UML Report : Sport Center Management System

KEIB Josselin and SANCESARIO Tom 15/12/2024

1.Introduction

1.1 Background of the Project

Our project focuses on developing a **web-based reservation system** tailored for a sports center. The system caters to three primary types of users: general users, administrators, and sports enthusiasts. It aims to streamline the management and reservation of sports fields and equipment, providing an efficient, user-friendly platform.

Users of the System

1. **General Users**:
   - Can create an account, log in, and make reservations for sports equipment or fields.
   - View their current reservations and cancel them if necessary.
2. **Administrators**:
   - Oversee the system by managing sports categories, equipment, and fields.
   - Approve or deny user access as needed.
   - Handle CRUD operations (Create, Read, Update, Delete) for sports-related assets.
3. **Sports Enthusiasts**:
   - Access detailed information about available sports fields and equipment.
   - Check real-time availability before making a reservation.

Objectives of the System

Our main goal is to make booking sports fields and equipment as smooth as possible for users while giving admins the tools to manage everything efficiently. Here's what we aim for:

1. **For General Users**:
   - Enable a seamless process for reserving equipment and fields.
   - Provide clear feedback for successful or failed reservations.
2. **For Administrators**:
   - Ensure efficient management of system resources, including sports categories, equipment, and fields.
   - Monitor user activity and reservations.
3. **System-wide Objectives**:
   - Enhance security through user authentication and role-based access.
   - Maintain a scalable and reliable architecture to handle multiple concurrent users.

Project Theme

This system aligns with themes of reservation management, with functionality similar to modern e-commerce or scheduling platforms but specialized for sports centers. It emphasizes user convenience, security, and system transparency, ensuring a smooth experience for all involved parties.

This background provides the foundation for understanding the users' needs and the system's objectives, driving the design and implementation choices detailed in the following sections.

1.2 Division of work

Our team, composed of Josselin KEIB and Tom SANCESARIO, worked collaboratively on all aspects of the project, particularly focusing on the **UML design**. Every task was approached together, with both team members contributing equally and sharing their opinions to ensure a clear and well-structured design.

The **use case diagrams** were created collectively to represent user interactions and system functionalities. Together, we developed the **activity diagrams**, illustrating the step-by-step workflows for core processes such as user

registration, login, and reservations. For the **class diagram**, we worked jointly to identify system components, define their relationships, and ensure the structure accurately represented the system's architecture.

Throughout the project, we maintained continuous communication and exchanged ideas to validate each diagram and refine details where needed. By combining our efforts and perspectives, we produced a coherent and accurate set of UML diagrams that served as the foundation for understanding and implementing the system.

2. Requirements

2.1 User Stories

We outlined what users and admins really need from this system. For general users, it’s all about easy booking and management. For admins, it’s about keeping things organized and up-to-date. General users can register on the website by creating an account and must log in to access the system. Once logged in, they can browse available sports fields and equipment, check their availability, and make reservations by selecting a specific date. Users can also view their reservations in a dedicated section and cancel them if needed. For administrators, the system provides an admin panel that requires login for access. Administrators can manage system resources by adding, updating, or deleting sports categories, fields, and equipment to ensure the system remains current and functional. Additionally, administrators can view all user reservations to monitor system activity and manage user accounts, including approving registrations and assigning admin privileges. These stories form the backbone of the system's design, ensuring a user-friendly and efficient experience for all parties involved.

2.2 Alternative Flows

The system handles several alternative flows to ensure robustness and provide appropriate feedback in case of errors or unexpected scenarios:

1. **User Registration**:

   If the username or email provided during registration is already in use, the system displays an error message indicating that the account cannot be created with the given information.

2. **Login**:

   If a user enters an incorrect email or password, the system rejects the login attempt and displays an error message prompting the user to re-enter their credentials.

