**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

**KHOA CÔNG NGHỆ THÔNG TIN 1**

A red and yellow logo with a star

AI-generated content may be incorrect.

**BÁO CÁO LẬP TRÌNH WEB**

**XÂY DỰNG WEBSITE ĐẶT SÂN CẦU LÔNG VÀ CHO THUÊ VỢT**

|  |  |
| --- | --- |
| **Sinh viên thực hiện:** | **Lại Quốc Đạt** |
| **Giảng viên hướng dẫn:** | **Nguyễn Quốc Uy** |
| **Mã sinh viên** | **B22DCCN187** |
| **Lớp:** | **D22CQCN07-B** |
| **Nhóm:** | **01** |

**Hà Nội – 2025**

**LỜI CẢM ƠN**

Em xin gửi lời cảm ơn chân thành nhất đến Thầy Nguyễn Quốc Uy, người đã tận tình giảng dạy và hướng dẫn em trong suốt quá trình học tập và hoàn thiện báo cáo môn Lập trình Web. Những kiến thức quý báu, sự hỗ trợ nhiệt tình cùng những góp ý chi tiết của Thầy không chỉ giúp em hoàn thành tốt bài báo cáo mà còn mang lại cho em nhiều bài học giá trị về chuyên môn và tư duy lập trình. Em luôn trân trọng sự tâm huyết và nỗ lực của Thầy trong việc truyền đạt kiến thức và khích lệ tinh thần học tập của sinh viên. Kính chúc Thầy thật nhiều sức khỏe, hạnh phúc và tiếp tục gặt hái nhiều thành công trong sự nghiệp giảng dạy.

Em xin chân thành cảm ơn!

|  |
| --- |
| Hà Nội, tháng 5 năm 2024 |
| Sinh viên |
| Lại Quốc Đạt |

# **NHẬN XÉT, ĐÁNH GIÁ, CHO ĐIỂM**

**(Của giảng viên hướng dẫn)**

………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………………

**Điểm:** ………………………. (**Bằng chữ:** ……………………………………)

|  |
| --- |
| Hà Nội, tháng 5 năm 2025 |
| **CÁN BỘ-GIẢNG VIÊN HƯỚNG DẪN** |
| **NGUYỄN QUỐC UY** |

Chương 1: Giới thiệu chung

1. **Giới thiệu đề tài**

Trong thời đại công nghệ phát triển, nhu cầu tham gia các hoạt động thể thao như cầu lông ngày càng tăng, kéo theo sự cần thiết của các dịch vụ hỗ trợ tiện lợi và hiệu quả. Đề tài “Website đặt sân cầu lông và cho thuê vợt” được xây dựng nhằm cung cấp một nền tảng trực tuyến giúp người dùng dễ dàng đặt sân cầu lông và thuê vợt một cách nhanh chóng, tiện lợi.

Website không chỉ đáp ứng nhu cầu đặt lịch sân một cách linh hoạt mà còn tích hợp dịch vụ cho thuê vợt, mang lại trải nghiệm toàn diện cho người chơi. Thông qua việc ứng dụng các công nghệ lập trình web hiện đại, hệ thống đảm bảo giao diện thân thiện, dễ sử dụng và tối ưu hóa quy trình đặt dịch vụ. Đề tài này không chỉ thể hiện tính ứng dụng thực tiễn mà còn giúp em củng cố kiến thức về lập trình web, từ thiết kế giao diện đến quản lý cơ sở dữ liệu và tương tác người dùng.

Mục tiêu của website là mang đến một giải pháp tiện ích, tiết kiệm thời gian cho người chơi cầu lông, đồng thời tạo nền tảng để mở rộng các dịch vụ thể thao khác trong tương lai.

1. **Các chức năng chính**

**-** Đăng nhập, đăng kí, đổi mật khẩu.

- Quản lý người dùng.

- Quản lý loại sân và sân.

- Quản lý loại vợt và vợt.

- Quản lý ngày và giờ.

- Quản lý trạng thái đặt sân và thuê vợt.

- Xem bảng xếp hạng sân, vợt, người dùng.

- Tìm kiếm theo tên của sân hoặc vợt.

- Lọc và phân trang.

- Đặt sân và thuê vợt.

- Xem danh sách đã thuê.

1. **Kiến trúc triển khai**

Kiến trúc phân lớp (Layered Architecture) là một mô hình thiết kế phần mềm được sử dụng rộng rãi trong phát triển ứng dụng web. Mô hình này tổ chức code thành các lớp logic riêng biệt, mỗi lớp có trách nhiệm và chức năng cụ thể. Việc phân chia này giúp tạo ra một cấu trúc rõ ràng, dễ bảo trì và có thể mở rộng.

* Tầng trình diễn (Presentation Layer)

- Tầng này chịu trách nhiệm tương tác với người dùng cuối, xử lý các yêu cầu HTTP và trả về phản hồi. Đây là nơi giao tiếp giữa hệ thống và người dùng hoặc các hệ thống bên ngoài.

**- Chức năng chính**:

* Nhận yêu cầu từ người dùng (thông qua giao diện web, API, hoặc ứng dụng client).
* Trả về phản hồi (JSON, HTML, hoặc các định dạng khác).
* Xử lý giao diện người dùng (nếu có) hoặc định dạng dữ liệu API.
* Xác thực và phân quyền sơ bộ (thường thông qua tầng bảo mật).
* Tầng xử lý nghiệp vụ (Business Layer/Service Layer)

- Tầng này chứa logic nghiệp vụ cốt lõi của ứng dụng, nơi xử lý các quy tắc và tính toán liên quan đến chức năng của hệ thống.

**- Chức năng chính**:

* Thực hiện các quy tắc nghiệp vụ (business rules).
* Điều phối luồng xử lý giữa tầng trình diễn và tầng truy xuất dữ liệu.
* Đảm bảo tính toàn vẹn và logic của ứng dụng.
* Tầng truy xuất dữ liệu (Data Access Layer)

- Tầng này chịu trách nhiệm tương tác với cơ sở dữ liệu hoặc các nguồn dữ liệu khác, thực hiện các thao tác như truy vấn, lưu trữ, cập nhật hoặc xóa dữ liệu.

**- Chức năng chính**:

* Giao tiếp với cơ sở dữ liệu (SQL, NoSQL, hoặc các hệ thống lưu trữ khác).
* Thực hiện các thao tác CRUD (Create, Read, Update, Delete).
* Chuyển đổi dữ liệu giữa cơ sở dữ liệu và đối tượng trong ứng dụng.
* Tầng bảo mật (Security Layer)

- Tầng này chịu trách nhiệm đảm bảo an toàn cho ứng dụng, bao gồm xác thực (authentication), phân quyền (authorization), và bảo vệ dữ liệu.

**- Chức năng chính**:

* **Xác thực**: Kiểm tra danh tính người dùng (username/password, token, OAuth2, v.v.).
* **Phân quyền**: Quy định quyền truy cập vào các tài nguyên hoặc chức năng của hệ thống.
* **Bảo vệ dữ liệu**: Mã hóa dữ liệu, bảo vệ chống lại các cuộc tấn công như SQL Injection, XSS, CSRF.
* **Tương tác giữa các tầng**
* **Tầng trình diễn** nhận yêu cầu từ người dùng, gọi **tầng xử lý nghiệp vụ** để thực hiện logic, và trả về kết quả.
* **Tầng xử lý nghiệp vụ** gọi **tầng truy xuất dữ liệu** để lấy hoặc lưu dữ liệu.
* **Tầng bảo mật** hoạt động xuyên suốt, kiểm tra quyền truy cập và bảo vệ dữ liệu ở mọi tầng.
* **Luồng dữ liệu**: Client → Presentation → Business → Data Access → Database.

1. **Công nghệ sử dụng**
2. HTML

**HTML (HyperText Markup Language)**: Là ngôn ngữ đánh dấu dùng để cấu trúc nội dung trên website, tạo nên khung xương của các trang web như giao diện đặt sân, danh sách vợt cho thuê và thông tin dịch vụ. HTML đảm bảo nội dung được trình bày rõ ràng và dễ tiếp cận.

1. CSS

**CSS (Cascading Style Sheets)**: Được sử dụng để thiết kế giao diện trực quan, thân thiện với người dùng. CSS giúp định dạng màu sắc, bố cục và hiệu ứng, mang lại trải nghiệm thẩm mỹ và đồng nhất trên toàn bộ website.

1. JavaScript

**JavaScript**: Là ngôn ngữ lập trình phía client, đóng vai trò quan trọng trong việc tạo ra các tính năng tương tác động trên website. JavaScript được sử dụng để xử lý các sự kiện như kiểm tra tính hợp lệ của thông tin người dùng nhập (ví dụ: kiểm tra thời gian đặt sân), hiển thị thông báo thời gian thực (như xác nhận đặt sân thành công), và cập nhật động nội dung trang mà không cần tải lại, chẳng hạn như hiển thị trạng thái sân trống hoặc danh sách vợt có sẵn. Ngoài ra, JavaScript còn hỗ trợ tích hợp các API để gửi yêu cầu đến backend, đảm bảo giao tiếp mượt mà giữa giao diện người dùng và cơ sở dữ liệu. Sự linh hoạt của JavaScript giúp website trở nên sống động, dễ sử dụng và đáp ứng nhanh chóng nhu cầu của người chơi.

1. SpringBoot

**Spring Boot**: Là framework Java mạnh mẽ, được sử dụng để xây dựng phần backend, xử lý các logic nghiệp vụ như quản lý lịch đặt sân, thuê vợt và xác thực người dùng, đảm bảo hiệu suất cao và tính bảo mật của hệ thống.

1. MySQL

**MySQL**: Là hệ quản trị cơ sở dữ liệu quan hệ, lưu trữ thông tin về sân, vợt, lịch đặt và tài khoản người dùng. MySQL đảm bảo dữ liệu được quản lý hiệu quả, dễ truy xuất và hỗ trợ mở rộng trong tương lai.

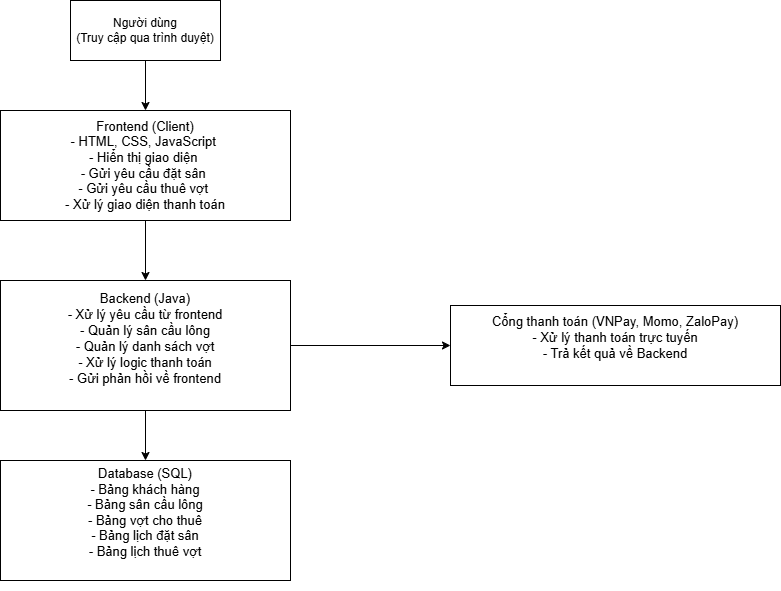
1. **FDD Diagram**



*Hình 1. FDD Diagram*

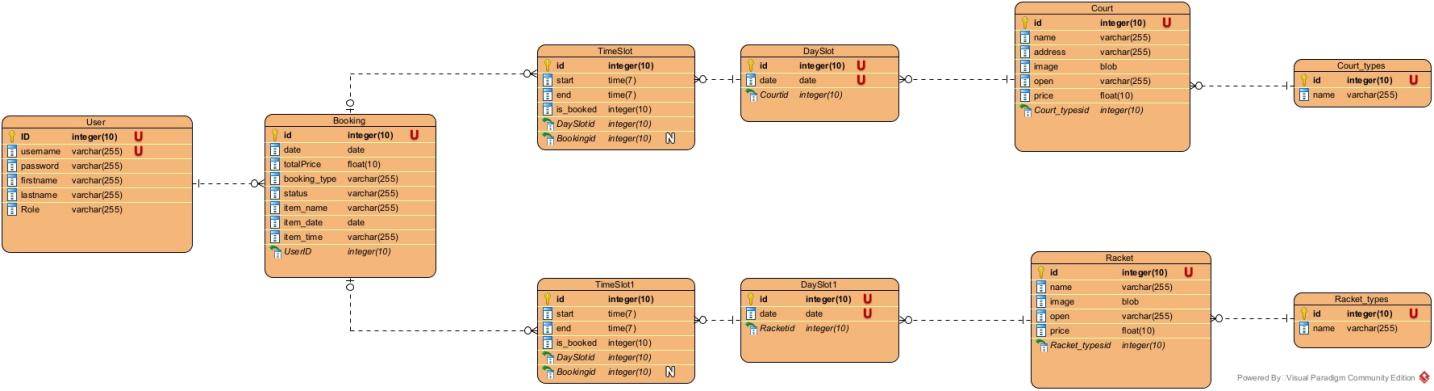
Chương 2: Thiết kế hệ thống

1. **System Architecture Diagram - Thiết kế kiến trúc tổng thể**



*Hình 2.1 System Architecture Diagram*

1. **ERD (Entity Relationship Diagram) - Sơ đồ mối quan hệ thực thể**



*Hình 2.2 Entity Relationship Diagram*

Quản hệ giữa các bảng:

- Bảng User (lưu thông tin người dùng) và Booking (lưu thông tin của đơn đặt): 1 - n

