package com.twitter.follow\_recommendations.common.clients.strato

import com.google.inject.name.Named

import com.google.inject.Provides

import com.google.inject.Singleton

import com.twitter.conversions.DurationOps.\_

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

import com.twitter.search.account\_search.extended\_network.thriftscala.ExtendedNetworkUserKey

import com.twitter.search.account\_search.extended\_network.thriftscala.ExtendedNetworkUserVal

import com.twitter.finagle.ThriftMux

import com.twitter.finagle.mtls.authentication.ServiceIdentifier

import com.twitter.finagle.thrift.Protocols

import com.twitter.follow\_recommendations.common.constants.GuiceNamedConstants

import com.twitter.follow\_recommendations.common.constants.ServiceConstants.\_

import com.twitter.frigate.data\_pipeline.candidate\_generation.thriftscala.LatestEvents

import com.twitter.hermit.candidate.thriftscala.{Candidates => HermitCandidates}

import com.twitter.hermit.pop\_geo.thriftscala.PopUsersInPlace

import com.twitter.onboarding.relevance.relatable\_accounts.thriftscala.RelatableAccounts

import com.twitter.inject.TwitterModule

import com.twitter.onboarding.relevance.candidates.thriftscala.InterestBasedUserRecommendations

import com.twitter.onboarding.relevance.candidates.thriftscala.UTTInterest

import com.twitter.onboarding.relevance.store.thriftscala.WhoToFollowDismissEventDetails

import com.twitter.recos.user\_user\_graph.thriftscala.RecommendUserRequest

import com.twitter.recos.user\_user\_graph.thriftscala.RecommendUserResponse

import com.twitter.service.metastore.gen.thriftscala.UserRecommendabilityFeatures

import com.twitter.strato.catalog.Scan.Slice

import com.twitter.strato.client.Strato.{Client => StratoClient}

import com.twitter.strato.client.Client

import com.twitter.strato.client.Fetcher

import com.twitter.strato.client.Scanner

import com.twitter.strato.thrift.ScroogeConvImplicits.\_

import com.twitter.wtf.candidate.thriftscala.CandidateSeq

import com.twitter.wtf.ml.thriftscala.CandidateFeatures

import com.twitter.wtf.real\_time\_interaction\_graph.thriftscala.Interaction

import com.twitter.wtf.triangular\_loop.thriftscala.{Candidates => TriangularLoopCandidates}

import com.twitter.strato.opcontext.Attribution.\_

object StratoClientModule extends TwitterModule {

// column paths

val CosineFollowPath = "recommendations/similarity/similarUsersByFollowGraph.User"

val CosineListPath = "recommendations/similarity/similarUsersByListGraph.User"

val CuratedCandidatesPath = "onboarding/curatedAccounts"

val CuratedFilteredAccountsPath = "onboarding/filteredAccountsFromRecommendations"

val PopUsersInPlacePath = "onboarding/userrecs/popUsersInPlace"

val ProfileSidebarBlacklistPath = "recommendations/hermit/profile-sidebar-blacklist"

val RealTimeInteractionsPath = "hmli/realTimeInteractions"

val SimsPath = "recommendations/similarity/similarUsersBySims.User"

val DBV2SimsPath = "onboarding/userrecs/newSims.User"

val TriangularLoopsPath = "onboarding/userrecs/triangularLoops.User"

val TwoHopRandomWalkPath = "onboarding/userrecs/twoHopRandomWalk.User"

val UserRecommendabilityPath = "onboarding/userRecommendabilityWithLongKeys.User"

val UTTAccountRecommendationsPath = "onboarding/userrecs/utt\_account\_recommendations"

val UttSeedAccountsRecommendationPath = "onboarding/userrecs/utt\_seed\_accounts"

val UserStatePath = "onboarding/userState.User"

val WTFPostNuxFeaturesPath = "ml/featureStore/onboarding/wtfPostNuxFeatures.User"

val ElectionCandidatesPath = "onboarding/electionAccounts"

val UserUserGraphPath = "recommendations/userUserGraph"

val WtfDissmissEventsPath = "onboarding/wtfDismissEvents"

val RelatableAccountsPath = "onboarding/userrecs/relatableAccounts"

val ExtendedNetworkCandidatesPath = "search/account\_search/extendedNetworkCandidatesMH"

val LabeledNotificationPath = "frigate/magicrecs/labeledPushRecsAggregated.User"

