

Software Specification for Training Class Project

<https://c4ace192.caspio.app/new-training-class/public/home?sid=cee7fb50533c4a63867f130eb256fba6>

Software Requirements Specification for Training Class Project

Table of Contents

Revision History	2
Introduction	3
Overall Description	3
User Classes and Characteristics	3
System Features	4
Data Requirements	6
User Interface Requirements	9
Nonfunctional Requirements	13
References	15
Appendix 1: Domain Diagram	16
Appendix 2: Use Case Documentation	17
Appendix 3: UI Documentation	26
Appendix 4: User Guide - Training Class	29

Revision History

Component Name	Date	Description of/ Reason for Changes
Domain Class Model	3/12/2025	Previous domain class model was flawed, so I made a new one.
System Features	3/26/2025	Revisions to better describe use cases and diagrams.
System Features	4/27/2025	Edited to align with the final project prototype.
Domain Class Model	4/27/2025	Edited to align with the final project prototype.
Data Dictionary	4/27/2025	Edited to align with the final project prototype.

Introduction

This report delivers the final version of the Software Requirements Specification (SRS) document, which has been updated and expanded based on feedback and project evolution throughout the semester.

Overall Description

Product Description and Scope

Stay Well Studios aims to provide high-quality personal and group training sessions to help clients achieve their fitness goals. As it stands, managing and scheduling training schedules for both personal and semi-private is challenging due to manual registration and communication with each client. To address this, this system will involve the development of a scheduling and tracking system to allow the owners to efficiently plan classes with specific trainers while also monitoring their activity for all sessions.

This system aims to streamline class schedules by ensuring that clients and trainers are effectively matched up and organized on record. With the process of automation, the business will improve operational efficiency, client satisfaction, and reduce scheduling error. The app will provide a structured approach to class scheduling, tracking and configuring, allowing the business to scale effectively with increased clientele and include an improved user experience for trainers and clients.

User Classes and Characteristics

Client

- This user would be responsible for registering to scheduled group classes and view available sessions. They require a user-friendly interface to sign up for classes and manage their schedules. Their goal is to find classes that fit their fitness needs.

Trainer

- This user would be responsible for conducting personal and semi-private training sessions. They need to have access to view their scheduled classes, schedule group classes, and update training activity. Trainers rely on an intuitive system that allows them to manage and communicate with their clients.

Owner

- This user is able to schedule group classes, edit and create classes, and monitor overall activity. They require a management dashboard to overview trainer activity and class scheduling.

System Features

Background

To develop a robust information system for Stay Well Studio (SWS), the team conducted an in-depth analysis of discussions with co-owners Jackie and Tom to understand their operational challenges and business goals. This process involved identifying pain points in their current workflow such as difficulties in managing memberships, payments, scheduling, and inventory, as well as inefficiencies in class management and attendance tracking. Additionally, the team considered their need for future opportunities for online content monetization and using language learning model "ChatGPT" to help brainstorm solutions.

Based on these insights, the team defined key system requirements by prioritizing the most critical functionalities that would streamline operations and improve client engagement. The proposed system in this deliverable focuses on automated class scheduling, attendance tracking, and class configuration. These core features were selected to address immediate needs while laying the foundation for future enhancements that will further optimize SWS's business processes

CM1: Class Scheduling

Description - MH

As a client, I want to browse available group classes, view trainer availability, and schedule my training sessions based on my preferred time slots.

Acceptance Criteria, Extensions, Tasks

- CM1.0: Client must have an account registered and assigned as a "Client" role before accessing scheduling. *MH*
- CM1.1: System displays a list of available classes, including date, time, trainer, and capacity. *MH*
- CM1.2: Clients can sign up for a class if it has not reached maximum capacity. *MH*
- CM1.3: Clients can view their currently scheduled classes in a dedicated "My Classes" section. *SH*
- CM1.4: Clients can cancel a scheduled class at least 4 hours before start time. *CH*

CM2: Class Configuration

Description - SH

As an owner, I want to configure class properties such as occupant capacity, trainer assignments, session durations.

Acceptance Criteria, Extensions, Tasks

- CM2.0: Owner must have an account registered and assigned as Owner. *MH*
- CM2.1: Owners can access class configuration settings (occupant capacity, trainer assignments, session durations) through the admin panel. *MH*
- CM2.2: Displays participant size, time, availability, and trainer. *MH*
- CM2.3: Owner can edit any property related to a class on a specific class. *SH*

- CM2.3a: Owner cannot leave a required prompt empty. *SH*
- CM2.4: Owner can delete any scheduled class. *MH*
- CM2.4a: If a class is deleted, all registered clients will notice the change in their dashboard. *SH*

CM3: Class Management

Description - MH

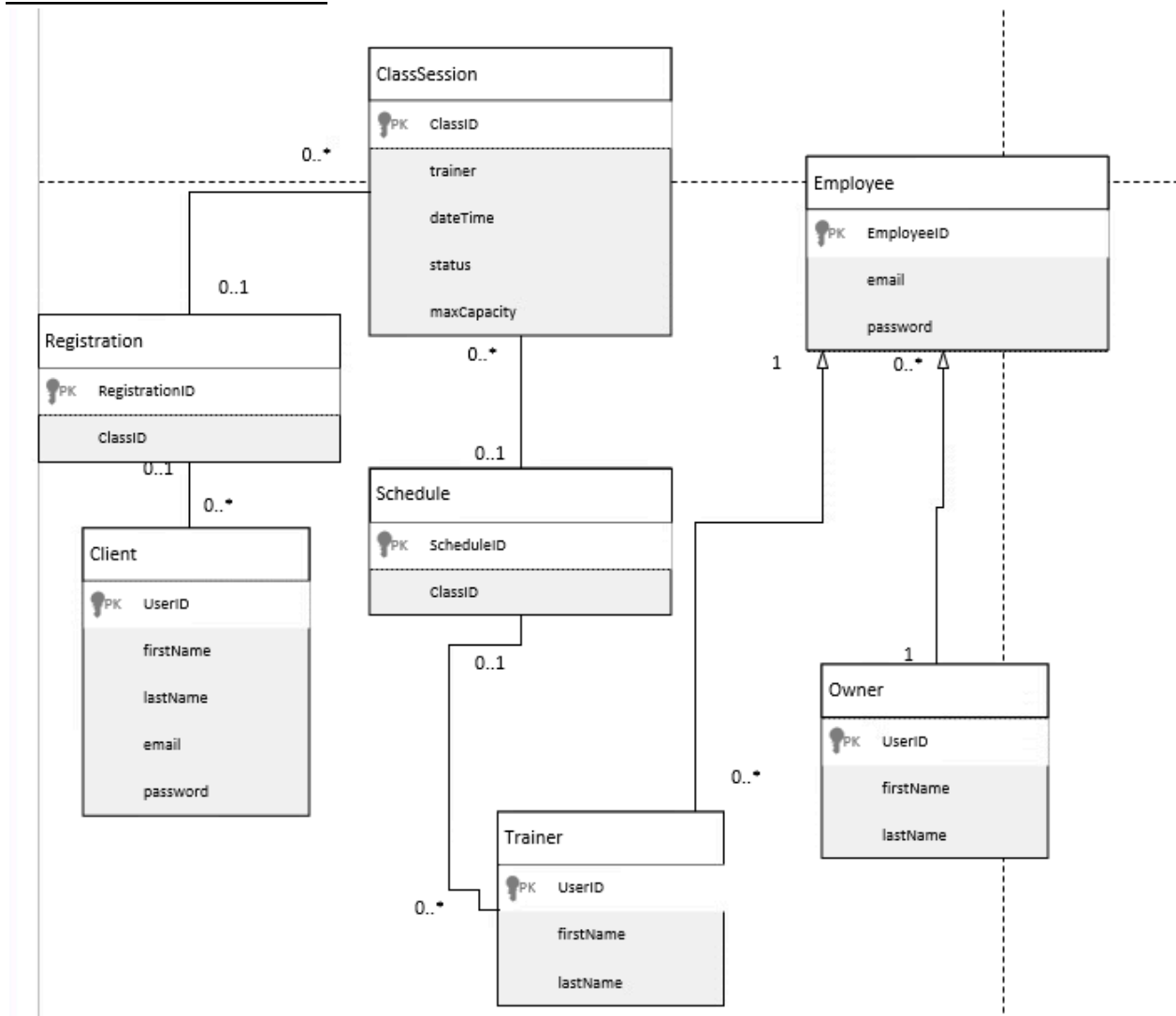
As a trainer, I want to create and manage group classes by setting schedules, updating class details, and tracking client attendance.

