getaLife

Minardi, Claudia
minardi.claudia@gmail.com

Ortega, Julian
jortegac@gmail.com

Stablum, Francesco
stablum@gmail.com

March 21, 2014

Contents

1 Introduction

Looking for events online has become a common practice in order to get information on how to spend ones free time. For this reason there has been a proliferation of websites and underlying databases that aim to gather event data. What they typically offer is a search form and then a tabular view of the matching events.

What we offer is an innovative interface for performing similar queries. In our system, the geographical map is center stage and gets populated by the events that match a certain query.

This document is structured as follows: Section 2 details the functionality and main objectives of the application. Section 3 discusses the datasets and services used by the application. Section 4 details the inferencing performed by the knowledge base and the new information generated from it. Section 5 contains details about the architecture of the application as well as some observations about some bumps found during the endeavor of building the application. Section 6 encompasses the work that would be of interest to carry out to take the application to the next level. Finally, Section 7 specifies the conclusions derived from the realization of this project.

2 Purpose of the web application

The idea at the base of this proposal is to build a Web Application capable of locating events in the Netherlands in a city-based way and provide information about them.

The application targets every kind of user, and provides them with the possibility of viewing what are the events and the activities in the vicinity that could be interesting to attend. By showing these events on a map the user can easily select them based on their current (or desired) location, and access the additional information (time of the event, what kind of event it is and so on).

Based on this general description, we can highlight some basic functionalities that, assembled together, will form the core of the web application.

2.1 Searching for events

The user, by accessing the web page, will be able to perform a search for events (See Figure 1 by specifying the time period of interest and the desired location. This last field can be filled either manually (e.g. I live in Amsterdam but next week I'm going to visit Utrecht, so I am interested in that particular city), or automatically, by registering the actual position of the user.

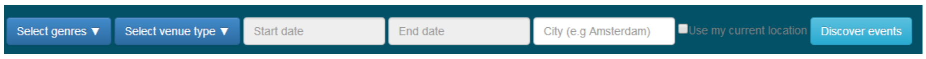


Figure 1: getaLife's search bar.

The search for events will provide also additional filtering, allowing the user to search for events of a particular genre (e.g.movies, jazz music) or venues of a particular kind (e.g. museums, theaters). This kind of feature allows for a higher level of customization.

It is worthwhile mentioning that since the main goal of the application is to provide means to discover events, all the search parameters are optional. If the user where to click the search button without indicating any sub-selection criterias via the search parameters, all the venues, and their events, available in the knowledge base will be retrieved and displayed to the user. In essence, the parameters provide means to reduce the scope of the discovery to a more limited range of possibilities.

2.2 Showing events on a map

Once the application correctly gathered the data about events and the locations in which those are held, the information will be displayed on the map, which constitutes the central element of the User Interface. As an example, Figure 2 shows venues placed on the map resulting for a scoped search for events in *Utrecht*.

This kind of visualization was chosen because of its simple realization (see Section 3.2) and of the advantages it brings: the user can easily recognize the locations closest to their own location or more accessible, and at the same time still have an overview of all the results of his search; additionally, the chosen map implementation provides a neat and simple way of displaying information about a particular location or event, as will be further explained in the following sections.

2.3 Displaying information about events

Once the events are correctly displayed on the map, the user will be able to access all the gathered information about them.

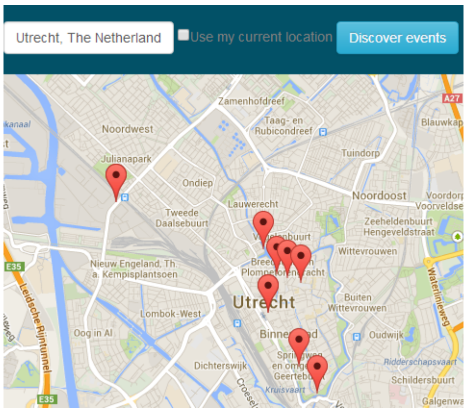
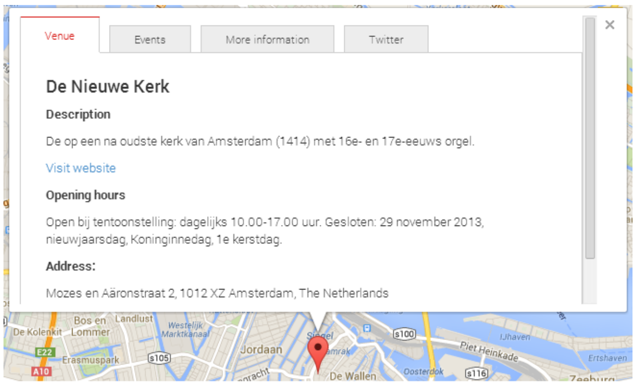


Figure 2: Venues in Utrecht.

The goal is to show information about the venue (See Figure 3) in which the event is held (where is it, opening hours..), accompanied by information on all the events hosted in that particular location (See Figure 4), such as a general description, the starting and ending time, external links and so forth.



De Nieuwe Kerk

Description

De op een na oudste kerk van Amsterdam (1414) met 16e- en 17e-eeuws orgel.

Visit website

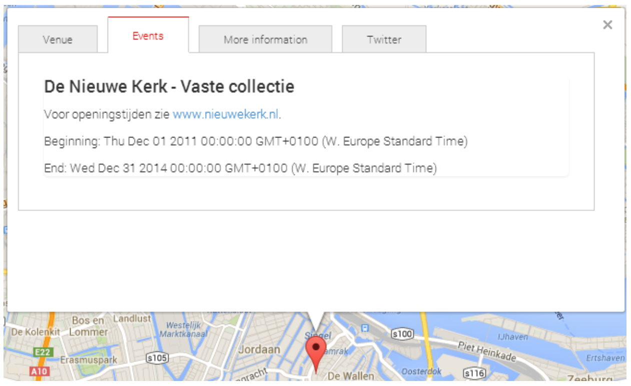
Opening hours

Open bij tentoonstelling: dagelijks 10.00-17.00 uur. Gesloten: 29 november 2013, nieuwjaarsdag, Koninginnedag, 1e kerstdag.

