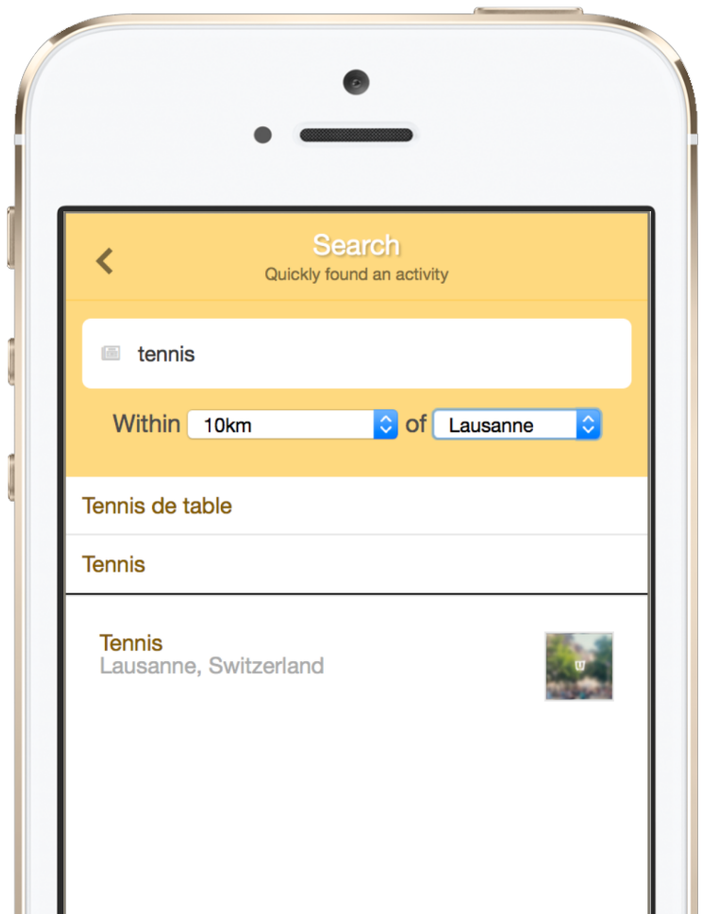
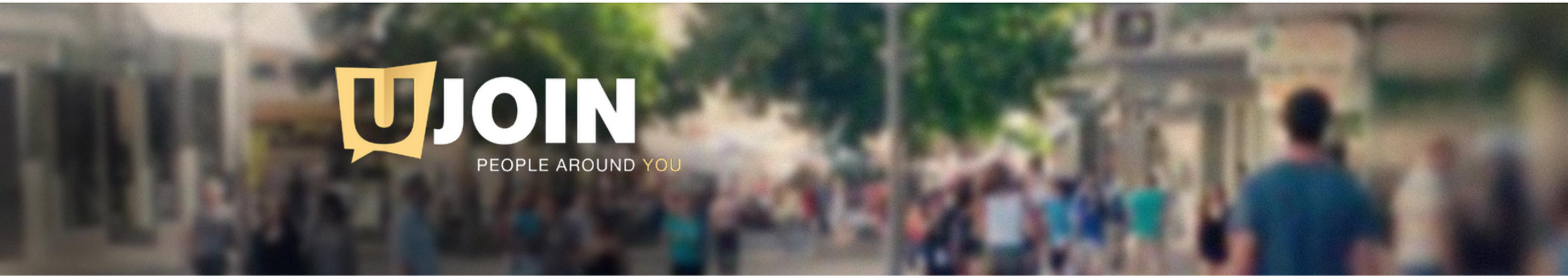
Web Search Engine

for a Social Network



Loïc Serafin

Supervised by Laurent Rime



uJoin gives the possibility to find **activities** nearby
and to discover **people** of your area

