package com.twitter.product\_mixer.core.service.debug\_query

import com.twitter.finagle.Service

import com.twitter.product\_mixer.core.pipeline.pipeline\_failure.PipelineFailure

import com.twitter.product\_mixer.core.pipeline.pipeline\_failure.ProductDisabled

import com.twitter.product\_mixer.core.pipeline.product.ProductPipelineResult

import com.twitter.scrooge.{Request => ScroogeRequest}

import com.twitter.scrooge.{Response => ScroogeResponse}

import com.twitter.util.Future

import com.twitter.product\_mixer.core.{thriftscala => t}

import com.twitter.util.jackson.ScalaObjectMapper

/\*\*

\* All Mixers must implement a debug query interface. This can be a problem for in-place migrations

\* where a service may only partially implement Product Mixer patterns. This service can be used as

\* a noop implementation of [[DebugQueryService]] until the Mixer service is fully migrated.

\*/

object DebugQueryNotSupportedService

extends Service[ScroogeRequest[\_], ScroogeResponse[t.PipelineExecutionResult]] {

val failureJson: String = {

val message = "This service does not support debug queries, this is usually due to an active " +

"in-place migration to Product Mixer. Please reach out in #product-mixer if you have any questions."

ScalaObjectMapper().writeValueAsString(

ProductPipelineResult(

transformedQuery = None,

qualityFactorResult = None,

gateResult = None,

pipelineSelectorResult = None,

mixerPipelineResult = None,

recommendationPipelineResult = None,

traceId = None,

failure = Some(PipelineFailure(ProductDisabled, message)),

result = None,

))

}

override def apply(

thriftRequest: ScroogeRequest[\_]

): Future[ScroogeResponse[t.PipelineExecutionResult]] =

Future.value(ScroogeResponse(t.PipelineExecutionResult(failureJson)))

}