### 1. ****Xác định đối tượng và chức năng chính****

#### ****Đối tượng sử dụng:****

* **Quản trị viên:** Quản lý toàn bộ hệ thống, thiết lập thông tin tòa nhà, khu vực, phòng ban, cấu hình cảnh báo, xem báo cáo thống kê.
* **Nhân viên bảo trì:** Nhận thông báo về trang thiết bị cần bảo trì, cập nhật trạng thái bảo trì, báo cáo sự cố.
* **Người dùng/Người thuê (nếu có):** Tra cứu thông tin trang thiết bị chung, lịch bảo trì, đặt lịch sử dụng thiết bị (nếu cần).

#### ****Chức năng chính:****

**- Quản lý trang thiết bị:**

* + Thêm, sửa, xóa trang thiết bị (bao gồm: tên, mã, vị trí, loại thiết bị, tình trạng, ngày bảo trì cuối cùng…).
  + Phân loại thiết bị theo loại (điện, cơ khí, an ninh, ...).

**- Quản lý vị trí và hạ tầng:**

* + Quản lý thông tin tòa nhà, khu vực, tầng, phòng.
  + Liên kết trang thiết bị với vị trí cụ thể trong tòa nhà.

**- Bảo trì và sửa chữa:**

* + Lịch bảo trì định kỳ: tự động gửi thông báo bảo trì cho nhân viên bảo trì khi đến hạn.
  + Ghi nhận lịch sử bảo trì, sửa chữa và trạng thái hiện tại của trang thiết bị.

**- Báo cáo & Thống kê:**

* + Thống kê số lượng thiết bị theo loại, tình trạng, vị trí.
  + Báo cáo sự cố, tình trạng hoạt động và hiệu suất của trang thiết bị.
  + Dashboard trực quan hiển thị số liệu quan trọng (số thiết bị cần bảo trì, số thiết bị hoạt động, ...).

**- Thông báo và cảnh báo:**

* + Gửi email hoặc tin nhắn cảnh báo khi trang thiết bị gặp sự cố hoặc đến hạn bảo trì.

**- Quản lý người dùng:**

* + Tạo, chỉnh sửa thông tin tài khoản cho quản trị viên và nhân viên bảo trì.
  + Phân quyền truy cập và hiển thị các tính năng tương ứng.

### 2. ****Kiến trúc hệ thống****

#### ****Công nghệ đề xuất:****

**Backend:**

* + Sử dụng **Django** làm framework phát triển web, vì Django hỗ trợ nhanh chóng trong việc xây dựng hệ thống quản lý với ORM, admin panel, bảo mật, và khả năng mở rộng.

**Frontend:**

* + Có thể dùng Django Template để xây dựng giao diện đơn giản.
  + Nếu muốn giao diện hiện đại, có thể kết hợp với **React** hoặc **Vue.js** để xây dựng SPA (Single Page Application).
  + CSS framework như **Bootstrap** hoặc **Tailwind CSS** để xây dựng giao diện responsive.

**Database:**

* + Sử dụng SQLITE/SqlAlchemy cho hệ thống quản lý trang thiết bị.

**Thông báo & Tích hợp:**

* + Tích hợp hệ thống gửi email (ví dụ: SMTP, SendGrid) hoặc SMS để gửi cảnh báo bảo trì.
  + Nếu có hệ thống IoT (thiết bị thông minh trong tòa nhà), có thể tích hợp API để cập nhật trạng thái thiết bị theo thời gian thực.

### 3. ****Thiết kế cơ sở dữ liệu****

#### ****Các bảng chính:****

**Buildings (Tòa nhà):**

* + id, tên tòa nhà, địa chỉ, số tầng, …

**Locations (Vị trí/Phòng):**

* + id, building\_id (FK), tầng, phòng, …

**Equipment (Trang thiết bị):**

* + id, mã thiết bị, tên thiết bị, loại thiết bị, vị trí\_id (FK), tình trạng, ngày bảo trì cuối, …

**Maintenance (Bảo trì/Sửa chữa):**

* + id, equipment\_id (FK), mô tả công việc, ngày bảo trì, trạng thái (đã hoàn thành/chưa hoàn thành), …

**Users (Người dùng):**

* + id, họ tên, username, mật khẩu, vai trò (quản trị viên, nhân viên bảo trì,…)

**Logs (Lịch sử sử dụng & bảo trì):**

* + id, equipment\_id (FK), user\_id (FK), hành động (thêm, sửa, xóa, bảo trì), timestamp, …

### 4. ****Quy trình phát triển theo SCRUM****

#### ****Sprint Planning:****

* **Product Backlog:** Tập hợp các user story:
  + “Với tư cách là quản trị viên, tôi muốn thêm, sửa, xóa trang thiết bị để quản lý tòa nhà.”
  + “Với tư cách là nhân viên bảo trì, tôi muốn nhận thông báo bảo trì để thực hiện kiểm tra và sửa chữa.”
  + “Với tư cách là người dùng, tôi muốn xem báo cáo thiết bị để nắm bắt tình trạng trang thiết bị.”
* **Ước tính và phân chia nhiệm vụ:**
  + Với nhóm 2 người, mỗi Sprint kéo dài 1-2 tuần.
  + Phân chia công việc: Một người tập trung vào backend (Django, database), người còn lại tập trung vào giao diện và tích hợp API.

#### ****Daily Scrum:****

* Họp ngắn hàng ngày (10-15 phút) để báo cáo tiến độ và giải quyết các trở ngại.

#### ****Sprint Review & Retrospective:****

* Demo các tính năng đã hoàn thành.
* Nhận phản hồi và điều chỉnh kế hoạch cho Sprint tiếp theo.

#### ****Quản lý mã nguồn trên GitHub:****

* Tạo repository và chia branch cho từng tính năng.
* Sử dụng Pull Request để review code.
* Sử dụng Issues và Projects (kanban) để theo dõi tiến độ và bug.