Generic Crowdsourcing System for Dynamic Environmental Surveys

Miguel Santos Horta Antunes Leitão
miguel.s.leitao@tecnico.ulisboa.pt

Instituto Superior Técnico, Lisboa, Portugal

January 2021

Abstract

Urban ecology is a subject that involves different scientific areas, e.g., ecology, biology, geography and sociology. There is, however, a lack of reliable documentation about Urban Green Spaces (UGS) and how they affect the lives of the population, as well as no efficient way to collect it. This information has enormous value for biological and ecological studies and even for urban planning. This report proposes a solution for this problem through the development of a generic system to create crowdsourcing applications. With this framework it will be possible to build a mobile application to collect crowdsourcing data, a web console to analyse data and a server to answer the calls and implement the features from both clients. The created applications will be easily adaptable to solve any real-life problem that may arise. The system will allow the development of an application able to extensively gather detailed information about UGS by dynamically surveying its visitors about their experience while in those green spaces. This application will be able to deploy different surveys for different states of the system, collect fitness and geographical data from external sensors and provide data validation mechanisms. All these functionalities and any other can be integrated with the generic part of the framework with little programming effort.

Keywords: crowdsourcing, urban green spaces, surveys, well-being, planning

1. Introduction

1.1. Motivation

Urban Green Spaces (UGS) are public spaces that have elements like trees or some other type of vegetation and possibly water bodies, like lakes, water streams or fountains. These spaces are an important part of any city or big urban area. Unlike in small towns and rural areas, urban areas’ residents do not have easy access nor frequent contact with nature and so it is important to provide them with this indispensable connection.

The existence and easy access to appealing UGS encourage urban dwellers and attendants to visit them. Besides its ecological benefits, like e.g., temperature regulation, water retention, biodiversity protection, UGS are known to be spaces dedicated to leisure, and amongst the most common uses for these spaces are relaxation and exercising. The mentioned activities are ways to keep and improve health both at the physical (exercise) and mental (relaxation) levels. Apart from the benefits above, these spaces also provide the opportunity for social gatherings, that strengthen visitors’ sense of belonging and community, also contributing positively to their well-being. Thus, people’s perception of well-being can be influenced by the layout and infrastructures found in the UGS they visit.

1.2. Problem

There is a lack of both full mapping of UGS in most mapping platforms and an understanding on people’s motivations and preferences towards UGS. These problems have been noted and registered by researchers and practitioners (e.g., planners and decision-makers), who would benefit from this missing information either purely for research purposes or to apply it in other various fields such as urban planning.

It is important to promote and facilitate the attendance to UGS through the collection of current data about urban dwellers’ uses and preference of green spaces. The deficiency in this type of data was noted and exposed by authors in several pieces of work related to this subject and can be divided in three main categories: Visitors Profiling, Relationship with UGS and Visits Motivation. This issue contributes to proving that researchers and practitioners are in need of a more complete database, not only with geographic data about UGS that could be used for research and assisting in future urban planning, but also social data about the visitors themselves, all that could be used to improve UGS and
promote the attendance to parks and other green spaces. More important than to solve the immediate problem, which is this shortage, is to think about a long term and broad solution because the real problem might not be the actual missing of information but the nonexistence of an efficient mechanism to collect it.

There are multiple crowdsourcing applications already developed. These apps focus on solving various problems in a wide range of subjects. There will always be new problems that will require different type of data collection and processing also requiring a different application. The existing applications allow the creation of simple surveys and the gathering of their respective answers for analysis by the survey creator or someone else with the appropriate permissions. What these applications do not encompass is the possibility of extension, the possibility of anyone with little programming knowledge to implement their own version of the existing features or even create new ones in order to help solving a different and specific problem in another context. Also not possible at the moment is information collection from external sources, sometimes the classic survey format may not be enough and sensor data could be used to complement users’ input and enrich the overall drawn conclusions. Lastly, there are no data manipulation mechanisms available, the purpose of collecting data is so that it can be studied and so it should be allowed the mechanisms to handle the data the as the people studying it see fit.

