2022

**Josiah Rowden**

**A-level Project**

**Start date: 08/02/2022**

**End date: N/A**

Cadet Link

A picture containing text, cup

Description automatically generatedLogo

Description automatically generated with low confidence

Contents

[The Problem: 3](#_Toc115334461)

[The Current System: 3](#_Toc115334462)

[Stakeholders 4](#_Toc115334463)

[Why this Problem is suited to be solvable by computational methods 4](#_Toc115334464)

[Computational methods I will use: 4](#_Toc115334465)

[Problem Decomposition 4](#_Toc115334466)

[Divide and conquer 5](#_Toc115334467)

[The Analysis 6](#_Toc115334468)

[Similar Products 6](#_Toc115334469)

[Meetings 8](#_Toc115334470)

[Initial Survey 8](#_Toc115334471)

[Requirements 12](#_Toc115334472)

[Success criteria 13](#_Toc115334473)

[The Design 14](#_Toc115334474)

[The Overall Design 14](#_Toc115334475)

[A modular Design: 14](#_Toc115334476)

[User interface: 16](#_Toc115334477)

[Log in and Initial User system 24](#_Toc115334478)

[Central dashboard 25](#_Toc115334479)

[Stock handling System 26](#_Toc115334480)

[Kit Request System 27](#_Toc115334481)

[Full User System 29](#_Toc115334482)

[Event Management System 30](#_Toc115334483)

[Help system to explain how a system works 30](#_Toc115334484)

[Test Planning 30](#_Toc115334485)

[Errors I will look for and How I will look for them 30](#_Toc115334486)

[Iterative testing 30](#_Toc115334487)

[The Software Development 31](#_Toc115334488)

[References 32](#_Toc115334489)

[Common abbreviations and phrases 33](#_Toc115334490)

# The Problem:

Through my experience in a cadet force I have learnt many things, one of these things is communication is verry important, however as highlighted by experience, communication is not always easy for a Cadet force, especially one that is split over two schools such as mine. Getting new kit, communicating about lessons and activities for each week, and receiving information about trips are, despite the efforts of our CFAV’s (Cadet Force Adult Volunteers), less efficient than what would be desirable – especially trying to get new Uniform. These issues are normally not because of Our CFAV’s but mainly due to the amount of paperwork required in these tasks, paperwork that I believe could be streamed lined by computational methods.

## The Current System:

In this section I will outline some of the current systems related to what this project is about. Although I might not decide to replace or improve the systems, what I mention here I necessary to understand how my system that I will develop will integrate into the existing system. As such I will outline how the system currently works, the problems with this system, the good things about how it currently works.

* **Kit Request:** This involves emailing a CFAV, normally Lt Howson our SSI (Senior School Instructor) and quarter master either collectively or individually. The email should contain the size and type of uniform requested and the reason why you are requesting new kit; details that are often missed meaning the email needs to be resent. This email can get Lost in inboxes and sometimes the CFAV’s are simply too busy to deal with a request. This results in long delays for new kit or in some cases new kit is never issued; this is obviously not ideal.
* **Information Transfer:** Our CFAV’s Produce a document that outlines who is receiving what training when and where along with any other information like dress each week; we refer to this as orders. The problem is not the Orders format but how they get to cadets – this is a problem that is exacerbated because our cadet force is between two schools – orders are generally emailed, Whatsapped or sent through teams often reaching cadets a day before or even on the day.
* **Kit/ Stock Management System:** Uniform is kept track of on a manually updated Excel spreadsheet. This has the inherent problems of manually entered data of human error. As I understand it the spreadsheet is stored locally on the QM’s device meaning there is only one UpToDate copy and any others that other CFAV’s have will go out of date as soon as something changes with the QM’s version.
* **Communication and Coordination:** Communication is generally done through email, and physical meetings although the Senior NCO’s do have an informal group chat on WhatsApp. In this system there is no central space for communication, which limits coordination and collaboration and overall cohesion of the Force leading to times where we are told what groups we are instructing when we turn up.

## Stakeholders

Who use the current system and would use the new system and how.

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Role | Interaction | Availability |
| Cpt Aggrey | User: Contingent commander | Would use to issues orders and post information about trips | Once a week in person, email |
| Lt Howson | User: Quartermaster/SSI | Would use the app to sort out uniform issues and know what Equipment is needed for each session | Once a week in person, email |
| A Cadet | User | Would Use the App to Make Requests and Find orders for next session | For TJWA cadets any school time,  Benenden, once a week, possible email/WhatsApp |

## Why this Problem is suited to be solvable by computational methods

This problem is suited to suited to be solvable by computational methods for a variety of reasons. My planned solution is to use a web-based app that would be accessible to download on any device or be accessed through the internet. Each cadet would have their own log in Information about trips, orders for this week, what kit to bring (and whenever its short sleeves or not), scores for section competitions and a way for uniform issues to be logged would all be displayed in an easy-to-use personalised central display. *The Exiting System is outlined in the first section of The Problem*

The benefits over the existing system would be, a central space for communication, essay use from anywhere (with internet connection), a full computerised database which would allow easy data retrieval, comparison, and entry, removing a lot of human error and allowing for easier management of stores and hopefully lead to cadets getting new kit more sooner.

## Computational methods I will use:

### Problem Decomposition

This Project is verry modular as each feature are mostly independent of each other; this will allow me to tackle each problem individually, hopefully leading to a more complete project even if all planned features are not implemented. As such this also leads to the ability to easily add new features even after launch. As such I have split the purposed features on terms of most to least important:

1. **Database**, **Login and Signup** and **Dashboard** systems
2. **Uniform request** system
3. **Trip manager**, **Section and** **Troop manger** and **Order**s systems
4. Diagram

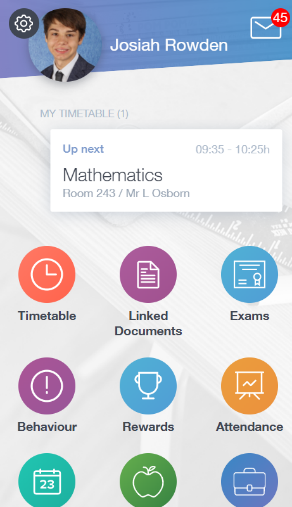
   Description automatically generatedThe ability to **talk to an NCO**, and any other systems could be to implement

### Divide and conquer

I have split each section into different html/php pages and works on each page till I get it at a degree of functionality that I consider it workable or completely finish it I use the below diagram to help me chose what page to tackle next:

# The Analysis

## Similar Products

* **Edulink One:** Edulink is a Software that My school uses to set Homework; Promote trips; generally, Communicate. It has a central dashboard which acts as hub for all the features of the App, something that I am planning to replicate in my design.



