package com.twitter.usersignalservice.config

import com.twitter.conversions.DurationOps.\_

import com.twitter.finagle.memcached.{Client => MemcachedClient}

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.storehaus.ReadableStore

import com.twitter.usersignalservice.base.BaseSignalFetcher

import com.twitter.usersignalservice.base.AggregatedSignalController

import com.twitter.usersignalservice.base.FilteredSignalFetcherController

import com.twitter.usersignalservice.base.MemcachedSignalFetcherWrapper

import com.twitter.usersignalservice.base.Query

import com.twitter.usersignalservice.base.SignalAggregatedInfo

import com.twitter.usersignalservice.signals.AccountBlocksFetcher

import com.twitter.usersignalservice.signals.AccountFollowsFetcher

import com.twitter.usersignalservice.signals.AccountMutesFetcher

import com.twitter.usersignalservice.signals.NotificationOpenAndClickFetcher

import com.twitter.usersignalservice.signals.OriginalTweetsFetcher

import com.twitter.usersignalservice.signals.ProfileVisitsFetcher

import com.twitter.usersignalservice.signals.ProfileClickFetcher

import com.twitter.usersignalservice.signals.RealGraphOonFetcher

import com.twitter.usersignalservice.signals.ReplyTweetsFetcher

import com.twitter.usersignalservice.signals.RetweetsFetcher

import com.twitter.usersignalservice.signals.TweetClickFetcher

import com.twitter.usersignalservice.signals.TweetFavoritesFetcher

import com.twitter.usersignalservice.signals.TweetSharesFetcher

import com.twitter.usersignalservice.signals.VideoTweetsPlayback50Fetcher

import com.twitter.usersignalservice.signals.VideoTweetsQualityViewFetcher

import com.twitter.usersignalservice.signals.NegativeEngagedUserFetcher

import com.twitter.usersignalservice.signals.NegativeEngagedTweetFetcher

import com.twitter.usersignalservice.thriftscala.Signal

import com.twitter.usersignalservice.thriftscala.SignalType

import com.twitter.util.Timer

import javax.inject.Inject

import javax.inject.Singleton

@Singleton

class SignalFetcherConfig @Inject() (

notificationOpenAndClickFetcher: NotificationOpenAndClickFetcher,

accountFollowsFetcher: AccountFollowsFetcher,

profileVisitsFetcher: ProfileVisitsFetcher,

tweetFavoritesFetcher: TweetFavoritesFetcher,

retweetsFetcher: RetweetsFetcher,

replyTweetsFetcher: ReplyTweetsFetcher,

originalTweetsFetcher: OriginalTweetsFetcher,

tweetSharesFetcher: TweetSharesFetcher,

memcachedClient: MemcachedClient,

realGraphOonFetcher: RealGraphOonFetcher,

tweetClickFetcher: TweetClickFetcher,

videoTweetsPlayback50Fetcher: VideoTweetsPlayback50Fetcher,

videoTweetsQualityViewFetcher: VideoTweetsQualityViewFetcher,

accountMutesFetcher: AccountMutesFetcher,

accountBlocksFetcher: AccountBlocksFetcher,

profileClickFetcher: ProfileClickFetcher,

negativeEngagedTweetFetcher: NegativeEngagedTweetFetcher,

negativeEngagedUserFetcher: NegativeEngagedUserFetcher,

statsReceiver: StatsReceiver,

timer: Timer) {

val MemcachedProfileVisitsFetcher: BaseSignalFetcher =

MemcachedSignalFetcherWrapper(

memcachedClient,

profileVisitsFetcher,

ttl = 8.hours,

statsReceiver,

keyPrefix = "uss:pv",

timer)

val MemcachedAccountFollowsFetcher: BaseSignalFetcher = MemcachedSignalFetcherWrapper(

memcachedClient,

accountFollowsFetcher,

ttl = 5.minute,

statsReceiver,

keyPrefix = "uss:af",

timer)

val GoodTweetClickDdgFetcher: SignalType => FilteredSignalFetcherController = signalType =>

FilteredSignalFetcherController(

tweetClickFetcher,

signalType,

statsReceiver,

timer,

Map(SignalType.NegativeEngagedTweetId -> negativeEngagedTweetFetcher)

)

val GoodProfileClickDdgFetcher: SignalType => FilteredSignalFetcherController = signalType =>