1.3. Solution

In this work it is proposed the development of a generic system that allows the creation of crowdsourcing applications that can then be specified into a wide range of themes. The system will provide the traditional functionalities found in this kind of applications but also add some additional features not currently available in other tools. Features like (a) a better definition of the gathered data so that there is full control over what information is collected in order to better fit the needs of the system’s creator, (b) a larger amount of data sources or sensors both internal and external capable of getting data not obtainable through user input and (c) data cleaning mechanisms through which data can be handle in the most relevant way to fulfill the system’s purpose, this will improve the results of the applications while allowing to apply to other areas the useful concept and tool that is crowdsourcing.

Starting from the provided generic system, anyone with basic programming experience and knowledge will be able, through the creation of simple extension files, to make an application with the common crowdsourcing features but also with extra functionalities dedicated to solving their specific situations and problems.

Using this generic system it will be possible to create a crowdsourcing application to solve the lack of information about UGS issue. This application will take advantage of the framework’s extensibility to incorporate modules with the functionalities that will allow the dynamic collection of data from the urban dwellers that visit green spaces.

2. Background

In this section the two main concepts behind this report are being explored: Urban Green Spaces and Survey Applications.

2.1. Urban Green Spaces

Urban Green Spaces is in a sense a somewhat vague term but a good definition was made by Plunz et al. when defined UGS as components of the Urban Green Infrastructure, which is the network of interconnected multi-functional green spaces which if strategically planned and managed it can deliver multiple benefits - ecological, social, and economic for cities and its urban-dwellers. The UGI comprises private and public spaces such as parks, urban forests, allotment gardens, green roofs, derelict lands, street trees, among others, which are accessible for leisure.

There is several literature that shows the links between UGS and their visitors’ well-being and health along with the communities where they are located. These are the most natural and semi-natural structures existing in the urban areas and so where most of the biodiversity can be found. Biodiversity and the nature itself provide a variety of services - Ecosystem Services (ES) - which are benefits nature provide to people, such as temperature regulation, air quality improvement, or even recreation, relaxation, among others.

Even with the need for further research about the correlation between UGS and well-being of their visitors and their communities, it is possible to infer, from work already developed in this field, that UGS affect positively the stress level and mental health, self-esteem and sense of community of those who visit them. They also promote and stimulate social interaction, through a sense of belonging and the creation or strengthening of neighbourhood ties and even assisting in ethnic inclusion.

One important step in this project will be assessing the uses and perception about the UGS by their visitors. This assessment will be done via questions asked to the green spaces’ visitors. Each of these questions were designed to get the information essential to achieving this project’s goal. This is supported by several research papers and case studies in this area that emphasize the importance of the data gathered to improve the understanding of these questions.
The main subject were a lack of information what noted were Visitors Profiling, Relationship with UGS and Visits Motivation.

2.2. Survey Systems

Traditionally, sources of information used by research and academic purpose were required to follow some standards and scientific methodologies. Yet, this vision has been challenged by the understanding of the increased value of citizens as a source of valid information, that as said by Goodchild, individuals should be considered as intelligent sensors, each of them an independent synthesizer and interpreter of local information [2].

This, triggered extensive research and development of work related with this subject to explore the true potential of this field and its application to a wide variety of problems. Work from different authors applied in different situations led to the emergence of different terminology and definition throughout the years. The terms that have arose refer both to information generated and to processes used to generate it and although the definitions might diverge, they all have the same core principle, citizen involvement in carrying out different activities while collecting data. Some of the most used terminology regarding peoples’ participation are Crowdsourcing, Volunteer Geographic Information, Citizen Science, User-Created Content (UCC), Public Participation in Scientific Research (PPSR), Public Participation Geographic Information Systems (PPGIS) and Map Hacking [6]. These methodologies have been applied in some of the following areas, as communications, crime, disasters, ecology, education, environmental monitoring, among others. One of the biggest advantages regarding crowdsourcing geographic information is that researchers no longer need to travel to a specific location in order to collect data, the same can be done by regular people that are already on-site.

