Follow My Steps

Rodrigo França Martins
Dept. of Computer Science and Engineering
IST/Technical University of Lisbon
Lisbon, Portugal
rodrigofranca@tecnico.ulisboa.pt

ABSTRACT

Computer storage has become significantly cheaper and we are facing great developments regarding sensing technologies allowing to efficiently understand personal activities and experiences. These advancements are supported by the growth of the quantified self-movement, in which life activities are tracked by using wearable sensors in the hope of better understanding human performance in a variety of tasks. Unfortunately, despite being easy to gather data, displaying it in a personal relevant way, avoiding scalability issues rising from the large amount of data collected, it has been a predicament for many interface developers. This work is focused on overcoming these issues and to provide a simple and elegant user interaction. Its main goal is to understand which are the best visualizations that allow people to recall experiences, within a certain period of their lives, in a personal relevant way. The system structure is divided into two versions designed for PC and mobile platforms, which are focused on time and location, respectively. Follow My Steps browser version implements a customizable interface that enables the addition and the removal of interactive and personalized visualizations, which can be filtered without compromising the scalability and the holistic view of the interface, that displays the submitted user's data. The Follow My Steps mobile version is a smooth application that provides a variety of information, related to visited places by users, based on their current location. The results, generated by the usability tests, reveal that the system is easy to use, functionalities were well integrated, without inconsistencies, and the interface presents a simple design. However, a set of older participants felt the need of a technical support while using the web interface for the first time.

Keywords

Lifelogging, Information Visualization, Dashboard, Quantified self

INTRODUCTION

Nowadays, with the exponential growth of the technology that we use in our daily lives, we can collect and store massive amounts of personal data and use it to study patterns and improve our self-awareness. Intelligent devices, like smartphones, are equipped with sophisticated sensors. These sensors, together with advanced computing power and network connectivity, offer opportunities to track everyday experiences, accurately and automatically, generating lifelogs that have great potential in providing valuable information about us. In this work, we will be addressing visualization techniques that help users remember past memories, through the use of their own personal information. The main objective is to help them visualizing their data, collected throughout a certain period of their lives, allowing them to act and make choices upon their own experiences. This data is used to display and report the memories of a selected period of time, such as, day, week or month, throughout the visualization techniques. While developing our solution, it is important to keep in mind the concepts of the lifelogging.

We focused on studying the best way to present visualizations to users, about their past experiences based on lifelogging data collected over an extended period of time, in a personal relevant way.

The aimed solution needs first to address the problems of scalability, personal semantics and holistic view. Additionally, the solution needs to be configurable, allowing users to have the best experience possible while using our system. It is important to acknowledge, when we reflect about configurability, that users provide different sorts of data, according to several factors that affect their data collection, and they must be able to select the interface visualization content. Since each user is able to input different types of information, we design our system to parse a limited group of data structures and store it using a common and similar basis. For different types of data, we create a new parser and a new database table. On the other hand, if the structure is identical the system uses the same resources. For example, it uses the same parser and database table while analyzing and storing csv and xlsx files, decreasing the space used and reusing code.

Follow My Steps browser solution provides several customizable visualizations, based on time, that can be added to a dashboard and manipulated in order to display particular pieces of information.

Depending on the data inputted by users, the system can display the places where they have been, how frequent they were in a certain place, what they have eaten at a certain day, the distance covered, the movies watched, among others. This can be used to enhance knowledge about personal data, helping users to decide and improve their life decisions in the future.

As well as, Follow My Steps web application incorporates an operation that sends an “emailable” report, based on time, which allows to receive, periodically, an email with

the information regarding the activities performed within a specific period of time. For instance, the system will send a daily, monthly or yearly report, with information regarding the activities executed on a particular day, month or year, accordingly with the user's preferences.

Besides, our project browser solution, we created a mobile version that provides insights about users' activities based on their current location. This simple app was designed to detect and analyze the position in order to ascertain if users have been at their current or near places. If a place has been visited, the app deploys the information related to the previous stopovers, displaying photos, places nearby, how many times they were in that place, among others.

ARCHITECTURE

The model used on our solution is structured with two main components, the client-side, developed to provide users a front-end system to visualize their personal data and the server-side, developed to parse and compute the incoming requests. The communication between these components is achieved by exchanging JSON objects from specific modules within these components.

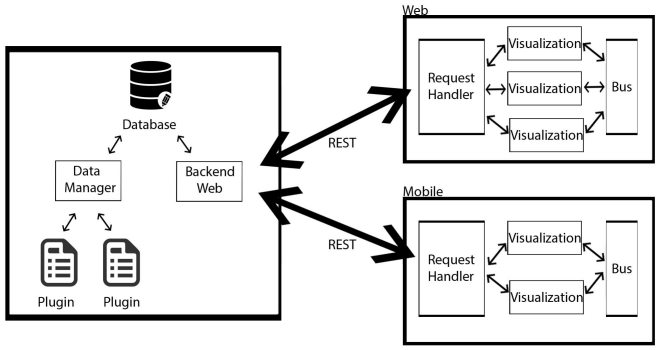

Figure 1. Architecture of Follow My Steps System

Server-Side

In our solution the server-side is made of four major modules, the backend web, the database, the data manager and the plugins, that process and retrieve data from and to the client-side, respectively.