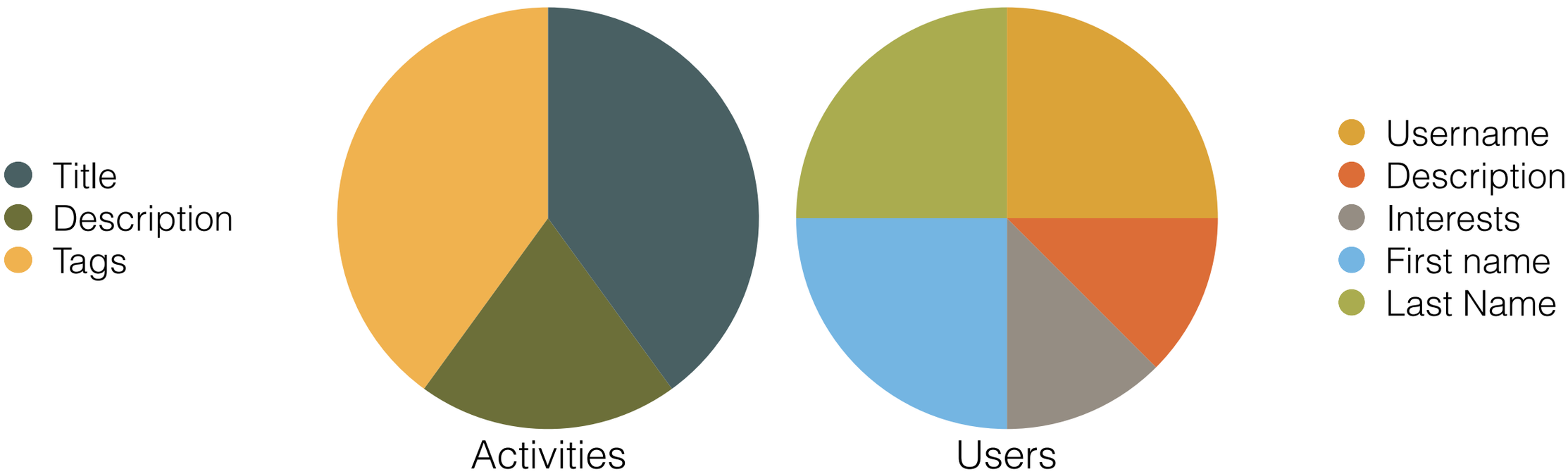
http://www.ujoin.ch/

Definition of the problem

- What does the user want to search ?
- What are the different criteria of a query ?
- How should we rank the different results ?

Defining the different criteria of a query

and how to weight them



Geolocation filter

- User can give a city and a radius distance
- Results will only be geolocated in that area

Image source: maps.google.ch/

Search engine

- Indexes the documents (Databases)
- Searches within its indexes for matching elements and rank them (SPH04 based on Okapi BM25)
- Used by tumblr, joomla, …

Ranking the results

Okapi BM25

by Stephen E. Robertson, Karen Spärck Jones, and others

Given a query Q, containing keywords $q_1, ..., q_n$, the BM25 score of a document is:

