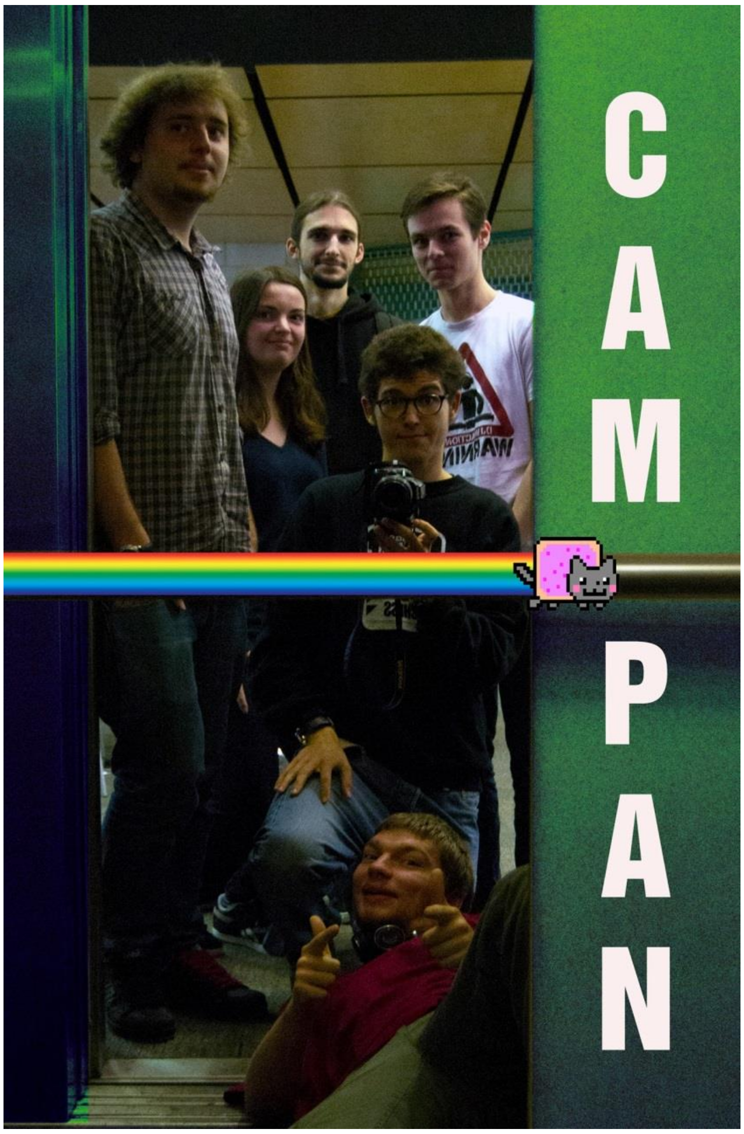
Volunteer Database & Mailing List

- Final Report -

Team **CAMPAN**



Andrei Ilisei
Ana Dumitraș
Kristian Krastev
Milan Zolota
Nikolay Nikolov
Patrick Johnston

TABLE OF CONTENTS

1 Introduction

Our client for this project is Bristol City Council Museums, Art Galleries and Archives.

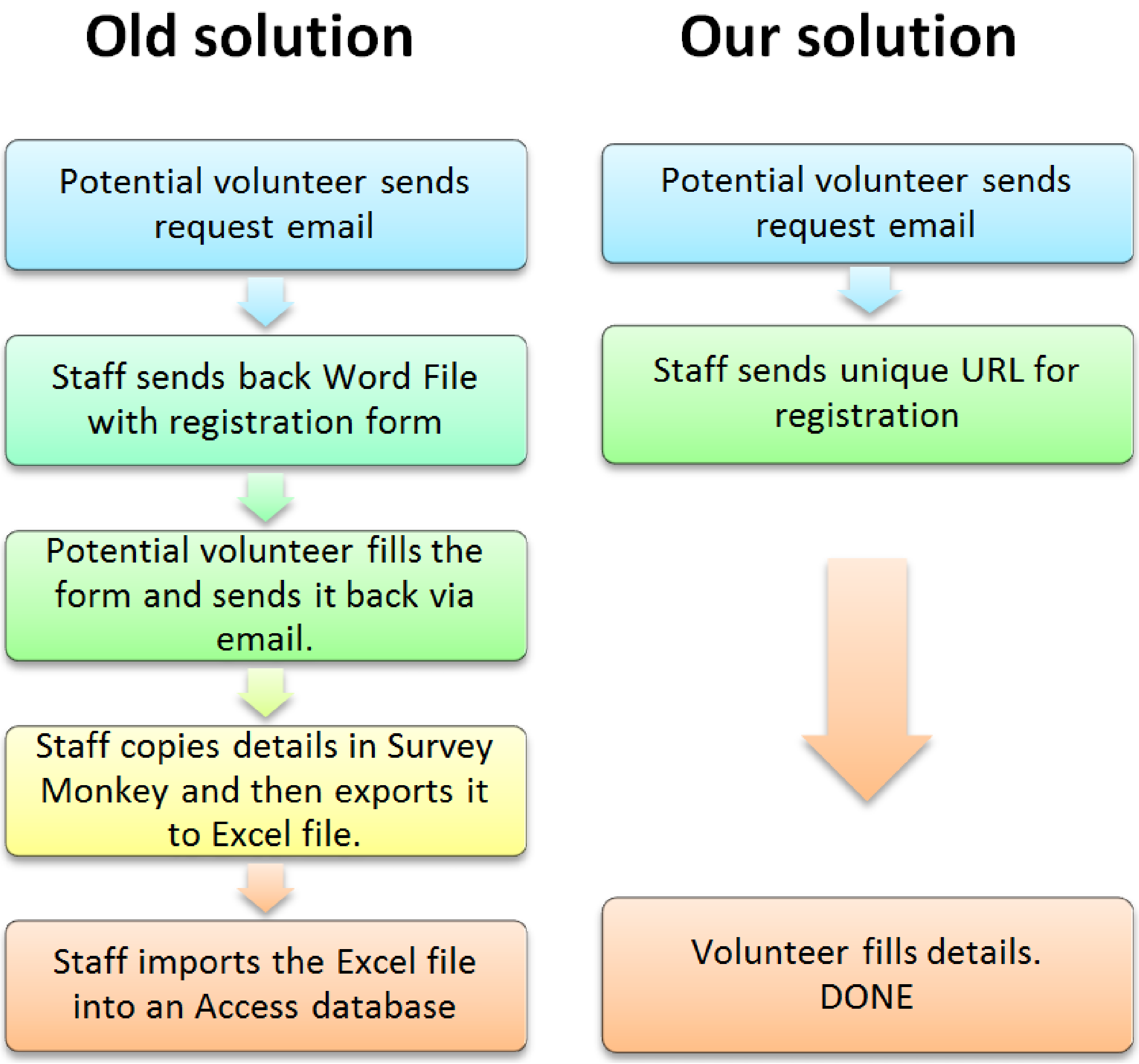
The volunteer database management and registration system used by them is out of date, inflexible and entirely non-scalable.

Currently the process of registering a new volunteer involves the volunteer sending an email requesting to register. A Microsoft Word document containing the registration form is then sent to them to fill in and email back. After receiving the completed form, the team that manages volunteers has to manually insert its content into Surveymonkey. The data is then exported as a Microsoft Excel spreadsheet, which is finally merged with a master Access database using a VBA macro.

We took up the task to create a software product that allows our clients to store the database on a server and manage it using a user friendly web interface providing a simplified registration process for both admins and volunteers.

We came up with a solution that overcomes all the difficulties and issues mentioned above. We focused mainly on improving the overall experience of both our client's volunteer management team and the volunteers themselves. We also made sure that our product is robust and complies with all the web security standards and the Data Protection Act 1998 as required by .

Our product is comprised of the classic LAMP stack. The system is hosted on a server running a Linux distribution, the HTTP requests is handled and served by Apache, the database management is handled by MySQL, and the website is written mainly in PHP.


Old solution
Potential volunteer sends request email
Staff sends back Word File with registration form
Potential volunteer fills the form and sends it back via email.
Staff copies details in Survey Monkey and then exports it to Excel file.
Staff imports the Excel file into an Access database
Our solution
Potential volunteer sends request email
Staff sends unique URL for registration
Volunteer fills details. DONE

2 Description

2.1 Operation

The system consists of two perspectives: the main one is to serve the needs of our clients for database administration, another one is designed for volunteers. Each provides a different interface and functionality.

The Admin perspective enables the volunteer management team to:

- view information about volunteers
- send e-mails to volunteers/groups of volunteers
- register and delete users
- create and delete sub admins
- manage sub admins' permissions
- edit volunteer's profile
- edit the registration form
- create groups and allocate volunteers to them
- generate statistics on fields from the database
- create and edit events, and allocate groups to them
- keep track of a changelog functionality
- create, delete and restore backups of the database
- access and edit a contact page

The Volunteer perspective lets volunteers:

- register in the database
- edit their profile
- view information about events
- access a contact page

2.2 Purpose

The purpose of the Volunteer Database Management Website and the server-side application it interfaces, is to combine all of the above functionality and interface into a single software product. This product will facilitate our clients in managing their volunteers and make the process of doing their job more efficient.

2.3 Target audience

As indicated by the two perspectives the product is aimed at two groups of users: the volunteering management staff-team at the Bristol City Council Museums Archives and Art Galleries and the people volunteering for them.

The first one - that of the staff-team, is in charge of administering volunteers. It is their responsibility to find volunteers that match certain criteria and assign suitable tasks that to them. They need to provide a list of volunteers to event organisers that need them - without disclosing any sensitive information. The people in the volunteer management team have moderate IT skills and thus the web-based solution we developed should meet their needs perfectly. We believe that they would have no problems using the system as it provides a user-friendly graphical interface for the use of all of its functionalities, combined with a help-tip feature to avoid any confusion.

The second target group consists of volunteers of various age-groups and social backgrounds, helping at various events when requested by museum's staff. The level of IT competence among these people varies greatly. Having said this we believe that using our web-based system designed for volunteers is in no way more complicated than the previously used one. On the contrary, the new registering process requires less effort and e-mailing, plus a volunteer would now be able to review their profile and information about upcoming events they can take part in.

2.4 Availability

The system can be used by any internet-connected device with a sufficiently modern browser. Nevertheless, it is not a public website, but rather a specialised one, accessible only to the above target groups provided they have credentials for it. The core element of the product - the volunteer database - can only be accessed by the staff of Bristol Museums and by any individuals whom they decide to grant access to using the user management facility of the system. Similarly a non-admin can only access the volunteer perspective if they have received a unique URL and successfully registered on the system.

All of our website's features (from an admin's perspective) are available for You to view and test using the following credentials at www.campan.tk:

- **username:** staff@bristol.ac.uk
- **password:** test

3 Overview

3.1 Original plan

When we initially considered taking up this particular project we almost immediately agreed that the solution to the problem at hand must be web-based. At first it seemed that it would not be a project too difficult to complete apart from the fact that we had little experience with most of the technologies involved in its implementation - namely PHP, HTML, CSS, JavaScript, jQuery and MySQL. Since the opportunity to get familiar with them and acquire new programming skills was rather appealing to us this did not present a problem. Considering this, as well as the diversity among all the proposed projects we had to choose from, it was clear to us that in order to obtain a good result for the implementation of the Volunteering Database & Mailing List, we needed to do our best and implement as much additional features and functionality that could help make the final product stand out. As we advanced through the working process we found out that not all of our initial expectations matched reality and that some of the tasks we had to complete required a lot more work and attention than previously envisaged. Nevertheless we still managed to both implement both the core functionality and the we had in mind and according to our clients the product goes beyond meeting their basic needs. Overall we learned a lot about web technologies, and databases and we gained precious experience in working on a real life project for the needs of actual clients.

3.1.1 Initial proposal

3.1.1.1 Why this project?

We would like to do the Volunteer Database & Mailing List project for several reasons.

To begin with, we all believe that databases are one of the most important parts of computer science and we would like to explore this area as a team and gain experience in using it in big projects. This will also prepare us for our Databases unit and will surely give us a better perspective of it.

As our lecturers have mentioned quite a few times, the most important part of a project are the very basic functions. However, we feel that the difference between a very good and a great project is how well the secondary functions are implemented, as well as the core functionality. In this case, the client has provided us with mostly basic functions, which will give us a lot of creative freedom, and we would like to show that in addition to being able to provide what the client wants, we can also add our own improvements. We will show that we are not just programmers – we are computer scientists who can do more than just write code.

In addition, we are particularly interested in the fact that this project involves using a number of platforms. This will make us more versatile and give us insight into different applications that will prove quite useful for our careers in the future, when we are responsible for applications that interact with others, using a number of different methods.

3.1.1.2 Implementation idea

Our team suggest that a web application interacting with a new MySQL database should be created. This application would be able to:

- Register new volunteers (this would enable new volunteers to register directly, without having to ask for a Word document form and sending it back)
- Handle new volunteer applications (this would essentially enable the Council to skip the whole “Word-to-Excel” procedure that uses Surveymonkey and is done manually. Adding a new volunteer to the database would be a matter of one click of a button, instead of having to handle Word and Excel documents)
- Import entries from the old Access database to the new MySQL database
- Import entries that are piled up on the Surveymonkey page to the new database
- Show sensitive information (e.g. equities information) only to users with privileges
- Show non-sensitive information to all members of staff
- Manage current entries in the database (e.g. adding, editing, deleting an entry, etc…)
- Provide statistical analysis, such as plotting figures over time, automatically

3.1.2 Initial Schedule

We managed to work efficiently and finish most of the tasks before their deadlines, as provisioned by the initial schedule. In some cases we postponed some tasks due to realising that there are some dependencies causing the order of implementation to change, that we were not aware of in the beginning; this was the case for implementing the backup system for the database and scripting the installation package.

Our initial schedule can be found in **Appendix B.**

3.2 Extensibility and Improvements

Fortunately we were able to implement all the features initially requested by our clients. In addition to this we came up with extra functionalities, design solutions and features by trying to anticipate our clients' needs.

In the case where any upgrades or updates for the system are required, this can easily be achieved by us or another developer due to the structure of our chosen interface architecture (model - view - controller). We also did our best to thoroughly comment the code and make it as readable as possible.

3.3 Extra functionality with regards to the initial SRS

- One of the initial improvements was to make the registration form, which defines the structure of the volunteer table in the database, dynamically editable by admins using a graphical user interface rather than hardcoding it so that it could only be changed through altering the code or the database itself. This functionality which our clients and we decided on later during the development process required a lot of change to the structure of our database and website, which though not easily we were able to cope with and ultimately implement.
- We extended the functionality of the Changelog page so that an admin can search for specific actions or users.
- We developed a page that displays upcoming events that can both be seen by volunteers and managed by admins with the possibility for creating and assigning groups to them. It is quite similar to a blogging platform with reduced functionality.
- Last but not least we extended the types of permissions that sub-admins can be granted in order to give a more flexible set of options for master admins.

3.4 Uncompleted functionality with regards to the initial SRS

- Our clients decided that they do not want to import their old database - they preferred personally getting in touch with everyone on the mailing list in order to confirm their participation as volunteers and make sure that the information stored about them is up to date by making them use the new registration process introduced by our system.
- It was originally planned to implement a secure HTTP connection using SSL/TLS. However, our clients and we decided that this would be excessive, given the privacy of the required unique URL sent via email for registration, the small number of people knowing about and using the system and the small possibility for a safety hazard situation occurring. Another reason for not implementing it, was the fact that the clients needed to pay for a certificate for us to be able to do so, which they were not interested in doing.

4 Development Process

4.1 Getting started / Approach

We started this project having little to no experience with the tools that we decided to use. Apart for the fact that Patrick Johnston had done a small blog project in PHP using Codeigniter and Milan Zolota had some experience with web development, it was the first time that we were involved in a full-on web-based project requiring a database, a server-side application and a front end interface to function together to deliver a complete content-management software system.

One of the most difficult problem that we encountered in the beginning was the fact that our clients did not have an exact idea of what they want from the product. All we knew is that they needed a better system for managing their volunteers. We had to come up with all the design of the website by creating a few design prototypes and letting them choose what they prefer and how, if so, they would like it to change. There was also a fair amount of functionality that they actually needed, but were initially unable to formulate, as they lacked technical knowledge needed to conceive of the full spectrum of possibilities available to them through our services. We made sure to aid them during this process and offer the best solutions even when they were beyond our aptitude at the time. Since this was the first major project we had worked on, the task of designing the website and its functionalities from scratch was harder than it seemed at first.

We all went through a few crash courses on PHP, HTML and CSS in order for the development process to kick-off as fast as possible. Some of us became more proficient in the use of particular technologies than others and vice versa, which enabled us to work concurrently on different aspects of the system. The most useful resource that we used for this was: http://codeacademy.com/. This approach turned out quite efficient, allowing us to quickly develop a prototype of the website that could be presented to our clients. The learning process never ended as we needed to implement more and more functionalities that needed advanced features of the programming languages that we needed.

In order to be able to present demos to our clients and to be able to immediately test different sections of our system we set up a publicly accessible server. We also used a free domain to access our website: http://campan.tk/. This way, our clients could track our progress at any time.

4.2 Distribution of work

We divided the workload in small tasks such that they had as little interdependencies as possible. This tactic helped us set several internal deadlines for accomplishing tasks that ultimately enhanced our capacity of getting things done.

Assigning the tasks was an interesting process since we had to take into account the skills needed for accomplishing it and the skills that each person had. Also, we tried to find the best assignment such that everybody does something that they enjoy working on. Usually 2 or 3 of us were assigned to a task but we always helped each other when we encountered difficulties.

Always knowing who is working on what increased our productivity and helped us keep track of the tasks that still needed to be finished. Also, it was a good way of knowing who is responsible for each section of the website.

4.3 Version control & Backup technique

With only six people on the development team, there was no need for specialised techniques to keep track of the development progress. We decided from the start that version control software would be excessive; the way in which work was distributed between members of the team ensured that no file conflicts would have to be resolved. Summarising changes to files to one-another, combined with the large amount of pair-programming between us and group work meetings resulted in no need for version history or a diff engine.

We made regular backups to our work by zipping everything and storing it both on the server and on our computers. Partial backups were done by each of us every time we tried any major change to the functionality of the website. Alternatively a sandboxing technique was sometimes used to upgrade to new functionality by preserving existing versions of files and working on copies of them.

4.4 Communication & feedback

Communication with clients has been done through various channels:

- We had meetings with the clients once or twice for each release.
- For urgent matters we emailed or texted them
- For non-urgent matters that needed for them to meet and discuss the options we used www.basecamp.com, a project management website. This was the main channel of communication since most of the time we needed their feedback on a recently implemented feature.

4.5 External Tools and Plugins

- **CodeIgniter:**
  - Source: http://ellislab.com/codeigniter/
  - License/ Copyright: Free to use, copy, modify, and distribute;
  - Description: Rapid development, PHP-interactive web-development framework, including many useful, lightweight and quick classes necessary to implement much of our low-level functionality. Highly regarded and extremely well documented.
- **Highcharts.js** - jQuery plugin for generating charts
  - Source: http://www.highcharts.com/
  - License/ Copyright: Free for educational purposes and non-profit projects; Creative Commons Attribution-NonCommercial 3.0 License;
  - Description: One of the best plugins that we found for graphically displaying statistical data; it is easy customisable and has a large set of options that can be chosen for a custom graphic.
- **Validation.js** - jQuery plugin for input form validation
  - Source: http://bassistance.de/jquery-plugins/jquery-plugin-validation/
  - License/ Copyright: MIT License;
  - Description: In order to ensure that the volunteers submit their details correctly, a plugin is used to validate all possible input in the registration form and the user profile page. The plugin is used for concrete fields such as postcodes, dates, etc.
- **Date Picker** - jQuery plugin for graphical picking dates from a pop-up calendar
  - Source: http://jqueryui.com/datepicker/
  - License/ Copyright: MIT License;
  - Description

4.6 Agile nature of approaching tasks

During the development process we had to rewrite some sections of the website due to the change in the clients' requirements; the whole registration form was changed because they wanted it to be editable which resulted in us implementing a system that manages and dynamically generates it.

A typical timeline of implementing a feature would consist of the following steps:

1) Design feature and identify dependencies;
2) Implement the feature;
3) Test it;
4) Request feedback from clients;
5) Implement changes if requested, go back to 4.

At each stage of the development process we reviewed what we had implemented and tested thus far and what we still needed to develop and test. When starting or finishing a new task, we would check the SRS to make sure that what we are implementing corresponds to the specifications and when in doubt, we contacted the clients.

4.7 Bug tracking and identifying

Thoroughly testing each component or function of the website after implementing it enabled us to avoid any major bugs. Most of the bugs were identified using relevant print statements or inspecting the source code using the browser utilities.

In order to keep track of the bugs discovered we created a shared Google Drive document that was constantly updated. Every time we found a problem we took notes down in that document including a small description of it. This also allowed us to discuss possible fixes for it and find the best solution.

5 Tasks

5.1 Learning the languages that we used

Collaborators: Everyone (25 hours each)

Description:
Most of us had little-to-no experience with the technologies that we were going to use; we spent some time getting used to the tools that we needed, finding their respective advantages and disadvantages and deciding between us which ones best fit our needs.

We each had some basic knowledge of HTML and CSS, to the point that we knew what the common tags did, and understood some of the syntax. However, never-ending was our research into some of the quirkier parts of these languages, and new discoveries were uncovered every day. We decided that both HTML and CSS were simple enough not to need a framework to help development, nor a WYSIWYG editor.

MySQL was a solid choice for the RDBMS, Oracle has a well-regarded documentation. As with PHP, it is of the most widely used SQL variants, and therefore has plenty of online code-snippets and Q&A.

None of us had any experience with Javascript, this was the most unfamiliar territory. Whilst it is extremely popular – being the predominant client-side scripting language – it is least resemblant of languages we have had previous experience with. The extent to which we use Javascript will be simple tasks, written in either pure Javascript or with the assistance of jQuery, and importing plugins, integrating them to work with our website as intended.

We settled on using PHP for the scripting language used for generating the web-pages and accessing the database; it is the most commonly used language for such purposes and therefore has the greatest number of code examples, resources and a good documentation. We agreed to use the web-development framework CodeIgniter. CodeIgniter uses the model-view-controller interface architecture, a structure that is new to our team and took some getting familiar with; but after some time, we reaped the benefits. CodeIgniter has helpful classes and functions, to name a few: pagination (creating pages of a certain number of items, like a forum thread), secure file uploading, automatic form validation. The most widely used and conducive class was the database class; with minimal effort, it will protect the database from SQL injections and cross site scripting, avoid mixing SQL code with PHP code, and ease the transition to a different RDBMS should the need arise.

