Đặc tả ứng dụng xếp lịch thi đấu đá bóng lượt đi - lượt về

# 1. Mô tả tổng quan

Ứng dụng xếp lịch thi đấu bóng đá lượt đi - lượt về giúp quản lý và tự động tạo lịch thi đấu giữa các đội trong một giải đấu. Ứng dụng đảm bảo mỗi đội sẽ thi đấu với tất cả các đội khác hai lần (một lần trên sân nhà và một lần trên sân khách). Hệ thống cung cấp các chức năng quản lý thông tin đội bóng, sân vận động, kết quả thi đấu, bảng xếp hạng và xuất lịch thi đấu dưới dạng báo cáo.

Ứng dụng hỗ trợ việc sắp xếp lịch một cách khoa học, đảm bảo thời gian giữa các trận đấu hợp lý và tránh tình trạng đội bóng phải thi đấu liên tiếp trên sân nhà hoặc sân khách.

# Đối tượng sử dụng và phạm vi sử dụng

***Đối tượng sử dụng:***

**Ban tổ chức giải đấu:** Quản lý thông tin đội bóng, sân vận động, lịch thi đấu và kết quả trận đấu.

**Trọng tài:** Cập nhật kết quả trận đấu và các thông tin liên quan như thẻ phạt, cầu thủ ghi bàn.

**Người quản lý đội bóng:** Theo dõi lịch thi đấu, kết quả, và bảng xếp hạng.

***Phạm vi sử dụng:***

* Đề xuất cho các giải bóng đá phong trào, giải giao hữu hoặc các giải bóng đá có quy mô vừa và nhỏ.
* Số đội tham gia: Từ 4 đội trở lên.
* Hình thức thi đấu: Lượt đi - lượt về tính điểm.

# Chức năng chính

## Quản lý giải đấu

### 3.1.1. Mô tả chức năng

Chức năng này cho phép người dùng tạo mới, cập nhật và xóa giải đấu. Hệ thống hỗ trợ cấu hình các thông tin chi tiết về giải đấu, quản lý danh sách đội tham gia và theo dõi trạng thái giải đấu theo từng giai đoạn.

### 3.1.2. Yêu cầu chi tiết

#### 1. Tạo mới, cập nhật, xóa giải đấu

* **Tạo giải đấu mới**:
  + Nhập các thông tin cơ bản.
  + Thêm danh sách đội tham gia.
  + Xác nhận và lưu vào hệ thống.
* **Cập nhật giải đấu**:
  + Chỉnh sửa thông tin (tên, mô tả, thời gian, thể thức, danh sách đội).
  + Thay đổi trạng thái giải đấu.
* **Xóa giải đấu**:
  + Chỉ cho phép xóa nếu giải chưa bắt đầu.
  + Nếu đã diễn ra, chỉ có thể hủy giải đấu thay vì xóa.

#### 2. Cấu hình thông tin giải đấu

* **Các thông tin cần thiết**:
  + Tên giải đấu
  + Mô tả giải đấu
  + Số đội tham gia
  + Thể thức thi đấu (Vòng tròn, Vòng loại, Lượt đi - lượt về)
  + Thời gian diễn ra (ngày bắt đầu - kết thúc)
  + Địa điểm tổ chức (sân vận động)
* Người quản lý có thể thay đổi thông tin này trước khi giải đấu bắt đầu.

#### 3. Danh sách các đội tham gia giải

* Cho phép:
  + Thêm đội bóng vào giải (chỉ các đội có trong hệ thống).
  + Loại bỏ đội bóng khỏi giải (nếu chưa bắt đầu thi đấu).
* Hiển thị danh sách đội bóng kèm thông tin:
  + Tên đội
  + Huấn luyện viên trưởng
  + Sân nhà
  + Số trận đã đấu, số điểm hiện tại

#### 4. Quản lý trạng thái giải đấu

* Trạng thái của giải đấu có thể là:
  + Chưa bắt đầu: Đang trong giai đoạn chuẩn bị.
  + Đang diễn ra: Đã có trận đấu diễn ra.
  + Đã kết thúc: Kết thúc mùa giải, không thể chỉnh sửa thêm.
  + Hủy bỏ: Giải bị hủy do lý do khách quan.

