package com.twitter.frigate.pushservice.config

import com.twitter.abdecider.LoggingABDecider

import com.twitter.abuse.detection.scoring.thriftscala.TweetScoringRequest

import com.twitter.abuse.detection.scoring.thriftscala.TweetScoringResponse

import com.twitter.audience\_rewards.thriftscala.HasSuperFollowingRelationshipRequest

import com.twitter.channels.common.thriftscala.ApiList

import com.twitter.datatools.entityservice.entities.sports.thriftscala.\_

import com.twitter.decider.Decider

import com.twitter.discovery.common.configapi.ConfigParamsBuilder

import com.twitter.escherbird.common.thriftscala.QualifiedId

import com.twitter.escherbird.metadata.thriftscala.EntityMegadata

import com.twitter.eventbus.client.EventBusPublisher

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.finagle.thrift.ClientId

import com.twitter.frigate.common.base.\_

import com.twitter.frigate.common.candidate.\_

import com.twitter.frigate.common.history.\_

import com.twitter.frigate.common.ml.base.\_

import com.twitter.frigate.common.ml.feature.\_

import com.twitter.frigate.common.store.\_

import com.twitter.frigate.common.store.deviceinfo.DeviceInfo

import com.twitter.frigate.common.store.interests.InterestsLookupRequestWithContext

import com.twitter.frigate.common.store.interests.UserId

import com.twitter.frigate.common.util.\_

import com.twitter.frigate.data\_pipeline.features\_common.\_

import com.twitter.frigate.data\_pipeline.thriftscala.UserHistoryKey

import com.twitter.frigate.data\_pipeline.thriftscala.UserHistoryValue

import com.twitter.frigate.dau\_model.thriftscala.DauProbability

import com.twitter.frigate.magic\_events.thriftscala.FanoutEvent

import com.twitter.frigate.pushcap.thriftscala.PushcapUserHistory

import com.twitter.frigate.pushservice.ml.\_

import com.twitter.frigate.pushservice.params.DeciderKey

import com.twitter.frigate.pushservice.params.PushFeatureSwitchParams

import com.twitter.frigate.pushservice.params.PushFeatureSwitches

import com.twitter.frigate.pushservice.params.PushParams

import com.twitter.frigate.pushservice.send\_handler.SendHandlerPushCandidateHydrator

import com.twitter.frigate.pushservice.refresh\_handler.PushCandidateHydrator

import com.twitter.frigate.pushservice.store.\_

import com.twitter.frigate.pushservice.store.{Ibis2Store => PushIbis2Store}

import com.twitter.frigate.pushservice.take.NotificationServiceRequest

import com.twitter.frigate.pushservice.thriftscala.PushRequestScribe

import com.twitter.frigate.scribe.thriftscala.NotificationScribe

import com.twitter.frigate.thriftscala.\_

import com.twitter.frigate.user\_states.thriftscala.MRUserHmmState

import com.twitter.geoduck.common.thriftscala.{Location => GeoLocation}

import com.twitter.geoduck.service.thriftscala.LocationResponse

import com.twitter.gizmoduck.thriftscala.User

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

import com.twitter.hermit.predicate.socialgraph.RelationEdge

import com.twitter.hermit.predicate.tweetypie.Perspective

import com.twitter.hermit.predicate.tweetypie.UserTweet

import com.twitter.hermit.store.semantic\_core.SemanticEntityForQuery

import com.twitter.hermit.store.tweetypie.{UserTweet => TweetyPieUserTweet}

import com.twitter.hermit.stp.thriftscala.STPResult

import com.twitter.hss.api.thriftscala.UserHealthSignalResponse

import com.twitter.interests.thriftscala.InterestId

import com.twitter.interests.thriftscala.{UserInterests => Interests}

import com.twitter.interests\_discovery.thriftscala.NonPersonalizedRecommendedLists

