package com.twitter.unified\_user\_actions.enricher

import com.twitter.unified\_user\_actions.enricher.internal.thriftscala.EnrichmentInstruction

import com.twitter.unified\_user\_actions.enricher.internal.thriftscala.EnrichmentPlan

import com.twitter.unified\_user\_actions.enricher.internal.thriftscala.EnrichmentStage

import com.twitter.unified\_user\_actions.enricher.internal.thriftscala.EnrichmentStageStatus

import com.twitter.unified\_user\_actions.enricher.internal.thriftscala.EnrichmentStageType

import com.twitter.unified\_user\_actions.thriftscala.ActionType

import com.twitter.unified\_user\_actions.thriftscala.AuthorInfo

import com.twitter.unified\_user\_actions.thriftscala.EventMetadata

import com.twitter.unified\_user\_actions.thriftscala.Item

import com.twitter.unified\_user\_actions.thriftscala.MultiTweetNotification

import com.twitter.unified\_user\_actions.thriftscala.NotificationContent

import com.twitter.unified\_user\_actions.thriftscala.NotificationInfo

import com.twitter.unified\_user\_actions.thriftscala.ProfileInfo

import com.twitter.unified\_user\_actions.thriftscala.SourceLineage

import com.twitter.unified\_user\_actions.thriftscala.TweetInfo

import com.twitter.unified\_user\_actions.thriftscala.TweetNotification

import com.twitter.unified\_user\_actions.thriftscala.UnifiedUserAction

import com.twitter.unified\_user\_actions.thriftscala.UnknownNotification

import com.twitter.unified\_user\_actions.thriftscala.UserIdentifier

trait EnricherFixture {

val partitionedTopic = "unified\_user\_actions\_keyed\_dev"

val tweetInfoEnrichmentPlan = EnrichmentPlan(

Seq(

// first stage: to repartition on tweet id -> done

EnrichmentStage(

EnrichmentStageStatus.Completion,

EnrichmentStageType.Repartition,

Seq(EnrichmentInstruction.TweetEnrichment),

Some(partitionedTopic)

),

// next stage: to hydrate more metadata based on tweet id -> initialized

EnrichmentStage(

EnrichmentStageStatus.Initialized,

EnrichmentStageType.Hydration,

Seq(EnrichmentInstruction.TweetEnrichment)

)

))

val tweetNotificationEnrichmentPlan = EnrichmentPlan(

Seq(

// first stage: to repartition on tweet id -> done

EnrichmentStage(

EnrichmentStageStatus.Completion,

EnrichmentStageType.Repartition,

Seq(EnrichmentInstruction.NotificationTweetEnrichment),

Some(partitionedTopic)

),

// next stage: to hydrate more metadata based on tweet id -> initialized

EnrichmentStage(

EnrichmentStageStatus.Initialized,

EnrichmentStageType.Hydration,

Seq(EnrichmentInstruction.NotificationTweetEnrichment),

)

))

def mkUUATweetEvent(tweetId: Long, author: Option[AuthorInfo] = None): UnifiedUserAction = {

UnifiedUserAction(

UserIdentifier(userId = Some(1L)),

item = Item.TweetInfo(TweetInfo(actionTweetId = tweetId, actionTweetAuthorInfo = author)),

actionType = ActionType.ClientTweetReport,

eventMetadata = EventMetadata(1234L, 2345L, SourceLineage.ServerTweetypieEvents)

)

}

def mkUUATweetNotificationEvent(tweetId: Long): UnifiedUserAction = {

mkUUATweetEvent(-1L).copy(

item = Item.NotificationInfo(

NotificationInfo(

actionNotificationId = "123456",

content = NotificationContent.TweetNotification(TweetNotification(tweetId = tweetId))))

)

}

def mkUUAMultiTweetNotificationEvent(tweetIds: Long\*): UnifiedUserAction = {

mkUUATweetEvent(-1L).copy(

item = Item.NotificationInfo(

NotificationInfo(

actionNotificationId = "123456",

content = NotificationContent.MultiTweetNotification(

MultiTweetNotification(tweetIds = tweetIds))))

)

}

def mkUUATweetNotificationUnknownEvent(): UnifiedUserAction = {

mkUUATweetEvent(-1L).copy(

item = Item.NotificationInfo(

NotificationInfo(

actionNotificationId = "123456",

content = NotificationContent.UnknownNotification(UnknownNotification())))

)

}

def mkUUAProfileEvent(userId: Long): UnifiedUserAction = {

val event = mkUUATweetEvent(1L)

event.copy(item = Item.ProfileInfo(ProfileInfo(userId)))

}

}