package com.twitter.recosinjector.decider

import com.twitter.decider.{Decider, DeciderFactory, RandomRecipient, Recipient}

case class RecosInjectorDecider(isProd: Boolean, dataCenter: String) {

lazy val decider: Decider = DeciderFactory(

Some("config/decider.yml"),

Some(getOverlayPath(isProd, dataCenter))

)()

private def getOverlayPath(isProd: Boolean, dataCenter: String): String = {

if (isProd) {

s"/usr/local/config/overlays/recos-injector/recos-injector/prod/$dataCenter/decider\_overlay.yml"

} else {

s"/usr/local/config/overlays/recos-injector/recos-injector/staging/$dataCenter/decider\_overlay.yml"

}

}

def getDecider: Decider = decider

def isAvailable(feature: String, recipient: Option[Recipient]): Boolean = {

decider.isAvailable(feature, recipient)

}

def isAvailable(feature: String): Boolean = isAvailable(feature, Some(RandomRecipient))

}

object RecosInjectorDeciderConstants {

val TweetEventTransformerUserTweetEntityEdgesDecider =

"tweet\_event\_transformer\_user\_tweet\_entity\_edges"

val EnableEmitTweetEdgeFromReply = "enable\_emit\_tweet\_edge\_from\_reply"

val EnableUnfavoriteEdge = "enable\_unfavorite\_edge"

}