package com.twitter.timelineranker.repository

import com.twitter.conversions.DurationOps.\_

import com.twitter.timelineranker.config.RequestScopes

import com.twitter.timelineranker.config.RuntimeConfiguration

import com.twitter.timelineranker.parameters.ConfigBuilder

import com.twitter.timelineranker.parameters.revchron.ReverseChronTimelineQueryContextBuilder

import com.twitter.timelineranker.parameters.util.RequestContextBuilderImpl

import com.twitter.timelineranker.source.ReverseChronHomeTimelineSource

import com.twitter.timelineranker.visibility.RealGraphFollowGraphDataProvider

import com.twitter.timelineranker.visibility.SgsFollowGraphDataFields

import com.twitter.search.earlybird.thriftscala.EarlybirdService

import com.twitter.timelineranker.decider.DeciderKey

import com.twitter.timelines.util.stats.RequestScope

import com.twitter.util.Duration

class ReverseChronHomeTimelineRepositoryBuilder(

config: RuntimeConfiguration,

configBuilder: ConfigBuilder)

extends CandidatesRepositoryBuilder(config) {

override val clientSubId = "home\_materialization"

override val requestScope: RequestScope = RequestScopes.HomeTimelineMaterialization

override val followGraphDataFieldsToFetch: SgsFollowGraphDataFields.ValueSet =

SgsFollowGraphDataFields.ValueSet(

SgsFollowGraphDataFields.FollowedUserIds,

SgsFollowGraphDataFields.MutedUserIds,

SgsFollowGraphDataFields.RetweetsMutedUserIds

)

override val searchProcessingTimeout: Duration = 800.milliseconds // [3]

override def earlybirdClient(scope: String): EarlybirdService.MethodPerEndpoint =

config.underlyingClients.createEarlybirdClient(

scope = scope,

requestTimeout = 1.second, // [1]

timeout = 1900.milliseconds, // [2]

retryPolicy = config.underlyingClients.DefaultRetryPolicy

)

val realGraphFollowGraphDataProvider = new RealGraphFollowGraphDataProvider(

followGraphDataProvider,

config.clientWrapperFactories.realGraphClientFactory

.scope(RequestScopes.ReverseChronHomeTimelineSource),

config.clientWrapperFactories.socialGraphClientFactory

.scope(RequestScopes.ReverseChronHomeTimelineSource),

config.deciderGateBuilder.idGate(DeciderKey.SupplementFollowsWithRealGraph),

config.statsReceiver.scope(RequestScopes.ReverseChronHomeTimelineSource.scope)

)

def apply(): ReverseChronHomeTimelineRepository = {

val reverseChronTimelineSource = new ReverseChronHomeTimelineSource(

searchClient,

realGraphFollowGraphDataProvider,

clientFactories.visibilityEnforcerFactory.apply(

VisibilityRules,

RequestScopes.ReverseChronHomeTimelineSource

),

config.statsReceiver

)

val contextBuilder = new ReverseChronTimelineQueryContextBuilder(

configBuilder.rootConfig,

config,

new RequestContextBuilderImpl(config.configApiConfiguration.requestContextFactory)

)

new ReverseChronHomeTimelineRepository(

reverseChronTimelineSource,

contextBuilder

)

}

}