Interaction in Smart Cities

HCI 4 Assessed Coursework

Contributors:

Petar Yordanov (1103620)

Silviya Sotirova (1103571)

28.11.2014

Introduction

The main idea of this project is to promote the use of technology in the city. People nowadays have unlimited access to technology and carry around multiple devices in order to use them for communication, entertainment, etc. Future Cities is a great initiative that appeared in the UK aiming to improve city life by solving lifestyle issues or even revolutionizing the way citizens perceive their urban surroundings. Sunny Glasgow is based on the idea that technology is an integral part of our everyday life and we can use it to improve our appreciation of the city we live in and nature in general. Following the Internet of Things (I-o-T) paradigm, it aims to deliver an interactive user experience and stimulate citizens to use their mobile devices in order to capture photos from different locations and contrasting weather conditions around the city of Glasgow and then upload them to a centralized server where the data is processed and aggregated. This will not only improve citizens' appreciation of sunny days in the city, but it will also foster tourism and will attract more young people as a living destination.

Planning

The initial idea was to create something that everyone could use on a daily basis and will keep its users in the long term.

Glasgow is a city with a big population and people use their mobile devices to browse the Internet through Wi-Fi networks all around the city. What is more, the weather in Scotland is known to be rainy for the most part throughout the year and bright-shining sun is a rarity. So it was decided that the application is going to be targeted at people who live in the city (in any age group) and mainly aiming to entertain while improving the citizens' appreciation of the city's architecture and scenery through the use of technology.

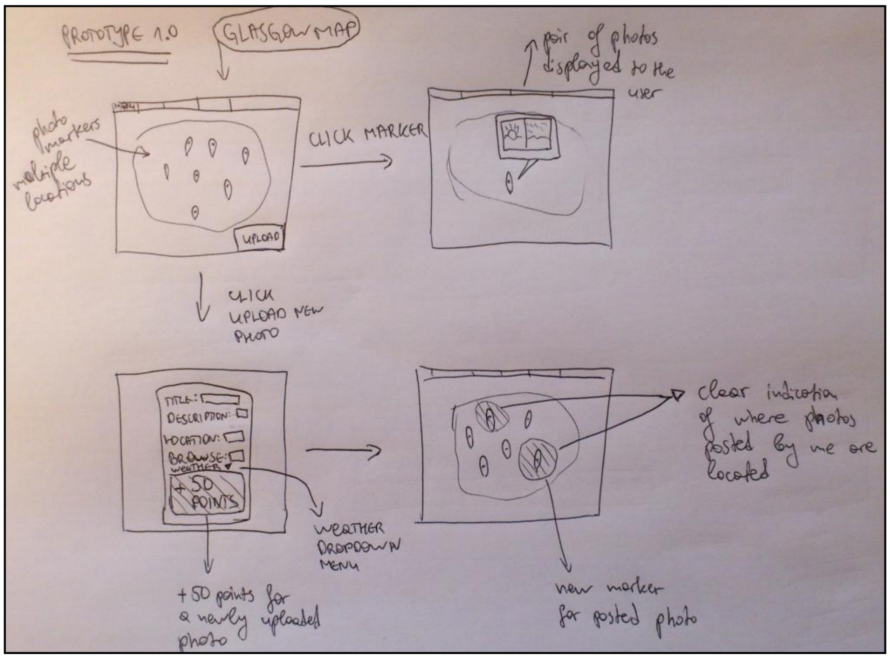
The initial decision process regarding system architecture involved identifying the main system components so that we could start building together a general framework. Going mobile was part of the plan from the start as after a couple of discussions and continuous evaluation we came to the conclusion that this would potentially attract much more users and make the application significantly more flexible in terms of system/technical requirements as it gives users an option.

The system needed a processing engine as well as a simple, easy to install and launch client application that would enable potential users to maximize the experience quality while using the website.

Design

The initial planning process was followed by numerous draft paper prototypes that helped in the process of incrementally shaping the final look of the application. We had multiple iterative versions of the same application that were continuously evaluated and improved during the weekly meetings.