### 3.1.3. Luồng hoạt động

#### 1. Tạo giải đấu mới

* Bước 1: Người quản lý vào giao diện tạo giải đấu.
* Bước 2: Nhập thông tin giải đấu (tên, thể thức, số đội, thời gian).
* Bước 3: Thêm danh sách đội tham gia.
* Bước 4: Xác nhận và lưu giải đấu.

#### 2. Chỉnh sửa thông tin giải đấu

* Bước 1: Người quản lý truy cập danh sách giải đấu.
* Bước 2: Chọn giải đấu cần chỉnh sửa.
* Bước 3: Cập nhật thông tin hoặc danh sách đội tham gia.
* Bước 4: Lưu thay đổi.

#### 3. Xóa hoặc hủy giải đấu

* Bước 1: Chọn giải đấu cần xóa hoặc hủy.
* Bước 2: Nếu giải chưa diễn ra → Xóa khỏi hệ thống.
* Bước 3: Nếu giải đã diễn ra → Chuyển trạng thái thành "Hủy bỏ".

#### 4. Cập nhật trạng thái giải đấu

* Hệ thống tự động thay đổi trạng thái dựa vào ngày bắt đầu/kết thúc.
* Người quản lý có thể cập nhật trạng thái thủ công nếu cần.

### 3.1.4. Giao diện dự kiến

***Trang danh sách giải đấu***

* + Hiển thị danh sách các giải đấu.
  + Chức năng tạo, chỉnh sửa, xóa hoặc hủy giải đấu.

***Trang chi tiết giải đấu***

* + Hiển thị thông tin giải.
  + Danh sách đội tham gia.
  + Trạng thái giải đấu.

***Trang quản lý đội bóng trong giải đấu***

* + Hiển thị danh sách đội.
  + Chức năng thêm/xóa đội.

### 3.1.5. Công nghệ & Tích hợp

* **Backend**: Java Spring Boot, Node.js hoặc .NET Core.
* **Database**: MySQL hoặc PostgreSQL.
* **Frontend**: React hoặc Angular.
* **Tích hợp API**:
  + **POST /api/tournaments**: Tạo giải đấu.
  + **GET /api/tournaments**: Lấy danh sách giải đấu.
  + **PUT /api/tournaments/{id}**: Chỉnh sửa giải đấu.
  + **DELETE /api/tournaments/{id}**: Xóa giải đấu.
  + **POST /api/tournaments/{id}/teams**: Thêm đội vào giải.

## Quản lý đội bóng

### 3.2.1. Mô tả chức năng

Chức năng này cho phép quản lý thông tin đội bóng, danh sách cầu thủ và huấn luyện viên, đồng thời cập nhật tình trạng cầu thủ. Hệ thống cũng theo dõi thống kê thành tích của đội bóng trong giải đấu để hiển thị hiệu suất thi đấu.

### 3.2.2. Yêu cầu chi tiết

#### 1. Thêm, sửa, xóa thông tin đội bóng

* Người quản lý có thể:
  + Thêm đội bóng mới vào hệ thống.
  + Chỉnh sửa thông tin đội (tên đội, sân nhà, huấn luyện viên trưởng).
  + Xóa đội bóng (chỉ khi đội chưa tham gia giải đấu).
* Thông tin đội bóng bao gồm:
  + Tên đội
  + Sân nhà
  + Huấn luyện viên trưởng
  + Logo đội bóng
  + Danh sách cầu thủ

#### 2. Quản lý danh sách cầu thủ và huấn luyện viên

* Người quản lý có thể:
  + Thêm, sửa, xóa cầu thủ trong danh sách đội.
  + Thêm hoặc thay đổi huấn luyện viên trưởng.
* Danh sách cầu thủ bao gồm:
  + Tên cầu thủ
  + Ngày sinh (tuổi)
  + Vị trí thi đấu (Thủ môn, Hậu vệ, Tiền vệ, Tiền đạo)
  + Số áo
  + Tình trạng chấn thương
  + Quốc tịch

#### 3. Cập nhật thông tin cầu thủ

* Hệ thống hỗ trợ cập nhật:
  + Thay đổi số áo hoặc vị trí thi đấu.
  + Ghi nhận chấn thương (loại chấn thương, thời gian hồi phục).
  + Chuyển nhượng cầu thủ giữa các đội trong mùa giải.