The data collected will have heterogeneous properties since its sources may differ accordingly to the relevance that each user assigns to different sets of data. Several individuals can gather distinct information from various origins accordingly to their own interests and needs.

To support these data formats, the system is prepared to handle standard structures of data, that could, otherwise, cause consistency issues while storing and displaying it.
To solve this type of problems, we created two elements on our server-side, the plugins and the data manager or plugins manager.

The plugins will receive the data collected from the client-side and parse it in order to, homogeneously, store heterogeneous data from different datasets without changing its information. To accomplish that, each plugin, in our system, has an algorithm that knows how to arrange a specific type of data and where to send it. This also allows them to detect errors and missing data on the inputted information. Initially, we developed five plugins that process and handle data from Google Spreadsheets, CVS, LIFE, GPS and Photos.

The data manager receives the data sent by each plugin, computes it and creates the correspondent queries in order to store it on the database.

To store all the information inputted by the user, we created a database that not only provides storing operations but also has access to the backend web in order to post and get the information requested by the client-side. It is mandatory to manage data efficiently to allow users to perform multiple tasks easily and effortlessly. To accomplish that, the data stored is arranged into seven tables to cluster the information. These tables are used to store the files content, the definitions of the dashboard (to save progress), the variables (to use on the visualizations) and the information file (to provide quick details and update the files uploaded).

To handle the incoming messages, containing information with different data formats generated by the users' uploaded files, the system follows a set of requirements allowing to store, homogeneously, all the content within these messages. The most relevant requirement, that allows to connect data between database entries, is the timestamp. This property must be specified for each record that is stored, providing means to associate all the information in the tables with a specific period of time. Furthermore, and since our goal is to represent personal data based on a time interval, the timestamp attribute is present in most of the dashboard visualizations.

Client-Side

During the system usage, a huge number of requests will be exchanged between the client and the server side. To handle these requests, the client-side must have a module able to interpret and forward the messages to the proper modules. These messages contain information regarding the users' personal data, which will be displayed accordingly to their own interests and preferences. In order to provide a personalized interface, the system must ensure means to personalize the content displayed and to guarantee time consistency. In other words, the content displayed must be optional, editable, replaceable and be represented within the same period of time throughout the entire interface.

To allow users to manage, analyze and observe the data collected and stored, two interactive interfaces were developed, each one represented by a component on the client-side structure.

The first interface was built for web browsers and consists of three modules, the request handler, the visualizations and the bus.

The request handler act as an intermediary, which sends requests, made by the user, to the server and, then, receives the information, sending it to the proper visualization module. To avoid some problems such as performance,

efficiency and response time, the handler manages the requests from the multiple visualizations, prioritizing and sending the correct information to each one. It uses a cache that stores an object with information about past requests and the correspondent answers, to avoid recent requests repetition made to the server. The visualization modules are visual techniques connected among themselves through time, to preserve the interface coherency, which can be represented as Bar Charts, Line Charts, Maps, among others. Similarly, to the plugins, these modules are created individually, allowing developers to add or remove new visualization modules to the system. The main goal of the final module, the bus, is to ensure coherency among all the visualizations by displaying their respective information bounded to a given period of time. To link them, the bus is notified by the visualizations each time an update occurs, in order to communicate the changes to the remaining ones, by trigger a time event that each visualization is subscribed to. The second interface was built for mobile, displaying a simpler design driven by the purpose of providing an "on the go" solution for users interested in analyzing personal information on a portable device. This structure is composed of the same modules found in the first structure; the request handler, the visualizations and the bus. The main goal of this component is to allow users to remember, anywhere and in a lower detailed visualization, their past experiences according to their location.

IMPLEMENTATION

Here we discuss the functional purposes of each developed component that adds new features and functionalities to our system. Furthermore, we will explore and reflect the design approaches and explain the choices we made in this work.

Backend

One of the biggest achievements that we endeavor in this project is to represent data that has personal meaning to users accordingly to their own preferences. In order to achieve it we cannot restrict the type or the format of the data that is inputted and inserted throughout our interface usage. As a result of this requirement, we created the plugins which are pieces of software that allow us to transform the incoming data and store it with a predefined format on our database. Since this data scheme changes between different types of information, and depends on the user's personal choices, it was imperative to create distinct tables for dissimilar types of data.

Follow My Steps server creates these tables as the data flow into our system, in order to store the inputted files content (for example the GPX table is generated when the user upload a GPX file). This approach, consistently, produces tables as they are needed instead of creating them when the server starts to operate.

The scope of the generated tables number, created by the database, can fluctuate between zero and seven. These are arranged into two clusters, the common tables, which are generated whenever information disregarding the data format is uploaded, and the file tables, which are produced accordingly to the file structure uploaded by users.

The first cluster is made of the definitions table, created to store the progress and all information changed, by the user, in the interface. The files table was developed to register all the file paths and names uploaded, as well as the last time that they were modified, allowing to update the interface by keeping on track all the changed files. The types table, which was structured to preserve all the data types from the file variables that were uploaded (string, number or boolean), gives access to them every time the system needs to verify whether a column is or not of a certain type. This is particularly useful when we want to plot a new chart on the dashboard, ex: where the y-axis must be an integer. The second cluster is made of tables that store the content of each uploaded file. Within this cluster we generate the GPX table, for GPX files; the LIFE table, for LIFE files; the general table, for CSV and XLSX files and the photos table, for all photos type files (such as PNG and JPG).