@Provides

@Singleton

def stratoClient(serviceIdentifier: ServiceIdentifier): Client = {

val timeoutBudget = 500.milliseconds

StratoClient(

ThriftMux.client

.withRequestTimeout(timeoutBudget)

.withProtocolFactory(Protocols.binaryFactory(

stringLengthLimit = StringLengthLimit,

containerLengthLimit = ContainerLengthLimit)))

.withMutualTls(serviceIdentifier)

.build()

}

// add strato putters, fetchers, scanners below:

@Provides

@Singleton

@Named(GuiceNamedConstants.COSINE\_FOLLOW\_FETCHER)

def cosineFollowFetcher(stratoClient: Client): Fetcher[Long, Unit, HermitCandidates] =

stratoClient.fetcher[Long, Unit, HermitCandidates](CosineFollowPath)

@Provides

@Singleton

@Named(GuiceNamedConstants.COSINE\_LIST\_FETCHER)

def cosineListFetcher(stratoClient: Client): Fetcher[Long, Unit, HermitCandidates] =

stratoClient.fetcher[Long, Unit, HermitCandidates](CosineListPath)

@Provides

@Singleton

@Named(GuiceNamedConstants.CURATED\_COMPETITOR\_ACCOUNTS\_FETCHER)

def curatedBlacklistedAccountsFetcher(stratoClient: Client): Fetcher[String, Unit, Seq[Long]] =

stratoClient.fetcher[String, Unit, Seq[Long]](CuratedFilteredAccountsPath)

@Provides

@Singleton

@Named(GuiceNamedConstants.CURATED\_CANDIDATES\_FETCHER)

def curatedCandidatesFetcher(stratoClient: Client): Fetcher[String, Unit, Seq[Long]] =

stratoClient.fetcher[String, Unit, Seq[Long]](CuratedCandidatesPath)

@Provides

@Singleton

@Named(GuiceNamedConstants.POP\_USERS\_IN\_PLACE\_FETCHER)

def popUsersInPlaceFetcher(stratoClient: Client): Fetcher[String, Unit, PopUsersInPlace] =

stratoClient.fetcher[String, Unit, PopUsersInPlace](PopUsersInPlacePath)

@Provides

@Singleton

@Named(GuiceNamedConstants.RELATABLE\_ACCOUNTS\_FETCHER)

def relatableAccountsFetcher(stratoClient: Client): Fetcher[String, Unit, RelatableAccounts] =

stratoClient.fetcher[String, Unit, RelatableAccounts](RelatableAccountsPath)

@Provides

@Singleton

@Named(GuiceNamedConstants.PROFILE\_SIDEBAR\_BLACKLIST\_SCANNER)

def profileSidebarBlacklistScanner(

stratoClient: Client

): Scanner[(Long, Slice[Long]), Unit, (Long, Long), Unit] =

stratoClient.scanner[(Long, Slice[Long]), Unit, (Long, Long), Unit](ProfileSidebarBlacklistPath)

@Provides

@Singleton

@Named(GuiceNamedConstants.REAL\_TIME\_INTERACTIONS\_FETCHER)

def realTimeInteractionsFetcher(

stratoClient: Client

): Fetcher[(Long, Long), Unit, Seq[Interaction]] =

stratoClient.fetcher[(Long, Long), Unit, Seq[Interaction]](RealTimeInteractionsPath)

@Provides

@Singleton

@Named(GuiceNamedConstants.SIMS\_FETCHER)

def simsFetcher(stratoClient: Client): Fetcher[Long, Unit, HermitCandidates] =

stratoClient.fetcher[Long, Unit, HermitCandidates](SimsPath)

@Provides

@Singleton

@Named(GuiceNamedConstants.DBV2\_SIMS\_FETCHER)

def dbv2SimsFetcher(stratoClient: Client): Fetcher[Long, Unit, HermitCandidates] =

stratoClient.fetcher[Long, Unit, HermitCandidates](DBV2SimsPath)

@Provides

@Singleton

@Named(GuiceNamedConstants.TRIANGULAR\_LOOPS\_FETCHER)

def triangularLoopsFetcher(stratoClient: Client): Fetcher[Long, Unit, TriangularLoopCandidates] =

stratoClient.fetcher[Long, Unit, TriangularLoopCandidates](TriangularLoopsPath)