#### 4. Thống kê thành tích của đội trong giải đấu

* Hệ thống tự động cập nhật thành tích của đội dựa trên kết quả trận đấu:
  + Số trận thắng, hòa, thua.
  + Số bàn thắng, số bàn thua.
  + Hiệu số bàn thắng.
  + Điểm số trên bảng xếp hạng.
* Thống kê có thể xem theo:
  + Từng mùa giải.
  + Toàn bộ lịch sử tham gia giải đấu.

### 3.2.3. Luồng hoạt động

#### 1. Quản lý đội bóng

* Bước 1: Người quản lý vào danh sách đội bóng.
* Bước 2: Chọn thêm mới, chỉnh sửa hoặc xóa đội.
* Bước 3: Lưu thông tin đội bóng vào hệ thống.

#### 2. Quản lý danh sách cầu thủ

* Bước 1: Chọn đội bóng cần chỉnh sửa danh sách cầu thủ.
* Bước 2: Thêm mới, chỉnh sửa hoặc xóa cầu thủ.
* Bước 3: Lưu dữ liệu vào hệ thống.

#### 3. Cập nhật thông tin cầu thủ

* Bước 1: Chọn cầu thủ cần cập nhật.
* Bước 2: Chỉnh sửa số áo, vị trí, tình trạng chấn thương.
* Bước 3: Xác nhận và lưu thay đổi.

#### 4. Xem thống kê thành tích đội bóng

* Bước 1: Truy cập bảng thống kê.
* Bước 2: Chọn đội bóng và mùa giải cần xem.
* Bước 3: Hệ thống hiển thị số trận thắng, thua, số bàn thắng/thua, điểm số.

### 3.2.4. Giao diện dự kiến

***Trang danh sách đội bóng***

* + Hiển thị danh sách đội, logo, sân nhà.
  + Chức năng thêm, sửa, xóa đội bóng.

***Trang quản lý cầu thủ***

* + Danh sách cầu thủ của từng đội.
  + Cho phép thêm, sửa, xóa cầu thủ.
  + Cập nhật tình trạng chấn thương.

***Trang thống kê đội bóng***

* + Hiển thị thành tích đội bóng theo từng mùa giải.
  + Thống kê số trận thắng, hòa, thua, bàn thắng, bàn thua.

### 3.2.5. Công nghệ & Tích hợp

* **Backend**: Node.js, Java Spring Boot hoặc .NET Core để xử lý dữ liệu.
* **Database**: MySQL hoặc PostgreSQL để lưu thông tin đội bóng, cầu thủ.
* **Frontend**: React hoặc Angular để hiển thị thông tin.
* **Tích hợp API**:
  + **POST /api/teams**: Thêm đội bóng.
  + **GET /api/teams**: Lấy danh sách đội bóng.
  + **PUT /api/teams/{id}**: Chỉnh sửa thông tin đội bóng.
  + **DELETE /api/teams/{id}**: Xóa đội bóng.
  + **POST /api/players**: Thêm cầu thủ.
  + **GET /api/players/{teamId}**: Lấy danh sách cầu thủ của đội.
  + **PUT /api/players/{id}**: Cập nhật thông tin cầu thủ.

## Tạo lịch thi đấu

### 3.3.1. Mô tả chức năng

Chức năng này hỗ trợ tự động tạo lịch thi đấu theo thể thức lượt đi - lượt về bằng thuật toán tối ưu. Hệ thống đảm bảo không có xung đột về thời gian và địa điểm, đồng thời cho phép người quản lý điều chỉnh lịch thủ công nếu cần. Danh sách trận đấu được hiển thị rõ ràng theo ngày, đội bóng và sân thi đấu.

### 3.3.2. Yêu cầu chi tiết

#### 1. Tự động xếp lịch thi đấu

* Hệ thống sử dụng thuật toán để sắp xếp lịch thi đấu dựa trên:
  + Thể thức giải đấu (lượt đi - lượt về, vòng bảng, vòng loại trực tiếp).
  + Số lượng đội tham gia.
  + Số lượt trận (mỗi đội đấu với các đội khác bao nhiêu lần).
  + Thời gian nghỉ giữa các trận của mỗi đội.
  + Tránh trùng lịch với các trận đấu khác trên cùng sân.
* Nếu giải đấu có số đội lẻ, hệ thống tự động thêm lượt nghỉ (bye round).

