[#](http://www.napier.ac.uk/)

**Graphical user interface

Description automatically generated**

**Title:** Coursework 1

**Matriculation Number:** 40512375

**Module Title:** Information Systems Engineering

**Module Number:** SET11106

**Name of Module Leader:** Dr Simon Powers

**Date of Submission:** 19/11/2021

Contents

[Introduction 2](#_Toc88210222)

[Overview of the current system, and critical evaluation of its problems 2](#_Toc88210223)

[Requirements Definition 3](#_Toc88210224)

[Functional requirements 5](#_Toc88210225)

[Non-functional requirements 6](#_Toc88210226)

[Basic High-Level Use Case Diagram of the Current System 6](#_Toc88210227)

[Detailed Written Use Case Description 6](#_Toc88210228)

[Activity diagram 8](#_Toc88210229)

[Critically assess the Systems Lifecycle approach 9](#_Toc88210230)

[Conclusion 9](#_Toc88210231)

[References 9](#_Toc88210232)

# Introduction

In current report will overview the current system that is used by the charity Community Help in order to register new clients and volunteers in their system, make the appropriate checks to register them, and how the clients have access to the services the charity is providing, in our case the delivery service and all the actions that need to be taken so the delivery will be successful and on time. We will identify all the current problems and we will explain how we can eliminate them with the development of a website that will help in storing the data and protect the clients from data leakages. We will create a diagram with all the current actions of the current system and we will describe how the computerized system will be working. Also, another diagram for the new system will be developed. Last we will explain which approach we will follow to achieve all the above and what we will need.

# Overview of the current system, and critical evaluation of its problems

According to the manager, the current system can not contribute to the achievement of the organisations goals of expanding its services across all Scotland’s branches because there is no connection to the system and the branches. Also, they can not know how many clients there are in every area and how many volunteers so can run campaigns to recruit more of them in the specific areas if it is needed. One other problem is that they can not ask every client for feedback about the delivery and that way they can not know if all the volunteers are doing a good job or they may need some mentoring. When the manager gets the information from the administrator in order to provide the quarterly report to the Board of Trustees some information may be transferred with mistakes or lost in the way.

The current non-computerised system that the charity Community Help uses is managed by one person the administrator. That means that if the administrator is absent for any reason the operations during his absences may freeze and the organisations won't be able to complete its task. Also, the administrator’s workload is overloaded because he has to do everything by himself and manually which makes him vulnerable to mistakes.

All the new clients and volunteers are signed up by the administrator by phone or via email. He saves all the information like names, addresses, contact details, etc in a spreadsheet (a different one for clients and a different one for volunteers). In that case, we may have 2 factors of human error, 1st the volunteer/client spells or writes a piece of the information wrong or the administrator does it. Also, the spreadsheets and the necessary documents like PVG checks and cars insurances are scanned files and sent to the administrator. Scanned files sometimes are not readable and this can cause more delays to the whole procedure. In order to keep all these details updated, he has to update them manually, and again is very common that a change may be wrong.

When a delivery is requested by phone or email from a client and the administrator saves the details again manually in the grocery delivery spreadsheet. Here again, we may have the same problem with the signing-up procedure. Then the administrator has manually to find an available volunteer for the current date and time that leaves close, check if he has done a delivery again for the same client because is preferred and then he has to contact the volunteer to make sure that he can make the delivery. If not he has to search all over again and find another available volunteer. All these are completed manually by one person and due to the workload is easy for something to go wrong. In addition, this consumes a huge amount of time that can be spent on some other tasks.

After the delivery has been completed the volunteer sends a copy of the receipts to the administrator so the charity can pay him back for the amount he spent. Here we may have again a bad quality scan and the wrong amount may be sent to the volunteer or the administrator may misread the amount or type the wrong amount to be sent. The administrator reimburses the amount manually by the funds they collect in cash from the client once a month(cash has to be deposited in the organization’s account). If for any reason the clients won't be at home when the charity is supposed to collect the money that may lead to delayed payments to the volunteer or the charity will not be able to organise another delivery for this client even if he needs it due to inefficient funds.