import com.twitter.interests\_discovery.thriftscala.RecommendedListsRequest

import com.twitter.interests\_discovery.thriftscala.RecommendedListsResponse

import com.twitter.livevideo.timeline.domain.v2.{Event => LiveEvent}

import com.twitter.ml.api.thriftscala.{DataRecord => ThriftDataRecord}

import com.twitter.ml.featurestore.lib.dynamic.DynamicFeatureStoreClient

import com.twitter.notificationservice.genericfeedbackstore.FeedbackPromptValue

import com.twitter.notificationservice.genericfeedbackstore.GenericFeedbackStore

import com.twitter.notificationservice.scribe.manhattan.GenericNotificationsFeedbackRequest

import com.twitter.notificationservice.thriftscala.CaretFeedbackDetails

import com.twitter.notificationservice.thriftscala.CreateGenericNotificationResponse

import com.twitter.nrel.heavyranker.CandidateFeatureHydrator

import com.twitter.nrel.heavyranker.{FeatureHydrator => MRFeatureHydrator}

import com.twitter.nrel.heavyranker.{TargetFeatureHydrator => RelevanceTargetFeatureHydrator}

import com.twitter.onboarding.task.service.thriftscala.FatigueFlowEnrollment

import com.twitter.permissions\_storage.thriftscala.AppPermission

import com.twitter.recommendation.interests.discovery.core.model.InterestDomain

import com.twitter.recos.user\_tweet\_entity\_graph.thriftscala.RecommendTweetEntityRequest

import com.twitter.recos.user\_tweet\_entity\_graph.thriftscala.RecommendTweetEntityResponse

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

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

import com.twitter.rux.common.strato.thriftscala.UserTargetingProperty

import com.twitter.scio.nsfw\_user\_segmentation.thriftscala.NSFWProducer

import com.twitter.scio.nsfw\_user\_segmentation.thriftscala.NSFWUserSegmentation

import com.twitter.search.common.features.thriftscala.ThriftSearchResultFeatures

import com.twitter.search.earlybird.thriftscala.EarlybirdRequest

import com.twitter.search.earlybird.thriftscala.ThriftSearchResult

import com.twitter.service.gen.scarecrow.thriftscala.Event

import com.twitter.service.gen.scarecrow.thriftscala.TieredActionResult

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

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

import com.twitter.servo.decider.DeciderGateBuilder

import com.twitter.simclusters\_v2.thriftscala.SimClustersInferredEntities

import com.twitter.stitch.tweetypie.TweetyPie.TweetyPieResult

import com.twitter.storehaus.ReadableStore

import com.twitter.strato.columns.frigate.logged\_out\_web\_notifications.thriftscala.LOWebNotificationMetadata

import com.twitter.strato.columns.notifications.thriftscala.SourceDestUserRequest

import com.twitter.strato.client.{UserId => StratoUserId}

import com.twitter.timelines.configapi

import com.twitter.timelines.configapi.CompositeConfig

import com.twitter.timelinescorer.thriftscala.v1.ScoredTweet

import com.twitter.topiclisting.TopicListing

import com.twitter.trends.trip\_v1.trip\_tweets.thriftscala.TripDomain

import com.twitter.trends.trip\_v1.trip\_tweets.thriftscala.TripTweets

import com.twitter.tsp.thriftscala.TopicSocialProofRequest

import com.twitter.tsp.thriftscala.TopicSocialProofResponse

import com.twitter.ubs.thriftscala.SellerTrack

import com.twitter.ubs.thriftscala.AudioSpace

import com.twitter.ubs.thriftscala.Participants

import com.twitter.ubs.thriftscala.SellerApplicationState

import com.twitter.user\_session\_store.thriftscala.UserSession

import com.twitter.util.Duration

import com.twitter.util.Future

import com.twitter.wtf.scalding.common.thriftscala.UserFeatures

trait Config {

self =>

def isServiceLocal: Boolean

def localConfigRepoPath: String

def inMemCacheOff: Boolean

def historyStore: PushServiceHistoryStore

def emailHistoryStore: PushServiceHistoryStore

def strongTiesStore: ReadableStore[Long, STPResult]

def safeUserStore: ReadableStore[Long, User]

def deviceInfoStore: ReadableStore[Long, DeviceInfo]

def edgeStore: ReadableStore[RelationEdge, Boolean]

def socialGraphServiceProcessStore: ReadableStore[RelationEdge, Boolean]

def userUtcOffsetStore: ReadableStore[Long, Duration]