#### 2. Điều chỉnh lịch thi đấu thủ công

* Người quản lý có thể:
  + Thay đổi ngày, giờ trận đấu.
  + Thay đổi địa điểm thi đấu.
  + Hoán đổi lịch giữa hai trận đấu nếu cần thiết.
* Hệ thống cảnh báo nếu có xung đột về lịch hoặc sân bãi.

#### 3. Đảm bảo không có xung đột về thời gian và địa điểm

* Kiểm tra các yếu tố trước khi xác nhận lịch:
  + Một đội không thi đấu hai trận cùng ngày.
  + Một sân không có hai trận cùng thời điểm.
  + Trọng tài không bắt hai trận liên tiếp của cùng một đội.
* Nếu phát hiện xung đột, hệ thống đưa ra đề xuất điều chỉnh.

#### 4. Hiển thị danh sách trận đấu

* Hệ thống cung cấp danh sách trận đấu theo:
  + Ngày thi đấu (dạng lịch).
  + Đội bóng (danh sách trận đấu của từng đội).
  + Sân vận động (danh sách trận đấu theo từng sân).

### 3.3.3. Luồng hoạt động

#### 1. Tạo lịch thi đấu tự động

* Bước 1: Người quản lý nhập thông tin giải đấu (số đội, thể thức, sân bãi).
* Bước 2: Hệ thống sử dụng thuật toán để tạo lịch thi đấu tối ưu.
* Bước 3: Hệ thống kiểm tra xung đột và đề xuất điều chỉnh.
* Bước 4: Lưu lịch thi đấu vào hệ thống.

#### 2. Chỉnh sửa lịch thi đấu thủ công

* Bước 1: Người quản lý vào danh sách lịch thi đấu.
* Bước 2: Chỉnh sửa ngày, giờ hoặc sân thi đấu nếu cần.
* Bước 3: Hệ thống kiểm tra xung đột và đưa ra cảnh báo.
* Bước 4: Xác nhận lịch thi đấu đã chỉnh sửa.

#### 3. Hiển thị lịch thi đấu

* Bước 1: Người dùng truy cập lịch thi đấu.
* Bước 2: Chọn cách hiển thị (theo ngày, đội bóng hoặc sân vận động).
* Bước 3: Hệ thống hiển thị danh sách trận đấu theo tiêu chí đã chọn.

### 3.3.4. Giao diện dự kiến

***Trang tạo lịch thi đấu****:*

* + Nhập thông tin giải đấu (thể thức, số đội, sân).
  + Nhấn "Tạo lịch thi đấu".
  + Hệ thống tự động tạo lịch và hiển thị kết quả.

***Trang danh sách lịch thi đấu****:*

* + Hiển thị danh sách trận đấu theo ngày, đội bóng, sân vận động.
  + Cho phép chỉnh sửa lịch nếu cần.
  + Cảnh báo nếu có xung đột.

***Trang xem lịch thi đấu chi tiết****:*

* + Hiển thị thông tin từng trận (đội bóng, thời gian, sân).
  + Bộ lọc theo ngày, đội bóng hoặc sân thi đấu.

### 3.3.5. Công nghệ & Tích hợp

* **Backend**: Node.js, Java Spring Boot hoặc .NET Core để xử lý dữ liệu.
* **Database**: MySQL hoặc PostgreSQL để lưu lịch thi đấu.
* **Frontend**: React hoặc Angular để hiển thị thông tin.
* **Tích hợp API**:
  + **POST /api/matches/schedule**: Tạo lịch thi đấu.
  + **GET /api/matches**: Lấy danh sách trận đấu.
  + **PUT /api/matches/{id}**: Chỉnh sửa lịch thi đấu.

## Phân công trọng tài

### 3.4.1. Mô tả chức năng

Chức năng này giúp quản lý danh sách trọng tài, thực hiện phân công trọng tài cho các trận đấu một cách tự động hoặc thủ công. Hệ thống đảm bảo không có trọng tài nào điều khiển hai trận liên tiếp của cùng một đội và lưu trữ lịch sử phân công để đánh giá hiệu suất làm việc của trọng tài.

