Overall Logic

For Kafka retry and error handling, an outside service ( Exception-healer) was used.

The logic that was used for both Kafka Producer and Consumer was the following:

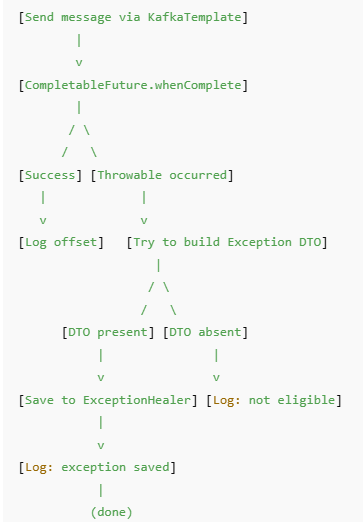
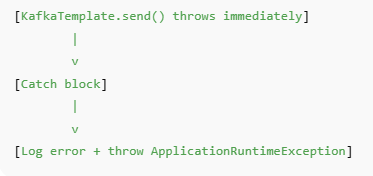
-if there exists an error, either if kafka is down, or some publishing error, database error etc, **we need to save the message that is sent /received so it will not be lost** due to this exceptions that until now were unhandled, if something went wrong, the message containing the important data was lost.

- Once the exception is added to exception-healer, it will be reprocessed based on the configuration made in there. A separate endpoint for reprocessing is specified along with polling interval, number of retries etc.

- If after reprocessing for the configured times, the error is still not resolved, the status will be changed to Failed and the exception will be stored for manual reprocessing

Producer Error Handling

The ELKafkaProducer is responsible for publishing messages to Kafka for the Engagement Letter flow. If failure happens when sending to Kafka the exception is saved into exception healer for further reprocessing .

If kafka is up: If Kafka is Down

Consumer Error Handling

The consumer receives a message, it deserializes it and attempts to insert the account form and user and form version and datapoints into the DB.

If errors occurs during database inserts, the exception will be saved, the message will not be lost . If error occurs during deserialization or other error occur during that phase, an error will be thrown and the message from kafka will stay unprocessed.



Exception Reprocessing logic

Once the exception is saved in exception healer, it will be retried based on configured rules. A reprocess module in engagement letter will have endpoints on wich the healer will reprocess the exceptions, so that exceptions are reprocessed according to their types.

Logic: for each task type(consumer, producer) there is and endpoint, when exception healer posts the exception dto on the endpoint, in the ReprocessingService there is configuration for the message to retake the normal flow intended in consumer/producer

Demo

Because setting up a local kafka broker for testing is not feasible because of the SASL\_SSL protocol needing to be configured on the local broker . Therefore the closest thing to test the code are unit tests on producer and consumer, that simulate cases like: kafka is down or Db failures.

To test the implementation of exception healer and check that reprocessing works I have forced an exception when kafka broker is down, so if the name of the producer is wrong, the publishing will throw and exception, then the exception and the message that should have been sent will be saved for reprocessing in exception healer.Once there, exception healer will reprocess it following the configuration setted up for the specific task type.

For the demo presentation I will send a video describing what services needs to be turned on ti work. A little overview:

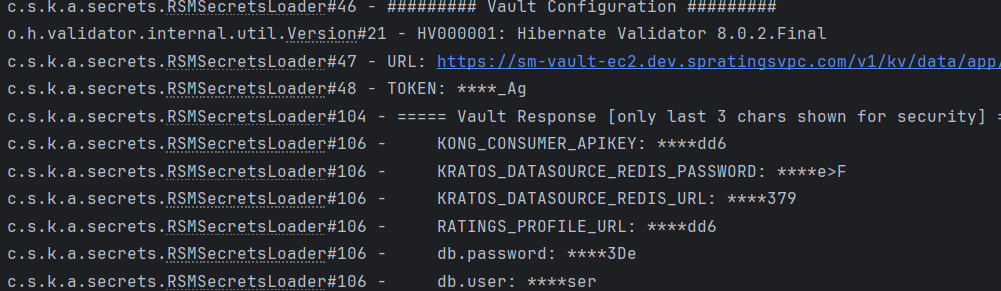
Start Engagement-letter with the kafka producer ENV value modified(just inert a number in front) 🡪 Start Exception-Healer service 🡪 Take a valid token, add it to swagger 🡪 Take a valid payload( there is one in Postman) and send it on engagement-letter Save-Form endpoint 🡪 It will save the exception to exception healer

Now, for the reprocessing to work kafka needs to work, **so change the ENV** value back and restart the app

Exception healer will poll for the task **IF it is a redis Leader,** otherwise it will not, so for completing the reprocessing 🡪 Check the Reprocess Task ID in the Db( Reprocess\_Task table) 🡪 go to exception-healer swagger, on the reprocess task endpoint add the reprocess task ID, send it 🡪 there will be an auth error(because of token) but the task will succeed 🡪Check it in engagement-letter logs

KAMLESH

After describing what was done, see what he says to proceed with next, mainly testing to see how all cases success/error work in real conditions.

Another question would be How to configure, or whom to ask for **adding okta token credentials in the vault** to have them at the start of the app. Currently we took the secrets from exception healer to use the token generation functionality. It should appear here: 

And this are the variables that are needed