5.2 Registration page and Editing it

Collaborators: Andrei (61 hours), Nikolay (63 hours), Patrick (8 hours)

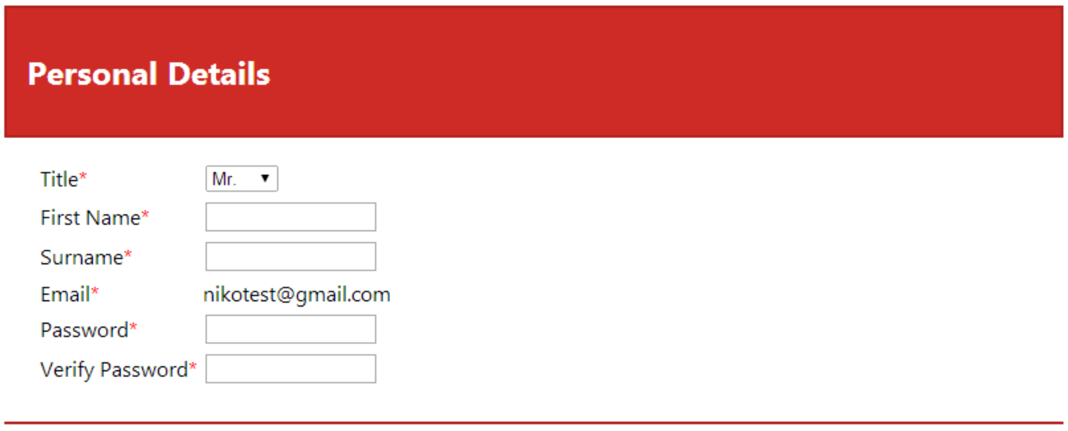
URL:

- www.campan.tk/registrationNew?unique=123asd - for registration form
- http://campan.tk/category/ - for editing the registration form

Description:
Volunteer registration is divided into two tasks: creating (and editing) the registration form and automatically generating the registration page based on the structure specified by the admins.

In order to achieve better structure of the registration form and ease the process of giving permissions to sub-admins to see different information about the volunteers, we split the registration form into categories. Each category is displayed in a box and can have various input fields. The input fields can have one of several types such as textbox – big or small –, dropdown menu, checkboxes etc.; this way, the registration form is fully customisable, giving our clients the opportunity to update it according to the required needs at any time.

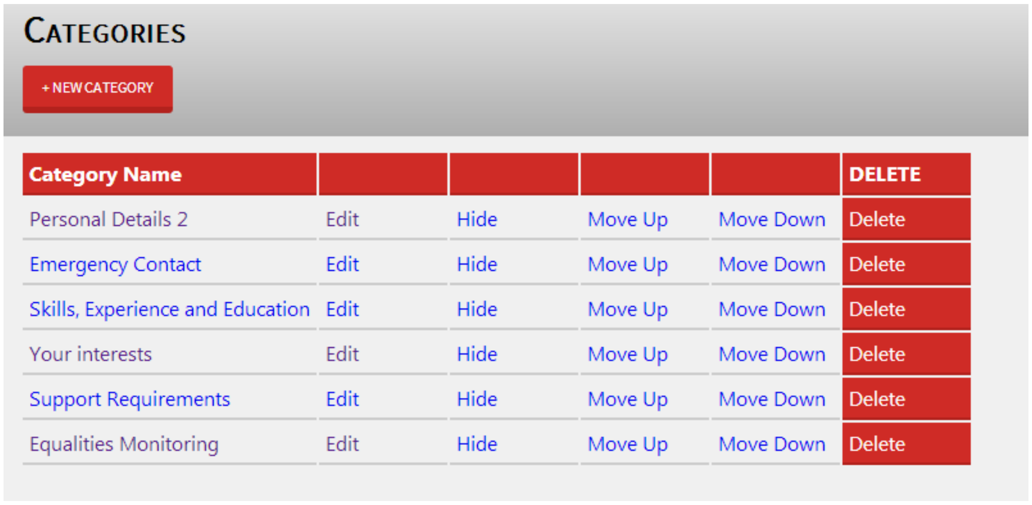
By default, there is one non-editable category (named 'Personal Details'), which contains the basic information about the volunteer, including their login details – their email and password – with which users may access the website.



Personal Details

Title*	Mr.
First Name*	
Surname*	
Email*	nikotest@gmail.com
Password*	
Verify Password*	

New categories can be created in "Manage Registration Form". The first page in each section lists all of the current categories and allows the admin to create new ones. There are several other actions that can be performed by the users: the order in which the categories appear on the registration page can be altered; modifying the visibility of categories and permanently deleting them from the database. All these things can be done at the click of a button.

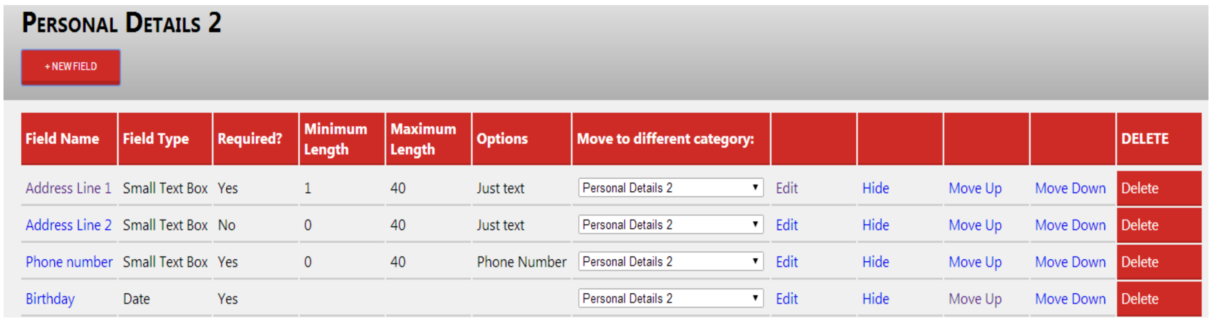


Categories

+ NEW CATEGORY

Category Name					DELETE
Personal Details 2	Edit	Hide	Move Up	Move Down	Delete
Emergency Contact	Edit	Hide	Move Up	Move Down	Delete
Skills, Experience and Education	Edit	Hide	Move Up	Move Down	Delete
Your interests	Edit	Hide	Move Up	Move Down	Delete
Support Requirements	Edit	Hide	Move Up	Move Down	Delete
Equalities Monitoring	Edit	Hide	Move Up	Move Down	Delete

Categories can also be edited. Similarly to the previous page, fields in a certain category can be moved, shown, hidden and deleted.

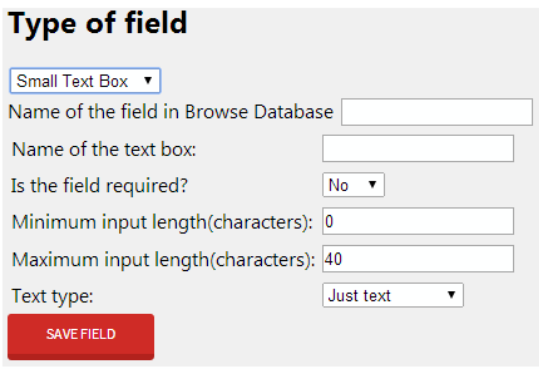


Personal Details 2

+ NEW FIELD

Field Name	Field Type	Required?	Minimum Length	Maximum Length	Options	Move to different category:					DELETE
Address Line 1	Small Text Box	Yes	1	40	Just text	Personal Details 2	Edit	Hide	Move Up	Move Down	Delete
Address Line 2	Small Text Box	No	0	40	Just text	Personal Details 2	Edit	Hide	Move Up	Move Down	Delete
Phone number	Small Text Box	Yes	0	40	Phone Number	Personal Details 2	Edit	Hide	Move Up	Move Down	Delete
Birthday	Date	Yes				Personal Details 2	Edit	Hide	Move Up	Move Down	Delete

When creating a new field, admins can specify its type (Drop-down menu, Small/Big Text Box, Text, Date, Checkbox) and restrict the input depending on the field.

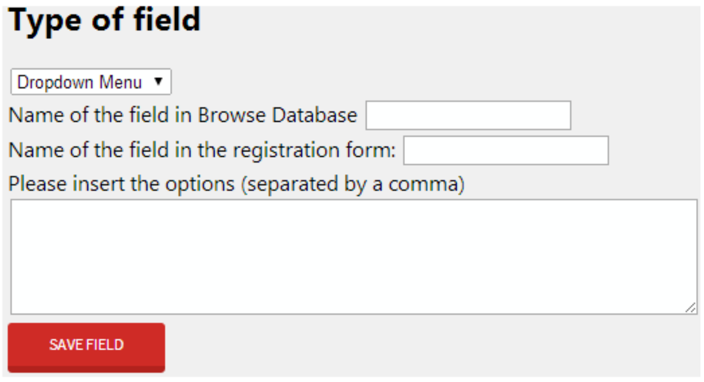


Type of field

Small Text Box

Name of the field in Browse Database	
Name of the text box:	
Is the field required?	No
Minimum input length(characters):	0
Maximum input length(characters):	40
Text type:	Just text

SAVE FIELD



Type of field

Dropdown Menu

Name of the field in Browse Database

Name of the field in the registration form:

Please insert the options (separated by a comma)

SAVE FIELD

Since admins can edit volunteers' profiles, they can optionally create an invisible field and leave their comments for the other admins to see.

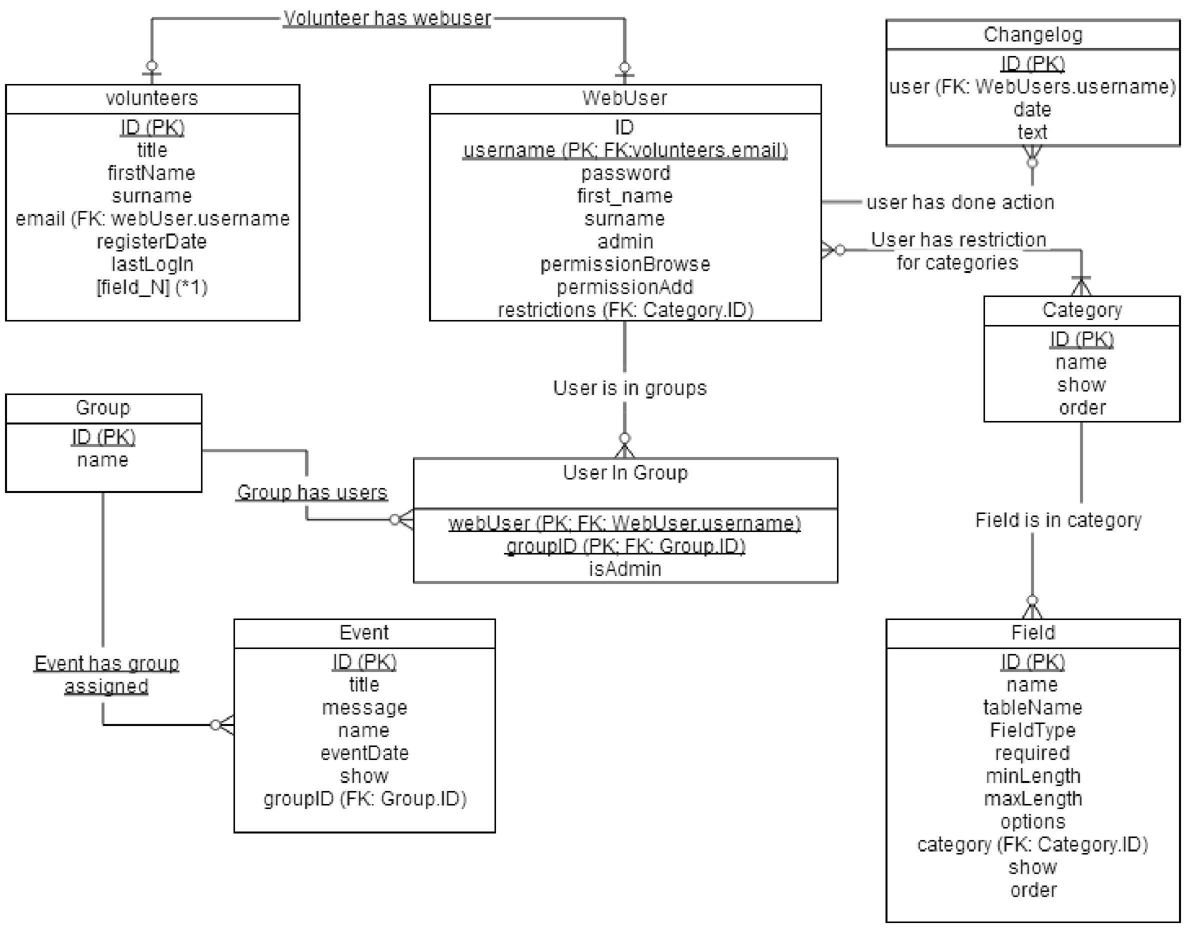
The registration page, including input verification, is generated automatically based on the current set of visible categories and fields.

One of our clients' wishes was to keep the database exclusive to people who have contacted them personally in order to ensure that they can monitor the database more easily. In order to achieve this, unique URLs are generated for every person who has requested to join the volunteering team and it is impossible to sign up on the system without a unique URL.

5.3 Database creation

Collaborators: Andrei (3+ hours), Milan (1+ hour)

Description:
Creating the database was a continuous process. What started as a database with 2-3 tables and just some relations between them, ended up consisting of 10 tables with numerous relationships in order to be able to store all data needed for a good website functionality. We also had to take into account all the security and ease-of-access aspects when creating new tables.



The **Volunteers** table stores the volunteers' personal details. Some columns, such as title and firstName, were manually created, while others are generated automatically (field_N).

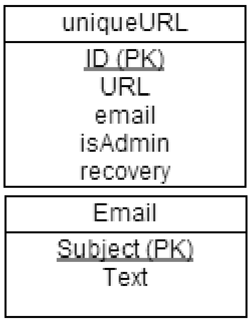
WebUser stores credentials and permissions about all users of the website (both volunteers and admins). The *admin* field indicates whether the user is an administrator. *permissionBrowse* represents the permission for full access to the volunteers, while *permissionAdd* represents the permission for generally administering the website. Restrictions attribute contains the IDs of the categories restricted to the admin if *permissionBrowse* is FALSE.

The **Group** table stores the name of every group together with their ID. **User in Group** table shows the relationship between users and groups - every row (user, group) in the table represents that the user is a member of the group. If a user is an admin it means that that admin has permissions to manage the group and in the same, see details about the volunteers in that group.

The **Category** and **Field** tables are used for storing the configuration and structure of the registration form and all this information is used for automatically generating it. *Show* marks the visibility of the category/fields, *order* stores the position of the object in the registration form. The rest of the columns are used for storing only details about the field and input restrictions.

The **Event** table stores all details about upcoming events that need volunteers.It also stores the ID of the group of volunteers who attended that event for admins' future reference.

Changelog keeps information about each and every action that a user has done on the website.



UniqueUrl stores the unique URL codes for registering and password recovery paired with the email of the user to whom they belong.

The **Email** table stores a list of predefined email templates which can be used for bulk emails.
.
The last two tables are completely separate from the rest of the database and have no relationship with other tables.

The advantage of the structure of our database is that the number of update/insert queries has been reduced to a minimum which increases the speed and efficiency. Most of the information will not be deleted or changed after it has been inserted.

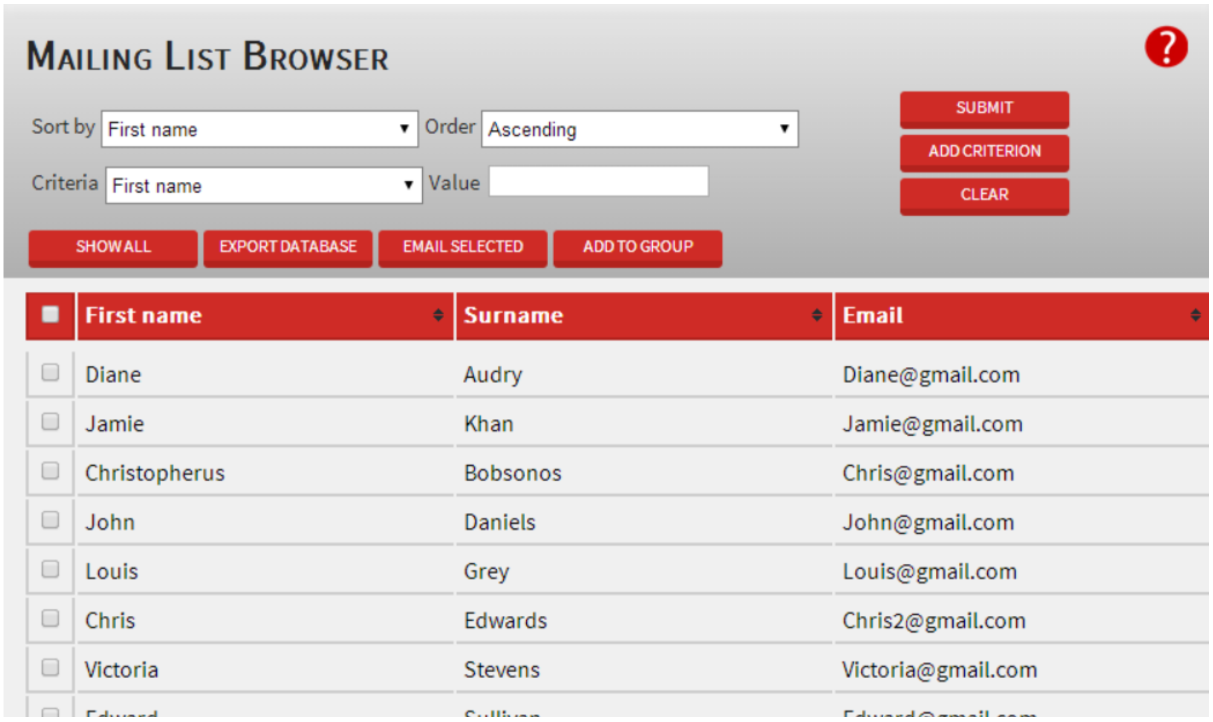
5.4 Browse database

Collaborators: Patrick (81 hours), Ana (22 hours), Kristian (39 hours), Milan (5 hours), Nikolay (5 hours)

URL: http://campan.tk/browse_database/

Description:
The Mailing List Browser or "Browse Database" is the replacement to opening up the old MS Access database file and seeking volunteers by hand. The new system incorporates the admin-subadmin hierarchy, thus, admins may see all volunteer data, whilst sub admins may not see certain fields, or can only see users within the groups the subadmin is, as designated in the user management.

Upon visiting the Mail List Browser, one sees the names and email addresses of the volunteers in a statically scrolling table with the header and menu being fixed to the top of the browsing window for convenience, giving the sense of a browser application interface rather than a simple web page.



Sort by: First name; Order: Ascending; Criteria: First name; Value:

	First name	Surname	Email
☐	Diane	Audry	Diane@gmail.com
☐	Jamie	Khan	Jamie@gmail.com
☐	Christopherus	Bobsonos	Chris@gmail.com
☐	John	Daniels	John@gmail.com
☐	Louis	Grey	Louis@gmail.com
☐	Chris	Edwards	Chris2@gmail.com
☐	Victoria	Stevens	Victoria@gmail.com
☐	Edward	Sullivan	Edward@gmail.com

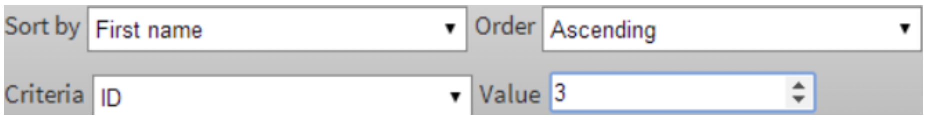
One may sort the table in ascending or descending order by any of the column headers, simply by clicking them.

The 'sort by' and 'criteria' drop-down menus have as options, all of the fields in a volunteer's profile. By clicking the 'add criterion' button, one may search using a conjunction of conditions which may or may not involve distinct fields used as criteria. Any criteria chosen for filtering is displayed alongside the table. Note, that the field used for sorting by use of the 'sort by' drop-down is not necessarily shown.

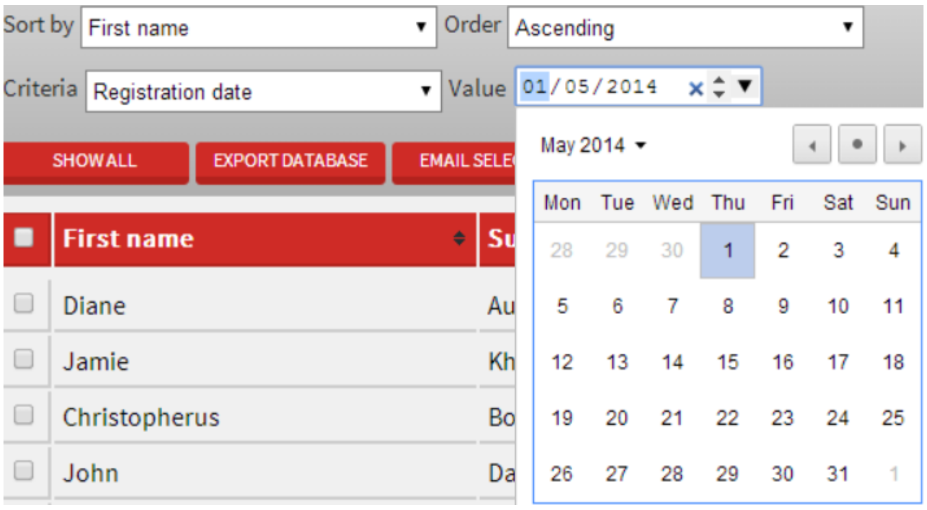
One may check some of the checkboxes – or check them all with the related column header – and click one of 'show all', 'email selected' or 'add to group' to use the selected volunteers as input to those functions.

Depending on the chosen criteria, a different type of input field is generated for the search input value to be entered, be it numerical, a date, an e-mail address, a drop-down menu or checkboxes.

Number input


Sort by: First name
Order: Ascending
Criteria: ID
Value: 3

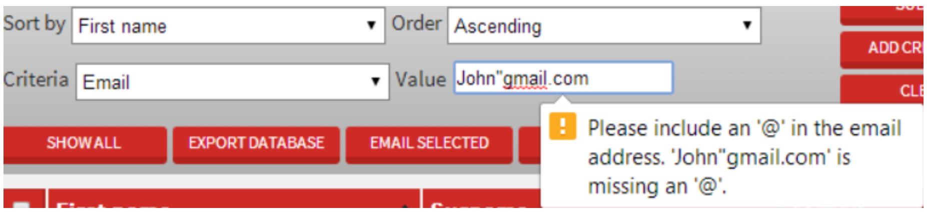
Date input


Sort by: First name
Order: Ascending
Criteria: Registration date
Value: 01/05/2014

Mon	Tue	Wed	Thu	Fri	Sat	Sun
28	29	30	1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	1

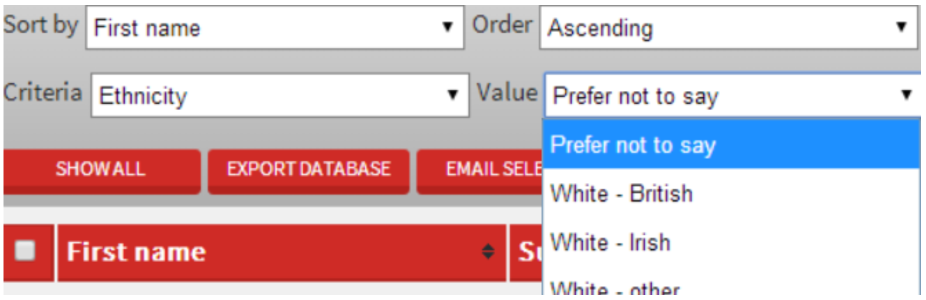
First name	Su
Diane	Au
Jamie	Kh
Christopherus	Bo
John	Da

Email input

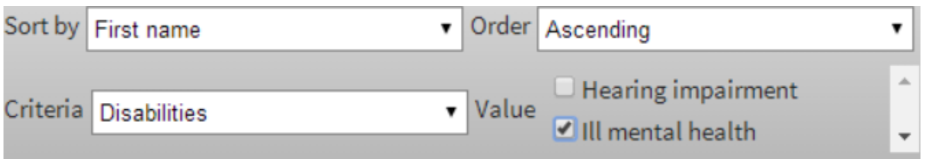

Sort by: First name
Order: Ascending
Criteria: Email
Value: John"gmail.com

Please include an '@' in the email address. 'John"gmail.com' is missing an '@'.

