■ Amazon E-commerce Mockup — Full Setup Guide

1. Environment Setup

Create and activate a virtual environment:
python -m venv venv
source venv/bin/activate # macOS/Linux
venv\Scripts\activate # Windows

Install dependencies using requirements.txt:
pip install -r requirements.txt

Setup `.env` file:
SECRET_KEY=your_secret_key
DEBUG=True
DATABASE_URL=postgres://user:password@localhost:5432/ecommerce
ALLOWED_HOSTS=localhost,127.0.0.1
JWT_SECRET=your_jwt_secret

2. Backend Setup (Django + DRF)

django-admin startproject backend
cd backend
python manage.py startapp users
python manage.py startapp ecommerce
Add apps in `settings.py` and configure PostgreSQL:
INSTALLED_APPS = [
...
'rest_framework',
'corsheaders',
'users',
'ecommerce',
]

DATABASES = {
'default': dj_database_url.config(default=os.getenv("DATABASE_URL"))
}

REST_FRAMEWORK = {
'DEFAULT_AUTHENTICATION_CLASSES':
('rest_framework_simplejwt.authentication.JWTAuthentication',)
}

3. Models & Serializers

Key models to create:

- User (extend AbstractUser)
- Product (name, description, price, stock, category, brand, image, rating)
- Category (name, slug)
- Order (user, total_price, shipping_address, payment_status)
- OrderItem (order, product, qty, price)
- Review (user, product, rating, comment)
For each model, create a matching DRF serializer.

4. API Endpoints

- Auth:
POST /api/users/register/
POST /api/users/login/
GET /api/users/profile/

- Products:
GET /api/products/
GET /api/products/:id/
POST /api/products/ (admin)
PUT /api/products/:id/
DELETE /api/products/:id/

- Orders:
POST /api/orders/
GET /api/orders/myorders/
GET /api/orders/:id/

- Reviews:
POST /api/products/:id/reviews/

5. Frontend Setup (React)

npx create-react-app frontend
cd frontend
npm install axios react-router-dom redux @reduxjs/toolkit
npm install react-icons react-toastify tailwindcss
Recommended project structure:
src/
components/
pages/
redux/
utils/
App.js
Setup Redux slices for user, product, cart, and order.

6. Integration

- Enable CORS in Django
- Connect frontend to backend using Axios
- Store JWT in localStorage
- Setup protected routes in React
- Handle login/logout and auto-refresh tokens

7. Extras

- Payments: integrate Stripe Checkout
- File storage: configure AWS S3 with boto3
- API docs: drf-yasg or drf-spectacular
- Deployment: Render/Railway (backend) + Netlify/Vercel (frontend)
- Freeze versions:
pip freeze > requirements.lock.txt