3. **Reservation**:

   - The system won’t allow users to book fields or equipment for a date that’s already passed. If they try, they’ll get a message saying, “ You can only reserve for future dates.”
   - If the selected field or equipment is already reserved for the chosen date, the system blocks the reservation and informs the user of the unavailability.

4. **Admin Resource Management**:

   - When adding or updating sports fields or equipment, if the input data (e.g., name, price, or sport category) is invalid or missing, the system rejects the action and prompts the admin to correct the input.
   - If an admin attempts to delete a resource linked to an active reservation, the system prevents the deletion and displays a message explaining the conflict.

5. **Reservation Cancellation**:

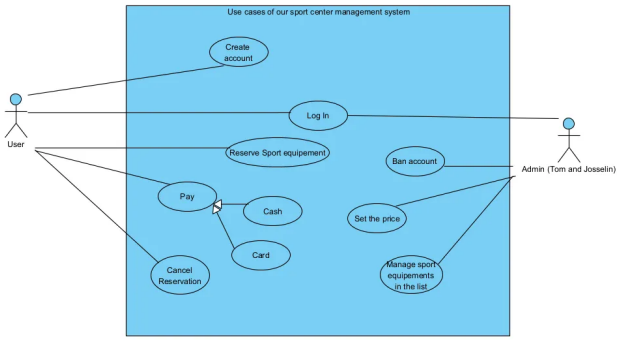
   If a user attempts to cancel a reservation that does not exist or belongs to another user, the system blocks the action and informs the user that no matching reservation could be found.

These alternative flows ensure that the system handles edge cases effectively, providing clear feedback to users while maintaining data integrity and system functionality.

2.3 Use cases

Use Case Diagram

The following diagram illustrates the main use cases of our **Sports Center Management System**, showing how users and administrators interact with the system's functionalities.



Use Case Specification

→ Here are detailed specifications for the key use cases:

Use case : Manage sport equipements in the list

- **ID** : 1
- **Primary actors :** Admin
- **Secondary actors :** None
- **Preconditions :**
  - Admin is logged in as an admin.
- **Main Flow:**
  - He can enter equipment names, pictures and prices.
  - He can delete the equipment.
  - The system stores the equipment information.
- **Postconditions :** None
- **Alternative flow :** None

Use Case: Pay

- **ID:** 2
- **Primary Actors:** User
- **Secondary Actors:** None

- **Preconditions:**
  - The user is logged in.
  - The reservation of equipment has been made.
- **Main Flow:**
  - The user can choose between payment methods (cash or card).
  - *If a card is selected, the system prompts for card details. (difficult)*
  - The payment is processed and confirmed.
  - A notification is sent to the user
- **Postconditions:** The payment is successfully made, and the reservation is confirmed.
- **Alternative Flow:**
  - If payment fails, the user is prompted to retry or select another payment method, or the user can let it down.

Use Case: Reserve

- **ID:** 3
- **Primary Actors:** User
- **Secondary Actors:** Admin (if he had created a sport, an equipment or a field)
- **Preconditions:**
  - The user is logged in and on the page Reservation
  - Equipment or field exist and related to the sport wanted.
- **Main Flow:**
  - The user clicks on the button of what he wants to reserve(equipment or field) depending on the sport.
  - The user enters a date.
  - If the process is successful then,
  - The reservation is in its basket.
- **Postconditions:** None
- **Alternative Flow:** If the chosen reservation date is already taken, the user had to enter another date.

Use Case: Cancel Reservation

- **ID:** 4
- **Primary Actors:** User
- **Secondary Actors:** Admin (if he had created a sport, an equipment or a field)
- **Preconditions:**
  - The user is logged in.
  - The user already had an existing reservation that can be canceled.
- **Main Flow:**
  - The user selects which reservation he wants to cancel.
  - The system processes the cancellation request.
  - If the process is successful then

- The system removes the reservation on the database.
- The cancellation is confirmed.