Graphical user interface, application

Description automatically generated

A screenshot of a phone

Description automatically generated with medium confidenceHow EduLink’s Login system works is by having a two-stage log in system; it first asks for the school ID or Postcode to differentiate the school. It then asks for a user log in to differentiate the person, this allows multiple schools to use the same login system. This is something I’m thinking of implementing if the system grows beyond my cadet force.

As Mentioned previously, Edulink uses a central Dashboard system where links to all the functions of the app such as the timetable can be accessed from, it also has some quick access information like next lesson and such. One of the novel things that Edulink does is has the ability for client schools to choose the functions they want. Although I think this is a verry good feature I believe it works due to the amount of functions Edulink can provide to the user, as such I believe a feature like this for CadetLink is beyond the scope of this project, though it is something to keep in mind if it does grow larger.

* **MOD Portal:** *description to be filled in when I have the info to do so*
* **Winchester:** *description to be filled in when I have the info to do so*

## Meetings

I have Met with Lt Howson (*Stakeholder*) discussing What he would want In Such a software that I would want to develop. He was Optimistic about the prospect of a web-based app saying “It’s all going to be in one place. It means things can be updated and shared easier and hopefully more people can understand how it works.” And “It's less work for me, which means I can then do other work.” He also gave Me Some other Ideas for updates in that could come later – “If it was to do something to do with bar codes and things, that would be quite useful, I could scan, kit out and then gives me a full running total of each item with each bit of equipment. That'd be quite handy.”

## Initial Survey

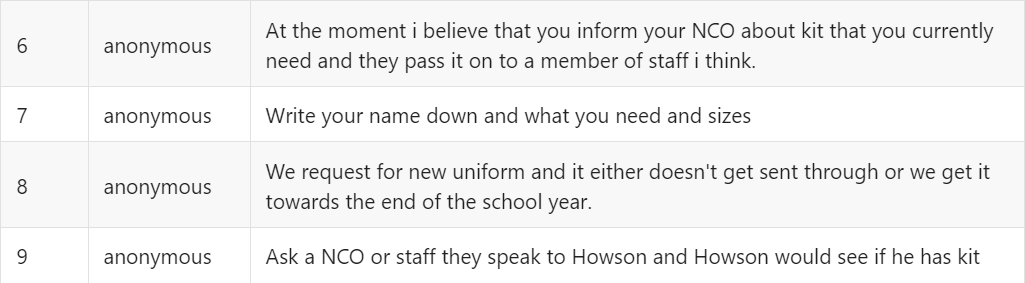
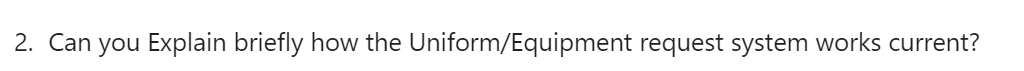
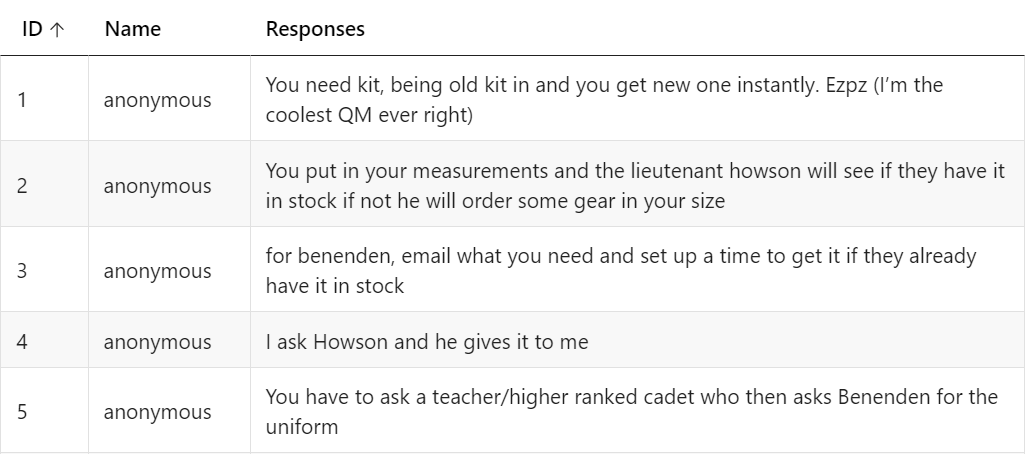
I wanted to conduct a survey of cadets (the end users of the app) to validate what I felt about the current system and gauge interest in a cadet app. I also asked them about any features they might want. I have had 10 responses at time of writing which is just under a third of the cadet force, so although not a census I believe it is a good representation of the cadet force as a whole.

A picture containing chart

Description automatically generated

The verry even spread surprised me a bit; I was Expecting this to lean more towards the difficult Side. I believe the reason why I thought this is because of the fact that our cadet force is split across two school, with the Quartermaster’s stores being at the Benenden whereas I am at John Wallis. Those who responded with easy or very easy are from Benenden as they have the whole week where they can speek the Quartermaster, Lt Howson, whereas those from John Wallis can only do so via email or on Thursdays, hence the split.

This Does show me that the current system Works, at least in part and as such I believe my app should work to supplement and improve the current system, not replace it.



