namespace java com.twitter.recos.user\_user\_graph.thriftjava

namespace py gen.twitter.recos.user\_user\_graph

#@namespace scala com.twitter.recos.user\_user\_graph.thriftscala

#@namespace strato com.twitter.recos.user\_user\_graph

namespace rb UserUserGraph

include "com/twitter/recos/recos\_common.thrift"

enum RecommendUserDisplayLocation {

MagicRecs = 0

HomeTimeLine = 1

ConnectTab = 2

}

struct RecommendUserRequest {

1: required i64 requesterId // user id of the requesting user

2: required RecommendUserDisplayLocation displayLocation // display location from the client

3: required map<i64,double> seedsWithWeights // seed ids and weights used in left hand side

4: optional list<i64> excludedUserIds // list of users to exclude from response

5: optional i32 maxNumResults // number of results to return

6: optional i32 maxNumSocialProofs // number of social proofs per recommendation

7: optional map<recos\_common.UserSocialProofType, i32> minUserPerSocialProof // minimum number of users for each social proof type

8: optional list<recos\_common.UserSocialProofType> socialProofTypes // list of required social proof types. Any recommended user

// must at least have all of these social proof types

9: optional i64 maxEdgeEngagementAgeInMillis // only events created during this period are counted

}

struct RecommendedUser {

1: required i64 userId // user id of recommended user

2: required double score // weight of the recommended user

3: required map<recos\_common.UserSocialProofType, list<i64>> socialProofs // the social proofs of the recommended user

}

struct RecommendUserResponse {

1: required list<RecommendedUser> recommendedUsers // list of recommended users

}

/\*\*

\* The main interface-definition for UserUserGraph.

\*/

service UserUserGraph {

// Given a request for recommendations for a specific user,

// return a list of candidate users along with their social proofs

RecommendUserResponse recommendUsers (RecommendUserRequest request)

}