Dropdown menu

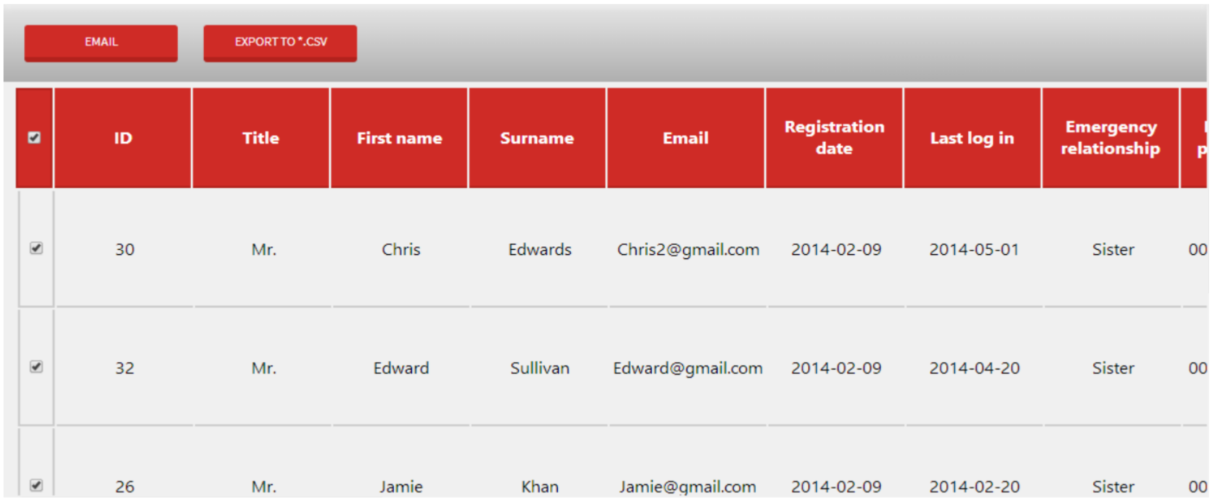

Sort by: First name
Order: Ascending
Criteria: Ethnicity
Value: Prefer not to say

Checkboxes


Sort by: First name
Order: Ascending
Criteria: Disabilities
Value: ☐ Hearing impairment ☑ Ill mental health

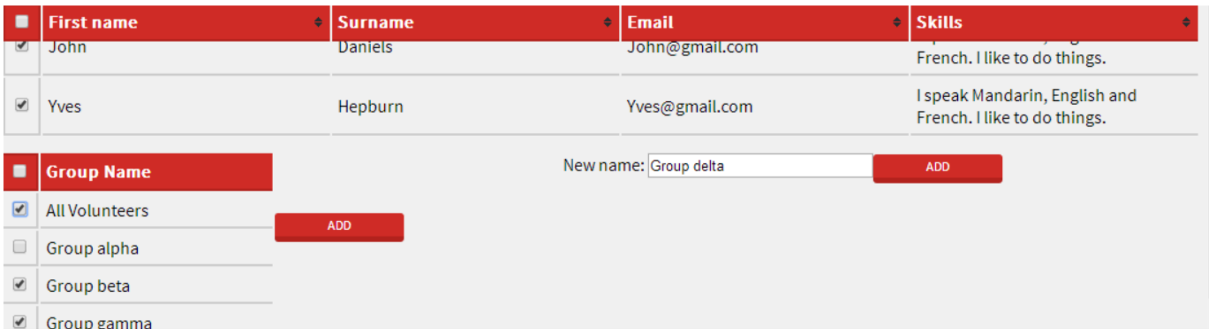
Checking multiple checkboxes within a single entry of a value will combine as a disjunction.

Clicking '*see more*' after selecting some volunteers will lead to a new window/tab that shows all the details about those volunteers that the currently logged in admin is able to see. This view has the ability to choose from this subset of volunteers – via an analogous checkbox system – and email those users, or export the generated table into a CSV file.



☑	ID	Title	First name	Surname	Email	Registration date	Last log in	Emergency relationship	
☑	30	Mr.	Chris	Edwards	Chris2@gmail.com	2014-02-09	2014-05-01	Sister	00
☑	32	Mr.	Edward	Sullivan	Edward@gmail.com	2014-02-09	2014-04-20	Sister	00
☑	26	Mr.	Jamie	Khan	Jamie@gmail.com	2014-02-09	2014-02-20	Sister	00

Clicking *'add to group*' after selecting some volunteers will generate a list of groups at the bottom of the page, one or more may be checked, and new groups may be created at this point. It is possible to navigate to this page from Group Management, in which case, a group will already be checked, depending on which group was being managed at the time that the 'Add volunteers' function was called.

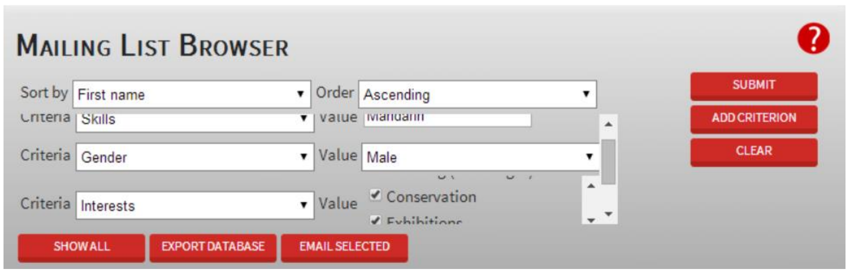


☐	First name	Surname	Email	Skills
☑	John	Daniels	John@gmail.com	French. I like to do things.
☑	Yves	Hepburn	Yves@gmail.com	I speak Mandarin, English and French. I like to do things.

☐	Group Name
☑	All Volunteers
☐	Group alpha
☑	Group beta
☑	Group gamma

New name: Group delta

As it is possible for there to be an arbitrary number of fields of criteria, it was decided with the clients that the list of criteria would need to be displayed as part of its own statically scrolling table.



MAILING LIST BROWSER

Sort by: First name — Order: Ascending
Criteria: Skills — Value: Mandarin
Criteria: Gender — Value: Male
Criteria: Interests — Value: ☑ Conservation ☑ Exhibitions

5.5 Menu

Collaborators: Milan (4 hours), Patrick (3 hours)

Description:
The menu uses a separate view (in our model-view-controller framework) and therefore it is independent of the rest of the website. This view is loaded every time on all the pages where it is needed. This way it was necessary to code the menu only once, instead of coding it for every page that displays it.

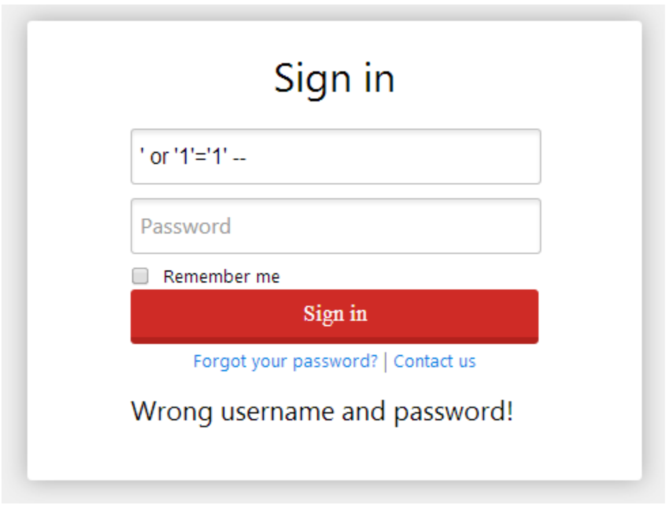
It is generated dynamically and its content depends on the type of the user that requested it and their permissions. For example, the content of the menu for volunteers is different to the content for admins.

The menu also acts as a small security feature, which checks if the user is logged in. Since all the pages that include the menu require the user to be logged in, this is just another line of defence.

5.6 Login page, Contact us page, Forgotten password page

Collaborators: Milan (15 hours)

Description:
These are most of the pages that are publicly available to anyone.


Sign in

' or '1'='1' --

Password

Remember me

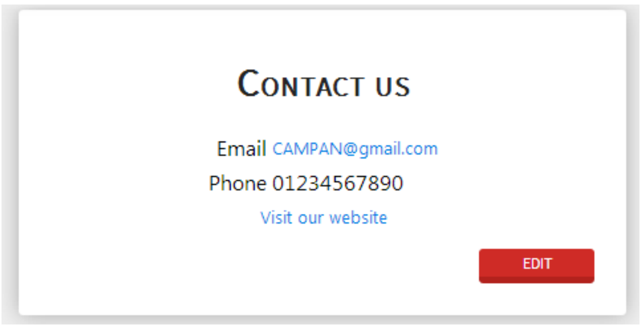
Sign in

Forgot your password? | Contact us

Wrong username and password!

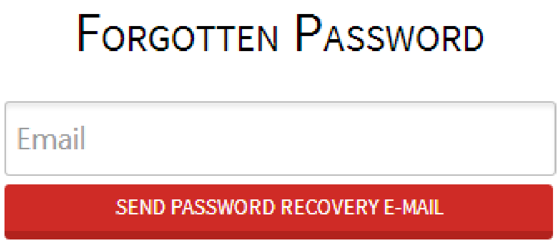
The login page is a page that can verify if a user is registered on our website. If invalid credentials are entered, an error message is displayed to the user. If they enter valid credentials of a user registered on the website, the are given access beyond the login page. By default, this is done using sessions, which expire after closing the browser, because of security reasons. But if a user ticks the 'Remember me' checkbox when logging in, cookies, which expire after a week, are used instead of sessions. If a user is already logged in and tries to access the login page, they are redirected to either the dashboard (for admins) or their profile page (for users).
The login system is protected from SQL injections.
The login page also contains links to the 'Contact us' and 'Forgotten password' pages.

The '**Contact us**' page is a simple, public page that lists an email, a phone number and a link. This page is public and can be accessed by anyone. Admins with the management permission are able to edit the email, phone number and link that are shown on this page. The purpose of this page is to provide both registered and unregistered users access to the contact details of the admin of the website.



Email CAMPAN@gmail.com
Phone 01234567890
Visit our website

The **'Forgotten password'** page gives users the ability to reset their password in case they forget it. After entering their email address on this page and submitting the form, they are sent an email with a unique URL. This URL leads to a page where they can choose their new password. This system is also protected from SQL injections.

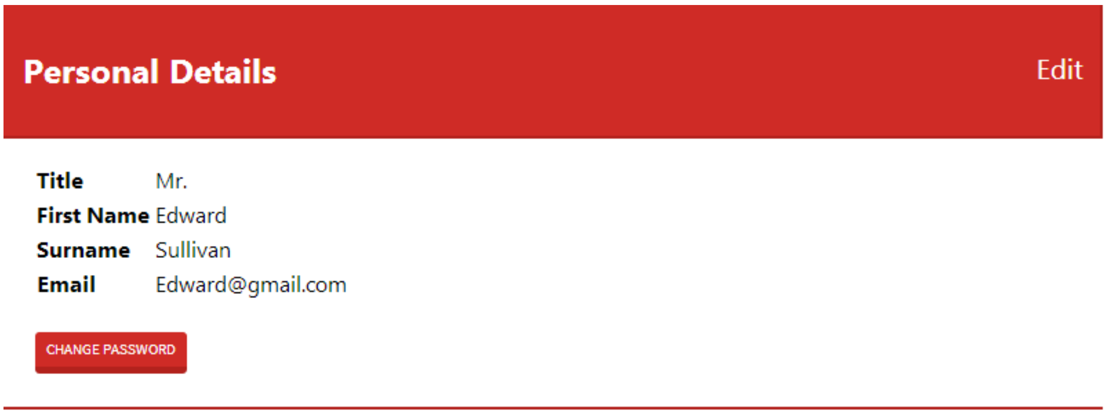


5.7 User profile page and Edit profile/details functionality

Collaborators: Andrei (32 hours), Nikolay (2 hours), Patrick (2 hours), Milan (6 hours)

URL: http://campan.tk/user_profileNew?user=25 (this is the profile of a random user)

Description:
A special page of the website has been dedicated to showing all the details contained in the database about a single volunteer. In terms of design, the page is similar to the registration form, but instead of having input fields, it displays what that user has already filled in at registration. The categories and the fields in the user profile page are generated automatically based on the current set of options defined by admins using the registration form management facility.

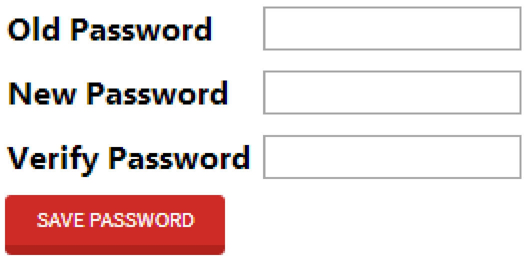


Title	Mr.
First Name	Edward
Surname	Sullivan
Email	Edward@gmail.com

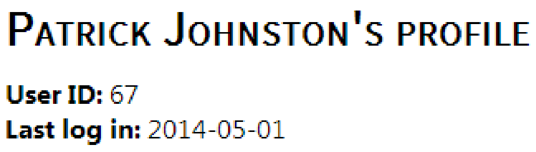
Special care has been taking for implementing the security and restriction measurements for this page. A user profile can be seen only by the volunteer it belongs to or by an admin that has permissions to see that volunteer's details. Furthermore, if an admin has permissions to see only a restricted set of details about a volunteer (e.g. no permissions to monitor Equalities), this information will not be shown to him, complying with the requested privacy standards.

On a volunteer's profile, there will be some extra information available only to admins such as events that the volunteer participated in or details stored in the fields hidden from volunteers.

Another feature implemented for this section of the website is the option to edit the profile. Clicking the "Edit" link at the top right of each category will redirect the user to a page, where they can change their details. A profile can be edited by the volunteer it belongs to, or by any admin that has the respective access. This feature is particularly useful for admins if they want to make comments about a specific volunteer, they can just edit a category that is hidden for volunteers and write their opinions there.



In this section of the website, it has also been made possible for a volunteer to change their password if requested on login. This procedure checks for the current password to match and hashes and salts the new password for security reasons.


Patrick Johnston's profile

User ID: 67
Last log in: 2014-05-01

In addition to the details mentioned above, the admins can also see the volunteer's unique ID and last log in date in order to monitor their activity.

5.8 Server set up

Collaborators: Milan (8 hours)

Description:
Because of the nature of our project, it was necessary to find a suitable web hosting. There were several options available, such as free or paid web hosting services or paid VPSs. In order to simulate the environment where the website was intended to run (our client's linux server), we decided to choose to use own VPS (virtual private server). This decision was also based on the fact that neither free, nor paid web hosting services provide enough freedom (superuser access to the server) for us, which we needed to see what exactly needs to be set up so that the website can properly run.

After buying a VPS, the latest version of Apache and MySQL needed to be installed from the official Ubuntu repositories and set up so that we could start building the website. To set Apache up, we needed to edit one of its configuration files such that http requests from campan.tk are served with our website. We also used Apache to prevent anyone but admins from downloading the backup files. As for MySQL, only a user and a database for our framework was required. The latest version of PHP was also needed on the server. We needed to use PHP v5.5 because of its newly included security features (password hashing and salting). Installing this version of had to be done manually because it was not included in the official Ubuntu repositories at that time.

Our team also extensively used phpMyAdmin for managing the MySQL database for our website. However, towards the end of the project, this tool was used less and less because our website gradually gained enough functionality to make phpMyAdmin almost redundant. At the end of our project, we only used this tool for tracking and fixing bugs.

Since we used the CodeIgniter framework, there were minor things that needed to be set up in its configuration files. The base URL (www.campan.tk) needed to be specified, the MySQL user and password for CodeIgniter needed to be specified and some libraries needed to be set to be autoloaded. The last thing that was optional, was removing the index.php that CodeIgniter inserts in every single of its URLs. This was done by editing the .htaccess file in the root folder of CodeIgniter to use Apache's URL rewriting module.

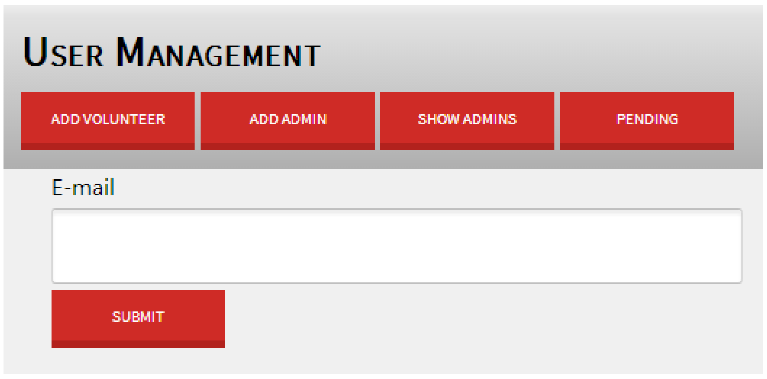
5.9 User management page

Collaborators: Milan (51 hours), Patrick (10 hours)

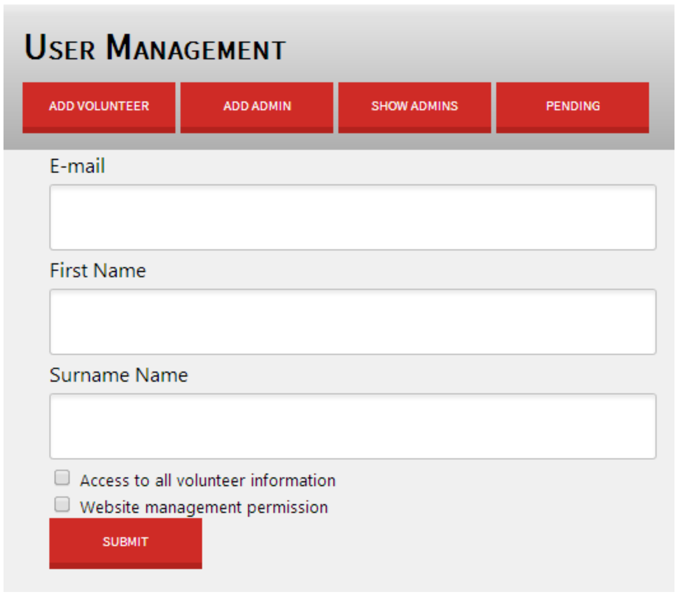
URL: http://campan.tk/user_management/

Description:
This page allows admins with the management permission to manage users. This page consists of five parts.

1. **Add volunteer:**
   This part of the page allows admins to add a new volunteer to the database. After entering a volunteer's email in the form and submitting, the admin's default email client opens a new email form with the volunteer's email address entered as the recipient and a unique URL entered in the body. After the volunteer receives the unique URL, they can use it to register on the website. This approach was requested by the client because they want to be able to edit the message they send to each volunteer as well as keep the message stored in their email client.



2. **Add admin**
   This part of the page allows admins to create new admins. After entering the email address, first name, surname, selecting the permissions the new admin should have and submitting the form, the admin's default email client will open a new email form with the new admin's email address entered as the recipient and a unique URL entered in the body. After the new admin receives the unique URL, they can use it to register on the website.



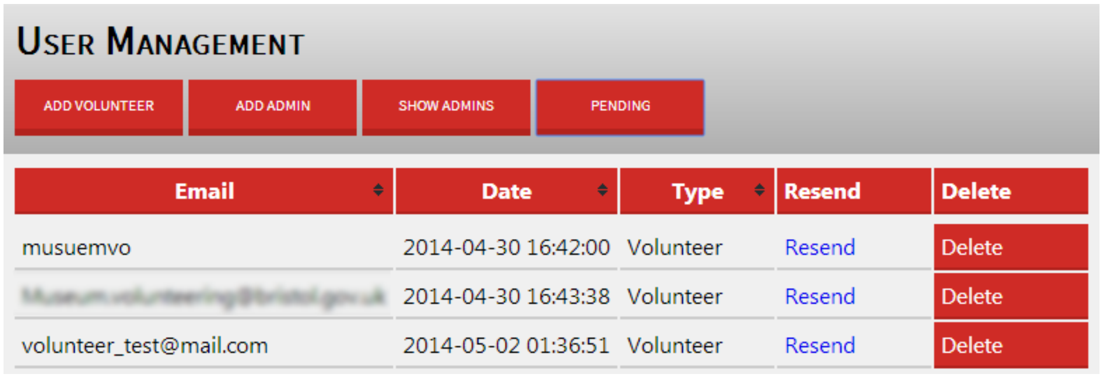
E-mail

First Name

Surname Name

☐ Access to all volunteer information
☐ Website management permission

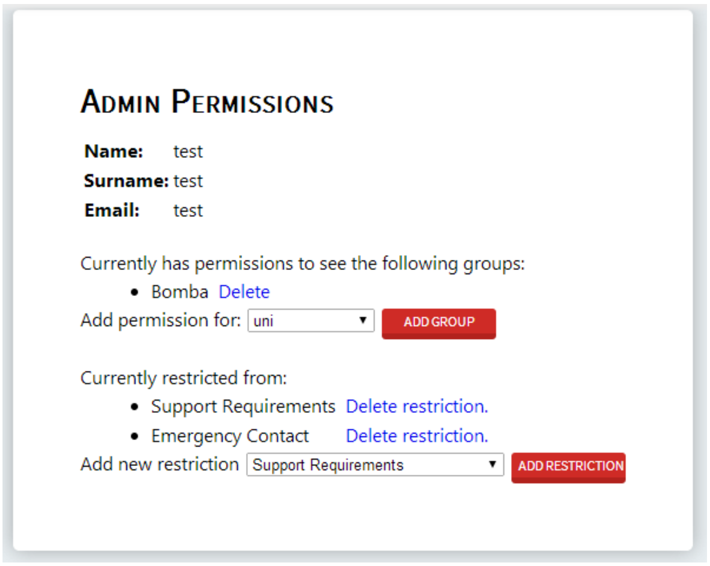
3. **Pending Registrations**
   This part of the page shows a list of pending registrations, i.e. users who still have not used their unique URL to register. An admin can delete the pending registration, so that the unique URL will not work, as well as resend the unique URL to its recipient in case they have not received it.



Email	Date	Type	Resend	Delete
musuemvo	2014-04-30 16:42:00	Volunteer	Resend	Delete
[redacted]	2014-04-30 16:43:38	Volunteer	Resend	Delete
volunteer_test@mail.com	2014-05-02 01:36:51	Volunteer	Resend	Delete

4. **Show admins**
   This part of the page shows the current list of admins and their privileges. An admin with the management permission can delete other admins as well as edit their permissions of which there are two types. The first - full volunteer access - allows an admin to see all the information about every volunteer in the database. The second permission gives an admin unlimited access to the whole website. This includes accessing this page, the manage registration form page, groups page, statistics page, manage backups page and changelog page. The permission also allows an admin to manage events and edit the contact details in the contact us page. By default, an admin without the full volunteer access permission cannot see any information about volunteers. Granting them said privilege can be done via the edit button in the group permission column.
5. **Editing group permissions and information restricitions (for admins)**
   This part of the page allows an admin to change the volunteer access of another admin who does not have the full access volunteer permission in two ways. Firstly, the groups of volunteers that the admin can access can be specified. Secondly,

categories of information that the admin should not be able to see can be selected. This functionality was specifically requested by the client because they believe that some information is quite sensitive and that it should be available to as few admins as possible.


