

# Rationale Final Application 06

Our main objectives for this application were to enrich the information that you retrieve from searching the Amsterdam Museum 'API' and to combine historical views with contemporary ones. Our target audience is anyone who is familiar with social media and its usage and wants to combine this with the Amsterdam Museum experience. This application is developed for the web, however it also works on tablets (might be a little slower than the laptop experience).

To enrich the information from Amsterdam Museum we came up with the idea to use social media APIs and combine these with the search results/query of the Amsterdam Museum input.

We started implementing the APIs one by one. This way we would always have a working application which we could expand in the way we wanted.

## **APIs | why and how?**

The implemented APIs work together with the Amsterdam Museum API to enrich the information about your search topic. For instance if you were to search for "rembrandtplein", twitter will show you tweets about rembrandtplein, google maps will show you the location on the map with the googlemaps marker, Flickr will show you additional images involving rembrandtplein and Amsterdam Museum will return the artworks involving this query. Basically the application revolves around the results that you get from the Amsterdam Museum API. By clicking on one of these results, the rest of the APIs get updated instantly and the updated information is shown.

## **Rationale per API**

### **Amsterdam Museum:**

The Amsterdam Museum API is probably the most important one since the rest of the APIs enrich the information from this API. When a user searches, the artworks from Amsterdam Museum are shown for that query. After this the other APIs react to selection of the search results.

### **Twitter:**

We decided to use twitter for the short messages and updates that it provides around a certain topic. When the user enters a query, this query is also parsed by the twitter API and we made it so that it will show you the most recent 15 tweets around that query. When you click on one of the tweets, for instance if you find an interesting user, the 15 most recent tweets of that user are loaded.

### **Google Maps:**

The Google Maps API serves the purpose of adding a location and a map to your search. For instance if you were to search for "leidseplein". This will show up on the map. At the same time the Amsterdam Museum (and all other) API(s) was updated. When clicking on one of the Amsterdam Museum results, the location property of this result gets parsed into the Google Maps API and is shown on the map. However if there is no location property in the selected result, the Google Maps component will show you an 'Artwork has no location' exception in the title. All the locations pin-pointed on the map are saved.

This means that when you do a new search, the previous location is still pinned on the map (you would probably need to zoom out to see the previous location).

### **Flickr:**

Flickr is a very nice resource to add additional (contemporary) image results to a location or an artwork. When an artwork is clicked, the location of the artwork is also parsed through the Flickr API to show additional images around that location (the location property from the selected Amsterdam Museum result). However sadly it occurs often that there is no location property in the Amsterdam Museum result. Our solution to this problem was to check for the location property initially and if this is not available, we parse the title property of the result to show additional Flickr images for that title. When the user clicks on a Flickr result/thumbnail, this result is shown in a new window (a bigger picture is popped up).

### **Design Decisions:**

The interface is built up out of windows which can be minimized, maximized, resized, dragged and reset (with: 'reset windows' button). The Amsterdam Museum component cannot be customized. This way we show the user that the rest of the windows revolve around the Amsterdam Museum component.

By giving the user freedom to drag and drop, resize and max-minimize windows to shift focus the user will be able to customize their interface. However this could lead to a chaotic interface that just looks messy with too many overlapping windows. We came up with the "Reset Windows" button to organize all the windows on the screen. Additional popped out Flickr windows are organized next to each other along with the rest of the components. These Flickr windows can also be closed by clicking on the X.

### **Bottlenecks/Problems:**

- The Amsterdam Museum results often have no location property. This is a shame because then the location cannot be shown on the Google Maps API and no additional Flickr images will be shown for this location. By parsing the artworks title into Flickr we can get some results in the Flickr API, however this is also not always the case.
- The Google API key does not work when you publish the application on a server. We need a different domain-bound API key for that. The local API key does work.
- Flex does not allow services to transfer data to the application without a crossdomain security policy. This is basically an .xml file containing the security rules for the service used in the application, in this case we are talking about the twitter service. When running locally this is also not a problem.
- The application runs slow when running on a mobile device. It would be nice if we could turn this into a mobile application as well.
- We have no name for the application yet.

The server problems can be fixed, however we would be happy with some suggestions for the name. :)

### **Improvements:**

Some useful tips we got:

- To make the Amsterdam Museum component more prominent and in that way showing the user that this is the main focus.
- By using # tags for the parsing of the query in Twitter, more relevant information can

me shown. We had thought about this, however we decided not to do this yet because we think that many twitter users do not use # tags in their tweets, these tweets would then not be shown but could still be relevant or interesting. We might change this in the future, for now we have chosen not to use the # tag. The easy solution for now is to just write the hashtag into the searchbox.