functional Requirements:

1. As a client, I want to see trainer details so that I can choose the right class.
2. As a client, I want to receive confirmation for my class booking so that I know it’s reserved.
3. As a client, I want to reserve a class so that I can secure my spot.
4. As a client, I want to log in so that I can book classes, view my membership, and class schedules.

##### 4.2.2 Admin Functional Requirements:

1. As an admin, I want to edit client information and membership details.
2. As an admin, I want to create a trainer profile (name, email id, phone number, address, time zone, language, emergency contact).
3. As an admin, I want to create a client profile (name, email id, phone number, address, time zone, language, emergency contact).
4. As an admin, I want to edit details of trainers.
5. As an admin, I want to add joining date and status (active, inactive, pending, frozen) for clients.
6. As an admin, I want to add available rooms of the studio with size, capacity, hourly rate, and status.
7. As an admin, I want to add classes and schedules.
8. As an admin, I want to add any additional notes regarding special discounts or other details.
9. As an admin, I want to view active classes, total users, pending rentals, and recent classes on the dashboard.
10. As an admin, I want to add trainer specialty, bio, and certifications.
11. As an admin, I want to block or delete current users.
12. As an admin, I want to add a new room with room name, capacity, size, and status.
13. As an admin, I want to edit user details like name, username, status (active, blocked) in case of login issues.
14. As an admin, I want to manage current studio rooms and add new rooms if needed.
15. As an admin, I want to add room details like description and equipment needed.
16. As an admin, I want to see the list of active users and their details.
17. As an admin, I want to view and manage room rental requests.
18. As an admin, I want to review rental request details.
19. As an admin, I want to approve or reject room rental requests and provide a rejection reason.
20. As an admin, I want to track all class bookings to manage studio operations.

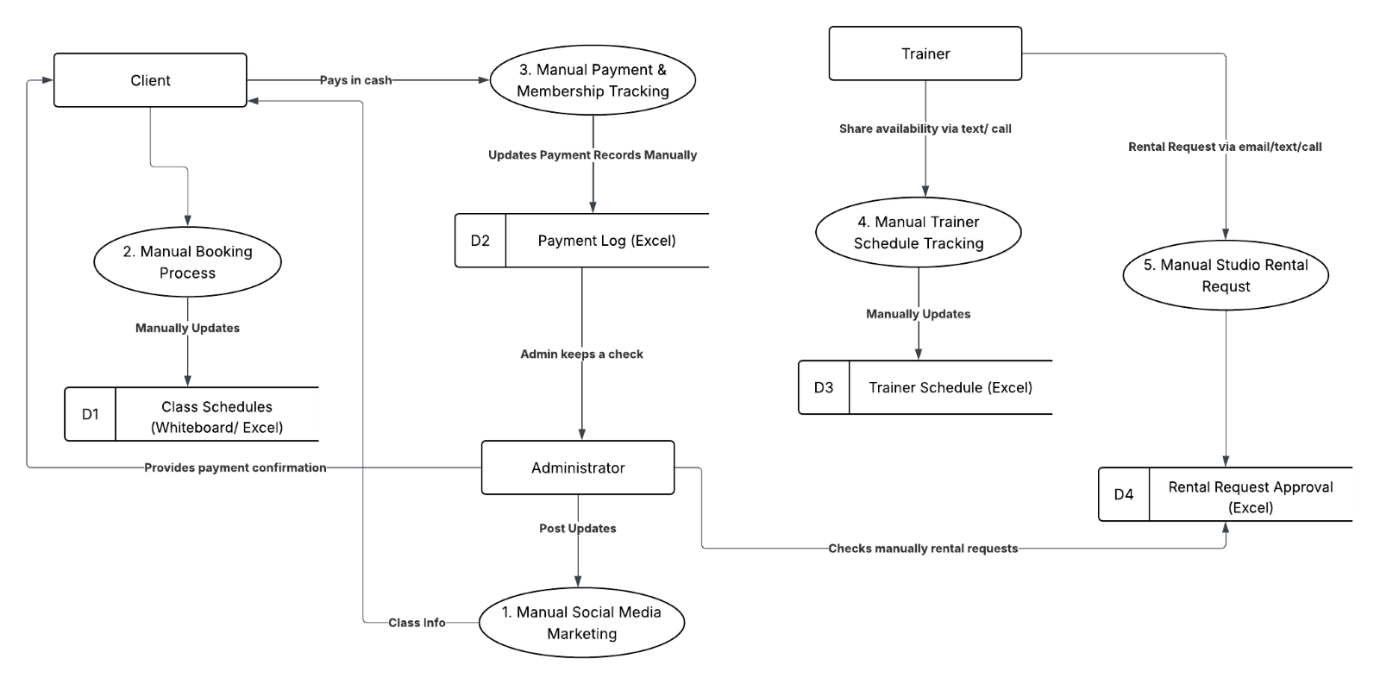
##### 4.2.3 Trainer Functional Requirements:

1. As a trainer, I want to see the number of spots filled in my scheduled class.
2. As a trainer, I want to view class information (date, timing, capacity).
3. As a trainer, I want to request a room rental by filling in date, time, purpose, and attendees.
4. As a trainer, I want to view my assigned classes.
5. As a trainer, I want to check my rental history and current status (approved, rejected).
6. As a trainer, I want to cancel a scheduled class.
7. As a trainer, I want to view available rooms.
8. As a professional trainer, I want to rent the studio to host my own class (future extension: with online payment).

