In this architecture, Visibility Timeout and the DeadLetter Queue (DLQ) play a critical role..

Visibility Timeout: Preventing Duplicate Processing

When a message is delivered from the `OrderQueue` to the Lambda function, it remains in the queue but is hidden (invisible) for a certain period—known as the visibility timeout.

This is important because if the Lambda function fails to process the message within that time, it becomes visible again and can be retried by another instance. This ensures messages are not lost and can be reprocessed if there's a transient error (e.g., timeout, throttling, temporary DynamoDB issue). It prevents duplicate processing, because while a message is invisible, other consumers won't see it.

DeadLetter Queue (DLQ): Handling Persistent Failures Gracefully

The DLQ is configured as a fallback mechanism when a message repeatedly fails. I set `` `maxReceiveCount` = 3` ``, meaning that If a message is retried 3 times without successful processing, it is moved to the DLQ (`` `OrderDLQ` ``). This isolates bad or poison messages that could be malformed, corrupted, or cause code exceptions. DLQ enables troubleshooting and manual intervention without impacting the rest of the system. So if a message causes the Lambda to fail 3 times, it gets sent to the DLQ for review, not retried endlessly.

Combined Benefit
Together, visibility timeout and the DLQ ensure:

- No message loss
- No infinite retries
- Clear paths for debugging failures
- fault tolerant processing