**Building a Fitting Room Simulation**

We will explore the problem statement, discuss our solution approach, and examine the classes that make it all come together and implement it.

**Problem Statement:**

* Imagine you are tasked with creating a simulation of a retail clothing store.
* Your goal is to model the store's fitting room system, complete with waiting areas, chairs, and fitting rooms.
* Customers will enter, try on clothes, and eventually leave the store.
* The challenge is to use Java and threading to implement this simulation efficiently.
* We will also employ semaphores, mutexes, and a client-server architecture for scalability and fault tolerance.

**Solution Approach:**

Let us outline our solution approach before diving into the details. We will create four main classes to achieve our goal:

1. FittingRoomSimulation: The main class responsible for program entry and coordination.

2. CentralServer: Manages the allocation of fitting rooms and communication with clients.

3. FittingRoomServer: Represents individual fitting room servers.

4. Client: Simulates customer behavior, requests fitting rooms, and communicates with the central server.

**Class Descriptions and Tasks:**

1. FittingRoomSimulation Class:

* This is our program's entry point.
* It takes command-line arguments for simulation parameters like time, the number of fitting rooms, and customers.
* Initializes central and fitting room servers.
* Creates customer threads and starts them.
* Simulates time and handles cleanup.

2. CentralServer Class:

* Represents the central server.
* Listens for incoming client connections.
* Creates a new thread (ClientHandler) for each client.
* Maintains a list of client sockets.
* Handles communication with individual customers.

3. FittingRoomServer Class:

* Represents a fitting room server.
* Listens for incoming client connections from customers.
* Allocates fitting rooms and manages their state.

4. Client Class:

* Represents a customer thread.
* Simulates customer entry and behavior.
* Connects to the central server using sockets.
* Sleeps for random intervals before entering.
* Sends requests to allocate and release fitting rooms.
* Prints messages indicating actions and simulates time spent in waiting areas and fitting rooms.