Project Title: Django Web Crawler for Product Scraping

Summary

The goal of this project is to build a Django-based web crawler to scrape product details from Esmerdis (https://www.esmerdis.com). The crawler should extract key product attributes such as name, price, description, and images, store them in a database, and expose the data via a REST API. The project should be containerized using Docker and designed to run periodically or on-demand.

—----—---—---------—---—---—---—-----—---—----—---—-----—---—-----—---—-----—---—-----—---—-----—----

Project Details

1. Scope of Work

- Develop a web crawler in Django to scrape product data from Esmerdis.
- Extract product details such as:
    - Product title
    - Price
    - Description
    - Images
    - Product category
    - Stock availability (if available)
- Store the scraped data in a database (PostgreSQL or MySQL).
- Implement a Django REST API (DRF) to retrieve stored products.
- Ensure the scraper can handle pagination and dynamically navigate through product pages.

2. Technical Stack

- **Backend**: Django
- **Scraping**: Scrapy / BeautifulSoup with requests or httpx
- **Database**: PostgreSQL / MySQL
- **API**: Django REST Framework (DRF)
- **Task Scheduling**: Celery + Redis (optional)
- **Containerization**: Docker

3. Features & Requirements

Minimum Requirements:

✅ Scrape product pages dynamically.

✅ Store data in a relational database.

✅ Provide an API to fetch stored product data.

Bonus Features (Optional, Extra Points):

🔹 Implement pagination handling.

🔹 Implement logging & error handling for failed requests.

🔹 Use Celery to run scheduled scraping tasks.

🔹 Store images locally or use cloud storage.

4. Expected Deliverables

- Django project with a crawler app
- Working scraper with pagination handling
- Product model in the database
- API endpoints for retrieving products (/api/products/)
- Dockerized setup (Dockerfile + docker-compose.yml)