ChatGPT link :- https://chatgpt.com/share/68ac1b23-556c-8007-a6e6-dbb4f1f049fb

Great question 👍 For your **Online Food Ordering System (Hibernate + JSP)**, let's break it down into **models (Hibernate entities)** + **relationships** so you know how many to develop.

🏗️ Core Models (Hibernate Entities)

You'll typically need **7–8 core models**:

1. **User** 👤
   - Fields: id, name, email, password, role (ADMIN/CUSTOMER/DELIVERY), phone
   - Relationships:
     - One User → Many Address
     - One User → Many Order
     - One User → Many Review
     - One User → One Cart

---

2. **Address** 🏠
   - Fields: id, user_id, line1, line2, city, pincode, landmark
   - Relationships:
     - Many Address → One User
     - One Address → Many Order

---

3. **Restaurant** 🍽️
   - Fields: id, name, cuisine, rating, open
   - Relationships:
     - One Restaurant → Many MenuItem

        - One Restaurant → Many Review

---

4. **MenuItem** 🍔
    - Fields: id, restaurant_id, name, description, price, veg, available
    - Relationships:
        - Many MenuItem → One Restaurant
        - Many MenuItem → Many CartItem
        - Many MenuItem → Many OrderItem

---

5. **Cart** 🛒
    - Fields: id, user_id, updatedAt
    - Relationships:
        - One Cart → One User
        - One Cart → Many CartItem

---

6. **CartItem** 🛍️
    - Fields: id, cart_id, menu_item_id, qty
    - Relationships:
        - Many CartItem → One Cart
        - Many CartItem → One MenuItem

---

7. **Order** 📦
    - Fields: id, user_id, address_id, status, total, paymentMode, createdAt
    - Relationships:
        - Many Order → One User
        - Many Order → One Address
        - One Order → Many OrderItem

---

8. **OrderItem** 📋
    - Fields: id, order_id, menu_item_id, price, qty
    - Relationships:

        - Many OrderItem → One Order
        - Many OrderItem → One MenuItem

---

9. **Review** ⭐
    - Fields: id, user_id, restaurant_id, rating, comment, createdAt
    - Relationships:
        - Many Review → One User
        - Many Review → One Restaurant

🔗 **Connection Overview**

Here's how everything connects:

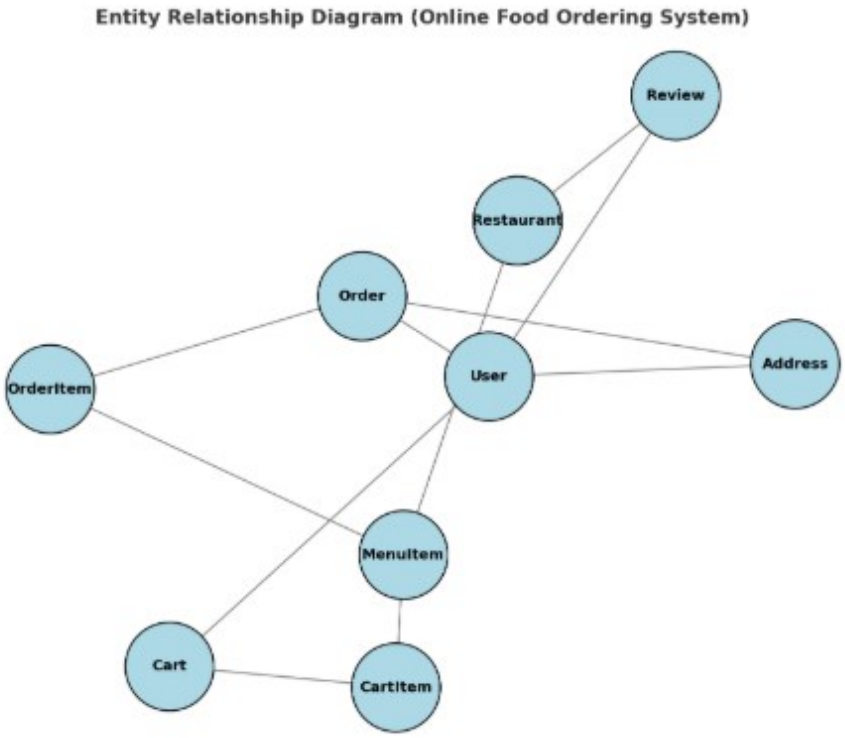
- **User ↔ Address** → One-to-Many
- **User ↔ Cart** → One-to-One
- **User ↔ Order** → One-to-Many
- **User ↔ Review** → One-to-Many
- **Restaurant ↔ MenuItem** → One-to-Many
- **Restaurant ↔ Review** → One-to-Many
- **MenuItem ↔ CartItem** → Many-to-One
- **MenuItem ↔ OrderItem** → Many-to-One
- **Cart ↔ CartItem** → One-to-Many
- **Order ↔ OrderItem** → One-to-Many

✅ That means you'll need to build **9 models/entities** in Hibernate.
✅ Once entities are ready, you'll connect them with annotations like:

@OneToMany(mappedBy = "user")

private List<Order> orders;

and so on.



🛠️ **Development Roadmap**

Week 1 – Setup & Planning

- Install tools:
    - **Backend**: Eclipse/IntelliJ, Tomcat, Hibernate JARs, MySQL
    - **Frontend**: JSP + CSS + Bootstrap (optional for styling)
    - **Collaboration**: GitHub repo (one person creates repo, other clones)
- Create pom.xml (if using Maven) with Hibernate + MySQL dependencies.
- Design **ERD (entities + relationships)** – done ✅
- Divide work:
    - Friend A → *User, Cart, CartItem, Order, OrderItem*
    - Friend B → *Restaurant, MenuItem, Address, Review*

Week 2 – Database & Hibernate Entities

- Create MySQL database (food_ordering).

- Define tables using Hibernate annotations (@Entity, @OneToMany, etc.).
- Implement entities for all 9 models.
- Test connections with Hibernate SessionFactory.
- Save and fetch sample data (ex: add a restaurant, user, menu item).

Week 3 – Basic JSP Pages & DAO Layer

- Implement **DAO (Data Access Objects)** for CRUD:
    - UserDAO → register/login users
    - RestaurantDAO → add/view restaurants
    - MenuItemDAO → add/view food items
    - OrderDAO → place/fetch orders
- Create simple JSP pages:
    - register.jsp, login.jsp
    - home.jsp (list restaurants)
- Connect JSP forms → Servlet → DAO → Hibernate

Week 4 – Cart & Orders

- Add Cart functionality:
    - addToCart.jsp, viewCart.jsp
    - Cart → CartItem mapping
- Implement placing order:
    - checkout.jsp (select address + payment)
    - Save into Order + OrderItem tables
- Show order history (myOrders.jsp).

Week 5 – Reviews & Admin Features

- Add reviews:
    - addReview.jsp, show ratings under restaurants
- Add Admin features:
    - Admin login → manage restaurants & menu items
    - Approve/reject user reviews

Week 6 – UI Polish & Testing

- Improve UI with Bootstrap (cards, navbar, tables).
- Add validations (empty fields, password rules, etc.).
- Debug Hibernate lazy loading errors (common issue).
- Write test data for at least:
    - 5 restaurants, 20 menu items, 10 users, 30 orders.
- Test with multiple users ordering at the same time.

Week 7 – Final Touch & Deployment

- Add session management (login required to order).
- Add proper redirects (after login → home, after order → orders page).
- Clean up unused code, comments, logs.
- Deploy on **Tomcat** locally.
- (Optional) Deploy on AWS free tier