All the spreadsheets and necessary files like PVGs scans, car insurance scans, and receipts are stored on a local computer. This may be one of the most important weaknesses of the current system because if this computer breaks or just the hard drive crash all the files will be lost and will be a total disaster for the organization. Because every branch has its files in different local computers these files can not be checked or shared between them if is needed. Last but not least that can not provide safety for the stored information of the clients and volunteers and can easily leak out of the organisation and expose them to great danger.

# Requirements Definition

For solving the problems mentioned above I propose a web application connected to a database which will be stored in an external server. The app will be a simple website where members will be able to register and use the services. The administrators will be also able to register a person if he can not do it by himself or he has no internet connection.

In the app, the client or the volunteer will be able to register themselves by completing the details and uploading the files that are needed in order to use the services provided by the charity. This will minimize human error because everyone will be completing his details and not for someone else and will be able to check them again for mistakes before submitting the form to register. The system will also check with the help of an algorithm if the file that is uploaded is readable or not. If not it will ask the user to upload another file. After the submission, a notification will be sent by email to the administrator so he can log in to his admin account and check if anything is missing so he can confirm the registration. If something is missing an email will be sent to the user with further help or if they have no internet he will call them. After the registration is completed the user will be able to log in to his account and view all his details. Also, he will be able to update his details if it is needed or if the system sends him a notification that his car insurance has expired and he has to upload a new one.

The system will be connected with google maps so every client and volunteer can be categorised by the postcode. This will let the charity know exactly how many clients and volunteers have in each area and where he needs to recruit more. With the help of this tool, every time a client request a delivery notification will be sent to the admin, and when he logs in a list of volunteers that match the client's postcode and the date and time of the delivery will be available for him to choose. Also, it will a tick will be on the volunteer's name if he has made a delivery again for this client. This will save time for the admin so he has not had to look at all the volunteers manually.

From the clients' side when the delivery is completed the clients will have the opportunity to leave star-based feedback for the delivery and the volunteer which will be saved to the volunteer’s profile but will be visible only from an admin. That way it will be easy to know who needs some mentoring. From the volunteer’s side, he can upload the receipts to the expenses and the photo will be checked again from the system if it is of good quality or not.

The site will be connected with the charity’s bank account so the client will be able to make his monthly payment on time or even add funds to his account if he is running low. Also when the clients send the copy of receipts the admin has checked that the amount is correct he can confirm the payment and the fund will be in the volunteer's account automatically.

Because the site will be connected to a database all admins from all the branches will have access to all the information and that will help the expansion of the organisation. It will be a much easier way to implement the system in any branch.

With the system, the manager can access the information needed for the report anytime because it will be in digital form.

Last but not least all the files will be saved in a server complied with the data protection guidelines which will make the members feel more secure with sharing their personal information and even if the pc breaks the data stored in the remote server can not be lost and can be retrieved anytime from anyone who will have access.

|  |  |  |  |
| --- | --- | --- | --- |
| MoSCoW | | | |
| Must | Should | Could | Won't |
| Secure | Maintainable | Profile page | Make payment |
| Compliant with data protection guidelines | Connectable with other branches | Update details | Direct to bank page for payment |
| Match up users capabilities | Admin account retrieve data from database | Validate changes | Funds added to clients profile |
| Home Page | Admin register user page | Save changes to the database | More than one admin accounts |
| Register user form | Notify driver | Driver upload receipt | Admin confirm registration |
| Upload files | Driver accepts delivery | Save documents in the delivery form |  |
| Validate files quality | List number of volunteers and clients by postcode | Admin checks amount |  |
| Save files and details to the database |  | Admin send payment to the driver |  |
| Show match list of volunteers to admin |  | Feedback page for delivery |  |
| Administrator account log in |  | Feedback saved on volunteers profile in the database |  |
|  |  | Show number of volunteers and clients on a map |  |
|  |  | Admin account modify users profile information |  |
|  |  | Save modified data to the database |  |
|  |  | Request a deliver |  |
|  |  | Delivery form |  |
|  |  | Save the delivery form to the client’s profile in the database |  |
|  |  | Login user page |  |
|  |  | Validate user |  |