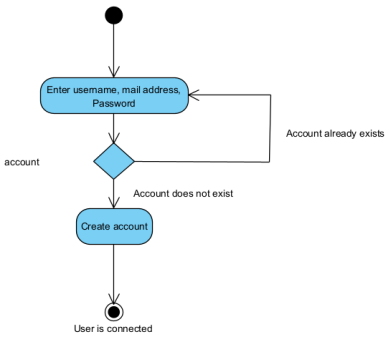
- **Postconditions:** None
- **Alternative Flow: None**

2.4 Activity diagram

The activity diagrams presented below illustrate the detailed steps involved in key use cases of the **Sports Center Management System**. Activity diagrams provide a clear visual representation of the workflows, decision points, and actions performed by users and administrators within the system.

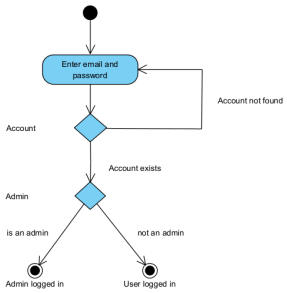
1. **User Registration Process**

   The first diagram describes the steps for creating a user account. The process begins with the user entering their details, such as username, email, and password. If the account already exists, the system provides an error message. Otherwise, the system successfully creates the account, and the user is connected to the system.



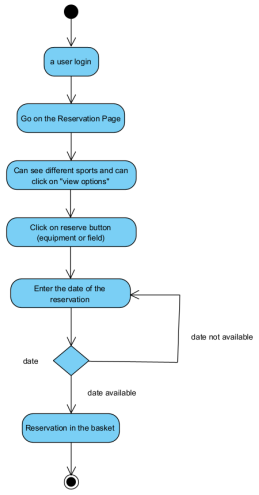
2. **User Login Process**

   The second diagram explains the user login workflow. The user provides their email and password. If the credentials are incorrect, the system rejects the login attempt. If the credentials are valid, the system verifies whether the user is an admin or a regular user and grants appropriate access based on their role.



3. **Reservation Process**

The third diagram demonstrates the workflow for making a reservation. After logging in, the user navigates to the reservation page and selects a sports field or equipment. The user enters the desired date, and the system checks availability. If the selected date is unavailable, the user is prompted to choose another date. If available, the reservation is successfully added to the user's basket.



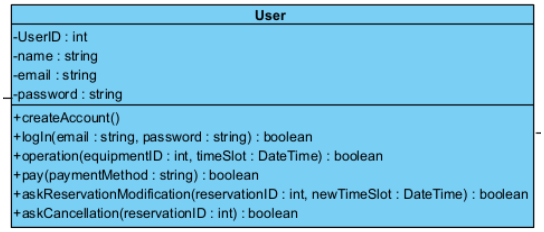
These activity diagrams effectively capture the logical flow of actions and decision points for user registration, login, and reservation processes, ensuring clarity in understanding system behavior and user interactions.

2.5 Classes

In this section, we identify the main classes of the **Sports Center Management System** and explain their responsibilities and interactions with other classes. The system is organized around core entities like users, reservations, equipment, fields, and payments, along with administrative functionalities.

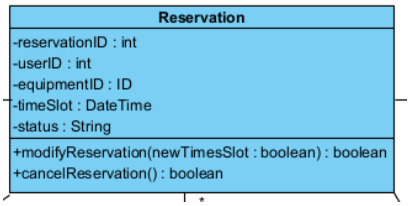
1. **User**:

   The **User** class represents general users who interact with the system. Its responsibility is to allow users to authenticate, make reservations, and manage their bookings. It collaborates with the **Reservation** class to handle user reservations and with the **Payment** class for payments.



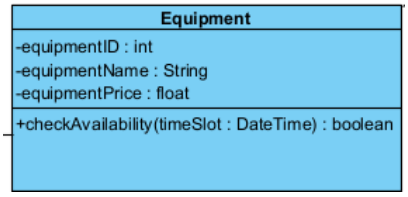
2. **Reservation**:

   The **Reservation** class manages the booking process. It links users to reserved equipment or fields and ensures reservation details like time slots and status are maintained. It collaborates with the **Equipment** and **Field** classes to check availability.