ADMIN PERMISSIONS

Name: test
Surname: test
Email: test

Currently has permissions to see the following groups:
- Bomba Delete

Add permission for: uni ADD GROUP

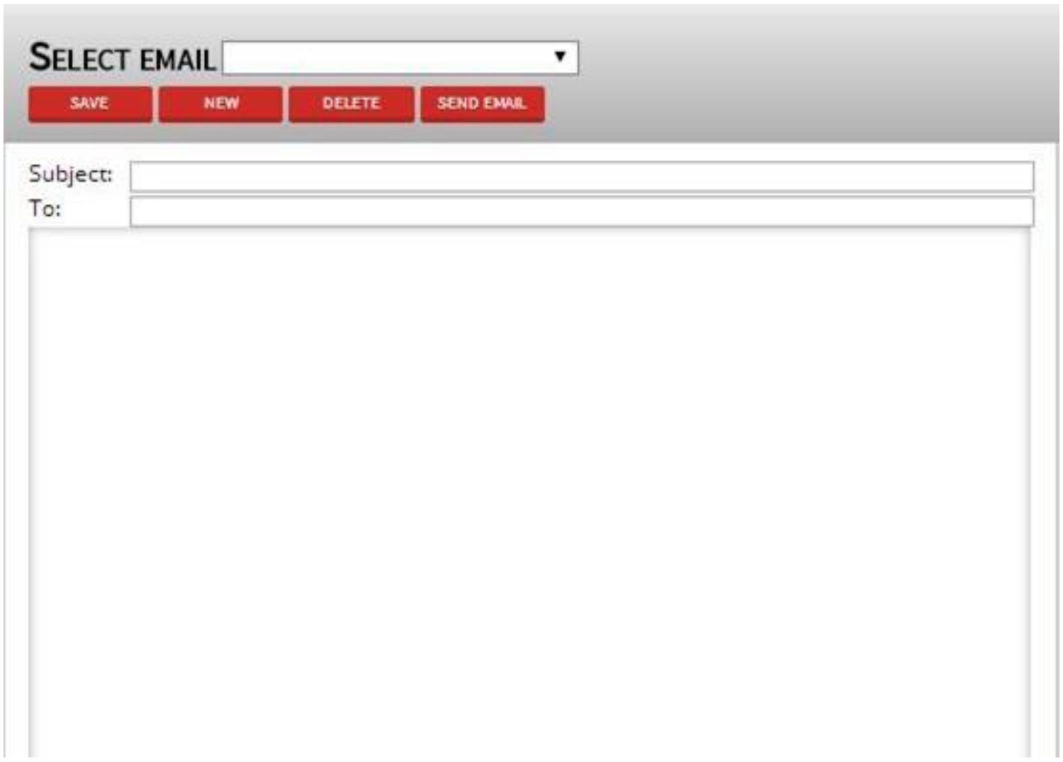
Currently restricted from:
- Support Requirements Delete restriction.
- Emergency Contact Delete restriction.

Add new restriction Support Requirements ADD RESTRICTION

5.10 Email volunteers page

Collaborators: Milan (5 hours), Ana (12 hours), Kristian (18 hours)

Description:
Admins can email volunteers by selecting the users they want to email from database browser, the “Show all” page, or the groups page and then pressing an ‘Email’ button. They will be redirected to the ‘Send email page’ shown below.



The admin can either select one of the email templates stored in the database or create their own. Templates already stored in the database can also be edited or deleted. To edit an already existing email, they have to select it, make some changes and the press 'Save'. This would update the entry with the same title in the database. For convenience and ease of access email templates must have a unique title. On saving a template with a title already existing in the database, a prompt is generated to make sure that overwriting its contents is truly the desired action to be committed by the user.

The email addresses of the people the user wants to email are already filled in but they can add more if they want or remove existing ones.

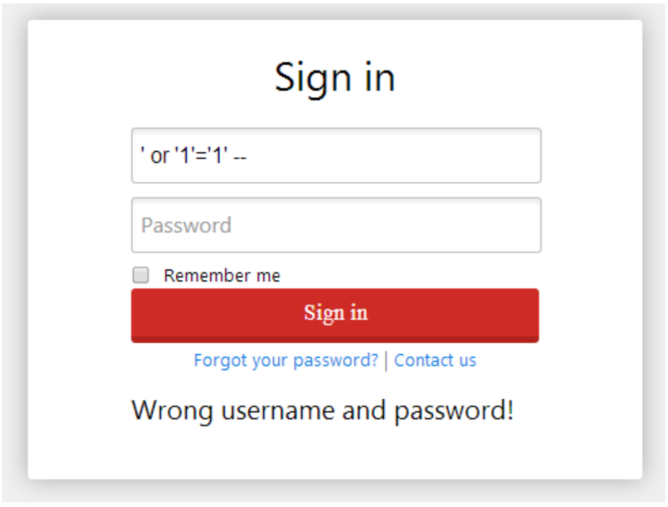
For sending email we use a mailto link. Pressing the 'Send Email' button opens the default email client. Emails are sent using Bcc (Blind carbon copy) in order to protect the privacy of the volunteers' email addresses. An additional copy of the message is sent to the official Bristol Museums address.

5.11 Security

Collaborators: Everyone (6 hours)

Description:
Security of the website was given a lot of attention because of the nature of the information the website contains. There are several things that were done in order to keep the information secure. These include:

1. All passwords are hashed and salted so that in the case they are leaked, the attacked will not be able to use them.
2. All forms are protected from SQL injections.
3. All forms that can be accessed by a volunteer are protected from XSS attacks.
4. All pages check for relevant permissions before they display any data to the user, so that a volunteer cannot access an admin page, or an admin without a relevant permission cannot access pages that require that permission(Manage users page, for example).



Wrong username and password!

5. It was made sure that files and folders on the server, such as the folders with the source codes, configurations files or the backup files cannot be accessed differently than intended (i.e. served by php after a http request). Also, the backup files cannot be accessed by anyone without the management permission.

5.12 Design

Collaborators: Kristian (90 hours), Milan (18 hours), Ana (17 hours)

Description:
Creating the design style and layout of the website was a continuous process that started almost at the very beginning of the development. Initially we simply needed a means to display and structure information generated on web pages which was mainly done using HTML. As we got closer to our first release and demos of our progress with the clients the need for a design solution emerged and what started as some styling attributes in our HTML tags needed to be transformed into proper CSS style files. Later on, the introduction of JavaScript and jQuery features was also required due to the dynamic nature of some of the pages and the need to create a satisfactory user experience.

Since a clear concept for the design requirements did not initially exist we were free to experiment with CSS which aided the learning process and enabled us to come up with prototype suggestions for the style by improvising. The CSS being the top component of the user side of web design seemed at first to be the easiest part of the project to implement. It turned out though, that producing aesthetic and well-functioning styling solutions is far from simple and straight-forward, especially for custom ones - i.e. without the use of ready-made templates provided by a framework (which happens to be our design approach).

The implementation of dynamic behaviour required us to use JavaScript and/or jQuery. Responsive behaviour on pages that need to change their contents according to user's actions, aligning fixed table headers with scrollable bodies on loading and resizing a web page, as well as other features of the graphical user interface could not have been made possible without the use of these tools.

All in all, a considerable competence in the use of the above technologies had to be acquired which resulted in both a product with an increased-quality user interface and some of us gaining valuable web design skills.

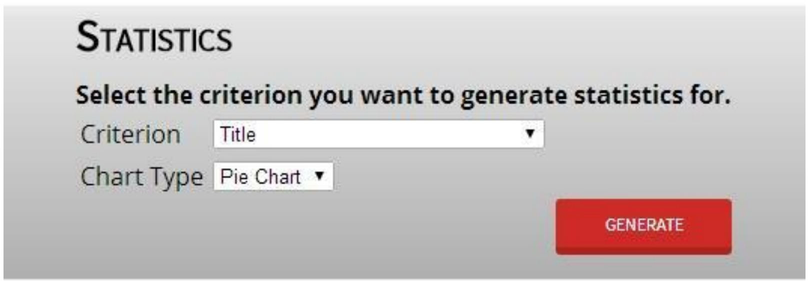
5.13 Statistics

Collaborators: Ana (20 hours), Andrei (3 hours)

URL: http://campan.tk/statistics/

Description:
The statistics generating utility on the website consists of two pages: one for selecting the criteria for generating visualised statistical information and the other one for displaying the result of that selection. The option of visualising statistics is only available to users who have admin privileges on the website. The button for accessing the page is located on the menu bar and, as mentioned above, is only visible for users who have permission to see it.

The first page only contains the menu for selecting criteria necessary for generating statistics. The user can select any field from the registration form and the type of chart he/she wishes to use.

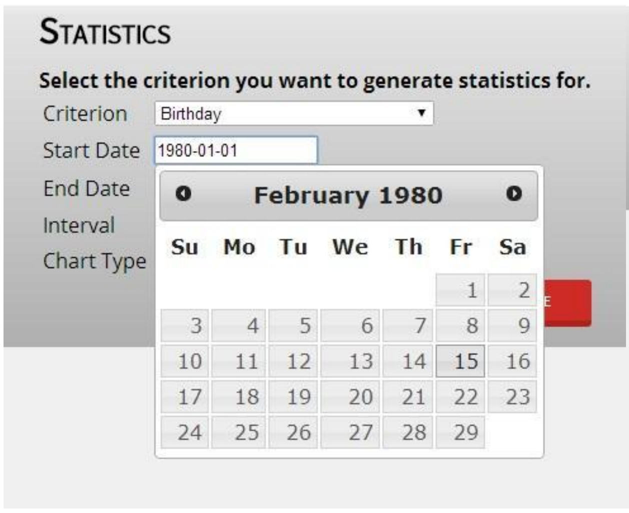


STATISTICS

Select the criterion you want to generate statistics for.

Criterion	Title
Chart Type	Pie Chart

There are 4 types of fields available depending on the selected criterion. The display of new fields based on the criterion selected is done in jQuery. Fields that have date type let the user select the start date, the end date and how to split the selected interval, either monthly or yearly. If the selected field is a checkbox or a drop-down list, statistics for all the options would be generated. The last type of field is a text field. When a field of this type is selected,



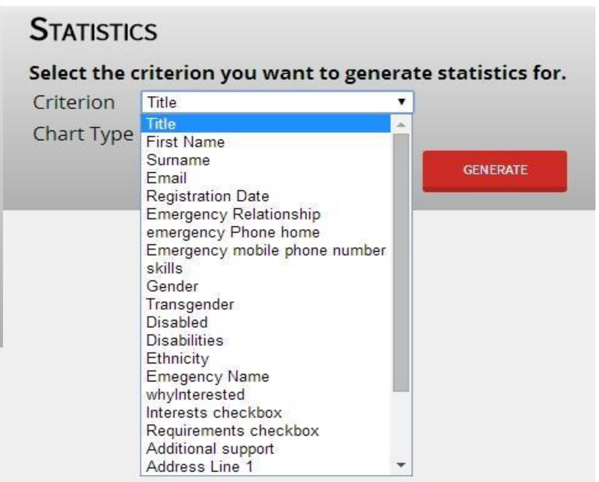
STATISTICS

Select the criterion you want to generate statistics for.

Criterion	Birthday
Start Date	1980-01-01
End Date	
Interval	
Chart Type	

February 1980

Su	Mo	Tu	We	Th	Fr	Sa
					1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	



STATISTICS

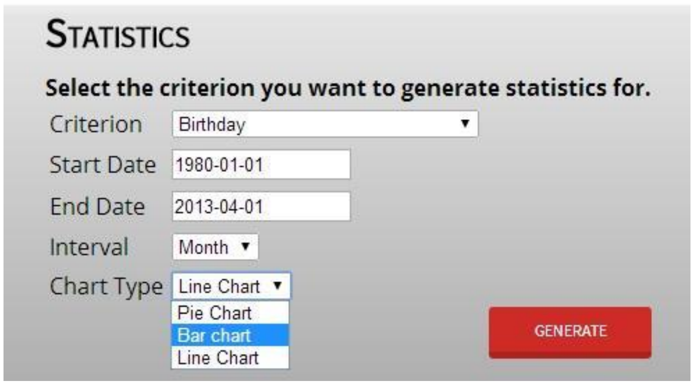
Select the criterion you want to generate statistics for.

Criterion	Title
Chart Type	

Title
First Name
Surname
Email
Registration Date
Emergency Relationship
emergency Phone home
Emergency mobile phone number
skills
Gender
Transgender
Disabled
Disabilities
Ethnicity
Emegency Name
whyInterested
Interests checkbox
Requirements checkbox
Additional support
Address Line 1

a text box appears and the user can write the keywords to search for.

There are 3 types of charts available. *Pie chart* and *Bar chart* are available for all fields. However, a line chart can only be displayed only if the selected field has the type date. Also, when the user selects a field that has type date, the line chart will be automatically selected for them. For selecting the date we are using the 'Datepicker' plugin.



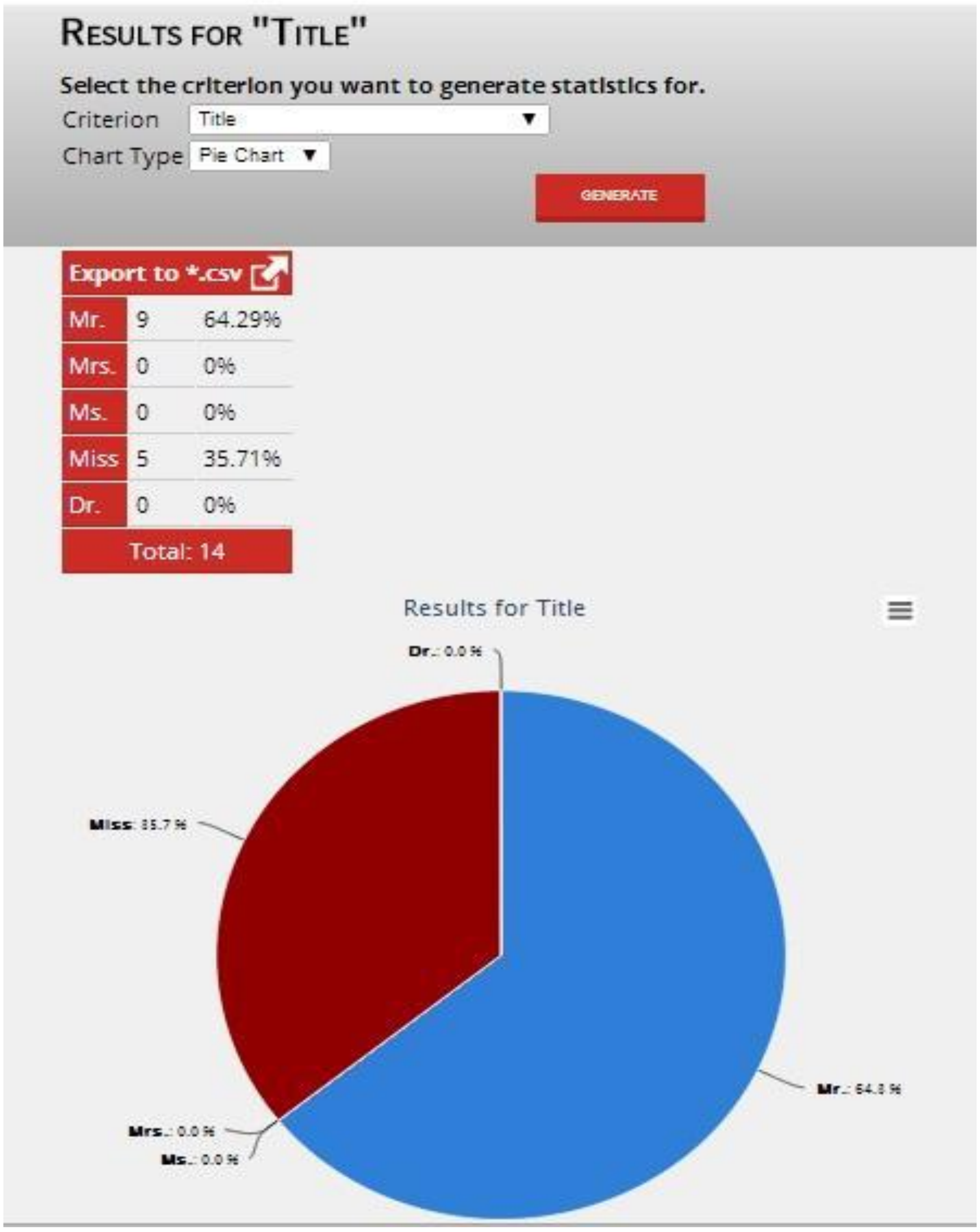
STATISTICS

Select the criterion you want to generate statistics for.

Criterion	Birthday
Start Date	1980-01-01
End Date	2013-04-01
Interval	Month
Chart Type	Line Chart

Pie Chart
Bar chart
Line Chart

After selecting the criterion and pressing the ‘Generate’ button the user will be redirected to the second page that displays the result of the selection.


RESULTS FOR "TITLE"

Select the criterion you want to generate statistics for.

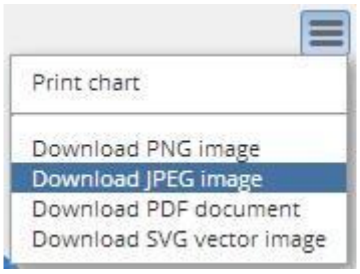
Criterion: Title

Chart Type: Pie Chart

GENERATE

Export to *.csv		
Mr.	9	64.29%
Mrs.	0	0%
Ms.	0	0%
Miss	5	35.71%
Dr.	0	0%
Total: 14		

The image above contains a pie chart of all possible titles - title is a field of type drop-down list.



The second page contains a table with the results and also a chart for better understanding the outcome.

The charts can be exported as a PNG image, JPEG image, PDF document or SVG vector image. Also the result can be exported as a CSV file by pressing the ‘Export to *.csv’ link located atop the result table which gives our clients the ability to generate various other visualisations using Excel. For exporting the data as a CSV file we are using a plugin written in jQuery.

For visualizing the result we have used ‘**Highcharts.js**’, a plugin written in pure JavaScript. Highcharts offers a variety of charts, but we have selected the ones that best display the type of data we are using. We have selected this plugin because it is easy to use and customize.

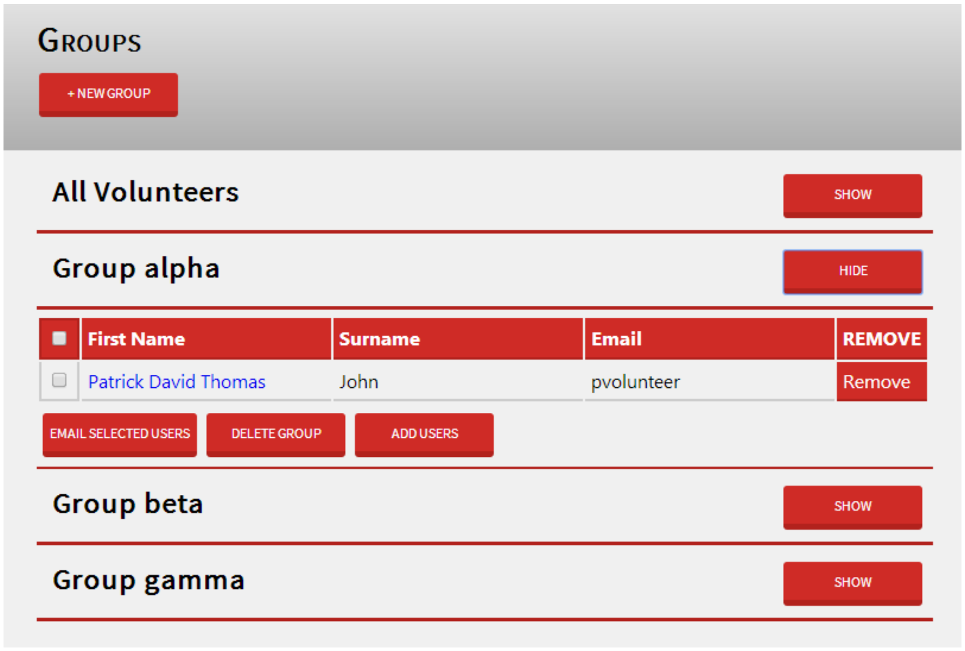
5.14 Groups

Collaborators: Nikolay (48 hours), Patrick (14 hours), Milan (4 hours), Kristian (5 hours)

URL: http://campan.tk/groups/

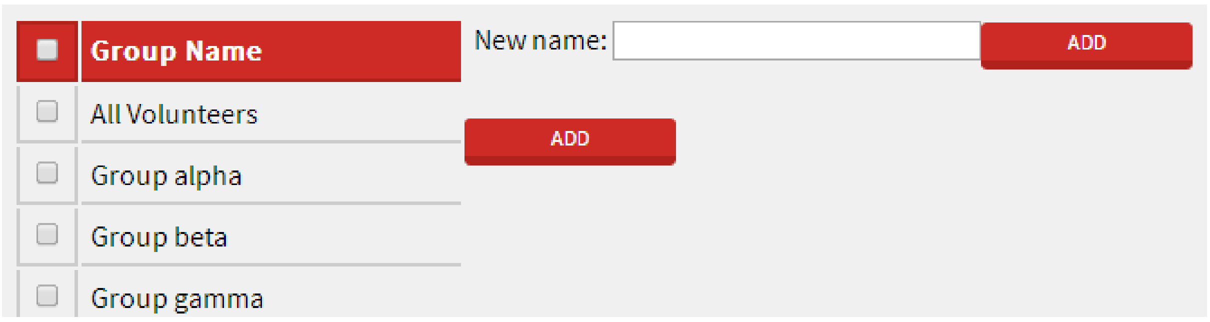
Description:
Since the number of users is fairly high (there are currently 1000 volunteers) and keeps growing, managing the volunteers could be intricate at times. In order to avoid that and to ensure scalability, the group feature was introduced. This way the admins can keep track of certain groups of people, e.g. those who have performed well at certain events.

Groups can be viewed and managed in the Groups page. This lets the user contact the members of a group (or a selected few of them) via email, remove people from groups and create new groups/delete existing ones



☐	First Name	Surname	Email	REMOVE
☐	Patrick David Thomas	John	pvolunteer	Remove

Adding users to groups, however, can be done from the Browse Database page since it contains the list of all volunteers and allows the admin to filter the list. Multiple volunteers can be added to multiple groups, including new groups which the user would like to create.



☐	Group Name
☐	All Volunteers
☐	Group alpha
☐	Group beta
☐	Group gamma

Neither of the pages allow two or more groups with the same name to exist. Empty names are not allowed either.