Every time a file is uploaded its content is analyzed by the server and send it to the proper plugin, which is prepared to parse it and arrange the data to homogeneously store it in the database.

There are a couple of properties that need to be saved that are common to every file. To keep coherency, we dedicated the first four columns of every table that stores file content to the row id, file path, file name and to the row timestamp. The id column is an integer that is automatically assigned by the database every time a new entry is inserted, which allows the client side to upload several new records without concerning about their database identification. The file path, the filename and the timestamp are information properties that depend on the file properties, which, consequently, need to be provided whenever a new file is uploaded.

The following columns diverge and to handle these changes each plugin must parse the assigned files content to organize the information according to the proper table specifications and restrictions.

Plugin

As mentioned before, each user will collect heterogeneous data with personal relevance, from different procedures and data sources, with origin consistency problems while storing it in the database. To counter this issue, we create fragments of code, called file parsers that allow the system to restructure the file content format in order to match with the database format. In Follow My Steps backend, we commonly designate file parsers as plugins. A parser is a program or a method that arrange input data, containing a specific format, so other programming entities can manage it. As mentioned before each plugin receives a file content and produces an output that is arranged, with a specific structure, in order to be stored.

We focus our efforts on developing plugins that go towards the most common data formats used to store daily information. Initially, five plugins were developed, the CSV

plugin, the XLSX plugin, the GPX plugin, the Photos plugin and the LIFE plugin.

CSV **–** The first plugin was developed to read and parse CSV files. The popularity of these files has been increasing due to their readability, manually editable simplicity, straightforward information schema, faster handling and easy generation.

To analyze and handle the CSV file content, the system interprets the file format as a table. Therefore, we use the first line to identify the columns and the following lines to identify the values. Each value, excluding those from the timestamp column, is arranged into a row on the database, where the system stores, among other information, the value and the value's column name. For each pair of key-value, where the key is the column and the value is its respective value, the system gets the correspondent time from the timestamp column. This column is specified by the user while uploading the file.

XLSX **–** Another file that became one of the most popular, around the world, is the Excel (XLSX files) due to its capacity to perform, among many functionalities, statistical analyses and spreadsheets allowing users to define the fonts, character attributes and cell appearance in the sheet. To arrange the data within these files the system uses an external module, called *SheetJS js*-xlsx [1], converting every filled cell of each row in a concatenated comma-separated value (CSV format). Once converted the system uses the same methods applied by the csv plugin to parse and store the information in the database. Similarly, users must specify the column that corresponds to the timestamp.

GPX **–** The GPX plugin was developed to parse GPS data stored on GPX (GPS eXchange) files. These files' format is an XML schema conceived to be exchanged among applications, containing geolocation data, such as tracks, track segments, track names, track points and other information. The importance and popularity of the GPS is a continuous growing over time as it allows users to locate themselves anywhere on the planet, freely, without technical details. On Follow My Steps system, it was essential to include positional information, in order to provide an insight of the places where users have been.

To understand this plugin, it is fundamental to learn the logic behind its modules. The first module is the *gpx-parse* that uses an algorithm to, basically, convert the GPX tracks information into a GPX object. This GPX object provides a synthesized version of the information within our file, which is subsequently used to create the arrangements needed to be inserted into the database. By analyzing this object, we can obtain all the data needed to support our visualizations based on location, such as the Map.

The problem of this approach resides on the huge number of information that is generated while collecting GPS data. In order to save all the information, the GPX files save the points, tracks, routes and all specified above. A simple walk of 1.4 km can produce over 115 track points and a 2 km walk can produce over 215 track points. For longer distances, the amount of information that is produced requires a lot of space and memory. In order to reduce this information quantity, we use another external module called *simplify-path* [2], that uses the Ramer–Douglas–Peucker algorithm that reduces the number of track points of each track allowing to create similar curves and lines with fewer points.

LIFE **–** The *LIFE* [3] plugin was created to parse an uncommon type of file format, developed by Professor Daniel Gonçalves from Instituto Superior Técnico, which provides a standard procedure to record information that the user collects throughout the day in a simple, expressive and detailed approach.

This format is, especially, useful because it is focused on structuring, through time, the activities performed by the user and offers syntax configurations to cluster them in three types, which are the spans, that represent the location or places that the user was performing during a particular time interval. The subspan, that provides additional activities about the places within a particular span and the trips, that represent the user displacements. Each of these activities is bounded to a timestamp, which must be manually inserted by users.

The file format also provides additional information about places and locations. With this data the system is able to register place inclusions, which represent places inside other places (for example, a shop inside a mall); canonical locations containing geographically information about a certain place; place categories, to arrange places in categories and name changes, which inform that a certain place has changed its name.

This plugin was developed to analyze the syntax of each activity and additional information and to handle them in order to generate the appropriate arrangements for the database.

Photos **–** Photographs are one of the most popular hobbies for many people all around the world. They help us to remember the past by preserving memories and reminding us about the people, places and activities we love.

The Photos plugin, unlike the others, is the only that does not require to parse the file content to store the information in the database. Instead of saving the photos content, this plugin only saves their file path, filename, street, city, country, timestamp, latitude, longitude and descriptions. This approach allows the system to store large amounts of photos without consuming a massive volume of space and to obtain the images content, through their file path, every time the user requests them.