Acceptance Criteria, Extensions, Tasks

- CM3.0: Trainer must have an account registered and assigned as a "Trainer" role. *MH*
- CM3.1: Trainer can create a class with class name, date and time, capacity.. *MH*
- CM3.2: Trainer can configure class details time, capacity, and name. *SH*
- CM3.2a: Trainer can only edit class sessions they have created. *MH*

Data Requirements

Domain Class Model



Data Dictionary

Field Name	Data Type	Field Length	Description	Constraints
UserGUID	string	20	User ID, autogenerated	Not null
First_Name	string	20	First name of user	Not null
Last_Name	string	20	Last name of user	Not null

System Requirements Specification for Training Class

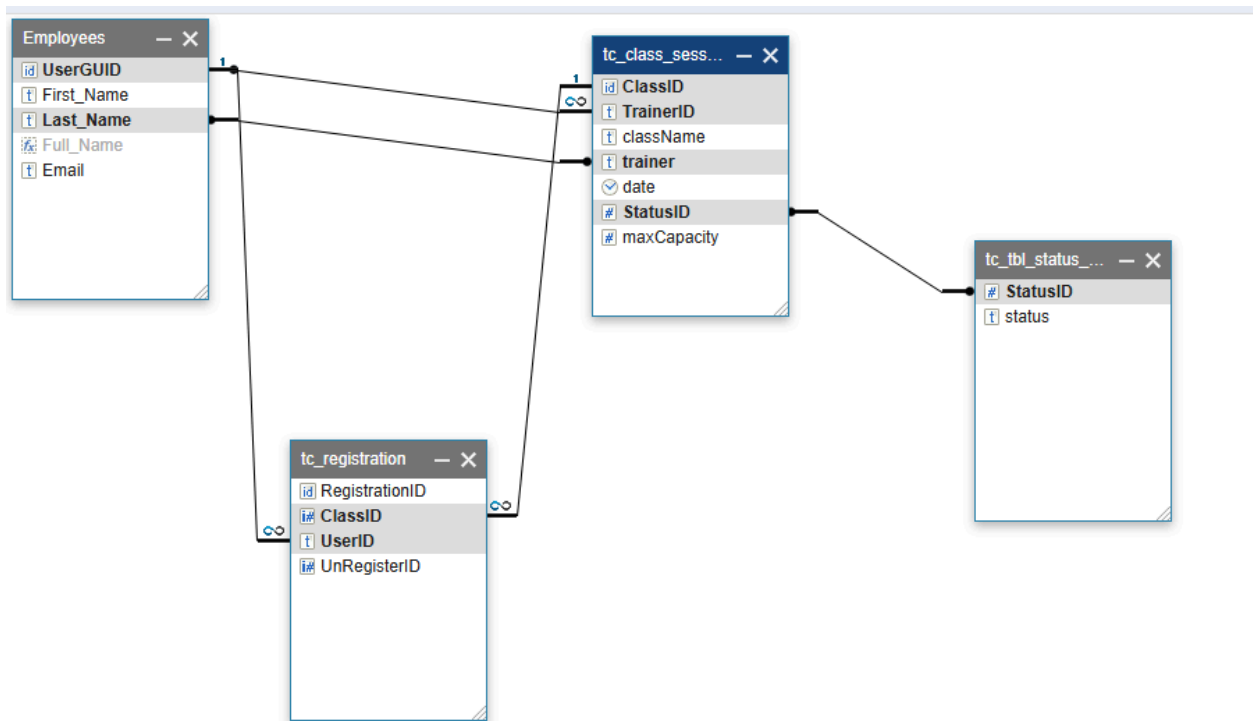
Email	string	30	Email of user	Not null
password	string	30	Password of user	Not null
ClassID	string	20	Class ID, auto number	Not null
className	string	25	Class name	Not null
trainer	string	20	Assigned trainer for a class session.	Maximum 1
dateTime	DateTime	20	Date and time of a class session.	Not null
statusID	string	20	Open or Full based on sign-ups.	Not null
maxCapacity	integer	2	Maximum number of sign-ups	Not negative
TrainerID	string	20	TrainerID for trainers	Not null
UserID	string	20	UserID for class signups	Not null
RegistrationID	string	20	Registration ID, auto generated	Not null

Reports

- **Class Schedule Report:** Displays a list of all upcoming class sessions, including details such as session ID, date, time, assigned trainer, and current occupancy. This is sorted by date and time.
- **Client Registration Summary:** Shows the list of clients registered for each class session to trainers or owners.
- **Trainer Class Roster:** Allows trainers to view a list of clients attending each of their assigned classes, helping with attendance tracking. This is sorted by date and time.