The response to question 2 is Much the same as question 1, though the response with ID 8 shows the levels’ of frustration that I have experienced waiting for kit and what sparked the idea for this app

Chart, pie chart

Description automatically generated

Another split between Always and Rarely. But Overall trending towards Always. This is roughly what I was expecting, The Benenden Cadets either get Orders through their school Email or the Teams, the senior NCO’s (SNCO’s) have a WhatsApp group chat which the Orders Normally get posted to meaning that if the John Wallis Cadets SNCO’s don’t get it through their school email we normally see it before Thursday training. Those who Responded with Rarely are John Wallis Cadets in their first or second year at Cadets who normally don’t get sent anything.

Through that App all the cadets would be sent Orders hence removing the Communication device shown here.

Chart, bar chart

Description automatically generated

The Spread here shows the Many means my cadet force relays Orders, the strength with this is that if one system goes down there is another way of communicating those orders. I note the amount of responses the say they are verbally or physically given Orders, this suggest that Cadets don’t know where to look for Orders and just wait until they are told them, which is usually on the day of training as we do not have another time where we gather as one group. This highlights the need for a centralised system where cadets can find at any time Orders for the upcoming training sessions.

there weren’t any responses to Question 4 worth noting

Chart, pie chart

Description automatically generated

This quite clearly shows what Cadets think of the app idea.

Chart, bar chart

Description automatically generated

Graphical user interface, text, application, email

Description automatically generatedAs does this. The person who put other did not specify why.

Comments on the Cadets Ideas

1. Short answer: No I am not planning to ad Emojis as I don’t see the point
2. This is Something I have been thinking of doing however don’t plan to for quite some while as the complexity of it will add complication and time to the development of the app and although is a nice feature is not necessary for the app to function. So maybe in the future
3. This Is planned to be Part of the Events System which will address the Orders problem
4. What you Outline here already exists, Its called Orders, the trick will is you seeing the Orders before Thursday training Session and as in question 3 this is planned to be part of the Event system.
5. I’m sure I can implement the word No into the app somewhere.

I will continue to Use Surveys to gauge how well I have designed the systems, especially the UI.

## Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| Requirement | Nº | Description | Rational |
| Kit Request System. | 1 | System Should be able to Manage a Kit Request from a Cadet, Notify and Present the Request to the QM which should them be able to use the system to find the Kit and Issue to the cadet. | This part of the System would replace the Current System which has many flaws and is a need if this system is to be Successful |
| Kit Management System | 2 | Allows the QM or other CFAV’s see the Stock Numbers and Add or Adjust where needed  And search for specific Stock | Replacement Of existing System: Will be simpler and easier to use than the Existing System |
| System User Interface | 3 | A User interface that is intuitive and similar to what users | If a Computer System is overly complicated it slows down the function of the users and makes them frustrated which will ultimately lead to the System not getting used. |
| System needs to work on a multitude of devices from Phones to PC’s. | 4 | Working meads there are limited issues with elements overlapping – all key data is visible and accessible to be able to use | The system needs to be Regularly available on any device a User might have so it can be used when they need to |
| Central place for all functions and data transfer | 5 | A system where all the functions of the app are connected together in an easy-to-use format | One of the problems of the current system is decentralised way of sharing data (as shown by the survey) this app is meant to remedy that but would fall into the same problem if all the systems that make up the app are not connected together. |
| Event Management System | 6 | A System where Admins create and manage events and users can view and confirm their attendance | By Implementing such a feature Cadets will have one simple place to look to know what they are doing each training session |
| User System | 7 | A system that can manage a user’s profile and be used to automatically give ownership to a request for example. | This will be useful to manage request but also as the App is supposed to be a closed system so only known people can access and use the app, a user system will enable the app to be separated from the wider internet. |
| Differentiate between admin and non-admin users | 8 | A system that can Differentiate between admin and non-admin users | This is so only desired users can access functions that are intended to be used by admin users |
| App is secure | 9 | Only known user to the CF can use the app. | This is to prevent unwanted users interfering with the app either maliciously or unknowingly |

## Success criteria

Using the 4-stage table in the **problem decomposition** section I will grade:

If I only I achieve systems in **group 1** I would consider the project to be a failure as the skeleton of the program would be operational, but it would lack any useable functionality. If I only achieve systems in **group 1** and 2 then the program would be operational with limited functionality, I believe this would be a good beta version and a limited success as the program would be somewhat useful. If I achieve groups **1**,**2** and **3** the program would be operational with a good amount of functionality; I would consider the program largely successful. If I complete all four groups, I would consider the program greatly successful as functionality is achieved to a greater deal than what I have planned and shows the program is useful.

*\* I consider a system complete where it runs with minimal bugs and full planned functionality is operational.*

Other Goals

Other than what Implementing systems, my other goals are for the code to be readable and easily understandable, this means extensively commenting code (ahhh!). The rationale behind this goal is as I will not all ways be at my cadet force and when I leave the system will still need to be maintained and if new features are desired, I hope others may be able to add to the program. This would also be useful if other cadet force wants to use the program as it will have to be adaptable to their needs.

# The Design

## The Overall Design

CadetLink will be mainly a solution to two main problems: Kit Request and Communicating Orders. As such I will need two main systems to solve these Problems, a Kit Request System, and an Events System. To be able to use these systems, I will also need a User Interface, a Data base System, a User, and log in System.

As previously stated, I plan to design an system similar to that of Edulink One with a GUI connecting a multitude of systems connected to a central ‘dashboard’.

##### Computer Languages

The Language I Plan to Use for Most of the Systems is PHP, as its wide usage means there is a lot of support for it and is stable. It is also Relatively Easy to understand which is one of my requirements so that other coders can use my code if so required. It also has easy SQL integration which will help greatly as my Systems require a lot of interaction with a database, especially with PDO which is the Data access extension I plan to use primarily because of its proof against SQL injection attacks

The use of PHP will be assisted by using JavaScript in areas that are not needed to interact with the database; this will speed up the Program as JavaScript is a client-side language which does not need to ping the server every time it wants to do something. I am also looking into Using AJAX to make the app seem smoother, but this is a longer-term goal.

