Chương Trình Training Cho Nhân Viên Mới

Bài toán: Xây dựng hệ thống quản lý công việc

I. Mục tiêu

1. Nắm vững kỹ thuật
    - Thiết kế và xây dựng các API RESTful chuyên nghiệp bằng Django và Django REST Framework.
    - Thiết kế CSDL quan hệ và sử dụng Django Models/ORM một cách hiệu quả.
    - Triển khai hệ thống xác thực an toàn bằng JWT (JSON Web Token).
    - Xử lý các tác vụ nền (background tasks) bằng Celery và Redis.
    - Viết Unit Test để đảm bảo chất lượng code.
2. Làm quen quy trình
    - Hiểu và áp dụng cấu trúc project chuẩn của công ty.
    - Thực hành quy trình làm việc với Git (Gitflow).
    - Viết code sạch (Clean Code), dễ đọc và dễ bảo trì.
3. Hoà nhập đội ngũ
    - Chủ động nghiên cứu và giải quyết vấn đề.
    - Sẵn sàng để tham gia vào các dự án thực tế của công ty.

II. Công nghệ bắt buộc

- Python 3.10+
- Django
- Django REST Framework
- SimpleJWT
- PostgreSQL
- Celery + Celery beat + Redis
- Pytest
- Docker & Docker Compose

III. Chức nâng yêu cầu

1. Module quản lí người dùng và phân quyền
    - Đăng ký: Tạo người dùng mới với vai trò mặc định là Staff. Admin có thể tạo người dùng với vai trò bất kỳ.
    - Xác thực: Đăng nhập để nhận về access và refresh token (JWT).
    - Phân quyền theo Role:
        - Admin: Toàn quyền hệ thống.
        - Manager: Có quyền quản lý (xem, sửa, xóa, gán việc) tất cả các công việc.
        - Staff: Chỉ có quyền trên các công việc mình tạo hoặc được giao.
    - Endpoint /api/users/me/ trả về thông tin người dùng đang đăng nhập

2. Module Quản Lý Công Việc (Tasks)
    - Model Task bao gồm các trường: title, description, status (Trạng thái), priority (Độ ưu tiên), owner (Người tạo), assignee (Người được giao), created_at, due_date (Hạn chót).
    - Trạng thái: Pending, In Progress, Completed, Cancelled
    - CRUD cho Task:
        - CREATE: Staff có thể tạo và tự gán cho mình. Manager và Admin có thể tạo và gán cho người dùng khác.
        - READ: Staff chỉ thấy task của mình. Manager và Admin thấy tất cả.
        - UPDATE: Staff chỉ được cập nhật task của mình. Manager và Admin được cập nhật mọi task.
        - DELETE: Chỉ Manager và Admin có quyền xóa.
    - Lọc và Tìm kiếm: Endpoint lấy danh sách công việc phải hỗ trợ lọc theo status, priority và tìm kiếm theo title.
3. Module Tác Vụ Nền
    - Thông báo qua Email (Celery Task):
        - Khi một công việc mới được tạo
        - Khi thay đổi trạng thái
    - Nhắc nhở Deadline (Celery Beat - Tác vụ định kỳ):
        - Mỗi ngày vào lúc 8:00 sáng, hệ thống quét tất cả các công việc có hạn chót trong vòng 24 giờ tới và gửi email nhắc nhở cho người được giao.
4. API Thông Kê
    - Endpoint /api/stats/ trả về thống kê công việc theo trạng thái và số lượng trong 7 ngày gần nhất
    - Chỉ Admin và Manager có thể truy cập.
5. Docker Hoá Toàn Bộ Hệ Thống
    - Các service: web, db, redis, worker, beat
    - Chạy hệ thống qua docker-compose up --build

IV. Thời gian

- Tổng thời gian hoàn thành: 4 ngày làm việc kể từ ngày nhận bài toán
- Gợi ý phân chia thời gian**:**
    - Ngày 1: Thiết lập môi trường, cấu trúc dự án, hoàn thành Module Quản lý Người dùng & Xác thực.
    - Ngày 2: Hoàn thành Module Quản lý Công việc, bao gồm logic CRUD và phân quyền chi tiết.
    - Ngày 3: Tích hợp Celery, Redis và hoàn thành các tác vụ nền (gửi mail, nhắc nhở).

- Ngày 4: Hoàn thành endpoint Báo cáo, viết Unit Test, dọn dẹp code và hoàn thiện tài liệu README.md

V. Tài liệu tham khảo

- https://docs.djangoproject.com/en/5.2/
- https://www.django-rest-framework.org/
- https://docs.celeryq.dev/en/stable/
- https://docs.celeryq.dev/en/stable/reference/celery.beat.html
- https://docs.docker.com/get-started/

VI. Tiêu chí đánh giá

Tiêu Chí	Cần Cải Thiện (0-4đ)	Đạt (5-7đ)	Xuất Sắc (8-10đ)
Hoàn thành Chức năng	Thiếu nhiều chức năng cốt lõi hoặc các chức năng chạy sai.	Hoàn thành đầy đủ các chức năng được yêu cầu.	Hoàn thành tất cả chức năng và có thể có thêm các cải tiến nhỏ, hợp lý.
Chất lượng Code	Code khó đọc, lặp lại, không theo chuẩn (PEP8).	Code được tổ chức tốt, dễ đọc, tuân thủ các nguyên tắc cơ bản.	Code rất sạch, cấu trúc rõ ràng, áp dụng các design pattern hiệu quả, dễ mở rộng.
Kiến trúc & Thiết kế	Cấu trúc project lộn xộn. Thiết kế model và API không hợp lý.	Cấu trúc project logic, tách biệt các module. Thiết kế API tuân thủ RESTful.	Kiến trúc hệ thống rõ ràng, thể hiện sự hiểu biết sâu sắc về các thành phần và sự tương tác giữa chúng.
Phân quyền & Bảo mật	Phân quyền sai, tồn tại lỗ hổng bảo mật (ví dụ: staff xem được task của người khác).	Áp dụng đúng các quyền cho từng vai trò. Các endpoint được bảo vệ.	Phân quyền chi tiết, chặt chẽ ở cả tầng database và API. Xử lý tốt các trường hợp edge case.
Kiểm thử (Testing)	Không có hoặc có rất ít test, độ bao phủ thấp.	Viết được Unit Test cho các logic quan trọng (happy path).	Test bao phủ toàn diện các chức năng và các trường hợp ngoại lệ (edge cases), giúp đảm bảo hệ thống ổn định.
Tài liệu & Quy trình	Không có README.md hoặc sơ sài. Lịch sử commit Git không rõ ràng.	Có file README.md hướng dẫn cài đặt và sử dụng. Tuân thủ quy trình Git cơ bản.	README.md rất chi tiết, chuyên nghiệp. Lịch sử commit sạch sẽ, tuân thủ Gitflow, dễ theo dõi.