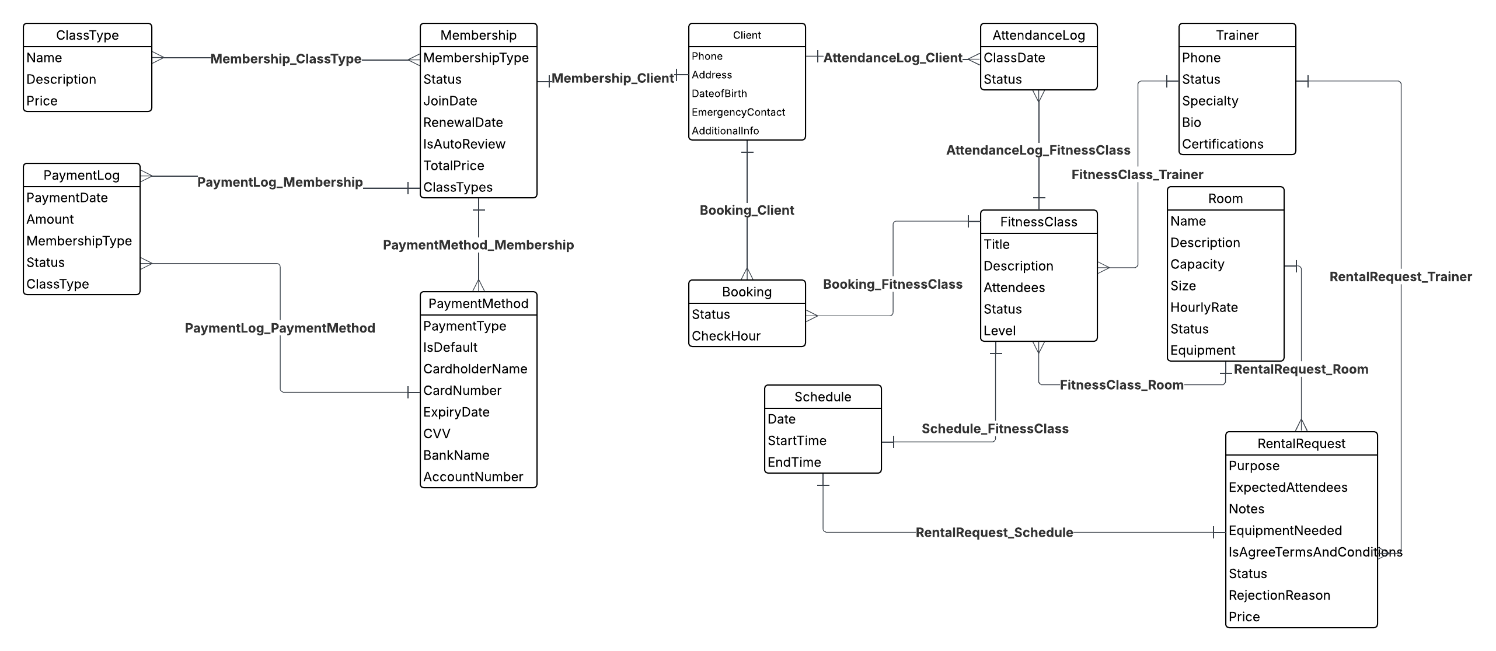
## **4.3 Process and Data Models**

### 4.3.1 AS-IS Process Model

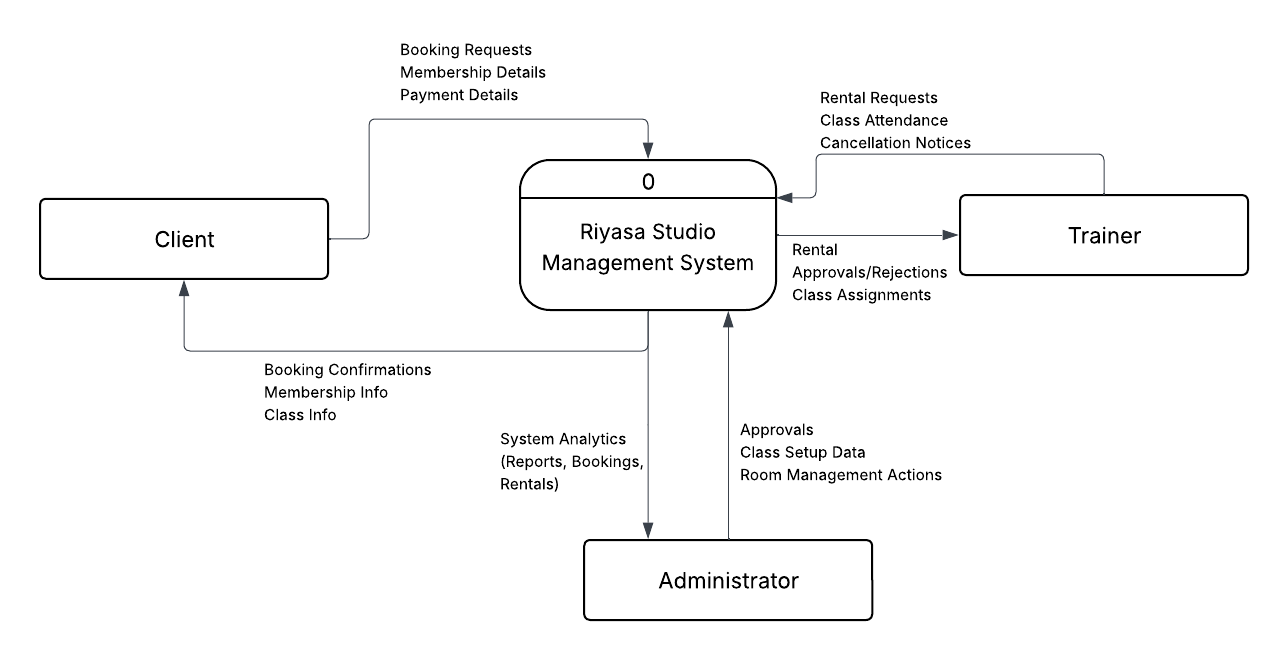
The As-Is process model highlights the current manual workflows used by Riyasa Studio. These include class bookings via phone calls, trainer availability tracked on whiteboards, and membership payments handled in cash without real-time tracking. These fragmented processes lead to inefficiencies, miscommunication, and limited scalability, clearly showing the need for a centralized automated system.