@Provides

@Singleton

@Named(GuiceNamedConstants.TWO\_HOP\_RANDOM\_WALK\_FETCHER)

def twoHopRandomWalkFetcher(stratoClient: Client): Fetcher[Long, Unit, CandidateSeq] =

stratoClient.fetcher[Long, Unit, CandidateSeq](TwoHopRandomWalkPath)

@Provides

@Singleton

@Named(GuiceNamedConstants.USER\_RECOMMENDABILITY\_FETCHER)

def userRecommendabilityFetcher(

stratoClient: Client

): Fetcher[Long, Unit, UserRecommendabilityFeatures] =

stratoClient.fetcher[Long, Unit, UserRecommendabilityFeatures](UserRecommendabilityPath)

@Provides

@Singleton

@Named(GuiceNamedConstants.USER\_STATE\_FETCHER)

def userStateFetcher(stratoClient: Client): Fetcher[Long, Unit, CondensedUserState] =

stratoClient.fetcher[Long, Unit, CondensedUserState](UserStatePath)

@Provides

@Singleton

@Named(GuiceNamedConstants.UTT\_ACCOUNT\_RECOMMENDATIONS\_FETCHER)

def uttAccountRecommendationsFetcher(

stratoClient: Client

): Fetcher[UTTInterest, Unit, InterestBasedUserRecommendations] =

stratoClient.fetcher[UTTInterest, Unit, InterestBasedUserRecommendations](

UTTAccountRecommendationsPath)

@Provides

@Singleton

@Named(GuiceNamedConstants.UTT\_SEED\_ACCOUNTS\_FETCHER)

def uttSeedAccountRecommendationsFetcher(

stratoClient: Client

): Fetcher[UTTInterest, Unit, InterestBasedUserRecommendations] =

stratoClient.fetcher[UTTInterest, Unit, InterestBasedUserRecommendations](

UttSeedAccountsRecommendationPath)

@Provides

@Singleton

@Named(GuiceNamedConstants.ELECTION\_CANDIDATES\_FETCHER)

def electionCandidatesFetcher(stratoClient: Client): Fetcher[String, Unit, Seq[Long]] =

stratoClient.fetcher[String, Unit, Seq[Long]](ElectionCandidatesPath)

@Provides

@Singleton

@Named(GuiceNamedConstants.USER\_USER\_GRAPH\_FETCHER)

def userUserGraphFetcher(

stratoClient: Client

): Fetcher[RecommendUserRequest, Unit, RecommendUserResponse] =

stratoClient.fetcher[RecommendUserRequest, Unit, RecommendUserResponse](UserUserGraphPath)

@Provides

@Singleton

@Named(GuiceNamedConstants.POST\_NUX\_WTF\_FEATURES\_FETCHER)

def wtfPostNuxFeaturesFetcher(stratoClient: Client): Fetcher[Long, Unit, CandidateFeatures] = {

val attribution = ManhattanAppId("starbuck", "wtf\_starbuck")

stratoClient

.fetcher[Long, Unit, CandidateFeatures](WTFPostNuxFeaturesPath)

.withAttribution(attribution)

}

@Provides

@Singleton

@Named(GuiceNamedConstants.EXTENDED\_NETWORK)

def extendedNetworkFetcher(

stratoClient: Client

): Fetcher[ExtendedNetworkUserKey, Unit, ExtendedNetworkUserVal] = {

stratoClient

.fetcher[ExtendedNetworkUserKey, Unit, ExtendedNetworkUserVal](ExtendedNetworkCandidatesPath)

}

@Provides

@Singleton

@Named(GuiceNamedConstants.DISMISS\_STORE\_SCANNER)

def dismissStoreScanner(

stratoClient: Client

): Scanner[

(Long, Slice[(Long, Long)]),

Unit,

(Long, (Long, Long)),

WhoToFollowDismissEventDetails

] =

stratoClient.scanner[

(Long, Slice[(Long, Long)]), // PKEY: userId, LKEY: (-ts, candidateId)

Unit,

(Long, (Long, Long)),

WhoToFollowDismissEventDetails

](WtfDissmissEventsPath)

@Provides

@Singleton

@Named(GuiceNamedConstants.LABELED\_NOTIFICATION\_FETCHER)

def labeledNotificationFetcher(

stratoClient: Client

): Fetcher[Long, Unit, LatestEvents] = {

stratoClient

.fetcher[Long, Unit, LatestEvents](LabeledNotificationPath)

}

}