Address:

Mozes en Aäronstraat 2, 1012 XZ Amsterdam, The Netherlands

Figure 3: Nieuwe Kerk in Amsterdam.



De Nieuwe Kerk - Vaste collectie

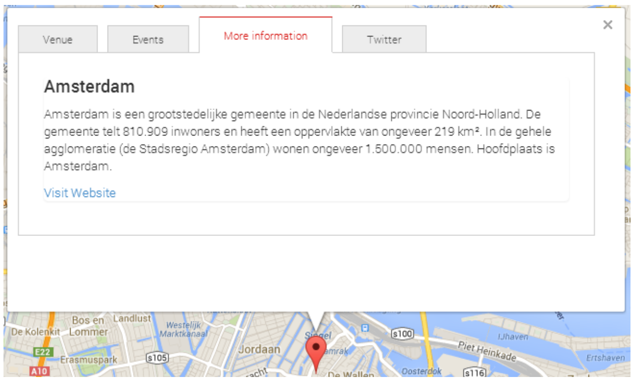
Voor openingstijden zie www.nieuwekerk.nl.

Beginning: Thu Dec 01 2011 00:00:00 GMT+0100 (W. Europe Standard Time)

End: Wed Dec 31 2014 00:00:00 GMT+0100 (W. Europe Standard Time)

Figure 4: Events in the Nieuwe Kerk.

This information will be additionally integrated with more general information about the venues e.g. description, name of the architect who designed it and its current purpose (See Figure 5).

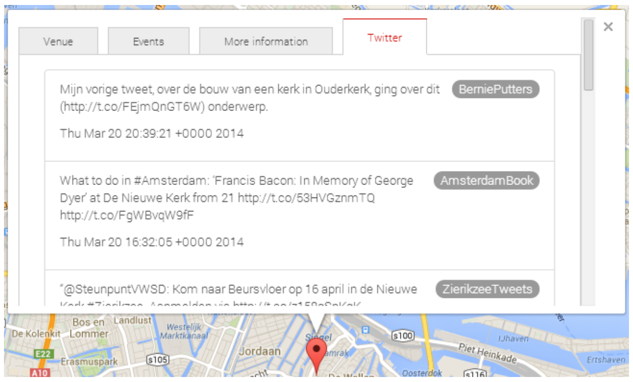


Amsterdam

Amsterdam is een grootstedelijke gemeente in de Nederlandse provincie Noord-Holland. De gemeente telt 810.909 inwoners en heeft een oppervlakte van ongeveer 219 km². In de gehele agglomeratie (de Stadsregio Amsterdam) wonen ongeveer 1.500.000 mensen. Hoofdplaats is Amsterdam.

Figure 5: Additional information for the Nieuwe Kerk.

Finally, the user will be able to see all the *tweets* about a particular venue done by Twitter users (See Figure 6); this last functionality was implemented to offer a *social* view of this particular web application: the user will be able to see what is the general opinion about that particular location or if other people are thinking of attending a particular event there, information that could guide his decision on which event to attend.



Mijn vorige tweet, over de bouw van een kerk in Ouderkerk, ging over dit (http://t.co/FEjmQnGT6W) onderwerp. BerniePutters

Thu Mar 20 20:39:21 +0000 2014

What to do in #Amsterdam: 'Francis Bacon: In Memory of George Dyer' at De Nieuwe Kerk from 21 http://t.co/53HVGznmTQ http://t.co/FgWBvqW9fF AmsterdamBook

Thu Mar 20 16:32:05 +0000 2014

"@SteunpuntVWSD: Kom naar Beursvloer op 16 april in de Nieuwe ZierikzeeTweets

Figure 6: Twitter tweets discussing the Nieuwe Kerk.

2.4 Statistics from the search results

Visualization of aggregate information about the search results across more dimensions is a feature that allows the user to get a better idea on what is charachterizing the results of the entered query.

An aspect that is interesting to explore is the relationship between a geographical location, venue types and the actual even genres that carachterize venues of a certain type.

A plot with stacked bars has been added to the map in order to provide an answer to that question.

Stacked visualization As can be seen in Figure 7, in the *stacked* visualization, on the x axis are placed all possible venue types extracted from the results of the searched query. The y axis represents the amount of events in the query's results, that belong to that specific venue type. Within the venue type's bar are visible several stacked layers with different colors. Each color represent a genre, which is clearly described in the legend.

In this way it's possible to see interesting correlations between venue types and event genres.

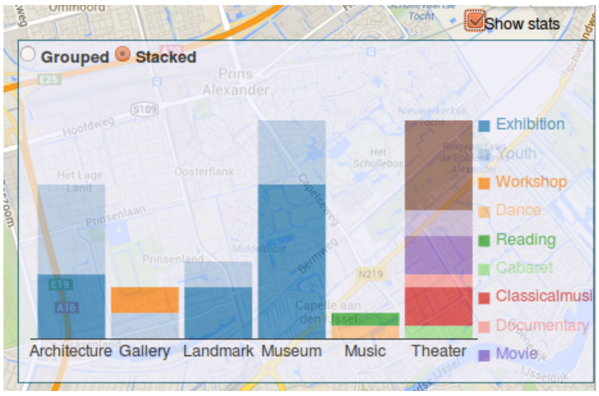


Figure 7: Stacked visualization of venue type/event genre aggregates.

Grouped visualization An additional *grouped* visualization variant is available. In this visualization all the layers are not anymore stacked but are placed on the x axis. This visualization is useful to get a better idea of the proportions between different *venue type/event genre* combinations. This is clearly visible in Figure 8.

