package com.twitter.cr\_mixer.source\_signal

import com.twitter.core\_workflows.user\_model.thriftscala.UserState

import com.twitter.cr\_mixer.model.GraphSourceInfo

import com.twitter.cr\_mixer.model.SourceInfo

import com.twitter.cr\_mixer.source\_signal.SourceFetcher.FetcherQuery

import com.twitter.cr\_mixer.thriftscala.SourceType

import com.twitter.cr\_mixer.thriftscala.{Product => TProduct}

import com.twitter.simclusters\_v2.common.UserId

import com.twitter.timelines.configapi

import com.twitter.util.Future

import javax.inject.Inject

import javax.inject.Singleton

@Singleton

case class SourceInfoRouter @Inject() (

ussSourceSignalFetcher: UssSourceSignalFetcher,

frsSourceSignalFetcher: FrsSourceSignalFetcher,

frsSourceGraphFetcher: FrsSourceGraphFetcher,

realGraphOonSourceGraphFetcher: RealGraphOonSourceGraphFetcher,

realGraphInSourceGraphFetcher: RealGraphInSourceGraphFetcher,

) {

def get(

userId: UserId,

product: TProduct,

userState: UserState,

params: configapi.Params

): Future[(Set[SourceInfo], Map[String, Option[GraphSourceInfo]])] = {

val fetcherQuery = FetcherQuery(userId, product, userState, params)

Future.join(

getSourceSignals(fetcherQuery),

getSourceGraphs(fetcherQuery)

)

}

private def getSourceSignals(

fetcherQuery: FetcherQuery

): Future[Set[SourceInfo]] = {

Future

.join(

ussSourceSignalFetcher.get(fetcherQuery),

frsSourceSignalFetcher.get(fetcherQuery)).map {

case (ussSignalsOpt, frsSignalsOpt) =>

(ussSignalsOpt.getOrElse(Seq.empty) ++ frsSignalsOpt.getOrElse(Seq.empty)).toSet

}

}

private def getSourceGraphs(

fetcherQuery: FetcherQuery

): Future[Map[String, Option[GraphSourceInfo]]] = {

Future

.join(

frsSourceGraphFetcher.get(fetcherQuery),

realGraphOonSourceGraphFetcher.get(fetcherQuery),

realGraphInSourceGraphFetcher.get(fetcherQuery)

).map {

case (frsGraphOpt, realGraphOonGraphOpt, realGraphInGraphOpt) =>

Map(

SourceType.FollowRecommendation.name -> frsGraphOpt,

SourceType.RealGraphOon.name -> realGraphOonGraphOpt,

SourceType.RealGraphIn.name -> realGraphInGraphOpt,

)

}

}

}