System Requirements Specification for Training Class

Caspio Relationships Table



User Interface Requirements

Application of Design Principles

Nielsen's 10 usability heuristics for user interface design are widely recognized principles that guide the creation of user-friendly digital products. Jakob Nielsen developed these principles which serve as a framework for evaluating and improving UI design by focusing on core aspects such as user control, error prevention, and visual clarity. According to Nielsen Norman Group (2025), these heuristics are not rigid rules but rather general principles that designers can apply to ensure a positive user experience. In this section, Nielsen's heuristics will be examined in relation to the UI of our application, with specific references to the wireframes and navigation diagrams provided in the appendix.

This heuristic ensures that users are always informed about what is happening. As demonstrated in Appendix 3: View Classes, all scheduled and active classes are clearly displayed for users, providing immediate feedback on the current system state. Real-time updates on class availability and changes are reflected through this screen.

Users are provided with multiple levels of control. Clients can easily join or leave classes via a single tap, as shown in (Appendix 3: Client. Trainers can instantly edit or delete their created classes (Appendix 3: Trainer), and owners maintain oversight with full access to modify any class (Appendix 3: Owner). These features prevent users from feeling trapped in unintended actions.

Efforts to prevent errors begin with clear form validations. If a user attempts to save a class without required information, the system displays a non-intrusive warning explaining what the problem is (Appendix 3: Owner). This proactive error prevention reduces user frustration and ensures integrity.

The interface prioritizes visible options over memory reliance. Once a user accesses the "My Classes" section, all available actions are immediately visible (e.g., Join, Leave, Edit), eliminating the need to remember hidden commands (Appendix 3: Client).

Unnecessary elements have been stripped away, and each screen is focused on core functionality. In Appendix 3: Client, the home screen presents a clean layout with only the essential buttons for navigation and if an account needs to be created, avoiding overwhelming users with options.

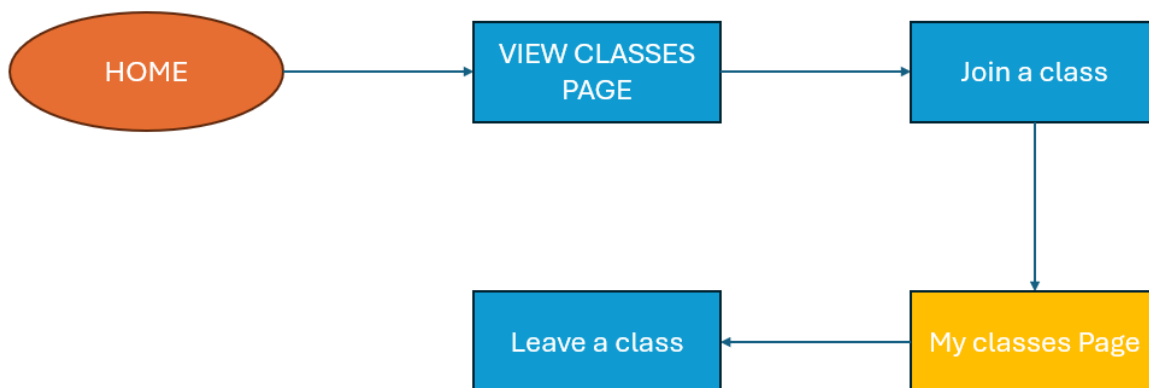
Whenever an issue occurs the system provides a clear message indicating the problem and how to resolve it (Appendix 3: Client). These messages will use plain language and avoid technical jargon.

System Requirements Specification for Training Class

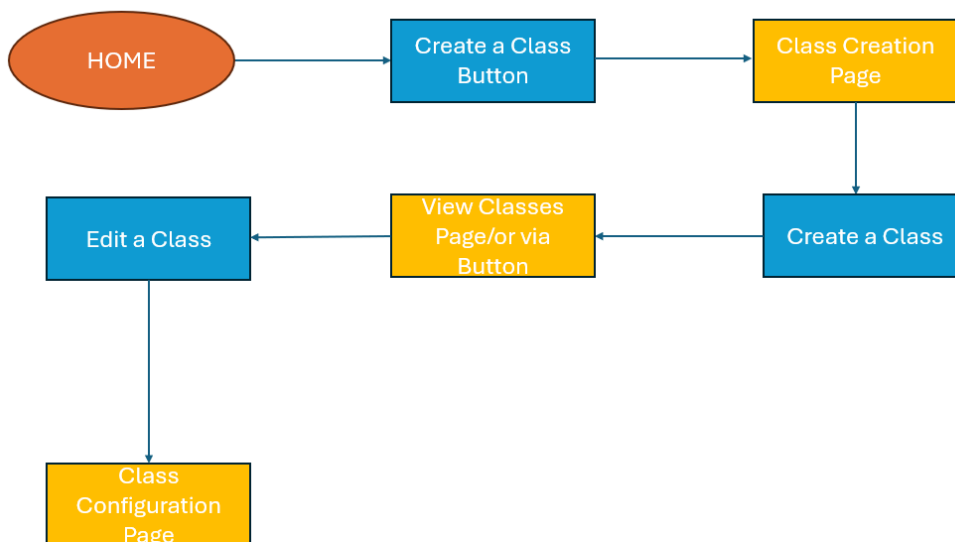
Overall, the user interface of our application demonstrates adherence to Nielsen's usability heuristics. By prioritizing visibility, error prevention, and user freedom, we have created an experience that is both functional and user-friendly. These heuristics provided a foundational lens through which we evaluated and improved our design, ensuring that users across all roles can interact with the system efficiently and confidently.

Navigation

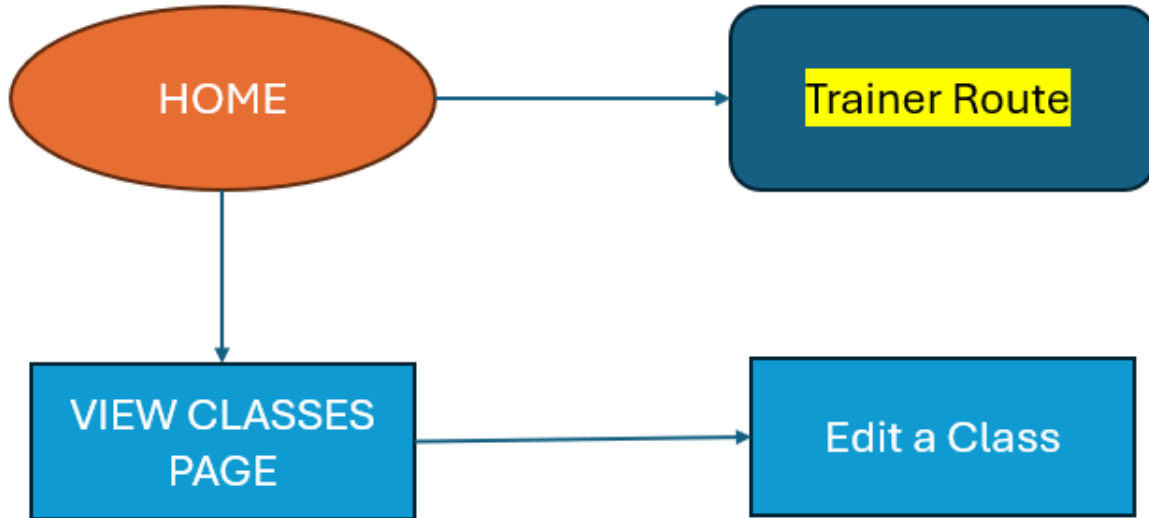
CLIENT:



TRAINER:



OWNER:



User Role Access

Under “View Classes”, clients can see this and join a class, trainers can see this and create a class, owner’s are free to do as they wish.