### 3.4.2. Yêu cầu chi tiết

#### 1. Quản lý danh sách trọng tài

* Người quản lý có thể thực hiện các thao tác sau:
  + Thêm, sửa, xóa trọng tài.
  + Lưu thông tin chi tiết:
    - Họ tên
    - Quốc tịch
    - Cấp bậc trọng tài (FIFA, quốc gia, địa phương)
    - Số trận đã điều khiển
    - Đánh giá trung bình từ các trận trước
* Hiển thị danh sách trọng tài kèm theo trạng thái (rảnh/đang làm nhiệm vụ).

#### 2. Phân công trọng tài cho trận đấu

* Hệ thống có thể **tự động** hoặc **thủ công** phân công trọng tài.
* Tiêu chí phân công:
  + **Không cầm còi hai trận liên tiếp của cùng một đội**.
  + **Không cầm còi nhiều trận liên tục trong thời gian ngắn**.
  + **Công bằng giữa các trọng tài (tránh một trọng tài bắt quá nhiều trận so với người khác)**.
* Nếu hệ thống tự động phân công, người quản lý vẫn có thể chỉnh sửa thủ công.

#### 3. Lưu lịch sử phân công trọng tài

* Hệ thống ghi lại toàn bộ lịch sử phân công trọng tài, bao gồm:
  + **Trận đấu đã điều khiển**
  + **Ngày diễn ra trận đấu**
  + **Đội bóng tham gia**
  + **Đánh giá hiệu suất làm việc (nếu có)**

#### 4. Đánh giá hiệu suất làm việc

* Người quản lý hoặc giám sát viên có thể đánh giá trọng tài sau mỗi trận đấu.
* Tiêu chí đánh giá có thể bao gồm:
  + **Độ chính xác trong quyết định**
  + **Số lần tham khảo VAR (nếu có)**
  + **Khả năng kiểm soát trận đấu**
* Hệ thống sẽ tổng hợp điểm đánh giá và cập nhật vào hồ sơ trọng tài.

### 3.4.3. Luồng hoạt động

#### 1. Quản lý danh sách trọng tài

* Bước 1: Người quản lý vào danh sách trọng tài.
* Bước 2: Thêm hoặc chỉnh sửa thông tin trọng tài.
* Bước 3: Lưu dữ liệu vào hệ thống.

#### 2. Phân công trọng tài

* Bước 1: Khi lên lịch trận đấu, hệ thống tự động tìm trọng tài phù hợp.
* Bước 2: Nếu cần, người quản lý có thể chỉnh sửa trọng tài được phân công.
* Bước 3: Lưu thông tin vào hệ thống.

#### 3. Ghi nhận lịch sử phân công

* Bước 1: Khi trận đấu kết thúc, hệ thống lưu lại trọng tài đã tham gia.
* Bước 2: Nếu có đánh giá, hệ thống cập nhật điểm số hiệu suất của trọng tài.

### 3.4.4. Giao diện dự kiến

***Trang danh sách trọng tài****:*

* + Hiển thị danh sách trọng tài cùng trạng thái (rảnh/bận).
  + Chức năng thêm, sửa, xóa trọng tài.

***Trang phân công trọng tài****:*

* + Danh sách trận đấu cần phân công trọng tài.
  + Gợi ý trọng tài phù hợp hoặc cho phép chọn thủ công.
  + Cảnh báo nếu có trọng tài điều khiển hai trận liên tiếp của một đội.

***Trang lịch sử phân công và đánh giá trọng tài****:*

* + Danh sách trận đấu mà trọng tài đã điều khiển.
  + Điểm đánh giá hiệu suất làm việc.

### 3.4.5. Công nghệ & Tích hợp

* **Backend**: Node.js, Java Spring Boot hoặc .NET Core để xử lý dữ liệu.
* **Database**: MySQL hoặc PostgreSQL để lưu danh sách trọng tài, lịch sử phân công.
* **Frontend**: React hoặc Angular để hiển thị thông tin.
* **Tích hợp API**:
  + **POST /api/referees**: Thêm trọng tài mới.
  + **GET /api/referees**: Lấy danh sách trọng tài.
  + **POST /api/matches/assign-referee**: Phân công trọng tài cho trận đấu.