It is common to find in literature about Crowdsourcing, Citizen Science or VGI mention of what started and fuelled all the work related to this area, this enabler being technological advancements made. What was briefly described by See [6], Haltofová [4] and Bubalo [2] was explored in detail by Goodchild [3]. They identified Web 2.0, Geotags, Global Positioning System (GPS), Graphics and Broadband Communication as being the foundations on which crowdsourcing applications are built. All these technologies are accessible using smartphones, which have become an indispensable object in our everyday life. Because smartphones have internet access from almost everywhere through mobile data, GPS signal and cameras and they can work as perfect sensors to gather data from and by their users. The previously mentioned GPS signal and geotags is a key factor in enabling geo-crowdsourcing applications. By appending the geographic location of the user to the data that has just been produced allows that location from where the data originated to be studied countless ways.

3. Generic Crowdsourcing System

In this section are briefly described the framework’s requirements, the different levels of its architecture and the implementation of the main features.

3.1. Requirements

Sensor Integration

It is required of the framework to allow the integration of external data sources, by collecting data from web sensors and device sensor either native hardware ones or user data from third-party platforms.

Easy Authentication

The framework should provide as easy and fast way for the users to register and log into the system, using for instance third-party platforms to handle authentication.

Multi-OS Operation

It is important for the framework to be able to reach as much OS and consequently as much users as possible through the usage of technologies able to run in as many different OS as possible.

Easy Development and Deployment

The framework should spare some of the programming effort by using technologies that ease the development process through built-in libraries and methods and the same programming language in more than one part of the system. The used technologies should also offer a free and easy method for the initial application deployment.

Dynamic Questions

The framework should be able to deploy dynamic questions to its users. These dynamic questions can be associated to a user status where the new questions will depend of previous answers, can be bounded to a specific period of time, user profile or even geographical location.

User Profiling

The framework should have a profile feature not only to provide the users with a sense of identity but also to collect useful data that researchers can use to draw some more conclusions by crossing user answer with useful personal information.

3.2. Architecture

To represent the system’s architecture was used a C4 model approach. This model allows to display software architecture at various levels of abstraction, containing four levels: Context Level, Container Level, Component Level and Code Level. From the context level to the code level this model will provide more and more details about sys-
This framework was created by a team of two students, myself and Guilherme Eugênio, another Electrical and Computer Engineer master’s student and the color schema used in the Figure 3 reflects the work distribution when it comes to the development of the system’s different components. The colors blue and white are used to represent components and containers that where developed by both team members, the color green was attributed to the components built by myself while the red components are the ones that were developed within the scope of Guilherme’s dissertation.

The applications’ Authentication modules connect with their server’s counterpart and the Third-Party Authentication Platforms. The Questions component receives Surveys from the server, both dynamic and static and sends the respective answers over to the Answers module back in the server. The mobile app’s Sensors component can connect to the External Sensors and is responsible for gathering additional data and send it to the server as part of a survey answer. These are only some of the most relevant components and interactions they establish in this architectural level, other interactions where not mentioned for simplification reasons.

3.3. Implementation

This section presents the framework’s data model, used technologies and how its most relevant features were implemented.

3.3.1 Data Model

The final level of the C4 Model is the Code level, for this project the simplified Unified Modeling Language (UML) diagram in the Figure 4 shows how data was perceived and modelled during the development of the framework, using class diagrams or entity relationship diagrams.
As before, only the most relevant aspects of this level will be described. A Survey might be created with contributions from one or more Researcher (1..*) and researchers may have as many survey as they want (*). Due to the dynamic nature of the surveys the presented questions may vary, and so an association is established between the Survey and Question. Each survey must have at least one question (1..*), but each question belongs to one and only one survey (1). A Survey may incorporate as much Sensor data as wanted but this collected data is exclusive of one survey only (1). The sensor may be of different types, such as, Web or Mobile native. Each Answer belongs to a single Survey but each survey may have a lot of answers (0..*). Between survey and its answer is established a composition, this means that when a survey is deleted so will all of its answers. Each Answer is produced by a single Normal user which if the application allows it might answer more than one time to a survey.

3.3.2 Technologies

The technology chosen to build the server was Node.js which is an open-source runtime environment that can run JavaScript across platforms without needing a browser. It is asynchronous and event-driven, easily scalable and ensures high performance due to the used compiler. On top of the Node.js environment was used the Express.js web application framework, although simple and flexible this framework provides a robust set of features to build web and mobile applications like a huge variety of HTTP utility methods and middleware.

