package com.twitter.product\_mixer.component\_library.side\_effect

import com.twitter.product\_mixer.core.functional\_component.side\_effect.PipelineResultSideEffect

import com.twitter.product\_mixer.core.model.marshalling.HasMarshalling

import com.twitter.product\_mixer.core.pipeline.PipelineQuery

import com.twitter.stitch.Stitch

import com.twitter.user\_session\_store.ReadWriteUserSessionStore

import com.twitter.user\_session\_store.WriteRequest

/\*\*

\* A [[PipelineResultSideEffect]] that writes to a [[ReadWriteUserSessionStore]]

\*/

trait UserSessionStoreUpdateSideEffect[

Request <: WriteRequest,

Query <: PipelineQuery,

ResponseType <: HasMarshalling]

extends PipelineResultSideEffect[Query, ResponseType] {

/\*\*

\* Build the write request from the query

\* @param query PipelineQuery

\* @return WriteRequest

\*/

def buildWriteRequest(query: Query): Option[Request]

val userSessionStore: ReadWriteUserSessionStore

final override def apply(

inputs: PipelineResultSideEffect.Inputs[Query, ResponseType]

): Stitch[Unit] = {

buildWriteRequest(inputs.query)

.map(userSessionStore.write)

.getOrElse(Stitch.Unit)

}

}