Under “My Classes”, you can only see classes you are enrolled in. Therefore, trainer’s can create, update, or delete their own classes. As Owner’s have the ability to create classes, too they can do the same as trainer’s.

<u>View Classes</u>	Client	Trainer	Owner
CREATE		✓	✓
READ	✓	✓	✓
UPDATE			✓
DELETE			✓
<u>My Classes</u>	Client	Trainer	Owner
CREATE		✓	✓

READ	✓	✓	✓
UPDATE		✓	✓
DELETE	✓	✓	✓

Modifications to Original Design

- Due to limitations in Caspio, the occupant list was scratched due to time constraints, however, it does not interfere with this prototype (Minimum Viable Product), it is possible to do and can be done at a later time, as a separate use case.
- Class registration for the client via *View Classes* was moved to a sign-up form as an input due to Caspio limitations which were unable to be mitigated.
- Trainers are unable to edit their classes from the view classes screen, it was moved to their *My Classes* screen where they can only see, edit, and delete their own classes.
- Owners are moved to the edit screen when they select a class where they can cycle through every class session to edit or delete.
- For Clients, they are able to select each of their signed-up classes for more information in the *My Classes* section and drop it from there too, however, for the right screen to update it must refresh the screen due to a Caspio filtering bug.
- Error messages regarding joining full classes are not present.

Nonfunctional Requirements

Usability

The application prioritizes a user-friendly experience through ease of use, error prevention, and efficient interactions. The interface uses familiar language, clear labeling, and intuitive navigation to reduce the learning curve for new users. Forms include input validation to help users avoid mistakes, and clear error messages guide recovery when issues occur.

Interaction flows are kept short and logical, allowing users—especially trainers and owners—to complete common tasks like editing or managing classes with minimal steps. Accessibility is considered through readable font sizes, high-contrast visuals, and keyboard-friendly navigation, though advanced accessibility features are not currently implemented.

We also designed for ergonomics, ensuring buttons are large and spaced well for touchscreens. While not all features are necessary for every user type, the app balances simplicity with role-based control to keep the experience smooth for clients while still powerful for trainers and owners.

Performance

The app is designed for speed and responsiveness. Key actions like loading pages, searching for classes, and submitting forms should complete within 1–2 seconds. Real-time updates upon refresh (like class changes) should appear quickly, and all interactions should feel smooth, even on mobile devices.

Security

The application includes role-based access control to ensure users can only access features relevant to their role (Client, Trainer, or Owner). Trainers can only view and edit their own classes, while Owners have full access to all class data. User data is stored securely and access to sensitive actions (like deleting classes) is restricted and confirmed to prevent misuse. All user sessions require authentication, and no sensitive data is displayed or stored in public-facing views.

Reliability

The application is expected to maintain high reliability by minimizing downtime and ensuring consistent performance across all user roles. Availability is targeted at 99.9% uptime (Caspio), allowing users to reliably access and interact with the system during all standard usage periods.

Scalability

The system is built to scale efficiently as the user base grows. The back-end architecture supports horizontal scaling to accommodate increased demand, especially during high-traffic periods like class registrations.

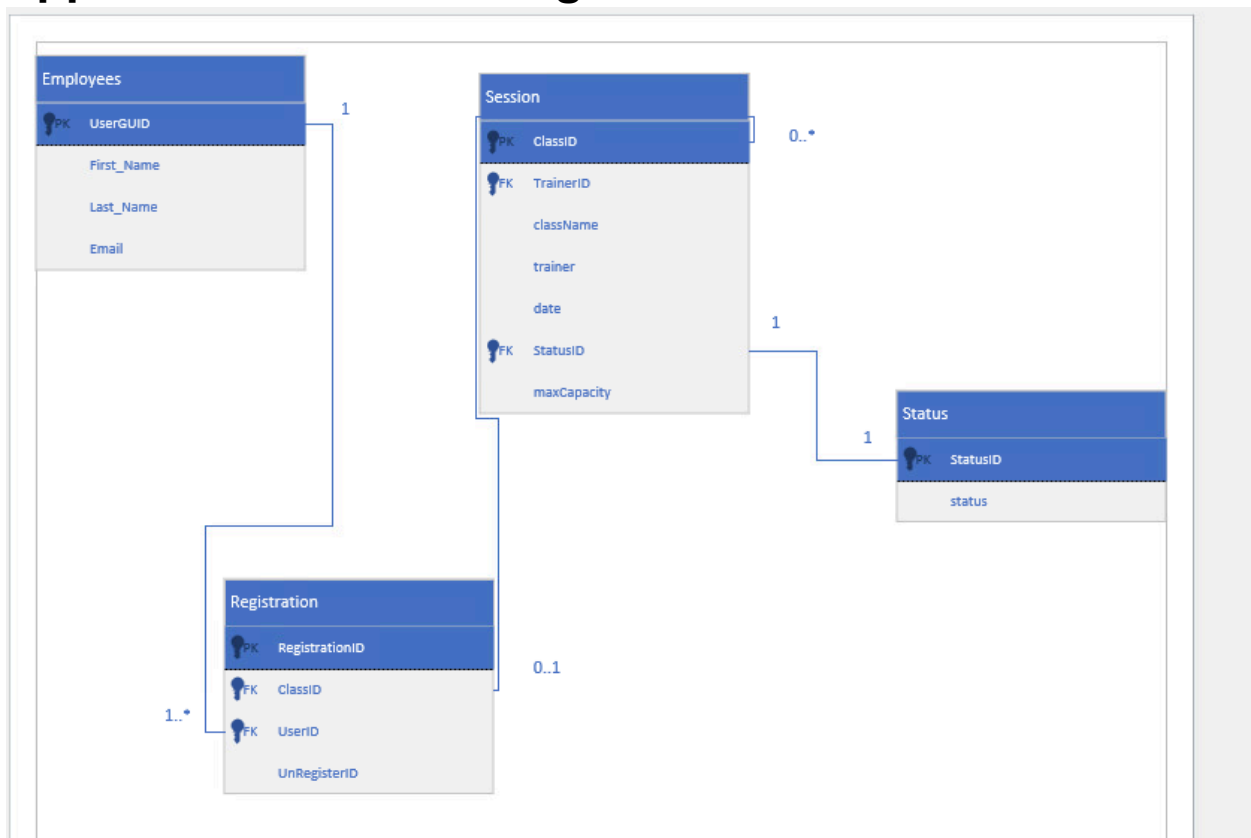
Reusability

Core components such as user role management, class scheduling, and form handling are designed as modular and reusable code blocks. These components can be reused or extended in future versions of the app or in new applications with similar functionality, reducing development time and improving maintainability.

