המכללה האקדמית להנדסה בכרמיאל

בראודה

Capstone Project Phase A

Project No. 24-1-D-33

Sports club management

Supervisor: Prof. Zeev Barzily

Name	ID	Email
Ahmed Mahamid	315038208	Ahmed.Mahamid@e.braude.ac.il

Table of Contents

Abstract

The efficient management of sports clubs is crucial for providing high-quality services, fostering member engagement, and ensuring operational sustainability. However, traditional approaches often face challenges such as inefficient course administration, instructor management, and appointment handling processes. This project aims to develop a comprehensive and user-friendly sports club management system to address these challenges and streamline operations.

The proposed system will incorporate a smart booking system that allows members to easily register for courses and manage appointments through a user-friendly interface. It will provide detailed information about each course, including schedules, costs, and instructor profiles, enabling members to make informed decisions. Additionally, the system will facilitate member feedback, enabling continuous improvement of the club's offerings.

For instructors, the system will offer dedicated tools to efficiently manage their schedules, specialties, and course details, enhancing the quality of instruction and timetable organization. Administrators will have access to powerful tools for managing users, instructors, facilities, and courses, enabling informed decision-making and strategic planning.

The sports club management system aims to revolutionize the club's operations, fostering operational efficiency, user adoption, revenue growth, data security, and maintainability. By providing a seamless and engaging experience for all stakeholders, the system will contribute to the overall success and growth of the sports club.

1. Introduction

In today's fast-paced world, sports clubs play a crucial role in promoting physical activity, fostering a sense of community, and providing a platform for personal growth and development. However, managing the intricate operations of a sports club can be a daunting task, often plagued by inefficiencies and manual processes. This is where our innovative sports club management system comes into play, offering a comprehensive and user-friendly solution to streamline operations and enhance the overall experience for members, instructors, and administrators alike.

The system aims to address the challenges faced by traditional sports club management approaches, and difficulties in managing course schedules, instructor availability, and member registrations using manual processes, leading to inefficiencies and potential conflicts.

The system leverages modern technologies and incorporates feedback from stakeholders to create a platform tailored to the unique requirements of sports clubs. By integrating features such as course administration, facility management, reservation systems, and financial management, I aim to streamline operations and enhance efficiency across all aspects of the club's operations.

For members, the system offers a user-friendly interface that allows them to easily browse and register for courses, view their training progress, schedule sessions with instructors, and provide feedback through opinion reports. Additionally, members can receive timely notifications and updates, ensuring they stay informed about important an announcements and events.

Instructors benefit from dedicated tools that enable them to manage their schedules, specialties, and course details with ease. This not only enhances their productivity but also ensures that members receive high-quality instruction and a consistent experience.

Administrators and managers gain access to powerful tools for managing users, instructors, facilities, and courses. They can efficiently handle tasks such as adding, editing, and removing users, instructors, and courses, as well as managing hall availability and bookings. Furthermore, the system provides valuable insights and data for informed decision-making and strategic planning.

By addressing the limitations of existing solutions and incorporating best practices from related research, my sports club management system aims to revolutionize the way sports clubs operate, fostering a seamless and engaging experience for all stakeholders.

2 Who the stakeholders are and how the solution will help them:

- **Members**:
  - **Interest**: They will have easy and user-friendly access to managing their sports activities at the club.
  - **Efficiency**: receive comprehensive information about the courses.
- Instructors:
  - **Interest**: They will receive efficient tools for daily management to improve the quality of instruction.
- Managers/Administrators:
  - **Interest**: They will see an upgrade in the ability to manage and access-maintained details in the system.
  - **Efficiency**: The ability to efficiently manage club operations and receive strategic data for informed decision-making.

This advanced solution I aim to offer will enhance the user experience for all stakeholders in the sports club.

3 Background and Related Work

The management of sports clubs has traditionally been a challenging task, often plagued by inefficiencies and manual processes. Studies have highlighted the difficulties in managing course schedules, instructor availability, and member registrations using traditional approaches, leading to potential conflicts and suboptimal experiences.

The increasing demand for efficient and user-friendly solutions has driven the need for digital transformation in sports club management. Researchers have emphasized the importance of providing seamless experiences for members, enabling them to easily access information, register for courses, and manage their schedules.