React Native was the technology chosen to build the mobile application, which is an open-source JavaScript Framework created by Facebook and used to develop mobile applications for both Android, iOS, UWP and even web. This tool allows the development native rendered apps using React and it was used in the development, fulfilling the requirement for multi-OS operation from the section 3.1. The development and deployment of the application was made through the Expo Command Line Interface (CLI) that is free third-party service which provides a managed application development workflow, that takes the development complexity away from the programmer by making available tools to help in various tasks like integrating native device features.

As for the web client, taking into account the choice made in terms of the framework for the mobile application, it would only be logical to choose React to build this part of the system. This is also an open-source JavaScript Framework created by Facebook that has as its main objective the creation of front-end web applications by rendering data to the Document Object Model (DOM).

The server connects to a database that was implemented with MongoDB, an open-source document-based database, easy to develop and scale where data is represented as a collection of documents and where the schema does not need to be well-structured. Ad-hoc queries, real-time integration make MongoDB good at changing data frequently.

To implement the cache used to store the users’ credentials during a session was used a Memcached server running on the same system as the server. Memcached is a free, open source, high-performance, distributed memory object caching system. This in-memory key-value storage is of fast deployment and easy use and there are available APIs and packages of this tool in many programming language.

3.3.3 Features

Surveys
The surveys feature is one of the most important features because the generic crowdsourcing system like every other crowdsourcing system relies heavily on the ability of delivering surveys to their user but also because for this specific framework a large part of the programming effort was directed to develop this functionality. This feature operates in the mobile application and the server parts of the system.

In the application there is available a screen where the users can visualize and answer surveys, this was made possible through the usage of the react-native-json-forms surveying tool. This is an open-source NPM package created to easily deploy forms from a JSON file in mobile applications built using React Native and Expo, created specifically to fulfill the generic crowdsourcing system’s surveying needs.

Figure 4: Data Model or Code Level Architecture.
On the server this features takes a form of a series of endpoints dedicated to handle all the functionalities inherent to the surveys. There are available endpoints to deploy surveys to the users, to receive and store new surveys from the researcher, to receive and process the surveys’ answers, there is also an endpoint responsible to provide feedback to the users’ answers.

When it comes to selecting the crowdsourcing questions, the framework allows dynamic surveys and static surveys. For the dynamic surveys the programmer must develop some kind of system of conditions that will return a certain survey depending on the system state, as for the static surveys, the framework will always return the same set of questions.

The programmer can integrate these functionalities by providing an implementation for the respective extension functions, here it must decide from where to get the survey, the logic behind their deployment and manage how to respond to the HTTP request that called the endpoint.

The function responsible for getting static surveys runs after the dynamic surveys one, it was implement a variable dynamicRes that allows to passed values between these functions.

![Diagram](image)

Figure 5: Dynamic and static surveys flow.

In the Figure 5 is represented the flow of the surveys feature, where the server tries to get a dynamic survey, if any condition is met it returns the corresponding survey otherwise the function returns null. Then static function runs and if receives a null value it responds with the static survey but if the received value differs from null, it simply forwards the received survey.

**Sensors**

To explore the untapped potential that are external data sources this framework implements a sensors module. The available sensors may gather information from native hardware components on the device or third-party platforms and applications installed on it that may contain useful user data. The sensor data original purpose is to provide additional information to the researchers but it was found also useful to users who are curious about the collected information and use it to keep track of their activity and surroundings.

Data from native sensors is limited to the device therefore limited to the mobile application side of the framework, but during the development of the system it was debated if the integration of the external sensors should be instead outsourced to the server. With the creation of the react-native-json-forms tool and the easy integration of extensible form elements it was made clear that sensor data collection could be executed by the app using this tool as interface. Other problem solved by moving all sensors to the application side was the rate limits of API keys for web sensors. With this feature in the mobile application is easier to implement mechanisms that allows users to access sensors using their own keys rather than having a single API key shared by every system’s user.