5.15 Events page

Collaborators: Ana (15 hours), Andrei (3 hours), Kristian (2 hours)

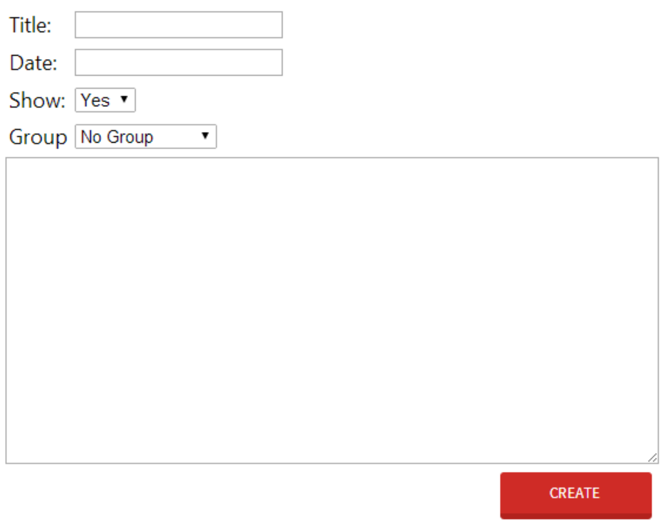
URL: http://campan.tk/event

Description:
The events section consists of two pages: one for displaying all the events, visible to all types of users with slightly different options available depending on whether the user has the required permissions to edit or create events, and one for creating new events or editing existing ones that is only available to users who have permission to access it.

From the perspective of a volunteer and admin without edit event privileges the display event page only shows the event's information. Accessing the "create new event" or "edit event" buttons is not only hidden but also forbidden. Even if the user finds the link to that page they are being redirected.

The display page shows 5 events per page. For each event any registered user can see the title, the author, the date when the event will take place and a description of the event. If the current user has edit/create event privilege they can see the group that is linked to this event, if there is one. Also, they are able to see the 'Edit', 'Delete' and 'Create new event' buttons and access the pages that these buttons link to.

For creating a new event, the fields are blank and they need to be filled in by the user before submitting the new event. They need to provide the title of the event, the date when is going to take place and a description of the event. Also, there is the possibility of linking an already existing group with the event. Linking a group to an event is not mandatory, the user can select the option of 'No group' which will leave the group field in the table empty. This option is displayed on the event view page as 'No group has been selected'. Not all events that are created need to be displayed. The user can decide if they want to show the event or not by setting 'Show' to either 'Yes' or 'No'. By default the option of showing the event will be selected.


Title:
Date:
Show: Yes
Group No Group

If the user wishes to edit an existing event, the 'Edit' button redirects them to the page that contains the menu shown above. The only difference is that the information for the event is be already filled. After making the changes, by pressing the 'Submit' button they can update the information in the database.

The Events page also gives admins the option to delete certain events. This can be done by pressing the 'Delete' button situated underneath each event's description.

5.16 Changelog

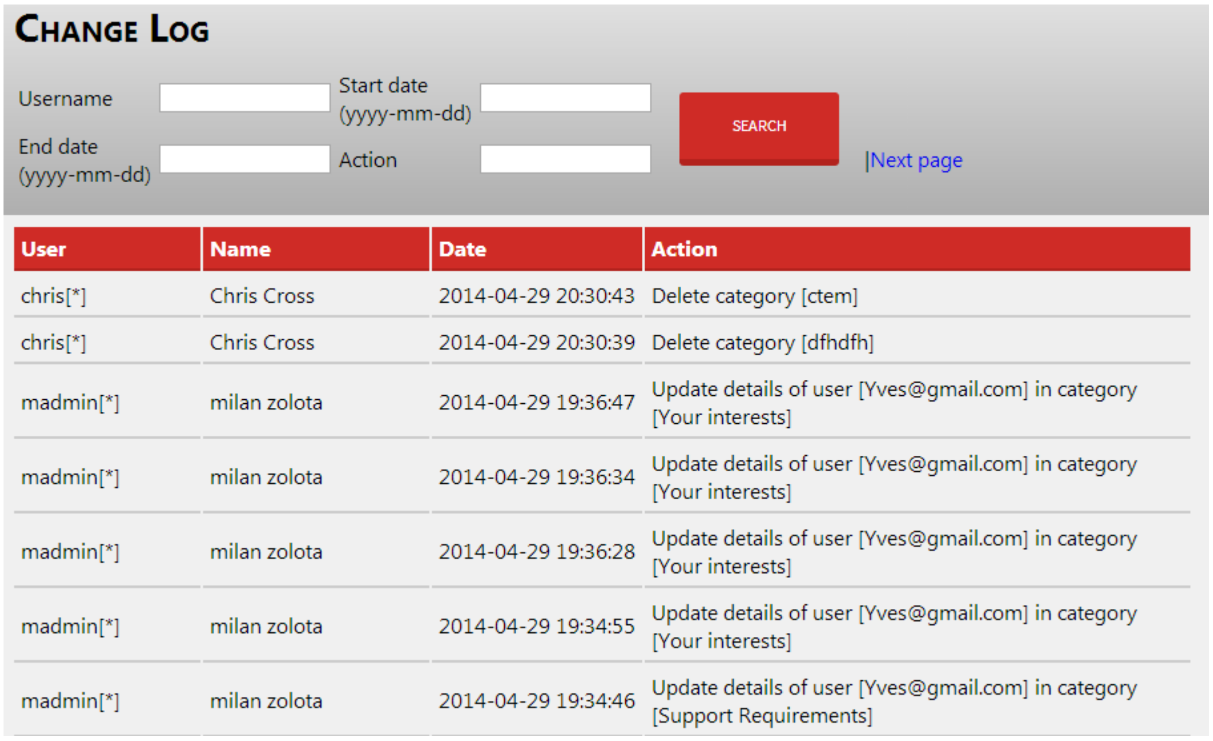
Collaborators: Andrei (18 hours), Milan (5 hours)

URL: http://campan.tk/log/

Description:
Since there are more admins and sub-admins on the website some mechanism for keeping track of the actions committed by users on the system. The main advantage of this feature is that if something goes wrong it will be easier to identify what caused the problems which will result in a more efficient way to solve it.

In order to successfully implement this feature we went through most of the code and added at the end of each function, calls to a special function that will store in the database the date, the user and what that user did. Any details saved in this manner cannot be deleted by any user and remain in the database forever.

For displaying the changelog a new page was created. On this page (that is only accessible by master admins) a user can see all activity on the website; 20 actions are displayed and a user can go through different pages to see older notes in the log.


CHANGE LOG

User	Name	Date	Action
chris[*]	Chris Cross	2014-04-29 20:30:43	Delete category [ctem]
chris[*]	Chris Cross	2014-04-29 20:30:39	Delete category [dfhdfh]
madmin[*]	milan zolota	2014-04-29 19:36:47	Update details of user [Yves@gmail.com] in category [Your interests]
madmin[*]	milan zolota	2014-04-29 19:36:34	Update details of user [Yves@gmail.com] in category [Your interests]
madmin[*]	milan zolota	2014-04-29 19:36:28	Update details of user [Yves@gmail.com] in category [Your interests]
madmin[*]	milan zolota	2014-04-29 19:34:55	Update details of user [Yves@gmail.com] in category [Your interests]
madmin[*]	milan zolota	2014-04-29 19:34:46	Update details of user [Yves@gmail.com] in category [Support Requirements]

Furthermore, we implemented a search option that can receive different search criteria (username, date interval, specific action) for an easier way of identifying the source of the problems that might occur. You can see this functionality in the above image.

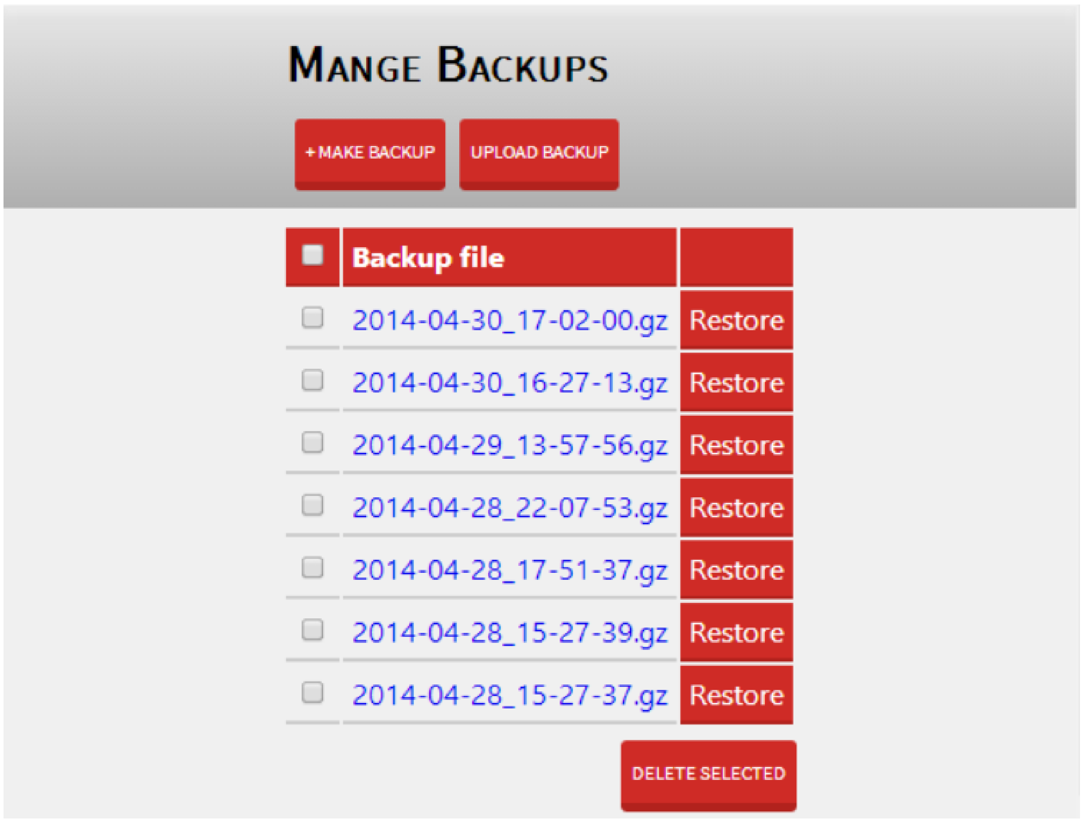
5.17 Backup management

Collaborators: Patrick (8 hours), Milan (3 hours)

URL: http://campan.tk/backup/

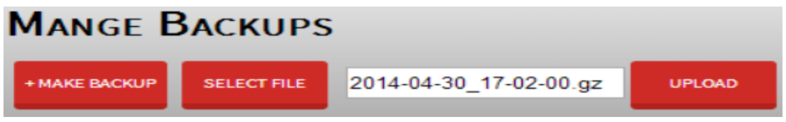
Description:
A simple, lightweight page for managing backups of the database (and only the database). This page can be used as part of the reinstallation process of the product if necessary and should be used regularly to store backups for the unlikely event of a database corruption or accidental removal of data. Backups are compressed and will be ~100KiB for their current database size, so even keeping hundreds of them is non-problematic.

Backups may be generated, downloaded, uploaded, restored from or deleted from the server entirely. When generating a backup, it is given a filename containing the current timestamp, and when the backups are listed, they are done so in ascending order of recency.



	Backup file	
☐	2014-04-30_17-02-00.gz	Restore
☐	2014-04-30_16-27-13.gz	Restore
☐	2014-04-29_13-57-56.gz	Restore
☐	2014-04-28_22-07-53.gz	Restore
☐	2014-04-28_17-51-37.gz	Restore
☐	2014-04-28_15-27-39.gz	Restore
☐	2014-04-28_15-27-37.gz	Restore

When uploading a file, a small file-input form is generated, and a dialog box opened that may only take Gzip files.



6 Testing

It was quite important to us to ensure that everything in the website works as intended. Hence, a lot of time was dedicated to testing it. Every time a bug was encountered, it was either fixed by the person who found it straight away, or reported to the others - orally if found during group work; online if found during solitary work.

6.1 Cross Browser compatibility

Although the website will be used mainly in Google Chrome, several tests have been made on different browsers to ensure that the user experience is as good as possible despite the platform. Due to the fact that everyone in the volunteering team uses Chrome, the testing was focused mainly on the pages which can be accessed by volunteers whose browser choice may vary.

Both the functionality and the visualisation of the website were tested. This was done using http://www.browserstack.com/ in which we were able to select practically any browser that exists and see how our website preforms in that browser.

Everything works well in the newest versions of popular browsers, except for some minor visualisation in Internet Explorer.

6.2 Security

6.2.1 XSS

The big number of user-input fields suggests that there could be potential vulnerabilities. Firstly, the website was tested for cross-site scripting by trying to execute various scripts (mostly alerts) by submitting them into the database.

6.2.2 SQL Injections

Similarly, various SQL queries were tried in order to alter the database.

6.3 General testing

Numerous different tests were done in order to ensure that the website functions correctly for both volunteers and admins with different permissions.

6.3.1 Black-box

Some of the testing was done from a user's point of view which means there was no assumed access to the database. In other words, everything was done on the website. For example, when deleting a group, the Groups page was simply refreshed to check whether the recently deleted group was shown.

6.3.2 White-box

The back-end of the project (i.e. the database) was trialled as well. Although the black-box testing showed whether or not some functions work, there are certain events which are not visible to the common user. For example, the database holds a table which represents the relationship between users and groups (e.g. user abc@mail.com is in Group One), so we checked whether or not deleting a group would also remove all traces of any members belonging to it.

7 User guide

7.1 Installation guide

Installation of our website is relatively simple. It requires a machine with php5.5 or newer, MySQL and apache2, all of which are a standard on linux machines nowadays. Then, the following steps are required:

1. Copy the website folder to the machine.
2. Import the MySQL database.
3. Create a MySQL user with permissions for this database.
4. Change the `$config['base_url']` in the `/application/config/config.php` file to be equal to the new domain. For example, `$config['base_url'] = 'www.newdomain.com';`
5. Change the `$db['default']['username']`, `$db['default']['password']`, `$db['defaulti']['username']` and `$db['defaulti']['password']` in the /`application/config/database.php` file to match the username and password of the newly created MySQL user.
6. Edit the `/etc/apache2/sites-enabled/default.conf` file (or similarly named file in the folder) and include the following code in it:

```
<VirtualHost *:80>
ServerName www.newdomain.com
ServerAlias newdomain.com
DocumentRoot /var/www/Campan/
RewriteEngine On
</VirtualHost>
```

The 'www.newdomain.com' and 'newdomain.com' should be changed to match the new domain name. The `DocumentRoot /var/www/Campan/` path should be changed to match the path where the website was copied to.

7. If not already done, set the new domain used for this website to point to the machine's public IP address.
8. Restart apache2, and everything should be up and running.

7.2 Browse database

Upon arriving at the Mailing List Browser page, one is bombarded with tasks they could perform. The most common functionality that will be used is the simple query → email volunteers sequence.

First, one must choose a criteria with which to filter volunteers by, the drop-down menu eases this process by listing all the fields with which one may view. Then the value is to be entered; depending on the chosen criteria-field, this may be done in one of several ways.

Date: One may type in the date manually, following the hinted syntax of `dd/mm/yyyy`, or if their browser supports it, they may use the graphical calendar display to select the date, at which point, the text representation of the date will populate the value field

Text: One types in the text they wish to filter with, depending on the field type (‘big text’ or ‘small text’), this may query for an exact match or one where the inputted word is searched within the database entries for the field.

Number: This can, and most likely will, be entered as text; there is however, the option to use arrows to increment and decrement the currently entered number; furthermore, there is numerical validation applied.

Email: Like text, but validated to ensure it follows the standard email format.

Checkboxes: Here, one may simply check the predefined options that they would want volunteers to have included in the field that is being used for filtering; resulting volunteers will have at least one of the checked values.

Drop-down menu: This is similar to choosing the field with which to filter by; one chooses an option, and only those volunteers who have that particular option will be in the result.

One may add a criterion at any point; if one adds too many, there is nothing to worry about, any values left blank will be ignored in the filtering process. One may specify the order by which to sort the resulting volunteers, and by which field; one may also sort the volunteers by any displayed field after generating the results by clicking on the column header in question. If no volunteers match the necessary criteria, a pop-up is displayed quoting “No results found”, and the Browse Database page returned to.

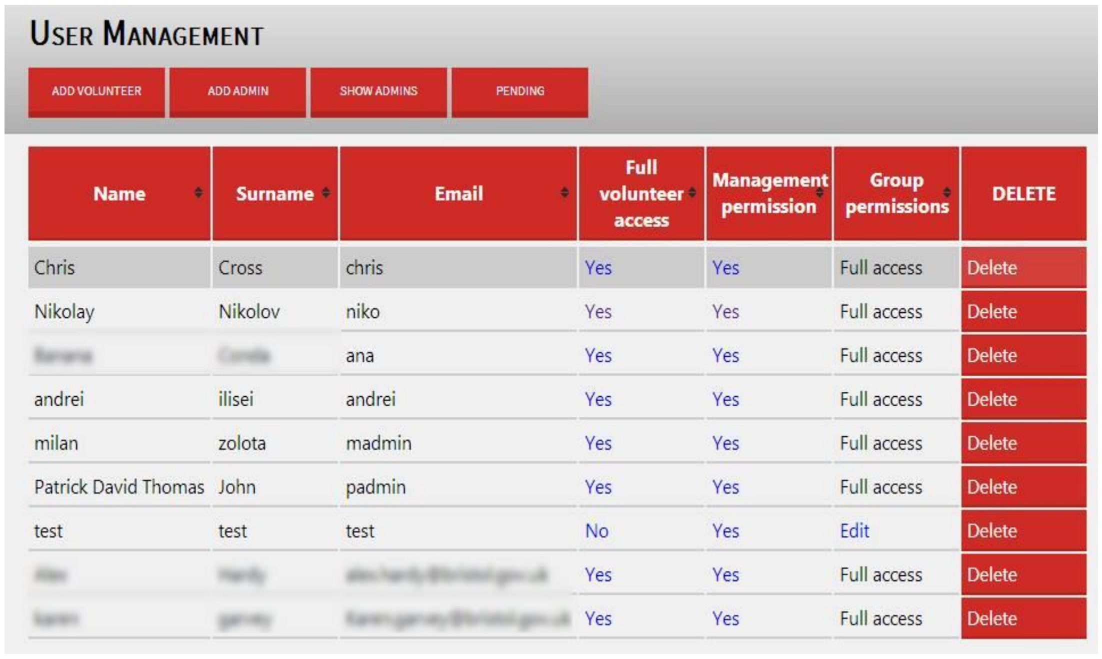
By checking specific volunteers, further operations may be performed. Having checked some volunteers, one may click ‘show all’, where volunteers can be explored to further depth. One may click ‘add to group’, which will cause a list of groups to appear at the bottom of the page; from here, one may choose to make a new group to put these volunteers in, or one can choose from the list of existing groups via the now-familiar checkbox system.

If one clicks ‘export database’, a CSV file of the entire volunteers table will be generated and become available for download by the admin.

7.3 User management

The user management page provides several functionalities.

1. **Adding new volunteers and admins**
   After clicking the 'Add volunteer' or 'Add admin' button, an admin user is requested to fill in relevant details for a new volunteer or an admin. For a volunteer, an email address if enough, whereas an email, first name, surname and permissions are required for an admin. After submitting this information, the admin user's default email client is opened with a new email template with the new volunteer's or admin's email address filled in. The body of the template will also include a unique URL that the new user has to use to register.
2. **Managing current admins**



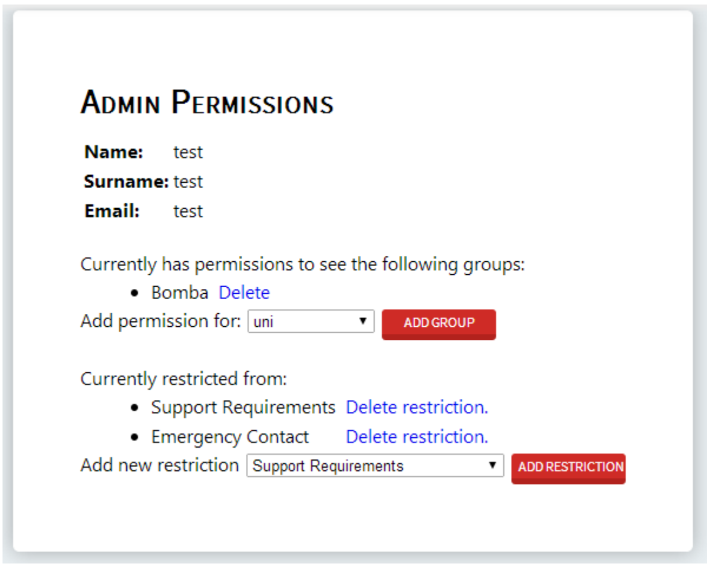
USER MANAGEMENT

ADD VOLUNTEER | ADD ADMIN | SHOW ADMINS | PENDING

Name	Surname	Email	Full volunteer access	Management permission	Group permissions	DELETE
Chris	Cross	chris	Yes	Yes	Full access	Delete
Nikolay	Nikolov	niko	Yes	Yes	Full access	Delete
[redacted]	[redacted]	ana	Yes	Yes	Full access	Delete
andrei	ilisei	andrei	Yes	Yes	Full access	Delete
milan	zolota	madmin	Yes	Yes	Full access	Delete
Patrick David Thomas	John	padmin	Yes	Yes	Full access	Delete
test	test	test	No	Yes	Edit	Delete
[redacted]	[redacted]	[redacted]	Yes	Yes	Full access	Delete
[redacted]	[redacted]	[redacted]	Yes	Yes	Full access	Delete

3. This part of the page allows an admin to manage all current admin. To understand this properly, the permission system needs to be introduced. There are two different permissions:
   A) **Full volunteer access permission** - An admin with this permission is given access to all volunteers' information, i.e. all volunteers are shown in the browse database page and all information about those volunteers is accessible. An admin without this permission cannot access any volunteers by default. This can be changed by clicking the 'Edit' link in the 'Group permission' column. After clicking the link, the page on the right is shown.

   On this page, it is possible to select the groups of volunteers that the admin can see. It is also possible to select the categories



ADMIN PERMISSIONS

Name: test
Surname: test
Email: test

Currently has permissions to see the following groups:
- Bomba Delete

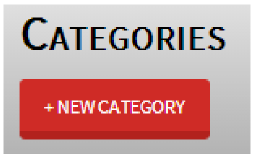
Add permission for: uni ▾ ADD GROUP

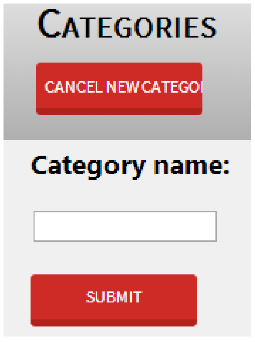
Currently restricted from:
- Support Requirements Delete restriction.
- Emergency Contact Delete restriction.

Add new restriction Support Requirements ▾ ADD RESTRICTION of information that the admin should not be able to see.

B) **Management permission** - An admin with this permission is given unlimited access to the whole website. This includes accessing this 'Manage users' page, the 'Manage registration form' page, Groups page, Stats page, 'Manage backups' page and Changelog page. The permission also allows an admin to manage events and edit the contact details in the contact us page, as well as the details of any volunteer.

7.4 Manage Registration Form

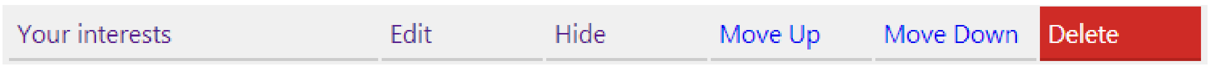




In the beginning the registration form is almost empty; it has only one non-editable category that requests for basic information.