## Quản lý sân vận động và thời gian thi đấu

### 3.5.1. Mô tả chức năng

Chức năng này giúp quản lý danh sách sân vận động, bao gồm các thông tin chi tiết về sân đấu. Hệ thống hỗ trợ gán sân vận động cho từng trận đấu dựa trên lịch trình sử dụng, sức chứa và điều kiện thời tiết. Đồng thời, hệ thống sẽ xác định khung giờ thi đấu hợp lý để tránh trùng lịch và tối ưu thời gian nghỉ giữa các trận.

### 3.5.2. Yêu cầu chi tiết

#### 1. Quản lý danh sách sân vận động

* Người quản lý có thể thêm, sửa, xóa và xem thông tin chi tiết của sân vận động, bao gồm:
  + **Tên sân** (ví dụ: SVĐ Quốc gia, SVĐ Mỹ Đình).
  + **Địa chỉ** (thành phố, quốc gia, vị trí trên bản đồ).
  + **Sức chứa** (tổng số khán giả có thể tiếp nhận).
  + **Loại mặt sân** (cỏ tự nhiên, cỏ nhân tạo).
  + **Điều kiện thời tiết hiện tại** (nhiệt độ, độ ẩm, khả năng mưa).

#### 2. Gán sân đấu cho từng trận đấu

* Hệ thống tự động hoặc cho phép người quản lý gán sân vận động cho các trận đấu dựa trên:
  + **Khả năng tiếp nhận**: Chỉ những sân có sức chứa phù hợp với quy mô trận đấu mới được sử dụng.
  + **Lịch sử sử dụng**: Kiểm tra xem sân đã được đặt trước cho trận đấu khác chưa.
  + **Điều kiện thời tiết**: Nếu sân không phù hợp do thời tiết xấu (mưa lớn, ngập nước), cần thay đổi địa điểm.
* Nếu sân đấu bị trùng lịch hoặc không thể sử dụng, hệ thống đề xuất sân thay thế.

#### 3. Xác định khung giờ thi đấu hợp lý

* Hệ thống đề xuất khung giờ thi đấu dựa trên:
  + **Tránh trùng lịch**: Kiểm tra sân đã có trận đấu khác cùng thời điểm chưa.
  + **Tối ưu thời gian nghỉ**: Nếu có nhiều trận trong ngày, cần bố trí khoảng thời gian nghỉ hợp lý giữa các trận.
  + **Điều kiện thời tiết**: Tránh tổ chức vào thời điểm quá nóng hoặc mưa lớn.
* Người quản lý có thể chỉnh sửa khung giờ nếu cần thiết.

### 3.5.3. Luồng hoạt động

#### 1. Quản lý sân vận động

* Bước 1: Người quản lý vào danh sách sân vận động.
* Bước 2: Thêm mới/sửa đổi sân vận động với các thông tin chi tiết.
* Bước 3: Lưu thông tin vào hệ thống.

#### 2. Gán sân đấu cho trận đấu

* Bước 1: Khi lên lịch trận đấu, hệ thống tự động tìm sân phù hợp.
* Bước 2: Kiểm tra khả năng tiếp nhận và lịch trình sử dụng sân.
* Bước 3: Nếu sân hợp lệ, hệ thống gán sân cho trận đấu.
* Bước 4: Nếu sân không khả dụng, hệ thống đề xuất sân thay thế.

#### 3. Xác định khung giờ thi đấu

* Bước 1: Hệ thống kiểm tra lịch thi đấu và tránh trùng giờ.
* Bước 2: Đề xuất thời gian hợp lý, đảm bảo đủ thời gian nghỉ giữa các trận.
* Bước 3: Người quản lý có thể điều chỉnh khung giờ nếu cần.

### 3.5.4. Giao diện dự kiến

***Trang danh sách sân vận động****:*

* + Hiển thị danh sách sân, kèm theo sức chứa và điều kiện thời tiết.
  + Chức năng thêm, sửa, xóa sân vận động.

***Trang gán sân và thời gian thi đấu****:*

* + Chọn sân vận động cho từng trận.
  + Gợi ý khung giờ phù hợp.
  + Cảnh báo nếu có trùng lịch.