- Bảng Booking và TimeSlot (lưu thông tin về giờ của sân): 1 - n

- Bảng Booking và TimeSlot1 (lưu thông tin về giờ của vợt): 1 - n

- Bảng TimeSlot và DaySlot (lưu thông tin về ngày của sân): n - 1

- Bảng TimeSlot1 và DaySlot1 (lưu thông tin về ngày của vợt): n -1

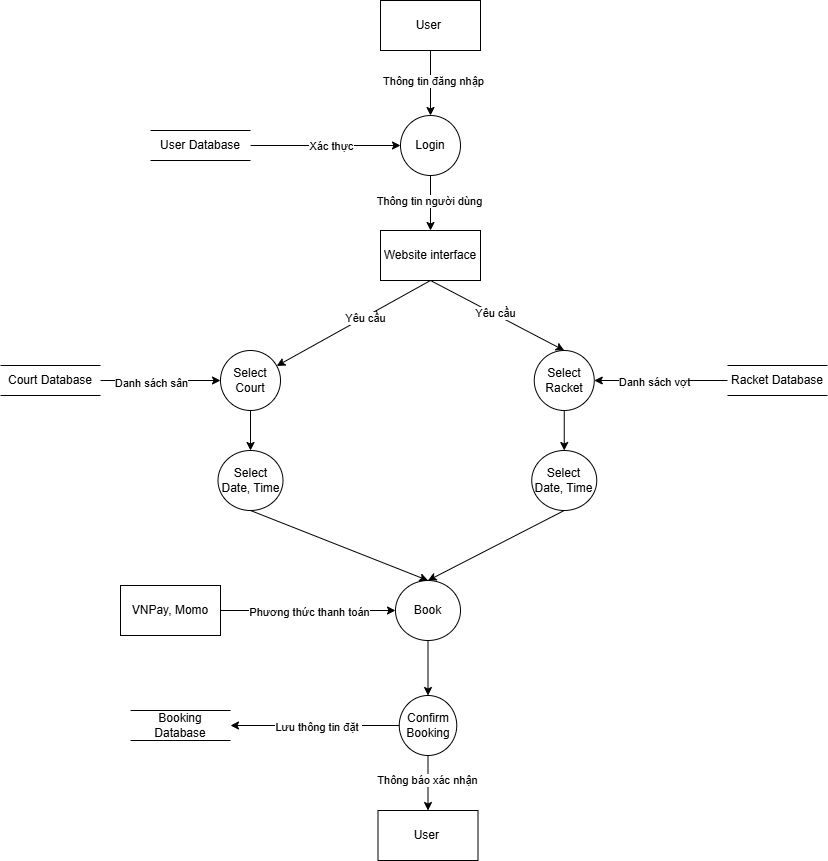
- Bảng Court (lưu thông tin về sân) và DaySlot: 1 - n

- Bảng Racket (lưu thông tin về vợt) và DaySlot1: 1-n

- Bảng Court\_types (lưu thông tin về loại sân) và Court: 1 - n

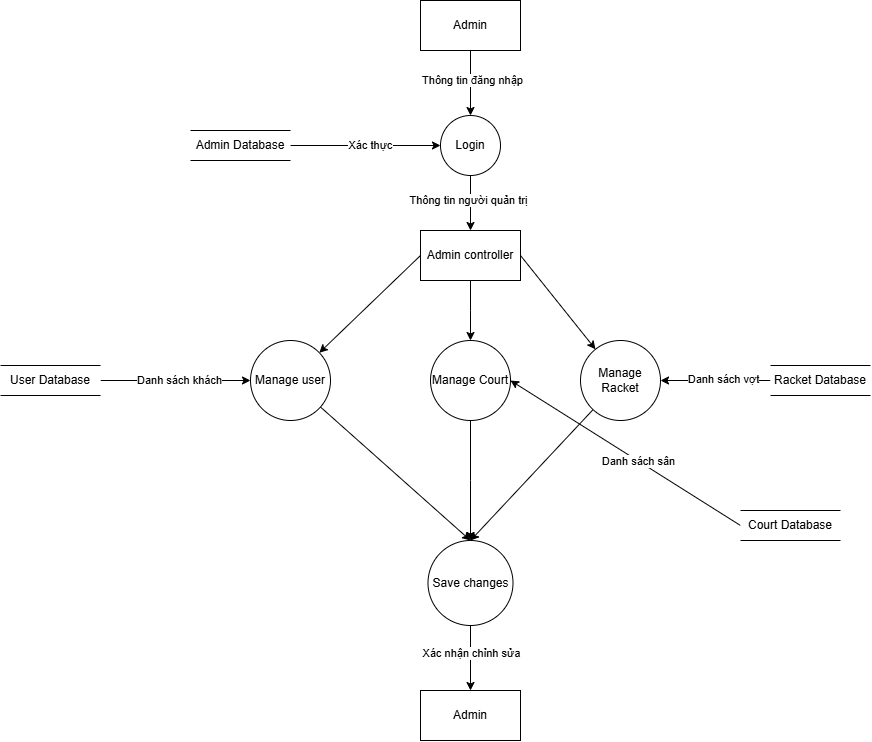
- Bảng Racket\_types (lưu thông tin về loại vợt) và Racket: 1- n

1. **Data Flow Diagram - Sơ đồ luồng dữ liệu**
2. User



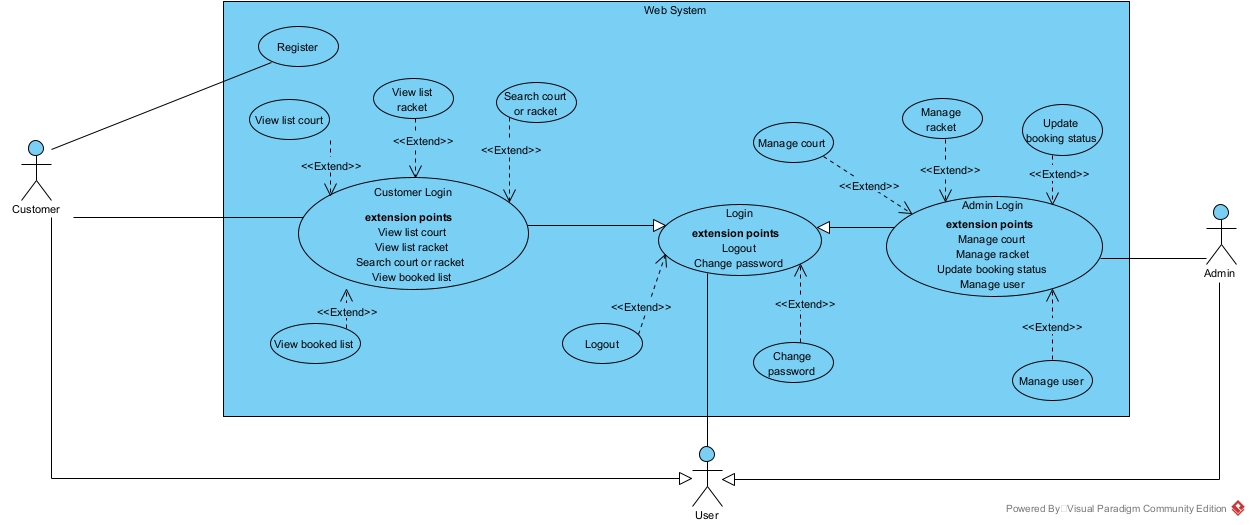
*Hình 2.3 Data Flow Diagram cho User*

1. Admin



*Hình 2.4 Data Flow cho Admin*

1. **Biểu đồ usecase**
2. Sơ đồ tổng quát



*Hình 2.5 Sơ đồ UC tổng quát*

Mô tả chi tiết UC:

- UC Login: UC này cho phép người dùng (Admin, Customer) đăng nhập vào hệ thống và thực hiện các chức năng của mình.

- UC Logout: UC này cho phép người dùng đăng xuất khỏi hệ thống.\

- UC Change password: UC này cho phép người dùng đổi mật khẩu tài khoản của mình.

- UC Register: UC này cho phép khách hàng đăng kí tài khoản của mình.

- UC View list court: UC này cho phép khách hàng xem danh sách các sân.

- UC Vỉew list racket: UC này cho phép khách hàng xem danh sách các vợt.

- UC View booked list: UC này cho phép khách hàng xem danh sách đặt sân và thuê vợt.

- UC Search court or racket: UC này cho phép khách hàng tìm kiếm sân hoặc vợt theo tên.

- UC Manage court: UC này cho phép admin quản lí sân.

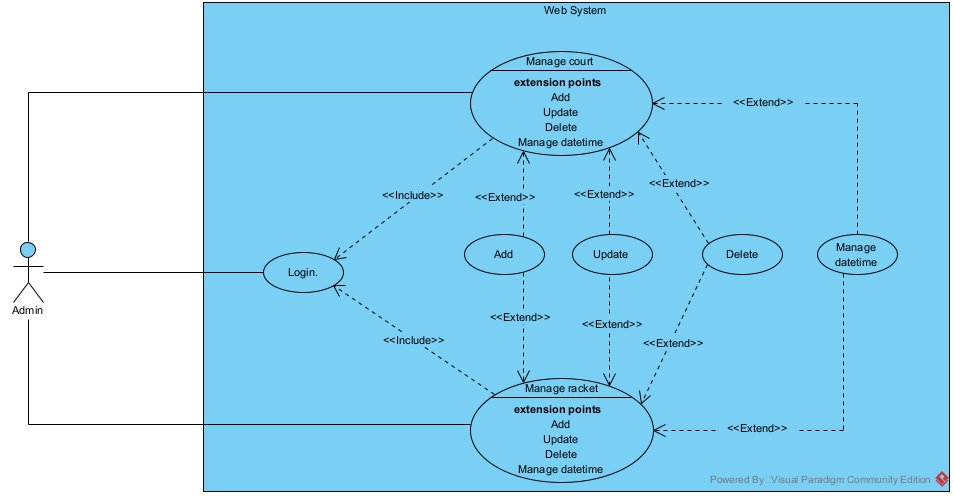
- UC Manage racket: UC này cho phép admin quản lí vợt.

- UC Manage user: UC này cho phép admin quản lí khách hàng.

- UC Update booking status: UC này cho phép admin cập nhật trạng thái đơn đặt.

