Guide: Building an AliExpress Clone with Django Rest Framework (DRF)
==

This guide outlines the steps from project planning through production deployment for developing an e-commerce API using DRF.

Phase 1: Planning and Setup
----

1. **Define Requirements & Features:** Outline core e-commerce functionalities (Auth, Products, Cart, Orders, Payments, Search, Admin).
2. **Design the Database (ERD):** Structure your models (User, Product, Category, Order, etc.) with an Entity-Relationship Diagram.
3. **Set Up the Development Environment:** Install Python, create a virtual environment, install Django and DRF (`pip install django djangorestframework`), and choose a production database like PostgreSQL.
4. **Initialize the Project:** Set up the main Django project and required apps (products, orders, users).

Phase 2: Backend Development (DRF)
----

1. **Define Models:** Write your database models in `models.py` and run migrations (`makemigrations`, `migrate`).
2. **Create Serializers:** Develop classes to handle data conversion between Django models and JSON format.
3. **Build Views and URLs:** Implement API logic using ViewSets or generic views and map them to URL endpoints.
4. **Implement Authentication:** Secure API endpoints using systems like JWT or token authentication and set up permissions.
5. **Add Core Features:** Integrate payment gateways (Stripe, PayPal) and configure static file/image storage (AWS S3).
6. **Enable CORS:** Configure `django-cors-headers` to allow access from your separate frontend application.

Phase 3: Testing and Frontend Integration
----

1. **Test the API:** Use tools like Postman or Pytest to test all endpoints thoroughly.
2. **Develop the Frontend:** Build a separate UI using React, Vue, or Angular to consume the API endpoints.

Phase 4: Production Deployment
----

1. **Prepare for Production:** Create `requirements.txt`, disable DEBUG, set environment variables for secrets, configure static file serving (WhiteNoise).
2. **Containerization:** Use Docker to containerize your application and database.
3. **Choose a Hosting Provider:** Render, Railway, AWS, etc.
4. **Set Up CI/CD:** Automate testing and deployment with GitHub Actions.
5. **Deploy:** Deploy containers, connect production database, verify environment variables.
6. **Monitoring and Maintenance:** Implement logging and monitoring tools to maintain system health.