3.1 PHP

PHP (Hypertext Preprocessor) is a widely used open-source scripting language that is especially suited for web development and can be embedded into HTML. PHP scripts are executed on the server, and the result is returned to the browser as plain HTML. PHP has been a popular choice for web developers due to its ease of use, speed, and extensive community support.

PHP's popularity is underscored by its usage statistics; according to W3Techs [1], PHP is used by 76.2% of all websites with a known server-side programming language. One of the critical advantages of PHP is its ability to handle dynamic content and interaction with databases, which is essential for creating a robust sports club management system. The performance and flexibility of PHP make it an excellent choice for backend development.

3.2 Laravel

Laravel is a PHP framework designed to make the development process a pleasing one for the developer while maintaining application functionality. Laravel attempts to take the pain out of development by easing common tasks used in most web projects, such as authentication, routing, sessions, and caching.

Laravel's expressive, elegant syntax and its ability to handle complex web applications securely and at a significantly faster pace than other PHP frameworks make it a popular choice. According to a survey by JetBrains [2], Laravel is the most popular PHP framework, used by 61% of PHP developers. Laravel’s built-in tools and libraries provide powerful features that simplify tasks such as database migrations, ORM (Object-Relational Mapping), and unit testing.

3.3 MySQL

MySQL is an open-source relational database management system (RDBMS) based on Structured Query Language (SQL). It is renowned for its reliability, ease of use, and performance. MySQL is the preferred choice for database management in many web applications due to its robust support for data integrity and transactions.

MySQL is highly compatible with PHP, which makes it a natural choice for developing a sports club management system. The combination of PHP and MySQL provides a solid foundation for building dynamic and data-driven web applications. According to the DB-Engines Ranking [3], MySQL is the second most popular database management system, underscoring its widespread adoption and reliability.

In conclusion, the selection of PHP, Laravel, and MySQL for the development of a sports club management system is backed by their widespread use, robust performance, and extensive community support. These technologies together provide a scalable, efficient, and developer-friendly environment that is well-suited to address the challenges in sports club management.

4 Expected Achievements

4.1 Streamlined Operations

The system aims to streamline various operations within the sports club, including:

Course administration: Enabling admins to efficiently manage courses, including creation, modification, and deletion.

Facility management: Providing functionality for admins to manage halls, availability, and bookings seamlessly.

Instructor management: Allowing admins to manage instructors, including adding, editing, and removing them with ease.

By automating and centralizing these processes, the system will enhance operational efficiency and reduce the workload on administrators.

4.2 Improved Member Experience

The system is designed to provide an enhanced experience for members, including:

User-friendly registration and account management processes.

Seamless course booking and reservation system, ensuring no conflicts.

Ability to view and track training progress.

Scheduling training sessions with instructors.

Receiving timely notifications and important announcements.

Providing feedback through opinion reports after training sessions.

These features will empower members to take control of their sports activities, fostering engagement and satisfaction.

4.3 Scalability:

PHP is a lightweight and efficient server-side scripting language, capable of handling many concurrent users and requests without significant performance degradation [4].

MySQL is a robust and scalable relational database management system (RDBMS) that can handle large amounts of data and support high-traffic applications [5].

Both PHP and MySQL have a wide range of caching mechanisms and optimization techniques that can be employed to improve performance and scalability as the system grows.

PHP and MySQL can be easily integrated with load balancing and clustering solutions, allowing the system to scale horizontally by distributing the workload across multiple servers.

4.4 Compatibility:

PHP is a cross-platform language, ensuring the sports club management system can be deployed on various operating systems, including Windows, Linux, and macOS.

MySQL is also cross-platform and can run on multiple operating systems, ensuring compatibility across different environments.

PHP has excellent support for web development, making it compatible with various web browsers and devices, including desktops, laptops, tablets, and smartphones.

Both PHP and MySQL have large and active communities, ensuring ongoing support, updates, and compatibility with emerging technologies and standards.

The Laravel framework, which will be used with PHP, is designed to be compatible with modern web development practices and follows industry standards, ensuring compatibility with various tools and libraries.

By leveraging PHP, MySQL, and the Laravel framework, the proposed sports club management system can achieve the scalability and compatibility requirements outlined in the provided document, accommodating an increasing number of users, courses, and transactions without compromising performance, while also ensuring accessibility across various devices and web browsers.