References

- Nielsen Norman Group. (2025). *10 usability heuristics for user interface design*.
<https://www.nngroup.com/articles/ten-usability-heuristics/>
- OpenAI. (2025). ChatGPT (Mar 14 version) [Large language model].
<https://chat.openai.com/chat>

Appendix 1: Domain Diagram



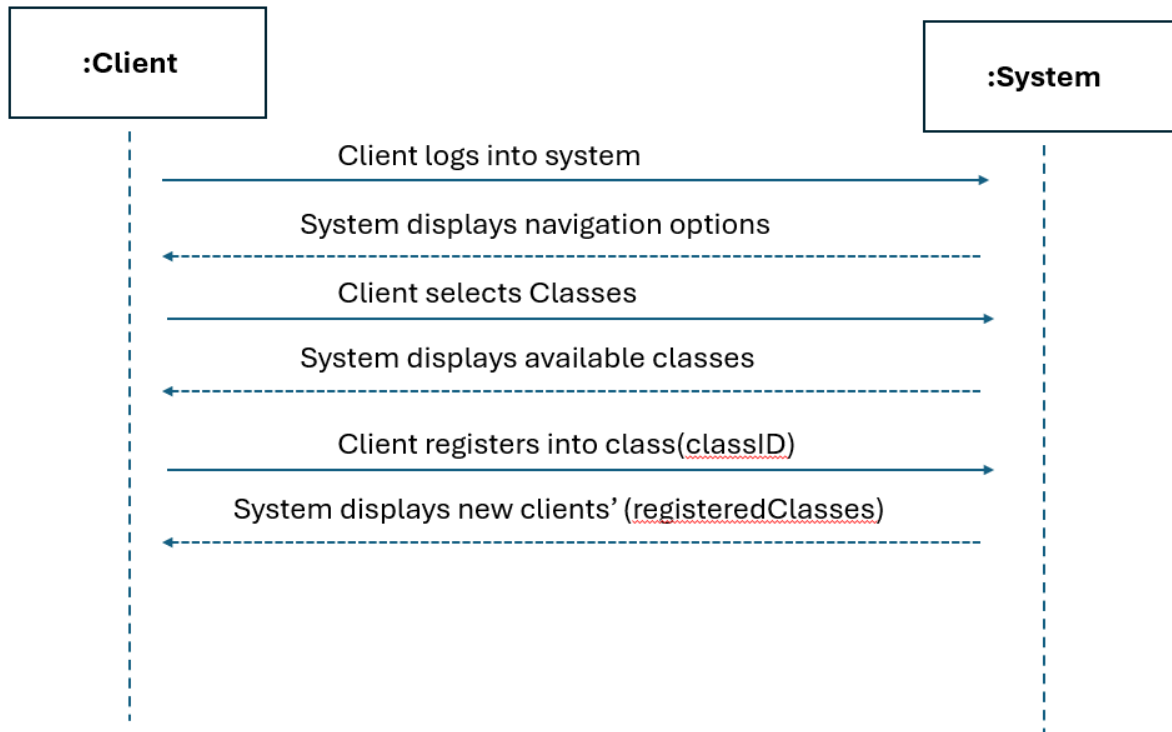
Appendix 2: Use Case Documentation

USE CASE NARRATIVE 1 - CLASS SCHEDULING

Use Case Name:	Class Scheduling
Triggering event:	A client wants to schedule a group class session.
Story or Brief Description:	A registered client browses available group classes, selects a preferred session, and successfully schedules their training session.
Primary actor:	Client
Preconditions :	A client must have internet access, an active account, and be logged into the system, the class must have available slots.
Postconditions (Success Guarantee):	The client successfully registers for the selected class, the scheduled class appears in the client's "My Classes" section.
Flow of Activities (Main Success Scenario:	<ol style="list-style-type: none"> 1. Client logs into the system 2. System displays available classes 3. Client navigates to the <u>View Classes</u> section 4. Client selects a preferred class. 5. Client confirms class booking. 6. Client receives class under their "My Classes" section. 7. They can view their registered classes and drop them.
Exception conditions/ Alternative Scenarios:	<p>3a. Client selects a full class</p> <p>4a. Client attempts to schedule a class they are already in</p> <p>5a. Client has a conflicting class time.</p>
Acceptance criteria	<ul style="list-style-type: none"> • The system displays an up-to-date list of available classes. • Clients can successfully register if all pre-conditions are met. • Clients cannot register for a full class. • The class appears in the "My Classes" section after success.

	<ul style="list-style-type: none">• The system prevents double-booking or scheduling conflicts.
Related models, mockups:	SSD Diagram 1