To interact with the database, I plan to use SQL for much of the same reasons I chose PHP; it is widely used so that it has a lot of support for it on the internet and also any other coder should understand SQL if they wanted to add or adapt my code for their purposes.

For the structure of the web pages, I will be using HTML.

And for the styling I will use CSS.

I do not plan to use any frame works at current.

### A modular Design:

I plan to split each section of the problem into different sub sections where I can more easily develop and test each new system; this will also help deliver a working solution to the problem even if I don’t finish all the planned functions. This divide and conquer approach will help me keep on top of the development and enable me to create systems with 100% (or close to that) functionality rapidly.

#### Development plan:

1. Log in and Initial User system
   1. Back-end database work to create necessary tables to store the username and password of a user
   2. Use of PHP to validate the data sent from the user and using SQL via PDO check that it matches with a known username and password.
   3. Use of HTML and CSS to create a GUIs for both phones and computers.
   4. Use of JavaScript for initial validation and error management system
2. Central dashboard
   1. Use of HTML and CSS to create a GUIs for both phones and computers.
   2. Back/logout button.
   3. Buttons to go to each function page
3. Stock handling System
   1. Back-end database work to create necessary tables and relationship between them
   2. Front end page that allows the User to view the stored data in the relevant tables in a useable way using PHP’s PDO data access extension to interact with the database via SQL
   3. Use of HTML and CSS to create GUIs for both phones and computers.
   4. Create PHP/JavaScript/AJAX functions that allow modification and deletion of the rows in the database
   5. PHP/JavaScript/AJAX function that will sort the data
4. Kit Request System
   1. Back-end database work to create necessary tables and relationship between them
   2. Font end page that allows users to request Kit (HTML, CSS) .
   3. Use of JavaScript for initial validation and error management system
   4. Back-end page that checks if kit in stock or not and performs appropriate action based on that and returns appropriate to user message.
   5. System that shows Users’ requests and allows them to modify and delete them (PHP, SQL, AJAX/ JavaScript?)
   6. System that shows the Requests to an Admin and allows them to modify/ respond to the request. (Intended User the CF QM)(PHP)
5. Full User System
   1. A system that allows Admins to create manage, and delete user profile (PHP, SQL, AJAX JavaScript),
   2. Use of HTML and CSS to create a GUIs for both phones and computers.
   3. A system that displays relevant information about each user that are stored from in the database. (PHP SQL)
6. Event Management System
   1. Creation of relevant database tables
   2. Use of HTML and CSS to create a GUIs for both phones and computers.
   3. A system that displays relevant information about each user that are stored from in the database. (PHP SQL)
   4. A system that displays upcoming events on the dashboard of all users
   5. A system that displays all events
   6. A system that allows an admin to create, manage and delete events
   7. The ability to add a file (Intended Use: Orders) to an event.
7. Help system to explain how a system works
   1. Use of HTML and CSS to create a GUIs for both phones and computers.

## User interface:

#### **Goals:**

#### **Must be Simple and Intuitive for the average User.**

#### **Must be Adaptable for All Devices**

#### **Use colour scheme that is Linked to User’s background I.E. The Army Cadets**

##### Pages and flow:

Users will Log in from One central page, regardless of if CFAV or not.

This will Move them to a **Dashboard** where they will be able to see a small amount of relevant data such as the next event and the info for that. This will also be the page where the buttons that link to the function pages such as the Kit Request and the Virtual Stores pages.

Users will be able to go back to the Dashboard from any function pages. Users will not be able to go from one function page to another. It is worth noting that function pages are grouped in related sections, I.E., the page to make a request and see your requests will be in one function group and only has one button on the **Dashboard** that is linked to it; when I refer to a function page from the dashboard assume I am talking about all the pages in that group. There will be pages that Can only be accessed from a function page, I.E., an edit row page of a table, they will link back to their parent function page.

Diagram

Description automatically generatedThis System is very similar to Edulink and other systems like it; this is intentional as after using said programs I believe the system work well enough. This will help the user navigate My Program if it is similar to something that they are used to.

*DFD to Show flow from Login to Function page. (right)*

###### Colours

I have Chosen Colours from the Army Cadet Website (see References) as it links My website to the Cadets, and it effectively conveys what I want.

|  |  |  |  |
| --- | --- | --- | --- |
| Colour Hex code | Colour Name | Sample | Usage |
| 002B17 | **Dark Green** |  | Will be used for background of generic elements – taken from the army cadets’ website |
| FFFFFF | **White** |  | **Will be Used for Text on elements with Dark Green back grounds and for back grounds of Elements I want to Emphasize** |
| 000000 | **Black** |  | **For text where White isn’t appropriate** |
| F5EA00 | **Yellow** |  | **Used on text and some Elements where I want to highlight them, for instance when a user hovers over a button - taken from the army cadets’ website** |
| 032F52 | **Ben Blue** |  | **One Of the colours of my Cadet force used for dashboard button** |
| AE1E30 | **TJWA Red** |  | **The other colour of my Cadet force used for dashboard button** |

###### Layout

*Graphical user interface, application

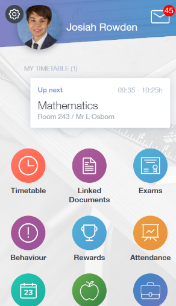
Description automatically generated****Concept of How I want the Dashboard to look – On a Laptop/ PC***

## 

Graphical user interface, chart, bubble chart

Description automatically generatedI want the Dashboard to be Simple, easy to use, yet still display the relevant information. As such I have Lage central body where the user attention will be attracted to that most because of the different colours. This is intentional as it is where almost everything the user will use to interact with the site is located.