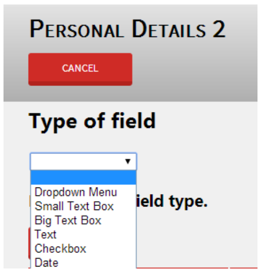
In order to add a new category, press the **New category** button at the top of the page. This will display a small textbox where the admin needs to insert the category name and then press **Submit** in order to save the category (you can see all these buttons in the side images). Pressing Submit will redirect the user to the page for editing that category. If there are already some categories created, the admin can do the following options:

- Hide a category: this will hide the category from the registration form but all details saved in the fields from that category will still be visible to admins. This action will also

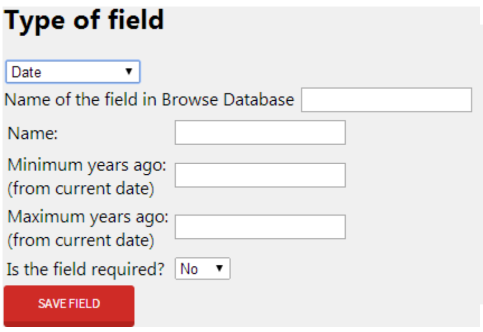


  make that category hidden for volunteers in the user profile page.
- Delete a category: this action deletes the category, all the fields in the category and all the information about volunteers stored in those fields. Pressing the button for this option will generate an alert that asks for deletion confirmation in order to prevent accidental actions.
- Move a category up or down in the registration form.

Clicking on the category name or on the 'Edit' link will redirect the user to the page for editing that category.

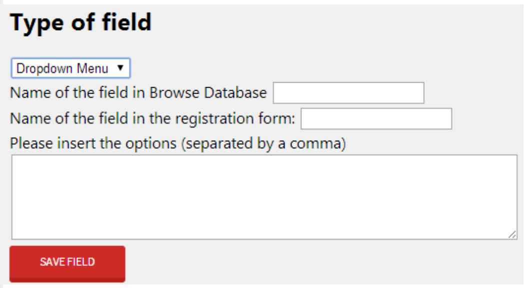


The process of adding a new field in a category is similar to the one for adding a new category. After pressing the **New field** button, the admin will be asked to select the type of the input field and then to provide appropriate options for that field. The set of options that an admin can select is customized according to the field type as it can be seen on the images in the next page.



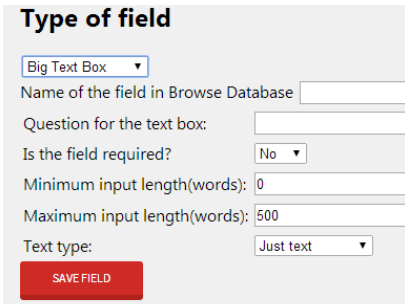
Type of field

	Date
Name of the field in Browse Database	
Name:	
Minimum years ago: (from current date)	
Maximum years ago: (from current date)	
Is the field required?	No



Type of field

	Dropdown Menu
Name of the field in Browse Database	
Name of the field in the registration form:	
Please insert the options (separated by a comma)	

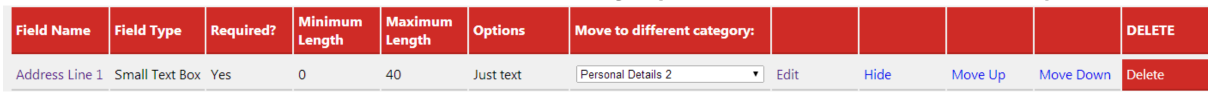


Type of field

	Big Text Box
Name of the field in Browse Database	
Question for the text box:	
Is the field required?	No
Minimum input length(words):	0
Maximum input length(words):	500
Text type:	Just text

Clicking on an existing field or on the **Edit** button corresponding to that field will redirect the user to a page that displays similar information as the **New field** menu with the current option already loaded in the input form.

Similar to categories, an admin has the following options for a field: **Move Up/ Move Down/ Hide/ Show/ Delete/ Move to a different category** as it can be seen in the image below.

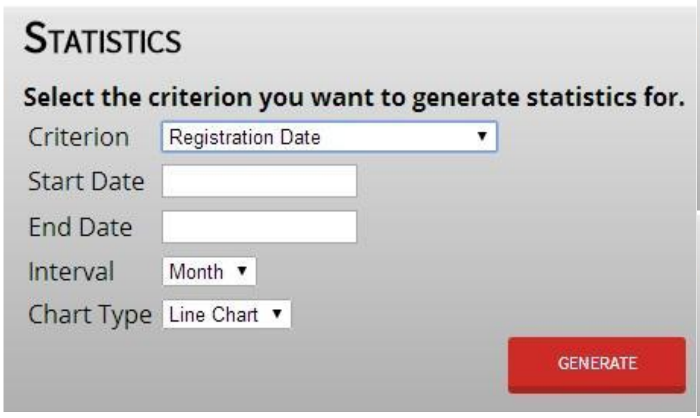


Field Name	Field Type	Required?	Minimum Length	Maximum Length	Options	Move to different category:					DELETE
Address Line 1	Small Text Box	Yes	0	40	Just text	Personal Details 2	Edit	Hide	Move Up	Move Down	Delete

7.5 Statistics

In order to generate statistics you first need to be on the Statistics page. This can be accessed by pressing the ‘Stats’ button from the top menu of the website.

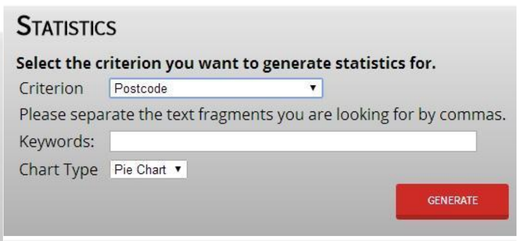
Selecting the criterion you want to generate statistics for can be done by clicking on the drop down list next to ‘Criterion’. Depending on the type of field that has been selected new menu options can appear. If the field selected is of type data, there will be the options of selecting the start date, end date and the interval either monthly or yearly. If the field selected is of type checkbox or drop down list nothing new will be generated. If the field is of type text, a new textbox will appear where the user can insert the values they want to search for.



STATISTICS

Select the criterion you want to generate statistics for.

Criterion	Registration Date
Start Date	
End Date	
Interval	Month
Chart Type	Line Chart



STATISTICS

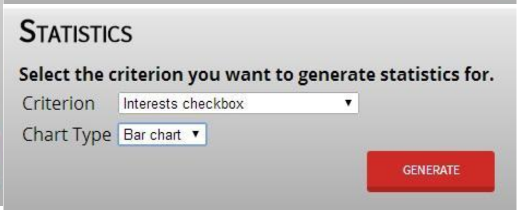
Select the criterion you want to generate statistics for.

Criterion: Postcode

Please separate the text fragments you are looking for by commas.

Keywords:

Chart Type: Pie Chart



STATISTICS

Select the criterion you want to generate statistics for.

Criterion: Interests checkbox

Chart Type: Bar chart

After filling in all these details, the user can select the type of chart they want to display. There are 3 options available: bar chart, pie chart and line chart. Line chart is only available for fields of type date.

8 Appendix

8.1 Appendix A - Code examples

8.1.1 PHP code for generating some of the fields in the registration form:

```
if ($field->fieldType=='bigText')
{
	if ($table==true)
	{
		echo '</table>';
		$table=false;
	}
	echo '<li><span>'.$field->name;
	if ($field->required==true)
		echo"<font color='red'>*</font>";
	echo '</span></li>';
	echo '<textarea cols="100%" rows="12" name="'.$field->ID.'"></textarea>';
}

else if ($field->fieldType=='DropDown')
{
	echo '<br><span>'.$field->name.'</span><br>';
	$options = explode(",",$field->options);
	echo '<select name='.$field->ID.'>';
	foreach ($options as $option)
		echo '<option>'.$option.'</option>';
	echo '</select><br><br>';
}

else if ($field->fieldType=='date')
{
	if ($table==false)
	{
		echo '<table>';
		$table=true;
	}
	echo '<tr>';
	echo '<td><span>'.$field->name;
	if ($field->required==true)
		echo"<font color='red'>*</font>";
	echo '<br>(yyyy-mm-dd)</span></td>';
	echo '<td><input type="text" id="datepicker" size="16" name="'.$field->ID.'"></td>';
	echo '</tr>';
}
```

The PHP code above checks the type of the current field and according to it, HTML is generated to display the requested input field. The small input fields (e.g. date, drop down menu, small text box) are put in a table in order to obtain a nice alignment in the registration form.

8.1.2 jQuery code for hiding/showing a menu on the Manage registration form page

```
$("#newFieldButton").click(function() {
	if ($("#newField").css('display')=='none')
	{
		$("#newField").show(500);
		$("#newFieldButton").attr('value', 'Cancel');
		$("#categories").animate({'marginTop':'0px'}, 10);
		document.getElementById("bode").style.position="fixed";
```

```
        }
        else
        {
                $("#newField").hide(250);
                $("#newFieldButton").attr('value', '+ New field');
                $("#categories").animate({'marginTop':'116px'}, 1100);
                document.getElementById("bode").style.position="static";
        }
})}
```

The jQuery code above checks for the current state of the menu (if it is hidden or shown) and hides or shows it when a button is clicked. It also changes some options in the style of the page for

8.1.3 Javascript code for changing Browse Database value entry

```
function SetValueInput(field, id, oldType)
{
        var enumIndex = enumNames.indexOf(field);
        var setIndex = setNames.indexOf(field);
        id = id.substring(id.lastIndexOf('_')+1);

        // Input type is currently a simple text box
        if (oldType == "input")
        {
        // Find the value input box
        var input = document.getElementById("value_"+id);
        // If selected field is an enum
        if (enumIndex != -1)
        {
                // Change the field selection's onchange w.r.t. 'select'

        document.getElementById("field_"+id).attributes.getNamedItem("onchange").textContent
= "SetValueInput(this.value,this.id,'select');";

                // Create the new select-option dropdown box for the value
                var select = document.createElement("select");
                select.name = "value_"+id;
                select.id = "value_"+id;
                select.innerHTML = "";
                var enumOffset = enumOffsets[enumIndex];
                var enumOffset_end = enumIndex+1 == enumNamesSize ? enumValuesSize :
enumOffsets[enumIndex+1];
                for (var ii = enumOffset; ii < enumOffset_end; ++ii)
                select.innerHTML += "<option>" + enumValues[ii] + "</option>";

                // Replace the current input box with new dropdown box
                input.parentNode.insertBefore(select, input);
                input.parentNode.removeChild(input);
                return;
        }

        // If selected field is a set
        if (setIndex != -1)
        {
                // Change the field selection's onchange w.r.t. 'checkbox'

        document.getElementById("field_"+id).attributes.getNamedItem("onchange").textContent
= "SetValueInput(this.value,this.id,'checkbox');";

                // Create the new checkbox table for the value
                var table = document.createElement("div");
                table.className = "chkbxs";
                table.id = "value_"+id;
                var innerHTML = "<table>";
                var setOffset = setOffsets[setIndex];
                var setOffset_end = setIndex+1 == setNamesSize ? setValuesSize :
setOffsets[setIndex+1];
                for (var ii = setOffset; ii < setOffset_end; ++ii)
                innerHTML += "<tr><td><input type=checkbox name=value_"+id+"_" +
setValues[ii] + ">" + setPrettyValues[ii] + "</td></tr>";
                innerHTML += "</table>";
                table.innerHTML = innerHTML;
                // Replace the current input box with new checkbox table
                insertAfter(table, input);
                input.parentNode.removeChild(input);
                return;
        }

        // Selected field is neither an enum nor a set
        input.type = fieldTypes[fieldNames.indexOf(field)];
        return;
        }
```

This is for a specific case of changing the value field, for when the previous type was a plain text box.

8.1.4 PHP code for restoring from a backup

```
function restoreBackup($filename, $restoring = 0)
{
        if ($restoring == 0)
        {
        $this->load->view('view_menu');
        $this->load->view('view_backup_restore');
        }
        elseif ($restoring == 1)
        {
        $file = gzopen(FCPATH."backups/$filename", 'rb');
        $SQL = '';
        while (!gzeof($file))
                $SQL .= gzread($file, 4096);

        $SQLi = $this->load->database('defaulti', TRUE);
        $SQLi->call_function('multi_query', $SQLi->conn_id, $SQL);
        while ($SQLi->call_function('next_result', $SQLi->conn_id));

        $this->mysql->changelog('Backup restored');
        redirect('backup/index');
        }
}
```

In the first case of restoring, we load a progress view, which will be displayed whilst the second case of restoring is being performed. In the second case of restoring, the parameter file is open and decompressed from Gzip in 4KiB chunks; an alternative database connection is open that allows for multiple SQL queries to be performs from one string of text; when the while-loop finishes, the database will have been restored, the change log updated accordingly, and the user is returned to the management page.

8.2 Appendix B - Initial Schedule

No.	TASK	Prerequisites	Start	End	Release
1	**Server**				
1.1	Set up the server	-	21 Nov	28 Nov	Alpha
1.2	Create changelog	3	14 Feb	22 Feb	Gamma
1.3	Implement secure HTTP protocol	3	14 Feb	1 Mar	Gamma
2	**Database and Database Management**				
2.1	Define database tables	Meeting with the client	21 Nov	30 Nov	Alpha
2.2	Create the database	1.1, 2.1	1 Dec	7 Dec	Alpha
2.3	Implement security (Hashing and salting passwords, parameterising queries)	3	14 Feb	25 Feb	Gamma
3	**Website**				
3.1.1	Define the structure and design of the website	Meeting with the client	21 Nov	1 Dec	Alpha
3.1.2	Define and create php functions for the alpha release	1.1, 2.1	1 Dec	5 Dec	Alpha
3.1.3	Create the registration page	3.1.2	5 Dec	16 Dec	Alpha
3.1.4	Create the login page	3.1.2	5 Dec	16 Dec	Alpha
3.1.5	Create the browse database page without support for emailing volunteers	3.1.2	5 Dec	16 Dec	Alpha
3.2.1	Define and create PHP functions for the beta release	1.1, 2.1	17 Dec	24 Dec	Beta
3.2.2	Create the user management page without support for emailing users	3.2.1	25 Dec	27 Jan	Beta
3.2.3	Create the requests management page with support only for registrations	3.2.1	25 Dec	27 Jan	Beta
3.3.1	Create the bulk email page	3.1.3, 3.1.5	25 Dec	27 Jan	Beta
3.3.2	Add support for emailing volunteers and users to the browse database and manage users pages	3.3.1	28 Jan	5 Feb	Gamma
3.3.3	Create the edit profile page	3.2.1	28 Jan	5 Feb	Gamma

3.3.4	Add support for event requests to the requests management page	3.2.3, 3.3.3	6 Feb	8 Feb	Gamma
3.3.5	Create the backup page	3.2.1	9 Feb	13 Feb	Gamma
4	**Installation**				
4.1	Basic installation	1, 2, 3	1 Mar	7 Mar	Gamma
4.2	Specify installation directory	1, 2, 3	8 Mar	12 Mar	Gamma
4.3	Ability to import previous database	1, 2, 3	12 Mar	19 Mar	Gamma
5	**Testing**	1, 2, 3, 4	20 Mar	1 May	Final

8.3 Appendix C – Timesheets

CAMPAN Timesheet for Alpha release						
Task	Andrei ILISEI	Ana DUMITRAS	Milan ZOLOTA	Patrick JOHNSTON	Nikolay NIKOLOV	Kristian KRASTEV
Learning New Skills	12	15	13	15	15	15
Server preparation	-	-	4	-	-	-
Database creation	3	-	1	-	-	-
Registration Page	15	-	3	1	15	-
Browse Database Page	-	14	-	14	-	14
Login Page	-	-	15	-	-	-
User Profile	5	-	1	-	-	-
TOTAL	**35**	**29**	**37**	**30**	**30**	**29**

CAMPAN Timesheet for Beta release						
Task	Andrei ILISEI	Ana DUMITRAS	Milan ZOLOTA	Patrick JOHNSTON	Nikolay NIKOLOV	Kristian KRASTEV
Manage Registration Page	32	-	-	6	35	-
Generate Registration Page	14	-	-	2	13	-
User Profile	5	-	-	1	2	-
User Management Page	-	-	30	-	-	-
Email	-	12	8	-	-	18
Menu	-	-	4	-	-	-
Browse Database Page	-	8	2	-	-	17
Design	-	11	8	-	-	15
TOTAL	**54**	**31**	**52**	**52**	**50**	**51**

CAMPAN Timesheet for Gamma release						
Task	Andrei ILISEI	Ana DUMITRAS	Milan ZOLOTA	Patrick JOHNSTON	Nikolay NIKOLOV	Kristian KRASTEV
Testing	16	17	15	18	18	18
Design	-	6	10	-	-	35
Statistics	7	20	-	-	-	-
Groups	-	-	4	14	28	5
Permissions	-	-	15	-	-	-
Security	2	-	2	-	-	-
User Management Page	-	-	6	3	-	-
Events Page	2	15	-	-	-	-
Browse Database Page	-	-	-	17	5	5
Changelog Page	18	-	5	-	-	-
Edit Profile Page	22	-	6	-	-	-
TOTAL	**67**	**58**	**65**	**62**	**61**	**63**

CAMPAN Timesheet for Final release						
Task	Andrei ILISEI	Ana DUMITRAS	Milan ZOLOTA	Patrick JOHNSTON	Nikolay NIKOLOV	Kristian KRASTEV
Report	20	20	20	20	20	20
Testing	17	30	16	18	25	10
Bug Fixes	12	15	13	14	10	9
Minor changes in functionality	8	12	11	13	9	23
TOTAL	57	77	60	65	64	62

CAMPAN Final Timesheet						
	Andrei ILISEI	Ana DUMITRAS	Milan ZOLOTA	Patrick JOHNSTON	Nikolay NIKOLOV	Kristian KRASTEV
Alpha	35	29	37	30	30	29
Beta	54	31	52	52	50	51
Gamma	67	58	65	62	61	63
Final	57	77	60	65	64	62
TOTAL	213	195	214	209	205	205

8.4 Appendix D – Contributions

Ana Dumitraș	1	
Andrei Ilisei	1	
Kristian Krastev	1	
Milan Zolota	1	
Nikolay Nikolov	1	
Patrick Johnston	1	

Individual Report
- Ana Dumitraș -

1. Introduction

My main role in the “Volunteer Database and Mailing List” project was to develop the Statistics, Events and Email pages. I also helped with the development of the overall design of the website and the browse database page.

The first step in developing our website was learning the technologies that we were going to use. I had some previous knowledge of HTML and CSS but no experience in using those languages. I used www.codeacademy.com to improve my HTML and CSS skills and learn php, JavaScript and jQuery. Also, for each page I developed or helped develop I spent some time learning how to use the necessary technologies.

2. Individual contribution

2.1. Statistics

One of my major contributions was developing the Statistics Page. I have built up the page together with Andrei. The page displays the information in two ways: firstly as a table and secondly as a chart. Both types of visualizing the data have the option of being exported in various formats. Exporting the results was one of the options the volunteering team was very keen on having on the website.

There were a number of jQuery plugins we have tested in order to display our results, but Highcharts proved to be the easiest to use and customize. The plugin offers a variety of charts to display data. We choose only 3 types of chart because we wanted to leave the process of displaying data as easy and possible. Also, most of the charts they offer do not fill our needs.

This is the code use for displaying a pie chart. The initial code, offered by Highcharts has been adapted to suit our needs. Some of the options were not necessary and even making the charts harder to read.

```
<?php
        if($chartType == "pie")
        {
?>
<script>
$(function () {
    $('#container').highcharts({
        chart: {
            backgroundColor:'transparent',
            plotBorderWidth: null,
            plotShadow: false
        },
        title: {
            text: 'Results for <?php echo $name;?>'
        },
        tooltip: {
           pointFormat: '{series.name}: <b>{point.percentage:.1f}%</b>'
        },
        plotOptions: {
            pie: {
                allowPointSelect: true,
                cursor: 'pointer',
                dataLabels: {
                    enabled: true,
                    color: '#000000',
                    connectorColor: '#000000',
                    format: '<b>{point.name}</b>: {point.percentage:.1f} %'
                }
            }
        },
        series: [{
            type: '<?php echo $chartType;?>',
            name: 'pie',
            data: [
                <?php
                             $count = 0;
                foreach($result as $key => $value)
                {
                    if($count>0)
                                    {
                                           echo ",";
                                    }
                                    echo "['".addslashes($key)."', ".$value."]";
                                    $count++;
                }
                             ?>
            ]
        }]
    });
});
</script>
<?php
       }
?>
```

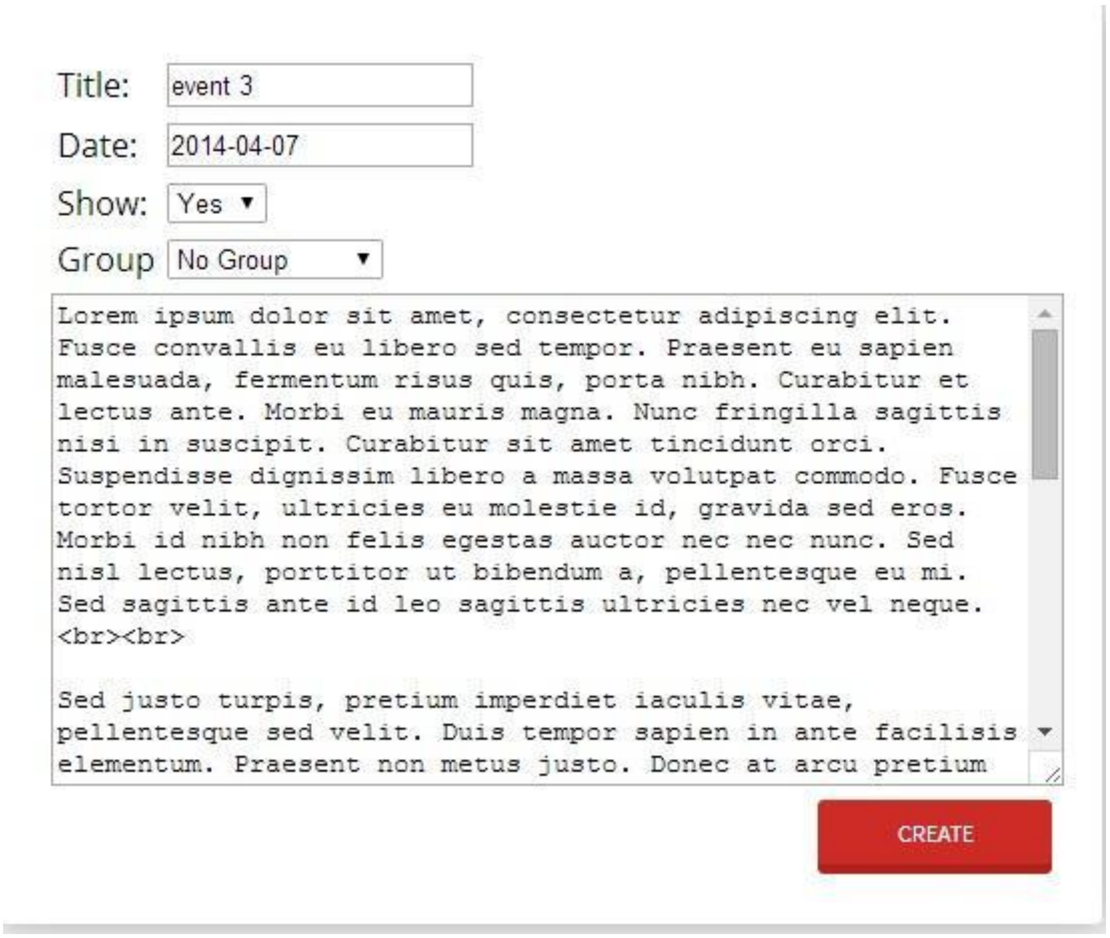
Using plugins that were developed by other people made me realize how important it is to comment your code and make it as readable as possible.