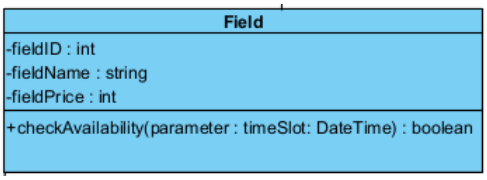
3. **Equipment**:

   The **Equipment** class represents sports equipment that can be reserved. Its primary role is to verify availability for a given time slot, collaborating with the **Reservation** class to manage bookings.



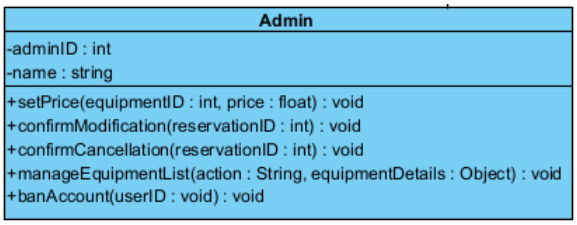
4. **Field**:

   Similar to equipment, the **Field** class represents sports fields. It ensures fields are available for reservation and collaborates with the **Reservation** class to maintain booking details.



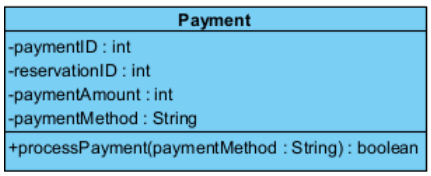
5. **Admin**:

   The **Admin** class manages system resources and user accounts. Its responsibilities include updating equipment/field pricing, handling user modification requests, and managing the list of sports resources. It collaborates with the **Equipment**, **Field**, and **User** classes.



6. **Payment**:

   The **Payment** class handles payment processing for reservations. It collaborates with the **Reservation** class to link payments to specific bookings.



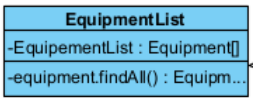
7. **Sport**:

   The **Sport** class categorizes the fields and equipment under specific sports. It works with the **Equipment** and **Field** classes to organize resources.

8. **Supporting Classes**:

   - **UserList**: Manages and retrieves all users in the system.

   - **EquipmentList**: Provides methods to manage equipment details.



   - **ReservationList**: Maintains and retrieves all reservation data.

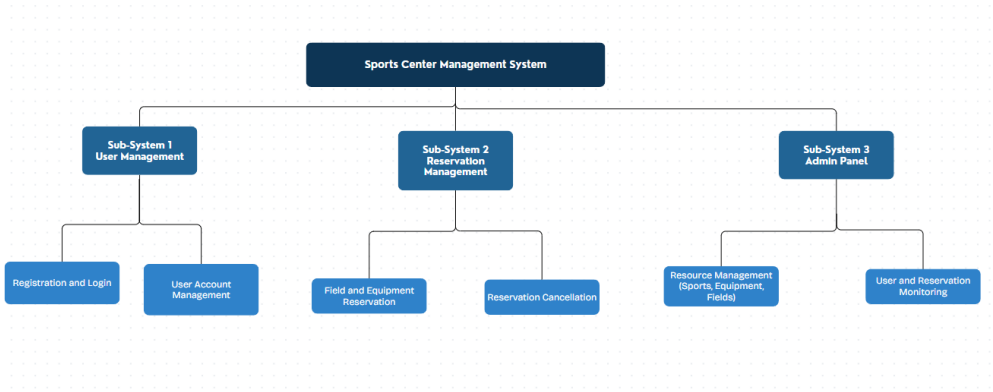
These classes collectively form the core of the system, ensuring smooth collaboration between users, reservations, resources, and administrative functionalities.