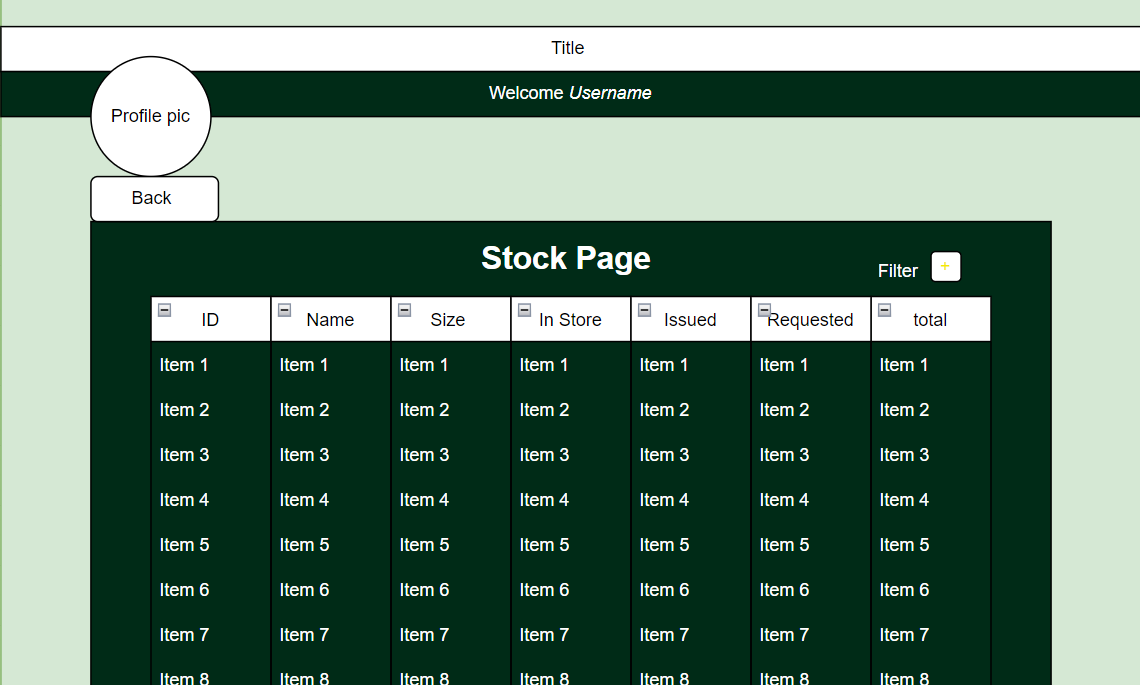
Edulink Computer Dashboard Layout

The Events section will be a table of some Kind with some key information about the displayed event and link to Orders Document. Previous or Later events will be viewed by pressing the respective buttons on the side of the events section.

The Log Out button and the Profile picture Positions are not finalized.

This is again very similar to Edulink One’s Page layout – especially on mobile on mobile, with an events section and Buttons leading to function pages. This is natural as Edulink is one of my main sites I’m taking inspiration from. Where it differs is the Size and number of buttons; I will have significantly less functions Pages than Edulink and as such I can afford to have lager buttons that feel the page, which should help users click the button they wanted first time.

2 EduLink’s Mobile Dashboard

***Concept of how I want a function page with a table to look***

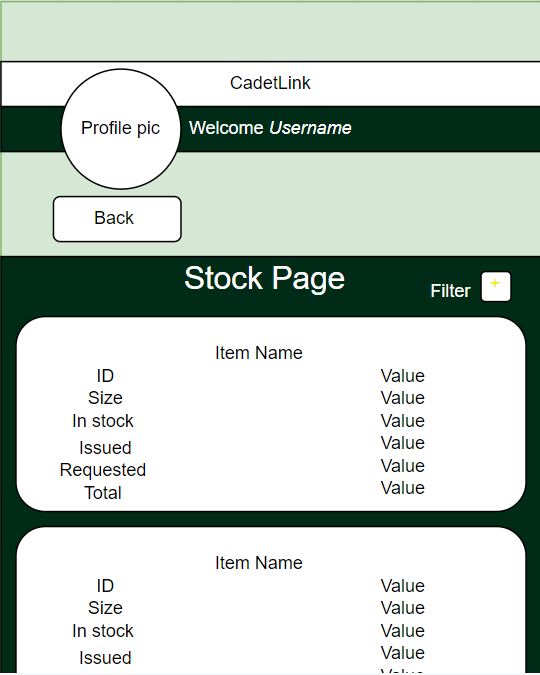
The overall Design is simple and to the point the table will be generated by an SQL query to the related table(s) in the data base

The Filter icon will open up a dropdown menu when clicked which will have varying ways of filtering the search.

##### On Mobile:

**Graphical user interface, text, application, chat or text message

Description automatically generated**For Usage on mobile, I plan to just scale Everything Down. The Margin for the central body is removed to make space for the largest buttons as possible. This is another reason why having bigger buttons is important.

For the Function Pages which on the laptop/ computer display would normally have a table I have decided to split the table down into Boxes for each Item as I deemed this to be the best way of showing the information on a smaller screen. I did consider having the user switch to a landscape mode and still use the table, but on testing this doesn’t work verry well

##### **Database Design:**

###### Previous System to store data

Chart, table

Description automatically generatedThe images to the left are an example of how uniform data is currently stored and managed. The user manually updates each field when needed, excluding totals column that is update by a simple formular: the sum of in store and issued.

There is Some elements of this that works well, it is simple to read and understand: data can be read easily. Most of the columns store data atomically which will be helpful when transferring to a data base. As such I want to use a similar layout when displaying the data to make it more familiar to the user and to uses the features that work well.

Table

Description automatically generatedHowever, there are some issues, for Uniform that does not have three sizes there are blank column. And as the data is entered manually it is prone to human Error.

#### **Goals:**

#### **To Use a Similar Format to Clients Exiting Spreadsheet.**

#### **To Use a relational data base.**

#### **To have all data in 3rd Normal For**

#### **To have a consistent naming Scheme so the data base is easily understood by another person.**

###### Requirements

* table(s) to store items in stock
* table(s) to store Requests
* table(s) to store users
* table(s) to store Events

###### Design:

**Items and Sizes tables**

This table store the Items that the Cadet force is responsible for. This covers Items unreserved in Store, Items Issued, Items Reserved in store, and Items that have been Ordered as represented by their respected column names. There is not a total as in client spreadsheet column as this can be Calculated from the sum of other tables by the program displaying the data if necessary.