5 Engineering process

5.1 Process

As the sole developer of this sports club management system project, I have adopted an iterative and incremental approach to the engineering process. This approach allows me to break down the development into manageable phases and make necessary adjustments along the way.

5.1.1 Requirement Gathering and Analysis:

To kick off the project, I met with the sports club managers to understand their pain points and gather requirements. During these meetings, I asked them about the daily challenges they face in managing the club and what they expect from a management system. Some key questions I asked included:

- What are the current processes for course administration, instructor management, and member registration?

- What are the major issues in these processes?

- What features or functionalities would you like to see in a new management system?

- How do you currently handle facility management, scheduling, and payments?

- What kind of reporting or analytics would be helpful for decision-making?

Based on their responses, I documented the functional and non-functional requirements, prioritizing them based on their importance and impact on the overall system.

5.1.2 Iterative Development Approach:

To address the requirements effectively, I have adopted an iterative development approach using the Agile methodology. This approach involves breaking down the project into smaller, manageable iterations or sprints, each focusing on delivering a specific set of features or functionalities.

5.1.3 The iterative process involves the following steps:

1. Planning: At the beginning of each iteration, I define the scope, prioritize the requirements, and create a detailed plan.

2. Design: Based on the requirements, I design the system architecture, database schema, and user interfaces for the features to be developed in the current iteration.

3. Development: Using PHP, Laravel, MySQL, CSS, Bootstrap/Tailwind CSS, HTML5, and JavaScript, I implement the planned features, following best practices and coding standards.

4. Testing: Throughout the development process, I conduct unit tests, integration tests, and user acceptance testing to ensure the quality and functionality of the developed features.

5.1.4 Constraints and Considerations:

As a solo developer, I need to be mindful of the following constraints and considerations:

1. Time Management: Effective time management is crucial to ensure the timely finish of each iteration.

2. Resource Allocation: As the sole developer, I need to allocate my time and effort judiciously across various tasks, such as coding, testing, and documentation.

3. Scalability and Maintainability: From the outset, I need to design the system with scalability and maintainability in mind, ensuring that it can accommodate future growth and changes without significant rework.

By following this iterative and incremental engineering process, leveraging modern technologies, and considering the constraints and considerations, I aim to deliver a high-quality, user-friendly, and efficient sports club management system that meets the stakeholders' requirements.

5.2 Product

5.2.1 Diagrams

5.2.1.1 Use case diagram

Actors:

Trainee: Represents a registered user with more privileges, such as booking classes and making payments., tracking participation, and paying.

Guest: Represents an unauthenticated user with limited access.

Trainer: Uses the website to view the courses they teach, participant details for those courses, and other course information.

Administrator: Uses the website to view and manage all information, including courses, trainers, facilities, users, and payments.

Guest:

Explore Courses: Browse a list of available courses.

View News and Activities: Stay updated with club news and upcoming events.

View Course Details: Get in-depth information on specific courses.

Sign Up: Create an account to become a member.

Register for Membership: Sign up to become a member of the sports club.

Contact Support: Reach out to the club for inquiries and support.

Register for Courses: Must sign up and become a member before registering for courses.

Member/Trainee:

View Courses: See a list of available courses, including information on subject, location, dates, price, and requirements.

View Course Details: See detailed information about a specific course, including description, syllabus, requirements, and equipment.

Register for Course: Choose a course and pay for participation.

Track Participation: See a list of the courses the member is participating in, including information on dates, times, and locations.

Cancel Participation: Unregister from a course.

Trainer:

Login: Enter username and password to access information.

View Courses: See a list of the courses the trainer teaches, including information on the subject, location, dates, and requirements.

View Course Details: See detailed information about a specific course, including description, syllabus, requirements, and equipment.

View Participant List: See a list of participants registered for each course the trainer teaches.

Update Course Details: Update information about a specific course, such as description, syllabus, requirements, and equipment.

Administrator:

Login: Enter username and password to access information.

Manage Courses:

Add Course: Create a new course, including setting subject, location, dates, price, and requirements.

Delete Course: Remove an existing course.

Update Course Details: Update information about an existing course, such as subject, location, dates, price, and requirements.