def cachedTweetyPieStoreV2: ReadableStore[Long, TweetyPieResult]

def safeCachedTweetyPieStoreV2: ReadableStore[Long, TweetyPieResult]

def userTweetTweetyPieStore: ReadableStore[TweetyPieUserTweet, TweetyPieResult]

def safeUserTweetTweetyPieStore: ReadableStore[TweetyPieUserTweet, TweetyPieResult]

def cachedTweetyPieStoreV2NoVF: ReadableStore[Long, TweetyPieResult]

def tweetContentFeatureCacheStore: ReadableStore[Long, ThriftDataRecord]

def scarecrowCheckEventStore: ReadableStore[Event, TieredActionResult]

def userTweetPerspectiveStore: ReadableStore[UserTweet, Perspective]

def userCountryStore: ReadableStore[Long, Location]

def pushInfoStore: ReadableStore[Long, UserForPushTargeting]

def loggedOutPushInfoStore: ReadableStore[Long, LOWebNotificationMetadata]

def tweetImpressionStore: ReadableStore[Long, Seq[Long]]

def audioSpaceStore: ReadableStore[String, AudioSpace]

def basketballGameScoreStore: ReadableStore[QualifiedId, BasketballGameLiveUpdate]

def baseballGameScoreStore: ReadableStore[QualifiedId, BaseballGameLiveUpdate]

def cricketMatchScoreStore: ReadableStore[QualifiedId, CricketMatchLiveUpdate]

def soccerMatchScoreStore: ReadableStore[QualifiedId, SoccerMatchLiveUpdate]

def nflGameScoreStore: ReadableStore[QualifiedId, NflFootballGameLiveUpdate]

def topicSocialProofServiceStore: ReadableStore[TopicSocialProofRequest, TopicSocialProofResponse]

def spaceDeviceFollowStore: ReadableStore[SourceDestUserRequest, Boolean]

def audioSpaceParticipantsStore: ReadableStore[String, Participants]

def notificationServiceSender: ReadableStore[

NotificationServiceRequest,

CreateGenericNotificationResponse

]

def ocfFatigueStore: ReadableStore[OCFHistoryStoreKey, FatigueFlowEnrollment]

def dauProbabilityStore: ReadableStore[Long, DauProbability]

def hydratedLabeledPushRecsStore: ReadableStore[UserHistoryKey, UserHistoryValue]

def userHTLLastVisitStore: ReadableStore[Long, Seq[Long]]

def userLanguagesStore: ReadableStore[Long, UserLanguages]

def topTweetsByGeoStore: ReadableStore[InterestDomain[String], Map[String, List[

(Long, Double)

]]]

def topTweetsByGeoV2VersionedStore: ReadableStore[String, PopTweetsInPlace]

lazy val pushRecItemStore: ReadableStore[PushRecItemsKey, RecItems] = PushRecItemStore(

hydratedLabeledPushRecsStore

)

lazy val labeledPushRecsVerifyingStore: ReadableStore[

LabeledPushRecsVerifyingStoreKey,

LabeledPushRecsVerifyingStoreResponse

] =

LabeledPushRecsVerifyingStore(

hydratedLabeledPushRecsStore,

historyStore

)

lazy val labeledPushRecsDecideredStore: ReadableStore[LabeledPushRecsStoreKey, UserHistoryValue] =

LabeledPushRecsDecideredStore(

labeledPushRecsVerifyingStore,

useHydratedLabeledSendsForFeaturesDeciderKey,

verifyHydratedLabeledSendsForFeaturesDeciderKey

)

def onlineUserHistoryStore: ReadableStore[OnlineUserHistoryKey, UserHistoryValue]

def nsfwConsumerStore: ReadableStore[Long, NSFWUserSegmentation]

def nsfwProducerStore: ReadableStore[Long, NSFWProducer]

def popGeoLists: ReadableStore[String, NonPersonalizedRecommendedLists]

def listAPIStore: ReadableStore[Long, ApiList]

def openedPushByHourAggregatedStore: ReadableStore[Long, Map[Int, Int]]

def userHealthSignalStore: ReadableStore[Long, UserHealthSignalResponse]