  
Fig 4.3.1 Riyasa Studio Management System — (AS-IS Process Model)

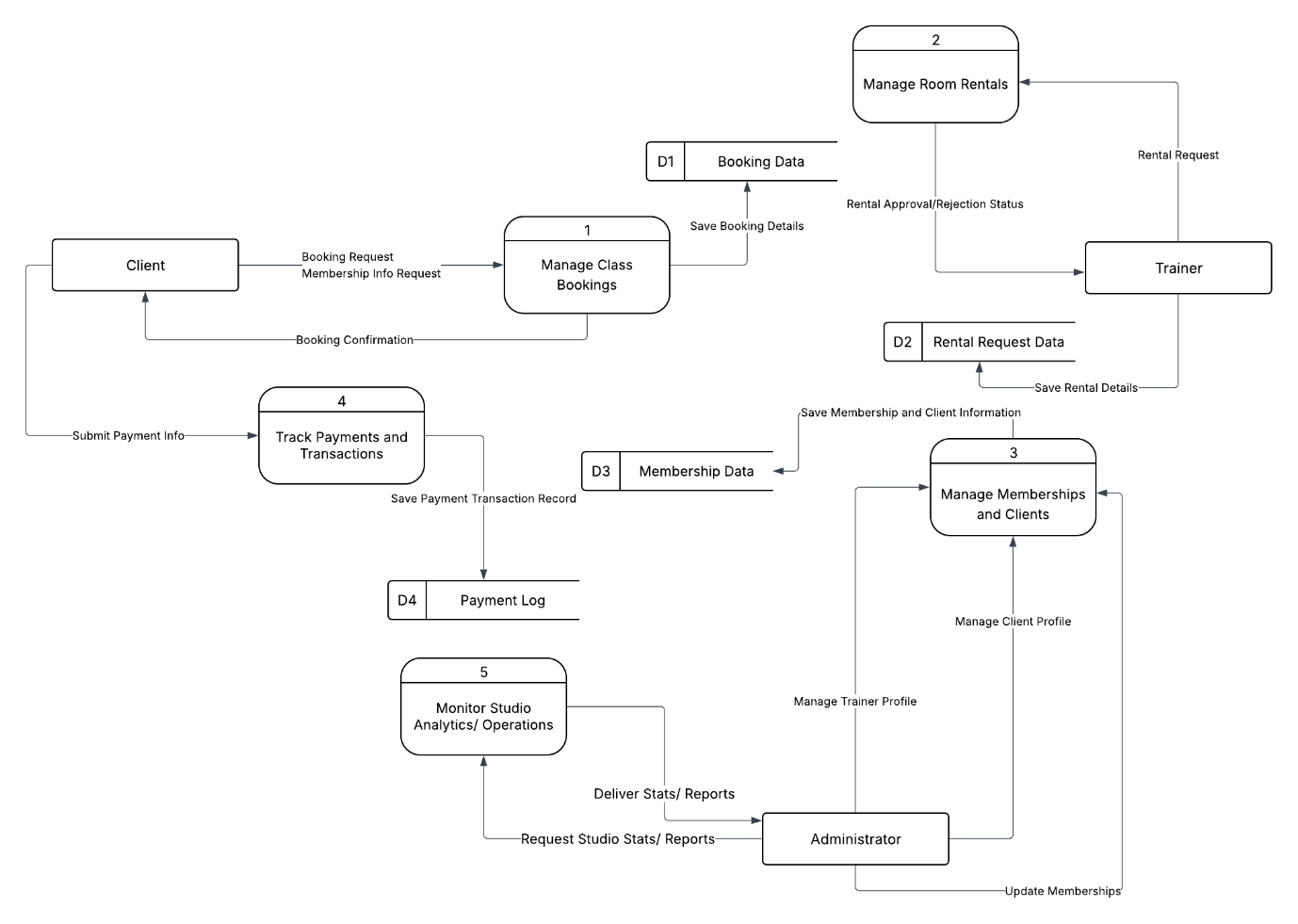
### 4.3.2 Domain Model

Fig 4.3.2 Riyasa Studio Management System — Domain Model (To-Be System)

### 4.3.3 Context Diagram (Top Level DFD)

Fig 4.3.3 Riyasa Studio Management System — Context Diagram

### 4.3.4 Level 0 DFD

fig.4.3.4 Riyasa Studio Management System — Context Diagram Decomposed into Level 0 Diagram

Note: [Link of Data and Process Models done on LucidChart](https://lucid.app/lucidchart/64c9b1ac-d021-47d9-a1b0-ae1e3db7e0ab/edit?viewport_loc=-853%2C285%2C2437%2C1365%2CYcda8NISKwFQ&invitationId=inv_41e104cf-6e0b-4a83-9019-0cc2d578addb)

## **5. System Architecture Design**

### 5.1 System Architecture Overview

The Riyasa Studio Management System has been developed as a web-based thin-client application, within a client-server model using the Mendix low-code platform. Clients, trainers, and administrators interact with the system through standard web browsers. The application logic, data access logic, and data storage are all managed centrally in the Mendix Cloud environment. This architecture ensures accessibility, scalability, and lower maintenance costs for Riyasa Studio.

### 5.2 Architecture Choice Justification

**Thin-Client Web-Based System:**  
 Riyasa Studio requires easy access for clients, trainers, and admins from different locations (home, studio, mobile devices). A web-based thin-client approach supports this by only requiring a web browser and an internet connection, reducing technical barriers for users.

**Centralized Data Storage:**  
 All data — client information, trainer details, class schedules, bookings, rentals, and memberships — is stored securely in the Mendix Cloud database. Central storage simplifies data management, backup, and security, and supports real-time updates.

**Cost Efficiency and Maintainability:**  
 A web-based approach eliminates the need to maintain individual installations on each client device. Updates to the system can be done once centrally without requiring client-side intervention. This reduces IT overhead, ideal for a startup studio.