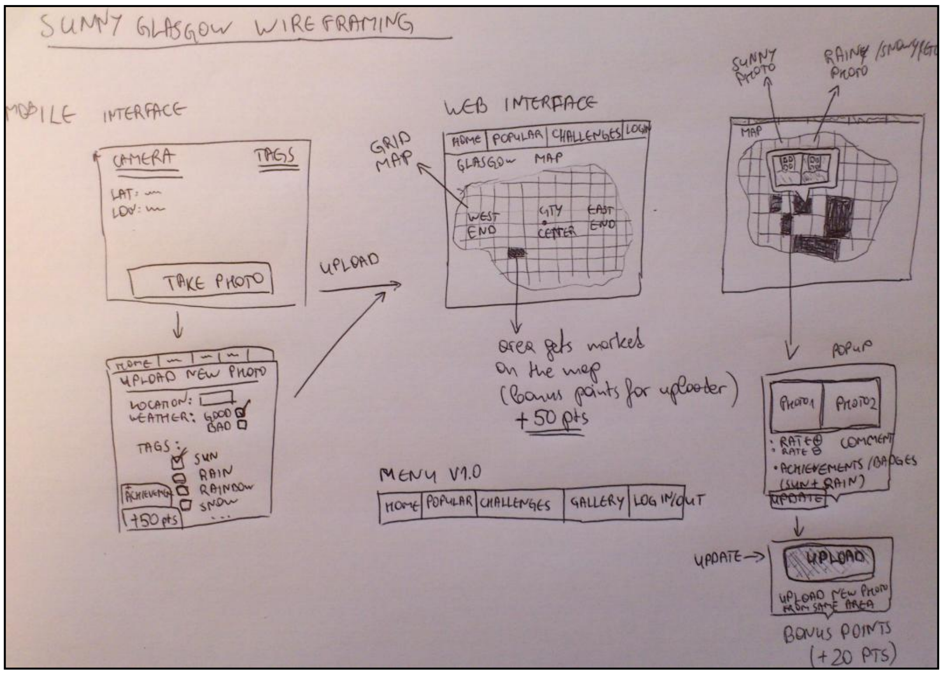
One of the initial paper prototypes can be seen below:


PROTOTYPE 1.0
GLASGOW MAP
photo markers multiple locations
MENU
UPLOAD
CLICK MARKER
pair of photos displayed to the user
CLICK UPLOAD NEW PHOTO
TITLE:
DESCRIPTION:
LOCATION:
BROWSE:
WEATHER
+50 POINTS
WEATHER DROPDOWN MENU
+50 points for a newly uploaded photo
clear indication of where photos posted by me are located
new marker for posted photo

Prototype 1.0 1

There are multiple interaction views/ screens on this initial prototype. As clearly seen on the top left, there is a map, showcasing user uploaded photos in the form of markers with on-click pop-ups displaying photos and their metadata (title, description, weather type, etc). On the bottom left there is a dialog popup box that appears when a user wants to upload a new photo. In order to stimulate potential users of the application to keep adding/updating images, there is a point-based system as a form of incentive (example, each transaction increases profile points by 50).

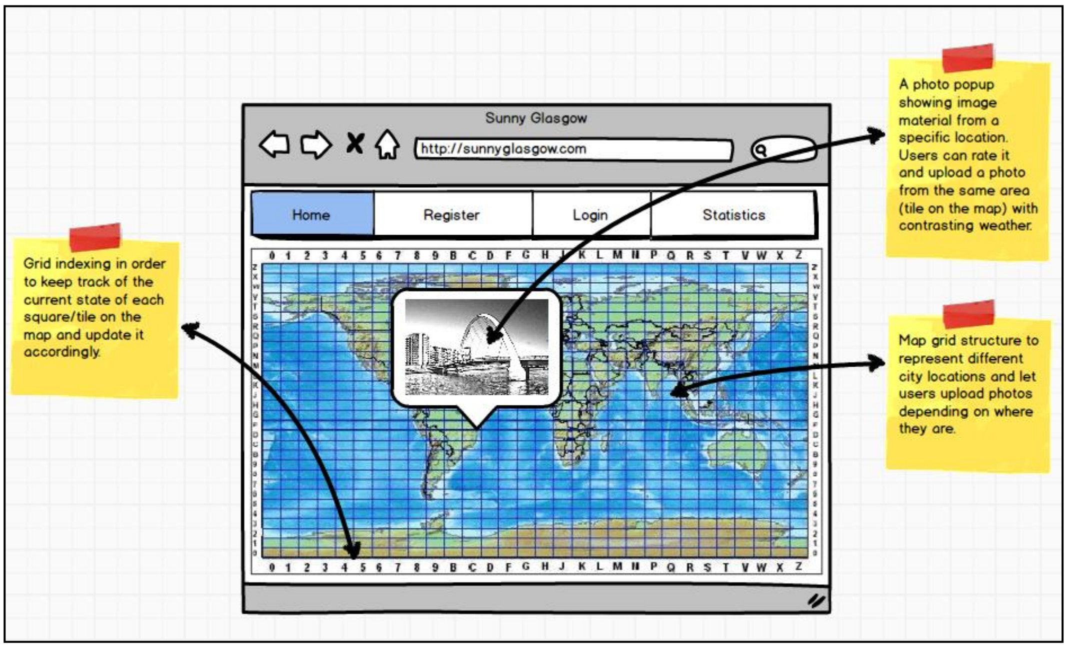
After a brief prototype evaluation some improvements were made in v1.0.2 regarding data visualization:



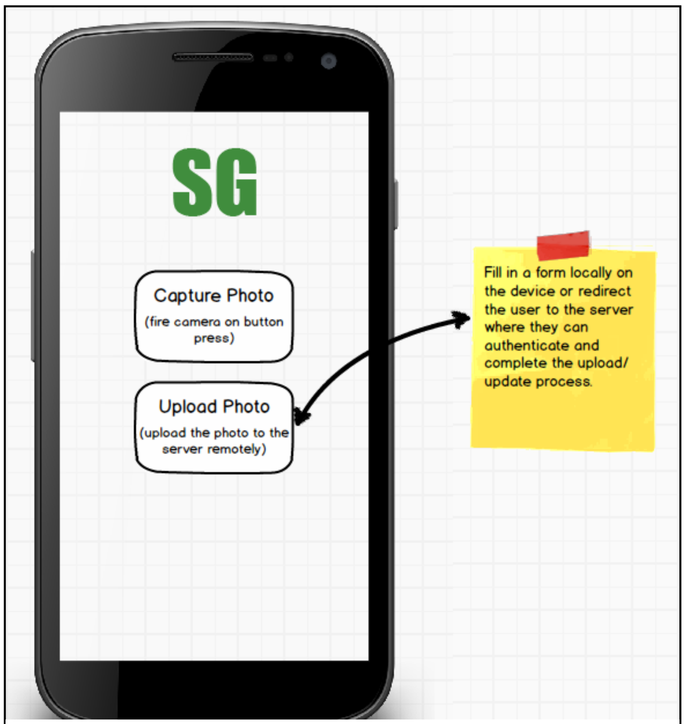
Prototype 1.0 2

This prototype of the application builds on the basic framework from version 1.0.1. The main difference here is the visualization style. Instead of only showing image simple metadata such as title and description, there are also additional options that are considered – ratings, comments, achievements (for example, certain weather conditions, such as a double rainbow or hail, which are very rare), badges (denoting user status/ rank in order to group/distinguish users and possibly give them different privileges), etc. There is also a grid structure visible on the map. Prototype 1.0.1 is showing user exact location but does not display the correlations and distance between photos on the map while the grid is a consistent structure that can be used throughout the whole working space to render tiles on the map that can get updated with data over time. Also, on the top right there is an indication that data-populated grid tiles will be color-coded so that users know how many photos and where on the map they are at any moment.

Below is a digital representation of prototype 1.0.2 created using the Balsamiq mockup tool:


Sunny Glasgow
http://sunnyglasgow.com
Home
Register
Login
Statistics
Grid indexing in order to keep track of the current state of each square/tile on the map and update it accordingly.
A photo popup showing image material from a specific location. Users can rate it and upload a photo from the same area (tile on the map) with contrasting weather.
Map grid structure to represent different city locations and let users upload photos depending on where they are.

Prototype 1.0 2.1

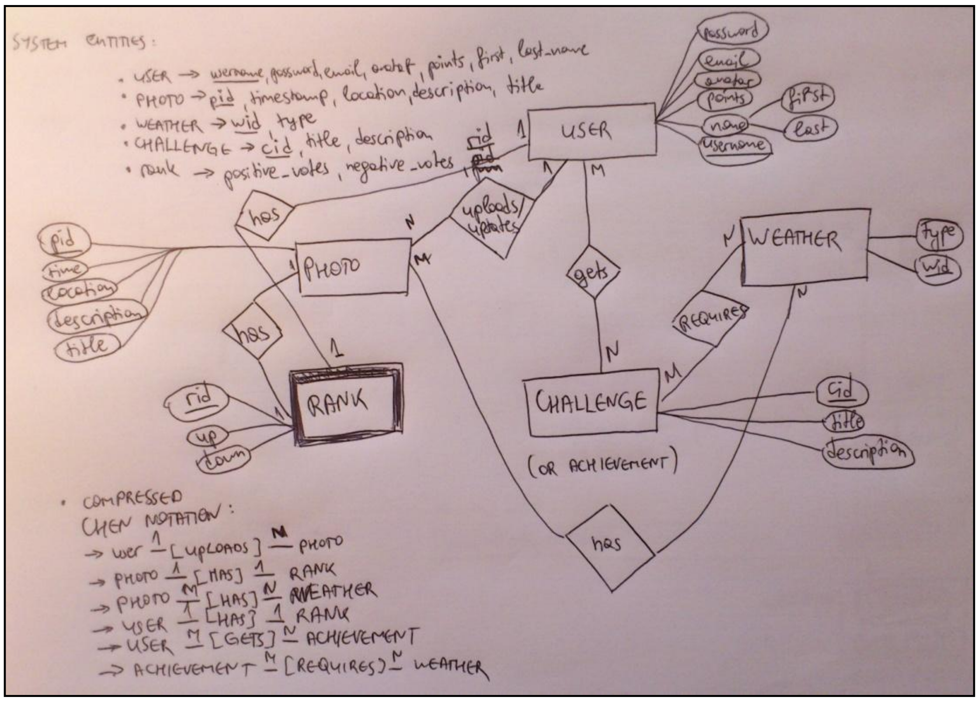

SG
Capture Photo
(fire camera on button press)
Upload Photo
(upload the photo to the server remotely)
Fill in a form locally on the device or redirect the user to the server where they can authenticate and complete the upload/ update process.

Client Prototype 1.0 1

The Android application prototype was also starting to take shape by this point(image on the left).

Having a skeleton structure up front, the next step was to start designing separate system components, such as the data store, in more detail. An entity-relationship diagram as well as an architectural framework diagram was constructed. Those were incrementally updated during the development process.

