package com.twitter.representationscorer.columns

import com.twitter.contentrecommender.thriftscala.ScoringResponse

import com.twitter.representationscorer.scorestore.ScoreStore

import com.twitter.simclusters\_v2.thriftscala.ScoreId

import com.twitter.stitch

import com.twitter.stitch.Stitch

import com.twitter.strato.config.ContactInfo

import com.twitter.strato.config.Policy

import com.twitter.strato.catalog.OpMetadata

import com.twitter.strato.data.Conv

import com.twitter.strato.data.Lifecycle

import com.twitter.strato.data.Description.PlainText

import com.twitter.strato.fed.\_

import com.twitter.strato.thrift.ScroogeConv

import javax.inject.Inject

class ScoreColumn @Inject() (scoreStore: ScoreStore)

extends StratoFed.Column("recommendations/representation\_scorer/score")

with StratoFed.Fetch.Stitch {

override val policy: Policy = Common.rsxReadPolicy

override type Key = ScoreId

override type View = Unit

override type Value = ScoringResponse

override val keyConv: Conv[Key] = ScroogeConv.fromStruct[ScoreId]

override val viewConv: Conv[View] = Conv.ofType

override val valueConv: Conv[Value] = ScroogeConv.fromStruct[ScoringResponse]

override val contactInfo: ContactInfo = Info.contactInfo

override val metadata: OpMetadata = OpMetadata(

lifecycle = Some(Lifecycle.Production),

description = Some(PlainText(

"The Uniform Scoring Endpoint in Representation Scorer for the Content-Recommender." +

" TDD: http://go/representation-scorer-tdd Guideline: http://go/uniform-scoring-guideline"))

)

override def fetch(key: Key, view: View): Stitch[Result[Value]] =

scoreStore

.uniformScoringStoreStitch(key)

.map(score => found(ScoringResponse(Some(score))))

.handle {

case stitch.NotFound => missing

}

}