Table 1. Moscow

## Functional requirements

1. Home Page
2. Register user form
3. Upload files
4. Validate files quality
5. Save files and details to the database
6. Save files and details to the database
7. Administrator account log in
8. Show match list of volunteers to admin
9. List number of volunteers and clients by postcode
10. Feedback page for delivery
11. Feedback saved on volunteers profile in the database
12. Admin Select driver
13. Notify driver
14. Driver accepts delivery
15. Admin register user page
16. Admin account retrieve data from database
17. Login user page
18. Validate user
19. Profile page
20. Update details
21. Validate changes
22. Save changes to the database
23. Driver upload receipt
24. Save documents in the delivery form
25. Admin checks amount
26. Admin send payment to the driver
27. Show number of volunteers and clients on a map
28. Admin account modify users profile information
29. Save modified data to the database
30. Request a deliver
31. Delivery form
32. Save the delivery form to the client’s profile in the database
33. Make payment
34. Direct to bank page for payment
35. Funds added to clients profile
36. Admin confirm registration

## Non-functional requirements

1. Secure
2. Compliant with data protection guidelines
3. Match up users capabilities
4. Connectable with other branches
5. Maintainable
6. More than one admin accounts

Table 2. Functional, Non-functional requirements

# Basic High-Level Use Case Diagram of the Current System

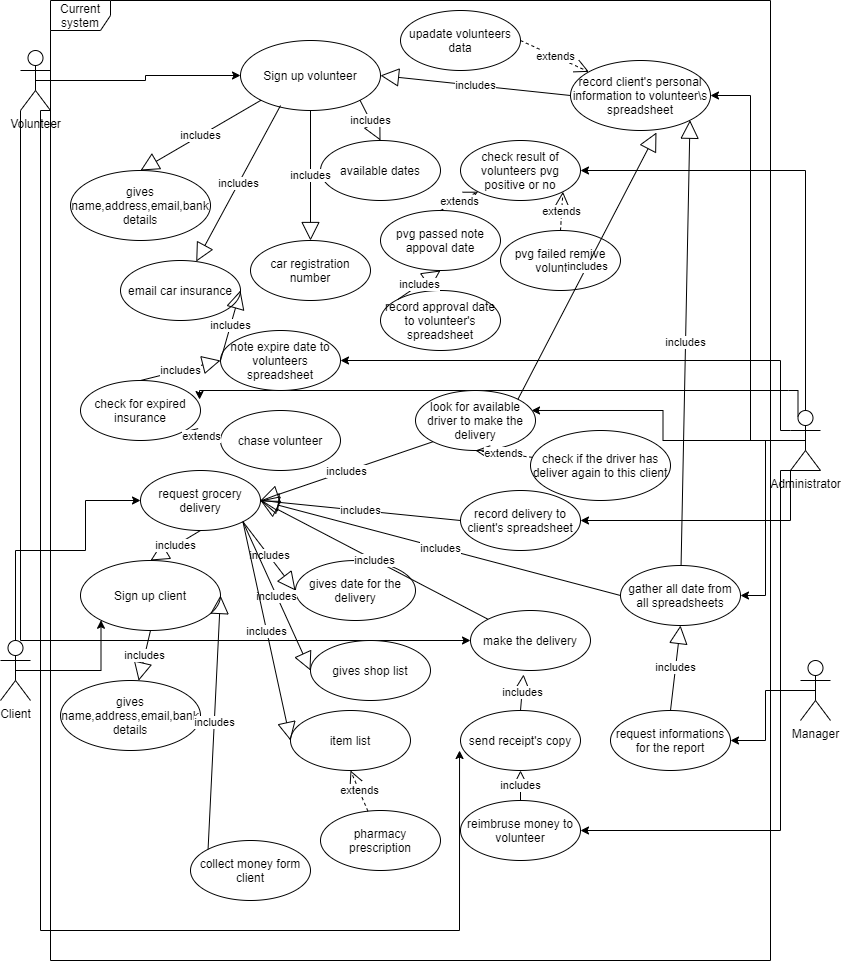


Table 3. High-Level Use Case Diagram

# Detailed Written Use Case Description

**Actors:**

John: Client

Mike: Admin

Cameron: Volunteer

**Scenario:**

John land on the charity’s site home page.