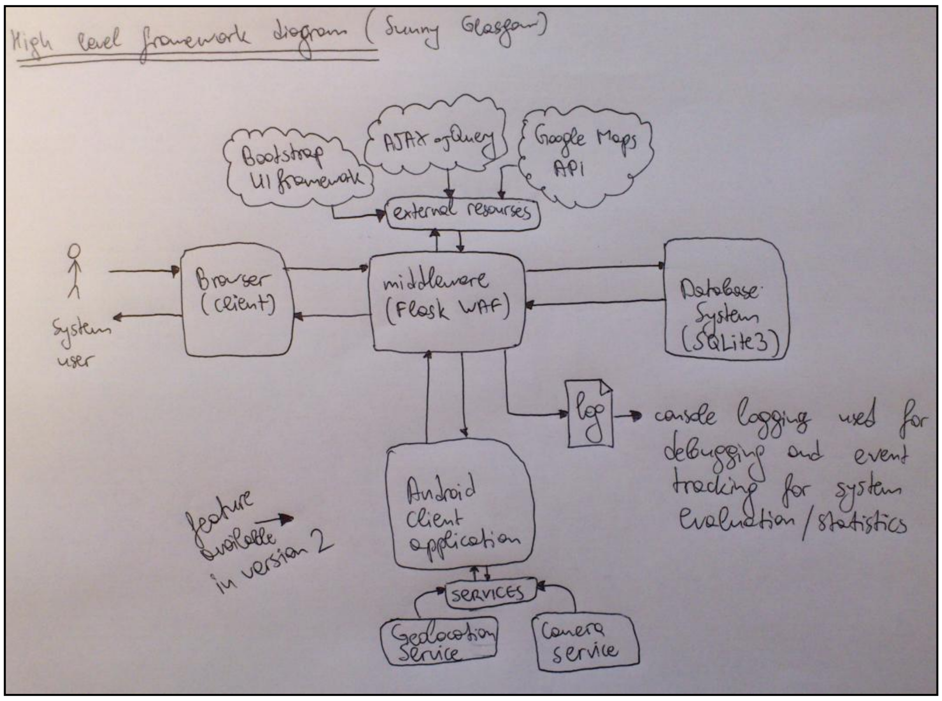
Entity structure changes were introduced on the go. For instance, less relational complexity would be imposed by more attributes associated with the same entity, but this would violate the ACID properties, more specifically, model atomicity – all of those considerations and trade-offs were taken into account while designing the SQL schema model. Compressed Chen notation was also used to optimize the design process. Here is an advanced ER diagram:


SYSTEM ENTITIES:
• USER → username, password, email, avatar, points, first, last_name
• PHOTO → pid, timestamp, location, description, title
• WEATHER → wid, type
• CHALLENGE → cid, title, description
• rank → positive_votes, negative_votes, rid
USER
password
email
avatar
points
name
first
last
username
PHOTO
pid
time
location
description
title
has
uploads/updates
gets
WEATHER
type
wid
REQUIRES
RANK
rid
up
down
CHALLENGE
(OR ACHIEVEMENT)
cid
title
description
has
1
N
M
• COMPRESSED CHEN NOTATION:
→ user 1 [UPLOADS] M PHOTO
→ PHOTO 1 [HAS] 1 RANK
→ PHOTO M [HAS] N WEATHER
→ USER 1 [HAS] 1 RANK
→ USER M [GETS] N ACHIEVEMENT
→ ACHIEVEMENT M [REQUIRES] N WEATHER

ER Design 1.0.1

The framework design was developed and evaluated synchronously with the data store planning process. The initial idea was to have a RESTful API for the backend views, which in turn were going to directly communicate with the DBMS via session queries. The data from the views would then be serialized to JSON and parsed on client-side. The other option was to pull more of the processing on the server side to reduce the client load as JavaScript would need system/network resources to dynamically render data on the map. Both approaches have their pros and cons: generally, introducing a RESTful API leads to consistency and is much easier to scale as it involves a simple change to the controller (main python class in this case). However, data can be processed much more efficiently on server side as it has direct linkage to the data store. On the other hand, filtering/preprocessing on the server means less data gets transmitted over the network via a single request on average, which potentially causes a rise in the number of requests. At the end, the second approach was chosen as it proved to be better in terms of speed as development progressed.