**Scalability for Growth:**  
 As the business expands, Mendix’s scalable cloud services can handle increased users, classes, and rental operations without major changes to infrastructure.

### 5.3 How Architecture Fits Organizational Requirements

**Operational Requirements:**

* + **System Integration:** The system can easily integrate with payment gateways and marketing tools via web APIs when needed.
  + **Portability:** Users can access the system through any modern web browser across desktops, tablets, and smartphones.
  + **Maintainability:** Updates are deployed centrally via the cloud platform with minimal downtime.

**Performance Requirements:**

* + **Speed:** Web-based operations ensure quick loading and booking confirmation for users.
  + **Capacity:** The system can initially support up to 100 simultaneous users, scalable with Mendix resources.
  + **Availability:** The system will have at least 99% uptime, relying on Mendix’s managed cloud services.

**Security Requirements:**

* + **Access Control:** Role-based access control is implemented, ensuring that clients, trainers, and administrators can only access features appropriate to their role. Sensitive functions like managing classes, rentals, and system settings are restricted to authorized users.
  + **Authentication:** Secure login is required for all users.
  + **Encryption:** While payment features are not yet fully activated, future plans involve SSL encryption for payment processing.

**Cultural/Political Requirements:**

* + **Language:** The initial release is English-only. The system is designed to be expandable for multilingual support if needed.
  + **Legal:** Basic client information is collected and stored securely following standard data protection practices.

### 5.4 Hardware and Software Environment

|  |  |
| --- | --- |
| Component | Specification |
| Client Devices | Modern web browsers (Chrome, Edge, Safari) on Windows/macOS/Linux; Mobile browsers on iOS/Android |
| Web Server | Mendix Cloud (hosted environment) |
| Database Server | Managed Database by Mendix Platform |
| Network | Always-on broadband internet recommended for users |

## **6. Use Scenarios**

Based on the user stories, the following core use scenarios were developed. Each scenario represents a specific path through the system based on user goals and actions.

### 6.1 Admin Use Scenarios

**Scenario A1: Creating and Managing Client and Trainer Profiles**

* Admin logs into the Admin Main Menu.
* Navigates to the Clients or Trainers section.
* Creates a new Client or Trainer profile (RIY-15, RIY-19).
* Activates the profile by setting the joining date and status (RIY-16).
* Edits client or trainer details if updates are needed (RIY-22, RIY-21).

**Scenario A2: Managing Rooms and Facilities**

* Admin accesses the Rooms section.
* Adds new rooms with size, capacity, and hourly rates (RIY-23, RIY-25).
* Updates room descriptions and equipment details (RIY-26).
* Manages all studio rooms and their availability (RIY-24).

**Scenario A3: Managing Classes and Trainers**

* Admin accesses the Classes section.
* Creates new classes and schedules (RIY-18).
* Adds specialty, bio, and certifications for trainers (RIY-20).
* Adds any specific notes or discounts for clients (RIY-17).

**Scenario A4: Monitoring Dashboard Metrics**

* Admin views active classes, users, rentals, and recent activities from the dashboard (RIY-13).
* Tracks all class bookings to support studio operations (RIY-10).

**Scenario A5: Handling Room Rentals**

* Admin views room rental requests (RIY-27).
* Reviews rental details (RIY-29).
* Approves or rejects rental requests and records rejection reasons (RIY-28).

**Scenario A6: Managing Users**

* Admin views active users and their details (RIY-30).
* Edits user information and changes user statuses as needed (RIY-31).
* Blocks or deletes users when necessary (RIY-32).

### 6.2 Client Use Scenarios

**Scenario C1: Browsing and Booking Classes**

* Client logs into the Client Main Menu (RIY-1).
* Views upcoming classes and available trainers (RIY-3).
* Reserves/book classes based on availability (RIY-2).
* Receives booking confirmation after a successful reservation (RIY-5).

**Scenario C2: Managing Membership and Bookings**

* Client views membership details and renews memberships if needed (RIY-1).
* Manages class bookings by viewing schedule and booking history.

### 6.3 Trainer Use Scenarios

**Scenario T1: Managing Assigned Classes**

* Trainer logs into the Trainer Main Menu.
* Views assigned classes (RIY-33).
* Manages class schedules and cancellations (RIY-34).
* Views detailed class information (RIY-36).
* Checks number of enrolled clients for each class (RIY-35).

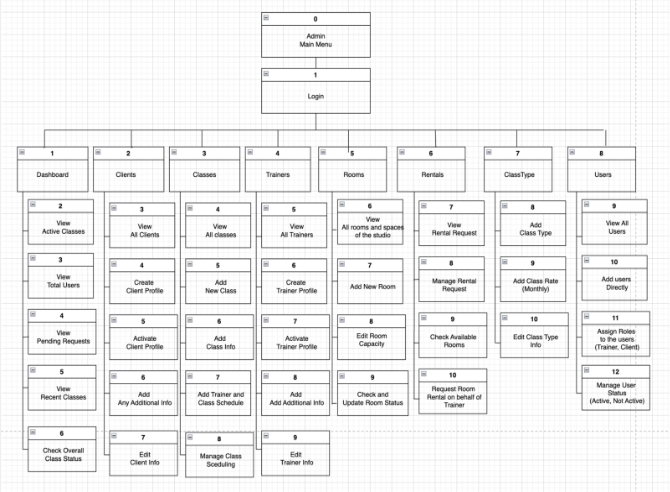
**Scenario T2: Handling Room Rentals**

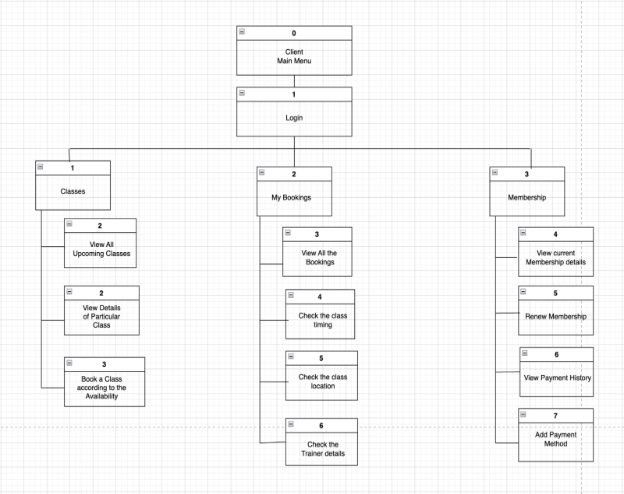
* Trainer views available rooms (RIY-37).
* Requests rentals by providing time, purpose, and expected attendees (RIY-38).
* Checks rental history and status (approved/rejected) (RIY-39).