FilteredSignalFetcherController(

profileClickFetcher,

signalType,

statsReceiver,

timer,

Map(SignalType.NegativeEngagedUserId -> negativeEngagedUserFetcher)

)

val GoodProfileClickDdgFetcherWithBlocksMutes: SignalType => FilteredSignalFetcherController =

signalType =>

FilteredSignalFetcherController(

profileClickFetcher,

signalType,

statsReceiver,

timer,

Map(

SignalType.NegativeEngagedUserId -> negativeEngagedUserFetcher,

SignalType.AccountMute -> accountMutesFetcher,

SignalType.AccountBlock -> accountBlocksFetcher

)

)

val realGraphOonFilteredFetcher: FilteredSignalFetcherController =

FilteredSignalFetcherController(

realGraphOonFetcher,

SignalType.RealGraphOon,

statsReceiver,

timer,

Map(

SignalType.NegativeEngagedUserId -> negativeEngagedUserFetcher

)

)

val videoTweetsQualityViewFilteredFetcher: FilteredSignalFetcherController =

FilteredSignalFetcherController(

videoTweetsQualityViewFetcher,

SignalType.VideoView90dQualityV1,

statsReceiver,

timer,

Map(SignalType.NegativeEngagedTweetId -> negativeEngagedTweetFetcher)

)

val videoTweetsPlayback50FilteredFetcher: FilteredSignalFetcherController =

FilteredSignalFetcherController(

videoTweetsPlayback50Fetcher,

SignalType.VideoView90dPlayback50V1,

statsReceiver,

timer,

Map(SignalType.NegativeEngagedTweetId -> negativeEngagedTweetFetcher)

)

val uniformTweetSignalInfo: Seq[SignalAggregatedInfo] = Seq(

SignalAggregatedInfo(SignalType.TweetFavorite, tweetFavoritesFetcher),

SignalAggregatedInfo(SignalType.Retweet, retweetsFetcher),

SignalAggregatedInfo(SignalType.Reply, replyTweetsFetcher),

SignalAggregatedInfo(SignalType.OriginalTweet, originalTweetsFetcher),

SignalAggregatedInfo(SignalType.TweetShareV1, tweetSharesFetcher),

SignalAggregatedInfo(SignalType.VideoView90dQualityV1, videoTweetsQualityViewFilteredFetcher),

)

val uniformProducerSignalInfo: Seq[SignalAggregatedInfo] = Seq(

SignalAggregatedInfo(SignalType.AccountFollow, MemcachedAccountFollowsFetcher),

SignalAggregatedInfo(

SignalType.RepeatedProfileVisit90dMinVisit6V1,

MemcachedProfileVisitsFetcher),

)

val memcachedAccountBlocksFetcher: MemcachedSignalFetcherWrapper = MemcachedSignalFetcherWrapper(

memcachedClient,

accountBlocksFetcher,

ttl = 5.minutes,

statsReceiver,

keyPrefix = "uss:ab",

timer)

val memcachedAccountMutesFetcher: MemcachedSignalFetcherWrapper = MemcachedSignalFetcherWrapper(

memcachedClient,

accountMutesFetcher,

ttl = 5.minutes,

statsReceiver,

keyPrefix = "uss:am",

timer)

val SignalFetcherMapper: Map[SignalType, ReadableStore[Query, Seq[Signal]]] = Map(

/\* Raw Signals \*/

SignalType.AccountFollow -> accountFollowsFetcher,

SignalType.AccountFollowWithDelay -> MemcachedAccountFollowsFetcher,

SignalType.GoodProfileClick -> GoodProfileClickDdgFetcher(SignalType.GoodProfileClick),

SignalType.GoodProfileClick20s -> GoodProfileClickDdgFetcher(SignalType.GoodProfileClick20s),

SignalType.GoodProfileClick30s -> GoodProfileClickDdgFetcher(SignalType.GoodProfileClick30s),

SignalType.GoodProfileClickFiltered -> GoodProfileClickDdgFetcherWithBlocksMutes(

SignalType.GoodProfileClick),

SignalType.GoodProfileClick20sFiltered -> GoodProfileClickDdgFetcherWithBlocksMutes(

SignalType.GoodProfileClick20s),

SignalType.GoodProfileClick30sFiltered -> GoodProfileClickDdgFetcherWithBlocksMutes(