An advanced version of the high-level system architectural diagram can be seen below:


High level framework diagram (Sunny Glasgow)
Bootstrap UI framework
AJAX + jQuery
Google Maps API
external resourses
System user
Browser (client)
middleware (Flask WAF)
Database System (SQLite3)
log
console logging used for debugging and event tracking for system evaluation / statistics
Android client application
feature available in version 2
SERVICES
Geolocation Service
Camera service

Framework 1

Technologies

The tools that were used for during the development process closely reflect the initial planning outline and were chosen after thorough research of multiple options. The final prototype implementation makes use of the following technologies:

- ***Server development***:
  - Flask WAF (web application framework): neither of us had used this micro-framework for webapp development before, but after some research it was decided that it would be most suitable for this project due to its lightweight structure, short learning curve and relatively good documentation on the official web page; the web application framework was compared against other python-based development frameworks such as Django and Pylons and the main advantage it was concluded to have against them is that it is more compact and suitable for rapid development purposes
  - SQLite 3 DB: this database management system was chosen after it was compared against a few other DB engines on the following criteria: configuration requirements (time), scalability, flexibility; at the start of the development process it was not clear how much data we are going to be able to generate and store in the database so scalability and flexibility were of great importance; as it became clear later, the models for the project changed/ were improved a couple of times and the relational local storage SQLite made it very easy to re-initialize and re-populate the database with sample data on demand; other database systems such as MySQL or MongoDB (NoSQL) would not be as suitable for this incremental prototyping process due to their time-consuming configuration and poor library support (PyMongo; in contrast, Flask's SQLAlchemy is a great library for ORM /object relational mapping/ that has detailed documentation on the main page of the framework)

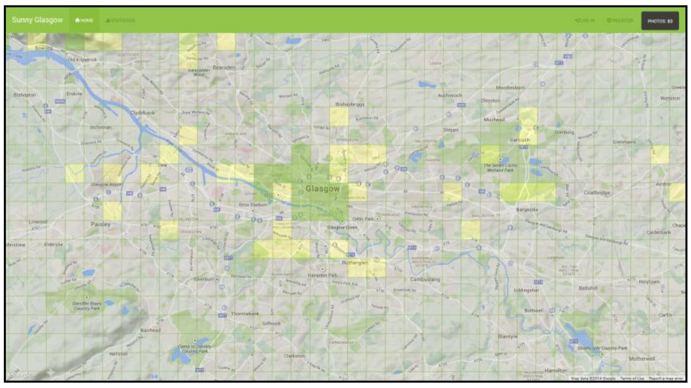
  - Google Maps API v3 – the JavaScript API is a key technology for this system as it provides all the necessary tools for dynamic map rendering that paired with jQuery helped to achieve an immersive user experience; other researched options included OSMatrix and d3.js but they were not as easily configurable as Google Maps
  - Twitter Bootstrap 3 – the CSS framework was chosen in order to achieve an attractive graphical user interface and ensure that style consistency is maintained throughout the whole web application
  - jQuery 2.1.1 – JavaScript event handling

- ***Client development*** (Android and Windows Phone):
  - NodeJS (v0.10.33) in order to build PhoneGap projects
  - PhoneGap 3.6.3 – serving application builds and native code compilation before device testing and evaluation
  - Android SDK - used in the Android application prototyping:
    - ADB (Android Debug Bridge) – in order to run the client on real devices and continuously test newly integrated features
    - AVD (Android Virtual Device) – used for simulations in the initial development stages as well as for user evaluation in the final agile prototyping sprint
  - Apache Ant (v1.9.4) for project management via PhoneGap
  - jQuery Mobile and CSS – used for background event management in JavaScript and styling respectively

Implementation

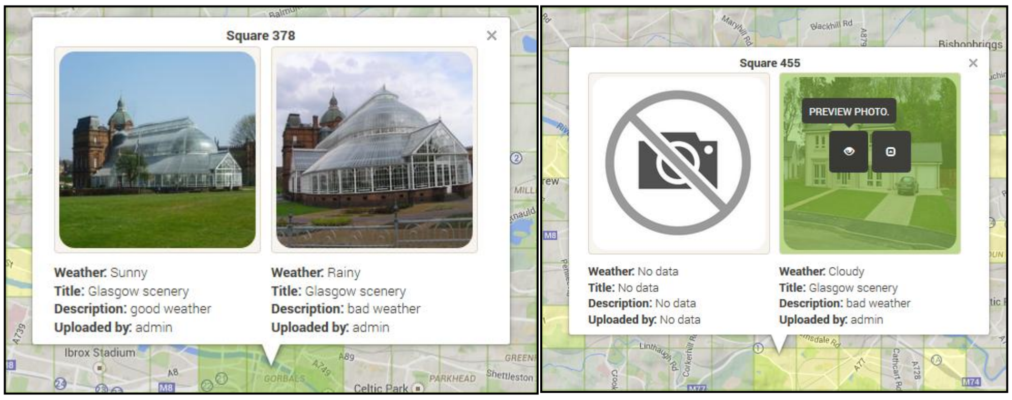
This section shows the main implementation components that were continuously evaluated and optimized during development.