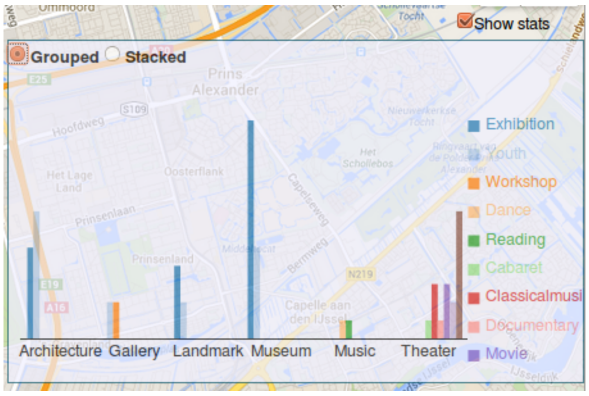


Figure 8: Grouped visualization of venue type/event genre aggregates.

The visualization of the stats panel can be easily toggled via a checkbox placed on the upper right side of the map. The visualization type can also be switched via a radio control.

The plot, produced in SVG format, has been implemented with *D3*, a flexible javascript framework that is widely used for information visualization [13].

3 Datasets and services

This section discusses the datasets and services used by the application.

3.1 ArtsHolland

The main dataset at the base of this idea is the one provided by ArtsHolland [19] which contains information about events and venues in the Netherlands' biggest cities, publicly accessible cultural heritage sites, and reviews, articles and blog posts on these entities.

The data is stored in RDF format, thus obtainable with SPARQL queries, and manageable in XML or JSON. This makes it highly suited for this project and easily integrable in a web application. Through this dataset it is possible to obtain various information, such as:

- information about venues (name, description, location)
- exact date, starting and ending time of the event
- the genre of the event (exhibition, etc..)
- the name of the event
- the description (both short and long)
- the webpage of the event

3.2 Google Maps API

3.2.1 Map UI

Google Maps Javascript API v3 [18] was used for purposes of displaying a map on the screen, alongside the necessary components that are illustrated in the map, namely the location markers and the information windows. Figure 9 exemplifies the view of the map. Section 2 presented a thorough view of the utilization of the visual map components.

Figure 9: getaLife's map view.

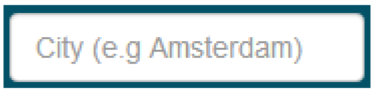


Figure 10: City input box in the UI.

3.2.2 Places autocomplete

The Google Maps JavaScript API v3 library for places [1] was used to facilitate the input of a city area name in which events should be searched for (See Figure 10).

When the text changes withing the box, the API is configured to return city names within the Netherlands, as the ArtsHolland data only involves information regarding said country, and upon selection it will autocomplete the input term with the selected city name (See Figure 11).

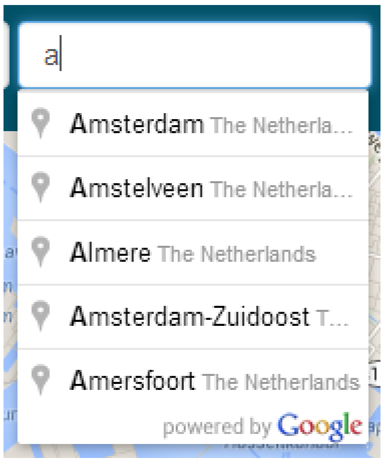

a
Amsterdam The Netherla...
Amstelveen The Netherla...
Almere The Netherlands
Amsterdam-Zuidoost T...
Amersfoort The Netherlands
powered by Google

Figure 11: City input box autocompletion.

3.2.3 Location detection

The mechanism of location detection has been implemented thanks to the native geolocation functionalities in HTML5 [4] .
For this reason two factors should be held into account:

- If the browser does not support HTML5, the feature will not be made available to the user;
- For the feature to be available the user has to authorize the browser to register their actual position;

If the two previous conditions are met, the browser saves the user's current position, that will be converted by the web application into coordinates in a format compatible with Google APIs for further use (see Section 3.2.4).

3.2.4 Geocoding and reverse geocoding

The Google Maps JavaScript API v3 Geocoding Service [2] is used in two places within the application: when selecting a location for the search parameters and when displaying the address of a venue.

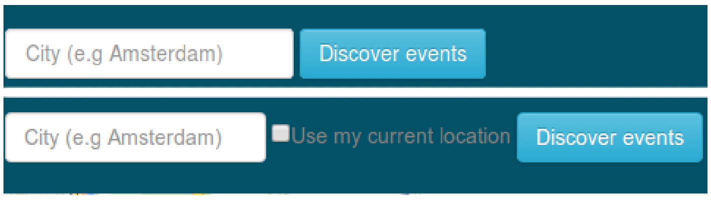
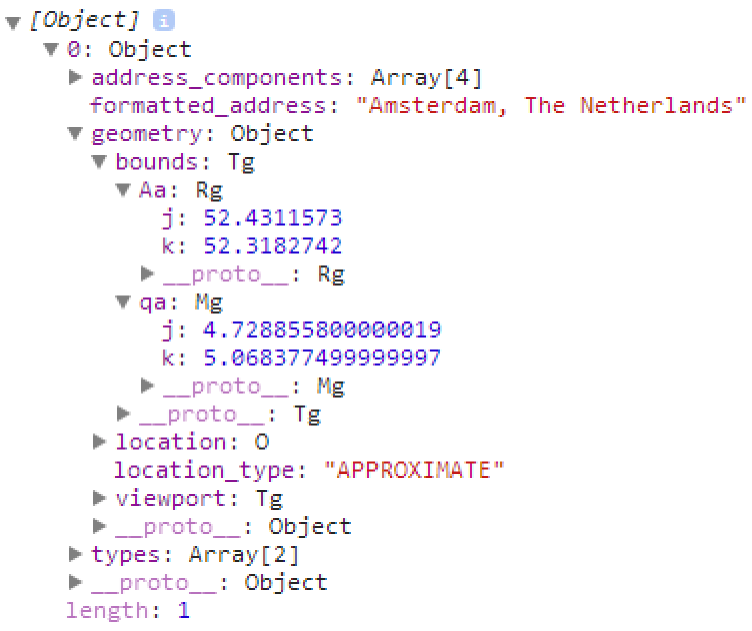


Figure 12: Search panel with and without geolocation services.