SignalType.GoodProfileClick30s),

SignalType.GoodTweetClick -> GoodTweetClickDdgFetcher(SignalType.GoodTweetClick),

SignalType.GoodTweetClick5s -> GoodTweetClickDdgFetcher(SignalType.GoodTweetClick5s),

SignalType.GoodTweetClick10s -> GoodTweetClickDdgFetcher(SignalType.GoodTweetClick10s),

SignalType.GoodTweetClick30s -> GoodTweetClickDdgFetcher(SignalType.GoodTweetClick30s),

SignalType.NegativeEngagedTweetId -> negativeEngagedTweetFetcher,

SignalType.NegativeEngagedUserId -> negativeEngagedUserFetcher,

SignalType.NotificationOpenAndClickV1 -> notificationOpenAndClickFetcher,

SignalType.OriginalTweet -> originalTweetsFetcher,

SignalType.OriginalTweet90dV2 -> originalTweetsFetcher,

SignalType.RealGraphOon -> realGraphOonFilteredFetcher,

SignalType.RepeatedProfileVisit14dMinVisit2V1 -> MemcachedProfileVisitsFetcher,

SignalType.RepeatedProfileVisit14dMinVisit2V1NoNegative -> FilteredSignalFetcherController(

MemcachedProfileVisitsFetcher,

SignalType.RepeatedProfileVisit14dMinVisit2V1NoNegative,

statsReceiver,

timer,

Map(

SignalType.AccountMute -> accountMutesFetcher,

SignalType.AccountBlock -> accountBlocksFetcher)

),

SignalType.RepeatedProfileVisit90dMinVisit6V1 -> MemcachedProfileVisitsFetcher,

SignalType.RepeatedProfileVisit90dMinVisit6V1NoNegative -> FilteredSignalFetcherController(

MemcachedProfileVisitsFetcher,

SignalType.RepeatedProfileVisit90dMinVisit6V1NoNegative,

statsReceiver,

timer,

Map(

SignalType.AccountMute -> accountMutesFetcher,

SignalType.AccountBlock -> accountBlocksFetcher),

),

SignalType.RepeatedProfileVisit180dMinVisit6V1 -> MemcachedProfileVisitsFetcher,

SignalType.RepeatedProfileVisit180dMinVisit6V1NoNegative -> FilteredSignalFetcherController(

MemcachedProfileVisitsFetcher,

SignalType.RepeatedProfileVisit180dMinVisit6V1NoNegative,

statsReceiver,

timer,

Map(

SignalType.AccountMute -> accountMutesFetcher,

SignalType.AccountBlock -> accountBlocksFetcher),

),

SignalType.Reply -> replyTweetsFetcher,

SignalType.Reply90dV2 -> replyTweetsFetcher,

SignalType.Retweet -> retweetsFetcher,

SignalType.Retweet90dV2 -> retweetsFetcher,

SignalType.TweetFavorite -> tweetFavoritesFetcher,

SignalType.TweetFavorite90dV2 -> tweetFavoritesFetcher,

SignalType.TweetShareV1 -> tweetSharesFetcher,

SignalType.VideoView90dQualityV1 -> videoTweetsQualityViewFilteredFetcher,

SignalType.VideoView90dPlayback50V1 -> videoTweetsPlayback50FilteredFetcher,

/\* Aggregated Signals \*/

SignalType.ProducerBasedUnifiedEngagementWeightedSignal -> AggregatedSignalController(

uniformProducerSignalInfo,

uniformProducerSignalEngagementAggregation,

statsReceiver,

timer

),

SignalType.TweetBasedUnifiedEngagementWeightedSignal -> AggregatedSignalController(

uniformTweetSignalInfo,

uniformTweetSignalEngagementAggregation,

statsReceiver,

timer

),

SignalType.AdFavorite -> tweetFavoritesFetcher,

/\* Negative Signals \*/

SignalType.AccountBlock -> memcachedAccountBlocksFetcher,

SignalType.AccountMute -> memcachedAccountMutesFetcher,

SignalType.TweetDontLike -> negativeEngagedTweetFetcher,

SignalType.TweetReport -> negativeEngagedTweetFetcher,

SignalType.TweetSeeFewer -> negativeEngagedTweetFetcher,

)

}