def reactivatedUserInfoStore: ReadableStore[Long, String]

def weightedOpenOrNtabClickModelScorer: PushMLModelScorer

def optoutModelScorer: PushMLModelScorer

def filteringModelScorer: PushMLModelScorer

def recentFollowsStore: ReadableStore[Long, Seq[Long]]

def geoDuckV2Store: ReadableStore[UserId, LocationResponse]

def realGraphScoresTop500InStore: ReadableStore[Long, Map[Long, Double]]

def tweetEntityGraphStore: ReadableStore[

RecommendTweetEntityRequest,

RecommendTweetEntityResponse

]

def userUserGraphStore: ReadableStore[RecommendUserRequest, RecommendUserResponse]

def userFeaturesStore: ReadableStore[Long, UserFeatures]

def userTargetingPropertyStore: ReadableStore[Long, UserTargetingProperty]

def timelinesUserSessionStore: ReadableStore[Long, UserSession]

def optOutUserInterestsStore: ReadableStore[UserId, Seq[InterestId]]

def ntabCaretFeedbackStore: ReadableStore[GenericNotificationsFeedbackRequest, Seq[

CaretFeedbackDetails

]]

def genericFeedbackStore: ReadableStore[FeedbackRequest, Seq[

FeedbackPromptValue

]]

def genericNotificationFeedbackStore: GenericFeedbackStore

def semanticCoreMegadataStore: ReadableStore[

SemanticEntityForQuery,

EntityMegadata

]

def tweetHealthScoreStore: ReadableStore[TweetScoringRequest, TweetScoringResponse]

def earlybirdFeatureStore: ReadableStore[Long, ThriftSearchResultFeatures]

def earlybirdFeatureBuilder: FeatureBuilder[Long]

// Feature builders

def tweetAuthorLocationFeatureBuilder: FeatureBuilder[Location]

def tweetAuthorLocationFeatureBuilderById: FeatureBuilder[Long]

def socialContextActionsFeatureBuilder: FeatureBuilder[SocialContextActions]

def tweetContentFeatureBuilder: FeatureBuilder[Long]

def tweetAuthorRecentRealGraphFeatureBuilder: FeatureBuilder[RealGraphEdge]

def socialContextRecentRealGraphFeatureBuilder: FeatureBuilder[Set[RealGraphEdge]]

def tweetSocialProofFeatureBuilder: FeatureBuilder[TweetSocialProofKey]

def targetUserFullRealGraphFeatureBuilder: FeatureBuilder[TargetFullRealGraphFeatureKey]

def postProcessingFeatureBuilder: PostProcessingFeatureBuilder

def mrOfflineUserCandidateSparseAggregatesFeatureBuilder: FeatureBuilder[

OfflineSparseAggregateKey

]

def mrOfflineUserAggregatesFeatureBuilder: FeatureBuilder[Long]

def mrOfflineUserCandidateAggregatesFeatureBuilder: FeatureBuilder[OfflineAggregateKey]

def tweetAnnotationsFeatureBuilder: FeatureBuilder[Long]

def targetUserMediaRepresentationFeatureBuilder: FeatureBuilder[Long]

def targetLevelFeatureBuilder: FeatureBuilder[MrRequestContextForFeatureStore]

def candidateLevelFeatureBuilder: FeatureBuilder[EntityRequestContextForFeatureStore]

def targetFeatureHydrator: RelevanceTargetFeatureHydrator

def useHydratedLabeledSendsForFeaturesDeciderKey: String =

DeciderKey.useHydratedLabeledSendsForFeaturesDeciderKey.toString

def verifyHydratedLabeledSendsForFeaturesDeciderKey: String =

DeciderKey.verifyHydratedLabeledSendsForFeaturesDeciderKey.toString

def lexServiceStore: ReadableStore[EventRequest, LiveEvent]

def userMediaRepresentationStore: ReadableStore[Long, UserMediaRepresentation]

def producerMediaRepresentationStore: ReadableStore[Long, UserMediaRepresentation]

def mrUserStatePredictionStore: ReadableStore[Long, MRUserHmmState]

def pushcapDynamicPredictionStore: ReadableStore[Long, PushcapUserHistory]

def earlybirdCandidateSource: EarlybirdCandidateSource

def earlybirdSearchStore: ReadableStore[EarlybirdRequest, Seq[ThriftSearchResult]]

def earlybirdSearchDest: String

def pushserviceThriftClientId: ClientId

def simClusterToEntityStore: ReadableStore[Int, SimClustersInferredEntities]

def fanoutMetadataStore: ReadableStore[(Long, Long), FanoutEvent]