To return successfully their content, the plugin uses an external module designated as *base64*-img [4].

In order to populate the database with all the information, we analyze the image metadata, using another external module called *exif* [5], containing several details, such as the place where the image was taken, providing the latitude and longitude coordinates and the day the image was

1. *https://github.com/SheetJS/js-xlsx*
2. *https://www.npmjs.com/package/simplify- path*
3. *https://github.com/domiriel/LIFE*
4. *https://www.npmjs.com/package/base64-img*
5. *https://www.npmjs.com/package/exif*

created. To ascertain the street, city and country we use the same external module used in the GPX plugin, the *reverse-geocoding* [6].

Finally, to rotate the upside-down images we use the *exif-image-auto-rotation* [7] module that rectifies the image orientation. Unfortunately, this module does not work in every image, it is required to have *exif* properties in order to be able to access and adjust the image orientation.

Services

Follow My Steps backend provides several services to the front-end, improving the interface quality and the user experience. These services englobe the add, update and remove files, the save interface definitions, the send report, the send feedback and the mobile access.

Since every method on the server side runs asynchronously the user keeps the capacity to interact with the interface while the server synthesizes and execute all requests.

Add Files – Every time the user submits a new file to the interface, and send its content to the server, the post handler manages the request and infer which is the proper plugin to process the incoming data.

After the plugin operates and parses this data, the handler adds the new file to a subscription list that triggers an event every time a change occurs. The process of detecting the file content changes is implemented on an external module, named *file-changed* [8], which returns an array with all the paths of the files that were modified.

Update Files – The file system implemented on Follow My Steps allows updating the files content stored without specific requests from the user or the client-side. The system uses an external module named *file-changed* that provides the list of the paths of every file that was modified after being submitted and stored in the *database*. Each time that the user adds or removes content from a certain file, that has been uploaded, the module will notify the update handler, which will delete all of its previously stored information and reinsert its entire content.

Delete Files – In order to provide full control over the submitted files, we developed a backend service allowing the user to specify a file, through its path, in order to remove it from the database.

When the server receives an HTTP delete request, with a file path as the parameter, the system runs through tree tables that incorporate data from this file and expunge every record related to it. These three tables are the same we populate while adding a new file to the database.

Save – As a result of the interface daily usage, the Save operation became a mandatory service and one of the most important provided by the backend. Deprive users to save changes could mean the obsolescence of this system. So we developed a service that receives data from all the visualizations inserted on the dashboard, as well as all of the settings, and saves them on a table designated as definitions.

Report – This *Report* service is the unique backend service that interacts with the user outside the Follow My Steps project. It uses an email account, created solely for this purpose, to deliver a report, with data from a certain day, month or year, to an email address that is passed as an argumentation the client-side. This email is dispatched through the use of a resourceful external module, designated as *nodemailer* [9]**,** which handles all the processes needed to structure and send the email.

To complete this module, we installed the *node-schedule* [10], which allows users to program an email delivery to a certain date and time. Whenever an email is dispatched, the reporting process leads this function to create a new schedule.

User Feedback – All the processes presented above run methods that communicate with the *Backend Web* server module, notifying whether or not an operation was successfully concluded. Unfortunately, the user does not have any feedback regarding these processes. To contradict this information absence, we installed a module designated as *socket.io* [11], which is able to create a virtual bridge that enables real-time bidirectional event-based communication, allowing to send and receive information both on the client and server side. The system uses this wonderful module to send information to the user about all the processes running on the backend.

Mobile Access – Mobile Access is the unique operation dedicated to the Follow My Steps mobile app, from the list of processes presented above. This service produces a six random digit code that is sent to the browser, through the *socket.io* [11] module. This method restricts the number of accesses to the user's data by guaranteeing that only individuals in contact with the browser version are able to see it, excluding third parties and inconvenient guests that might find the user's IP address and try to visualize user's personal information.

Security and Privacy – As we know, the internet is a global computer network, which provides large amounts of information, using standardized communication protocols. To store and share this information in a huge global network, users must be certain that only they can see and grant access to their data. This approach would require complex methods to avoid security issues and privacy information leakage.

The Follow My Steps system uses a local server, which means that the messages exchanged between the front-end, of the web browser, and the backend are processed internally, without leaving the user's machine. However, the mobile app needs to be connected to the network, so we create the security measures that were mentioned in the section above (***Mobile Access***).

6. *https://www.npmjs.com/package/reverse-geocoding*
7. *https://www.npmjs.com/package/exif-image-auto-rotation*
8. *https://www.npmjs.com/package/file-changed*
9. *https://www.npmjs.com/package/nodemailer*
10. *https://www.npmjs.com/package/node-schedule*
11. *https://www.npmjs.com/package/socket.io*

Front-end

The front-end of a software, application or website, is the interactive interface displayed to the users. In our system, we developed two interfaces, one for the web browser and other for the mobile app.

Web Browser Interface

Follow My Steps is a project developed to support users to visualize their data, collected throughout a certain period of their lives, allowing them to remember their own memories. To achieve its goal, this system transforms the raw information, inserted in the uploaded files, into customizable and elegant visualizations, that can be inserted into the interface to fulfill the user's intentions.