The react-native-json-forms tool creates forms from pre-implemented components listed in a JSON file, this package is extensible allowing the implementation of new form elements. The SensorElement was created as part of this tool’s extension and provides a generic element that can be instantiated as different types of sensors. This element must be declared in the form extension file in order to receive all the form elements of the type “ext:sensor”, these objects must also contain a sub-type field that will identify to which sensor do they correspond as exemplified in the listing 1. There must also be provided a SensorArray file that will list all the different sensors to be implemented, the SensorElement will then iterate through this array calling the correct sensor components according to the form specification JSON file.

```json
{
  "pages": [ {
    "name": "page1",
    "elements": [ {
      "type": "ext:sensor",
      "subtype": "sensorSubType",
      "name": "Sensor Name",
      "id": "SENSOR_ID"
    } ]
  }
}
```

Listing 1: Sensor element JSON example.

**Data Validation**

The data validation mechanisms available consists of a console where researchers can analyse the collected data, approve, edit or delete that same data. This console was implemented in the web client part of the system, where researchers after logging into the website will be presented with a single page application containing all the data validation methods developed by the programmer. The provided template of the framework only presents...
the researchers with a list of the survey answers and the possibility of excluding them from the database but other mechanisms can be integrated.

This console also allows users with administrative credentials to login and manage the registered users. The administrator can remove user from the system but there it is also possible to promote normal users to researchers and vice-versa. This management mechanisms can also be perceived as a sort of data validation.

4. Results
This section contains a description of the Urban Green Spaces application along with the implementation of its more interesting features. Also in this section are presented some of the results obtain in the application and framework’s usability tests.

4.1. Urban Green Spaces Application
The best way to test a system is to put it to use. In this section is presented how the generic crowdsourcing system was used to engage a situation where was identified a lack of an efficient way to collect data about Urban Green Spaces. To do so it was deployed a crowdsourcing application capable of mapping UGS and register user’s preferences about these green spaces, a web console was also made available for the researchers to validate and analyse the collected data and a server was brought online in order to support all the back-end need of both these clients.

Members of the Faculdade de Ciências da Universidade de Lisboa (FCUL) who have also encountered the mentioned problem suggested and collaborated in the design and development of an implementation of the system that would help gather data about the UGS in Lisbon. The needs and requirements of these collaborators were taken into account to develop a system able to delivery dynamic surveys, integrate sensors and provide a way to validate the collected data.

4.1.1 Dynamic Surveys
The dynamic surveys feature was designed to serve the users with the most appropriate surveys improving the quality of the collected data. The user can be presented with surveys requesting personal information (details surveys), they may be asked to mark on the map relevant points (mapping surveys) or to answer a survey about their experience and preferences in a specific UGS (base surveys). This last type of surveys its a sequence of sub-surveys that will also be presented dynamically depending on the answers that user has been giving.

The Figure 6 allows to visualise the interactions required by this feature. In the mobile application, the user can trigger the request for both the base and mapping surveys by clicking in different buttons while the details survey is deployed automatically by the system. The details and mapping surveys fairly simple but the base survey required the development of a status mechanism to identify which part of the base survey should be presented next. The status is variable that can take values from 0 to n parts of the base survey and a copy of it is kept in both the mobile application, server and database, assuring synchronisation.

The application requests a survey to the server through the /api/surveys/ endpoint to which it sends type of survey it wants and the current user status. This request reaches the server which then responds to the HTTP request with the dynamically selected survey that will then displayed on the mobile app’s screen. In that same screen the user answers the survey’s questions which are sent to the server, depending on the answer, the user status will be updated and the answer stored. The new status is then generated by the server and sent to the application, if it indicates that the user finished the base survey the app will ask the server for some feedback.

Figure 6: Dynamic surveys basic server flow.

In the Figure 7 is a flowchart of the server’s dynamicSurvey function implementation. If the application requested a mapping survey the server simply returns the survey if the survey type is base the function will start by checking if the user has already provided is personal details. If not, a details survey should be sent otherwise a base survey is selected.

Before sending a base survey question, the status is checked to assert if the user is starting a survey. If so, the necessary initialization is made, otherwise the case the getBaseSurvey is called to evaluate which part of the base survey should be sent.

At a point in the base survey the user is asked to select the UGS from where the survey is being answered, the user is then presented with a list of UGS which are dynamically selected depending on the user’s location.
The new status generation done is after the server receives an answer and passes it through a set of conditionals, for each status are considered all the possible answers. All the surveys in the server are stored in the file system and organised into an array, this facilitates their deployment since each status will correspond to a numeric value and a position in the array.