I decided that, like what is done on the user’s spreadsheet. I will split sizes from the items in its own format to avoid repeated data. Although it would be relatively easy to Store them together, I believe it would be easier for comparison if they were separated out. For instance, a User can search for an Item with chest of 160 and multiple items would appear; this would be a useful feature for fitting out a new cadet.

They are split into the Item and all the individual sizes that item may have; as such they are connected together with a one-to-many relationship; sizes have a foreign key ItemID which links to the ID of items. Sizes has another foreign key that link to a look up table for the different size types, Height, Chest etc. This was Done so that if I wanted just all the different size types I could have one table with them all. This relationship is one-to-one.

The Item Type table is a look up table for all the different types of Items stored in the Database, it is linked to from the Items and ItemRequest tables.

**ItemRequest and SizesRequest tables**

These tables store Item Requests from users making them through the system. As such they have a UserID foreign Key to denote what user is making the request. This is link to the users table with a one-to-one relationship. The StockID column is another foreign key and this links to the ID column of the ID table; this so that the number columns in Items table can be updated where an appropriate request is made. SizesRequest is to ItemRequest Exactly the same as Sizes is to Items; the reason I decided to have it and not just work off the sizes of the item in items that is linked to by the StockID is to reduce complexity and so a user can request an item with a size that may not be in the database.

The Purpose column is one requested by the Client, they want to make sure that every request is valid, so they requested a reason or purpose for each request be submitted when a user makes a Request.

DateNeeded is planned to be optional, the idea going forward within my cadet force is that Senior NCO’s start using their CFIT skills to actually take some lessons, as it would be helpful to have equipment for Such activities, Item Request is a way of doing that and being able to specify what day you need that equipment is verry handy. It in Date Time format so it can be as specific as possible.

DateRequested is a field that will be automatically filled out when the request is submitted, its purpose is so that the QM or whoever is issuing the Items can see the request that has been waiting the longest and hopefully get to them first.

Status is a column that will tell the user at what point their request is at. Planned Statuses are ISSUED, TO BE ISSUED, ORDERED, AWAITING ORDER. These roughly correspond to the number Columns in the Items table. This is intended so that a Request with an ORDERED status would increase the value NumOrdered in the Items table when made.

**User Table**

The Users table is fairly self-explanatory: Every user will have a Cadet Number and Password that, stored in the Cnum and Pwd column respectively. The Cadet Number will relate to their Winchester Number, which from the MOD Database system for managing cadets that works at a higher level than the scope of CadetLink – this was a feature requested by the Client. The Users Password will be set by a CFAV when setting up the account to a default value, the User will then be asked to change it when they log in for the first time. The value Stored for the Password will be the hashed and salted version of what the user enters

The columns fname, lname, and rank store the first name, last name, and the Rank of the current user respectively. I decided to split these up so they can be updated easily, although first name and the last name shouldn’t need to be changed if initially correctly entered their rank may. Another reason for keeping the First name and the Last name separate is so for smaller screens I could have the option of only displaying one to save space on the screen.

Troop and section are the Groups the user is part of, for example troop: Chard, section: 1. They will both have a separate look up table.

The Profile pic URL stores the location of the users Profile Picture.

**Event Tables:**

***Note –*** *when I refer to an Event this means either a regular training session or a field day or an annual camp, if necessary, this will be specified.*

The event tables are there to store events and their relevant information such as the event name, the time it starts and finishes, and the start and end date, if it is a one-day event like a regular weekly training session then the end date is not required, and the system will be smart enough to understand this and won’t display an end date. Other data like the Orders documents will be stored as well. From this I plan to write a program that reads and interprets the data so it can produce a condensed note form and display the relevant information to the relevant people.

## Log in and Initial User system

Diagram

Description automatically generated

Technical Explanation

The User will be presented with a GUI as described in the User Interface section with two input fields for their Cadet Number and Password. Once entered I plan to use JavaScript to locally validate the entered data to ensure it is in the correct form as JavaScript is quick. Once validated via java script the data will be sent off to a PHP page where the data will once a gain be validated, this is as JavaScript is run locally can be turned off this would prevent a security issue if it was turned off and data was sent to the data base. If at any stage of validation, the inputted data fails it will be sent back to the user with a corresponding error massage.

Once passed validation the Cadet number and password will be checked against the data base to see if they match. I plan to hash the passwords I store so I will have to use something like PHP’s password\_verify function depending what hashing function I use.

After verifying the details, the necessary data for that users will be collected from the database a loaded into PHP Sessions Variables to be used for other functions

After this the user will be redirected to their dashboard.

Intended use

1. Known users will use the details provided by their CFAV’s to log into their accounts on the App
2. Differentiate an admin user from a non - admin user

Rational

The intended user of this site is the members of a cadet force; a login system stops people from outside that organisation. This will help guard against malicious acts against the website as hopefully only people known to the admins in real life will be able to use the app. Without a login system users could not be distinguished and as such systems such as the kit request system would not work as designed and would have to redesigned and in my opinion would not work as well.

This is also where the app finds out whenever this is an admin or non-admin user, I cannot think of another way of Differentiating the users that doesn’t require some form of Login systems

This system can also be expanded if multiple cadet forces use the app, adding such a system should be fairly simple, allowing quick and easy growth to the app.

**Requirement(s) met:**

* **Nº 8:** **Differentiate between admin and non-admin users**
* **Nº 9: App is secure**

## Central dashboard

Diagram

Description automatically generatedTechnical Explanation

(Basic and Full Dashboard System) Initially the dashboard will be a simple central page where links can be found to all the relevant pages to the user other pages. The page will check the session variable set up in the login process, decide if the user is an Admin or not and then display the correct links to the relevant functions page

(Only Full Dashboard System) The system will pull data from the events table and then display in order of which event is the closest to current date display the relevant information for each event also displaying a button to confirm attendance. Users will then be able to ‘flick through’ events using arrow buttons.