Visualizations

The main goal of Follow My Steps dashboard is to present users the data that was uploaded to the system's database. This would not add any additional value if presented with the same format found on the submitted files. To invert this situation, we developed a set of nine visualization categories, which are *Images*, *Bar Charts*, *Area Charts*, *Line Charts*, *Pie Charts*, *Map*, *Timeline*, *Heatmap Calendar* and *Text*, that provide friendly approaches to analyze and interpret personal information.

Images – One of the most usual visualizations that human brain associate to our memories and past experiences are the photos. They can illustrate vast volumes of information that cannot be expressed by words "*a picture is worth more than a thousand words*".

To build an interface that is focused on showing past experiences it was mandatory to include photos on the visualizations set. Each visualization can contain up to nine photos and all of them are independent, allowing to drag and resize a picture without affecting the others.

With these features, we can arrange and personalize our photos visualizations to highlight important or gorgeous images with simplicity and we can insert more than one picture without creating different visualizations.

Figure 2. Images Visualization

Charts – Charts are the most important tools to represent data in various aspects. A chart is a graphical representation of information, in which the data is represented by symbols, such as bars in a bar chart, lines in a line chart, or slices in a pie chart.

Follow My Steps provides several chart types that allow to interactively display users' personal data and to perform arrangement actions, enabling the selection of smaller time intervals, through the click and drag action inside the graph. The charts display a tooltip whenever the mouse hovers one of their values.

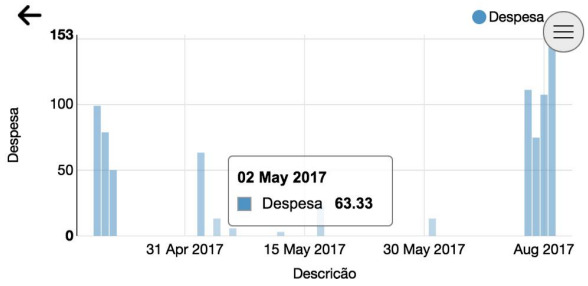

Figure 3. Bar Chart Visualization

Map – Another usual visualization that our human brain associates to the location memories and past experiences is the map. The *Map* is probably the most complex visualization on Follow My Steps system and, therefore, the most complete of them.

Each map inserted on the interface dashboard has four different layers that represent interactive elements, which can be hidden or set to visible by unchecking/checking the checkboxes present in the right bottom corner. The first interactive layer encloses the photo markers, which correspond to the photos taken at 50 meters or less radius distance from the marker location. The second interactive layer encloses the location markers, which are the places where users have been during a specific time of their lives. The third interactive layer corresponds to the user current location marker on the map. And the final interactive layer is designated as routes, which correspond to the tracks where the user has been, and it is represented as blue lines that connect two or more locations.

Figure 4. Map Visualization

Timeline – In order to analyze their personal data, within a certain period of time, users have to change constantly the period of time. The *Timeline* visualization is made of a horizontal bar with the month names, to guide the user while selecting a new date, and two circles, that represents the starting and ending dates of the interface time interval.

Figure 5. Timeline Visualization

Heatmap Calendar– To provide a simple visualization that englobes the daily information, from an uploaded file, we developed a heatmap calendar. This visualization can be arranged in months or in days, according with the user preferences. Whenever a day or month is hovered by the mouse, the visualization presents a tooltip with information regarding that specific date. The colors, which can be selected by the user, are arranged on a scale. The first represents the lowest and the last one the greatest value found on the visualization file.

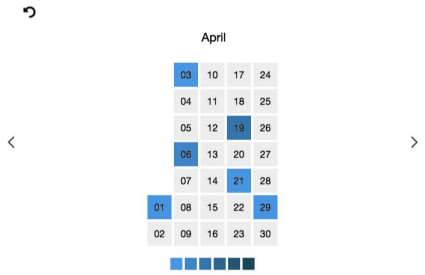

Figure 6. Heatmap Calendar Visualization

Text **–** The *Text* visualization is a string that can be used, for example, to add descriptions, notes, ideas and any other complementary information on the interface dashboard. This string can be a word or a sentence that the user writes on the input field presented in the *Text* container.

Mobile App Interface

The system browser version is focused on showing personal data within a certain period of time. To focus on position, we developed a mobile app that shows information about where users were and their current locations. The app is simpler than the browser version because it contains predefined visualizations that cannot be moved or erased.

When the user accesses the app, for the first time, the system launches an alert requiring the IP and Port of the machine that is running the server. The alert is composed of two text input fields, associated with these variables that will only dismiss if the user types a connectable address. After this step, the backend produces a code that is sent to the client-side, which is displayed on the web interface, allowing the user to type it on the mobile app to successfully connect to the server-side.

Once the code is verified and accepted, the app ascertains its current position, requests the nearest locations, the photos taken and the activities performed on the last day that the user was in the current place. The system also runs a period of time to update the interface accordingly with the mobile current position.

The map visualization, present on the app, is a similar and simpler version of the map created for the web interface. This map only provides two interactive elements, which are the location markers that represent the places where the user has been and the "my location" marker that corresponds to the user's current position. Additionally, the map has a visual element, which is a circle that shows the covered area selected.

The user's "my location" popup content is identical to the information shown by the same popup developed on the browser version. On the other hand, the locations popup has the same differences. They keep the time, days, hours and minutes at a certain place, but remove the heatmap calendar and the time range.

