Web technologies: Report

Youssef Boudiba, Thibaut Deweert, Geatan Boey, Linda de Corte

Contents

1 Introduction

You may know the game "Geoguessr", where you get a random location of google streetview and you have to guess where you are, and the closer your guess is the more points you get. Our game has the same idea of guessing a location on a map. Only in our game everyone can upload a picture with a location and an hint. Other people can then guess where the picture is made. They can guess 3 times, every time they lose 20 points and when they ask for an hint they lose half the points.

1.1 installation of the project

1. Install MongoDB `https://www.mongodb.com/download-center?jmp=nav#community`
2. Install node.js `https://nodejs.org/en/`
3. Install Sails.js `http://sailsjs.com/get-started`
   npm -g install sails
4. Start MongoDB service (make sure you point to the database path in the project folder: /data/db)
   mongod –dbpath [Path-to-Project]/data/db
5. Navigate to the project folder and start the Sails.js service
6. The project should be running, visit `http://localhost:1337/`

2 the web application architecture

For this application we chose to use the framework Sails.js. This is a MVC framework for node.js.

- Node.js, an execution environment for event-driven server-siede and networking application
- front-end: For the front-end we chose to include Angular.js for this, because it is a very popular JavaScript framework, which extends HTML to make it more dynamic.
- Database: we chose MongoDB, this is a a very big noSQL database services. We chose to use it because it is flexible and is easy to use.

We chose Sails.js framework because it lets us write everything in the same language and takes over some tasks, for example it automatically sets up an API.

3 Implemented functionalities

3.1 User

3.1.1 The user

3.1.2 registering

3.1.3 edit user

3.1.4 User profile page

3.1.5 quizes on user profile

3.2 Quiz

3.2.1 overview page

3.2.2 filtering

Quizzes can be filtered on location. On the top of the quiz overview page a user can fill in a location and press the ?filter? button. Only quizzes that are within a 30 km range of the given location are then displayed on the page. It works as follows. Upon pressing the ?filter? button the text from the input field gets transformed to coordinates. This is achieved by using the Google Maps Geocoder which can transform addresses to coordinates. This set of coordinates is then compared with every quiz. For each quiz the distance is calculated between the two, if the distance is shorter then 30 km the corresponding quiz is shown. All this happens on the client side. If the typed in location is not found, an alert message indicates so. To remove the filter, the user can simply press the ?X?.

3.2.3 creating a quiz

Creating a new quiz can be done by clicking the ?Add? button on the quiz overview page. On the popup that opens the user can fill in some information about his photo quiz. The user can add a personal touch to the quiz by adding a custom message. Eg: ?Visited this museum during my last vacation?. There?s also the option to provide a hint. The player of the quiz can use this hint if he or she has no clue about the location. (See X.X) The message and hint are optional fields. The middle field can be used for uploading a picture and is mandatory. Currently it only supports jpeg images. We use HTML5 form validation on the client side to comply with this restriction. (See X.X). After selecting an image and pressing ?next? HTML5 FormData is used to transfer the image file to the server side. The server first reads the exif data from the image and searches for GPS coordinates. The image file is then uploaded to Imgur - an online image host - via the Imgur API. Imgur automatically deletes all exif data, so it is not possible by the players of the game to find the GPS coordinates of die image. When the image is uploaded to Imgur, we save the

URL and the delete hash (see X.X) of the image. All this information is stored in the quiz database.
On the client side we now get a second popup showing a map with a marker. If the uploaded image included some GPS coordinates then the marker wil be on that position. The user has then the possibility to move the marker to a correct position, if that was not the case yet. If the image had no GPS coordinates in then the user can simply indicate the correct position with the marker.

3.2.4 individual quiz

3.2.5 comments

3.2.6 search

4 the other requirements

4.1 AJAX

4.2 HTML 5

We have used 4 HTML features in our application

1. GEO-location, we use this one when you are doing a quiz, it will first center the map on your location.
2. local storage, we use this to store the API secret on the users computer. So they will stay logged in.
3. form tags, we have used this when you add a new quiz, for the file upload.
4.

4.3 Web service

4.3.1 Facebook

4.3.2 Imgur

For storing the all the images used for the quizzes we used Imgur (http://imgur.com). Using there public available API we can send the images. As return we get a JSON object containing information about the uploaded picture. For this project we only use the link, id and deletehash parameters.

4.4 Google maps

We have used Google maps on several occasions. The first place where we use the google-maps API is when we are filtering based on location. The location the user fills in is send to Google. Google than sends back the gps locations of the location that matches closest with what the user filled in.

The next place we use the Google maps API is when a user is adding a picture. We first look if a picture has Exif location data in his or her application. If they have we make a google map with a marker in the center of the location. Otherwise we center on the VUB. The user can than move the marker. when they save we ask the Google maps API to give us the location.

And finally we use it when people guess the location, we again make a map with an marker, but this one is based on your GEO-location, if you do not want to share it we will use the location of the VUB for the start point. You can then again move the marker to chose the location. When you think you found the location, you save and we will ask the Google Maps API for the location. We will than do a calculation to see if you were close enough. If not you have 3 chances to guess, again by moving the marker.

4.5 Provide data

5 conclusion