In the database are stored answer objects containing the user who provided that answer, the date when it did so, the state of the answer (finished or not) and most important the actual answer data.

4.1.2 Device and Web Sensor Integration

From the device were implemented a set of sensors with the support of the Expo’s Sensors library that provides various APIs for accessing device sensors to measure motion, orientation, pressure, magnetic fields, and step count. There were created accelerometer, gyroscope, barometer, magnetometer and pedometer elements which provide numeric value for the corresponding variables, the implementations of these sensors was adapted from the one provided in Expo’s documentation for this library and only work in the devices containing the hardware to support them. From the Expo Location library it was created a geolocation sensor that give the user’s latitude, longitude and geocode.

For this application the user is asked to share the fitness data from the Google Fit app. This data can be useful for both researcher and users since they can use it to keep track of their activity while using the UGS application.

A weather sensor was also implemented, the framework gets the information from this sensor through the Open Weather Map’s API. This is an online global weather service that provides weather forecast based on a geographical location and it is necessary a previously obtained API key to access its weather data. To simplify the UGS app user’s job the an API key was obtained and made available for all the users to use in the application’s configuration file.

4.2. Evaluation

Extraneous events to the elaboration of this project encumbered the system and application testing that was planned to be carried out. Nevertheless, is was possible to perform some tests and collect feedback about the UGS app from a small but miscellaneous group of people. The group contains people with both technical and non-technical, scientific and non-scientific backgrounds being the majority of the elements university students. Females represent 52.9% of the group and males are only 47.1%, 58.8% are iOS user and the remaining 41.2% use Android. A significant amount of people found the application hard to install (41.2%) which may be related to the deployment in the Expo’s servers that require the installation of the Expo client in order to access the app, these are extra step that people are not used to in the app installation process. The majority (64.7%) were first time users of a crowdsourcing application while 17.6% have used crowdsourcing applications from two to four times and another 17.6% have used this type of applications five or more times.

It was carried out an After Scenario Questionnaire (ASQ) about completing a survey. 52.9% of the users answer 2 when asked about how easy it was to complete a survey, 17.6% answered 1, 11.8% answered 4 and 5.9% answered 3, 6 and 7. Overall it is safe to say that the majority of the users found the surveys easy to answer. The surveys were not only easy but also fast to answer was the opinion of the users since 29.4% of the users answered 1 and 2 when asked about the amount of time spent answering a survey, 17.6% answered 3 and 11.8% answered 4 and 7. When asked about the satisfaction regarding the available support information 41.2% answered 1, 23.5% answered 2 and 11.8% answered 3, 4 and 7 which allows the conclusion that the support material for is more than enough to answer a survey in the app.

The UGS application obtained an average value of 67.94 in the System Usability Scale (SUS) which is an average value according to Usability.gov [1]. The main reasons for this score are that 35.3% of the users answer 5 when asked if they would need the support of a technical person to be able to use this app, meaning that the application complexity may need to be rethought. Also, when asked if they found the app very cumbersome to use 35.3% answered 3 which may be motivated by the application flow and complexity too.

Besides this data some users reported map re-
lated bugs and suggested for the application to be made available in the App Store and Google Play Store.

5. Conclusions

With this project it was possible to create a framework able to fill a gap in the world of crowdsourcing applications. This gap is for the existence of a tool with the ability of easily developing a crowdsourcing system that allows full-stack customization. The system makes possible the creation of crowdsourcing applications dedicated to any subject by integrating features that prove to be more adequate and helpful to achieve each application’s goal.

The implemented system is composed by three main parts, a mobile application used by the normal users to collect the crowdsourcing data, a web client console application where the researchers responsible for creating the system are able to analyze the collected information and a server that answers the calls from both of these applications. The server should manage the data read and wrote to the database and may be responsible for some computation required to satisfy some of the requests from the clients.