SSD 1 - CLASS SCHEDULING



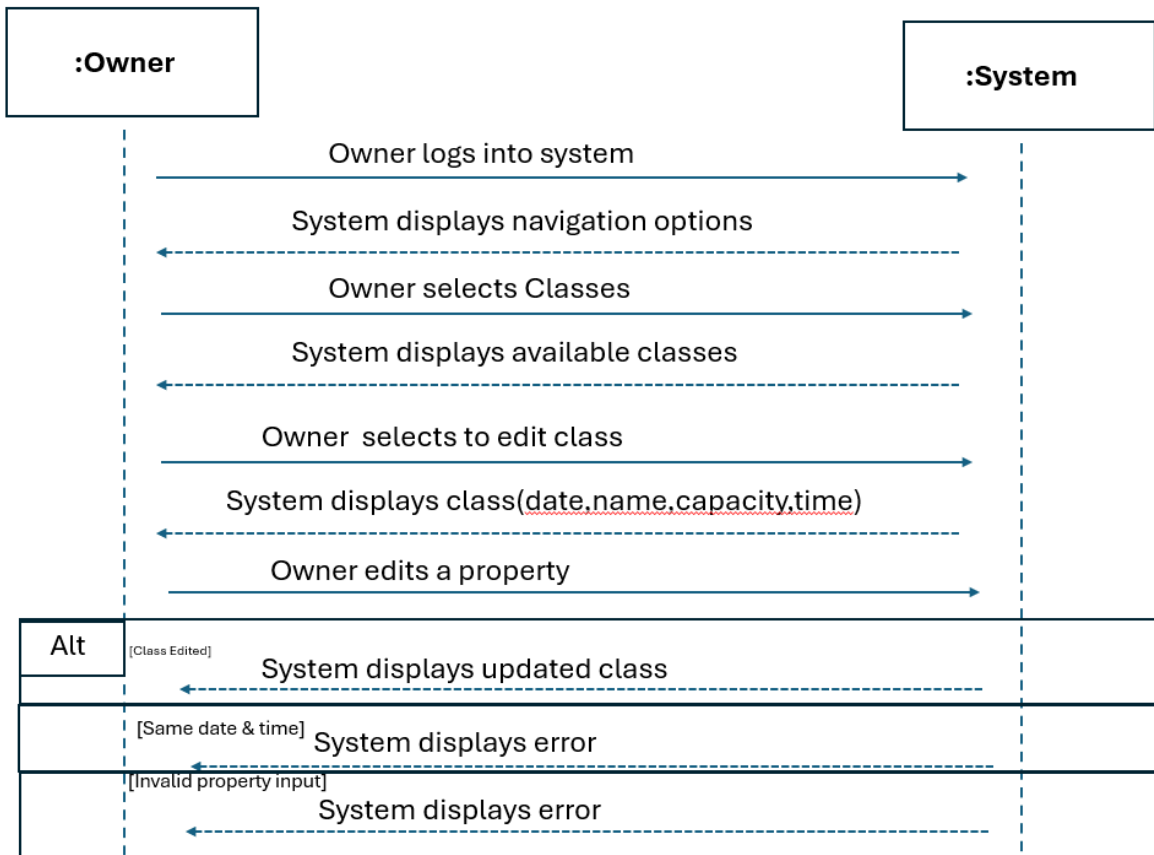
USE CASE NARRATIVE 2 - CLASS CONFIGURATION

Use Case Name:	Class Configuration
Triggering event:	An owner wants to configure or track class sessions.
Story or Brief Description:	An owner administrates group classes by setting schedules, updating class details, and tracking available classes.
Primary actor:	Owner
Preconditions :	<ul style="list-style-type: none"> The owner must have an account registered and assigned as a Owner. At least one class category or type must exist in the system.
Postconditions (Success Guarantee):	<ul style="list-style-type: none"> The system successfully updates the class with the owner's configurations. The owner can track and manage attendance records.
Flow of Activities (Main Success Scenario:	<ol style="list-style-type: none"> Owner logs into the system System displays available navigation modules Owner navigates to the <u>View Classes section</u> System displays available classes Owner selects the option to <u>Edit a Class</u> System prompts the owner to edit class details (name, schedule, duration, max capacity) Owner confirms the details and submits the class update request. System integrates the updates into the respective class.
Exception conditions/ Alternative Scenarios:	<p>4a. Owner enters incomplete details: The system prompts the trainer to fill in required fields.</p> <p>5a. Class schedule conflicts with an existing class: The system notifies the Owner of the conflict..</p>
Acceptance criteria	<ul style="list-style-type: none"> A trainer can also use this to edit or remove their own class.

System Requirements Specification for Training Class

	<ul style="list-style-type: none">• The system must prevent duplicate or overlapping class schedules.• The system must allow real-time tracking of client attendees.• The owner can create, update, delete, or modify any class.
Related models, mockups:	SSD Diagram 2

SSD 2 - CLASS CONFIGURATION



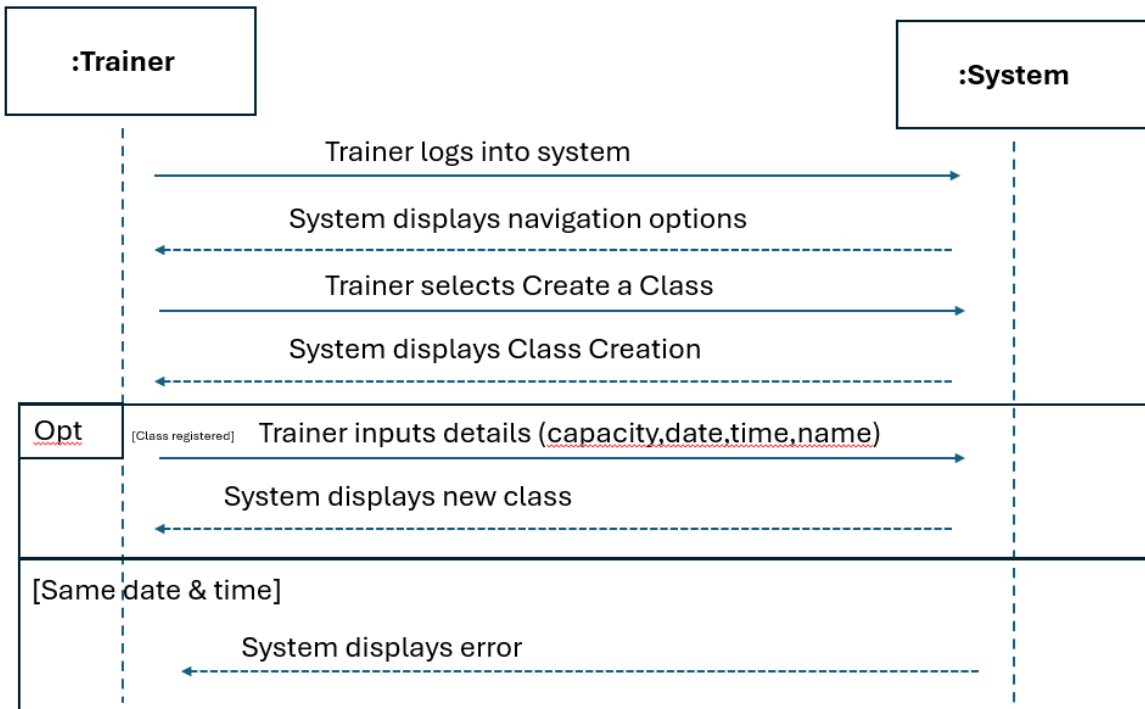
USE CASE NARRATIVE 3 - CLASS MANAGEMENT

Use Case Name:	Class Management
Triggering event:	A trainer wants to create, manage, or track attendance for a group class.
Story or Brief Description:	A trainer creates and manages group classes by setting schedules, updating class details, and tracking client attendance.
Primary actor:	Trainer
Preconditions :	<ul style="list-style-type: none"> • The trainer must have an account registered and assigned as a Trainer. • At least one class category or type must exist in the system.
Postconditions (Success Guarantee):	<ul style="list-style-type: none"> • The system successfully creates or updates the class with the trainer's configurations. • Clients can view and enroll in the class if availability exists. • The trainer can track and manage attendance records.
Flow of Activities (Main Success Scenario:	9. Trainer logs into the system 10. Systems displays navigation options 11. Trainer selects the option to <u>Create a New Class</u> 12. System prompts the trainer to input class details (name, schedule, duration, max capacity) 13. Trainer confirms the details and submits the class creation request. 14. System integrates the class into the <u>Classes</u> section. 15. Clients can now view and register for the class.
Exception conditions/ Alternative Scenarios:	4a. Trainer enters incomplete details: The system prompts the trainer to fill in required fields.

