package com.twitter.product\_mixer.component\_library.feature.featurestorev1

import com.twitter.ml.api.transform.FeatureRenameTransform

import com.twitter.ml.featurestore.catalog.entities

import com.twitter.ml.featurestore.lib.EntityId

import com.twitter.ml.featurestore.lib.UserId

import com.twitter.ml.featurestore.lib.entity.Entity

import com.twitter.ml.featurestore.lib.entity.EntityWithId

import com.twitter.ml.featurestore.lib.feature.TimelinesAggregationFrameworkFeatureGroup

import com.twitter.ml.featurestore.lib.feature.{Feature => FSv1Feature}

import com.twitter.product\_mixer.component\_library.model.candidate.TweetAuthorIdFeature

import com.twitter.product\_mixer.component\_library.model.candidate.TweetCandidate

import com.twitter.product\_mixer.core.feature.featuremap.FeatureMap

import com.twitter.product\_mixer.core.feature.featurestorev1.\_

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

import com.twitter.timelines.configapi.FSParam

import scala.reflect.ClassTag

object FeatureStoreV1TweetCandidateAuthorIdFeature {

def apply[Query <: PipelineQuery, Value](

feature: FSv1Feature[UserId, Value],

legacyName: Option[String] = None,

defaultValue: Option[Value] = None,

enabledParam: Option[FSParam[Boolean]] = None

): FeatureStoreV1CandidateFeature[Query, TweetCandidate, \_ <: EntityId, Value] =

FeatureStoreV1CandidateFeature(

feature,

TweetCandidateAuthorIdEntity,

legacyName,

defaultValue,

enabledParam)

}

object FeatureStoreV1TweetCandidateAuthorIdAggregateFeature {

def apply[Query <: PipelineQuery](

featureGroup: TimelinesAggregationFrameworkFeatureGroup[UserId],

enabledParam: Option[FSParam[Boolean]] = None,

keepLegacyNames: Boolean = false,

featureNameTransform: Option[FeatureRenameTransform] = None

): FeatureStoreV1CandidateFeatureGroup[Query, TweetCandidate, \_ <: EntityId] =

FeatureStoreV1CandidateFeatureGroup(

featureGroup,

TweetCandidateAuthorIdEntity,

enabledParam,

keepLegacyNames,

featureNameTransform

)(implicitly[ClassTag[UserId]])

}

object TweetCandidateAuthorIdEntity

extends FeatureStoreV1CandidateEntity[PipelineQuery, TweetCandidate, UserId] {

override val entity: Entity[UserId] = entities.core.Author

override def entityWithId(

query: PipelineQuery,

tweet: TweetCandidate,

existingFeatures: FeatureMap

): EntityWithId[UserId] =

entity.withId(UserId(existingFeatures.get(TweetAuthorIdFeature)))

}