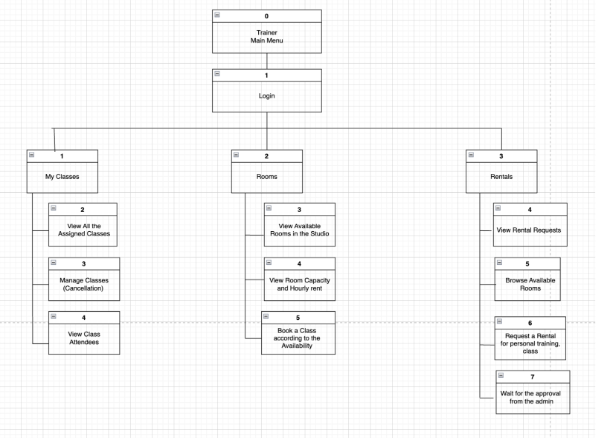
**Scenario T3: Hosting Personal Classes**

* A professional trainer rents the studio independently to conduct personal classes (RIY-11).

## **7. Interface Structure Design**







## **8. Interface Standards**

### 8.1 Interface Metaphor

* None

### 8.2 Interface Objects

|  |  |
| --- | --- |
| **Object** | **Description** |
| **Class** | A scheduled fitness session clients can book |
| **Client** | A user registered to attend fitness classes |
| **Trainer** | A fitness trainer who manages and delivers classes |
| **Room** | Physical space where classes are held |
| **Rental Request** | Request to rent a studio room |
| **Class Type** | Category of classes (e.g., Yoga, Zumba) |
| **Membership** | Client's subscription plan for attending classes |
| **Booking** | A specific reservation made by a client for a class |
| **Payment** | Client’s transaction records and methods |
| **User Roles** | Access rights assigned to users (Admin, Trainer, Client) |

### 8.3 Interface Actions

|  |  |
| --- | --- |
| **Action** | **Description** |
| **View** | Access and see lists/details (e.g., classes, clients, rooms, rentals) |
| **Create** | Add new information (e.g., new class, trainer profile, rental request) |
| **Edit** | Modify existing data (e.g., edit class info, update trainer profile) |
| **Manage** | Administer operations (e.g., schedule classes, approve rentals, manage users) |
| **Activate** | Set client or trainer profiles to active status |
| **Submit** | Send a new rental or booking request |
| **Book** | Reserve a spot in a fitness class |
| **Renew** | Update membership subscription |
| **Upload** | (Future Scope) Optionally upload trainer certifications or client documents |
| **Download** | (Future Scope) Export booking or payment history (for clients/admin) |

### 8.4 Interface Icons

* **Riyasa Fitness Studio Logo** appears on all main screens.
* **Icons for Classes, Clients, Trainers, Rentals, Rooms, Membership** for quick access.
* **Status Icons** (Active/Inactive) for users and rentals.

### 8.5 Interface Navigation Structure

The app will have different menus based on **user roles**:

* **Admin**
  + Dashboard, Clients, Classes, Trainers, Rooms, Rentals, Class Type, Users
* **Client**
  + Classes, My Bookings, Membership
* **Trainer**
  + My Classes, Rooms, Rentals

Each menu offers **view**, **create**, **edit**, **manage**, or **submit** actions specific to that role.

**Notes:**

* Interface consistency is maintained across Admin, Trainer, and Client panels.
* Easy navigation: all modules start from a **Main Menu** post-login.
* Simple **two-click access** to perform major actions.
* User-friendly design following principles of **layout**, **content awareness**, **aesthetics**, **consistency**, and **minimal user effort** based on textbook guidelines

## **9. Design Decision Explanation**

### 9.1 System Design Strategy

The Riyasa Fitness Studio Management System was developed using **Mendix**, a leading low-code development platform.  
 The decision to use Mendix was based on its ability to:

* Rapidly prototype and build the application
* Provide prebuilt UI components and workflows
* Integrate easily with backend databases
* Support responsive web design without complex manual coding

Using Mendix enabled faster development and iterative refinement based on user feedback, aligning well with an **Agile-like approach** recommended in modern system development.

### 9.2 Navigation Design

The navigation structure was designed using **Mendix page layouts and navigation profiles**:

* Each user role (Admin, Client, Trainer) was given a **separate navigation menu** customized for their functionalities.
* **Main menu pages** and **quick action buttons** were created using Mendix's built-in navigation tree and button widgets.
* Navigation consistency was enforced by reusing **navigation layouts** across pages, promoting minimal user effort.

Mendix automatically handles **role-based navigation profiles**, ensuring users only see relevant menu items after login.

### 9.3 Layout Design

Layout design followed Mendix's responsive design templates:

* **Top navigation bar** with branding (studio logo) and account controls (profile, logout).
* **Page layouts** were standardized using Mendix's "Atlas UI" design system.
* **Card-based grids** and **datagrids** were used for listing records (e.g., classes, clients, rooms).
* **Form layouts** were used for data input screens, ensuring fields were clearly labeled and grouped.

All layouts were tested on desktop, tablet, and mobile views to ensure full responsiveness.

