DJ Mall :

1. Project Overview

DJ Mall (short for django mall) is a web-based ecommerce platform built with Django, designed to connect store owners and clients.
Store owners can post and manage their products, while clients can browse items, place orders, and pay cash on delivery (COD).
The platform simplifies online selling for small businesses and offers clients a secure, convenient way to shop without requiring online payments.

2. Stakeholders

- Product Owner: Mr Guelmouna.Z
- Development Team: **ARAB Yacine G1 SIR**
- End Users: Store owners and customers (clients)

3. Functional Requirements

- User registration and authentication (two roles: client and store owner).
- Product management (add, edit, delete, and view items).
- Order placement and tracking (status: pending, confirmed, delivered, etc.).
- Messaging/contact system between clients and store owners.
- Transaction history for both buyers and sellers.
- Notification system for store owners when a new order is placed.
- Cash on Delivery (COD) as the primary payment method.

4. Non-Functional Requirements

- High performance: Quick response times for browsing and ordering.
- Security: Secure authentication and data handling using Django's built-in protection.
- **Scalability:** Modular structure allowing more stores and users over time.
- Availability: Reliable access to data and services.
- Multilingual support: Ready for Arabic, French, and English through Django translation tools.

5. UI/UX Design

- Modern, minimalistic interface focused on clarity and ease of use.
- Responsive design adaptable to both desktop and mobile browsers.
- Dashboards:
- Client dashboard – view orders, update profile, contact sellers.
- Store dashboard – manage products, view incoming orders, update order statuses

6. Technical Specifications

- Platform: Web (desktop and mobile browsers).
- Backend: Django Framework (Python).
- Database: SQLite
- Frontend: Django Templates with Bootstrap for a clean and responsive UI.
- Authentication: Django user model with role-based permissions.

7. Risk Management

– Risk: Development delays due to unfamiliarity with Django.

  Mitigation: Task division among members and short weekly progress reviews.

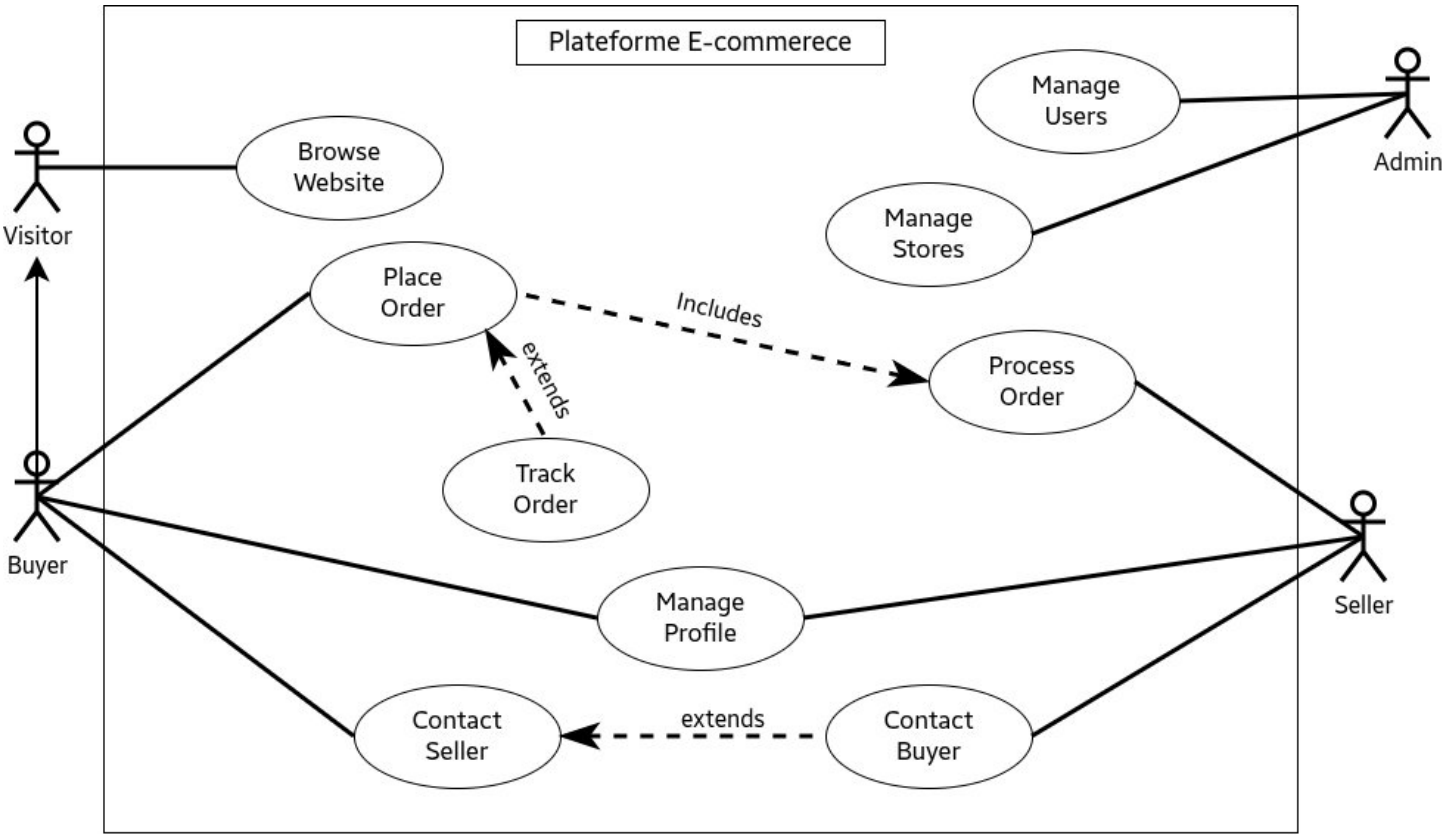
– Risk: Security vulnerabilities or data leaks.

  Mitigation: Use Django's built-in authentication, password hashing, and CSRF protection.

– Risk: Scope creep (adding too many features).

  Mitigation: Focus only on core features (auth, products, orders, messaging) for the first version.

8. Use Case Diagram



9. Class Diagram