Manage Trainers:

Add Trainer: Create a new account for a trainer.

Update Trainer Details: Update account information for a trainer.

Delete Trainer: Delete a trainer's account.

Manage Facilities:

Add Facility: Create a new facility, including setting name, size, and equipment quantity.

Update Facility Details: Update information about an existing facility, such as name, size, and equipment quantity.

Delete Facility: Delete an existing facility.

Manage Users:

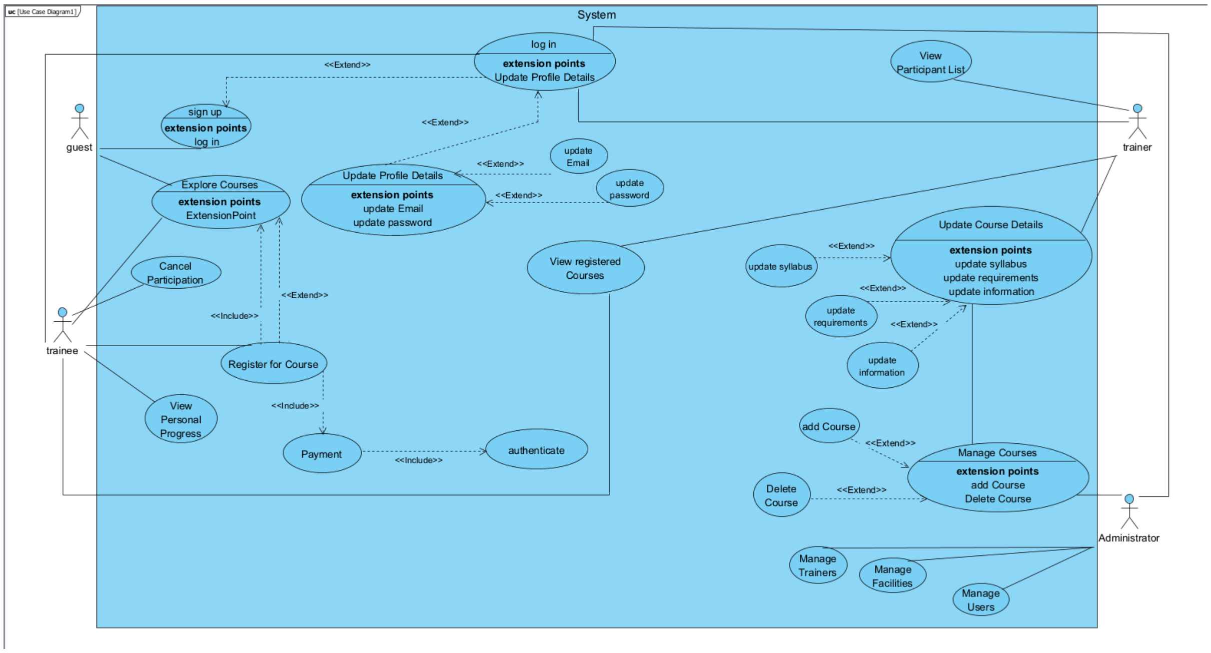
View User List: See a list of all users registered on the website.

Update User Details: Update account information for a user.