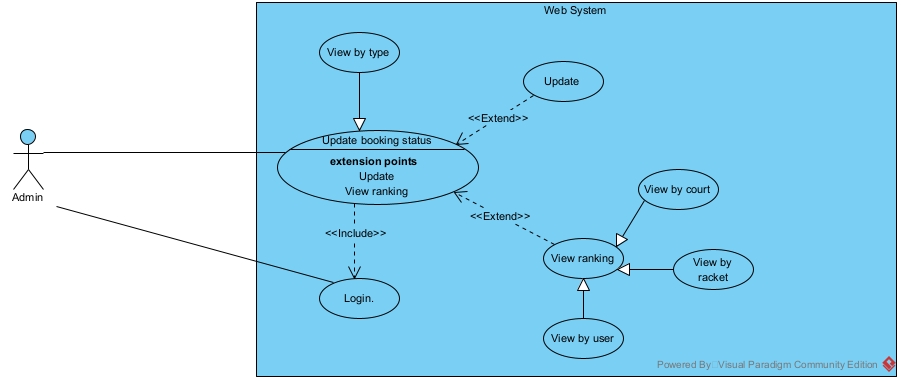
1. UC chi tiết cho admin

- Quản lí sân và vợt



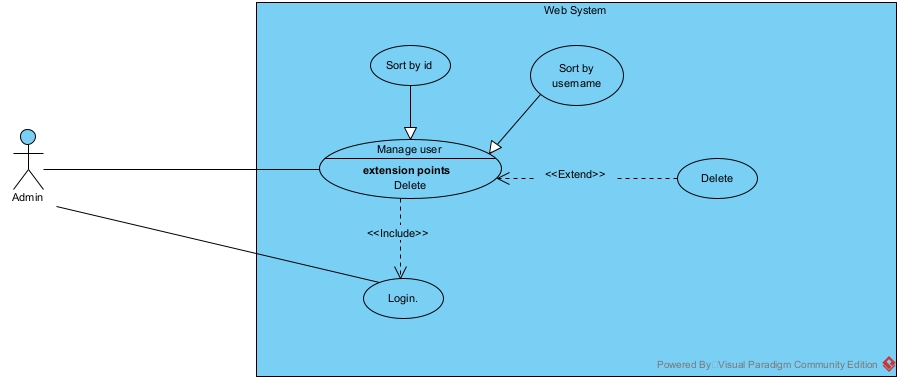
*Hình 2.6 Sơ đồ UC quản lí sân và vợt*

- Quản lí đơn đặt



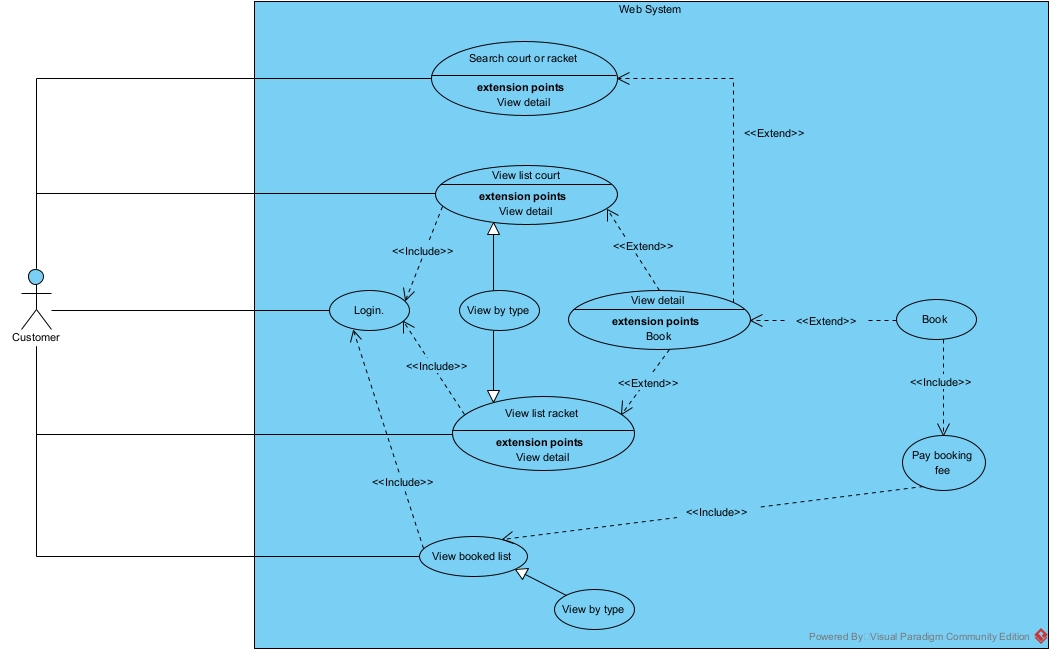
*Hình 2.7 Sơ đồ UC quản lí đơn đặt*

- Quản lí user



*Hình 2.8 Sơ đồ UC quản lí user*

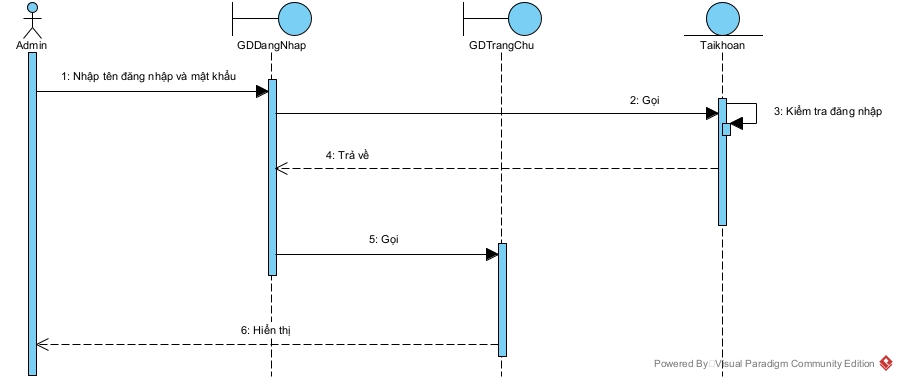
1. UC chi tiết cho khách hàng



*Hình 2.9 Sơ đồ UC chi tiết cho khách hàng*

1. **Sequence Diagram - Sơ đồ trình tự**
2. User

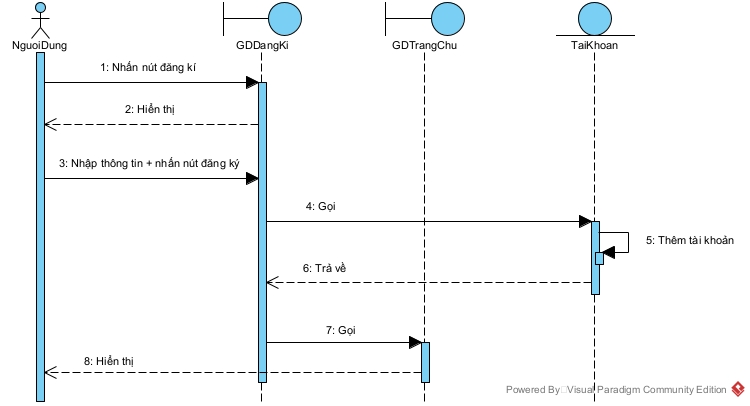
- Đăng nhập



*Hình 2.10 Sequence Diagram (SQ) đăng nhập*

|  |  |
| --- | --- |
| Tên usecase | Đăng nhập |
| Tên actor | User |
| Tiền điều kiện | User muốn đăng nhập |
| Hậu điều kiện | User đăng nhập thành công |
| Kịch bản | 1. User truy cập vào giao diện đăng nhập của hệ thống. 2. Hệ thống hiển thị ra 1 khung gồm tên đăng nhập và mật khẩu. 3. User nhập tên đăng nhập và mật khẩu và ấn vào nút “đăng nhập” 4. Hệ thống thông báo “đăng nhập thành công” và vào giao diện trang chủ. |
| Ngoại lệ | 3.1.1 User không nhập tên đăng nhập hoặc mật khẩu nhưng vẫn ấn nút “đăng nhập”  3.1.2 Hệ thống thông báo “Vui lòng nhập tên đăng nhập hoặc mật khẩu”  4.1 Hệ thống thông báo “Sai mật khẩu hoặc tên đăng nhập” |

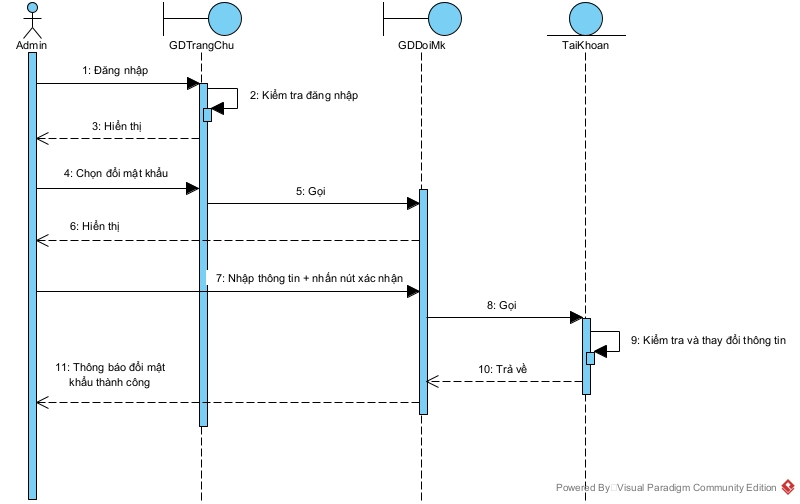
- Đăng ký



*Hình 2.11 SQ đăng ký*

|  |  |
| --- | --- |
| Tên usecase | Đăng ký |
| Tên actor | User |
| Tiền điều kiện | User muốn đăng kí |
| Hậu điều kiện | User đăng kí thành công |
| Kịch bản | 1. User truy cập vào giao diện đăng kí của hệ thống 2. Hệ thống hiện ra 1 khung gồm họ, tên, tên đăng nhập, mật khẩu và nhập lại mật khẩu. 3. User nhập đầy đủ thông tin và ấn vào nút “đăng kí” 4. Hệ thống hiển thị đăng kí thành công và chuyển đến giao diện đăng nhập |
| Ngoại lệ | 3.1.1 User nhập thiếu thông tin và ấn nút “đăng kí”  3.1.2 Hệ thống hiển thị yêu cầu nhập đầy đủ thông tin.  4.1 Hệ thống hiển thị tên đăng nhập đã tồn tại.  4.2 Hệ thống hiển thị “nhập lại mật khẩu không khớp với mật khẩu” |

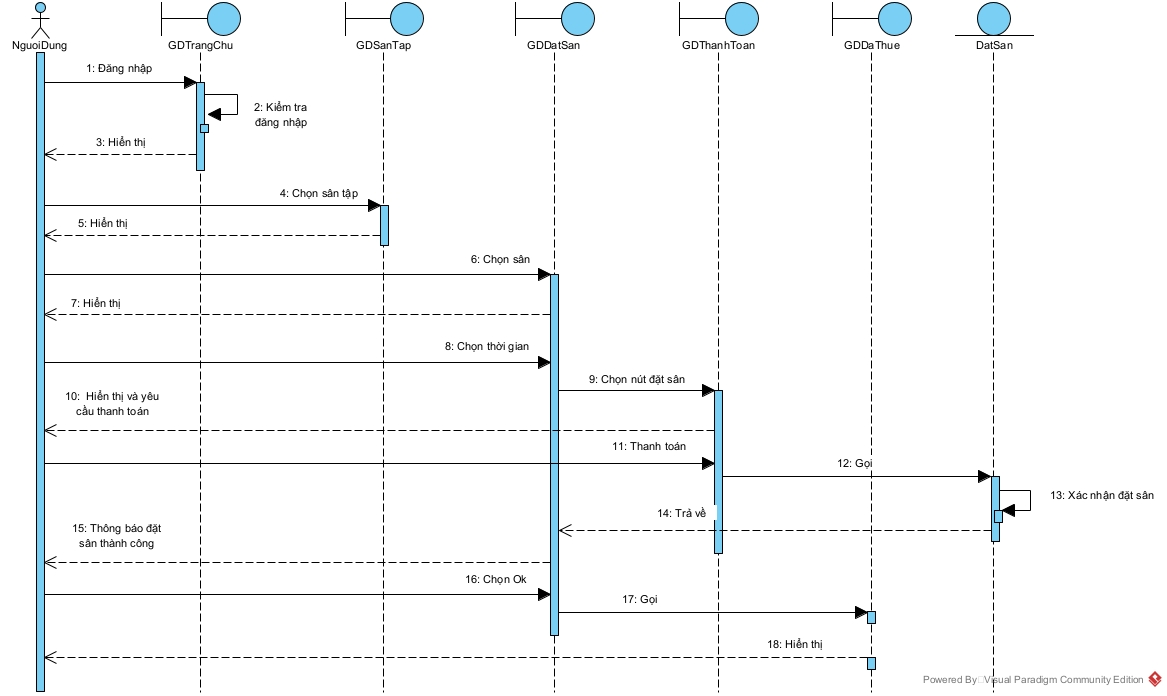
- Đổi mật khẩu



*Hình 2.12 SQ đổi mật khẩu*

|  |  |
| --- | --- |
| Tên usecase | Đổi mật khẩu |
| Tên actor | User |
| Tiền điều kiện | User muốn đổi mật khẩu |
| Hậu điều kiện | User đổi mật khẩu thành công |
| Kịch bản | 1. User truy cập vào giao diện đổi mật khẩu của hệ thống. 2. Hệ thống hiển thị ra 1 khung gồm mật khẩu hiện tại, mật khẩu mới và nhập lại mật khẩu. 3. User nhập đầy đủ thông tin và ấn vào nút “đổi mật khẩu” 4. Hệ thống thông báo “đổi mật khẩu thành công” |
| Ngoại lệ | 3.1.1 User không nhập đầy đủ thông tin nhưng vẫn ấn nút “đổi mật khẩu”  3.1.2 Hệ thống thông báo “Vui lòng nhập đầy đủ thông tin”  4.1 Hệ thống thông báo “Sai mật khẩu hiện tại”  4.2 Hệ thống thông báo “Nhập sai mật khẩu mới” |

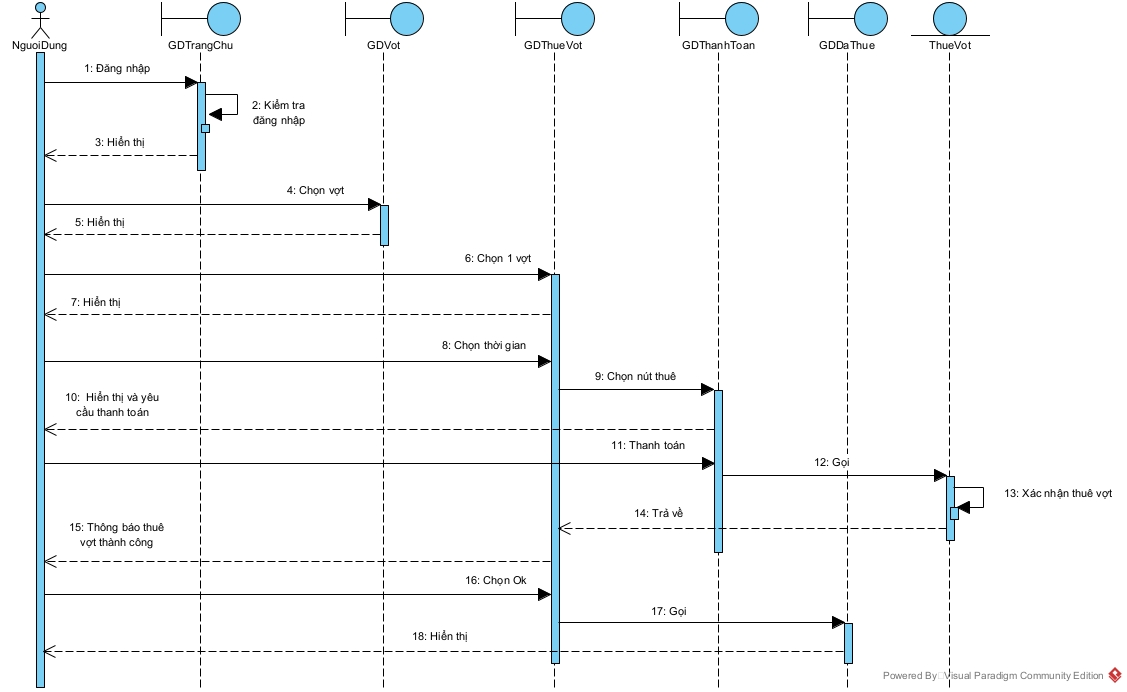
- Đặt sân



*Hình 2.13 SQ đặt sân*

|  |  |
| --- | --- |
| Tên usecase | Đặt sân |
| Tên actor | User |
| Tiền điều kiện | User muốn đặt sân |
| Hậu điều kiện | User đặt sân thành công |
| Kịch bản | 1. User truy cập vào giao diện đặt sân của hệ thống. 2. Hệ thống hiển thị ra các sân và loại sân. 3. User chọn 1 sân. 4. Hệ thống hiển thị đầy đủ thông tin của sân đó và nút đặt sân. 5. User chọn thời gian đặt và ấn vào nút đặt sân. 6. Hệ thống hiện ra giao diện thanh toán. 7. User thanh toán và ấn nút OK. 8. Hệ thống sẽ hiển thị danh sách đã thuê và trạng thái. |
| Ngoại lệ | 5.1.1 User không chọn thời gian đặt và ấn nút đặt sân.  5.1.2 Hệ thống thông báo “Vui lòng chọn thời gian” |

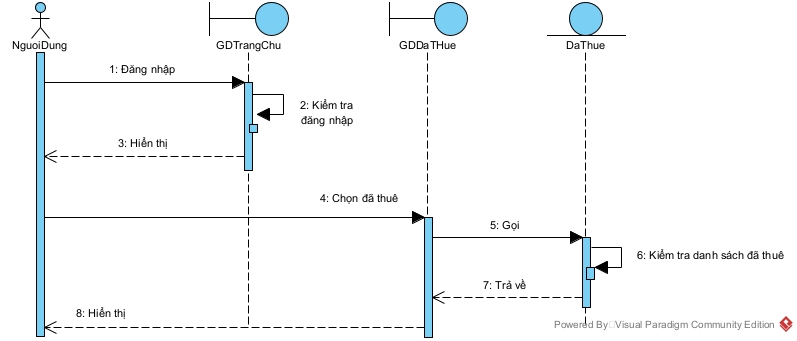
- Thuê vợt



*Hình 2.14 SQ thuê vợt*

|  |  |
| --- | --- |
| Tên usecase | Thuê vợt |
| Tên actor | User |
| Tiền điều kiện | User muốn thuê vợt |
| Hậu điều kiện | User thuê vợt thành công |
| Kịch bản | 1. User truy cập vào giao diện thuê vợt của hệ thống. 2. Hệ thống hiển thị ra các vợt và loại vợt. 3. User chọn 1 vợt. 4. Hệ thống hiển thị đầy đủ thông tin của vợt đó và nút đặt. 5. User chọn thời gian đặt và ấn vào nút đặt. 6. Hệ thống hiện ra giao diện thanh toán. 7. User thanh toán và ấn nút OK. 8. Hệ thống sẽ hiển thị danh sách đã thuê và trạng thái. |
| Ngoại lệ | 5.1.1 User không chọn thời gian đặt và ấn nút đặt.  5.1.2 Hệ thống thông báo “Vui lòng chọn thời gian” |