When selecting a location for the search parameters from the autocomplete box, the geocoding service is queried with this location name and the geocoordinate bounds for this location are extracted, which in turn are used as part of the query to the RDF store, ensuring that the venues to be retrieved are inside this bounding box.

For example, if the selected city is *Amsterdam, The Netherlands* the bounding box can be determined from the data retrieved from Google (See Figure 13). The relevant geocoordinates, in this particular case, are:

- **North:** 52.4311573
- **South:** 52.3182742
- **West:** 4.728855800000019
- **East:** 5.068377499999997

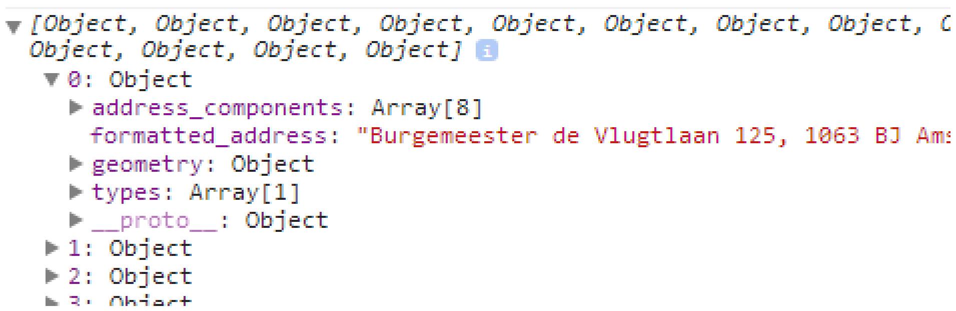


```
▼ [Object]
  ▼ 0: Object
    ▶ address_components: Array[4]
      formatted_address: "Amsterdam, The Netherlands"
    ▼ geometry: Object
      ▼ bounds: Tg
        ▼ Aa: Rg
            j: 52.4311573
            k: 52.3182742
          ▶ __proto__: Rg
        ▼ qa: Mg
            j: 4.728855800000019
            k: 5.068377499999997
          ▶ __proto__: Mg
        ▶ __proto__: Tg
      ▶ location: O
        location_type: "APPROXIMATE"
      ▶ viewport: Tg
      ▶ __proto__: Object
    ▶ types: Array[2]
    ▶ __proto__: Object
    length: 1
```

Figure 13: Geocoding information for Amsterdam.

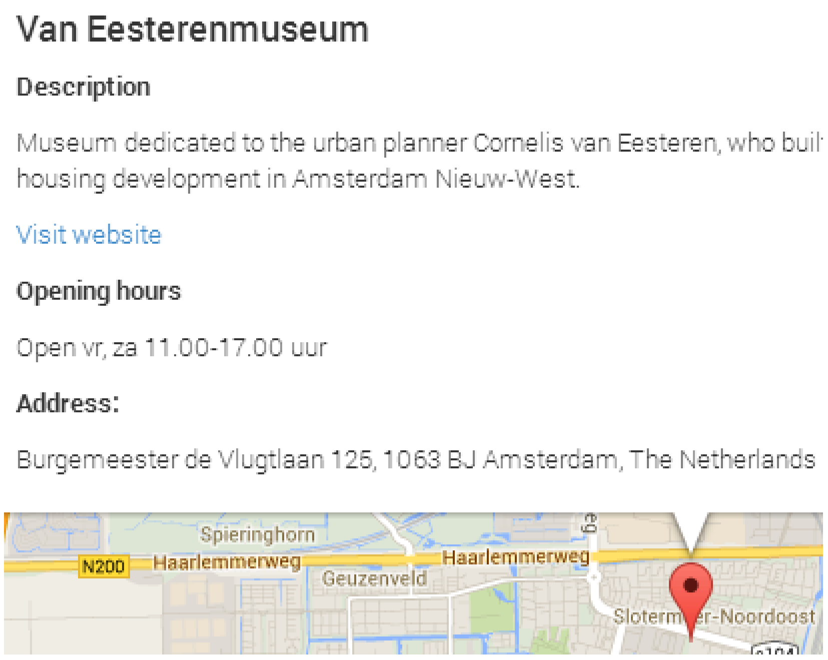
The reverse geocoding service from Google [3] is used when retrieving the venues information from the RDF store. Each venue has a latitude and longitude that is used to display a marker on the map. However, far more interesting for the user, would be to be able to know the location's address. To achieve just this, Google's service is queried with the geocoordinates on the venue and from the response the address is taken out and then used in the venue information window .

Figure 14 shows the JSON response of reverse geocoding the location of *Van Eesteren-museum* showcasing the formatted address. Figure 15 displays the venue information window containing the more user-friendly address.



```
▼ [Object, Object, Object, Object, Object, Object, Object, Object, C
  Object, Object, Object, Object]
   ▼ 0: Object
     ▶ address_components: Array[8]
       formatted_address: "Burgemeester de Vlugtlaan 125, 1063 BJ Ams
     ▶ geometry: Object
     ▶ types: Array[1]
     ▶ __proto__: Object
   ▶ 1: Object
   ▶ 2: Object
```

Figure 14: Reverse geocoding JSON response for *Van Eesterenmuseum*.



Van Eesterenmuseum

Description

Museum dedicated to the urban planner Cornelis van Eesteren, who buil
housing development in Amsterdam Nieuw-West.