The final prototype implementation of the home page follows the structure of the user interface prototype. There is a color-coded indexed grid structure on the map (green- populated tile; yellow- single photo space available) with each square having an integer id:



GUI 1

You can click on the map tiles as well as zoom in and out on the map in order to browse through the photos. Hovering above photo thumbnails lets users preview or update them. Here are examples of how the square content looks like:



Square 378

Weather: Sunny
Title: Glasgow scenery
Description: good weather
Uploaded by: admin

Weather: Rainy
Title: Glasgow scenery
Description: bad weather
Uploaded by: admin

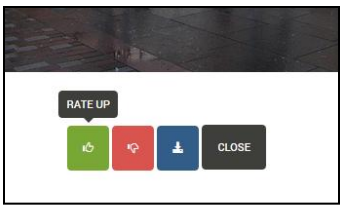
Square 455

Weather: No data
Title: No data
Description: No data
Uploaded by: No data

Weather: Cloudy
Title: Glasgow scenery
Description: bad weather
Uploaded by: admin

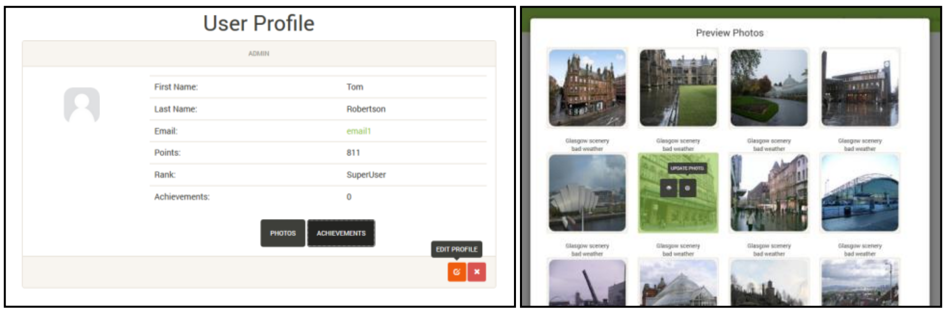
GUI 2

Previewing the photo lets a user rate it or download the full-size image (ratings are mocked up):



GUI 3

Once you login you can see your user information, update/upload image material, as well as well as preview your photos on the dashboard page:



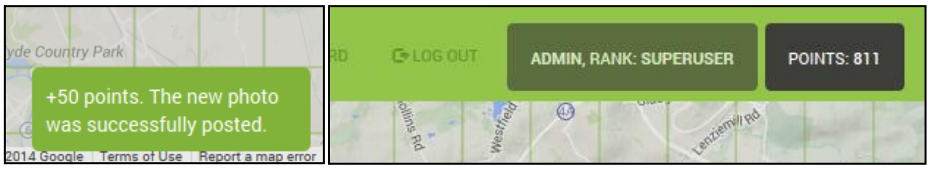
User Profile

ADMIN

First Name:	Tom
Last Name:	Robertson
Email:	email1
Points:	811
Rank:	SuperUser
Achievements:	0

GUI 4

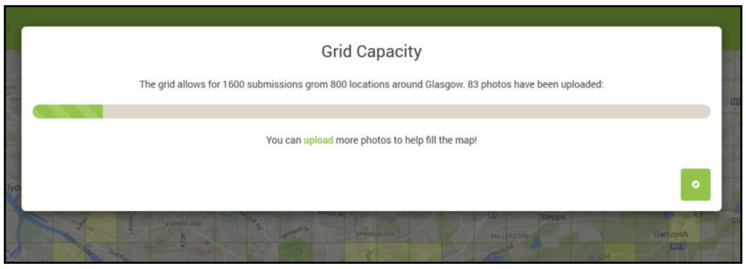
When you start uploading/ updating photos, you receive points (adding a new photo: 50, updating: 20 points; jQuery notification bottom left) and you can keep track of your rank at the top right of the page:



+50 points. The new photo was successfully posted.

GUI 5

The current map capacity can be checked at any time by clicking the button “Photos” (visible for non-registered users). It shows a progress bar as seen below:

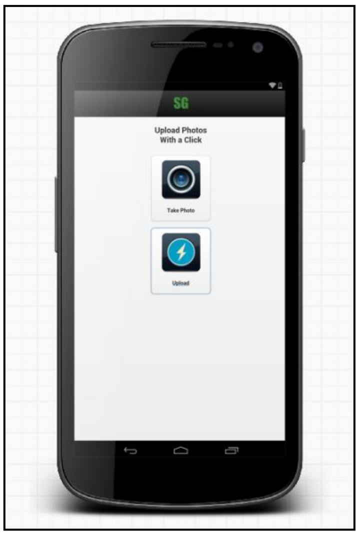

Grid Capacity

The grid allows for 1600 submissions grom 800 locations around Glasgow. 83 photos have been uploaded:

You can upload more photos to help fill the map!

GUI 6

The final Android application can also be seen below. It enables users to take photos (camera button) and directs them to the web server page where they can proceed to log in and upload them (“Upload” button):



GUI 7

Evaluation

1)Executive Summary

A number of techniques were used in the evaluation of the system. The primary approaches we used were Think alouds in laboratory settings as well as a distributed questionnaire.

The questionnaire included questions to navigate the user through the application, small tasks to make the users interact in detail with the system and questions for their point of view.

The evaluation team made critical contributions to the evaluation process, both in terms of their insights into the effectiveness of program activities and the assistance they provided us – the evaluators, in carrying out the evaluation.

2) Evaluation Goals

The primary goal of this evaluation was to measure how easy it is to use the application and whether it is going to be interesting for the end users - a summative evaluation. This assessment was done by sending the survey to people that are involved with our city – Glasgow, and after that we did some data analysis which allowed us to grade our application.