## 9.4 Input Design

Input forms were designed using Mendix's **form widget library**:

* **Text boxes**, **dropdowns**, **date pickers**, and **reference selectors** were used to make data entry easy and accurate.
* **Validation rules** (required fields, valid email formats, phone number formats) were set directly in Mendix through attribute settings and microflows.
* Wherever possible, **reference selectors** linked inputs to existing entities (e.g., selecting trainer name from a list rather than free text).

This reduced user errors and aligned with good input validation practices .

### 9.5 Output Design

Key outputs were designed using Mendix's:

* **Datagrids** to display lists (clients, classes, rooms, rentals)
* **Dashboards** created using **data widgets** (e.g., count widgets, charts) for Admin users
* **Confirmation pop-ups** and **success messages** for user actions like booking a class or approving a rental.

Outputs were designed to be real-time and provide immediate feedback, improving system usability and transparency.

## 9.6 Data Storage Design

Mendix automatically manages data persistence using its **built-in object-relational mapping (ORM)** system:

* Domain models were designed visually in Mendix, defining **entities**, **attributes**, and **associations** (relationships).
* Important entities included: **Client**, **Trainer**, **Class**, **Room**, **Rental**, **User**.
* Associations were normalized where appropriate to avoid redundancy (e.g., many-to-many between Clients and Classes).

Mendix's data storage model ensured consistent, secure, and scalable handling of studio data.

### 9.7 Security Design

Security was implemented using Mendix's built-in **access control features**:

* **User roles** (Admin, Client, Trainer) were created in the Mendix security model.
* **Page-level and entity-level permissions** were set to restrict actions based on user role.
* Secure login was enforced, and sensitive actions (like approving rentals) were only available to Admins.

Additionally, validation microflows prevented unauthorized data updates or access beyond assigned permissions.

### 9.8 Architecture and Technology Decisions

The system follows a **client-server architecture** supported by Mendix:

* The **front end** (UI) is web-based and responsive.
* The **backend** (database and logic) is managed by Mendix’s cloud services or an on-premise server if desired.
* Mendix enables quick integration with APIs if future expansion is needed (e.g., online payments, third-party scheduling).

This architecture ensures high scalability and easy maintenance, consistent with textbook best practices .

### 9.9 Acquisition Decision

Rather than purchasing a prebuilt fitness management system, we chose a **custom-build approach using Mendix**:

* Customization flexibility
* Faster time-to-market compared to traditional development
* Easier to adapt and evolve the application in future iterations

This approach also aligned with the studio's need for specific rental, membership, and trainer management workflows that off-the-shelf software would not fully support.

## **10. Testing Plan and Test Data**

### 10.1 Test Planning

As the design and development of the Riyasa Studio Management System progressed, a comprehensive testing plan was prepared to ensure the reliability, functionality, and overall quality of the system. Testing is a critical phase of system development, where the system is verified against the original requirements and specifications. Thorough testing helps identify and correct errors early, improves user satisfaction, and ensures that the system performs correctly under real-world conditions. Special emphasis was placed on testing integrated processes that involve multiple components of the system — for example, booking a class, submitting rental requests, and updating payments — to ensure that they function smoothly when combined.

The testing plan focuses specifically on integration testing, which evaluates whether different modules of the system (such as client booking, membership management, rental approval, and payment handling) work together as intended. Several non-trivial business processes were selected for this testing phase because they involve complex interactions between users, interfaces, database operations, and internal workflows. Each test case defines the testing objective, identifies the key interfaces involved, provides the specific input data used during the test, outlines the sequence of actions performed, and records both the expected and actual results. Special attention was given to testing real-world user flows to ensure that the system not only functions correctly in isolated cases, but also in end-to-end scenarios. The results of the testing demonstrate that the Riyasa Studio Management System is stable, robust, and ready for operational use.

### 10.2 Integration Test Cases (TP-001, TP-002, TP-003, TP-004)

#### 10.2.1 Test Plan 1

|  |  |
| --- | --- |
| **Field** | **Value** |
| Program ID: | Riyasa Studio Management System |
| Version number: | 1.0 |
| Tester: | Sakshi Rakshe |
| Date Designed: | March 20, 2025 |
| Date Conducted: | April 21, 2025 |
| Results: | ☑ Passed ⬜ Open items |

|  |  |
| --- | --- |
|  |  |
| Test ID: | TP-001 |
| Requirement Addressed: | Integration Testing - Client Profile Management and Booking Cancellation |
| Objective: | Verify that clients can successfully edit their personal profiles, cancel booked classes, and the system accurately updates client records, class enrollment counts, and history logs. |

|  |  |  |
| --- | --- | --- |
| **Interface ID** | **Data Field** | **Value Entered** |
| 1 | Client Name Update | Sakshi R. |
| 2 | Phone Number Update | +91-9876543210 |
| 3 | Emergency Contact Update | +91-9123456780 |
| 4 | Cancel Booking | Class: Morning Yoga |
| 5 | Check Class Enrollment Count | After cancellation |
| 6 | Booking History Update | Mark as "Cancelled" |

**Script**

1. Client logs into account.
2. Client updates profile details (Name, Phone, Emergency Contact).
3. Client navigates to "My Bookings" and cancels a booked class.
4. System updates class enrollment count automatically.
5. System records booking cancellation under client's history.

**Expected Results/Notes**

* Profile updates reflect immediately.
* Class booking is cancelled without system errors.
* Class available slots increase after cancellation.
* Client booking history correctly records the cancellation event.
* Admin dashboard also reflects updated enrollment numbers.