Visit website

Opening hours

Open vr, za 11.00-17.00 uur

Address:

Burgemeester de Vlugtlaan 125, 1063 BJ Amsterdam, The Netherlands

Figure 15: Displaying venue address.

3.3 Foursquare

Foursquare's Search Venues API [6] was used for purposes of performing inferencing in the knowledge base (see Section 4). Missing information (i.e. the geolocation and homepage) about some venues in the ArtsHolland data was extracted from Foursquare to make the stored information more complete.

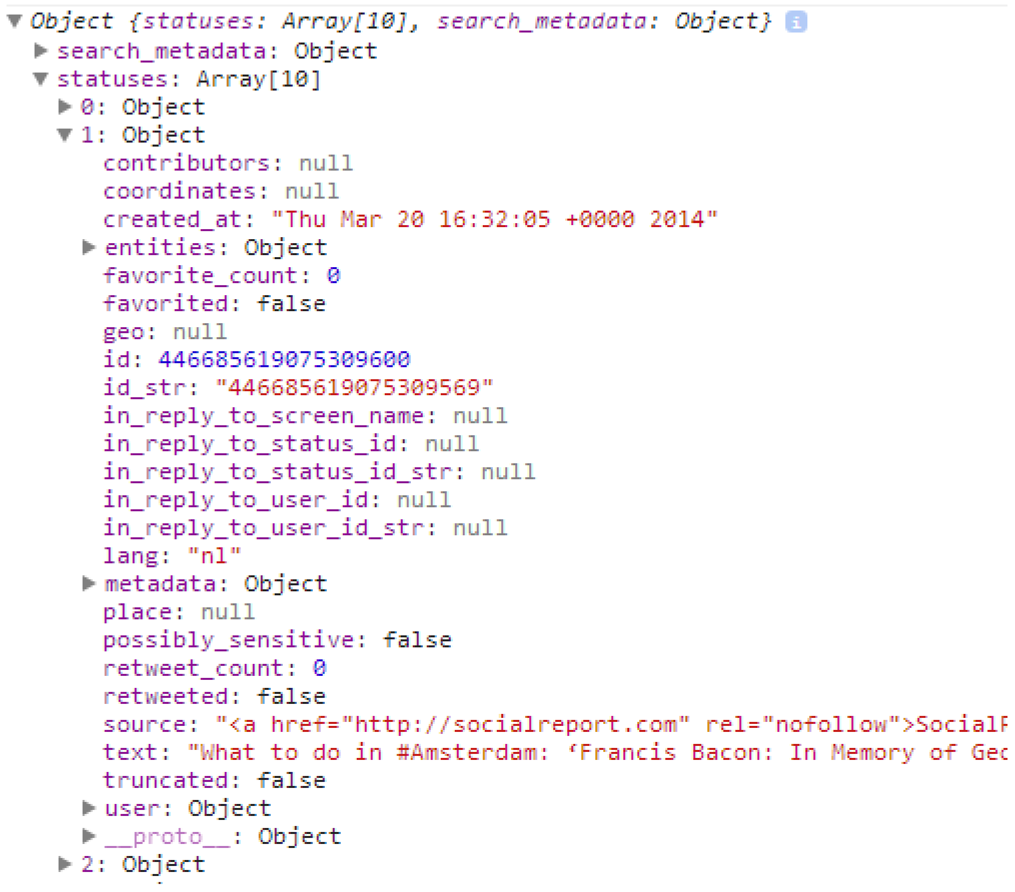
As an example, the following block of code represents data in Turtle format about the *Mirliton theater*:

```
fs:4cb441d775ebb60cc2eae0ad a fs:Venue;
      fs:title "Mirliton theater"@nl;
      geo:lat "52.0905950494175"^^xsd:float;
      geo:long "5.115401744842529"^^xsd:float;
      foaf:homepage <http://www.mirliton.nl>.
```

3.4 Twitter

Twitter's search API [5] is used to retrieve a mix of both recent and popular tweets about the venue being currently displayed (see Figure 6 in Section 2.3). When the user clicks on a venue marker on the map, the venue name and it's geocoordinates are extracted and used to query for tweets discussing the venue, giving preference to tweets created near the venue's location. Figure 16 shows an example response from the Twitter Search API.

Each tweet's id, text, user and date are extracted. The text, user and date are displayed to the user, and the id is utilized to construct a hyperlink to the original tweet.

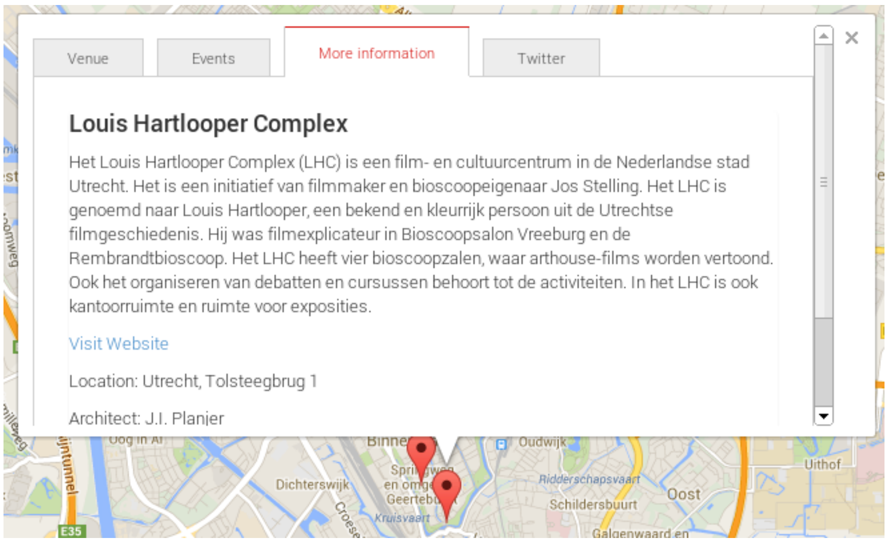


```
▼ Object {statuses: Array[10], search_metadata: Object}
  ▶ search_metadata: Object
  ▼ statuses: Array[10]
    ▶ 0: Object
    ▼ 1: Object
        contributors: null
        coordinates: null
        created_at: "Thu Mar 20 16:32:05 +0000 2014"
      ▶ entities: Object
        favorite_count: 0
        favorited: false
        geo: null
        id: 446685619075309600
        id_str: "446685619075309569"
        in_reply_to_screen_name: null
        in_reply_to_status_id: null
        in_reply_to_status_id_str: null
        in_reply_to_user_id: null
        in_reply_to_user_id_str: null
        lang: "nl"
      ▶ metadata: Object
        place: null
        possibly_sensitive: false
        retweet_count: 0
        retweeted: false
        source: "<a href="http://socialreport.com" rel="nofollow">Social
        text: "What to do in #Amsterdam: ‘Francis Bacon: In Memory of Ge
        truncated: false
      ▶ user: Object
      ▶ __proto__: Object
    ▶ 2: Object
```