Hit the register button

Complete the register form

Requests a delivery by email

He completes the delivery form

Sends the delivery form to the charity

Mike gets delivery notifications

He logs in to the site as an admin

He types the client’s name

Clicks the match up button

The list of volunteers that lives near John and are available for the date and time requested appearing on Mike’s screen

Mike looks for a volunteer’s name with a tick on the side

He can’t find one because the client is a new user and this is his first delivery

Mike chooses a driver from the matching list

He contacts the driver to inform him of the delivery

That driver is unavailable for personal reasons

Mike finds another driver

He contacts the new driver

Cameron confirms his availability

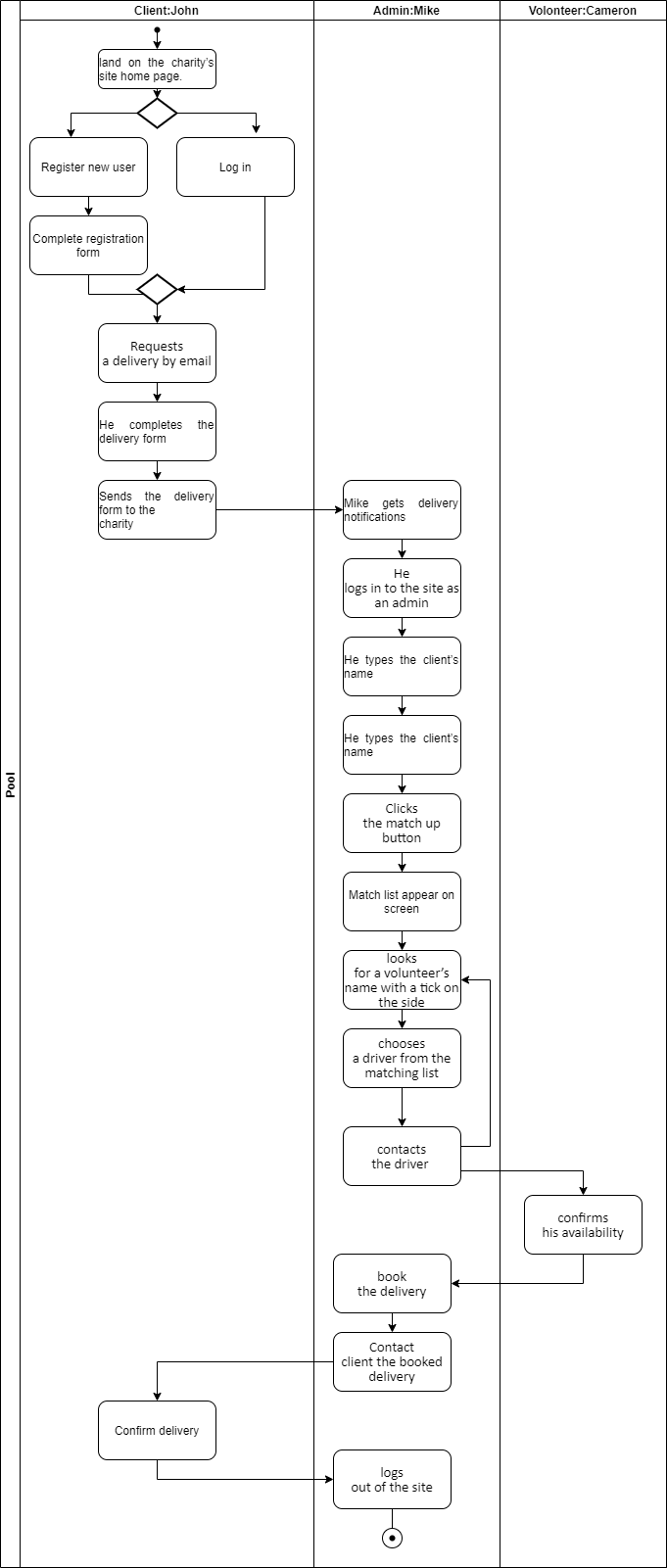
Mike confirms the delivery is booked for John’s requested time and date

