This API might have following functions:

1. Similarity between two plain texts, phrases or words:
   1. Concept similarity: Talking about the same concept. Does not matter for example if one is negative sentence and one is positive sentence.
      1. I like football; I do not like football
   2. Concept and meaning similarity: Talking about the same concept, by considering the syntax.
2. Similarity between two events:
   1. Concept similarity by considering only the event textual description like event title and event description.
   2. Event similarity by considering all event data like textual data, date, time, address and some other available data.
3. Similarity between two groups:
   1. Concept similarity by considering only the group textual description like group title and group description.
   2. Group similarity by considering all group data like textual data, address, location, members and some other available data.
4. Similarity between two projects:
   1. So far since we do not know what kind of data would be available, the project similarity would be based on whole textual data.
5. Similarity between two persons
   1. Similarity based on their interests.
   2. Similarity based on their skills.
   3. Similarity based on two texts provided by two persons.
   4. Similarity based on textual data of two persons (such as above data).
   5. Similarity based on textual data and some other data like major, DOB, gender and etc.
6. Match weight between an event and a person
7. Match weight between a group and a person
8. Match weight between a project and a person

For each of above entities (event, group, project and users), we better to have one specific collection of words. Each collection would have some words that will be used for above functions.

Generating of these collections might be done in two step:

1. Extracting a bigger collection by computer automatically (coarse selection)
2. Filtering out or in some words (fine selection)

For each above function, there might be different methods to do calculation. While other APIs call this API, they might select which method should be used (by considering a default method for each function).

All similarity functions could be called in array manner, so, by a single call, these functions will calculate the similarity between bunch of entities.

All Matching functions could be called by array of persons, so, by a single call, these functions will calculate the Match weight between all persons and related of entity.

**I thought a lot on how we could define the API ports, so have it really easy to use. BTW, let’s talk about them tomorrow to have a comprehensive thought about it.**

[

{ fieldname1:fieldvalue1, fieldname1:fieldvalue1, fieldname1:fieldvalue1},

{ fieldname1:fieldvalue1, fieldname1:fieldvalue1, fieldname1:fieldvalue1},

{ fieldname1:fieldvalue1, fieldname1:fieldvalue1, fieldname1:fieldvalue1},

{ fieldname1:fieldvalue1, fieldname1:fieldvalue1, fieldname1:fieldvalue1}

]

[fielname1,fieldnam3]

{fname:””,lname:””,keywoerds:[],bio:””}

[keywords,bio]

keywords:[a,b,c]

keywords:[{kt:1,kv:a},{}]

keywords:{k1:a,k2:b}

[{rank:”5”,match\_weight:”0”}, {rank:”1”,match\_weight:”0”},{}]

person\_project

person\_person

person, project, event, group, other

event\_project

**Parameter 1**

*List of inputs — JSON Format*

*P1: [{key:value,…},{}]*

**Parameter 2**

*Match source — JSON Format*

*P2: {key:value,…}*

**Parameter 3**

*Define parameter category name for matching*

Require value: *parameter* 1 category

*P3: String*

**Parameter 4**

*Define parameter category name for matching*

Require value: *parameter* 2 category

*P4: String*

**Parameter 5**

*Define parameter field names for matching*

Require array: *parameter* 1 values for matching

*P5: [Strings]*

**Parameter 6**

*Define parameter field names for matching*

Require array: *parameter* 2 values for matching

*P6: [Strings]*

Result = FunName(P1,P2,[P3],[P4],[P5],[P6])

[{rank:”5”,match\_weight:”0”}, {rank:”1”,match\_weight:”0”},{},…]