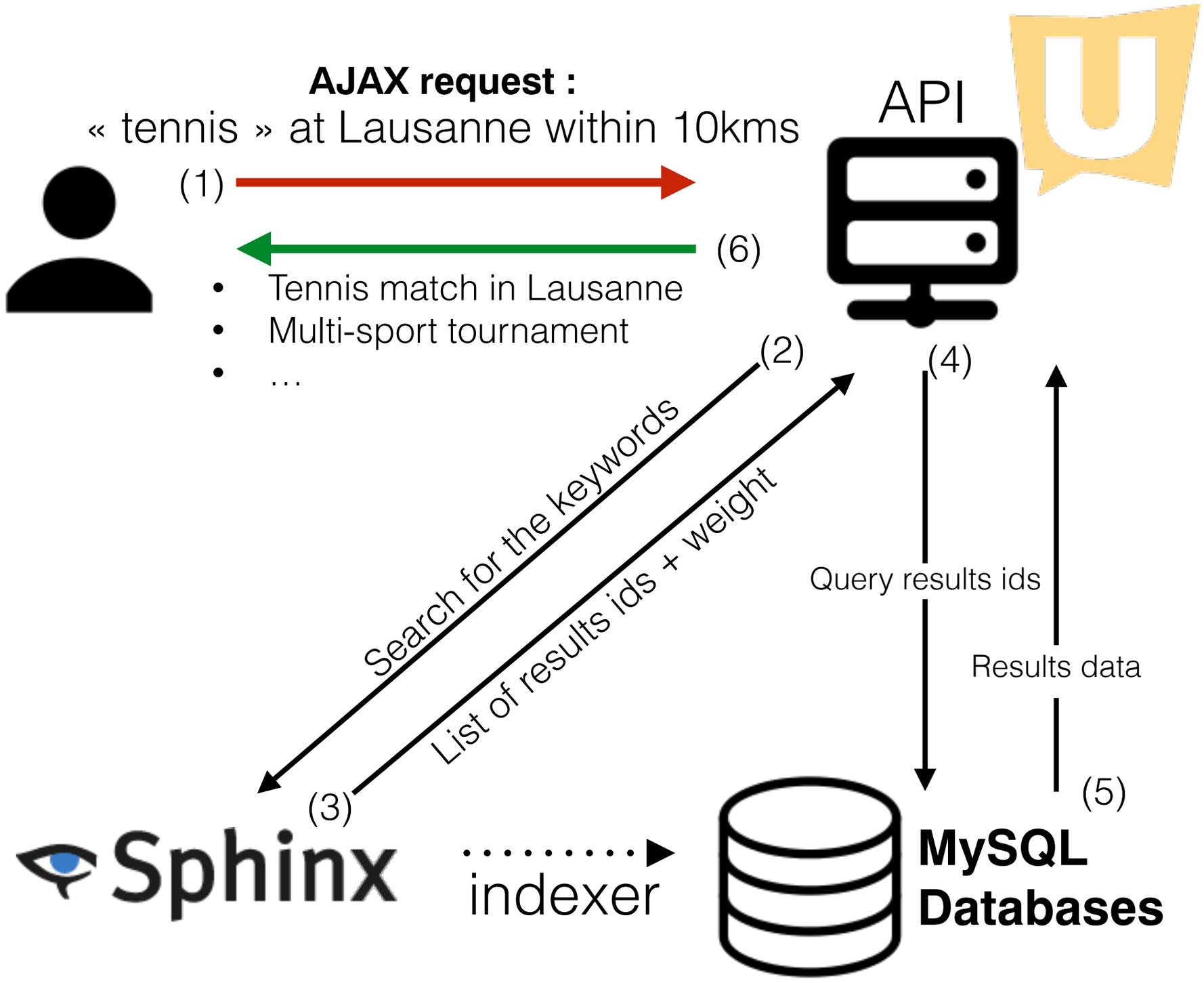
$$score(D,Q) = \sum_{i=0}^{n} IDF(q_i) * \frac{f(q_i,D) * (k_1+1)}{f(q_i,D) + k_1 * (1 - b + b * \frac{|D|}{avgdl})} \quad (1)$$

where $f(q_i, D)$ is q_i's term frequency in the document D, $avgdl$ is the average document length, k_1 and b are parameters chosen by the implementation of the function (in this case Sphinx), $IDF(q_i)$ is the inverse document frequency weight of the query term q_i defined by:

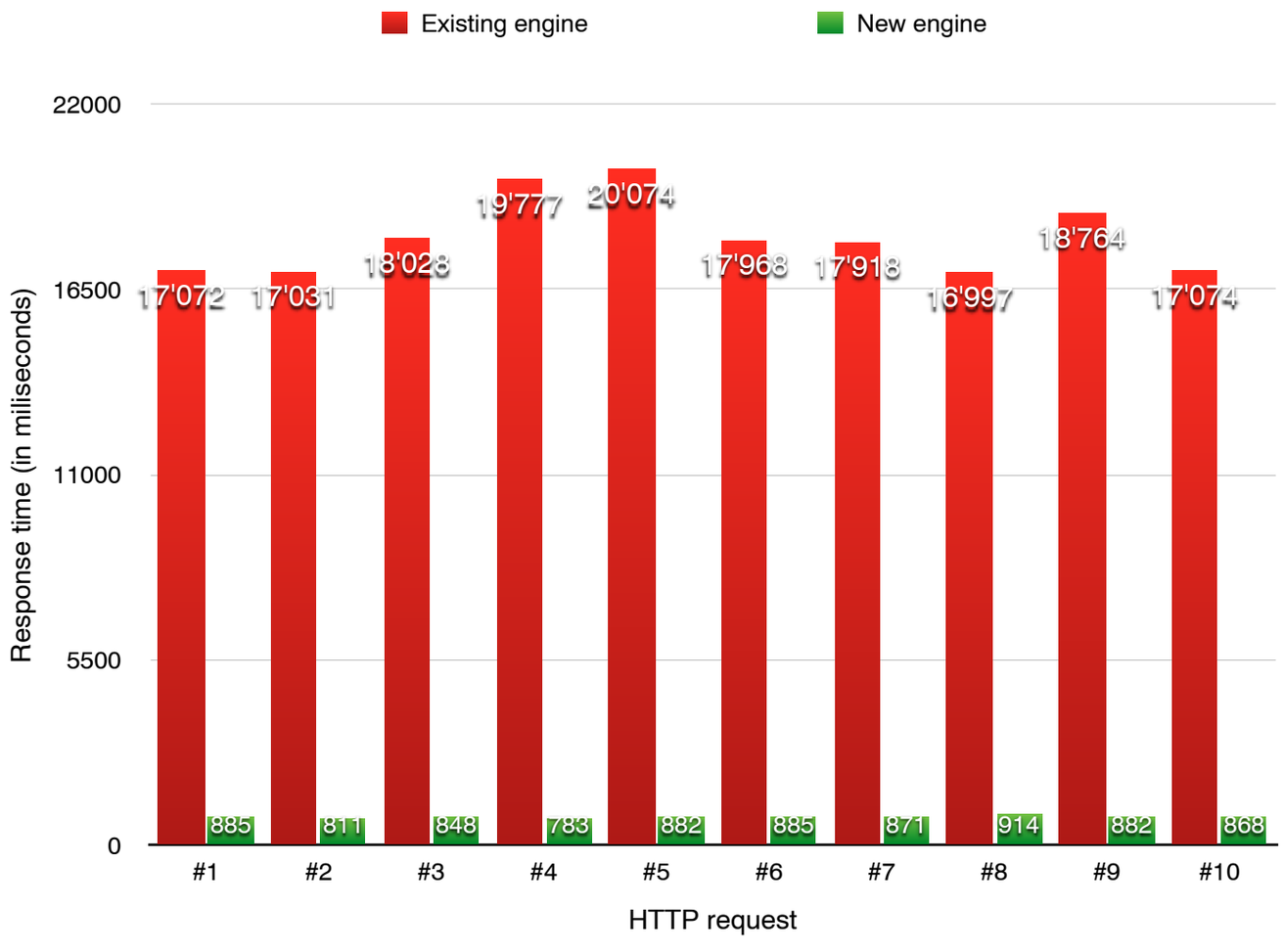
$$IDF(q_i) = log \frac{N - n(q-i) + 0.5}{n(q_i) + 0.5} \quad (2)$$

where N is the total number of documents and $n(q_i)$ the number of documents containing q_i.

Search engine overview



Comparison with the existing engine



Average: 18070.3ms
Standard deviation: 1080.8ms

Server specifications:

- Intel(R) Xeon(R) CPU E5-2630L 0 @ 2.00GHz
- 512 MB Memory
- 20GB SSD Disk
- Ubuntu 14.10
- Apache2

Average: 778ms
Standard deviation: 37.3ms

40’000 entries in the database, ~600 results matching

My contributions

- Interface Sphinx with uJoin's databases
- Search activities in the uJoin application
- Search uJoiners in the uJoin API
- Implement geolocation filtering
- Implement search history for each user

Future development

- Search for tags, feeds, …
- Optimize search on the interests criteria
- Have a dedicated page for global searching