package com.twitter.timelineranker.recap\_author

import com.twitter.timelineranker.model.CandidateTweetsResult

import com.twitter.timelineranker.model.RecapQuery

import com.twitter.util.Future

import com.twitter.timelineranker.model.RecapQuery.DependencyProvider

import com.twitter.timelineranker.parameters.recap\_author.RecapAuthorParams

/\*\*

\* A repository of recap author results.

\*

\* For now, it does not cache any results therefore forwards all calls to the underlying source.

\*/

class RecapAuthorRepository(source: RecapAuthorSource, realtimeCGSource: RecapAuthorSource) {

private[this] val enableRealtimeCGProvider =

DependencyProvider.from(RecapAuthorParams.EnableEarlybirdRealtimeCgMigrationParam)

def get(query: RecapQuery): Future[CandidateTweetsResult] = {

if (enableRealtimeCGProvider(query)) {

realtimeCGSource.get(query)

} else {

source.get(query)

}

}

def get(queries: Seq[RecapQuery]): Future[Seq[CandidateTweetsResult]] = {

Future.collect(queries.map(query => get(query)))

}

}