- Xem danh sách đã thuê

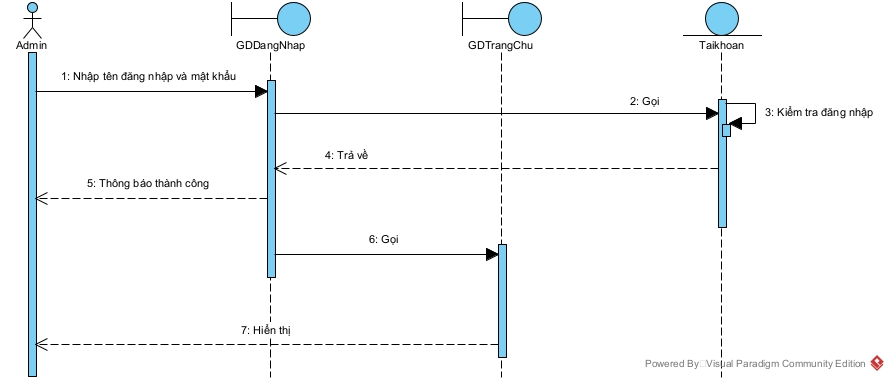


*Hình 2.15 SQ xem danh sách đã thuê*

|  |  |
| --- | --- |
| Tên usecase | Xem danh sách đã thuê |
| Tên actor | User |
| Tiền điều kiện | User xem danh sách đã thuê |
| Hậu điều kiện | Hệ thống hiển thị danh sách đã thuê |
| Kịch bản | 1. User truy cập vào giao diện đã thuê của hệ thống 2. Hệ thống hiển thị danh sách đã thuê |
| Ngoại lệ |  |

1. Admin

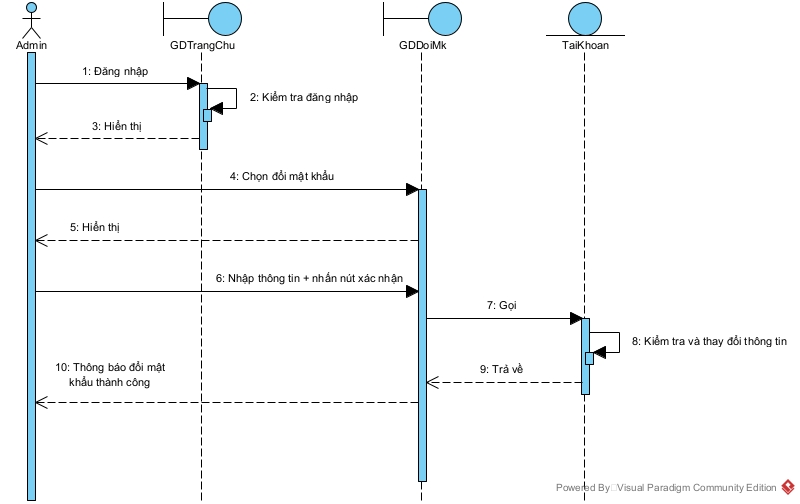
- Đăng nhập



*Hình 2.16 SQ đăng nhập admin*

|  |  |
| --- | --- |
| Tên usecase | Đăng nhập |
| Tên actor | Admin |
| Tiền điều kiện | Admin muốn đăng nhập |
| Hậu điều kiện | Admin đăng nhập thành công |
| Kịch bản | 1. Admin truy cập vào giao diện đăng nhập của hệ thống. 2. Hệ thống hiển thị ra 1 khung gồm tên đăng nhập và mật khẩu. 3. Admin nhập tên đăng nhập và mật khẩu và ấn vào nút “đăng nhập” 4. Hệ thống thông báo “đăng nhập thành công” và vào giao diện trang chủ. |
| Ngoại lệ | 3.1.1 Admin không nhập tên đăng nhập hoặc mật khẩu nhưng vẫn ấn nút “đăng nhập”  3.1.2 Hệ thống thông báo “Vui lòng nhập tên đăng nhập hoặc mật khẩu”  4.1 Hệ thống thông báo “Sai mật khẩu hoặc tên đăng nhập” |

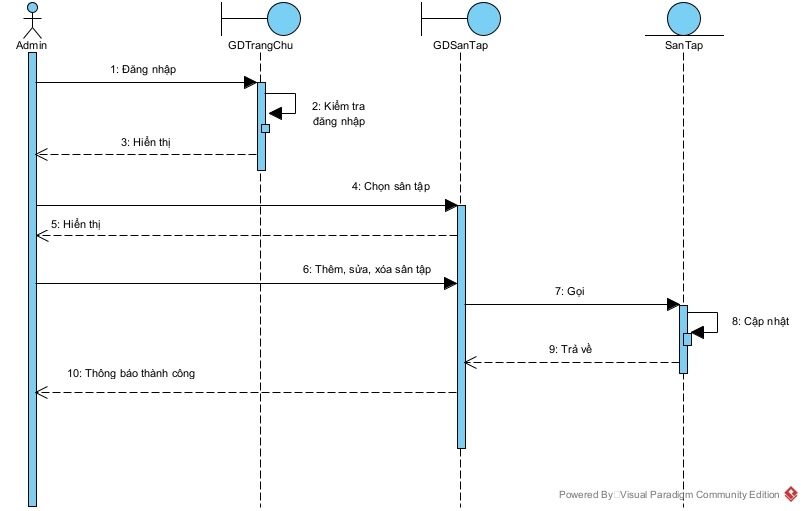
- Đổi mật khẩu



*Hình 2.17 SQ đổi mật khẩu admin*

|  |  |
| --- | --- |
| Tên usecase | Đổi mật khẩu |
| Tên actor | Admin |
| Tiền điều kiện | Admin muốn đổi mật khẩu |
| Hậu điều kiện | Admin đổi mật khẩu thành công |
| Kịch bản | 1. Admin truy cập vào giao diện đổi mật khẩu của hệ thống. 2. Hệ thống hiển thị ra 1 khung gồm mật khẩu hiện tại, mật khẩu mới và nhập lại mật khẩu. 3. Admin nhập đầy đủ thông tin và ấn vào nút “đổi mật khẩu” 4. Hệ thống thông báo “đổi mật khẩu thành công” |
| Ngoại lệ | 3.1.1 Admin không nhập đầy đủ thông tin nhưng vẫn ấn nút “đổi mật khẩu”  3.1.2 Hệ thống thông báo “Vui lòng nhập đầy đủ thông tin”  4.1 Hệ thống thông báo “Sai mật khẩu hiện tại”  4.2 Hệ thống thông báo “Nhập sai mật khẩu mới” |

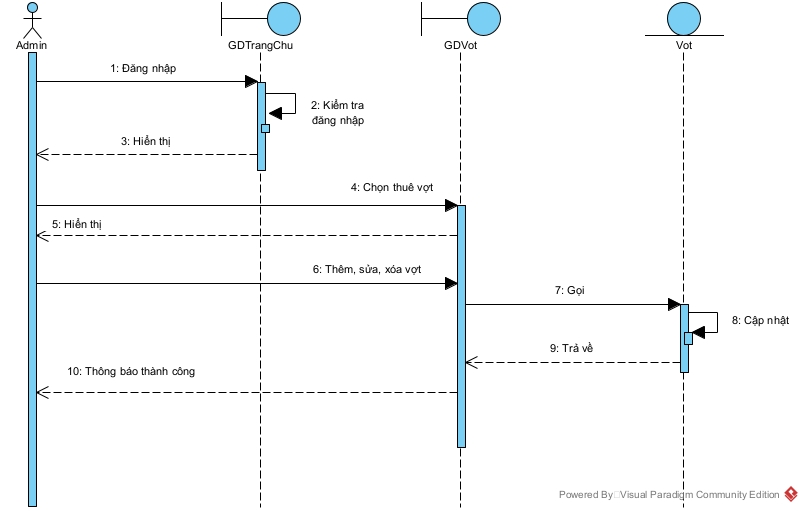
- Quản lí sân



*Hình 2.18 SQ quản lí sân*

|  |  |
| --- | --- |
| Tên usecase | Quản lí sân |
| Tên actor | Admin |
| Tiền điều kiện | Admin thêm, sửa, xóa sân |
| Hậu điều kiện | Admin thêm sửa, xóa sân thành công |
| Kịch bản | 1. Admin truy cập vào giao diện quản lí sân.  2. Hệ thống hiển thị ra danh sách các sân và loại sân.  3. Admin thêm, sửa, xóa 1 sân hoặc loại sân.  4. Hệ thống thông báo “cập nhật thành công” |
| Ngoại lệ | 3.1.1 Admin khi thêm hoặc sửa 1 sân thì phần giá tiền số âm.  3.1.2 Hệ thống thao báo số tiền không được phép âm. |

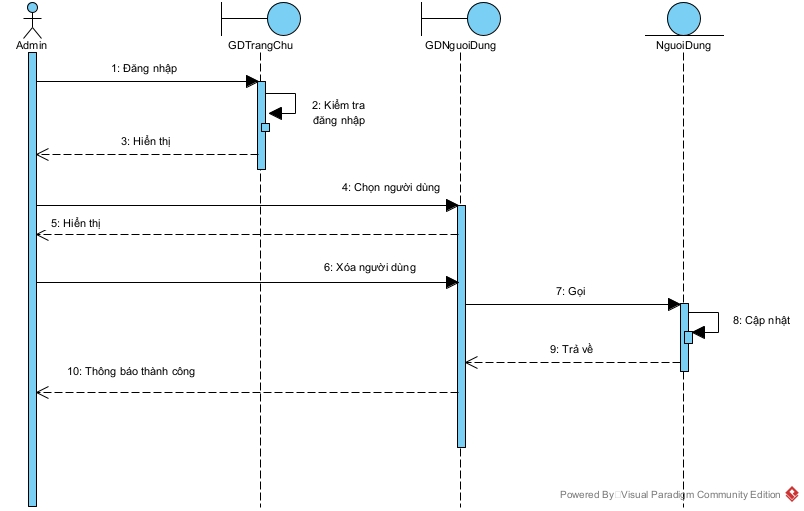
- Quản lí vợt



*Hình 2.19 SQ quản lí vợt*

|  |  |
| --- | --- |
| Tên usecase | Quản lí vợt |
| Tên actor | Admin |
| Tiền điều kiện | Admin thêm, sửa, xóa vợt |
| Hậu điều kiện | Admin thêm sửa, xóa vợt thành công |
| Kịch bản | 1. Admin truy cập vào giao diện quản lí vợt.  2. Hệ thống hiển thị ra danh sách các vợt và loại vợt.  3. Admin thêm, sửa, xóa 1 vợt hoặc loại vợt.  4. Hệ thống thông báo “cập nhật thành công” |
| Ngoại lệ | 3.1.1 Admin khi thêm hoặc sửa 1 vợt thì phần giá tiền số âm.  3.1.2 Hệ thống thao báo số tiền không được phép âm. |

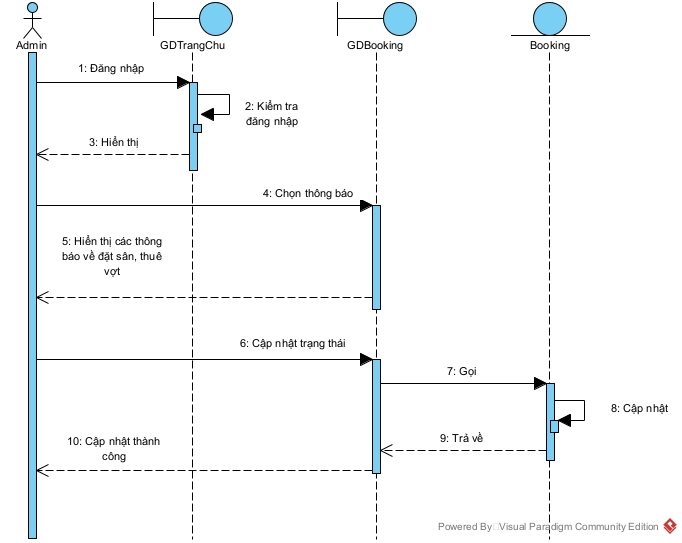
- Quản lí khách hàng



*Hình 2.20 SQ quản lí khách hàng*

|  |  |
| --- | --- |
| Tên usecase | Quản lí khách hàng |
| Tên actor | Admin |
| Tiền điều kiện | Admin xóa khách hàng |
| Hậu điều kiện | Admin xóa khách hàng thành công |
| Kịch bản | 1. Admin truy cập vào giao diện quản lí vợt.  2. Hệ thống hiển thị ra danh sách các khách hàng.  3. Admin xóa 1 khách hàng.  4. Hệ thống thông báo “cập nhật thành công” |
| Ngoại lệ |  |

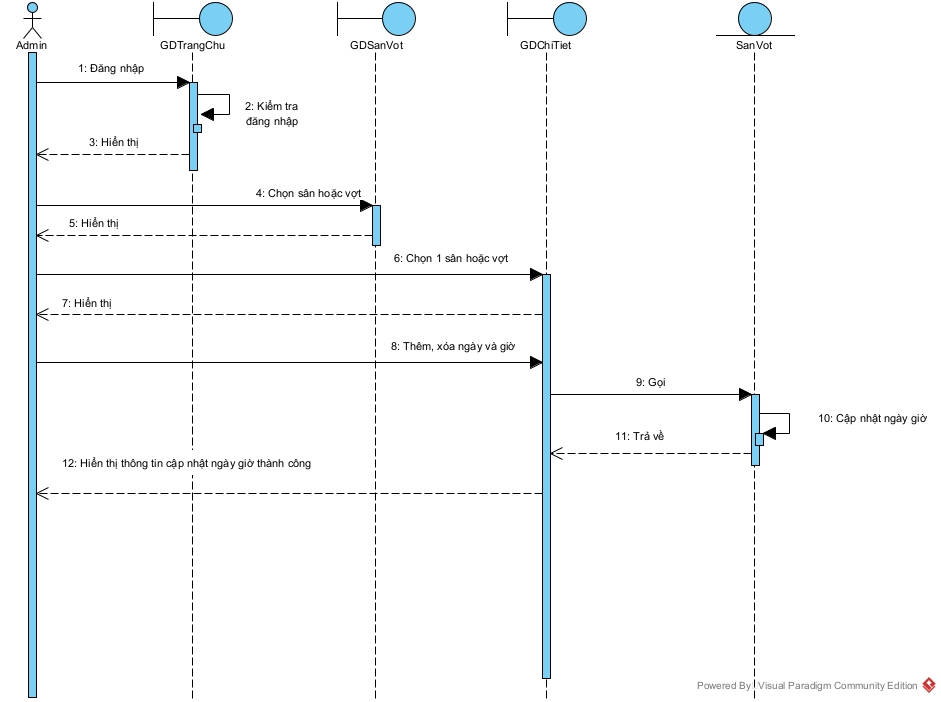
- Quản lí đơn đặt



*Hình 2.21 SQ Quản lí đơn đặt*

|  |  |
| --- | --- |
| Tên usecase | Quản lí booking |
| Tên actor | Admin |
| Tiền điều kiện | Admin cập nhật booking |
| Hậu điều kiện | Admin cập nhật booking thành công |
| Kịch bản | 1. Admin truy cập vào giao diện quản lí booking.  2. Hệ thống hiển thị ra danh sách các booking.  3. Admin cập nhật trạng thái của booking.  4. Hệ thống thông báo “cập nhật thành công” |
| Ngoại lệ |  |

