package com.twitter.tweetypie

package service

import com.twitter.quill.capture.QuillCapture

import com.twitter.tweetypie.thriftscala.\_

import org.apache.thrift.transport.TMemoryBuffer

import com.twitter.finagle.thrift.Protocols

import com.twitter.quill.capture.Payloads

import com.twitter.tweetypie.service.QuillTweetService.createThriftBinaryRequest

import org.apache.thrift.protocol.TMessage

import org.apache.thrift.protocol.TMessageType

import org.apache.thrift.protocol.TProtocol

object QuillTweetService {

// Construct the byte stream for a binary thrift request

def createThriftBinaryRequest(method\_name: String, write\_args: TProtocol => Unit): Array[Byte] = {

val buf = new TMemoryBuffer(512)

val oprot = Protocols.binaryFactory().getProtocol(buf)

oprot.writeMessageBegin(new TMessage(method\_name, TMessageType.CALL, 0))

write\_args(oprot)

oprot.writeMessageEnd()

// Return bytes

java.util.Arrays.copyOfRange(buf.getArray, 0, buf.length)

}

}

/\*\*

\* Wraps an underlying TweetService, logging some requests.

\*/

class QuillTweetService(quillCapture: QuillCapture, protected val underlying: ThriftTweetService)

extends TweetServiceProxy {

override def postTweet(request: PostTweetRequest): Future[PostTweetResult] = {

val requestBytes = createThriftBinaryRequest(

TweetService.PostTweet.name,

TweetService.PostTweet.Args(request).write)

quillCapture.storeServerRecv(Payloads.fromThriftMessageBytes(requestBytes))

underlying.postTweet(request)

}

override def deleteTweets(request: DeleteTweetsRequest): Future[Seq[DeleteTweetResult]] = {

val requestBytes = createThriftBinaryRequest(

TweetService.DeleteTweets.name,

TweetService.DeleteTweets.Args(request).write)

quillCapture.storeServerRecv(Payloads.fromThriftMessageBytes(requestBytes))

underlying.deleteTweets(request)

}

override def postRetweet(request: RetweetRequest): Future[PostTweetResult] = {

val requestBytes = createThriftBinaryRequest(

TweetService.PostRetweet.name,

TweetService.PostRetweet.Args(request).write)

quillCapture.storeServerRecv(Payloads.fromThriftMessageBytes(requestBytes))

underlying.postRetweet(request)

}

override def unretweet(request: UnretweetRequest): Future[UnretweetResult] = {

val requestBytes = createThriftBinaryRequest(

TweetService.Unretweet.name,

TweetService.Unretweet.Args(request).write)

quillCapture.storeServerRecv(Payloads.fromThriftMessageBytes(requestBytes))

underlying.unretweet(request)

}

override def cascadedDeleteTweet(request: CascadedDeleteTweetRequest): Future[Unit] = {

val requestBytes = createThriftBinaryRequest(

TweetServiceInternal.CascadedDeleteTweet.name,

TweetServiceInternal.CascadedDeleteTweet.Args(request).write)

quillCapture.storeServerRecv(Payloads.fromThriftMessageBytes(requestBytes))

underlying.cascadedDeleteTweet(request)

}

}