package com.twitter.follow\_recommendations.products.sidebar

import com.twitter.follow\_recommendations.common.base.BaseRecommendationFlow

import com.twitter.follow\_recommendations.common.base.IdentityTransform

import com.twitter.follow\_recommendations.common.base.Transform

import com.twitter.follow\_recommendations.flows.ads.PromotedAccountsFlow

import com.twitter.follow\_recommendations.flows.ads.PromotedAccountsFlowRequest

import com.twitter.follow\_recommendations.blenders.PromotedAccountsBlender

import com.twitter.follow\_recommendations.common.models.DisplayLocation

import com.twitter.follow\_recommendations.common.models.Recommendation

import com.twitter.follow\_recommendations.flows.post\_nux\_ml.PostNuxMlFlow

import com.twitter.follow\_recommendations.flows.post\_nux\_ml.PostNuxMlRequestBuilder

import com.twitter.follow\_recommendations.products.common.Product

import com.twitter.follow\_recommendations.products.common.ProductRequest

import com.twitter.follow\_recommendations.products.sidebar.configapi.SidebarParams

import com.twitter.stitch.Stitch

import javax.inject.Inject

import javax.inject.Singleton

@Singleton

class SidebarProduct @Inject() (

postNuxMlFlow: PostNuxMlFlow,

postNuxMlRequestBuilder: PostNuxMlRequestBuilder,

promotedAccountsFlow: PromotedAccountsFlow,

promotedAccountsBlender: PromotedAccountsBlender)

extends Product {

override val name: String = "Sidebar"

override val identifier: String = "sidebar"

override val displayLocation: DisplayLocation = DisplayLocation.Sidebar

override def selectWorkflows(

request: ProductRequest

): Stitch[Seq[BaseRecommendationFlow[ProductRequest, \_ <: Recommendation]]] = {

postNuxMlRequestBuilder.build(request).map { postNuxMlRequest =>

Seq(

postNuxMlFlow.mapKey({ \_: ProductRequest => postNuxMlRequest }),

promotedAccountsFlow.mapKey(mkPromotedAccountsRequest)

)

}

}

override val blender: Transform[ProductRequest, Recommendation] = {

promotedAccountsBlender.mapTarget[ProductRequest](getMaxResults)

}

private[sidebar] def mkPromotedAccountsRequest(

req: ProductRequest

): PromotedAccountsFlowRequest = {

PromotedAccountsFlowRequest(

req.recommendationRequest.clientContext,

req.params,

req.recommendationRequest.displayLocation,

None,

req.recommendationRequest.excludedIds.getOrElse(Nil)

)

}

private[sidebar] def getMaxResults(req: ProductRequest): Int = {

req.recommendationRequest.maxResults.getOrElse(

req.params(SidebarParams.DefaultMaxResults)

)

}

override def resultsTransformer(

request: ProductRequest

): Stitch[Transform[ProductRequest, Recommendation]] =

Stitch.value(new IdentityTransform[ProductRequest, Recommendation])

override def enabled(request: ProductRequest): Stitch[Boolean] =

Stitch.value(request.params(SidebarParams.EnableProduct))

}