### 3.5.5. Công nghệ & Tích hợp

* **Backend**: Node.js, Java Spring Boot hoặc .NET Core.
* **Database**: MySQL hoặc PostgreSQL để lưu danh sách sân, lịch trình thi đấu.
* **Frontend**: React hoặc Angular để hiển thị thông tin.
* **Tích hợp API**:
  + **POST /api/stadiums**: Thêm sân vận động mới.
  + **GET /api/stadiums**: Lấy danh sách sân.
  + **POST /api/matches/schedule**: Lên lịch trận đấu với sân và thời gian phù hợp.

## Cập nhật kết quả và bảng xếp hạng

### 3.6.1. Mô tả chức năng

Chức năng này cho phép người quản lý nhập kết quả từng trận đấu, bao gồm số bàn thắng, cầu thủ ghi bàn, thẻ phạt và chấn thương. Hệ thống sẽ tự động cập nhật bảng xếp hạng, hiển thị thống kê chi tiết và xác định đội vô địch, đội xuống hạng nếu có quy định.

### 3.6.2. Yêu cầu chi tiết

#### 1. Nhập kết quả trận đấu

* Người dùng có thể nhập thông tin chi tiết về kết quả trận đấu:
  + Số bàn thắng của mỗi đội
  + Danh sách cầu thủ ghi bàn (phút ghi bàn, kiểu bàn thắng: penalty, đá phạt, phản lưới nhà, v.v.)
  + Thẻ phạt (thẻ vàng, thẻ đỏ, phút nhận thẻ, cầu thủ bị phạt)
  + Chấn thương (cầu thủ bị chấn thương, thời gian xảy ra, mức độ nghiêm trọng)
  + Thời gian kiểm soát bóng của mỗi đội
  + Số lần sút trúng đích, số pha cản phá của thủ môn

#### 2. Tự động cập nhật bảng xếp hạng

* Khi kết quả trận đấu được nhập, hệ thống sẽ tự động cập nhật bảng xếp hạng theo các quy tắc sau:
  + Thêm 3 điểm cho đội thắng, 1 điểm cho trận hòa, 0 điểm cho đội thua.
  + Tính hiệu số bàn thắng (bàn thắng ghi được - bàn thua).
  + Cập nhật số trận đã đấu, số trận thắng, hòa, thua.
  + Sắp xếp bảng xếp hạng theo điểm số, hiệu số bàn thắng và số bàn thắng ghi được.

#### 3. Hiển thị thống kê chi tiết

* Hệ thống hiển thị bảng thống kê theo từng đội bóng:
  + Số trận đã đấu
  + Số trận thắng, hòa, thua
  + Số bàn thắng, số bàn thua, hiệu số bàn thắng
  + Điểm số hiện tại
  + Tỷ lệ kiểm soát bóng trung bình
  + Danh sách cầu thủ ghi nhiều bàn thắng nhất

#### 4. Xác định đội vô địch & đội xuống hạng

***Xác định đội vô địch****:*

* + Đội có điểm số cao nhất sẽ được hệ thống gắn nhãn **"Vô địch"**.
  + Nếu có hai đội bằng điểm, hệ thống sẽ xét hiệu số bàn thắng, số bàn thắng ghi được hoặc thành tích đối đầu (nếu có quy định).

***Xác định đội xuống hạng****:*

* + Nếu giải đấu có quy định xuống hạng, hệ thống sẽ xác định các đội có thứ hạng thấp nhất và gắn nhãn **"Xuống hạng"**.
  + Tiêu chí xuống hạng có thể là điểm số, hiệu số bàn thắng hoặc kết quả đối đầu giữa các đội cuối bảng.

### 3.6.3. Luồng hoạt động

#### 1. Nhập kết quả trận đấu

* Bước 1: Người quản lý chọn trận đấu từ danh sách.
* Bước 2: Nhập thông tin kết quả (số bàn thắng, cầu thủ ghi bàn, thẻ phạt, chấn thương).
* Bước 3: Nhấn "Lưu kết quả", hệ thống xử lý dữ liệu.

#### 2. Hệ thống cập nhật bảng xếp hạng

* Bước 1: Hệ thống tính toán điểm số, hiệu số bàn thắng, số trận đấu.
* Bước 2: Cập nhật bảng xếp hạng và lưu vào cơ sở dữ liệu.
* Bước 3: Hiển thị bảng xếp hạng mới trên giao diện.