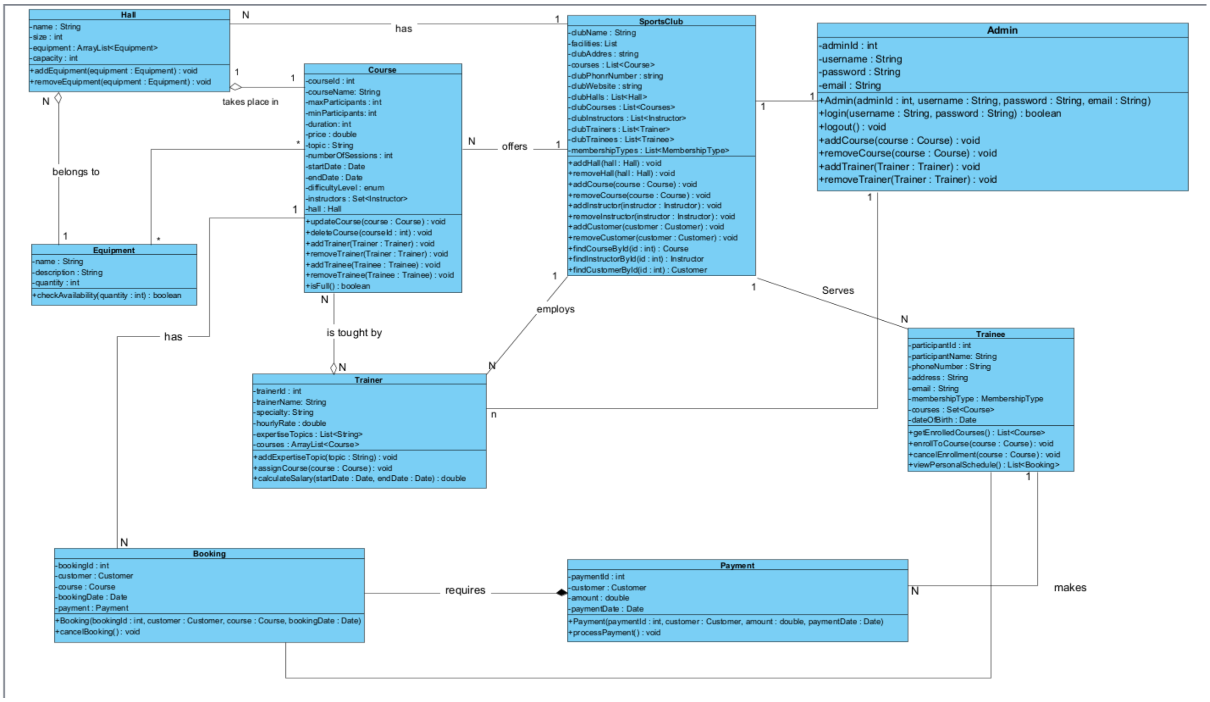
Delete User: Delete a user's account.

Manage Payments:

Use case diagram:



5.2.1.2 Class diagram:


Hall
-name : String
-size : int
-equipment : ArrayList<Equipment>
-capacity : int
+addEquipment(equipment : Equipment) : void
+removeEquipment(equipment : Equipment) : void
N
has
1
1
1
takes place in
N
belongs to
1
*
Course
-courseId : int
-courseName: String
-maxParticipants : int
-minParticipants: int
-duration: int
-price : double
-topic : String
-numberOfSessions : int
-startDate : Date
-endDate : Date
-difficultyLevel : enum
-instructors : Set<Instructor>
-hall : Hall
+updateCourse(course : Course) : void
+deleteCourse(courseId : int) : void
+addTrainer(Trainer : Trainer) : void
+removeTrainer(Trainer : Trainer) : void
+addTrainee(Trainee : Trainee) : void
+removeTrainee(Trainee : Trainee) : void
+isFull() : boolean
N
offers
1
SportsClub
-clubName : String
-facilities: List
-clubAddres : string
-courses : List<Course>
-clubPhonrNumber : string
-clubWebsite : string
-clubHalls : List<Hall>
-clubCourses : List<Courses>
-clubInstructors : List<Instructor>
-clubTrainers : List<Trainer>
-clubTrainees : List<Trainee>
-membershipTypes : List<MembershipType>
+addHall(hall : Hall) : void
+removeHall(hall : Hall) : void
+addCourse(course : Course) : void
+removeCourse(course : Course) : void
+addInstructor(instructor : Instructor) : void
+removeInstructor(instructor : Instructor) : void
+addCustomer(customer : Customer) : void
+removeCustomer(customer : Customer) : void
+findCourseById(id : int) : Course
+findInstructorById(id : int) : Instructor
+findCustomerById(id : int) : Customer
1
1
Admin
-adminId : int
-username : String
-password : String
-email : String
+Admin(adminId : int, username : String, password : String, email : String)
+login(username : String, password : String) : boolean
+logout() : void
+addCourse(course : Course) : void
+removeCourse(course : Course) : void
+addTrainer(Trainer : Trainer) : void
+removeTrainer(Trainer : Trainer) : void
1
1
Equipment
-name : String
-description : String
-quantity : int
+checkAvailability(quantity : int) : boolean
has
N
is tought by
N
1
employs
N
1
Serves
N
Trainer
-trainerId : int
-trainerName: String
-specialty: String
-hourlyRate : double
-expertiseTopics : List<String>
-courses : ArrayList<Course>
+addExpertiseTopic(topic : String) : void
+assignCourse(course : Course) : void
+calculateSalary(startDate : Date, endDate : Date) : double
n
Trainee
-participantId : int
-participantName: String
-phoneNumber : String
-address : String
-email : String
-membershipType : MembershipType
-courses : Set<Course>
-dateOfBirth : Date
+getEnrolledCourses() : List<Course>
+enrollToCourse(course : Course) : void
+cancelEnrollment(course : Course) : void
+viewPersonalSchedule() : List<Booking>
1
N
Booking
-bookingId : int
-customer : Customer
-course : Course
-bookingDate : Date
-payment : Payment
+Booking(bookingId : int, customer : Customer, course : Course, bookingDate : Date)
+cancelBooking() : void
requires
Payment
-paymentId : int
-customer : Customer
-amount : double
-paymentDate : Date
+Payment(paymentId : int, customer : Customer, amount : double, paymentDate : Date)
+processPayment() : void
N
makes

