package com.twitter.recos.user\_user\_graph

import com.twitter.logging.Logger

import com.twitter.recos.user\_user\_graph.thriftscala.\_

import com.twitter.util.Future

trait LoggingUserUserGraph extends thriftscala.UserUserGraph.MethodPerEndpoint {

private[this] val accessLog = Logger("access")

abstract override def recommendUsers(

request: RecommendUserRequest

): Future[RecommendUserResponse] = {

val time = System.currentTimeMillis

super.recommendUsers(request) onSuccess { resp =>

val timeTaken = System.currentTimeMillis - time

val logText =

s"In ${timeTaken}ms, recommendUsers(${requestToString(request)}), response ${responseToString(resp)}"

accessLog.info(logText)

} onFailure { exc =>

val timeTaken = System.currentTimeMillis - time

val logText = s"In ${timeTaken}ms, recommendUsers(${requestToString(request)} returned error"

accessLog.error(exc, logText)

}

}

private def requestToString(request: RecommendUserRequest): String = {

Seq(

request.requesterId,

request.displayLocation,

request.seedsWithWeights.size,

request.seedsWithWeights.take(5),

request.excludedUserIds.map(\_.size).getOrElse(0),

request.excludedUserIds.map(\_.take(5)),

request.maxNumResults,

request.maxNumSocialProofs,

request.minUserPerSocialProof,

request.socialProofTypes,

request.maxEdgeEngagementAgeInMillis

).mkString(",")

}

private def responseToString(response: RecommendUserResponse): String = {

response.recommendedUsers.toList.map { recUser =>

val socialProof = recUser.socialProofs.map {

case (proofType, proofs) =>

(proofType, proofs)

}

(recUser.userId, recUser.score, socialProof)

}.toString

}

}