Figure 16: Twitter's response for a query about *"De Nieuwe Kerk"*.

3.5 DBPedia

To further integrate the data obtained up until this point, the application exploits the great variety of information available on DBPedia. Every location hosting events located on the map contains, in the information window that opens on click, a section reserved specifically for this kind of data: when possible, further information about the location were obtained (such as name of the architect who designed it, its current purpose and so on); whenever this information was not available, generic information about the citythe location is situated in is shown.



Louis Hartlooper Complex

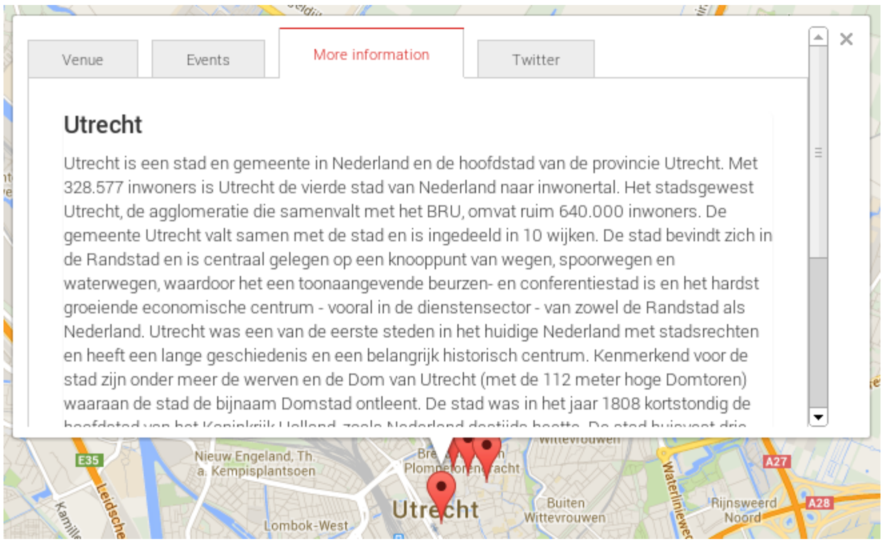
Het Louis Hartlooper Complex (LHC) is een film- en cultuurcentrum in de Nederlandse stad Utrecht. Het is een initiatief van filmmaker en bioscoopeigenaar Jos Stelling. Het LHC is genoemd naar Louis Hartlooper, een bekend en kleurrijk persoon uit de Utrechtse filmgeschiedenis. Hij was filmexplicateur in Bioscoopsalon Vreeburg en de Rembrandtbioscoop. Het LHC heeft vier bioscoopzalen, waar arthouse-films worden vertoond. Ook het organiseren van debatten en cursussen behoort tot de activiteiten. In het LHC is ook kantoorruimte en ruimte voor exposities.

Visit Website

Location: Utrecht, Tolsteegbrug 1

Architect: J.I. Planjer

Figure 17: Example of venue-related information from DBPedia.



Utrecht

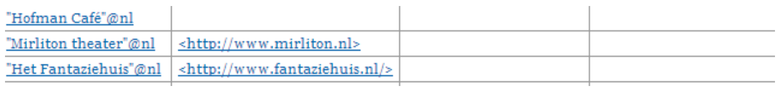
Utrecht is een stad en gemeente in Nederland en de hoofdstad van de provincie Utrecht. Met 328.577 inwoners is Utrecht de vierde stad van Nederland naar inwonertal. Het stadsgewest Utrecht, de agglomeratie die samenvalt met het BRU, omvat ruim 640.000 inwoners. De gemeente Utrecht valt samen met de stad en is ingedeeld in 10 wijken. De stad bevindt zich in de Randstad en is centraal gelegen op een knooppunt van wegen, spoorwegen en waterwegen, waardoor het een toonaangevende beurzen- en conferentiestad is en het hardst groeiende economische centrum - vooral in de dienstensector - van zowel de Randstad als Nederland. Utrecht was een van de eerste steden in het huidige Nederland met stadsrechten en heeft een lange geschiedenis en een belangrijk historisch centrum. Kenmerkend voor de stad zijn onder meer de werven en de Dom van Utrecht (met de 112 meter hoge Domtoren) waaraan de stad de bijnaam Domstad ontleent. De stad was in het jaar 1808 kortstondig de

Figure 18: Example of city-related information from DBPedia.

To be able to find a greater amount of information about Netherland-specific location, *nl.dbpedia.org* was used instead of the generic *dbpedia.org*.

4 Inferencing

As stated in Section 3.3, some of the venues in the ArtsHolland data was missing some information, which was crucial to the web-application. In concrete, some venues were missing the geocoordinates, and without them the venue cannot be displayed on the map, and by transition neither can the events associated to it. Figure 19 shows an excerpt of venue data. As it can be seen, a venue like the *Mirliton theater* is missing the latitude and longitude values.