/\*\*

\* PostRanking Feature Store Client

\*/

def postRankingFeatureStoreClient: DynamicFeatureStoreClient[MrRequestContextForFeatureStore]

/\*\*

\* ReadableStore to fetch [[UserInterests]] from INTS service

\*/

def interestsWithLookupContextStore: ReadableStore[InterestsLookupRequestWithContext, Interests]

/\*\*

\*

\* @return: [[TopicListing]] object to fetch paused topics and scope from productId

\*/

def topicListing: TopicListing

/\*\*

\*

\* @return: [[UttEntityHydrationStore]] object

\*/

def uttEntityHydrationStore: UttEntityHydrationStore

def appPermissionStore: ReadableStore[(Long, (String, String)), AppPermission]

lazy val userTweetEntityGraphCandidates: UserTweetEntityGraphCandidates =

UserTweetEntityGraphCandidates(

cachedTweetyPieStoreV2,

tweetEntityGraphStore,

PushParams.UTEGTweetCandidateSourceParam,

PushFeatureSwitchParams.NumberOfMaxUTEGCandidatesQueriedParam,

PushParams.AllowOneSocialProofForTweetInUTEGParam,

PushParams.OutNetworkTweetsOnlyForUTEGParam,

PushFeatureSwitchParams.MaxTweetAgeParam

)(statsReceiver)

def pushSendEventBusPublisher: EventBusPublisher[NotificationScribe]

// miscs.

def isProd: Boolean

implicit def statsReceiver: StatsReceiver

def decider: Decider

def abDecider: LoggingABDecider

def casLock: CasLock

def pushIbisV2Store: PushIbis2Store

// scribe

def notificationScribe(data: NotificationScribe): Unit

def requestScribe(data: PushRequestScribe): Unit

def init(): Future[Unit] = Future.Done

def configParamsBuilder: ConfigParamsBuilder

def candidateFeatureHydrator: CandidateFeatureHydrator

def featureHydrator: MRFeatureHydrator

def candidateHydrator: PushCandidateHydrator

def sendHandlerCandidateHydrator: SendHandlerPushCandidateHydrator

lazy val overridesConfig: configapi.Config = {

val pushFeatureSwitchConfigs: configapi.Config = PushFeatureSwitches(

deciderGateBuilder = new DeciderGateBuilder(decider),

statsReceiver = statsReceiver

).config

new CompositeConfig(Seq(pushFeatureSwitchConfigs))

}

def realTimeClientEventStore: RealTimeClientEventStore

def inlineActionHistoryStore: ReadableStore[Long, Seq[(Long, String)]]

def softUserGeoLocationStore: ReadableStore[Long, GeoLocation]

def tweetTranslationStore: ReadableStore[TweetTranslationStore.Key, TweetTranslationStore.Value]

def tripTweetCandidateStore: ReadableStore[TripDomain, TripTweets]

def softUserFollowingStore: ReadableStore[User, Seq[Long]]

def superFollowEligibilityUserStore: ReadableStore[Long, Boolean]

def superFollowCreatorTweetCountStore: ReadableStore[StratoUserId, Int]

def hasSuperFollowingRelationshipStore: ReadableStore[

HasSuperFollowingRelationshipRequest,

Boolean

]

def superFollowApplicationStatusStore: ReadableStore[(Long, SellerTrack), SellerApplicationState]

def recentHistoryCacheClient: RecentHistoryCacheClient

def openAppUserStore: ReadableStore[Long, Boolean]

def loggedOutHistoryStore: PushServiceHistoryStore

def idsStore: ReadableStore[RecommendedListsRequest, RecommendedListsResponse]

def htlScoreStore(userId: Long): ReadableStore[Long, ScoredTweet]

}