The map also includes a fixed button that provides an interaction that locates and navigate towards the user's "my location" marker. Since this visualization fills the entire window, the app contains an arrow button, on the right bottom corner, allowing the user to scroll down to the following sections. The first of the four sections that complement the map gives an insight about the user's current location, with information about the street, city and country; the second displays a list of fifteen or less items that correspond to the closest locations; the third and four shows the schedule of the last day in the current location and the photos taken on the current place;

Figure 7. Mobile Interface

USABILITY EVALUATION

The concept of *Usability Evaluation* refers to evaluating a product with representative users, allowing to identify several issues on the earlier stages of the product development.

To evaluate the Follow My Steps system we conducted usability tests with 21 participants. Each usability test was divided into two sections; the web tests, containing 12 tasks, and the mobile tests, containing 6 tasks. During the tests procedures we monitored the number of clicks, the time spent and studied the user's choices used to accomplish

the required tasks.

From the 21 participants that underwent the experimental evaluation, 14 were male and 7 were female. Through the results observation we infer that the gender did not affect the performance or the efficiency of the tasks required on this evaluation process. However, the age had a major influence on understanding how the interface works. The worst performances are connected to users aged between 46 and 60 years old.

The selected participants to evaluate this system knew how to interact with computers and mobile phones, and were used to work with other web interfaces and mobile apps. This knowledge was a major requirement to guarantee that we observe and study the problems faced throughout the interface interactions, instead of the problems faced throughout the computer or mobile phone usage.

Web Browser

After analyzing the ***Figure 8,*** we can infer that the average number of clicks per task is, on the majority of the cases, greater than two times the number of minimum clicks. There are distinct reasons that can be related to this behavior, considering the fact that users did not had contact with the interface until the beginning of the usability tests. The major disadvantage of users, was their unawareness regarding the shortcuts provided by the system. This information was not revealed to the user, in order to study their reaction towards the keyboard. From what we observed, only 3 out of 21 users successfully used the keyboard as shortcuts to perform operations that supported the tasks performances

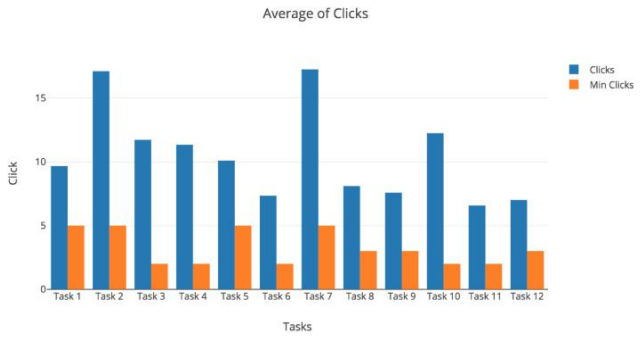

Figure 8. Web Browser Clicks Average

From the analysis of the ***Figure 9***, the task 2 (*Find how much cash I spent in September, 2017*) was, on the overall, the assignment that took more time. This observation is obtained by analyzing the median of each task (in this case is 1 minute and 54 seconds). However, the higher time to perform an assignment was registered by a user that required 6 minutes and 14 seconds to, successfully, complete task 3 (*Find how many times I've been on IST*). During the usability tests, we observed a higher difficulty from the participants aged between 45 and 60 years old. The set of results, produced by their tests, contained values that diverged on a large scale from the values produced by other results. The values that are detached from the variation range are considered as outliers, which are, commonly, ignored by analysts on statistical operations. In contrast, we observed faster results from users that began to feel acquainted with the interface usage. We registered, on two tasks, similar execution times between the user performance and the minimum value obtained.

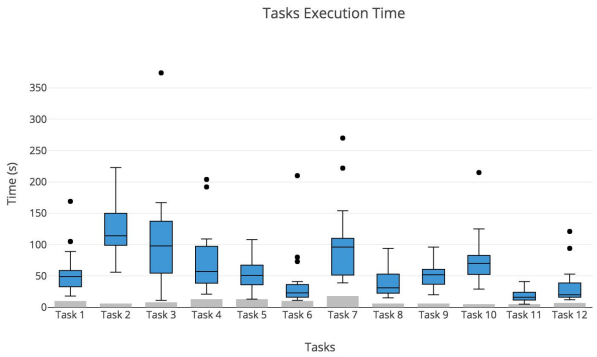

Figure 9. Web Browser Execution Time Average

Mobile App

On the mobile version, the system is considerably less complex and it requires less interaction to perform the tasks with success. Unlike the browser version, the mobile does not have any shortcuts to help users to perform the tasks faster and with fewer taps.

On the usability tests for this version we do not provide an experimental period at the beginning of the test, instead, we start directly with the tasks. The reason why we skip the experimental period resides in our intention on ascertain how intuitive the icons and the interaction on the mobile are. If we observe the graph, ***Figure 10***, we can infer that the user required more taps to perform the task 2 (*Verify how long I have been here the last time I was in this place*). An explanation for this behavior can be the time necessary to perceive that the calendar of the last day, on the current location, does not provide clickable interactions and to find the hours the user need to subtract the final hour and the initial hour. On the other hand, the task that demonstrates lesser clicks was task 6 (*Set the update time to 5 minutes*), and one of the reasons is because these actions are associated with the settings menu, so users know exactly what to do and where to find it.

