

### Objective: Build a Scalable E-Commerce Backend API with product listing, cart management, order processing, and authentication.

### Assignment: Develop a High-Performance E-Commerce API

### Requirements:

1. **User Authentication:** Implement **JWT authentication** for users (Customers & Admins).
2. **Product Management:**
   1. API to **add, edit, delete, and fetch products**.
   2. Products should support **categories, stock availability, and images**.
3. **Cart System:** 
   1. Add/Remove items from the cart.
   2. Store cart data **persistently** for users.
4. **Order Processing:**
   1. Customers can **place an order** (cart → order).
   2. API to **track order status** (Pending, Shipped, Delivered).
5. **Discount & Coupon System:**
   1. Apply discount codes during checkout.
   2. Restrict coupons based on **expiry & user eligibility**.
6. **Scalability Considerations:**
   1. Implement **Redis caching** for faster product retrieval.
   2. Use **rate limiting** to prevent API abuse.

### Technical Requirements:

* Use **Flask** or **Django REST Framework (DRF)**.
* Use **MongoDB** as the database.
* Implement **background task processing (Celery/RQ) for order notifications**.
* Deploy APIs on **AWS with CI/CD setup**.
* Ensure **high security (input validation, rate limiting, encryption)**.

### Bonus (Optional Enhancements):

➕ Implement **GraphQL API** for more flexible data fetching.  
 ➕ Add **WebSockets for real-time order tracking**.  
 ➕ Create a **basic admin panel for managing orders and users**.

### Submission Guidelines:

📌 Submit via **GitHub repo**.  
 📌 Provide a **README.md** with API documentation.  
 📌 Deploy the API and share a **Postman collection or Swagger UI link**.

### Evaluation Criteria:

✔ **API design & efficiency** ✔ **Database optimization & indexing** ✔ **Security & authentication** ✔ **Performance scaling (caching, rate limiting, background tasks)** ✔ **Bonus features (if any)**

📩 **Submit your GitHub repo and API documentation within 3 days.**