System Requirements Specification for Training Class

	<p>6a. Class schedule conflicts with an existing class: The system notifies the trainer of the conflict.</p> <p>6a. Trainer attempts to edit another trainer's class: The system restricts access and displays an error message.</p>
Acceptance criteria	<ul style="list-style-type: none">• The trainer must be able to create, update, and delete their own classes.• The system must prevent duplicate or overlapping class schedules for the same trainer.• The system must allow real-time tracking of client attendees.• The trainer must not be able to edit or remove classes created by other trainers.
Related models, mockups:	SSD Diagram 3

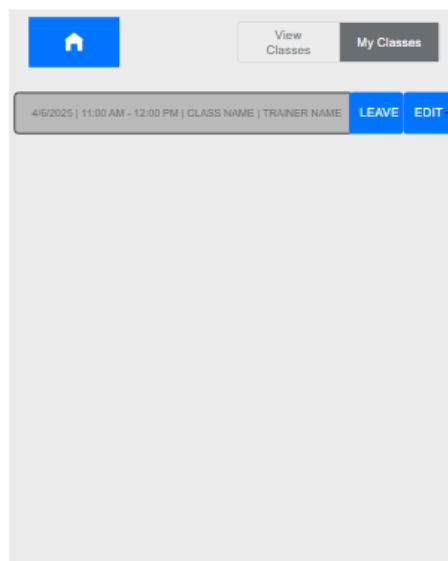
SSD 3 - CLASS MANAGEMENT



Appendix 3: UI Documentation

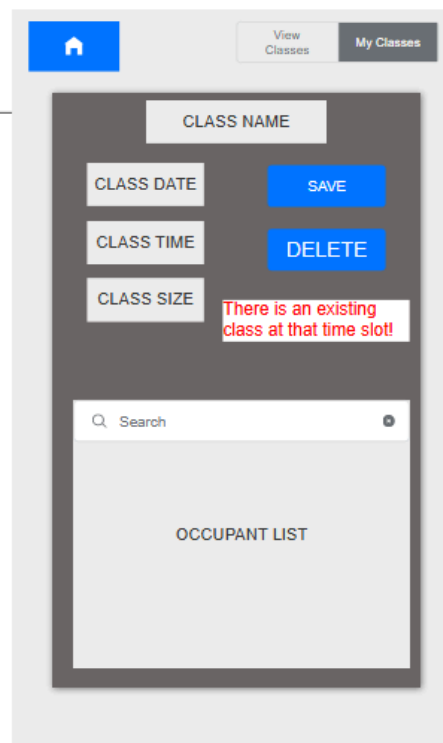
(Screen Layouts)