The existing tools to create systems like this offer some limitations that the generic crowdsourcing system is able to overcome. The system stands out from existing technologies for implementing innovative features such as dynamic surveys, integration of external data sources like sensors and data validation mechanisms. When extending the system the programmer is able to select the desired features and also provide its own implementation for those features as well as for new ones. The mentioned full-stack customization comes from the possibility of the extension of all the system parts, the application components can and should be extended to implement the screens where the user can input the crowdsourcing data, the web client must be altered in order to implement the functionalities required for data analysis and validation by the researcher and the server’s endpoints can be modified to serve all the back-end data and computation required for all those features to be used by the system. This gives the system’s programmer and consequently the researchers complete control over the system, the data it collects and the processing done to that data.

The tools and technologies used in the actual development of the system were chosen in order to facilitate the fulfillment of all the set requirements. JavaScript is the dominant programming language in this system since the server was built in Node.js and Express.js, the web client and mobile application were created resorting to the React and React Native frameworks, respectively and the database is MongoDB to take advantage of its document-based characteristics. The mobile application was developed using the Expo framework and platform which also facilitated the publication of the app online by doing it onto its servers. The server and web client parts of the system were deployed on a private Ubuntu server running Apache.

In particular this system should be able to solved the real-life problem proposed by researchers from FCUL. This problem expects the system to facilitate the development of a crowdsourcing tool that collects data about Urban Green Spaces. There is a lack of documented mapping and general information about these spaces and the users’ perception of them. The researchers and practitioners pretend to collect data about the UGS general characteristic such as the existing elements in those spaces, both natural and artificial. It is also of the interest of these researchers to collect profiling data about UGS visitors and finally, information about the relationship the visitors establish with those spaces. The relationship data should allow researchers to understand which UGS are preferred and which are avoided, also when and why are these places visited and who visits them. These was accomplished through the development of an extension/application of the system. This extension implements a series of surveys deployed dynamically to the user, the dynamic feature was customizable in the mobile and server extension files and was based in a user status system that allow to assess when is the most convenient moment for a user to answer each specific question, this assessment is based on the user’s previous answer and other factors such as user location.

The development of other applications from the generic crowdsourcing system by other collaborators of the system worked as a proof of concept for the ability of the framework to have different usages and the possibility of easily integrating extension modules. The eFlechten lichen app uses simplified version of the surveys module created for the UGS application where instead of serving dynamic surveys, it presents always the same static survey. On the other hand, the UGS application uses immediate feedback which is a part of the feedback module developed for eFlechten that besides immediate feedback allows the implementation of complex differentiated feedback methods.

The surveys are a cornerstone for crowdsourcing application and so this task received additional attention during the development process. The lack of a tool that allowed the system to explore the full potential of the surveys feature led to the development of a tool to display and answer surveys on the mobile application. The React Native JSON Forms is Node.js package that allows to create forms in
React Native from a JSON file containing the form description. This tool is compatible with forms created using SurveyJS, allowing people without programming knowledge to design surveys. The SurveyJS tool only make available the most common types of input in form but in the React Native JSON Forms can be found some extra features that allow the integration of extensions containing new form elements.

In conclusion, the objective of this project was to develop a generic system able to create crowdsourcing applications that may vary in subject but are able to implement features different from the ones available in other applications and systems of this type. This framework was used to solve the UGS problem, through the development of an application able to collect reliable data about users’ experiences in green spaces. It is important for the generic system to encompass as many functionalities as possible to make the extension of the system an easy task. The system is designed to reduce the programmer’s effort, this can be proved through the analysis of the code from both the generic system and the UGS extension. While the system consists of 35764 (80.52%) lines of code the extension was developed with only 8654 lines (19.48%).

Acknowledgements

First I would like to acknowledge and thank my two thesis supervisors, Prof. Dr. João Nuno Silva for allowing me to realize my enthusiasms for system’s architecture and web development and for the opportunity to extend my knowledge in this area with this project and Dra. Ana Catarina Luz for the tireless support over the entire duration of this project and for giving a real-life application and purpose to my work.

A special thank is due to my friends and family that through their support and patience helped me carry my academic path to its conclusion. It is not an easy task to write and dissertation and it becomes an even harder thing to do during a pandemic and I am aware how tiresome might have been putting up with me in more than one occasion for that and all those who did it, I am most grateful.

Um bem-haja a todos!

References