**Actual Results/Notes**

* Profile updated successfully
* Class cancellation successful
* Enrollment count updated
* History log updated
* No system errors encountered

#### 10.2.2 Test Plan 2

|  |  |
| --- | --- |
| **Field** | **Value** |
| Program ID: | Riyasa Studio Management System |
| Version number: | 1.0 |
| Tester: | Sakshi Rakshe |
| Date Designed: | March 28, 2025 |
| Date Conducted: | April 27, 2025 |
| Results: | ☑ Passed ⬜ Open items |

|  |  |
| --- | --- |
|  |  |
| Test ID: | TP-002 |
| Requirement Addressed: | Trainer Rental Request Management |
| Objective: | Verify that trainers can submit a rental request, admin can approve it, and the room status updates accordingly. |

**Test Cases**

|  |  |  |
| --- | --- | --- |
| **Interface ID** | **Data Field** | **Value Entered** |
| 1 | Trainer Login | Trainer: Riya Fitness |
| 2 | Request Room Rental | Main Studio, 2 hours, Yoga Workshop |
| 3 | Admin Login | Admin user |
| 4 | Approve Rental Request | Rental ID: 1023 |
| 5 | Verify Room Status Update | Main Studio – Status: Reserved |

**Script**

1. Trainer logs into system.
2. Submits rental request for "Main Studio" room for specific date/time.
3. Admin logs in separately.
4. Admin approves the rental request.
5. System updates the room availability.
6. Trainer receives confirmation.

**Expected Results/Notes**

* Rental request submitted and stored.
* Admin sees pending request.
* Approval updates room status to "Reserved".
* Trainer receives approval confirmation.
* No double booking allowed for same time slot.

#### 10.2.3 Test Plan 3

|  |  |
| --- | --- |
| **Field** | **Value** |
| Program ID: | Riyasa Studio Management System |
| Version number: | 1.0 |
| Tester: | Sakshi Rakshe |
| Date Designed: | April 27, 2025 |
| Date Conducted: | April 27, 2025 |
| Results: | ☑ Passed ⬜ Open items |

|  |  |
| --- | --- |
|  |  |
| Test ID: | TP-003 |
| Requirement Addressed: | Class Booking and Payment Integration |
| Objective: | Verify that booking a class and submitting payment updates client booking record and admin dashboard counts. |

**Test Cases**

|  |  |  |
| --- | --- | --- |
| **Interface ID** | **Data Field** | **Value Entered** |
| 1 | Client Login | Client: Sakshi R. |
| 2 | Book Fitness Class | Morning Zumba, 6 PM |
| 3 | Make Payment | ₹500 using UPI |
| 4 | Admin Dashboard View | Update in "Total Active Bookings" counter |
| 5 | Class Enrolment Count | Updated from 8/10 to 9/10 |

**Script**

1. Client logs in and books Morning Zumba class.
2. System confirms booking details.
3. Client proceeds to make payment.
4. Payment success triggers booking confirmation.
5. Admin dashboard real-time updates:
   1. Active bookings counter increases.
   2. Class enrollment for Morning Zumba updates.

**Expected Results/Notes**

* Client booking is saved.
* Payment confirmation securely received.
* Admin dashboard reflects the new booking count without delay.
* Morning Zumba class shows one additional client booked.

#### 10.2.4 Test Plan 4

|  |  |
| --- | --- |
| **Field** | **Value** |
| Program ID: | Riyasa Studio Management System |
| Version number: | 1.0 |
| Tester: | Sakshi Rakshe |
| Date Designed: | April 27, 2025 |
| Date Conducted: | April 27, 2025 |
| Results: | ☑ Passed ⬜ Open items |

|  |  |
| --- | --- |
|  |  |
| Test ID: | TP-004 |
| Requirement Addressed: | Admin Management of Class Information |
| Objective: | Verify that when an admin updates class details (e.g., timing, trainer assigned), the system correctly updates the information across client bookings and trainer schedules without errors. |

**Test Cases**

|  |  |  |
| --- | --- | --- |
| **Interface ID** | **Data Field** | **Value Entered** |
| 1 | Admin Login | Admin User |
| 2 | Class Edit Page | Morning Yoga |
| 3 | Update Time | Change from 6:00 AM to 7:00 AM |
| 4 | Update Trainer Assigned | New Trainer: Priya Sharma |
| 5 | Verify Client Bookings Update | Clients booked see updated timing |
| 6 | Verify Trainer Schedule Update | New trainer sees assigned class |

**Script**

1. Admin logs into the system.
2. Navigates to Classes Management.
3. Selects "Morning Yoga" class.
4. Updates class time to 7:00 AM.
5. Assigns a new trainer to the class.
6. System automatically updates all existing client bookings to show the new time.
7. New trainer sees the updated schedule under their "My Classes."

**Expected Results/Notes**

* Class timing change reflected under client bookings ("My Bookings").
* New trainer assigned correctly, visible under trainer's schedule.
* No duplicate class records created.
* No client booking is lost or deleted.
* System updates processed without system errors.

## **11. Recommendations for Implementation**

### 11.1 Conversion Strategy

The Riyasa Studio Management System is the studio’s first operational software system, replacing manual and spreadsheet-based processes. A **Direct Conversion** strategy is recommended, in which the old manual processes will be immediately replaced by the new system upon launch. This approach is suitable because the system is not replacing a deeply embedded legacy system, and because the business needs to move rapidly to fully digital operations before studio opening. However, to minimize risk, thorough system testing and user training should be completed prior to go-live.

### 11.2 Data Conversion

All critical business data, including client profiles, trainer profiles, class schedules, room availability, and initial membership plans, should be manually entered into the new system by administrators during the pre-launch period. A double-check process is recommended to ensure that critical data is entered accurately before full launch.

### 11.3 Organizational Change Recommendations

* **User Training:** Short training sessions should be held for administrators and trainers, focusing on their role-specific tasks within the system.
* **Client Onboarding:** Clients should be provided with a simple, visual one-page guide explaining how to log in, view classes, and book sessions.
* **Help and Support:** During the first month after launch, a part-time technical support resource (or Admin) should be available to address any user issues and gather feedback for potential system improvements.

These steps will ensure that Riyasa Studio’s transition to the new system is smooth, successful, and sustainable as the business grows.