Intended use

1. Diagram

   Description automatically generatedUsers to use the Dashboard to Traverse the app, accessing the systems they need.
2. Users to view small packets of useful data relevant to them at the time of logging in.

Rational

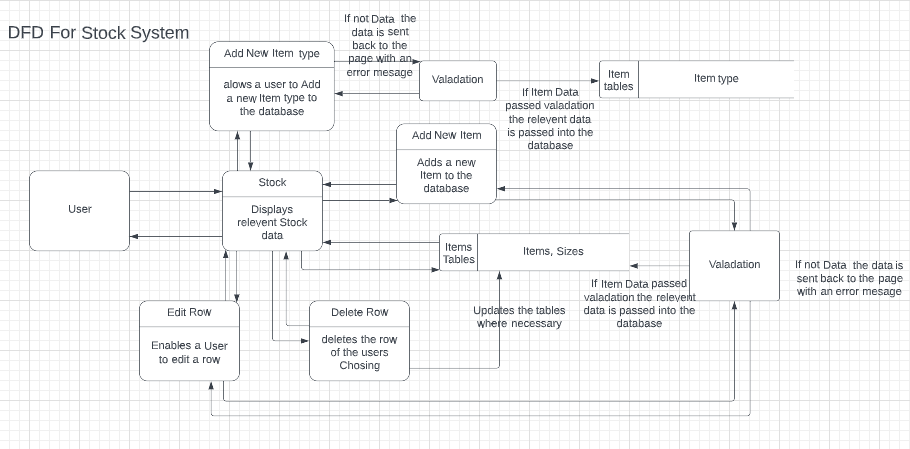
The idea of a central dashboard seems to be the most logical and functional way of connecting all the function pages together from the sites I have researched such as Edulink One. As such a system is common to most similar apps then the users should find the design familiar and understand how to use it immediately.

The data about events is designed to be quick access and remove the need to go further into the App to get the data, which will speed up the user’s usage of the app and improve user experience

**Requirement(s) met:**

* **Nº5: Central place for all functions and data transfer**

## Stock handling System



Technical Explanation

The Stock system is the main way an Admin will interact with the Backend database. The UI will be one like the function page as described in the UI section. Data that will be displayed will be the ID of the item; the items type; its sizes (displayed together in ss/ss/ss format); the number in stock, requested and number issued and the total like the client’s spreadsheet. This data will be pulled from the database on load by SQL query, if necessary collated together and then displayed in correct format for the device the user is viewing it on.

Users will be able to alter each number field by a click of a plus or minus button that will add one to that field, this will help for quick data entry. As What is being inputted is a known value there will be no need for validation, which will speed up the process. The user will also be able to change everything else, bar the item ID, in a more comprehensive separate edit page, this will go through validation though so will be a lot slower to use. The user will be able to add a new Item in a similarly looking page, this will also go through validation. The user will also be able to delete an item. The user will have the ability to add an Item type, this is to allow easy expandability as there might be an item that I don’t add to begin with.

There will also be a filter system to allow users to find the exact item they want or an all the items with a specific size, this will help with fitting out a new cadet.

Intended use

1. Admin User (CF QM) Can view and manage all items in the database can be seen and acted on in the correct and necessary ways

Rational

The stock system was requested by the client as one of the features to the app. By displaying the data in a table is designed so it is as similar to the user’s current excel spreadsheet.

The buttons to add to the number fields are designed to be a substituted to the ability in excel to just change it there and then whilst limiting human error: the thinking being that it is a lot harder to input an incorrect piece of data such as adding an extra digit as you can only increment the number by one each time.

The separate edit page is the simpler way to achieve that in terms of code, though it will slow down the editing process, hence why the buttons exist. This slow nature though will, I hope, prevent miss use and limit the human error.

**Requirement(s) met:**

* **Nº2: Kit Management System**

## Diagram, schematic Description automatically generatedKit Request System

Technical Explanation

The Kit Request system allows a user to make a Request for uniform or anything else stored in the database. The user will be asked for the relevant data such as the Item type and size the system will then create the request with that data and store it in a table. The system will also check the request and see if the request is in stock, updating the necessary number field in the items table and status field appropriately. If in stock, one (or the number requested) of these Items is classed as being “Reserved” and the request Is “to be issued”. If not in stock the one (or the number requested) is classed as being “Awaiting Ordered” and likewise the request is “Awaiting Order”.

The system will record the date and time of when the request was made and the user who made it. this is to help with prioritising who gets issued kit first.

The user will then be able to see the requests that they have made and delete it if they want to.

Diagram

Description automatically generated

The kit issue from Request System allows admins to view and the deal with request made by users in the aforementioned system. It will allow admins (aimed at the CF QM) to see the uniform all requests and filter through them, display the relevant data to allow the QM to process the request in a table as described by the UI section. The QM can then change the status of the request to the correct one once and during processing. For instance, if an order had been made and that item wasn’t in stock the status of the request would start at “Awaiting Order”. The QM could filter for all the Request “Awaiting Order”, Order all the Items needed and then change the status to On Order, when the stock arrived the status would be changed to “To Be Issued” and then once issued to “Issued” where the request would be archived and not show up on searched unless searched for. At any stage of the request the user who made the request will be able to view the request and check its status.

Diagram

Description automatically generated

Intended use

1. Users to make requests for uniform or anything stored in the data base
2. Admin user (intend user: CF QF) to be notified about request, be able to view all relevant data and act on them updating the status on each request until it is fully issued

Rational

Given that the inefficiencies in the current request system is part of what gave me the idea for this app the Kit Request System is a fairly integral part to this app. The challenge is that a lot of what happens about the request system is physical or out of the scope of this app; save programming a robot to go to the stores and retrieve the items much of what is done is down to the actual people such as the CF QM. What I can change though is how request get from cadets to the QM and thankfully that is where most of the inefficiency happen.