3) Methods and Techniques

The evaluation that we did was a summative evaluation We used this type of evaluation, because it was used to identify areas of weakness – the application can be improved based on potential users' suggestions. Apart from distributing a questionnaire which involved deploying server code online (http://ppyordanov.pythonanywhere.com/) and providing the client installation files we also did interview-based evaluation in a controlled environment (laboratory) using the Think aloud evaluation paradigm. We measured the time that each task was completed by each user during the evaluation.

4) Stake Holders (potential users of the application)

- Inexperienced users - users without development background
- Experienced users - smartphone application professionals

All the group members from the first group were smartphone application developers with experience of developing different smartphone platforms. Most of the group members had developed different smartphone applications on iPhone, Android and Windows Mobile platforms. The second group consisted of people without any background of smartphone application development.

5) Instruments and Data analysis

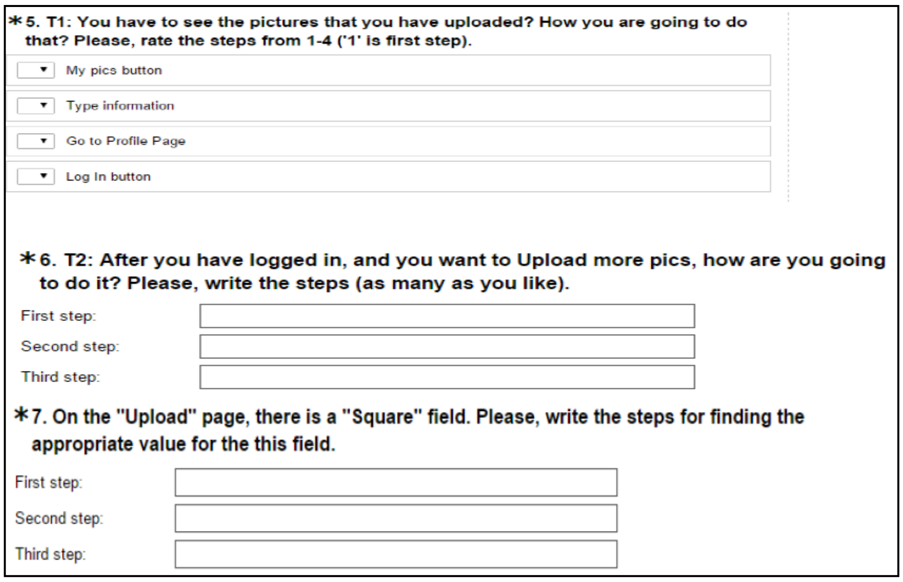
The instruments in the study were a set of questionnaire in which three tasks were defined in order to measure the level of usability in the designed prototype as well as a synchronously recorded Think aloud session for some of the participants. Continuously observing users and tracking their progress helped with the identification of usability issues. The selected tasks involved the use of the main features of the prototype: logging, user profile, taking/uploading a picture and map visualization, . The data gathered for each task included:

- Time to perform each task
- Total errors made for each task
- Think aloud comments were recorded and then key information was extracted

The experiment procedure consisted of three phases. Before performing the tasks, in the first phase, an introduction was given about the SunnyGlasgow application and it's features. All the participants had to answer some questions in order to look around and play with the application – these are the pre-questions. In addition, all subjects were given a description of their tasks.

In the second phase, participants were asked to perform the defined tasks using the prototype (session 1 involved familiarization with the server while the second session was aimed at using the client). We recorded the time when they started and finished each task.

Finally, in the third phase, users were asked about their opinion of the application – these were the post-questions. The defined tasks were as follows:



*5. T1: You have to see the pictures that you have uploaded? How you are going to do that? Please, rate the steps from 1-4 ('1' is first step).

▾	My pics button
▾	Type information
▾	Go to Profile Page
▾	Log In button

*6. T2: After you have logged in, and you want to Upload more pics, how are you going to do it? Please, write the steps (as many as you like).

First step:	
Second step:	
Third step:	

*7. On the "Upload" page, there is a "Square" field. Please, write the steps for finding the appropriate value for the this field.

First step:	
Second step:	
Third step:	

