1. **Social Network Analysis Tool**

**Objective**:  
Develop a tool to model and analyze a basic social network using Java Collections. This tool will track users, their connections (friends/followers), and provide analysis such as finding mutual friends, suggesting new friends, and identifying social cliques.

**Features**

1. **User Management**:
   * Add a new user.
   * Remove a user.
   * View all users.
2. **Connections**:
   * Add a connection (friendship) between two users.
   * Remove a connection.
   * Find mutual friends between two users.
   * Suggest potential friends based on mutual connections.
3. **Analysis**:
   * Find social cliques (groups of users who are all connected).
   * Identify the most connected users (top influencers).
   * Determine the shortest path between two users.

### ****Implementation Details****

#### Classes and Data Structures:

1. **User Class**:
   * Fields: id, name, friends (Set<Integer>).
2. **SocialNetwork Class**:
   * Use a HashMap<Integer, User> to store users by their IDs.
   * Use a HashMap<Integer, Set<Integer>> to represent the adjacency list for connections.

### ****E-Commerce Inventory and Order Management System****

**Objective**:  
Build a robust system for managing an e-commerce platform’s inventory, customer orders, and analytics. This project will utilize advanced Java Collections and focus on handling large data efficiently.

### ****Features****

1. **Inventory Management**:
   * Add, update, and remove products.
   * View all products.
   * Search for products by name, category, or price range.
2. **Order Management**:
   * Create and manage customer orders.
   * Track order status (e.g., Pending, Shipped, Delivered).
   * Generate invoices for orders.
3. **Customer Management**:
   * Add, update, and delete customer records.
   * View customer order history.
4. **Analytics**:
   * Find the most sold products.
   * Generate sales reports by category.
   * Calculate revenue over a time period.
5. **Optional Features**:
   * Apply discounts and promotional codes.
   * Predict inventory shortages based on sales trends.

### ****Implementation Details****

#### Data Structures:

1. **Product Class**:
   * Fields: id, name, category, price, stock.
   * Use a HashMap<Integer, Product> to manage inventory with id as the key.
2. **Order Class**:
   * Fields: orderId, customerId, productIds (List<Integer>), status, totalAmount, orderDate.
   * Use a HashMap<Integer, Order> to manage orders with orderId as the key.
3. **Customer Class**:
   * Fields: customerId, name, email, orders (List<Integer>).
   * Use a HashMap<Integer, Customer> to store customers with customerId as the key.
4. **Sales Analysis**:
   * Use a HashMap<String, Integer> to count product sales by category or individual product.