- Quản lí ngày giờ

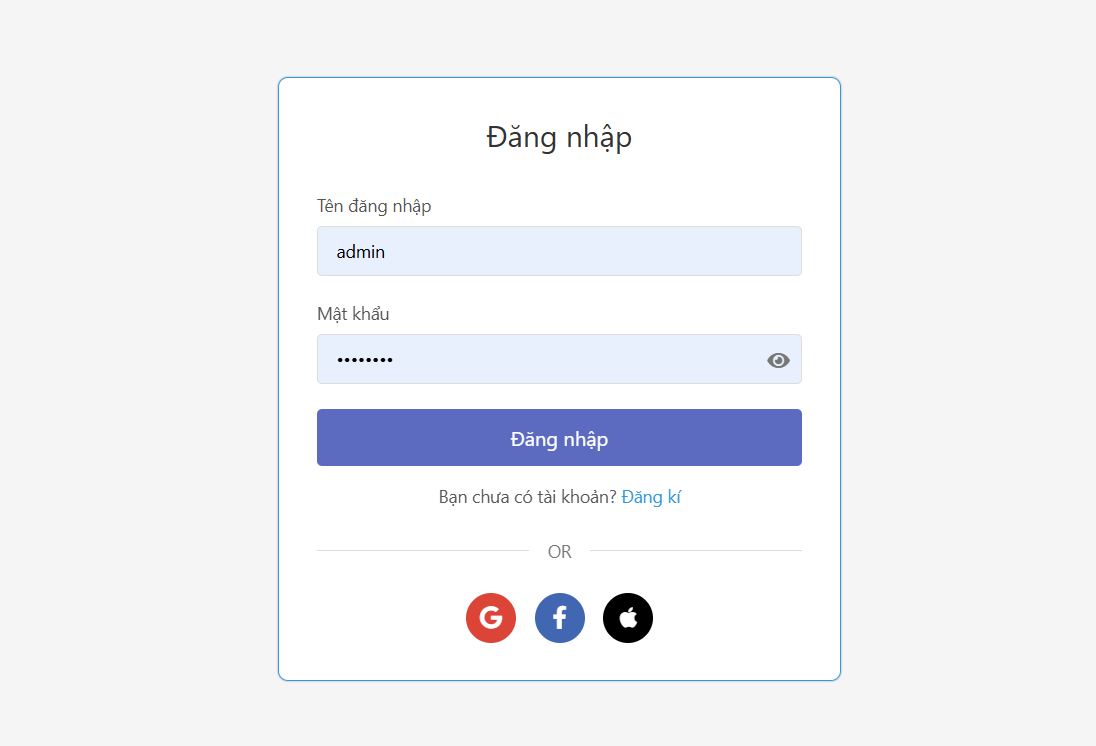


*Hình 2.22 SQ quản lí ngày giờ*

|  |  |
| --- | --- |
| Tên usecase | Quản lí ngày giờ |
| Tên actor | Admin |
| Tiền điều kiện | Admin cập nhật ngày giờ |
| Hậu điều kiện | Admin cập nhật ngày giờ thành công |
| Kịch bản | 1. Admin truy cập vào giao diện sân hoặc vợt.  2. Hệ thống hiển thị ra danh sách các sân hoặc vợt.  3. Admin chọn 1 sân hoặc vợt.  4. Hệ thống hiển thị chi tiết thông tin  5. Admin thêm, xóa ngày và giờ  6. Hệ thống thông báo cập nhật thành công |
| Ngoại lệ | * + 1. Admin xóa 1 giờ đã được đặt     2. Hệ thống thông báo không thể xóa giờ đã được đặt   5.2.1 Hệ thống thêm 1 giờ nhưng giờ bắt đầu lớn hơn giờ kết thúc  5.2.2 Hệ thống thông báo giờ kết thúc phải lớn hơn giờ bắt đầu |

Chương 3: Thiết kế chi tiết API

1. Giao diện đăng nhập



*Hình 3.1 Giao diện đăng nhập*

- POST /api/users/login: Đăng nhập người dùng

requestBody:

{

username [required]: string, // Tên đăng nhập

password [required]: string // Mật khẩu

}

{

message: string, // Thông báo kết quả đăng nhập

token: string, // JWT token để xác thực

user: {

id: integer, // ID của người dùng

username: string, // Tên đăng nhập

firstName: string, // Tên

lastName: string, // Họ

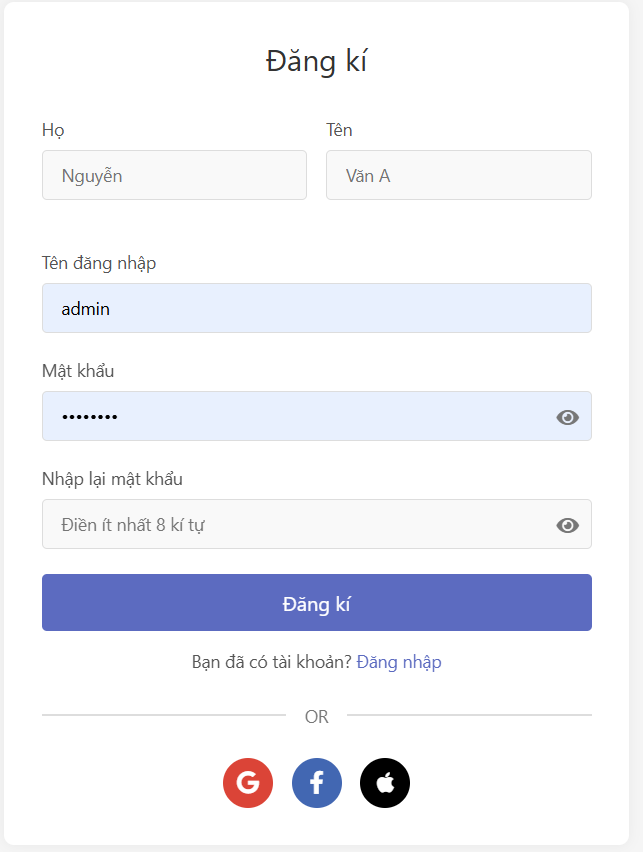
role: string // Vai trò (ADMIN/USER)

}

}

Sau khi đăng nhập thành công, hệ thống sẽ tạo ra một JWT (JSON Web Token) - một chuỗi mã hóa chứa thông tin người dùng, được sử dụng để xác thực và phân quyền trong các request tiếp theo, giúp bảo mật và quản lý phiên đăng nhập của người dùng một cách an toàn và hiệu quả.

1. Giao diện đăng kí



*Hình 3.2 Giao diện đăng kí*

- POST /api/users/register: Đăng ký tài khoản người dùng mới

requestBody:

{

lastName [required]: string, // Họ của người dùng

firstName [required]: string, // Tên của người dùng

username [required]: string, // Tên đăng nhập (email)

password [required]: string, // Mật khẩu (ít nhất 8 ký tự)

confirmPassword [required]: string // Xác nhận mật khẩu

}

{

message: string, // Thông báo kết quả đăng ký

user: {

id: integer, // ID của người dùng

lastName: string, // Họ

firstName: string, // Tên

username: string, // Tên đăng nhập

role: string // Vai trò (mặc định là "USER")

}

}

1. Giao diện đổi mật khẩu



*Hình 3.3 Giao diện đổi mật khẩu*

- POST /api/users/change-password: Đổi mật khẩu người dùng

requestBody:

{

userId [required]: integer, // ID của người dùng

currentPassword [required]: string, // Mật khẩu hiện tại

newPassword [required]: string, // Mật khẩu mới (ít nhất 8 ký tự)

confirmPassword [required]: string // Xác nhận mật khẩu mới

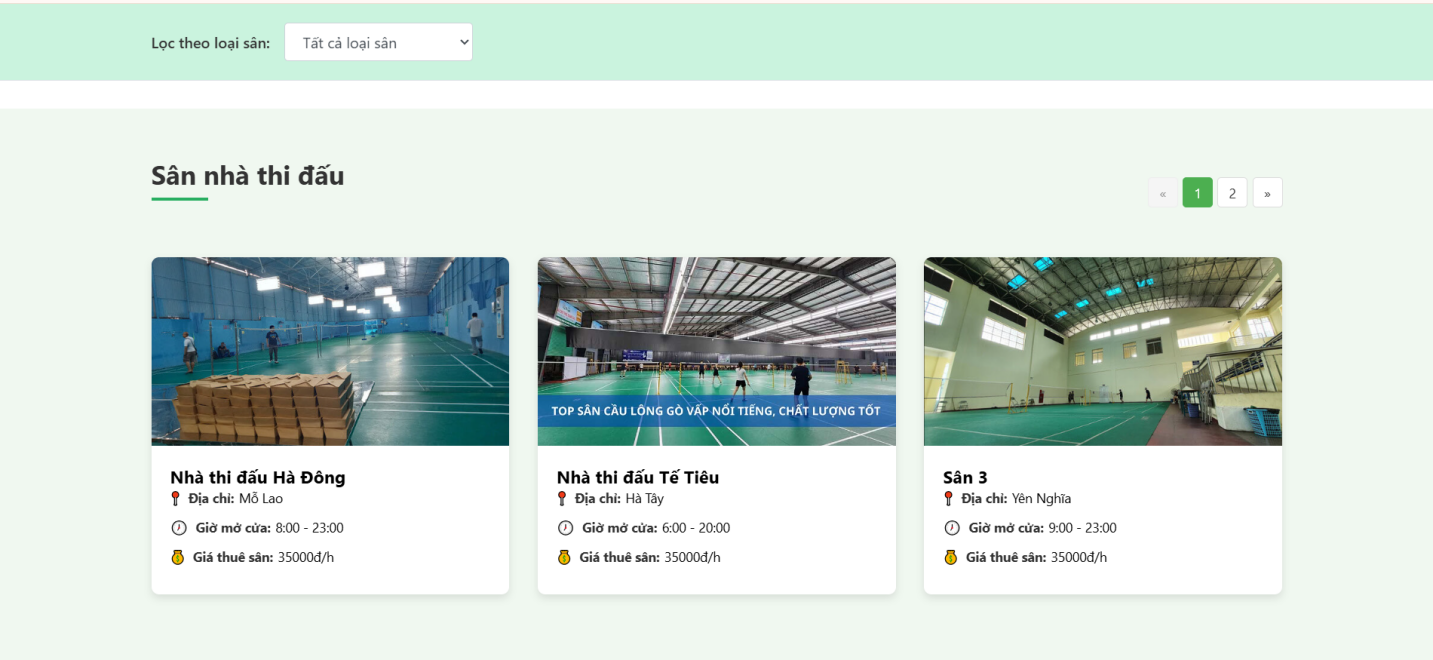
}

{

message: string // Thông báo kết quả đổi mật khẩu

}

1. Giao diện sân tập



*HÌnh 3.4 Giao diện sân tập*

- GET /api/court-types: Lấy danh sách tất cả loại sân

Headers:

{

'Authorization': 'Bearer <token>'

}

Output:

[

{

id: integer, // ID của loại sân

name: string // Tên loại sân

}

]

- GET /api/court-types/paginated: Lấy danh sách loại sân có phân trang

Input:

page: integer (default: 0) // Số trang

size: integer (default: 3) // Số loại sân mỗi trang

Headers:

{

'Authorization': 'Bearer <token>'

}

Output:

{

courtTypes: [ // Danh sách loại sân

{

id: integer,

name: string

}

],

currentPage: integer, // Trang hiện tại

totalPages: integer // Tổng số trang

}

- GET /api/courts/type/{courtTypeId}: Lấy danh sách sân theo loại

Input:

courtTypeId [required]: integer // ID của loại sân

Headers:

{

'Authorization': 'Bearer <token>'

}

Output:

[

{

id: integer, // ID của sân

name: string, // Tên sân

address: string, // Địa chỉ

imageUrl: string, // URL hình ảnh

hours: string, // Giờ mở cửa

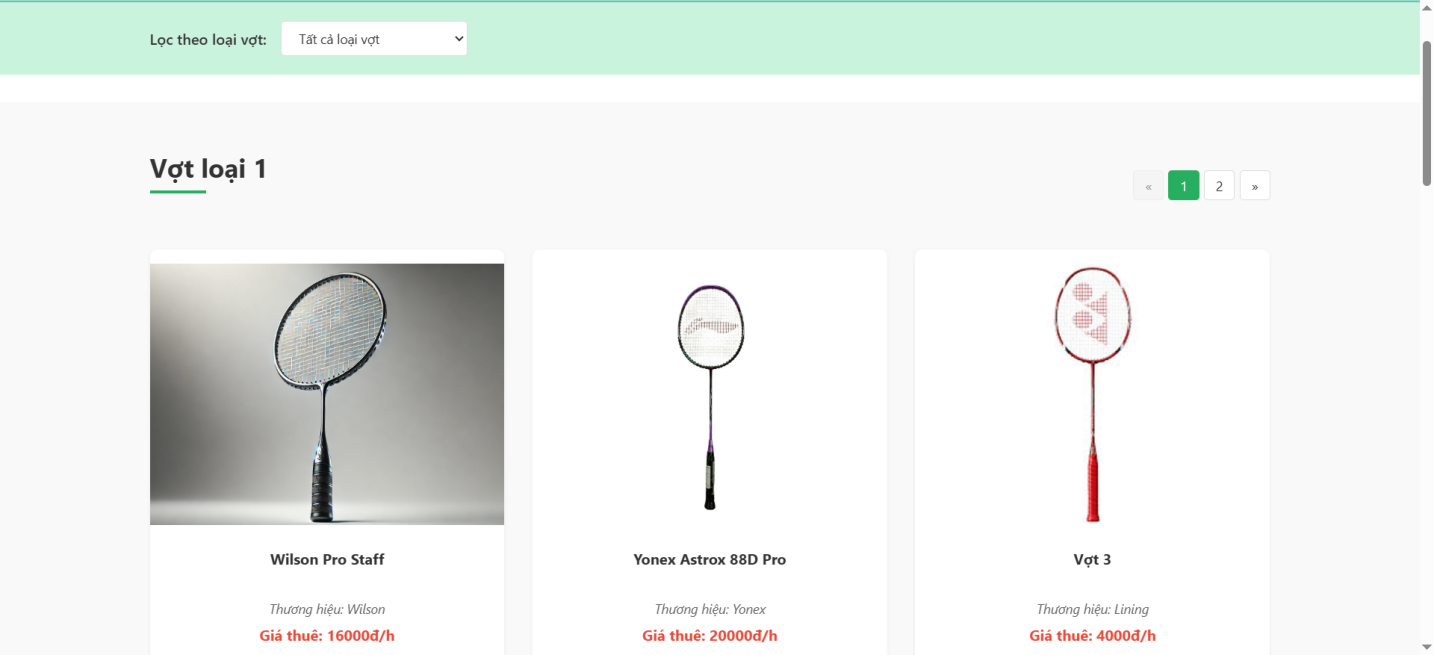
price: string, // Giá thuê sân

courtTypeName: string // Tên loại sân

}

]

