Flowable Hands-on Exercise: Signal Event

Scenario: Order Processing with Global Cancellation

A company processes customer **orders**, which go through a series of steps: **verification, packaging, and shipping**. However, if the customer **cancels the order**, all active order processes should be **terminated immediately** using a **Signal Event**.

This exercise will demonstrate how to:
✅ Use a **Signal Start Event** to trigger processes globally.
✅ Use an **Intermediate Throw Signal Event** to broadcast a cancellation signal.
✅ Use a **Signal Boundary Event** to interrupt running processes.

Step 1: Create a New BPMN Process in Flowable

1. Open **Flowable Modeler**.
2. Create a new **BPMN Process Model** named **Order Processing**.
3. Set **Process ID** as `orderProcessing`.

Step 2: Define the Process Flow

1⃣ Start Event

- Drag a **Start Event** onto the canvas.
- Name it **Order Received**.

2⃣ User Tasks (Order Processing Steps)

- Add three **User Tasks** named:
  1. **Verify Order**
  2. **Package Order**
  3. **Ship Order**
- Connect them in sequence.

3⃣ Signal Boundary Event (Order Cancellation)

- Drag a **Signal Boundary Event** onto the **Verify Order** task.
- Name it **Order Canceled**.
- Configure it to listen for a signal named `"orderCanceled"`.
- Set it as **Interrupting**, so it immediately stops the process.
- **Repeat this step for "Package Order" and "Ship Order" tasks** to allow cancellation at any stage.

4⃣ End Event (Order Completed)

- Connect the last task (**Ship Order**) to an **End Event** labeled **Order Completed**.

Step 3: Define a Global Order Cancellation Process

1. Create a **new BPMN Process Model** named **Order Cancellation**.
2. Set **Process ID** as `orderCancellation`.
3. Drag a **Signal Start Event** onto the canvas.
   - Name it **Global Order Cancellation Trigger**.
   - Configure it to listen for `"orderCanceled"`.
4. Add a **User Task** labeled **Notify Customer**.
5. Connect to an **End Event** labeled **Cancellation Completed**.

Step 4: Define a Signal Throwing Event

1. In the **Order Cancellation Process**, drag an **Intermediate Throw Signal Event**.
2. Name it **Send Order Canceled Signal**.
3. Configure it to send the **"orderCanceled"** signal.
4. Connect it before the **Notify Customer** task.

Step 5: Deploy and Test

1. Deploy both processes: **Order Processing** and **Order Cancellation**.
2. Start a new **Order Processing** instance with:

   ```
   {
     "orderId": "ORD123",
     "customerName": "Alice"
   }
   ```

3. Start a separate **Order Cancellation** instance with:

   ```
   {
     "orderId": "ORD123"
   }
   ```

4. Observe the following behaviors:
   - If the **order is canceled**, all active **Order Processing** instances are **terminated immediately**.
   - If there is no cancellation, the order follows the normal **processing flow**.

Expected Behavior

Scenario	Signal Event Triggered	Outcome
Normal order processing	✕ No	Order is verified, packaged, and shipped
Order is canceled	✓ Yes	Active process is interrupted, and customer is notified