Another task I was assigned was to choose a plugin for dealing with data of type date. I used a jQuery plugin called Datepicker. After successfully implementing it on the Statistics page I was asked to change it all over the website. The use of this plugin minimizes the error of an user wrongly inserting the date.

2.2. Events

The events page was one of the last pages to be developed. I have worked on it together with Andrei. The page was later linked to the Groups page as requested by our clients.

The page has two versions: one for volunteers and one for admins. Both of them can view the information about events. The admins also have the option of editing events, creating events or editing events.



Title:	event 3
Date:	2014-04-07
Show:	Yes
Group	No Group

```
Lorem ipsum dolor sit amet, consectetur adipiscing elit.
Fusce convallis eu libero sed tempor. Praesent eu sapien
malesuada, fermentum risus quis, porta nibh. Curabitur et
lectus ante. Morbi eu mauris magna. Nunc fringilla sagittis
nisi in suscipit. Curabitur sit amet tincidunt orci.
Suspendisse dignissim libero a massa volutpat commodo. Fusce
tortor velit, ultricies eu molestie id, gravida sed eros.
Morbi id nibh non felis egestas auctor nec nec nunc. Sed
nisl lectus, porttitor ut bibendum a, pellentesque eu mi.
Sed sagittis ante id leo sagittis ultricies nec vel neque.
<br><br>

Sed justo turpis, pretium imperdiet iaculis vitae,
pellentesque sed velit. Duis tempor sapien in ante facilisis
elementum. Praesent non metus justo. Donec at arcu pretium
```

We are using the same page to create new events and edit existing one. I had to make sure we check if the event id is already in the database and retrieve all the information about it so the user can view it and edit it if necessary.

I also had to make sure volunteers and users without edit permission can not access the edit page or view certain information about the events: for example, only admins with full privileges can see the which group is linked to the event.

2.3. Email

My contribution to the 'Send Email Page' involved creating the overall structure of the page. I came up with the structure of the page and what it should do. Together with Kristian, we have implemented the functionality of the page.

The main purpose of the page is to let the admins easily email groups of volunteers. The page displays template emails, stored in our database. It lets the user edit the emails and then opens the default email software for them, with all required fields already filled in.

2.4. Minor contributions

In addition to the pages above I have helped the other members of our group with their development process. Most of it involved testing their already implemented functions, linking pages that I developed with their pages or helping with tracking bugs.

3. Discussion

3.1. Group work

One of the facts I liked the most about the way we worked was that although we were developing different files we tried to work in the same room. This helped a lot when we were unsure how to develop a certain feature and we needed everybodys input. Although we prefer working together, this was not always possible during the holidays when we had to go home. However we made sure everybody was aware of the changes each of us was making so that we will not get in the way of one another.

Usually, there were between two or three people working on a task at a given time, having different levels of knowledge in the technologies we were using. This way we learned a lot from each other. The teams were created in such a way that over the development of the project each of us worked with everybody else.

I think overall we worked well as a group. The project was finished on time. We did not have any major conflicts. Everybody contributed to the development of our product. We all met regularly to both discuss our progress and work on the project. Also we tried to meet with our client whenever it was possible and if we had something that needed to be solved urgently we wrote on Basecamp or emailed them.

3.2. Work allocation

We tried allocating work as fairly as possible. However this was not always achievable because most of us did not have enough experience in web development and database management.

Division of tasks was done at the beginning when we were developing the System Requirements Specifications and the Schedule for the project. However we did not allocate a specific person to do a certain task until we actually started working on the task. Usually, the person who had the most knowledge about that task or had a better idea of how to implement it would be assigned the task.

The more experienced programmers did a slightly greater amount of work than everybody else, but this was mostly related to the fact the they spent less time researching and learning how to use the technologies.

Individual Report
- Andrei Ilisei-

1 Introduction

My main role in this project was to develop the functionality that is available to volunteers or that directly interacts with volunteers' experience. at the same time, I have constantly helped my teammates with any problem that they encountered, from solving minor bugs to implementing complex functionality.

Since I am not particularly good at colours and design, I tried to avoid getting involved in any tasks that had to deal with this part; I have done just the basic design of the pages that I developed, leaving Kristian (who was in charge of design) to make them look good and consistent with the rest of the website.

When I first started working on this project, I had absolutely no idea what kind of tools we needed to develop the website. I had basic knowledge of HTML and CSS, but I have never done more than creating 2 or 3 basic web-pages. The process of learning PHP and how to use the CodeIgniter had a really steep learning curve, as I was discovering more and more features that became necessary. At the same time, Patrick helped us quite a lot with the basic functions of this framework (he used it for one of his school projects).

I found the courses on www.codeacademy.com very useful. Using them, I managed to learn most of the features of PHP, CSS and HTML in just a couple of days. I would say that anyone who needs to learn on their own a new programming language should use the courses available on that website.

2 Individual Contribution

2.1 Registration Page and its management

For this section of the website I worked together with Nikolay and we constantly received help from Milan and Patrick. Developing the registration page was a long a tedious process, but at the same time very rewarding. This was the first page that I developed (together with Nikolay) and we knew nothing about what we were doing, but slowly we managed to finish it and make it fully functional.

Unfortunately, after the Alpha release, our clients told us that they want to be able to edit the registration form. This meant that we had to start all over again implementing it. At the same time, we had to make a system that allowed the admins to edit any input field on it. We ended up developing a similar system to Survey Monkey for creating surveys, which allows the admin to completely customise the registration form.

2.2 User profile and Edit personal details

For this part of the website, I have mainly worked alone, but I always got help from my teammates when I needed. Having the code for generating the registration form already written helped me quite a lot, because I was able to refactor it for generating the user profile. This page works similar to the registration form with the difference that instead of having input fields, it displays the information that the volunteer has already filled in.

The more interesting functionality that I implemented for this section, was the option of editing a profile. I had to write code that generated one category of the registration form at a time and also, the functionality to update those details in the SQL database. The hardest part of this task was implementing the restrictions for sub admins and other volunteers as it had to be completely functional and consistent with privacy standards.. I had to make sure that a volunteer's profile can be seen only by that volunteers or by an administrator who has the rights to see that.

2.3 Changelog

Implementing this task was quite easy, whilst being fun as by that time, as I had all the necessary knowledge needed for implementing everything. Having different criteria for searching for a specific event, made the task of creating the query to get that information quite challenging. The SQL query had to be constructed piece by piece, since I did not know what the user inserted (or not inserted).

```
$query="SELECT * FROM changelog WHERE ";
$comma=false;
//check if the username input field has been filled
if ($_POST['userName']!==' ')
{
	$query=$query."user LIKE '%".$_POST['userName']."%'";
	$comma=true;
}
//check if the input field for Start Date has been filled
//if yes set date '>=startDate'
if ($_POST['startDate']!=='')
{
	if ($comma)
		$query=$query." AND ";
	$query=$query."date>='".$_POST['startDate']." 00:00:00'";

	$comma=true;
}
//check if the End Date has been filled
//if yes set date '<= endDate'
if ($_POST['endDate']!=='')
{
	if ($comma)
		$query=$query." AND ";
	$query=$query."date<='".$_POST['endDate']." 23:59:59'";
	$comma=true;
}
//check if the user is searching for a specific action
if ($_POST['action']!=='')
{
       if ($comma)
              $query=$query." AND ";
       $query=$query." text LIKE '%".$_POST['action']."%'";
}
//add the sorting options
$query=$query." ORDER BY date DESC";
$logs=$this->db->query($query)->result();
```

After implementing the page for displaying the logs (and searching through it), all of us had to go through all of our code and add appropriate SQL queries where an action needed to be saved.

2.4 Events

For the events page, I worked together with Ana. We developed a basic blogging system that is easy to use both for volunteers and admins. On that page, admins can post information about upcoming events or announcements that all volunteers could see.

Designing this page was really interested, since we got a chance to use the CodeIgniter library for pagination (splitting all the events into pages with a fixed number of events per page). Configuring the functionality was easier than I expected and saved us a lot of time. For the Changelog, I did not know about that function and I ended up writing my own function for browsing through the logs stored in the database; although it was a good exercise, I would prefer not going through that effort again.

3 Discussion

3.1 Group Work

From my personal point of view, I am more than happy about the way in which the team collaborated. I have already worked with Patrick, Ana and Kristian on the business plan for High Tech Enterprise, I knew all their skill levels, that they are hard working and amazing at working on group projects. We invited Milan and Nikolay to join our group and, in the end, I believe that it was one of the best choices that we could make. With Milan's previous knowledge in web development, Nikolay's and Patrick's smart solutions, Kristian's skills in design and Ana's perseverance, we have successfully finished our project.

Most of the time, we all gathered in MVB and worked as long as possible. We had fun days and we had stressed days; but together we always managed to accomplish the tasks that we had. Working all together boosted our productivity to the maximum, as there was no communication latency between us and we could get instant feedback from each other anytime we needed.

For most of the tasks, we worked in groups of two or three people and we often used pair programming, as it proved to be an extremely productive technique. I did not enjoy pair programming in my first year and I thought that it is a waste of time, but now, working on a real project with little knowledge about the tools that I am using, I realised that it is one of the best way to work on a group project.

3.2 Work allocation

We tried to decide who is doing what in a manner that would seem fair to everyone. At each milestone, we checked what we still had to do and each of us voluntarily picked some tasks. This way, everyone had the chance to do what they were interested in and everybody was enjoying working on their task. At the same time, if someone finished a task early, they would proceed to helping others with their work, fixing some bugs or testing certain features that have been implemented. Taking this approach ensured that everyone had something to do at all times and that everyone had a fair contribution to the project; it was also very productive in the beginning when we had different skill levels.

3.3 Opinion

I strongly believe that I have learned a lot this year through this project for a few reasons.

Firstly, I started the year with absolutely no web technologies knowledge, but now I feel that I have learnt quite a lot and that I gained all the necessary skills to start working on any type of website. Working on this project showed me exposed to me some of the most frequent problems that one might encounter while developing a website and how to approach them. However, the work that I have done was really enjoyable.

Secondly, I improved my team work and communication skills. Working in a team of six people encouraged us to need to report any changes to one other in real-time and discuss the approach of any new feature. I realised that working for a project in a team can increase the productivity, just because of the simple fact that there is always someone that can give you input on a certain matter.

On the other hand, one thing that I did not like about this project, is that it did not have any major task that involved algorithms or mathematical skills (as this is the part of computer science that I am keen on); most of the features implemented only required knowledge of the programming languages used.

Individual Report
- Kristian Krastev-

1. Introduction

This report describes my individual experience of working on the Software Product Engineering project within team CAMPAN and my contribution to the creation of the Volunteer Database & Mailing List web application.

The choice to work with Ana Dumitras, Andrei Ilisei, Patrick Johnston, Milan Zolota and Nikolay Nikolov was an easy one. To begin with I had prior experience working with the first three of them on the Hi-Tech Enterprise Group project in my first year, which I found extremely pleasant and productive process overall. As for Milan and Nikolay, whom I also know from my first year, they, along with Andrei Ilisei are my current flatmates which provided the possibility to collaborate on the project at any given time which proved to be extremely beneficial. Furthermore I admire all of my teammates for both their technical aptitudes and personal qualities and ability to work in a group. I believe that each of us contributing their own unique intelligence, creativity and teamwork skills is the factor that enabled us to complete this project which involved loads of hard work and determination, as well as entertainment, in order to ultimately produce a high quality ad hoc software solution to a real-life problem.

Prior to starting this project I had very little experience with programming for the Web. I had taken a web-design course in Adobe Dreamweaver mainly focusing on using a GUI as a design tool when I was 13 and had edited the source of some of my personal blog pages for customization purposes wherever the blogging platform allowed it.

With help from my friends and colleagues from CAMPAN, I developed the web pages listed in the following section and also worked on completing, changing or fixing various components of PHP scripts across the system. Having said this, my main contribution to the project ended up being the design of the layout and style of the website, as well as a significant part of the dynamic behaviour of web pages. Taking this role was not planned but rather happened naturally since I was interested in acquiring the technical knowledge needed to make the system’s graphical user interface aesthetically pleasing, easy to use and generally user-friendly. My job also involved communicating these design solutions with the clients and ensuring the product is tailored to their needs and preferences. Overall I found this task being a challenging but rewarding one to take on. I gained valuable web design skills and knowledge in the use of PHP, HTML, CSS, JavaScript and jQuery.

2. Individual contribution

2.1. Database Browser

The first component I started working on once the development process kicked off was the pages that display the volunteers' database and mailing browser and parts of the back-end functionality that it uses to generate information on those pages. I worked alongside or in pair with Patrick whose valuable help and experience let me advance quickly in learning and using the multiple languages and technologies this very core element of our system uses.

2.2. Send e-mail page

The email facility of the application was the first component that I had major contribution to. I developed it together with Ana for one of our early releases as it is secondary to the use of the database and mailing list browser and although not to big in size it also represents one of the core functionality of our system. It can be accessed from the main mailing list browser as well as from the "show all" page, displaying the full contents of a subset or the whole of the volunteering database, and also from the page responsible for managing groups of volunteers.
It involved some low-level interfacing of our database to generate stored e-mail templates and transfer data from the browser.

2.2. Groups & Events pages

Apart from providing the design of these pages I needed to restructure some of the contents generated for them and thus helped finalize their structure and functionality. I worked in collaboration with Nikolay and Andrei respectively and all decisions were communicated and taken after reaching a common agreement, which was a generally smooth process, leading to a successful group effort.

2.3. Design style & responsive behaviour

The task of designing the website's layout and style was the one I spent the most time working on and had major contribution to. Nevertheless if it was not for the help and assistance of Milan especially in the early development stages it would have been a much more difficult and even more time-consuming one.

Creating the design style and layout of the website was a continuous process that started almost at the very beginning of the development. Initially we simply needed a means to display and structure information generated on web pages which was mainly done using HTML. As we got closer to our first release and demos of our progress with the clients the need for a design solutions emerged and what started as some styling attributes in our HTML tags needed to be transformed into proper CSS style files. Later on the introduction of JavaScript and jQuery features was also required, due to the dynamic nature of some of the pages and the need to create a satisfactory user experience.

Since a clear concept for the design requirements did not initially exist I was free to experiment with CSS which aided the learning process and enabled us to come up with prototype suggestions for the style by improvising. The CSS being the top component of the user side of web design seemed at first to be the easiest part of the project to implement. It turned out though, that producing aesthetic and well-functioning styling solutions is far from simple and straight-forward, especially for custom ones - i.e. without the use of ready-made templates provided by a framework.

The implementation of dynamic behaviour required the use JavaScript and/or jQuery. Responsive behaviour on pages that need to change their contents according to user's actions, aligning fixed table headers with scrollable bodies on loading and resizing a web page, as well as other features of the graphical user interface could not have been made possible without the use of these tools.

All in all a considerable competence in the use of the above technologies had to be acquired to produce the final user interface and helped me gain valuable web design skills.

3. Project overview & Group work

This was my first experience of working in a team on a project of such scale. I genuinely feel privileged to have worked with the rest of the team. No major conflicts or within-team issues ever emerged during the whole development process and minor ones were a rarity that only resulted in the generation of more efficient and creative solutions to design problems. Right from the very beginning each and every one of us were determined to invest 100% of our capability and effort in the development process and this had a very beneficial impact on our overall group work experience. Meeting up to make progress on our product was a choice rather than an obligation at all times and was somewhat our quality-time professional and academic activity.

Although our skills and expertize varied over the different technical aspects of the project everyone made the best use of their talent and ability and worked equally hard with the same goal to make the most of it. What seemed at first like peaks impossible to conquer, was quickly mastered, and despite the steep learning curves we all had to cope with in one way or another, we soon found out that the sky *is* in fact the limit and that in order to advance one has to be determined to do so for the sole purpose of enjoying the work process rather than reaching fixed goals as there is always room for improvement and more to do. I believe that the product resulting from our cooperative endeavour is the direct reflection of it. We always agreed that our product has to match the highest of our own standards as well as our clients'.

4. Conclusion

I feel morally satisfied by the completion of this project and its associated product. My confidence as a developer was strongly enhanced by my participation in the project. I believe that I know have the strength and determination to tackle a greater variety of problems. The experience in working for real-life clients which I gained is an invaluable and highly motivating one. Not only do I have the craft of web design under my belt but I am ready to approach my next big-scale development job a step closer to being a professional.

Individual Report
- Milan Zolota -

1 Introduction

My role in the project was to make sure the VPS and all the required tools are properly set up for web development. Besides other things, I was also in charge of making the login system, permission system, the system for managing users and any security related issues.

Prior to the project, I had already designed, coded and managed several websites, and ran a web server. Because of this, I felt like my role, besides designing and coding, was giving guidelines to other members of my team. At the beginning of the project, I created the first few pages (the login page and a simple dashboard) and created a suitable database structure before anyone else started working on the project, so that it would be easier for them to see how php, HTML, CSS, jQuery, Javascript and MySQL are related. I believe this was a really good step because it was much easier for others to pick up the relevant skills needed for this project after seeing everything work together. During the first few months of the project I found myself to be more in the role of a teacher, rather than a coder. This was due to the fact that even though I worked individually on some specific parts of the website, I spent most of my time helping others and explaining concepts when they asked for help. Towards the end of the project, everyone gradually became confident in what they were doing, so even I had my share of proper coding hours.

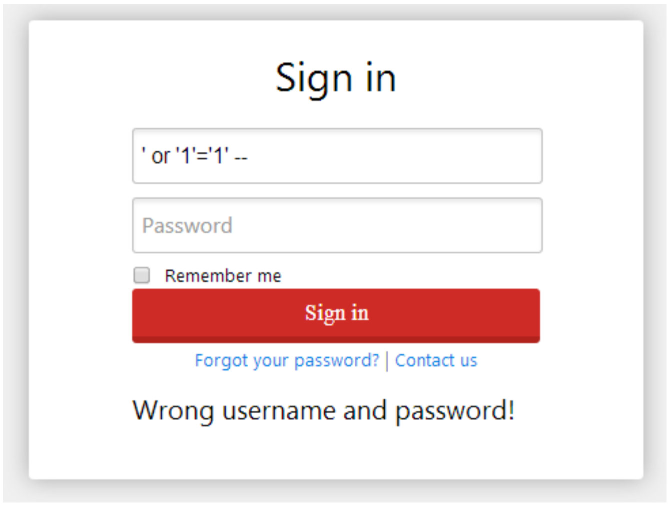
Because of what I mentioned above, I feel this project has given me more than just coding skills. I greatly enjoyed helping others and pointing them in the right direction and I believe this experience improved my communication skills and patience.

2 Individual Contribution

Besides the following things that I worked on individually and setting up my VPS, I basically took part in making almost any part of the website when helping others.

2.1 Login page

The login page is a page that can verify if a user is registered on our website. If invalid credentials are entered, an error message is displayed to the user. If they enter valid credentials of a user registered on the


Sign in

' or '1'='1' --

Password

Remember me

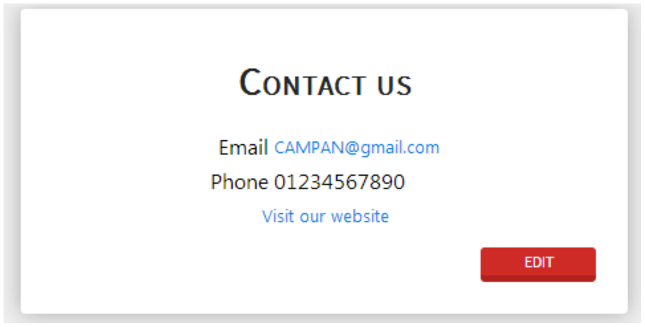
Wrong username and password!

website, they are given access beyond the login page.

The login page was the very first page created on our website. Because of this, making the page also required creating a database with a suitable structure and creating CodeIgniter models with relevant functions. Because of the nature of this page, security of this page (and the database) was given the highest importance. I made extensive research on various hacking methods and made sure our system is safe. All passwords stored in our database are hashed and salted using the function php 5.5 provides. This function dynamically changes the encryption algorithm so that it always complies with the newest standards for password encryption. The login system is also protected from SQL injection.

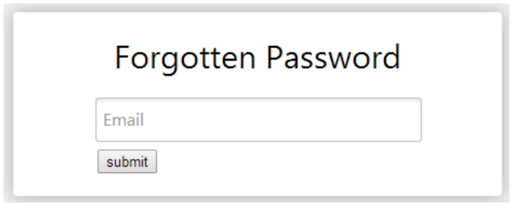
2.2 Contact us page

The 'Contact us' page is a simple, public page that lists an email, a phone number and a link. This page is public and can be accessed by anyone. Admins with the management permission are able to edit the email, phone number and link that are shown on this page. The purpose of this page is to provide both registered and unregistered users access to the contact details of the admin of the website.



2.3 Forgotten password page

The forgotten password page gives users the ability to reset their password in case they forget it. After entering their email address on this page and submitting the form, they are sent an email with a unique URL. This url leads to a page where they can choose their new password. This system is also protected from SQL injections



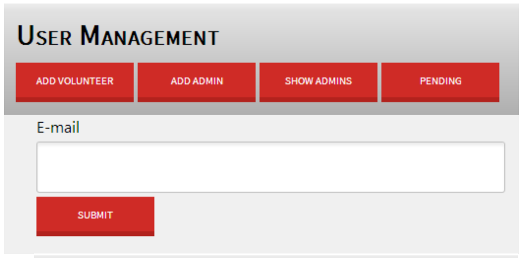
2.4 User management page

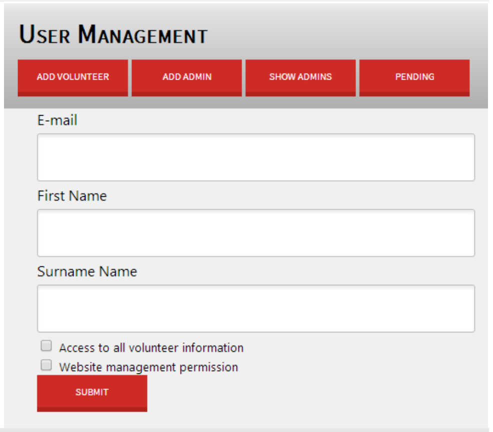
This page allows admins with the management permission to manage users. This page consists of five parts and is the core of our permission system on the website. Using the following system, we were able to separate admins and volunteers and admins with different permissions.

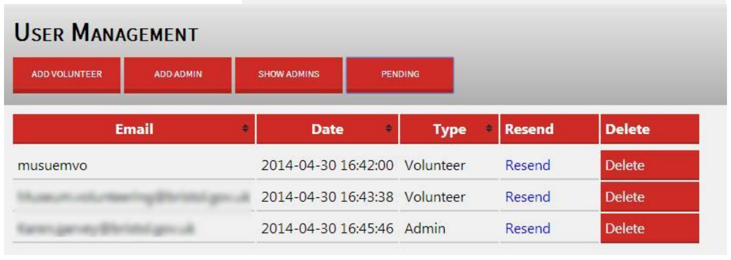
1. **Add volunteer:**
   This part of the page allows admins to add a new volunteer to the database. After entering a volunteer's email in the form and submitting, the admin's default email client will open a new email form with the volunteer's email address entered as the recipient

and a unique url entered in the body. After the volunteer receives the unique url, they can use it to register on the website.