1. Giao diện vợt



*Hình 3.5 Giao diện vợt*

- GET /api/racket-types: Lấy danh sách tất cả loại vợt

Headers:

{

'Authorization': 'Bearer <token>'

}

Output:

[

{

id: integer, // ID của loại vợt

name: string // Tên loại vợt

}

]

- GET /api/racket-types/paginated: Lấy danh sách loại vợt có phân trang

Input:

page: integer (default: 0) // Số trang

size: integer (default: 3) // Số loại vợt mỗi trang

Headers:

{

'Authorization': 'Bearer <token>'

}

Output:

{

racketTypes: [ // Danh sách loại vợt

{

id: integer,

name: string

}

],

currentPage: integer, // Trang hiện tại

totalPages: integer // Tổng số trang

}

- GET /api/rackets/type/{racketTypeId}: Lấy danh sách vợt theo loại

Input:

racketTypeId [required]: integer // ID của loại vợt

Headers:

{

'Authorization': 'Bearer <token>'

}

Output:

[

{

id: integer, // ID của vợt

name: string, // Tên vợt

brand: string, // Thương hiệu

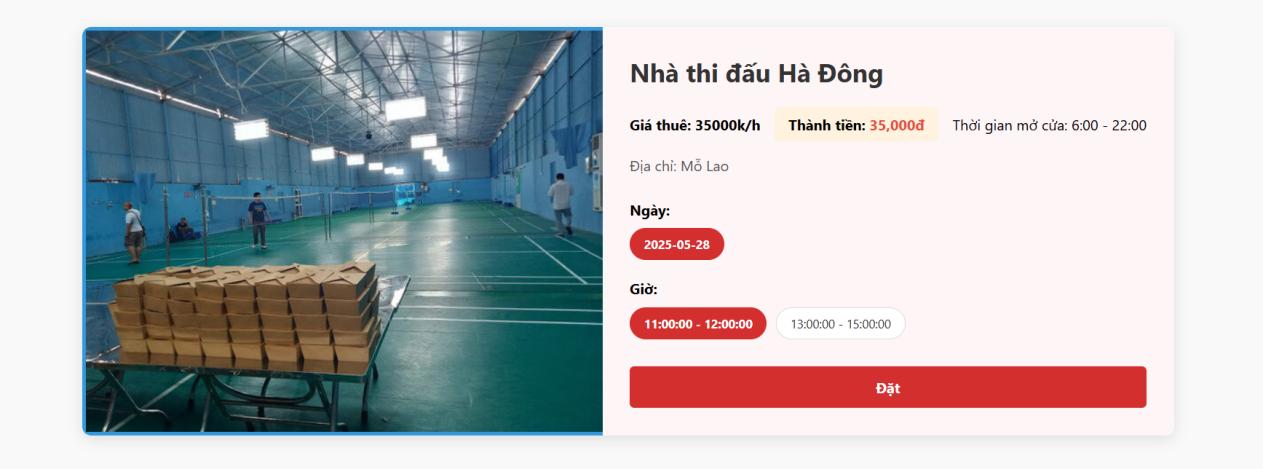
imageUrl: string, // URL hình ảnh

price: string // Giá thuê vợt

}

]

1. Giao diện đặt sân/vợt



*Hình 3.6 Giao diện đặt sân/vợt*

- GET /api/{type}s/{id}: Lấy thông tin chi tiết sân/vợt

Input:

type: string ('court' hoặc 'racket')

id: integer (ID của sân/vợt)

Headers:

{

'Authorization': 'Bearer <token>'

}

Output:

{

id: integer, // ID của sân/vợt

name: string, // Tên sân/vợt

price: string, // Giá thuê

imageUrl: string, // URL hình ảnh

address: string, // Địa chỉ (chỉ cho sân)

openingHours: string, // Giờ mở cửa (chỉ cho sân)

brand: string // Thương hiệu (chỉ cho vợt)

}

- GET /api/days/available/court/{courtId}: Lấy danh sách ngày có sẵn cho sân

Input:

courtId: integer (ID của sân)

Headers:

{

'Authorization': 'Bearer <token>'

}

Output:

[

{

id: integer, // ID của ngày

date: string // Ngày (format: YYYY-MM-DD)

}

]

- GET /api/days1/available/racket/{racketId}: Lấy danh sách ngày có sẵn cho vợt

Input:

racketId: integer (ID của vợt)

Headers:

{

'Authorization': 'Bearer <token>'

}

Output:

[

{

id: integer, // ID của ngày

date: string // Ngày (format: YYYY-MM-DD)

}

]

- GET /api/time-slots/day/{dayId}: Lấy danh sách khung giờ cho sân

Input:

dayId: integer (ID của ngày)

Headers:

{

'Authorization': 'Bearer <token>'

}

Output:

[

{

id: integer, // ID của khung giờ

startTime: string, // Giờ bắt đầu

endTime: string, // Giờ kết thúc

booked: boolean // Trạng thái đã đặt

}

]

- GET /api/timeslots1/day1/{dayId}: Lấy danh sách khung giờ cho vợt

Input:

dayId: integer (ID của ngày)

Headers:

{

'Authorization': 'Bearer <token>'

}

Output:

[

{

id: integer, // ID của khung giờ

startTime: string, // Giờ bắt đầu

endTime: string, // Giờ kết thúc

booked: boolean // Trạng thái đã đặt

}

]

- POST /api/bookings: Tạo đơn đặt sân/vợt

Input:

{

userId: integer, // ID người dùng

bookingType: string, // Loại đặt ('COURT' hoặc 'RACKET')

itemName: string, // ID sân/vợt

itemDate: string, // Ngày đặt

itemTime: string, // Thời gian đặt

timeSlotIds: integer[], // Danh sách ID khung giờ (cho sân)

timeSlot1Ids: integer[] // Danh sách ID khung giờ (cho vợt)

}

Headers:

{

'Content-Type': 'application/json',

'Authorization': 'Bearer <token>'

}

Output:

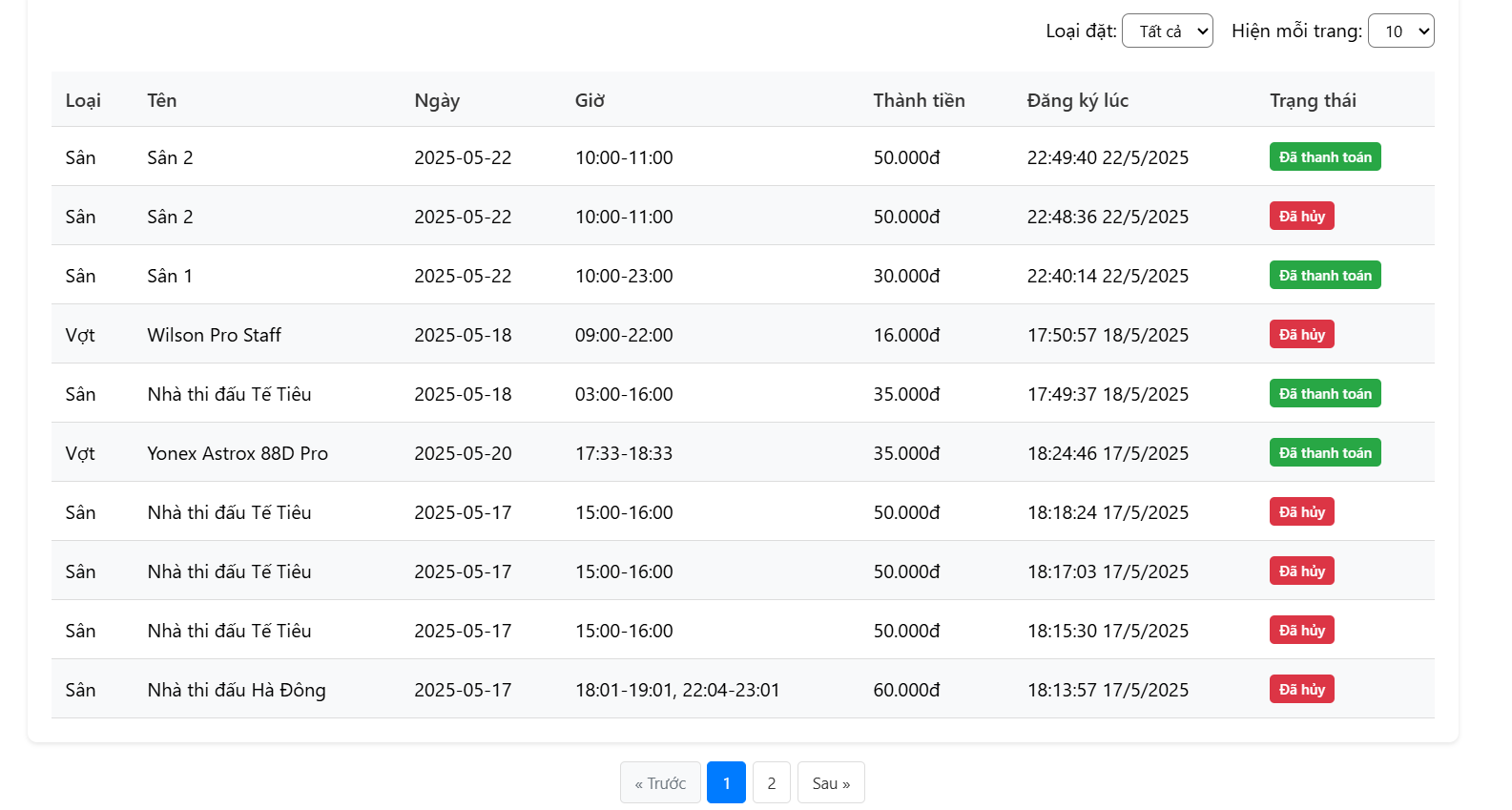
{

success: boolean, // Trạng thái tạo đơn

message: string // Thông báo

}

1. Giao diện xem danh sách đã thuê



*Hình 2.7 Giao diện xem danh sách đã thuê*

- GET /api/bookings/user/{userId}/paginated: Lấy danh sách đơn đặt của người dùng có phân trang

Input:

userId: integer (ID của người dùng)

page: integer (Số trang)

size: integer (Số đơn mỗi trang)

sortBy: string (Sắp xếp theo trường)

sortDirection: string (Hướng sắp xếp)

type: string (Loại đơn - COURT hoặc RACKET, tùy chọn)

Headers:

{

'Authorization': 'Bearer <token>'

}

Output:

{

bookings: [ // Danh sách đơn đặt

{

id: integer, // ID đơn

bookingType: string, // Loại đơn (COURT/RACKET)

itemName: string, // ID sân/vợt

itemDate: string, // Ngày đặt

itemTime: string, // Thời gian đặt

totalPrice: number, // Tổng tiền

bookingDate: string, // Thời gian đăng ký

status: string // Trạng thái (PENDING/SUCCESS/FAILED)

}

],

currentPage: integer, // Trang hiện tại

totalPages: integer // Tổng số trang

}

- GET /api/courts/{courtId}: Lấy thông tin chi tiết sân

Input:

courtId: integer (ID của sân)

Headers:

{

'Authorization': 'Bearer <token>'

}

Output:

{

id: integer, // ID của sân

name: string, // Tên sân

price: string, // Giá thuê

imageUrl: string, // URL hình ảnh

address: string, // Địa chỉ

openingHours: string // Giờ mở cửa

}

- GET /api/rackets/{racketId}: Lấy thông tin chi tiết vợt

Input:

racketId: integer (ID của vợt)

Headers:

{

'Authorization': 'Bearer <token>'

}

Output:

{

id: integer, // ID của vợt

name: string, // Tên vợt

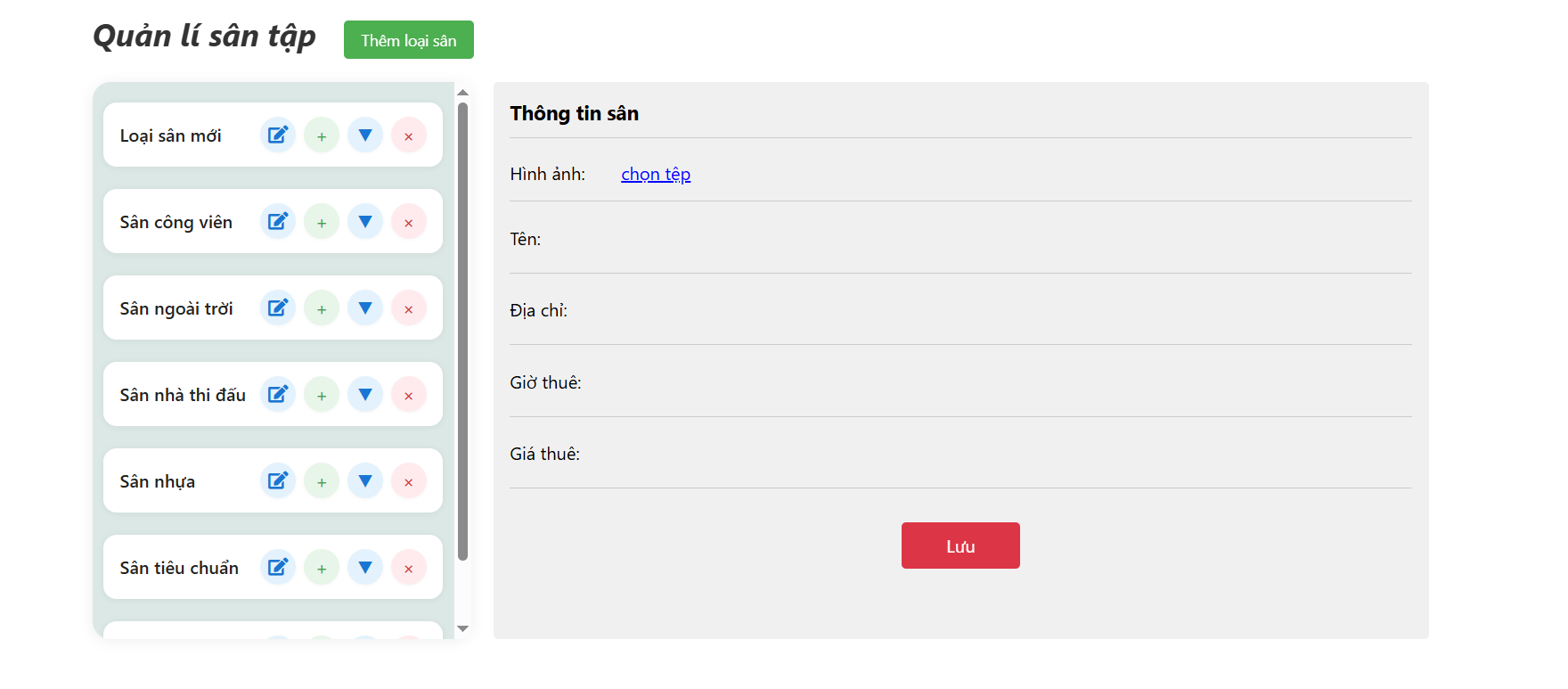
price: string, // Giá thuê

imageUrl: string, // URL hình ảnh

brand: string // Thương hiệu

}

1. Giao diện quản lí sân



*HÌnh 2.8 Giao diện quản lí sân*

- GET /api/court-types: Lấy danh sách loại sân

// Output

[

{

"id": integer, // ID loại sân

"name": string // Tên loại sân

}

]