6 Evaluation/Verification Plan

6.1 System requirements

6.1.1 Functional requirements

Number	Category	Description	Priority (1-5) 1 High, 5 Low
1	User Management	Allow users to create new accounts by providing necessary information.	4
2	User Security	Verify user identity during login to grant access to the system.	4
3	Course Administration	Enable admins to manage courses, including creation, modification, and deletion.	3
4	User Management	Allow admins to manage Trainers, including adding, editing, and removing them.	4
5	Facility Management	Provide functionality for admins to manage halls, including availability and bookings.	3
6	Reservation System	Allow users to make bookings for courses or his halls, ensuring no conflicts.	3
7	Financial Management	Enable users to make secure payments for course fees and manage transactions.	4
8	General	Notify user when his operation failed	3
9	User	View training progress	5
10	User	Login to the club	4

11	User	Register to the club	4
12	User	Logging out of the club	4
13	User	View and Edit profile	3
14	User	Schedule training sessions with trainers	2
15	User	View training progress	2
16	User	Receive notifications	3
17	User	Filling out an opinion report after training	3
18	Admin	Remove User	4
19	Admin	Provide functionality for admins to manage instructors, including adding, editing, and removing them.	3

6.1.2 Non-Functional requirements

Number	Category	Description	Priority (1-5) 1 High, 5 Low
1	Performance	The system should be able to handle many concurrent users without significant performance degradation.	3
2	Performance	Enable smooth navigation and scrolling throughout the app	3
3	Safety	Implement user authentication and authorization	4
4	Safety	The system should ensure confidentiality	3
5	Usability	Intuitive and easy-to-use interface	1
6	Reliability	The system should be available 24/7, with minimal downtime for maintenance or upgrades.	3
7	Maintainability	Regular updates to fix bugs and introduce new features	3
8	Maintainability	Modular and easily extensible codebase	2
9	Maintainability	Efficient debugging and error resolution processes	3
10	Privacy	Protection of user information	2
11	Scalability	The system should be scalable to accommodate an increasing number of	3

		users, courses, and transactions without compromising performance.	
12	Compatibility	The system should be compatible with various devices and web browsers to ensure accessibility for a diverse user base.	3

6.2 Testing

6.2.1 Unit testing

Unit testing is the process of examining individual components of a software application to verify that they function as intended.

These tests will help us identify and fix bugs early in the development process, thereby improving the quality of the code. Additionally, unit tests will allow us to refactor our code with confidence, knowing that we have a safety net in place to catch any unexpected behavior changes.

