package com.twitter.home\_mixer.controller

import com.twitter.finatra.thrift.Controller

import com.twitter.home\_mixer.marshaller.request.HomeMixerRequestUnmarshaller

import com.twitter.home\_mixer.model.request.HomeMixerRequest

import com.twitter.home\_mixer.service.ScoredTweetsService

import com.twitter.home\_mixer.{thriftscala => t}

import com.twitter.product\_mixer.core.controllers.DebugTwitterContext

import com.twitter.product\_mixer.core.functional\_component.configapi.ParamsBuilder

import com.twitter.product\_mixer.core.service.debug\_query.DebugQueryService

import com.twitter.product\_mixer.core.service.urt.UrtService

import com.twitter.snowflake.id.SnowflakeId

import com.twitter.stitch.Stitch

import com.twitter.timelines.configapi.Params

import javax.inject.Inject

class HomeThriftController @Inject() (

homeRequestUnmarshaller: HomeMixerRequestUnmarshaller,

urtService: UrtService,

scoredTweetsService: ScoredTweetsService,

paramsBuilder: ParamsBuilder)

extends Controller(t.HomeMixer)

with DebugTwitterContext {

handle(t.HomeMixer.GetUrtResponse) { args: t.HomeMixer.GetUrtResponse.Args =>

val request = homeRequestUnmarshaller(args.request)

val params = buildParams(request)

Stitch.run(urtService.getUrtResponse[HomeMixerRequest](request, params))

}

handle(t.HomeMixer.GetScoredTweetsResponse) { args: t.HomeMixer.GetScoredTweetsResponse.Args =>

val request = homeRequestUnmarshaller(args.request)

val params = buildParams(request)

withDebugTwitterContext(request.clientContext) {

Stitch.run(scoredTweetsService.getScoredTweetsResponse[HomeMixerRequest](request, params))

}

}

private def buildParams(request: HomeMixerRequest): Params = {

val userAgeOpt = request.clientContext.userId.map { userId =>

SnowflakeId.timeFromIdOpt(userId).map(\_.untilNow.inDays).getOrElse(Int.MaxValue)

}

val fsCustomMapInput = userAgeOpt.map("account\_age\_in\_days" -> \_).toMap

paramsBuilder.build(

clientContext = request.clientContext,

product = request.product,

featureOverrides = request.debugParams.flatMap(\_.featureOverrides).getOrElse(Map.empty),

fsCustomMapInput = fsCustomMapInput

)

}

}