Screen 1



MY CLASSES

Screen 2



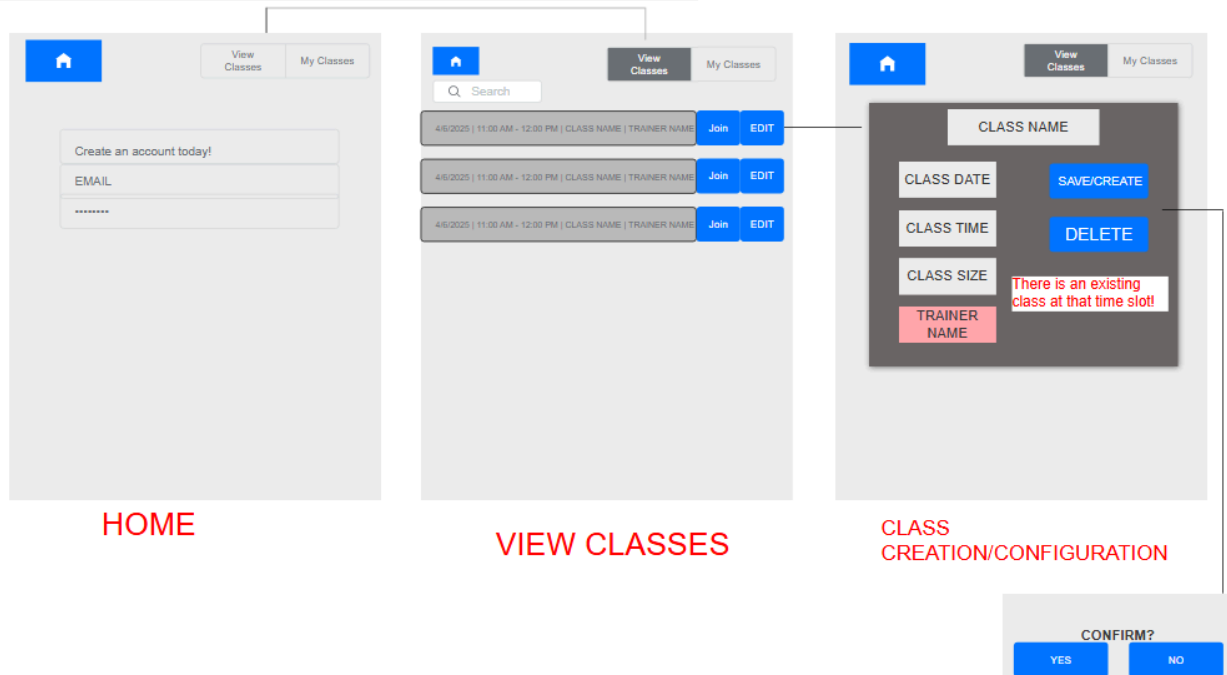
EDIT A SPECIFIC
OWNED CLASS

Screen 3

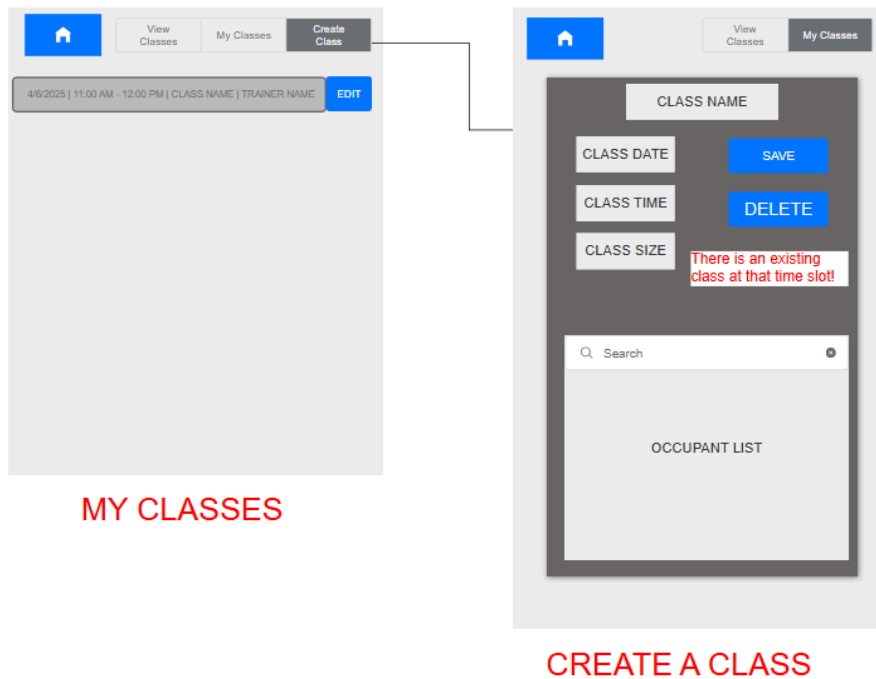
System Requirements Specification for Training Class

(Process Navigation Map)

OWNER:



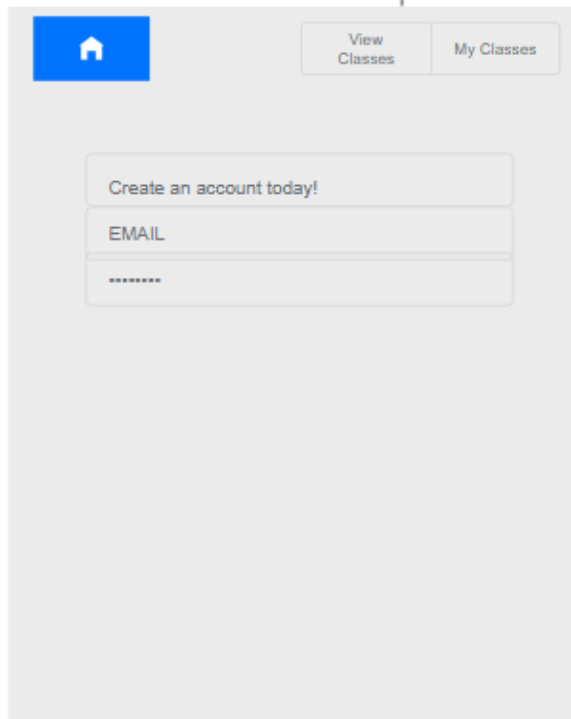
TRAINER:



Same screen for Editing it

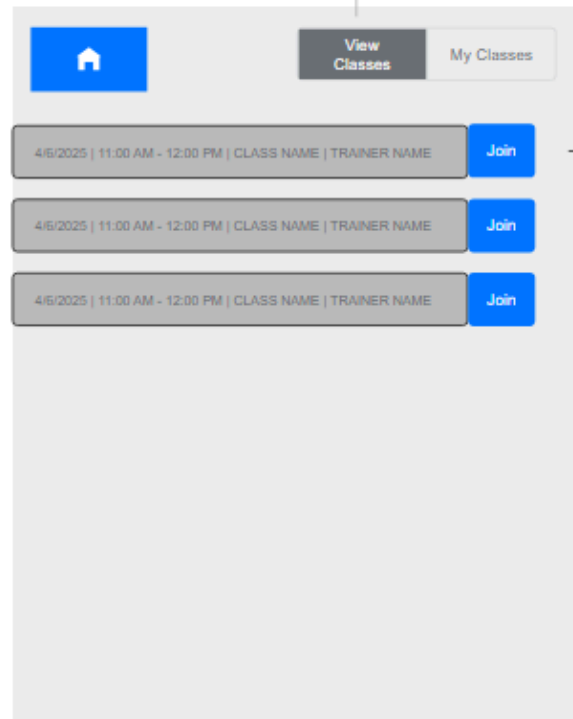
System Requirements Specification for Training Class

CLIENT:



Home page UI mockup. The header features a blue home icon, a 'View Classes' button, and a 'My Classes' button. The main content area contains a 'Create an account today!' prompt, an 'EMAIL' input field, and a password field with asterisks.

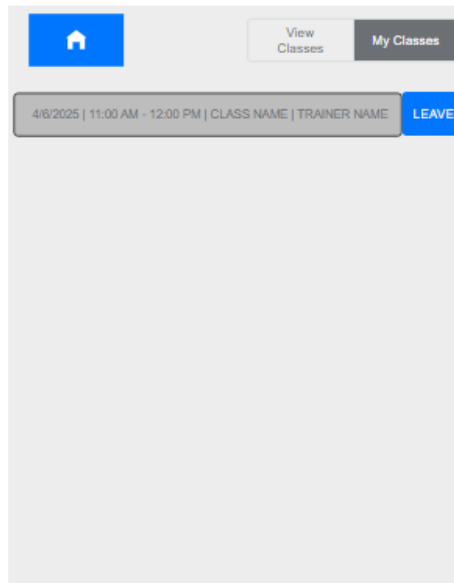
HOME



View Classes page UI mockup. The header features a blue home icon, a 'View Classes' button, and a 'My Classes' button. The main content area displays a list of three class entries, each with a date, time, class name, trainer name, and a 'Join' button.

VIEW CLASSES

cas



My Classes page UI mockup. The header features a blue home icon, a 'View Classes' button, and a 'My Classes' button. The main content area displays a list of one class entry with a date, time, class name, trainer name, and a 'LEAVE' button.

MY CLASSES

Appendix 4: User Guide - Training Class

[Public Access Link](#)

[Client Access Link](#)

Login Credentials:

[client@gmail.com](#)

Sammy3313!

1. View all available classes
2. Enter ClassID in register form to register for it (due to limitations had to do it this way)
3. My Classes
4. Can select which class to view more information about it
5. Can drop out of the classes by mouse-over over Signed-up Classes.
6. Due to a Caspio filtering bug, need to refresh page to fix right-sided section.

[Trainer Access Link](#)

Login Credentials:

[trainer@gmail.com](#)

Sammy3313\$

1. Type in required fields to create a class and submit.
2. My Classes
3. Can view, edit, and delete only classes created by **you**.

[Owner Access Link](#)

Login Credentials:

[owner@gmail.com](#)

Sammy3313\$

1. Hover mouse-over class, can delete or update it.
2. Create Class
3. Type in required fields to create class and submit
4. My Classes
5. Can update whichever values you want, ANY class.
6. View Classes
7. Can do step 1 and repeat.