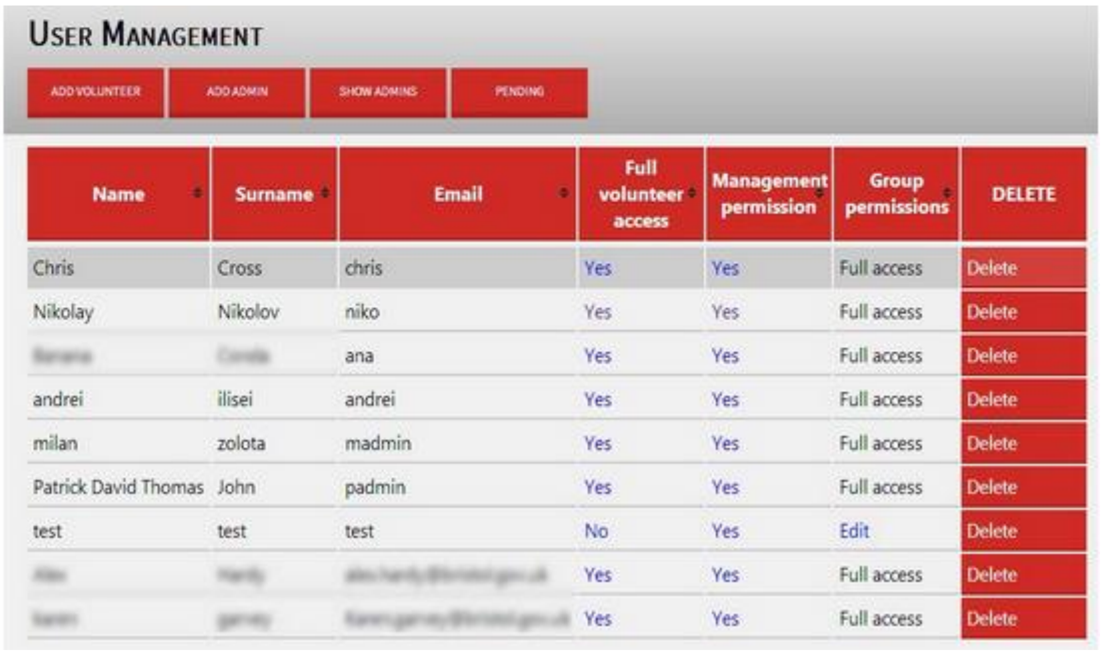
2. **Add admin**
   This part of the page allows admins to create new admins. After entering the email address, first name, surname, selecting the permissions the new admin should have and submitting the form, the admin's default email client will open a new email form with the new admin's email address entered as the recipient and a unique url entered in the body. After the new admin receives the unique url, they can use it to register on the website.
3. **Pending registrations**
   This part of the page shows a list of pending registrations, i.e. users who still haven't used their unique url to register. An admin can delete the pending registration, so that the unique url will not work, as well as resend the unique url to its recipient in case they haven't received it.
4. **Show admins**







Email	Date	Type	Resend	Delete
musuemvo	2014-04-30 16:42:00	Volunteer	Resend	Delete
[redacted]	2014-04-30 16:43:38	Volunteer	Resend	Delete
[redacted]	2014-04-30 16:45:46	Admin	Resend	Delete



Name	Surname	Email	Full volunteer access	Management permission	Group permissions	DELETE
Chris	Cross	chris	Yes	Yes	Full access	Delete
Nikolay	Nikolov	niko	Yes	Yes	Full access	Delete
[redacted]	[redacted]	ana	Yes	Yes	Full access	Delete
andrei	ilisei	andrei	Yes	Yes	Full access	Delete
milan	zolota	madmin	Yes	Yes	Full access	Delete
Patrick David Thomas	John	padmin	Yes	Yes	Full access	Delete
test	test	test	No	Yes	Edit	Delete
[redacted]	[redacted]	[redacted]	Yes	Yes	Full access	Delete
[redacted]	[redacted]	[redacted]	Yes	Yes	Full access	Delete

This part of the page shows the current list of admins and their permissions. An admin with the management permission can delete other admins as well as edit their permissions. There are two types of permissions. The first, full volunteer access, permission allows an admin to see all the information about every volunteer in the database. The second permission gives an admin unlimited access to the whole website. This includes accessing this page, the manage registration form page, groups page, stats page, manage backups page and changelog page. The permission also allows an admin to manage events and edit the contact details in the contact us page.

By default, an admin without the full volunteer access permission cannot see any information about volunteers. To give them access can be changed by pressing the edit button in the group permission column.

5. **Edit group permissions**
   This part of the page allows an admin to change the volunteer access of another admin who doesn't have the full access volunteer permission in two ways. Firstly, the groups of volunteers that the admin can access can be specified. Secondly, categories of information that the admin shouldn't be able to see can be selected.

2.5 Menu

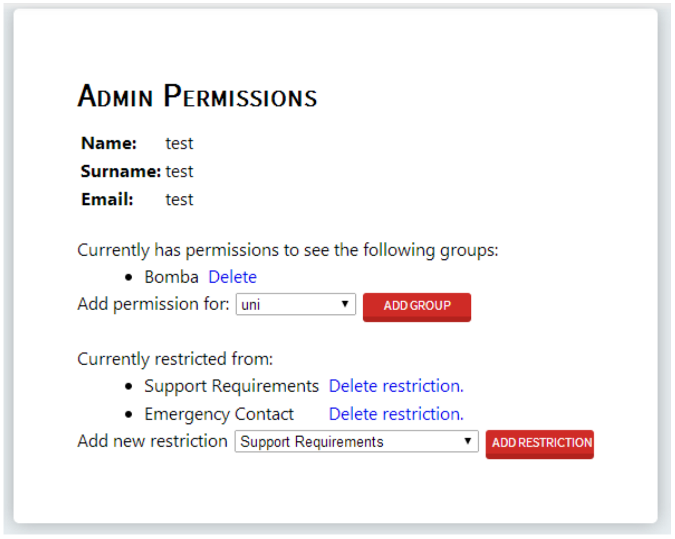
The menu is a separate view (in our model-view-controller framework) and therefore it's independent of the rest of the website. This view is loaded every time it is needed. This way it was necessary to code the menu only once, instead of coding it for every page that needs it.
It is generated dynamically and its content depends on the type of the user that requested it and their permissions. For example, the content of the menu for volunteers is different to the content for admins.
The menu also acts as a small security feature, which checks if the user is logged in. Since all the pages that include the menu require the user to be logged in, this is just another line of defense.

2.6 Other

There is a lot of other things that I have done which I cannot fully describe in this report because of the length limit of this report. Some of these include:

- Extensive security testing - Trying to forge cookies, access private website files such as the database backups, SQL injection, numerous XSS attacks, etc., and making sure that everything is as secure as possible.

- Creating a new design - On request of our client, I proposed and developed a new CSS design that was later implemented.



ADMIN PERMISSIONS

Name: test
Surname: test
Email: test

Currently has permissions to see the following groups:
- Bomba Delete

Add permission for: uni ADD GROUP

Currently restricted from:
- Support Requirements Delete restriction.
- Emergency Contact Delete restriction.

Add new restriction Support Requirements ADD RESTRICTION

3 Discussion

3.1 Group Work

The way we worked was mainly meeting up in the MVB and working all at the same time. This was mainly because we could discuss anything that we wanted to as well as help each other on the spot. I did not enjoy this approach very much because I can work much more effectively in a silent environment. However, I understood that it is best for the whole team to work this way.

3.2 Work allocation

Before each deadline, we made a list of tasks that needed to be done. Each of us then stated their preferences and was allocated a task. When someone finished their task early, they either proceeded to fix various bugs on our website, do some testing or help other people finish their tasks. I believe this way of work allocation was very fair because even when someone was given a simpler task to do, they were still useful for the team when they finished their part and worked the same number of hours as other people.

3.3 Opinion

At the beginning of the project I was a little skeptical about how much it would benefit me because of my prior experience with web development. However, I can honestly say I gained a lot of very relevant skills needed for my future career. I enriched my technical knowledge, mainly with security related issues, as well as improved my communication and teaching skills. I have already been able to present this project and what I have learnt to several employers and I believe I would not have been offered an internship with IBM if I had not done this project.

Individual Report
- Nikolay Nikolov -

1 Introduction

The aim of this report is to describe my contribution to the Volunteer Database and Mailing List project.

I had a number of reasons to agree to join my teammates for this project. To begin with, I live with three of them (Milan, Andrei and Kristian) which, as I expected, was extremely beneficial for the communication between us. Also, Milan and I collaborated on several pieces of coursework in our first year, so I was already quite comfortable with him as a partner. Moreover, I had talked to everyone in the team and I was absolutely positive that we are all hard working and experienced enough to successfully complete a project together.

After assembling a team, we were challenged to choose a project to work on. Although the decision was not exactly easy, we managed to narrow it down to three final picks, including the product we did develop - the Volunteering Database and Mailing List for the Bristol Museums, Galleries and Activities. Although the task may have seemed fairly easy at first, we (myself in particular) thought that that wouldn't necessarily be the case. Since most of us lacked experience in web programming, we estimated that we would dedicate the first month of working together to learning the basics of web technology (including languages such as HTML and PHP); we also expected that the rest of the development process would be spent tailoring a complete piece of software. The main reason for which I wished to do this project was that it would provide us with enough freedom to focus on making a well integrated product which would hopefully exceed our clients' expectations.

2 Individual Contribution

2.1 Registration page

The first task I had to complete was to recreate the registration form into a webpage, connected to the database. This was divided into two subtasks.

Firstly, creating the webpage itself. This consisted in making numerous user-input fields such as text boxes and drop-down menus. The page was modeled after the Word application which our clients used to give to the future volunteers to fill in. In addition to creating the page in simple HTML, Andrei and I used CSS to stylise it.

Afterwards, the page needed to be linked with the database in order to achieve functionality. This was done using PHP and Codeigniter functions which executed SQL queries to update the volunteer information in the database.

Since Andrei and I lacked experience in web programming at the time, we decided to work as a pair in order to ensure that both of us gained the necessary knowledge to complete the project.

2.2 Customisability of the registration form

After the fairly successful Alpha version, our clients were impressed with our quick progress, so they requested for the registration form to be customisable in case they decided to change it in the future.

Although the task led to a big increase in difficulty, I had gained enough confidence and knowledge to tackle it. And indeed, Andrei and I worked together again and succeeded.

We divided the registration in categories which contain input fields. In order to achieve this structure, we wrote pages for creating and managing categories and fields with various options such as hiding categories/features. Not all features were implemented immediately - some were introduced later which was not a problem due to our agile programming approach.

2.3 Rewriting the registration page

Since the signup page was originally static, it had to be completely rewritten in order to reflect the new dynamic registration form. This required predominantly PHP coding (PHP code generating HTML, to be exact) as opposed to the mostly HTML coding done for the original version of the page.

The view of the registration page was rewritten in such a way that all visible categories and fields are looked up in the database and appropriately displayed. This was done by PHP code which generates the respective HTML in order to display the registration page. Similarly, the controller was changed so that it can competently store all user-input information in the database.

2.4 Groups

In addition to the core features (i.e. being able to view all volunteers), the clients requested that we also implement several features in order to increase their benefits from the product which we'd provide.

One of those features was the introduction of groups. The potentially high number of volunteers suggests that managing them could possibly be rather difficult for the admin, so the ability to sort people in groups and contact whole groups at once was given to them.

Although I had mostly worked with Andrei as a pair so far, I implemented the groups mostly by myself. The reason for the organisational change was that by this point we had all significantly improved our skills, so we felt perfectly comfortable working without a partner.

The Groups page, similarly to the previous pages, lists data found in the database and allows the admin to email users, remove them from groups or delete groups altogether. Volunteers can be added to groups from Browse Database. I implemented this feature in collaboration with Kristian and Patrick.

Although the page is quite similar to the previous ones, it is more interactive. In order to achieve this, I had to include more Java Script which was challenging at times since I had only written simpler scripts for the previous tasks.

2.5 Testing

It is of high importance to me (and the team in general) that the product is reliable and bug-free. Hence, I participated in testing the whole website, not just my own code. Both black- and white-box testing was done.

2.6 Miscellaneous

Since this was a team project, I did not only work on the parts which were allocated to me, I participated in the making of most of the website. I performed various tasks, including, but not limited to, debugging, giving advice, implementing additional features and helping with research.

3 Discussion

3.1 Team organisation

Thankfully, we managed to stay fairly organised for the bigger part of the academic year. I believe that this was due to our good personal relationships and the fact that we were all quite motivated to work on the project.

We had regular meetings to ensure that everyone agrees with the current pace of work and that everyone understands what is going on even if we weren’t actually working on the project at that time due to other commitments. Although these meetings were sometimes just quick, informal chats, they were a great way of keeping everyone up to date.

When work needed to be done, we would quickly organise (either online, on the phone or in person) a meeting and we would start working as soon as possible for everyone. One of the main reasons why the communication between us was mostly smooth was that everyone was equal and the lack of a person trying to manage the others just because that’s their job made our professional relationship quite relaxed.

In addition, whenever we were unsure about the clients’ wishes, we would contact them straight away to keep ourselves informed. Depending on the information that we needed, we would either request a meeting or ask the question on Basecamp.

3.2 Group work

Although we would sometimes do work on certain parts of the project alone, we aimed to arrange to meet as much as possible. We did that in order to create a work environment where we were able to seek advice from the others, as well as stay informed about what the current status of the product is.

Employing the pair programming method proved to be extremely beneficial for the project. The fact that almost all of us lacked web programming skills in the beginning, yet we managed to implement most of the basic functionality for the Beta version, proves that it was indeed the right choice for us to work in pairs in the beginning. The only con of the method was that it usually required for both programmers in a pair to be available at the same time which wasn’t always possible.

3.3 Benefits

Working on this product was quite beneficial for me for a few reasons.

Firstly, as I mentioned, I experienced a significant increase in my web technology skills in under six months. I believe this will be quite valuable for me in the future.

Moreover, working for an actual client was rather unusual. Unlike coursework assignments, the requirements here were flexible and not always clear. Although this was slightly frustrating at times, it has taught me how to quickly adapt to change.

Being a part of a team for a reasonably long time was also quite beneficial since it was a good illustration of the need to coordinate with the others.

3.4 General opinion

On the whole, I found the development of the product to be a rather interesting experience. It was definitely different from any other projects which I have done before. Some aspects of it were enjoyable, while others were relatively unpleasant. However, the experience which I gained covers a large range of skills and it will be, without a doubt, absolutely invaluable.

Individual Report
- Patrick Johnston -

1 Introduction

I work in a group known as Team CAMPAN, along with Ana, Andrei, Chris, Milan and Nikolay. The project for which we developed and engineered software for was the Volunteer Database and Mailing List for the Bristol Museum, who after little deliberation, agreed on the development of a website to handle their requirements.

This is a familiar team, who I have very much enjoyed working with in the past, and hope to continue to work with in the future. I am always pleasantly surprised with the knowledge each of them have, the ways in which they think, the rate at which they work, and their ability to understand, as well as teach. Meeting up to work with my team is a joy, a great motivating factor, and the rewards of paired programming are reaped every time.

My experience with web development before starting this project was from my A-level computer project, which was one year long. The project was to develop a website that would show various weather-oriented reports from information gained by parsing a text file from METAR. From that project, I experimented with PHP – particularly with use of the popular framework CodeIgniter –, which was my introduction to the model-view-controller philosophy; Python, which I used to download and parse the METAR reports, lookup the location of the area referenced in the report and generate the database; MySQL, which was my RMDBS of choice at the time; tidbits of Javascript. From this, I learnt lots of good programming practices, as well as real experience with such areas of coding.

2 My individual contribution

Straight off the bat, I recommended the use of MySQL – which, due to its popularity, was accepted instantly – and CodeIgniter, based off my previous experience. I had an entire blog available for usage reference; and so after much recommendation from myself, and after some time passed for everyone to accept the MVC paradigm, we took the idea forward.

We made plans regarding the basic structure of the website; being one of the more experience members of the team, I thought I would pick an area that would be tricky to develop. As it turns out, all the areas of the website were equally difficult. None-the-less, I had chosen to work primarily on the Browse Database portion of the website; I made extensive use of the CodeIgniter database class and ensured my code was clean, well-commented, and variables well-named. My contribution with code was spread out across other parts of the site here-and-there, but I would usually prefer to guide one of the other team members through how to do it.

2.1 PHP Algorithms

The vast majority of the PHP used in the website is accessing the database, which will usually be a simple call like:

```
$this->disallowedFields = $this->db->select('ID')->where_in('category', $this->restrictions)->get('fields')->result();
```

The way the Browse Database PHP was written was reactive, it had to be written with a loose database structure in mind, due to the way the registration form can be customised. The names of the fields in the volunteer table are not suitable for representing them as the choices for criteria and columns headers in the displayed tables. Initially, a separate table was made for their displayed name, and a bijection between the two tables was necessary. Eventually, a new database structure was created, where each fields had their own table, and a field of this fields table was the displayed name; however, there were some hardcoded fields created, as well fields that exist, but are not displayed; I modified my code to get things working again accordingly.

Whilst developing support for different forms of input, some Javascript had to be generated by PHP in order to respond to the flexible database structure. The PHP iterated through each option of each checkbox list / drop-down menu and output a Javascript call that took the option and checkbox list / drop-down menu identifier. Taking checkboxes as input required specialised sections of code; options for the checkboxes were encoded specially, so that they could be used in parts of the HTML of the webpage and thus had to be decoded; input was taken as post data, for checkboxes, this meant the entire array of post data had to be searched for the checkboxes corresponding to a particular criteria; SQL had to be generated manually for the checked options due to the use of ‘find_in_set’, and so that checked values were combined as a disjunction (CodeIgniter has no way of setting the order of operations and has AND with precedence over OR).

When permissions were implemented, the code had to be modified to account for restrictions that some admins may have, this was not particularly difficult, as I had anticipated this change. Checking for which members belong in at least one of the same group as me is one of the functions I implemented in the MySQL and User models, and is little more than a pure SQL query.

The backup management page is a small page, and was developed almost solely by myself. It lists the files in the designated backups directory, sans the index.html that denies the directory listing; creates a backup of the current database and writes it to a compressed file whose name is the current timestamp; deletes the backups requested by the user; offers downloads of the backups directly; offers the upload of a backup; and handles restoration from a back-up. The restoration was the particularly tricky part, requiring me to configure a different driver to access the database with and switch to it temporarily for the function; I also had to handled the opening of a Gzipped file, which had to be done in chunks. Restoring from a backup can take some time, so a page is loaded whilst the restoration process is taking place to let the user know the site has not crashed.

PHP code of note:

Preparing SQL query after preprocessing

```
if ($field->fieldType == 'checkbox')
{
  if ($conjunction)
    $SQL .= ' AND ';
  else
    $conjunction = 1;
  $last = array_pop($field_entry['values']);
  $SQL .= "(FIND_IN_SET({$this->db->escape($last)}, {$this->db->escape_str($name)})>0";
  foreach ($field_entry['values'] as $v)
    $SQL .= " OR FIND_IN_SET({$this->db->escape($v)}, {$this->db->escape_str($name)})>0";
  $SQL .= ')';
  array_push($field_entry['values'], $last);
}
elseif ($field->fieldType == 'array')
  $like[$name] = $field_entry['value'];
else
  $where[$name] = $field_entry['value'];
…
foreach ($fieldsShown as $field)
  $this->db->select($field);
$this->db->select('ID,   firstName,   surname,   email')->order_by($sort,   $_POST['order']   ==
'Ascending' ? 'ASC' : 'DESC');
if ($SQL)
  $this->db->where($SQL);
if ($where)
  $this->db->where($where);
if ($like)
  $this->db->like($like);
if ($this->restricted)
  $this->db->where_in('email', $this->allowedUsers);
$this->view_data['users'] = $this->db->get('volunteersNew')->result();
```

Restoring from a backup

```
$file = gzopen(FCPATH."backups/$filename", 'rb');
$SQL = '';
while (!gzeof($file))
  $SQL .= gzread($file, 4096);
$SQLi = $this->load->database('defaulti', TRUE);
$SQLi->call_function('multi_query', $SQLi->conn_id, $SQL);
while ($SQLi->call_function('next_result', $SQLi->conn_id));
```

2.2 Javascript algorithms

Javascript is the area I have learnt the most due to this project. Almost all of the Javascript code that does not use jQuery was written by me, and as it turns out, I never wrote any jQuery-based code. The Javascript code was the most difficult to explain and took a significant amount of researching documentation and references to write. I wrote the code that checks all relevant checkboxes if the user checks the one in the column header, and deselects on second click etc.; this managed to make its way onto every page with a list of checkboxes, which I am happy about. I wrote all the code to transform how the value is entered for the filter on Browse Database depending on which field is selected for criteria. Finally, I tweaked the plugins we integrated whenever it was necessary to make them compatible with our website.

Javascript code of note:

Calls to routines that add drop-menu and/checkbox options, as well as criteria-field options

```
<script><?php
  foreach ($fieldTypes as $field)
    echo "AddToFieldTypes('" . addslashes($field->tableName) . "', '" . addslashes($field-
>fieldType) . "');\n";
  foreach ($fieldEnums as $enum)
    echo "AddEnum('" . addslashes($enum['k']) . "', '" . addslashes($enum['v']) . "');";
  foreach ($fieldSets as $set)
    echo "AddSet('" . addslashes($set['k']) . "', '" . addslashes($set['v']) . "', '" .
addslashes($set['v_pretty']) . "');";
?></script>
```

The function that checks all the relevant checkboxes,

```
function CheckAll(id){
  var selectAll = document.getElementById(id);
  var attr = document.createAttribute("onclick");
  attr.nodeValue = "CheckNone(this.id)";
  selectAll.attributes.setNamedItem(attr);
  var inputs = document.getElementsByTagName("input");
  for (var i = 0; i < inputs.length; ++i)
       if (inputs[i].name.substring(0,id.length) == id)
       inputs[i].checked = true;
}
```

3 Discussion

Our group works perfectly well together. We all respect each others' external deadlines, which is a major part in preventing tension between the group. We are all nothing but grateful for the work that another does. When we meet up in person, we have a bit of a laugh, which keeps up the positive spirit, which is in turn, a huge motivating factor. We set up a Facebook group for our team at the start of the academic year, and we have since been consistently active about checking it, posting anything on there that requires urgent attention or useful resources for later.

Work allocation was tricky; it was the situation where I knew what I wanted to do, but I would not be able to tell someone else something that they could do. None-the-less, I personally never felt over-worked with respect to this project; whatever I contributed, I felt it was my responsibility to maintain as best I could, and I believe I succeeded in doing so. Eventually, everyone in the group had something to be responsible for, and the entire project became well organised. Inevitably, there was code that got scrapped or abandoned, such is the nature of agile software development; however, by the same nature, there was little penalty in these losses, as such, project development never lost momentum.

I am confident I contributed my fair share; if not in text, then in wisdom. I helped with every aspect of the website, from design to implementation, from Javascript to SQL to PHP to CSS, from spitting out hundreds of lines of code to cleaning up and commenting it. I was always one that would teach to fish, rather than just catch a couple fish. Milan was very much in the same position as me, I imagine he helped out a lot at home, though I cannot say for sure. Everyone else, I am very proud of; many hours were put into the learning process so that they could keep up a good pace of progress.