package com.twitter.product\_mixer.core.module

import com.google.inject.Provides

import com.twitter.abdecider.ABDeciderFactory

import com.twitter.abdecider.LoggingABDecider

import com.twitter.finagle.stats.StatsReceiver

import com.twitter.inject.TwitterModule

import com.twitter.inject.annotations.Flag

import com.twitter.logging.\_

import com.twitter.product\_mixer.core.module.product\_mixer\_flags.ProductMixerFlagModule.ScribeABImpressions

import javax.inject.Singleton

object ABDeciderModule extends TwitterModule {

private val YmlPath = "/usr/local/config/abdecider/abdecider.yml"

@Provides

@Singleton

def provideLoggingABDecider(

@Flag(ScribeABImpressions) isScribeAbImpressions: Boolean,

stats: StatsReceiver

): LoggingABDecider = {

val clientEventsHandler: HandlerFactory =

if (isScribeAbImpressions) {

QueueingHandler(

maxQueueSize = 10000,

handler = ScribeHandler(

category = "client\_event",

formatter = BareFormatter,

level = Some(Level.INFO),

statsReceiver = stats.scope("abdecider"))

)

} else { () =>

NullHandler

}

val factory = LoggerFactory(

node = "abdecider",

level = Some(Level.INFO),

useParents = false,

handlers = clientEventsHandler :: Nil

)

val abDeciderFactory = ABDeciderFactory(

abDeciderYmlPath = YmlPath,

scribeLogger = Some(factory()),

environment = Some("production")

)

abDeciderFactory.buildWithLogging()

}

}