package com.twitter.follow\_recommendations.common.candidate\_sources.sims

import com.google.inject.Singleton

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.hermit.model.Algorithm

import com.twitter.product\_mixer.core.model.common.identifier.CandidateSourceIdentifier

import com.twitter.strato.generated.client.recommendations.similarity.SimilarUsersBySimsExperimentalOnUserClientColumn

import com.twitter.util.Duration

import javax.inject.Inject

@Singleton

class SimsExperimentalStore @Inject() (

simsExperimentalOnUserClientColumn: SimilarUsersBySimsExperimentalOnUserClientColumn)

extends StratoBasedSimsCandidateSourceWithUnitView(

fetcher = simsExperimentalOnUserClientColumn.fetcher,

identifier = SimsExperimentalStore.Identifier

)

@Singleton

class CachedSimsExperimentalStore @Inject() (

simsExperimentalOnUserClientColumn: SimilarUsersBySimsExperimentalOnUserClientColumn,

statsReceiver: StatsReceiver)

extends CacheBasedSimsStore(

id = SimsExperimentalStore.Identifier,

fetcher = simsExperimentalOnUserClientColumn.fetcher,

maxCacheSize = SimsExperimentalStore.MaxCacheSize,

cacheTtl = SimsExperimentalStore.CacheTTL,

statsReceiver = statsReceiver.scope("CachedSimsExperimentalStore", "cache")

)

object SimsExperimentalStore {

val Identifier = CandidateSourceIdentifier(Algorithm.Sims.toString)

val MaxCacheSize = 1000

val CacheTTL: Duration = Duration.fromHours(12)

}