package com.twitter.usersignalservice.columns

import com.twitter.stitch.NotFound

import com.twitter.stitch.Stitch

import com.twitter.strato.catalog.OpMetadata

import com.twitter.strato.catalog.Ops

import com.twitter.strato.config.Policy

import com.twitter.strato.config.ReadWritePolicy

import com.twitter.strato.data.Conv

import com.twitter.strato.data.Description

import com.twitter.strato.data.Lifecycle

import com.twitter.strato.fed.StratoFed

import com.twitter.strato.thrift.ScroogeConv

import com.twitter.usersignalservice.service.UserSignalService

import com.twitter.usersignalservice.thriftscala.BatchSignalRequest

import com.twitter.usersignalservice.thriftscala.BatchSignalResponse

import javax.inject.Inject

class UserSignalServiceColumn @Inject() (userSignalService: UserSignalService)

extends StratoFed.Column(UserSignalServiceColumn.Path)

with StratoFed.Fetch.Stitch {

override val metadata: OpMetadata = OpMetadata(

lifecycle = Some(Lifecycle.Production),

description = Some(Description.PlainText("User Signal Service Federated Column")))

override def ops: Ops = super.ops

override type Key = BatchSignalRequest

override type View = Unit

override type Value = BatchSignalResponse

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

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

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

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

userSignalService

.userSignalServiceHandlerStoreStitch(key)

.map(result => found(result))

.handle {

case NotFound => missing

}

}

}

object UserSignalServiceColumn {

val Path = "recommendations/user-signal-service/signals"

}