#### 3. Hiển thị thống kê và xác định đội vô địch, xuống hạng

* Bước 1: Hệ thống kiểm tra thứ hạng các đội.
* Bước 2: Nếu kết thúc mùa giải, hệ thống xác định đội vô địch, đội xuống hạng và gắn nhãn tương ứng.
* Bước 3: Hiển thị kết quả trên giao diện và cho phép xuất báo cáo.

### 3.6.4. Giao diện dự kiến

***Trang nhập kết quả trận đấu****:*

* + Danh sách các trận đấu cần nhập kết quả.
  + Form nhập thông tin kết quả (bàn thắng, cầu thủ ghi bàn, thẻ phạt, chấn thương).

***Trang bảng xếp hạng****:*

* + Hiển thị danh sách đội bóng theo thứ tự xếp hạng.
  + Các chỉ số: số trận, số thắng, hòa, thua, hiệu số, điểm số.
  + Nhãn "Vô địch", "Xuống hạng" nếu áp dụng.

### 3.6.5. Công nghệ & Tích hợp

* **Backend**: Node.js, Java Spring Boot hoặc .NET Core để xử lý dữ liệu.
* **Database**: MySQL hoặc PostgreSQL để lưu trữ kết quả trận đấu và bảng xếp hạng.
* **Frontend**: React hoặc Angular để hiển thị dữ liệu.
* **Tích hợp API**:
  + **POST /api/match-results**: Nhập kết quả trận đấu.
  + **GET /api/standings**: Lấy bảng xếp hạng cập nhật.

## Xem thông tin và báo cáo

* Tra cứu lịch thi đấu, kết quả và bảng xếp hạng theo nhiều tiêu chí (đội bóng, ngày thi đấu, sân vận động).
* Xuất báo cáo kết quả giải đấu dưới dạng bảng hoặc file Excel/PDF.
* Hỗ trợ API để tích hợp với hệ thống khác như trang web, ứng dụng di động.

# 3. Yêu cầu hệ thống

* Giao diện người dùng: Web/Mobile thân thiện, dễ sử dụng.
* Cơ sở dữ liệu: SQL hoặc NoSQL hỗ trợ lưu trữ thông tin đội bóng, lịch thi đấu, kết quả.
* Xử lý logic: Hỗ trợ thuật toán xếp lịch và cập nhật điểm số theo thể thức giải đấu.
* Bảo mật: Xác thực người dùng, phân quyền quản trị viên và đội bóng.

# 4. Công nghệ đề xuất

* Backend: Node.js/Java/Spring Boot
* Frontend: React/Vue.js
* Database: MySQL/PostgreSQL/MongoDB
* Triển khai: Docker/Kubernetes

# 5. Yêu cầu phi chức năng

* Hiệu suất: Hệ thống phải xử lý ít nhất 10.000 yêu cầu mỗi giờ mà không bị gián đoạn.
* Bảo mật: Dữ liệu người dùng phải được mã hóa, hệ thống cần hỗ trợ xác thực hai lớp (2FA).
* **Khả năng mở rộng:** Hệ thống phải có khả năng mở rộng theo số lượng người dùng và giải đấu.
* **Khả dụng:** Đảm bảo uptime tối thiểu 99.9%.
* **Tích hợp:** Hỗ trợ API RESTful để kết nối với các hệ thống bên ngoài.
* **Khả năng sao lưu:** Dữ liệu phải được sao lưu định kỳ hàng ngày và có cơ chế phục hồi nhanh chóng.

# 6. Sơ đồ luồng dữ liệu (DFD)

**Level 0:**

Người dùng nhập thông tin đội bóng và sân vận động.

Hệ thống xử lý và lưu trữ dữ liệu.

Hệ thống tạo lịch thi đấu và cập nhật kết quả.

**Level 1:**

Quản lý đội bóng -> Lưu trữ thông tin đội bóng.

Quản lý sân vận động -> Lưu trữ thông tin sân vận động.

Tạo lịch thi đấu -> Tạo danh sách trận đấu.

Cập nhật kết quả -> Cập nhật bảng xếp hạng.

Xuất báo cáo -> Tạo file báo cáo.