- POST /api/court-types: Tạo mới loại sân

// Input

{

"name": string // Tên loại sân

}

// Output

{

"id": integer,

"name": string

}

- PUT /api/court-types/{courtTypeId}: Sửa loại sân

// Input

{

"name": string // Tên loại sân mới

}

// Output

{

"id": integer,

"name": string

}

- DELETE /api/court-types/{courtTypeId}: Xóa loại sân

// Output (thành công)

204 No Content

// Output (thất bại)

{

"message": "Không thể xóa loại sân vì vẫn còn sân trong loại này"

}

- GET /api/courts/type/{courtTypeId}: Lấy danh sách sân theo loại

// Output

[

{

"id": integer, // ID sân

"name": string, // Tên sân

"address": string, // Địa chỉ

"hours": string, // Giờ mở cửa

"price": string, // Giá thuê

"imageUrl": string, // URL hình ảnh

"courtTypeId": integer // ID loại sân

}

]

- POST /api/courts: Tạo mới sân

// Input

{

"name": string,

"address": string,

"hours": string,

"price": string,

"courtTypeId": integer,

"imageUrl": string

}

// Output

{

"id": integer,

"name": string,

"address": string,

"hours": string,

"price": string,

"courtTypeId": integer,

"imageUrl": string

}

- PUT /api/courts/{courtId}: Sửa sân

// Input

{

"name": string,

"address": string,

"hours": string,

"price": string,

"imageUrl": string

}

// Output

{

"id": integer,

"name": string,

"address": string,

"hours": string,

"price": string,

"courtTypeId": integer,

"imageUrl": string

}

- DELETE /api/courts/{courtId}: Xóa sân

// Output (thành công)

204 No Content

// Output (thất bại)

{

"message": "Không thể xóa sân: Lỗi ..."

}

- GET /api/courts/{courtId}; Lấy chi tiết sân

// Output

{

"id": integer,

"name": string,

"address": string,

"hours": string,

"price": string,

"courtTypeId": integer,

"imageUrl": string

}

- POST /api/upload

// Input: FormData với trường "image" (file ảnh)

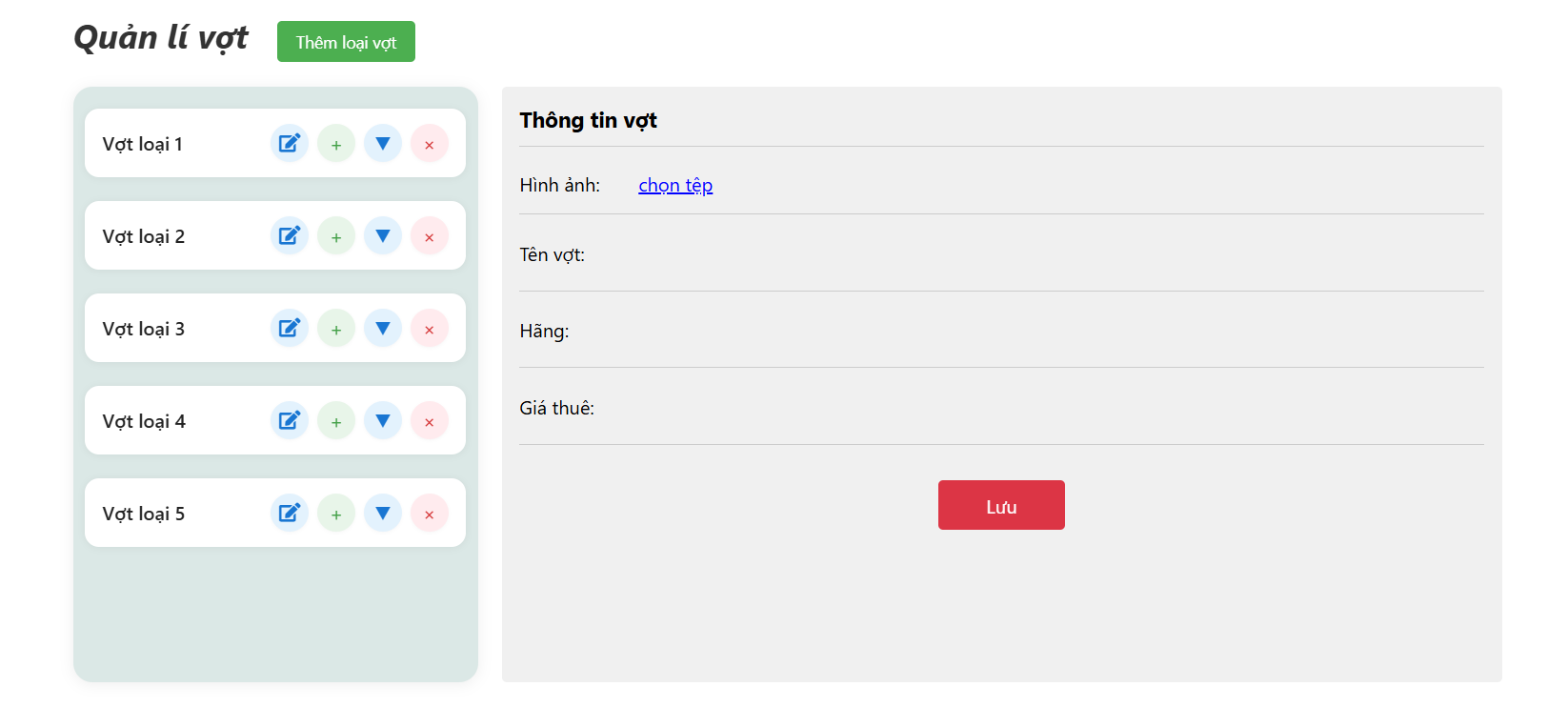
// Output

{

"imageUrl": string // Đường dẫn ảnh vừa upload

}

1. Quản lí vợt



*Hình 3.9 Giao diện quản lí vợt*

- GET /api/racket-types: Lấy danh sách loại vợt

// Output

[

{

"id": integer, // ID loại vợt

"name": string // Tên loại vợt

}

]

- POST /api/racket-types: Tạo mới loại vợt

// Input

{

"name": string // Tên loại vợt

}

// Output

{

"id": integer,

"name": string

}

- PUT /api/racket-types/{racketTypeId}: Sửa loại vợt

// Input

{

"name": string // Tên loại vợt mới

}

// Output

{

"id": integer,

"name": string

}

- DELETE /api/racket-types/{racketTypeId}: Xóa loại vợt

// Output (thành công)

204 No Content

// Output (thất bại)

{

"message": "Không thể xóa loại vợt vì vẫn còn vợt trong loại này"

}

- GET /api/rackets/type/{racketTypeId}: Lấy danh sách vợt theo loại

// Output

[

{

"id": integer, // ID vợt

"name": string, // Tên vợt

"brand": string, // Thương hiệu

"price": string, // Giá thuê

"imageUrl": string, // URL hình ảnh

"racketTypeId": integer // ID loại vợt

}

]

- POST /api/rackets: Tạo mới vợt

// Input

{

"name": string,

"brand": string,

"price": string,

"racketTypeId": integer,

"imageUrl": string

}

// Output

{

"id": integer,

"name": string,

"brand": string,

"price": string,

"racketTypeId": integer,

"imageUrl": string

}

- PUT /api/rackets/{racketId}: Sửa vợt

// Input

{

"name": string,

"brand": string,

"price": string,

"imageUrl": string

}

// Output

{

"id": integer,

"name": string,

"brand": string,

"price": string,

"racketTypeId": integer,

"imageUrl": string

}

- DELETE /api/rackets/{racketId}: Xóa vợt

// Output (thành công)

204 No Content

// Output (thất bại)

{

"message": "Không thể xóa vợt: Lỗi ..."

}

- GET /api/rackets/{racketId}: Lấy chi tiết vợt

// Output

{

"id": integer,

"name": string,

"brand": string,

"price": string,

"racketTypeId": integer,

"imageUrl": string

}

- POST /api/upload

// Input: FormData với trường "image" (file ảnh)

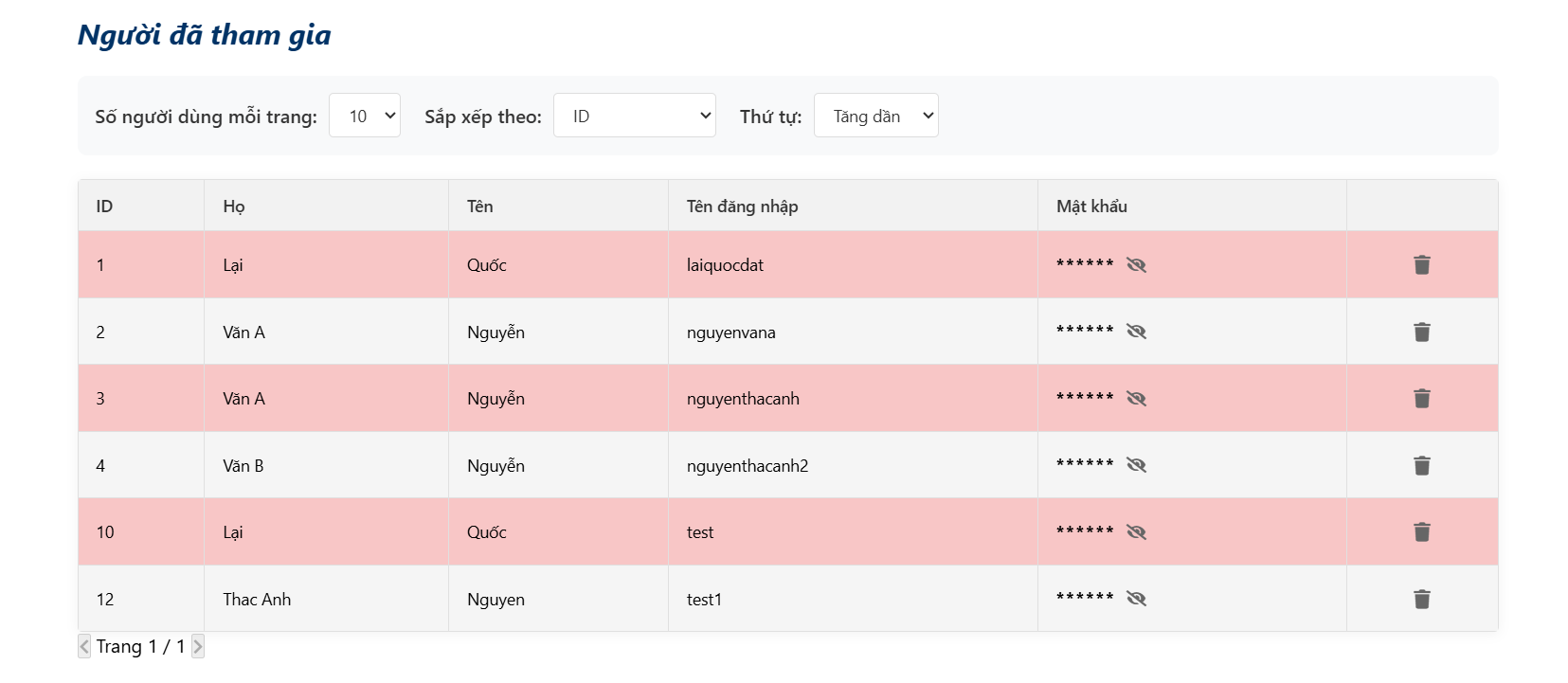
// Output

{

"imageUrl": string // Đường dẫn ảnh vừa upload

}

1. Quản lí người dùng



*3.10 Quản lí người dùng*

- GET /api/users/paginated: Lấy danh sách người dùng có phân trang

// Input (Query Parameters)

{

"page": integer, // Số trang (0-based)

"size": integer, // Số lượng người dùng mỗi trang

"sortBy": string, // Trường sắp xếp (id, username, etc.)

"sortDirection": string // Hướng sắp xếp (asc/desc)

}

// Output

{

"users": [

{

"id": integer, // ID người dùng

"lastName": string, // Họ

"firstName": string, // Tên

"username": string, // Tên đăng nhập

"password": string // Mật khẩu (đã mã hóa)

}

],

"currentPage": integer, // Trang hiện tại

"totalPages": integer // Tổng số trang

}

- DELETE /api/users/{userId}: Xóa người dùng

// Output (thành công)

{

"message": string // Thông báo thành công

}

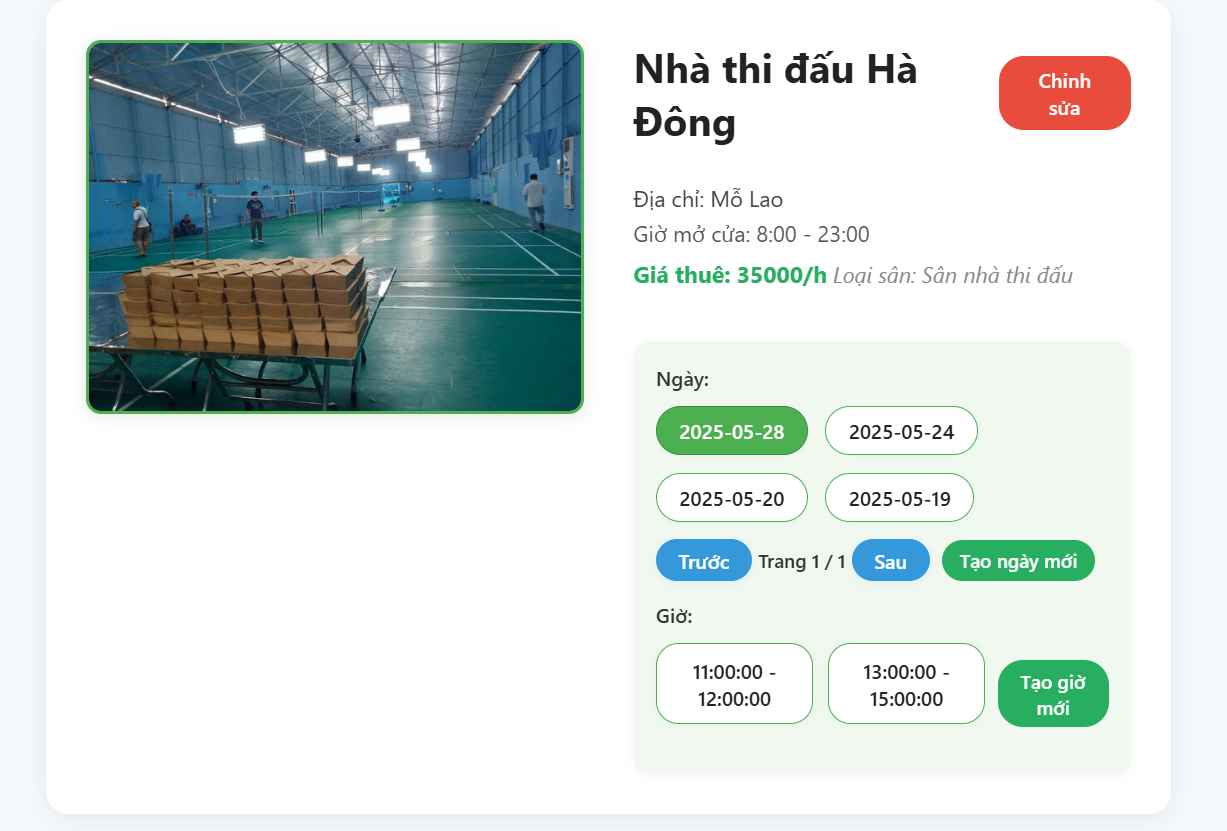
// Output (thất bại)