One of the problems with the current system is through information being passed through word of mouth, short notes, and emails, as such cadets doesn’t generally know what the QM wants to process the Request correctly. The QM’s Requirements that they need to process a request are the item, the size of the item, if necessary, the reason for the request (i.e. Lost old kit; Gown out of old kit or Was never issued); this is the requirement that is often forgotten, though the size forgotten a surprising amount.

The Kit Request System solves this first problem by having the users automatically recorded when the request is made, and the rest of the requirements as a required field when making the request.

Another problem is the Cadet not knowing the Proper name for the Item; this is solved out by having a drop-down selection for items with all the items stored in the database as options.

The main problem is requests getting lost in communication, this causes the most frustration. As CadetLink will have a one spot for requests there in no chance for the request email being lost in a sea of unrelated but equally important emails.

Another is communication, currently there is no central way of making a request. This System would resolve this issue by simply existing as it presents a clear way of making orders.

**Requirement(s) met:**

* **Nº1:** Kit Request System.

## Full User and User management System

Technical Explanation

The full user’s system will be a more comprehensive vision of a user system with the initial system being just a skeleton to make other function work. Like a stereotypical user system, the system will have the ability to Add, delete, and edit a user for any Admin account and from any users own account be able to change their password and possibly their profile picture as well as other personal details. The features that will be specific to this app will be tracking the number of requests they’ve made, and possibly storing their sizes for easy requests as suggested by a cadet who took the survey.

Intended use

1. To Allow Users to manage their own accounts
2. To Allow Admins to manage all accounts in their CF
3. To support other functions.

Rational

To have a user system for the app, there needs to be a way of managing the users without going to the database manually; this is why this system exists. The user having the ability to manage their own password is a mandatory for security, as if the only way to change passwords was through the admin management system, theoretically, the admin would know everyone’s password. Likewise, there has to be an admin password system as if the user forgets theirs then an admin can reset theirs.

**Requirement(s) met:**

* **Nº****7:** User System
* **Nº8:** Differentiate between admin and non-admin users
* **Nº9:** App is secure

## Event Management System

Technical Explanation

The Event Management will be able to create, manage and store events. This is so that cadets can see up coming events and know what they are going to be doing before the training session.

I plan to do this by having a table called events where I will store the relevant information for the event and notably the ability to upload a document, I intend to be Orders.

Intended use

Rational

## Help system to explain how a system works

## Test Planning

I plan to test in two separate ways: Iterative testing and post Development testing, this will check whenever I have met my goals throughout development and at the end of development. This will hopefully ensure I meet the development goals that I deliver a working system to the stakeholders.

### Errors I will look for and How I will look for them

*(In Order of how I Will check for them)*

1. Syntax Errors
2. Logical Errors
3. Runtime Errors
4. Validation Errors

### Iterative testing

After every major system that I develop or make changes to I plan to test the systems for the errors I have outlined in the above section, this will ensure that systems get finished to the level I want, not just get added. This is in line with the divide and conqueror methodology that I have mention I will use previously

I will first check for **Syntax Errors** in the PHP then in the JavaScript via the developer f12 menu. This is as a PHP Syntax Error will have a greater consequence on the overall running of the program compared to JavaScript and thus will be more obvious; in some cases, the PHP errors may even point to some JavaScript errors, especially when I’m using AJAX.

I will then check **Logical Errors**, starting with the stuff the PHP is set to display, such as table data, then what JavaScript displays, then CSS and finally HTML, for much the same reasons as in the previous paragraph

Then I will check functions on the page work, in other words Run time Errors

# The Software Development

# References

Army Cadet Website: armycadets.com

Edulink:

# Common abbreviations and phrases

* CFIT: Cadet Force Instructional Training
* CF: Cadet Force
* CFAV: Cadet Force Adult Volunteer
* NCO: Non-commissioned Officer
* Orders: A document my cadet force sends out weekly to communicate what is going to happen at that’s weeks training session
* QM: Quarter Master: handles stores i.e., uniform and equipment

// notes and at the moment rejected paragraphs

Rational

*Notes*

* *Most of the physical Items that are going to be virtually stored in these database tables are Military Uniform and Equipment – as such they all have a same size system (measured in cm) and NATO Stock Number, which is verry helpful. Size listed on the Items are all in centimetres and are formatted like 160/88 or 70/72/88 where each individual numbers represent a dimension like Chest or Inside Leg.*
* *When I Refer to an Item it is generally a piece of uniform but could also be something else military related like a TRF of a water Canteen.*
* **A Login and account and personalised Dashboard system** would allow Cadets and CFAV’s to access the information specific to them such as what section or troop they are in, what training they are going to undergo or lead next cadet session, what Kit and equipment they need to bring, dates and times of upcoming trips, and anything else that might be necessary to communicate to cadets without physical presence.
* **A Uniform and Equipment Database** would allow CFAV’s and senior NCO’s who are planning a lesson to know what Equipment is available and book that for their lesson. Would also allow cadets to order new uniform at any time of day or any location, automatically check if their size is in stock and present that data to the Quartermaster so it can be ordered and or issued the next cadet session. The computerised database would be able to run real time comparisons and remove much of the human error inherent to a paper database.

The operation of cadets, like the army itself requires a certain amount of logistical support, The difference is in Cadets we don’t have a whole Corp supporting us and although the efforts of our CFAV’s (Cadet Force Adult Volunteer) keep the process mostly up to date, something sometimes does still get forgotten or missed. Having to wait 4 months for new Kit Is not great, especially when remembrance parade is coming up.

This new app would be able to take a lot of the annoying tasks out of the logistics of cadets. Information about trips, Orders for this week, what Kit to bring (and whenever its short sleeves or not), scores for section competitions and a way for uniform issues to be logged would all be displayed in an easy-to-use central display. Other features like talking to an NCO could be implemented. Each cadet would have their own log in. the uniform issues would be logged, cross referenced against the Quartermaster’s database and would have the option to be displayed in an excel spread sheet.

Solvable by Computational method?

Use algorithms

* Solve the problem consistently
* Efficient
* Completes in reasonable amount of time