package com.twitter.follow\_recommendations.services

import com.twitter.finagle.stats.StatsReceiver

import javax.inject.Inject

import javax.inject.Singleton

import com.twitter.timelines.configapi.Params

import com.twitter.follow\_recommendations.common.utils.DisplayLocationProductConverterUtil

import com.twitter.follow\_recommendations.configapi.deciders.DeciderParams

import com.twitter.follow\_recommendations.logging.FrsLogger

import com.twitter.follow\_recommendations.models.{DebugParams => FrsDebugParams}

import com.twitter.follow\_recommendations.models.RecommendationRequest

import com.twitter.follow\_recommendations.models.RecommendationResponse

import com.twitter.follow\_recommendations.models.Request

import com.twitter.product\_mixer.core.model.marshalling.request.{

DebugParams => ProductMixerDebugParams

}

import com.twitter.product\_mixer.core.product.registry.ProductPipelineRegistry

import com.twitter.product\_mixer.core.pipeline.product.ProductPipelineRequest

import com.twitter.stitch.Stitch

@Singleton

class ProductMixerRecommendationService @Inject() (

productPipelineRegistry: ProductPipelineRegistry,

resultLogger: FrsLogger,

baseStats: StatsReceiver) {

private val stats = baseStats.scope("product\_mixer\_recos\_service\_stats")

private val loggingStats = stats.scope("logged")

def get(request: RecommendationRequest, params: Params): Stitch[RecommendationResponse] = {

if (params(DeciderParams.EnableRecommendations)) {

val productMixerRequest = convertToProductMixerRequest(request)

productPipelineRegistry

.getProductPipeline[Request, RecommendationResponse](productMixerRequest.product)

.process(ProductPipelineRequest(productMixerRequest, params)).onSuccess { response =>

if (resultLogger.shouldLog(request.debugParams)) {

loggingStats.counter().incr()

resultLogger.logRecommendationResult(request, response)

}

}

} else {

Stitch.value(RecommendationResponse(Nil))

}

}

def convertToProductMixerRequest(frsRequest: RecommendationRequest): Request = {

Request(

maxResults = frsRequest.maxResults,

debugParams = convertToProductMixerDebugParams(frsRequest.debugParams),

productContext = None,

product =

DisplayLocationProductConverterUtil.displayLocationToProduct(frsRequest.displayLocation),

clientContext = frsRequest.clientContext,

serializedRequestCursor = frsRequest.cursor,

frsDebugParams = frsRequest.debugParams,

displayLocation = frsRequest.displayLocation,

excludedIds = frsRequest.excludedIds,

fetchPromotedContent = frsRequest.fetchPromotedContent,

userLocationState = frsRequest.userLocationState

)

}

private def convertToProductMixerDebugParams(

frsDebugParams: Option[FrsDebugParams]

): Option[ProductMixerDebugParams] = {

frsDebugParams.map { debugParams =>

ProductMixerDebugParams(debugParams.featureOverrides, None)

}

}

}