The table below provides an overview of some of the unit tests we will conduct:

Component	**Test Case**	**Expected Result**
User Registration	Register a new user with valid details.	Registration successful, user data stored in Firebase

User Registration	Register a new user with invalid details (e.g. invalid email format or existing email)	Registration failed, error message was displayed.
User Login	Login with valid details	Login successfully, the user navigates to the main screen
User Login	Login with invalid details	Login failed, error message displayed
User Logout	Logout from the system.	user is logged out of the system.
Profile Management	Update user profile details	Profile update successful, changes reflected in Firebase
Profile Management	Edit profile picture	Profile picture changed successfully in database
Course Booking	Book a course with available slots.	Successful booking, user is enrolled in the course.
Payment Processing	process payment for a booked course.	Payment is processed successfully
View Course Details	View details of a specific course.	Course details are displayed correctly.
Add Hall	adding a new hall with valid data.	A new Hall is created with the provided information, and the

		equipment list is correctly assigned.
Add Hall	adding a new hall with empty name.	An exception is thrown indicating an invalid name.
Add Equipment	Remove equipment from a hall.	Equipment is successfully added to the hall.
Remove Equipment	Remove equipment from a hall.	Equipment is successfully removed from the hall
Add course	adding a new course with valid data.	A new Course is created with the provided information, Trainer and hall are assigned correctly.
Add course	Test adding a new course with a past start date.	An exception is thrown indicating an invalid start date.
Remove Course	Remove a course from the system.	Course is successfully deleted from the system.
Edit Course Details	Modify details of an existing course.	Course details are updated successfully. Add New Instructor
Add trainer	adding a new trainer with valid data.	A new trainer is created with the provided information.
Remove trainer	Remove an trainer from the system.	Trainer is successfully removed from the system.

6.2.2 Manual testing

Test Case	Steps	Expected Results
Navigation from the login screen to the main screen after successful login	Open the application, enter valid login credentials, and click on the 'login' button	The application should navigate to the main screen
Navigation from the main screen to the Profile screen using the navigation bar.	On the main screen, click on the 'Profile' tab in the navigation bar	The application should navigate to the Profile screen
Updating User Profile Information	Access the Profile screen, edit user profile information (e.g., name, email, phone number), and save the changes.	The updated user profile information should be successfully saved and reflected in the system.
Navigation from the main screen to a specific course	Click on a course from the section courses	The application should navigate to the specific course screen and display comprehensive information about the selected course
Viewing Registered Courses	Access the user's profile or dashboard and navigate to the section displaying registered courses.	The application should display a list of courses that the user has successfully registered for.
Logging Out	From any screen, log out of the application.	The application should successfully log the user out.

7 Criteria for Success

1. The software is developed and uploaded to the public.

2. User Adoption and Satisfaction

High adoption rate by members, instructors, and admins who will try the program.

Positive user feedback on usability.

Increased member engagement.

3. Average response time for common user actions (e.g., course booking, viewing schedules)

4. Number of different devices and browsers supported without issues

8 Process Challenges

8.1 Requirement Gathering and Analysis

One of the initial challenges in the engineering process is effectively gathering and analyzing requirements. This involves understanding the needs and expectations of stakeholders, including club owners, managers, instructors, and members. Ensuring clear communication and consensus among diverse stakeholders can be challenging, especially when dealing with varying priorities and preferences.

8.2 Planning the System

I have to figure out how to design the system so that it works well. This means choosing the right technology and deciding how everything will fit together. It's important to balance things like how fast the system needs to be, how secure it should be, and how easy it is to use.

8.3 Building the Software

Once I have a plan, I need to build the software. This involves writing the code and making sure it all works properly. It can be tough to keep track of everything and make sure it's all done on time, especially since I'm working on this project alone without a partner. This means I'll need to work hard to finish within the set timeframe.

8.4 Testing.

After the software is built, I need to test it to make sure it works like it's supposed to. This means trying out different things to see if they work and fixing any problems we find. Testing can take a lot of time and effort.

8.5 Working Alone

Building the software alone presents another challenge. Without a partner to bounce ideas off of or provide assistance, I'm solely responsible for solving every problem that arises. This means I have to rely solely on my own thinking and creativity to tackle each issue without the input or collaboration of a partner. Additionally, the absence of a partner means there's no one to share the workload with or provide additional ideas, making the process more challenging. As a result, working alone requires me to put in extra effort and work doubly hard to ensure everything is completed on time and to a high standard.

9 References

[1] https://w3techs.com/technologies/details/pl-php

[2] https://www.jetbrains.com/lp/devecosystem-2023/

[3] https://db-engines.com/en/ranking

[4] https://www.php.net/manual/en/intro-whatcando.php

[5] https://dev.mysql.com/doc/refman/8.0/en/scalability.html