Technical Exercise

Project Overview

Candidates will build a RESTful API for a simple **task management system** (or another domain if you prefer) using Django, Flask, or FastAPI. The API should allow users to:

- Create, read, update, and delete tasks.
- Assign tasks to users.
- Mark tasks as completed.
- Filter tasks based on status and due date.

Requirements

1. **API Development**
    - Use **Django (Django REST Framework), Flask, or FastAPI**.
    - Implement CRUD endpoints for managing tasks.
    - Use **JWT authentication**.
    - Implement **pagination** for listing tasks.
    - Support filtering tasks by status and due date.
2. **Database**
    - Use **PostgreSQL or SQLite** for data storage.
    - Design a simple schema for storing tasks and users.
3. **Unit Testing**
    - Write unit tests for critical endpoints using **pytest/unittest**.
    - Ensure at least **80% test coverage**.
4. **Dockerization**
    - Provide a **Dockerfile** and **docker-compose.yml** to set up the app and database locally.
5. **Documentation**
    - Include a **README.md** with setup instructions and document any key implementation decisions you consider relevant.
    - Provide API documentation using **Swagger (FastAPI) or drf-yasg (Django)**.
    - Use of any configuration file (pyproject.toml with poetry or uv), pre-commit or any other good practice that is considered necessary.
6. **Bonus Points**
    - Develop a basic front-end using Angular or React to interact with the API.
    - Add **rate limiting** to prevent API abuse.
    - Implement **background task processing** (e.g., Celery + Redis).
7. **Submission**
    - The code challenge has to be submitted in a github repository.

Presentation and Code Review

If necessary, you will need to present your project to the technical interview panel. During the presentation, you will need to explain the design decisions you made, which database you used, what requirements you had, show the tests, as well as the documentation and how to dockerize. If you have a front-end, this should also be shown. This will be done over Meet or Zoom and you will screen share either your GitHub repository or IDE.

After the presentation, the interview panel will conduct a code review of your project. You will be asked to explain your coding decisions and answer any questions related to the code.

The interview panel will evaluate your project based on the following criteria:

Clean Architecture: Your architecture should adhere to Clean Architecture principles, including separation of concerns and independence of components.

Test-Driven Development: Your project should follow TDD methodologies and include unit tests for all components.

Code quality: Your code should be well-organized, readable, and adhere to best practices.

Functionality: Your application should perform the required CRUD operations and user authentication without errors or bugs.

Presentation: Your presentation should be clear, concise, and demonstrate a good understanding of the project.