Evaluation 1

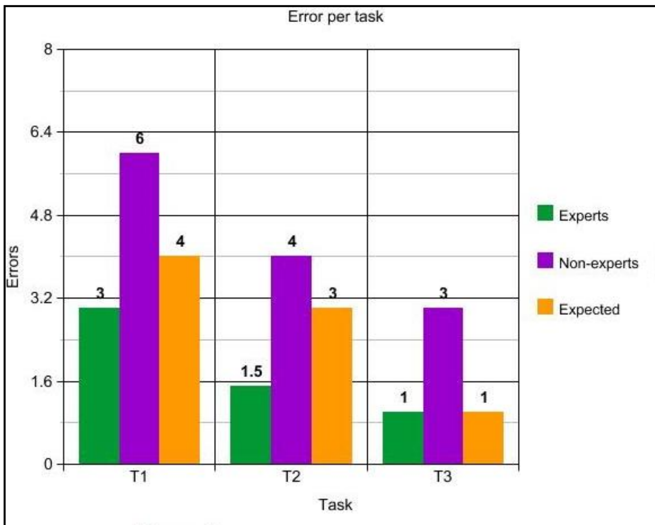


Figure:1

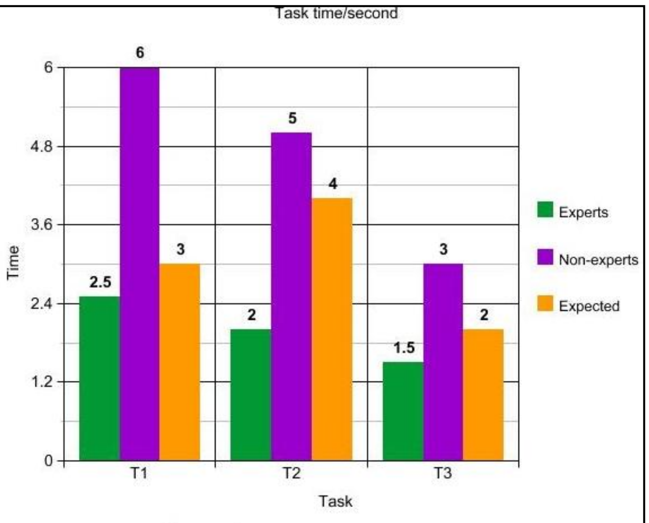
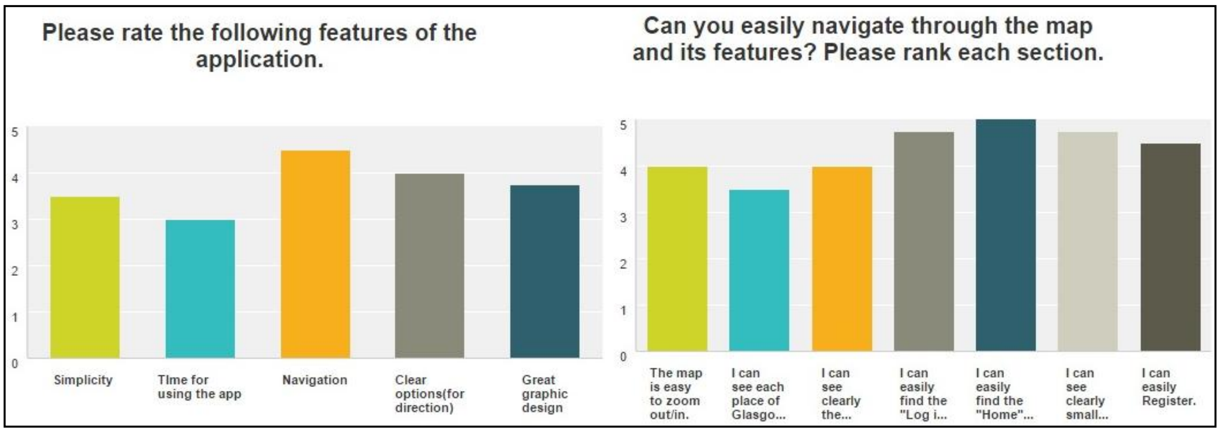


Figure:2

Evaluation 2

The average number of errors performing each task can be seen on **Figure 1**. It shows that all of the groups had good results in performing the mentioned task.

Figure 2 shows the average group time per task and the total average time for all users. In addition, the expected time was calculated by performing the same tasks by us – the evaluators. The expected time for each task was calculated by adding extra seconds or minutes, depending on the complexity of the task. It can be seen that the most time was spent on **T1** and **T2**.



Evaluation 3

Figure 3 shows two of the post-questions of the questionnaire. The y-axis for both of them is a ranking from 0 to 5. A we can see the charts show that all of the users has ranked the features of the application above the 2.5 (which is the average from the ranking). It can be inferred that the system has an intuitive UI, yet, it could be improved even more.

Conclusions and future work

Considering the user feedback we received during the evaluation process we can conclude that the system well implemented and user-friendly and it's a great idea for interaction. We believe that there still is some space for improvement and will be implementing functionality suggested by evaluators in a future release.

Users' think alouds and analysis of the collected data during the experiment helped us to develop a set of usability suggestions in order to improve the prototype in when developing for production. The list of the considered suggestions is given bellow:

- Adding an about page with some more information on the overall system
- Improving the client to include more functionality and enhance the file upload process
- It's a useful feature to add field for comments;
- Search area would be a valuable feature, especially in the future when there would even more data
- “Like” or “Favourite” buttons would be a fun and interesting options for the end user.

Resources

- Server Codebase: https://github.com/ppyordanov/HCI_4_Future_Cities/tree/master/Server/src/
- Client Codebase: https://github.com/ppyordanov/HCI_4_Future_Cities/tree/master/client/
- Materials (design, planning, prototype, evaluation): https://github.com/ppyordanov/HCI_4_Future_Cities/
- Online project: http://ppyordanov.pythonanywhere.com/