"Hofman Café"@nl			
"Mirliton theater"@nl	<http://www.mirliton.nl>		
"Het Fantaziehuis"@nl	<http://www.fantaziehuis.nl/>		

Figure 19: Venues with missing data.

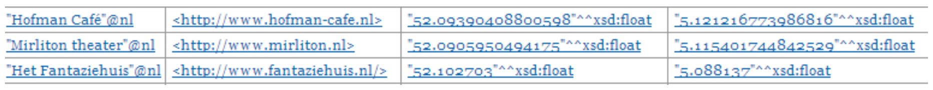
The following block of code represents data in Turtle format about the *Mirliton theater*, data that was extracted from Foursquare:

```
fs:4cb441d775ebb60cc2eae0ad a fs:Venue;
      fs:title "Mirliton theater"@nl;
      geo:lat "52.0905950494175"^^xsd:float;
      geo:long "5.115401744842529"^^xsd:float;
      foaf:homepage <http://www.mirliton.nl>.
```

These triple statetements in combination with an inferencing rule that determines that venues with the same homepage are in fact the same was then used as means to augment the AH data. The rule in question is shown in the following block of Turtle code:

```
foaf:homepage rdf:type owl:InverseFunctionalProperty .
```

Once the rule and the data from Foursquare are in the knowledge store, a second look at the data reveals more completeness, as seen in Figure 20



"Hofman Café"@nl	<http://www.hofman-cafe.nl>	"52.0939040880059 8"^^xsd:float	"5.121216773986816"^^xsd:float
"Mirliton theater"@nl	<http://www.mirliton.nl>	"52.0905950494175"^^xsd:float	"5.115401744842529"^^xsd:float
"Het Fantaziehuis"@nl	<http://www.fantaziehuis.nl/>	"52.102703"^^xsd:float	"5.088137"^^xsd:float

Figure 20: Venues with complete data.

5 Implementation considerations

5.1 Application Architecture

A three-tier application architecture model was following for the implementation of the web application. The general model is depicted in Figure 21.

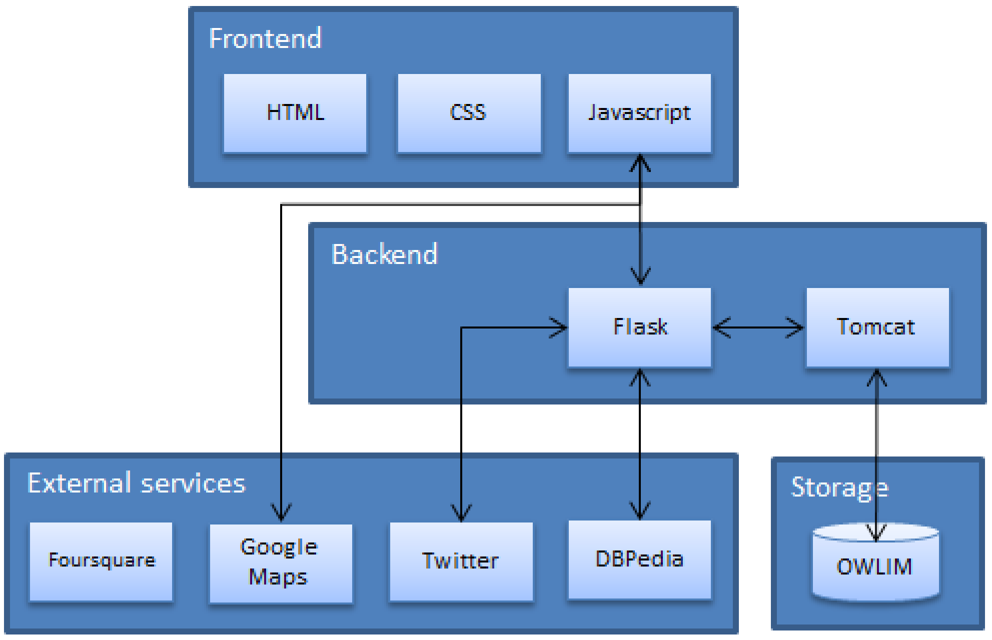


Figure 21: Architecture of getaLife.

The frontend was built using HTML, CSS and Javascript. The Javascript code queries *RESTfully* the Flask server in the backend, which is in charge of retrieving data from the OWLIM knowledge base, DBPedia and Twitter. The Javascript also communicates with the Google Maps Javascript API for geolocation related activities and map visualization. It's worth mentioning that OWLIM also allowed for the capability of providing our very own sparql endpoint.

The Foursquare external service is disconnected from everything due to some constraints in the retrieval of the data. See Section 5.4 for an explanation.

Some of the different software used by application are:

- Google Maps Javascript API v3 [18]
- JQuery 1.11.0 [12]
- Bootstrap 3.1.1 [9]
- Bootstrap Dialog [7]
- Bootstrap Dropdown Checkbox [10]
- Bootstrap Datepicker [8]
- Ion Tabs 1.0.2 [11]
- Python 2.7 [14]

- Flask 0.10 [15]
- OWLIM lite 5.4.6486 [16]
- Apache Tomcat 7.0.52 [17]

5.2 Getting data from ArtsHolland

Extracting data from the ArtsHolland Sparql endpoint [20] was quite problematic. The endpoint was quite unstable and was often down when data was needed for retrieval, in addition to frequently truncating the response to a query. In the end, queries with limits and offset were used to dump the data for Venues, Events, Production and Genres, but only for those entities that relate to events that are active beyond *2014-03-12*, anything before that (i.e. things that are not longer relevant), were of no interest.

Even though the data was dumped by querying the AH sparql endpoint, running the same query on the AH website itself yieled far more results. We are still baffled at this, and could not figure out why the reason for this was. Regardless, the (what seems limited) data that was managed to be dumped servers the purpose of being illustrative for the application's proof of concept.

