package com.twitter.frigate.pushservice.target

import com.twitter.featureswitches.FSCustomMapInput

import com.twitter.featureswitches.parsing.DynMap

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

import com.twitter.frigate.pushservice.util.NsfwInfo

import com.twitter.gizmoduck.thriftscala.User

object CustomFSFields {

private val IsReturningUser = "is\_returning\_user"

private val DaysSinceSignup = "days\_since\_signup"

private val DaysSinceLogin = "days\_since\_login"

private val DaysSinceReactivation = "days\_since\_reactivation"

private val ReactivationDate = "reactivation\_date"

private val FollowGraphSize = "follow\_graph\_size"

private val GizmoduckUserType = "gizmoduck\_user\_type"

private val UserAge = "mr\_user\_age"

private val SensitiveOptIn = "sensitive\_opt\_in"

private val NsfwFollowRatio = "nsfw\_follow\_ratio"

private val TotalFollows = "follow\_count"

private val NsfwRealGraphScore = "nsfw\_real\_graph\_score"

private val NsfwProfileVisit = "nsfw\_profile\_visit"

private val TotalSearches = "total\_searches"

private val NsfwSearchScore = "nsfw\_search\_score"

private val HasReportedNsfw = "nsfw\_reported"

private val HasDislikedNsfw = "nsfw\_disliked"

private val UserState = "user\_state"

private val MrUserState = "mr\_user\_state"

private val NumDaysReceivedPushInLast30Days =

"num\_days\_received\_push\_in\_last\_30\_days"

private val RecommendationsSetting = "recommendations\_setting"

private val TopicsSetting = "topics\_setting"

private val SpacesSetting = "spaces\_setting"

private val NewsSetting = "news\_setting"

private val LiveVideoSetting = "live\_video\_setting"

private val HasRecentPushableRebDevice = "has\_recent\_pushable\_rweb\_device"

private val RequestSource = "request\_source"

}

case class CustomFSFields(

isReactivatedUser: Boolean,

daysSinceSignup: Int,

numDaysReceivedPushInLast30Days: Int,

daysSinceLogin: Option[Int],

daysSinceReactivation: Option[Int],

user: Option[User],

userState: Option[String],

mrUserState: Option[String],

reactivationDate: Option[String],

requestSource: Option[String],

userAge: Option[Int],

nsfwInfo: Option[NsfwInfo],

deviceInfo: Option[DeviceInfo]) {

import CustomFSFields.\_

private val keyValMap: Map[String, Any] = Map(

IsReturningUser -> isReactivatedUser,

DaysSinceSignup -> daysSinceSignup,

DaysSinceLogin -> daysSinceLogin,

NumDaysReceivedPushInLast30Days -> numDaysReceivedPushInLast30Days

) ++

daysSinceReactivation.map(DaysSinceReactivation -> \_) ++

reactivationDate.map(ReactivationDate -> \_) ++

user.flatMap(\_.counts.map(counts => FollowGraphSize -> counts.following)) ++

user.map(u => GizmoduckUserType -> u.userType.name) ++

userState.map(UserState -> \_) ++

mrUserState.map(MrUserState -> \_) ++

requestSource.map(RequestSource -> \_) ++

userAge.map(UserAge -> \_) ++

nsfwInfo.flatMap(\_.senstiveOptIn).map(SensitiveOptIn -> \_) ++

nsfwInfo

.map { nsInfo =>

Map[String, Any](

NsfwFollowRatio -> nsInfo.nsfwFollowRatio,

TotalFollows -> nsInfo.totalFollowCount,

NsfwRealGraphScore -> nsInfo.realGraphScore,

NsfwProfileVisit -> nsInfo.nsfwProfileVisits,

TotalSearches -> nsInfo.totalSearches,

NsfwSearchScore -> nsInfo.searchNsfwScore,

HasReportedNsfw -> nsInfo.hasReported,

HasDislikedNsfw -> nsInfo.hasDisliked

)

}.getOrElse(Map.empty[String, Any]) ++

deviceInfo

.map { deviceInfo =>

Map[String, Boolean](

RecommendationsSetting -> deviceInfo.isRecommendationsEligible,

TopicsSetting -> deviceInfo.isTopicsEligible,

SpacesSetting -> deviceInfo.isSpacesEligible,

LiveVideoSetting -> deviceInfo.isBroadcastsEligible,

NewsSetting -> deviceInfo.isNewsEligible,

HasRecentPushableRebDevice -> deviceInfo.hasRecentPushableRWebDevice

)

}.getOrElse(Map.empty[String, Boolean])

val fsMap = FSCustomMapInput(DynMap(keyValMap))

}