3. Design

3.1 Architectural Design

The architectural design of the **Sports Center Management System** is organized into three main subsystems: **User Management**, **Reservation Management**, and **Admin Panel**. Each subsystem contains specific modules that handle distinct functionalities, ensuring a modular, maintainable, and scalable architecture.

System Architecture



Explanation of Subsystems and Modules

1. **Sub-System 1: User Management**
   - **Registration and Login**:
     This module allows users to create accounts and securely log in to the system. It verifies credentials and assigns roles (user or admin) to ensure proper access control.
   - **User Account Management**:
     This module enables users to manage their accounts by viewing or updating their personal details such as name, email, and password.
2. **Sub-System 2: Reservation Management**
   - **Field and Equipment Reservation**:
     Users can browse sports fields and equipment, check availability, and make reservations for specific dates. This module handles reservation requests and ensures no conflicts occur.
   - **Reservation Cancellation**:
     Users can cancel their reservations when needed. The system updates the availability of the respective sports fields or equipment accordingly.
3. **Sub-System 3: Admin Panel**
   - **Resource Management (Sports, Equipment, Fields)**:
     Administrators can manage system resources by adding, updating, or deleting sports categories, fields, and equipment. This ensures the system is always up to date.
   - **User and Reservation Monitoring**:
     Administrators can monitor all user accounts and reservation activities. They can view reservation details, manage user roles, and take actions such as banning accounts or resolving conflicts.

3.3 Class Design

The **Class Design** of the **Sports Center Management System** provides a detailed explanation of the attributes and methods of each class, along with a focus on their technical structure and interactions. This section builds upon the conceptual overview from Section 2.5 by offering a closer look at the internal design of the classes and the relationships that link them.

User Class

The **User** class is central to managing user interactions within the system. Each user is represented by a unique `UserID`, and additional attributes such as `name`, `email`, and `password` store essential information. The methods provided by the User class ensure user functionality, including `createAccount()` for account registration and `login()` for authentication. Users can make reservations using the `operation()` method and handle payments through `pay()`. The

class also enables reservation management through `askReservationModification()` and `askCancellation()`, ensuring user control over their bookings.

This class maintains a **one-to-many relationship** with the **Reservation** class since a single user can make multiple reservations.

Reservation Class

The **Reservation** class models the process of booking resources, such as sports equipment or fields. It includes attributes like `reservationID` (unique for each reservation), `userID` (linking the reservation to a user), and `timeSlot` to represent when the reservation occurs. A `status` attribute indicates whether the reservation is active or cancelled.

The class exposes methods like `modifyReservation()` to reschedule bookings and `cancelReservation()` to handle cancellations. These methods ensure that reservation details remain up to date. The **Reservation** class interacts with both the **Equipment** and **Field** classes, as reservations are linked to these resources.

Equipment Class

The **Equipment** class defines the structure for managing sports equipment that can be reserved. Its attributes include `equipmentID` to uniquely identify equipment, `equipmentName`, and `equipmentPrice`. The `checkAvailability(timeSlot)` method ensures that equipment can only be booked when available, preventing conflicts.

The **Reservation** class references the Equipment class through a **many-to-one relationship**, where multiple reservations can be made for the same piece of equipment as long as time slots do not overlap.

Field Class

The **Field** class shares similarities with the Equipment class but focuses on sports fields. It contains attributes such as `fieldID`, `fieldName`, and `fieldPrice`. The method `checkAvailability(timeSlot)` performs availability checks for the requested time slot, ensuring fields are reserved only when they are free.

The class maintains a **many-to-one relationship** with the **Reservation** class, as fields can have multiple reservations at different times.