{

"message": string // Mô tả lỗi

}

1. Quản lý ngày giờ của sân



*Hình 3.11 Giao diện ngày giờ của sân*

- GET /api/court-types: Lấy danh sách loại sân

// Output

[

{

"id": integer, // ID loại sân

"name": string // Tên loại sân

}

]

- GET /api/courts/{courtId}: Lấy thông tin chi tiết sân

// Output

{

"id": integer,

"name": string,

"address": string,

"hours": string,

"price": string,

"courtTypeId": integer,

"imageUrl": string

}

- PUT /api/courts/{courtId}: Cập nhật thông tin sân

// Input

{

"name": string,

"address": string,

"hours": string,

"price": string,

"courtTypeId": integer,

"imageUrl": string

}

// Output

{

"id": integer,

"name": string,

"address": string,

"hours": string,

"price": string,

"courtTypeId": integer,

"imageUrl": string

}

- GET /api/days/court/{courtId}: Lấy danh sách ngày của sân

// Output

[

{

"id": integer,

"date": string, // Format: DD/MM/YYYY

"courtId": integer

}

]

- POST /api/days: Thêm ngày mới

// Input

{

"date": string, // Format: DD/MM/YYYY

"courtId": integer

}

// Output

{

"id": integer,

"date": string,

"courtId": integer

}

- DELETE /api/days/{dayId}: Xóa ngày

// Output (thành công)

204 No Content

// Output (thất bại)

{

"message": string // Đang có ngày trong sân nên không thể xóa

}

- GET /api/time-slots/day/{dayId}: Lấy danh sách khung giờ của ngày

// Output

[

{

"id": integer,

"time": string, // Format: HH:mm

"dayId": integer,

"isBooked": boolean,

"booking": {

"customer": string, // Mã sinh viên

"registeredAt": string // Thời gian đặt

}

}

]

- POST /api/time-slots: Thêm khung giờ mới

// Input

{

"time": string, // Format: HH:mm

"dayId": integer

}

// Output

{

"id": integer,

"time": string,

"dayId": integer,

"isBooked": false

}

- DELETE /api/time-slots/{timeSlotId}: Xóa khung giờ

// Output (thành công)

204 No Content

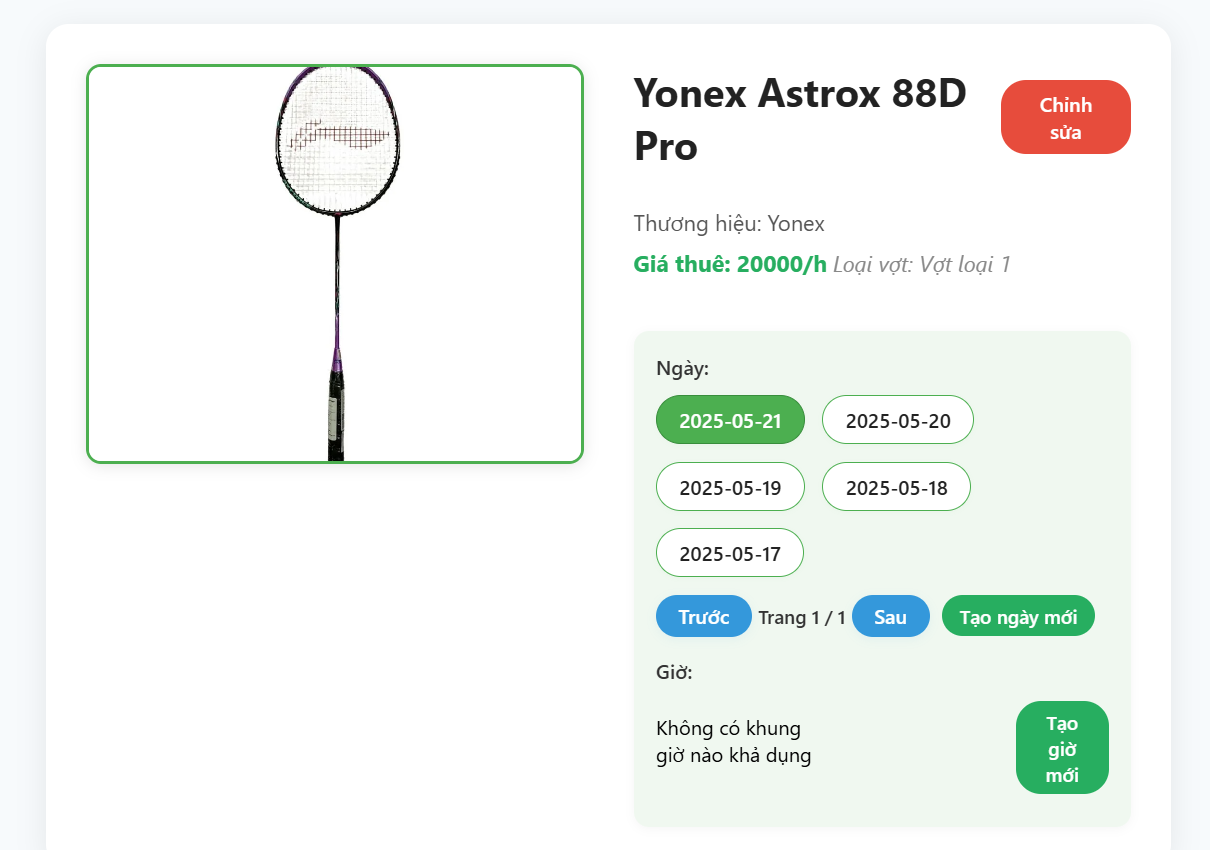
// Output (thất bại)

{

"message": string // Mô tả lỗi

}

1. Quản lý ngày giờ của vợt



- GET /api/racket-types: Lấy danh sách loại vợt

// Output

[

{

"id": integer, // ID loại vợt

"name": string // Tên loại vợt

}

]

- GET /api/rackets/{racketId}: Lấy thông tin chi tiết vợt

// Output

{

"id": integer,

"name": string,

"brand": string,

"price": string,

"racketTypeId": integer,

"racketTypeName": string,

"imageUrl": string

}

- PUT /api/rackets/{racketId}: Cập nhật thông tin vợt

// Input

{

"name": string,

"brand": string,

"price": string,

"racketTypeId": integer,

"imageUrl": string

}

// Output

{

"id": integer,

"name": string,

"brand": string,

"price": string,

"racketTypeId": integer,

"racketTypeName": string,

"imageUrl": string

}

- GET /api/days1/racket/{racketId}: Lấy danh sách ngày của vợt

// Output

[

{

"id": integer,

"date": string, // Format: DD/MM/YYYY

"racketId": integer

}

]

- POST /api/days1: Thêm ngày mới

// Input

{

"date": string, // Format: DD/MM/YYYY

"racketId": integer

}

// Output

{

"id": integer,

"date": string,

"racketId": integer

}

- DELETE /api/days1/{dayId}: Xóa ngày

// Output (thành công)

204 No Content

// Output (thất bại)

{

"message": string // Mô tả lỗi

}

- GET /api/timeslots1/day1/{dayId}: Lấy danh sách khung giờ của ngày

// Output

[

{

"id": integer,

"time": string, // Format: HH:mm

"day1Id": integer,

"isBooked": boolean,

"booking": {

"customer": string, // Mã sinh viên

"registeredAt": string // Thời gian đặt

}

}

]

- POST /api/timeslots1: Thêm khung giờ mới

// Input

{

"startTime": string, // Format: HH:mm

"endTime": string, // Format: HH:mm

"day1Id": integer

}

// Output

{

"id": integer,

"time": string,

"day1Id": integer,

"isBooked": false

}

- DELETE /api/timeslots1/{timeSlotId}: Xóa khung giờ

// Output (thành công)

204 No Content

// Output (thất bại)

{

"message": string // Mô tả lỗi

}

- POST /api/upload: Upload ảnh mới

// Input

FormData với field 'image' chứa file ảnh

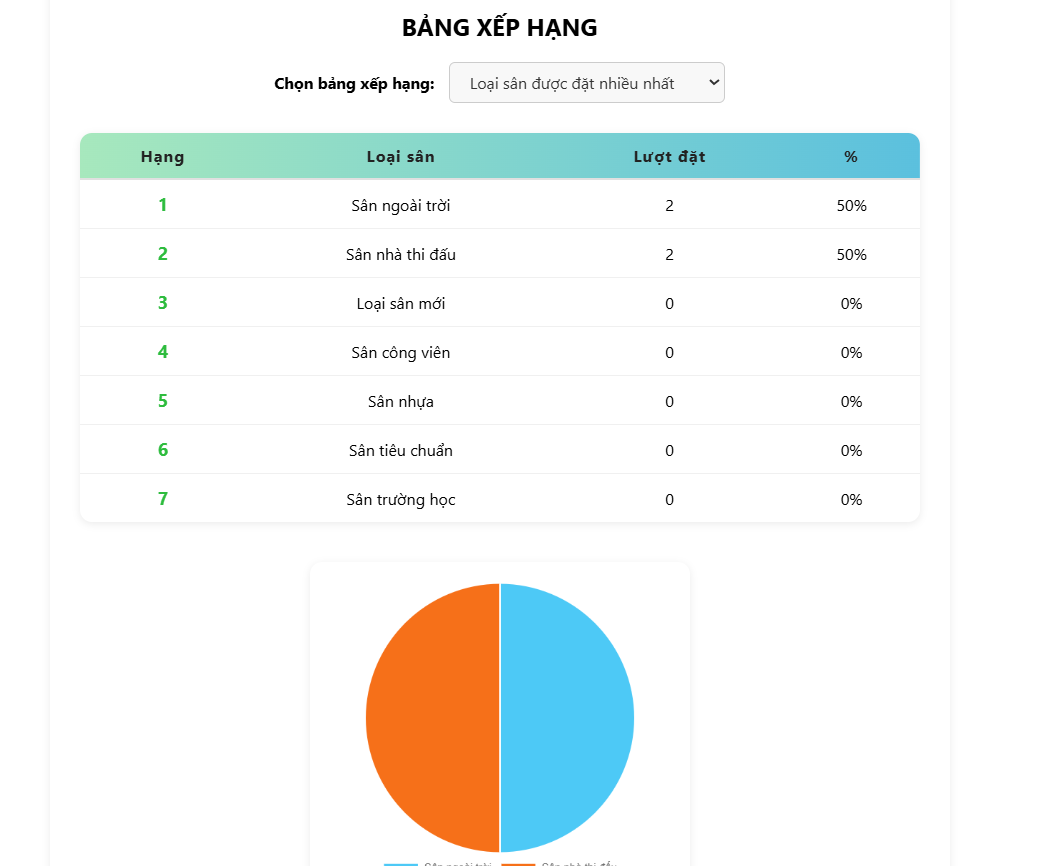
// Output

{

"imageUrl": string // URL của ảnh đã upload

}

1. Xem bảng xếp hạng



- GET /api/statistics/court-type-ranking: Lấy bảng xếp hạng loại sân

// Output

[

{

"courtTypeName": string, // Tên loại sân

"totalBookings": integer, // Tổng lượt đặt

"percentage": number // Phần trăm so với tổng số

}

]

- GET /api/statistics/court-ranking: Lấy bảng xếp hạng sân

// Output

[

{

"courtName": string, // Tên sân

"courtTypeName": string, // Tên loại sân

"totalBookings": integer, // Tổng lượt đặt

"percentage": number // Phần trăm so với tổng số

}

]

- GET /api/statistics/racket-type-ranking: Lấy bảng xếp hạng loại vợt

// Output

[

{

"racketTypeName": string, // Tên loại vợt

"totalBookings": integer, // Tổng lượt thuê

"percentage": number // Phần trăm so với tổng số

}

]

- GET /api/statistics/racket-ranking: Lấy bảng xếp hạng vợt

// Output

[

{

"racketName": string, // Tên vợt

"racketTypeName": string, // Tên loại vợt

"totalBookings": integer, // Tổng lượt thuê

"percentage": number // Phần trăm so với tổng số

}

]

- GET /api/statistics/user-booking-ranking: Lấy bảng xếp hạng người dùng theo lượt đặt

// Output

[

{

"fullName": string, // Họ tên người dùng

"username": string, // Tên tài khoản

"totalBookings": integer, // Tổng lượt đặt

"percentage": number // Phần trăm so với tổng số

}

]

- GET /api/statistics/user-spending-ranking: Lấy bảng xếp hạng người dùng theo chi tiêu

// Output

[

{

"fullName": string, // Họ tên người dùng

"username": string, // Tên tài khoản

"totalSpending": number, // Tổng tiền đã dùng

"percentage": number // Phần trăm so với tổng số

}

]

1. Cập nhật booking

- PUT /api/bookings/confirm/{id}

// Input

Query Parameter:

- accept: boolean (true để chấp nhận, false để từ chối)

// Output (thành công)

204 No Content

// Output (thất bại)

{

"message": string // Mô tả lỗi

}

1. API tìm kiếm

- GET /api/rackets/search?term={keyword}: Tìm kiếm vợt

// Output

[

{

"id": integer,

"name": string,

"brand": string,

"price": string,

"imageUrl": string,

"racketTypeId": integer

}

]

- GET /api/courts/search?term={keyword}: Tìm kiếm sân

// Output

[

{

"id": integer,

"name": string,

"address": string,

"hours": string,

"price": string,

"imageUrl": string,

"courtTypeId": integer

}

]

Chương 4: Kết quả