The dumped data was loaded in a local knowledgebase to avoid all the mentioned problems, and to ease the activities relating to inferencing (see Section 4), as the data was in need of being augmented or curated.

5.3 Getting data from DBPedia

In obtaining data from DBPedia we encountered a quite relevant obstacle: the names of the venues as presented in ArtsHolland did not correspond to the ones offered in the DBPedia dataset. For example, the venue presented as "De Nieuwe Kerk" from ArtsHolland, is found on DBPedia as "Nieuwe Kerk"; as such, when querying the dataset for venues, this venue is simply catalogued as *not found* and ignored.

In a future, more advanced implementation of this web application it would be nice to be able to automate the search for partial names, using SPARQL filtering function and regulare expression for string matching.

5.4 Getting data from Foursquare

The data extracted from Foursquare (see Section 3.3) was retrieved in a manual fashion, as the names for the venues in the ArtsHolland data did not necessarily match the name that the venue had in Foursquare, making it really difficult to automate the process, as doing so would likely leave the data with some strange results. To circumvent this, the queries to Foursquare and the subsequent data extraction was done manually and with much care to find the appropriate venues.

6 Future work

Before considering expanding the web application, it must be pointed out that the ArtsHolland dataset is quite outdated and not so easily manageable as initially thought. A nice objective for the future should be to find a way to keep the ArtsHolland dataset continuously updated, because as it has been shown with this project, it can be source of inspiration for useful applications.

Nonetheless, in the following paragraphs are presented some nice idea to expand the *getaLife* application.

Adding transportations information: once the current location of the user and the selected events are known, it would be quite easy to integrate a functionality that allows the user to choose a preferred transoportation mean, and show him the correct route to arrive to their destination. Such information is already partially available in ArtsHolland (e.g. Tram lines that connect you to the chosen event), so that could be a good point to start developing this new feature.

Exporting the events into calendars: an interesting possible improvement could be the exporting of the events in a standard format (for example *iCal*) or to specific applications, such as *Google Calendar*. This extraction could be done automatically, as a subscription service. This way a user could be constantly updated about his preferred events by just checking his calendar.

Studying trends: it can be useful to see, as time goes by, how certain event typologies become more or less popular than others. This kind of knowledge can of course be used for marketing purposes, but can also be helpful for policy making. The use of the map as a central element in the exploration of data, combined with graphic representation of the events' data in their several dimensions, can for example facilitate the observation that in a certain part of a city there is lack of cultural offer. Counteracting this tendence may improve the chances of establishing a sense of community in socially disgregated suburbs.

7 Conclusions

Despite the fact that the application fulfills all the initial requirements, as we pointed out in the previous section there is always room for improvement. Even though Semantic Web proved itself to be a great tool with which to handle data and knowledge, many are the problems that a developer can face, first and most important of all the lack of data and the different representation for the same entities chosen by different centralized datasets, such as DBPedia, ArtsHolland or Foursquare.

Another obstacle can be found in the fact that not all data centers use rdf notation to represent their own data, so a lot of different representations are available on the entirety of the World Wide Web.

Regardless of these difficulties, using Open Data allows for great accessibility of data, increasing enormously the potential of web applications, and guarantees for data reusability and recombination to obtain continuously new information. In conclusion, working with Semantic Web was certainly a way to understand why such a notation important and should be exploited to its full potential.

Note: a link to a functioning version of the web aplication can be found at `http://54.186.171.129:5000/`

References

[1] Google Maps JavaScript API v3, *Places Library* https://developers.google.com/maps/documentation/javascript/places, Visited: 18-03-2014

[2] Google Maps JavaScript API v3, *Geocoding Service* https://developers.google.com/maps/documentation/javascript/geocoding, Visited: 18-03-2014

[3] Google Maps JavaScript API v3, *Reverse Geocoding* https://developers.google.com/maps/documentation/javascript/geocoding#ReverseGeocoding, Visited: 18-03-2014

[4] W3Schools, *HTML5 Geolocation*, http://www.w3schools.com/html/html5_geolocation.asp, Visited: 20-03-2014

[5] Twitter, *Search API* https://dev.twitter.com/docs/using-search, Visited: 21-03-2014

[6] Foursquare, *Search Venues* https://developer.foursquare.com/docs/venues/search, Visited: 21-03-2014

[7] Bootstrap Dialog, http://nakupanda.github.io/bootstrap3-dialog/, Visited: 21-03-2014

[8] Datepicker for Bootstrap, http://www.eyecon.ro/bootstrap-datepicker/, Visited: 21-03-2014

[9] Bootstrap, http://getbootstrap.com/, Visited: 21-03-2014

[10] Bootstrap Dropdown Checkbox, https://github.com/Acquisio/bootstrap-dropdown-checkbox, Visited: 21-03-2014

[11] Ion.Tabs, https://github.com/IonDen/ion.tabs, Visited: 21-03-2014

[12] jQuery, http://jquery.com/, Visited: 21-03-2014

[13] D3, http://d3js.org/, Visited: 21-03-2014

[14] Python 2.7, https://www.python.org/download/releases/2.7/, Visited: 21-03-2014

[15] Flask 0.10, http://flask.pocoo.org/, Visited: 21-03-2014

[16] OWLIM lite 5.4.6486, https://www.ontotext.com/owlim, Visited: 21-03-2014

[17] Apache Tomcat 7.0.52, http://tomcat.apache.org/, Visited: 21-03-2014

[18] Google Maps Javascript API v3, https://developers.google.com/maps/documentation/javascript/, Visited: 21-03-2014

[19] Arts Holland Platform, http://dev.artsholland.com/, Visited: 21-03-2014

[20] Arts Holland Sparql Endpoint, http://api.artsholland.com/sparql, Visited: 21-03-2014