Admin Class

The **Admin** class provides the tools for system management. Administrators are identified by attributes like `adminID` and `name`. The methods in this class offer key administrative functionalities:

- `setPrice()` enables the modification of prices for fields or equipment.
- `confirmModification()` and `confirmCancellation()` allow admins to validate user requests related to reservations.
- `manageEquipmentList()` ensures equipment resources are kept up to date by supporting actions such as adding, updating, or removing items.
- `banAccount()` provides control over user access, ensuring that system integrity is maintained.

The **Admin** class has associations with **User**, **Equipment**, and **Reservation** classes, as it oversees both user activities and system resources.

Payment Class

The **Payment** class is responsible for processing transactions linked to reservations. It includes attributes such as `paymentID`, `reservationID`, `paymentAmount`, and `paymentMethod`. The method `processPayment()` validates the payment method (cash or card) and ensures successful completion of the transaction.

This class maintains a **one-to-one relationship** with the **Reservation** class, as each payment is tied to a specific booking.

Sport Class

The **Sport** class organizes resources (fields and equipment) into categories based on sports. It includes attributes like `sportID` and `sportName`. This class simplifies resource browsing and helps link equipment and fields to their respective sports.

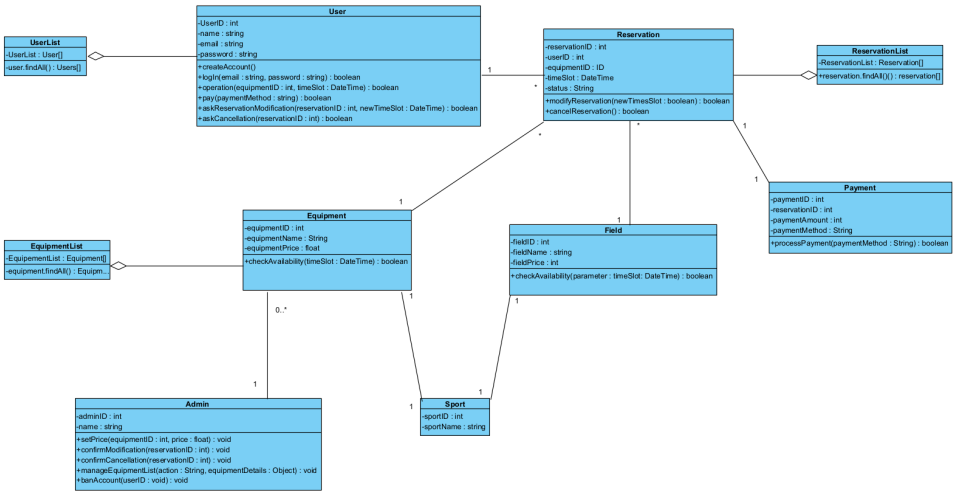
Class Relationships

The relationships between classes ensure smooth collaboration across the system:

- The **User** class interacts with **Reservation** to manage bookings.
- The **Reservation** class connects to both **Equipment** and **Field** to validate and manage resource availability.
- The **Payment** class ensures transactions are tied to specific reservations.
- The **Admin** class interacts with **User**, **Equipment**, and **Reservation** to oversee and manage the system's functionality.

These relationships, combined with clearly defined attributes and methods, provide a robust design for the **Sports Center Management System**.

Class Diagram



The class design presented here ensures that the system is modular, with each class focusing on a specific set of responsibilities. By clearly defining relationships and methods, the system maintains flexibility, allowing for future enhancements and scalability.

4. Implementation

4.1 Module 2: Field and Equipment Reservation

Introduction to the Module

Module 2 focuses on the **Field and Equipment Reservation** functionality of the Sports Center Management System. This module allows users to check the availability of sports fields or equipment, and if available, proceed to reserve the resource. The system ensures that reservation requests are validated before being saved and provides appropriate feedback to the user regarding success or failure.

Classes Involved in Module 2

1. **User Class**

   The `User` class initiates the reservation process. It provides methods like `operation(equipmentID, timeSlot)` to allow users to select a resource (field or equipment) and a time slot for booking.

