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

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.frigate.common.util.StatsUtil

import com.twitter.util.Future

/\*\*\*

\* A SourceSignalFetcher is a trait that extends from `SourceFetcher`

\* and is specialized in tackling Signals (eg., USS, FRS) fetch.

\* Currently, we define Signals as (but not limited to) a set of past engagements that

\* the user makes, such as RecentFav, RecentFollow, etc.

\*

\* The [[ResultType]] of a SourceSignalFetcher is `Seq[SourceInfo]`. When we pass in userId,

\* the underlying store returns a list of signals.

\*/

trait SourceSignalFetcher extends SourceFetcher[Seq[SourceInfo]] {

protected type SignalConvertType

def trackStats(

query: FetcherQuery

)(

func: => Future[Option[Seq[SourceInfo]]]

): Future[Option[Seq[SourceInfo]]] = {

val productScopedStats = stats.scope(query.product.originalName)

val productUserStateScopedStats = productScopedStats.scope(query.userState.toString)

StatsUtil

.trackOptionItemsStats(productScopedStats) {

StatsUtil

.trackOptionItemsStats(productUserStateScopedStats) {

func

}

}

}

/\*\*\*

\* Convert a list of Signals of type [[SignalConvertType]] into SourceInfo

\*/

def convertSourceInfo(

sourceType: SourceType,

signals: Seq[SignalConvertType]

): Seq[SourceInfo]

}