Mike book the delivery

Contact John for the booked delivery

Client confirms delivery

He logs out of the site.

Activity diagram 

# Critically assess the Systems Lifecycle approach

For achieving on time all the required goals that the charity has set in order to be able to expand the system in the next 6 months and develop a functioning, robust, scalable and trustworthy/ secure, and maintainable app the best suited is the agile methodology with the extreme programming (XP). XP aims to deliver high-quality software in a short period of time time. (Extreme Programming (XP), 2021) so we can have the app ready in the period requested. Four developers will be needed to so 2 of them will be developing the front end and the other 2 will be working on the back end of the site. Also, there will be a tracker where will overview if the deliveries are completed in the scheduled time, follow the prioritized flow that the client has given and give feedback to the charity for how the development of the site is progressing. The tracker will also feel the gaps and give help to the developers if it is necessary (What is XP in Agile? (Roles, Processes, Tools), 2021)

An office near the charity or in the charity is required so the charity’s representative will be able to come in contact with the team at the start of every sprint and take part in the meetings as this is one of the most important characteristics of the XP( the client should be on the site to give answers to any questions that the developing team might have) (Beck, 2000).

The tools that will be used will be the On-site customer in order for the developed site will be maintainable for the charity. Testing will be occurring at the end of every sprint to make sure that the program is running without bugs and is fully functional. Pair programming also will ensure that fewer mistakes will happen in the code because 2 pairs of eyes are always better than one and that will help the team also stay on schedule and develop a robust website. XP approach in every sprint delivers the developed app a complete application that works on its own. That is achieving the site to be scalable so every time that something needs to be added can be done in a different sprint without causing any problem to the previous stages. With the second team of developers working in the back end and the test that will be running in every sprint will be made sure that the site will be compliant with the data security guidelines so the users will feel safe to share their data (What is XP in Agile? (Roles, Processes, Tools), 2021).

# Conclusion

After we interviewed the charity’s administrator and manager we have located all the current problems that they mentioned and some others that they haven’t mentioned for the current problem. Our proposal gives the solution to all these problems and the diagram of the current system helped us to identify even more gaps the system has and develop a correct description and diagram on how the system will eliminate all the found problems. We proposed the approach that should be taken and how many members must the team have to develop the website and be functioning, robust, scalable, secure, and maintainable. All the steps that will happen during the development and how this will achieve all the above are mentioned in our report above.

# References

# References

Beck, k. (2000). *Extreme programming explained: Embrace change.* MA.: Addison Wesley Longman, Inc.

*Extreme Programming - Roles*. (2021, 11 14). Retrieved from Tutorials Point: https://www.tutorialspoint.com/extreme\_programming/extreme\_programming\_roles.htm

*Extreme Programming (XP)*. (2021, 11 15). Retrieved from Agile Allience: https://www.agilealliance.org/glossary/xp/#q=~(infinite~false~filters~(postType~(~'post~'aa\_book~'aa\_event\_session~'aa\_experience\_report~'aa\_glossary~'aa\_research\_paper~'aa\_video)~tags~(~'xp))~searchTerm~'~sort~false~sortDirection~'asc~page~1)

*What is XP in Agile? (Roles, Processes, Tools)*. (2021, 11 14). Retrieved from Click up: https://clickup.com/blog/what-is-xp-in-agile/

# Appendix

Table 1. Moscow

Table 2. Functional, Non-functional requirements

Table 3. High-Level Use Case Diagram

Table 4. Activity diagram