2. **Reservation Class**

   The `Reservation` class handles the core logic of creating and managing reservations. It validates inputs, checks the availability of resources, and updates reservation details upon confirmation.

3. **Equipment Class**

   The `Equipment` class is responsible for verifying the availability of equipment using the `checkAvailability()` method when a reservation request is made.

4. **Field Class**

   Similar to the Equipment class, the `Field` class checks the availability of sports fields at the requested time slot using the `checkAvailability()` method.

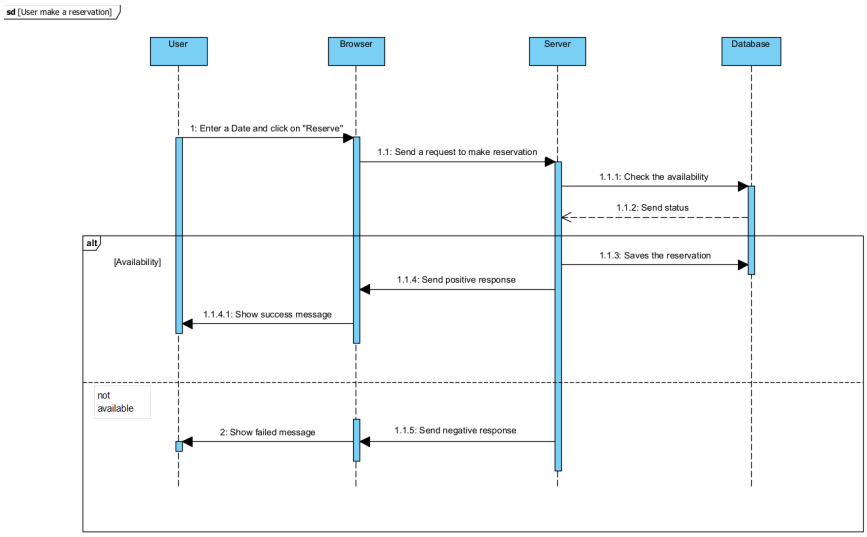
5. **Database**

   The database interacts with the `Reservation` class to store confirmed reservations and maintain the updated status of fields or equipment.

Sequence Diagram

The sequence diagram below illustrates the step-by-step interactions that occur in Module 2 when a user attempts to make a reservation:

- The **User** initiates the process by entering a date and clicking the "Reserve" button.
- The **Browser** sends a reservation request to the **Server**.
- The **Server** communicates with the **Database** to check the availability of the requested field or equipment.
- Depending on the availability status:
  - If the resource is **available**, the reservation is saved in the database, and a positive response is sent back to the user, who receives a success message.
  - If the resource is **not available**, a failure message is returned to the user.



Explanation of the Diagram

1. The user starts the process by interacting with the **Browser** to request a reservation.
2. The **Browser** sends the reservation request to the **Server**, which handles the reservation logic.
3. The **Server** communicates with the **Database** to perform two key operations:
   - Check the availability of the resource (`checkAvailability`).
   - If available, save the reservation (`saveReservation`).
4. The **Server** responds to the **Browser** based on the availability status:
   - If successful, the user sees a confirmation message.
   - If unsuccessful, the user sees a failure message indicating that the resource is not available.

The classes **User**, **Reservation**, **Equipment**, and **Field** work together to ensure that only valid reservations are saved and users receive clear feedback on their requests. This process highlights the seamless communication between system components to maintain accuracy and efficiency.

4.2 Database Design

The **Sports Center Management System** relies on a relational database to store and manage information about users, sports, equipment, fields, and reservations. The database schema is designed to ensure data consistency, integrity, and efficient querying. Below is an explanation of the tables, their structure, and relationships.

1. User Table

The **User** table stores all user-related information, including both general users and administrators.