## **Appendix A- a brief user manual**

User Manual

[User Manual for RIYASA: Fitness Studio Management App](https://scribehow.com/shared/Requesting_A_Rental_For_A_Fitness_Class__VmyJQ8NiSseuvHstgeDxXQ)

To ensure smooth adoption and effective utilization of the Riyasa Studio Management System, a comprehensive User Manual was developed using Scribe, an intuitive documentation tool. The manual provides step-by-step visual guidance tailored to different user roles, including administrators, trainers, and clients.

The primary objectives of the User Manual are to:

* Facilitate a quick learning curve for all users with minimal technical background.
* Provide clear, actionable instructions for critical tasks such as class bookings, trainer scheduling, rental management, and client profile updates.
* Serve as an ongoing reference resource to minimize dependency on technical support.

The manual includes annotated screenshots, sequential workflows, and simplified explanations to accommodate users at all proficiency levels. By leveraging Scribe's interactive documentation features, the manual ensures that users can independently navigate the system, reducing training time and improving overall system adoption rates.

## **Appendix B- original project proposal**

The original project proposal submitted as part of Deliverable 1 can be accessed using the following link:

[MIS573\_Rakshe\_Prayag\_Vaidya\_Ghorpade-Project\_Proposal\_ (1).docx](https://wpi0-my.sharepoint.com/:w:/g/personal/srakshe_wpi_edu/EVKdx3L3SaNGiabH8CaaXtEB886NZQTqHAU1kHmdcTI_Qw?e=l2o68e)

Note: This document is stored on WPI OneDrive. Ensure you are logged in with your WPI credentials to access the file.

## **Appendix C- our experience as systems analysts**

Each Team Member’s Reflections

During the development of the Riyasa Studio Management System, I learned that system design is not only about creating models but also about thinking critically about how the system will work once it is deployed. Working on the testing plan and preparing recommendations for system implementation made me realize how important it is to look ahead — not just at building the system, but at how users will experience it in real life. One key lesson for me was understanding the connection between system requirements and testing: if the requirements are clear, designing meaningful, non-trivial test cases becomes much easier. Another thing that stood out was how planning the conversion and organizational change is just as important as technical design, especially when users are not familiar with digital systems. A surprise I encountered was how much planning it takes even for simple processes like client bookings or trainer rentals to be reliable during testing. If I were to do a similar project again, I would want to start thinking about user adaptation strategies and realistic testing scenarios earlier in the project timeline. Overall, this experience helped me build a more complete view of what it means to deliver a usable and successful system.

* Vikrant Prayag

Building the Riyasa Studio Management System was a valuable learning experience that strengthened both my technical and business skills. I learned how to design and develop scalable applications using Mendix, apply user-centric design principles, and create practical solutions aligned with business needs. Through this project, I also gained hands-on experience with feasibility analysis, Agile development, and technical documentation. Working closely on system design decisions taught me the importance of balancing functionality, cost-efficiency, and scalability, especially for startups. This experience has prepared me to approach future projects with a more strategic mindset, combining technical proficiency with business understanding to deliver impactful digital solutions.

- Noopura Vaidya

Working on the Riyasa Studio Management System gave me a real-world understanding of the systems analysis and design process. I learned that successful system design is not just about functionality, but about understanding user needs, defining clear system requirements early, and validating feasibility across technical, economic, and organizational dimensions. One of the biggest lessons was seeing how much time and iteration the requirements gathering phase actually requires in real projects, even when they seem simple at first. A surprise for me was how quickly small, overlooked details — like role-specific permissions or rental overlaps — could complicate the system flow later if not addressed early. If I were to approach this again, I would spend even more time upfront validating use cases and edge conditions before moving into system modeling. This project helped me connect technical development with user-centered design and strategic business thinking, which will be crucial in my future career.

-Ajinkya Ghorpade

Working on the Riyasa Studio Management System project has given me a deep appreciation for the importance of clear system design and careful planning. One key lesson I learned is how critical it is to thoroughly understand user requirements and translate them into structured models like use cases, data flow diagrams, and domain models. Initially, I thought that building a system would be mostly about technical development, but I realized that successful system design requires constant communication, detailed analysis, and forward-thinking architecture decisions. A surprise for me was seeing how deeply integration across different parts of the system — such as client bookings, rental requests, and payments — affects the overall design. Every change in one module impacts others. If I were to do a similar project again, I would spend even more time up front designing detailed process flows and planning for user training, because those steps truly make implementation smoother. Overall, this project helped me grow my skills as a systems analyst and appreciate the full development lifecycle.

* Sakshi Rakshe

## **Appendix D – Links**

Cost Benefit Analysis: [CBA\_RIYASA\_Final.xlsx](https://wpi0.sharepoint.com/:x:/s/gr-MIS573SystemDesignandDevelopment/EXgIWDkiEOFJp8kIAfGpX5IBWDOIcNAkCqfAfGfze2-JUg?e=a9O2xf)

[User Interface Diagram](https://app.diagrams.net/?src=about#G1WSTQT1rRrhSDR8QrFeYiR_J2QctVs5kc#%7B%22pageId%22%3A%22VO2Kz955KxXBQW-z3FlA%22%7D)

## **Prototype Link**

Link: [Riyasa Studio Prototype Link](https://riyasa-fitnessstudioapp-sandbox.mxapps.io/index.html?profile=Responsive)

For admin

User\_name : demo\_administrator

Password: Abc1234567890

For Client:

User\_name : [ajinkyagh@gmail.com](mailto:ajinkyagh@gmail.com)

Password: Abc1234567890

User\_name : [noopura@gmail.com](mailto:noopura@gmail.com)

Password: Abc1234567890

For Trainer:

User\_name : [stevan@gmail.com](mailto:stevan@gmail.com)

Password: Abc1234567890

User\_name : [john@gmail.com](mailto:john@gmail.com)

Password: Abc1234567890