package com.twitter.graph\_feature\_service.worker.controllers

import com.twitter.discovery.common.stats.DiscoveryStatsFilter

import com.twitter.finagle.Service

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.finatra.thrift.Controller

import com.twitter.graph\_feature\_service.thriftscala

import com.twitter.graph\_feature\_service.thriftscala.Worker.GetIntersection

import com.twitter.graph\_feature\_service.thriftscala.\_

import com.twitter.graph\_feature\_service.worker.handlers.\_

import javax.inject.Inject

import javax.inject.Singleton

@Singleton

class WorkerController @Inject() (

workerGetIntersectionHandler: WorkerGetIntersectionHandler

)(

implicit statsReceiver: StatsReceiver)

extends Controller(thriftscala.Worker) {

// use DiscoveryStatsFilter to filter out exceptions out of our control

private val getIntersectionService: Service[

WorkerIntersectionRequest,

WorkerIntersectionResponse

] =

new DiscoveryStatsFilter[WorkerIntersectionRequest, WorkerIntersectionResponse](

statsReceiver.scope("srv").scope("get\_intersection")

).andThen(Service.mk(workerGetIntersectionHandler))

val getIntersection: Service[GetIntersection.Args, WorkerIntersectionResponse] = { args =>

getIntersectionService(args.request).onFailure { throwable =>

logger.error(s"Failure to get intersection for request $args.", throwable)

}

}

handle(GetIntersection) { getIntersection }

}