- **id**: This is the primary key, an auto-incremented integer that uniquely identifies each user.
- **username**: A string that holds the user's name; it is a required field.
- **email**: A string that stores the user's email address. It is unique to prevent duplicate registrations and cannot be null.
- **password**: A string that securely stores the user's password. This is a required field.
- **isAdmin**: A boolean flag to distinguish between regular users and administrators. By default, its value is set to `false`.
- **createdAt** and **updatedAt**: These timestamps are automatically managed by Sequelize to track when a user record was created or last updated.

2. Sport Table

The **Sport** table categorizes different types of sports and is referenced by other tables to link fields and equipment to a specific sport.

- **id**: The primary key, an auto-incremented integer to uniquely identify each sport.
- **name**: A required string that holds the name of the sport.
- **imageUrl**: A nullable string that stores the URL of an image representing the sport.
- **createdAt** and **updatedAt**: Timestamps automatically managed by Sequelize to track creation and updates.

3. Equipment Table

The **Equipment** table manages sports equipment available for reservation. Each piece of equipment is associated with a specific sport.

- **id**: The primary key, an auto-incremented integer that uniquely identifies each equipment item.
- **name**: A required string representing the equipment's name.
- **price**: A decimal value that specifies the cost of reserving the equipment.
- **sportId**: A foreign key referencing the `Sport(id)` column to link equipment to its corresponding sport.
- **imageUrl**: A nullable string that stores the URL of an image for the equipment.
- **createdAt** and **updatedAt**: Timestamps that automatically record when the equipment entry was created or updated.

4. Field Table

The **Field** table stores information about sports fields available for reservation. Each field is associated with a specific sport.

- **id**: The primary key, an auto-incremented integer that uniquely identifies each field.
- **name**: A required string representing the name of the field.
- **price**: A decimal value that indicates the cost of reserving the field.
- **sportId**: A foreign key referencing the `Sport(id)` column to link fields to their corresponding sport.
- **imageUrl**: A nullable string storing the URL of an image representing the field.
- **createdAt** and **updatedAt**: Timestamps that track when a field record was created or last updated.

5. Reservation Table

The **Reservation** table manages all bookings made by users for equipment or fields. It connects users to the resources they have reserved and records booking details.

- **id**: The primary key, an auto-incremented integer to uniquely identify each reservation.
- **type**: An enum field that specifies the type of reservation (`equipment` or `field`).
- **item_id**: An integer referencing the ID of the reserved item, either from the **Equipment** or **Field** tables. This ensures flexibility to handle different types of reservations.
- **date**: A required date field indicating the reservation date.
- **user_id**: A foreign key referencing the `User(id)` column to link the reservation to a specific user.
- **createdAt** and **updatedAt**: Timestamps managed by Sequelize to record when the reservation was made and last updated.

Relationships Between Tables

1. The **User** table has a **one-to-many** relationship with the **Reservation** table, as each user can make multiple reservations.
2. The **Sport** table is referenced by both the **Equipment** and **Field** tables, establishing a **one-to-many** relationship where each sport can have multiple fields and equipment items.
3. The **Reservation** table uses the `item_id` field to reference either the **Equipment** or **Field** table, allowing it to handle both types of reservations through a flexible design.
4. The **Reservation** table links back to the **User** table using the `user_id` foreign key, creating a clear association between users and their bookings.

The database design ensures data integrity, consistency, and flexibility to support the core functionalities of the system. The relationships between tables allow seamless interaction across users, sports, equipment, fields, and reservations, forming the backbone of the Sports Center Management System.

Conclusion

The development of the **Sports Center Management System** was a valuable experience that allowed us to apply software engineering principles to solve real-world challenges. By working collaboratively, we designed and implemented a functional, user-friendly system that meets the needs of both users and administrators.

This project helped us sharpen our skills in designing systems, solving real-world problems, and working in a structured, collaborative way. It reinforced the value of teamwork, precision, and adaptability in creating a reliable and efficient solution.

In the end, this project reflects our ability to transform ideas into a working system and has equipped us with the skills and confidence to take on future challenges.