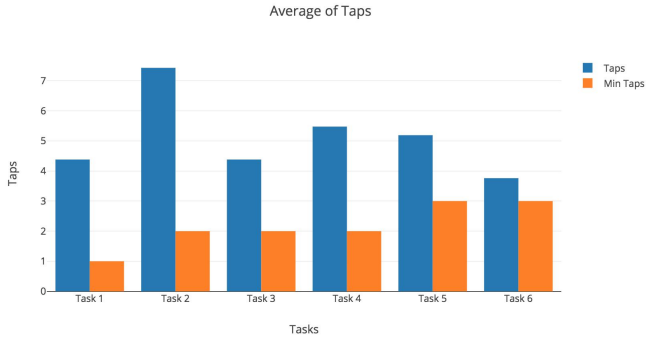

Figure 10. Mobile App Clicks Average

As we can observe in the ***figure 11*** the task that took more time, for users, to accomplish was the task 2 (*Verify how long I have been here the last time I was in this place*). Its median value is 29 seconds and the max value registered was 2 minutes and 7 seconds.

The faster task is the task 6 (*Set the update time to 5 minutes*) with the median equal to 14 seconds and the higher value registered was 45 seconds.

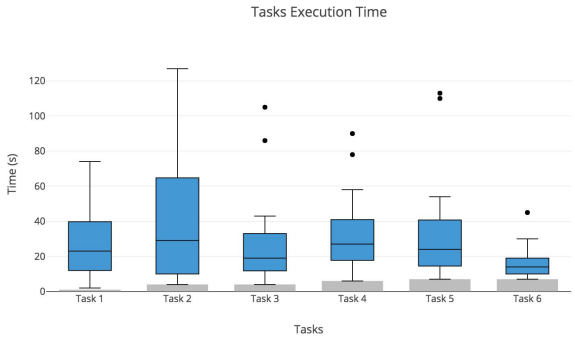

Figure 11. Mobile App Clicks Average

DISCUSSION

These usability tests provided important metrics to infer the major difficulties and facilities encountered while using the Follow My Steps interface.

As we mentioned earlier, at the beginning of the tests, the dashboard of the interface was empty, allowing users to build their own interfaces accordingly with their preferences. Each test started with the experimental period and we noticed that the visualizations absence was not emotionally positive for a set of users. On the suggestions section, from the web questionnaire, we read an opinion stating that the system should provide default themes, contained predefined visualizations, in order to display random information at the beginning of the usage. The problem with this approach consists on the fact that the system does not possess any data at that time, and the information needs to be uploaded by the user. A simple solution would be to redirect the user to the *Add Files* menu, in order to impose the requirement of inserting files on the *database*.

During the interface usage, the participants felt confident while performing the majority of the tasks.

We observed that most of users chose the map visualization to perform the tasks presented during the usability tests. All the tasks, except task 6 (*Find the photos I took on 'Pragal Railway Station'*) and task 10 (*Find the transportation method that I used to go to Alegro*), related with the visualizations, had more than one method to be successfully concluded. For example, users can use the map, the bar chart, the line chart, the area chart or the pie chart visualizations to ascertain the number of times that they had been on a certain place.

One of the possible reasons for this behavior is the association among locations and maps. When we are submitted to questions or tasks regarding specific places (such as, find how many times you have been on Lisbon) we tend to think on maps to search for the information requested.

A similar behavior was recorded while observing the user's performance on task 2 (*Find how much cash I spent in September, 2017*). However, in this specific task, users preferred the *Bar Chart* visualization technique in order to ascertain the value for this specific month. This option may be related to the fact that we associate the *Bar Chart* visualization to sets of values arranged over time.

To provide support to users we developed a *Help Menu*, which presents quick feedback on how the interface works, through a set of videos conceived only for this purpose. Throughout the tests, we observed that only 6 out of 21 users used the help to verify how to perform a certain task.

We conclude that the complexity of certain actions can involve some pre-learning experience. Despite this complexity, in overall, users were pleased with the simplicity of the interface and ease while executing the system's operations.However, there is still plenty of work that can be done to improve this system, considering the comments and suggestions provided by users collected throughout the usability tests and the final questionnaires. These improvements are related to a couple of small corrections on the interface and in mechanisms to turn things easier for users to handle with system's features.

CONCLUSIONS

The practice of Lifelogging started to increase, over the years, with the expansion of lifeloggers tracking their personal activities, by collecting their daily data, such as exercising, sleeping, or eating.

With the growth of smartphones in our lives, we can gather and store large amounts of data that can be used to help us remember past experiences through several visualization techniques.

This extraordinary increase of computer storage power generated several issues when developers try to display this quantity of information. One of these issues is scalability, which is the system's ability of handling the growth of the upcoming data inputted by users.

Our approach design was structured to implement a customizable interface, developed to allow users to personalize it according to their own preferences and interests. This particularity enabled us to understand which are the visualization techniques users associate to a certain set of data (map, bar chart, etc...), and how they choose to visualize it.

To evaluate this system, we submitted it into several usability tests, which were focused on usability and utility, to identify some major and minor issues, in terms of design and functionalities, and to study the best visualization techniques used to represent specific personal information.

From the results obtained, we infer that users adopt the map visualization while trying to observe information related to a specific location. For example, the number of

times or the days on that location. We, also, infer that data with measurable values are commonly represented in a bar chart. For example, the daily expenses or the number of travels.

To conclude this work, we deduce, from the questionnaires and users' feedback, that Follow My Steps is a helpful system that provides insights about the users' past memories and experiences. However, it has several aspects that could be improved to enrich not only its user's experience but also some of its backend mechanisms.

During the usability evaluation, we observed several difficulties faced by the majority of users, while starting the interface tests. This set of initial difficulties have a considerable influence on users' willingness to learn the interface features and on their motivation state.

To smooth this problem, we could implement an initial help system to guide users through a set of predefined actions, such as uploading new files and/or the inserting new visualizations to the dashboard, in order to aid them to learn basic operations provided by the system.

Another issue that we want to highlight is the updating process of the file update feature. This process, as mentioned earlier, runs on the server side of this application and deletes specific previous stored information from the database to insert the changed file full content. The time spent to execute each update operation depends on the file size context instead of the changes performed. This is the most important feature, on the server-side, that needs to be updated in order to reduce the amount of information that needs to be managed.

REFERENCES

1. Heike Otten, Lennart Hildebrandt, Till Nagel, Marian Dörk, and Boris Müller: "Are there networks in maps? An experimental visualization of personal movement data". In IEEE VIS 2015 Workshop.
2. Gemmell J., Aris A., Lueder R.: "Telling stories with MyLifeBits". In Multimedia and Expo, 2005. ICME 2005. IEEE International Conference on (2005), IEEE, pp. 1536–1539. 2
3. Alice Thudt, Dominikus Baur and Sheelagh Carpendale et al. 2013. "Visits: A spatiotemporal visualization of location histories". In Proceedings of the eurographics conference on visualization. p. 79-83.
4. Jae Ho Jeon, Jongheum Yeon, Sang-goo Lee, and Jinwook Seo. "Exploratory Visualization of Smartphone-based Lifelogging Data using Smart Reality Testbed". In Proc. Big Data and Smart Computing, pp. 29-33, 2014
5. Rúben Gouveia and Evangelos Karapanos: "Footprint tracker: Supporting diary studies with lifelogging". In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, CHI '13, pages 2921– 2930, New York, NY, USA, 2013.
6. Vaiva Kalnikait, Abigail Sellen, Steve Whittaker and David Kirk: "Now Let Me See Where I Was: Understanding How Lifelogs Mediate Memory". CHI 2010, April 10–15, 2010.
7. Mohit Shah, Brian Mears, Chaitali Chakrabarti and Andreas Spanias: "LIFELOGGING: ARCHIVAL AND RETRIEVAL OF CONTINUOUSLY RECORDED AUDIO USING WEARABLE DEVICES". In IEEE International Conference on Emerging Signal Processing Applications, ESPA, 2012
8. Bruno Pagno and Luciana Nedel: "Everyday Visualization". In IEEE VIS 2015 Workshop.
9. D. Huang, M. Tory, and L. Bartram: "Data in everyday life: Visualizing time-varying data on a calendar". Proc. Poster Compendium IEEE VIS, 2014.
10. Yang Yang and Cathal Gurrin: "Personal lifelog visualization". SenseCam 2013 Extended Abstracts, San Diego, USA.
11. Yang Yang, Hyowon Lee and Cathal Gurrin: "Lifelogging: New Challenges for Information Visualization on Mobile Platforms". CHI'13, April 27 – May 2, 2013, Paris, France.
12. Hyowon Lee, Alan F. Smeaton, Noel O'Connor, Gareth Jones, Michael Blighe, Daragh Byrne, Aiden Doherty, and Cathal Gurrin: "Constructing a SenseCam Visual Diary as a Media Process". Multimedia Systems 341-349, 2008
13. Tiffany Ng, Ou Jie Zhao and Dan Cosley: "pieTime: Visualizing Communication Patterns". 2011 IEEE
14. Sudheendra Hangal, Monica S. Lam and Jeffrey Heer: "MUSE: Reviving Memories Using Email Archives". UIST'11, October 16-19, 2011
15. Tsai Ling Fung and Kwan-Liu Ma: "Visual Characterization of Personal Bibliographic Data using A Botanical Tree Design". In IEEE VIS 2015 Workshop.
16. Larsen, Jakob Eg, et al. 2013: "QS Spiral: Visualizing periodic quantified self data". In CHI 2013 Workshop on Personal Informatics in the Wild: Hacking Habits for Health & Happiness.
17. D. McDuff, A. Karlson, A. Kapoor, A. Roseway, and M. Czerwinski: "AffectAura: an intelligent system for emotional memory," in Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, New York, NY, USA, 2012, pp. 849–858.
18. Dominikus Baur, IEEE, Frederik Seiffert, Michael Sedlmair, and Sebastian Boring et al. 2010: "The Streams of Our Lives: Visualizing Listening Histories in Context". IEEE Trans Vis Comput Graph, 2010.
19. Zaher Hinbarji, Rami Albatal, Noel O'Connor and Cathal Gurrin et al. 2016: "LoggerMan, A Comprehensive Logging and Visualization Tool to Capture Computer Usage". In 22st International Conference on MultiMedia Modelling (MMM 2016), 4-6 Jan, 2016, Miami