2022

**Josiah Rowden**

**A-level Project**

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

**End date: N/A**

Cadet Link

A picture containing text, cup

Description automatically generated

Logo

Description automatically generated with low confidence

Contents

[The Problem: 3](#_Toc121209367)

[The Current System: 3](#_Toc121209368)

[Stakeholders 4](#_Toc121209369)

[Why this Problem is suited to be solvable by computational methods 4](#_Toc121209370)

[Computational methods I will use: 4](#_Toc121209371)

[Problem Decomposition 4](#_Toc121209372)

[Divide and conquer 5](#_Toc121209373)

[The Analysis 6](#_Toc121209374)

[Similar Products 6](#_Toc121209375)

[Research 8](#_Toc121209376)

[Meetings 9](#_Toc121209377)

[Initial Survey 9](#_Toc121209378)

[Aims of the App: 13](#_Toc121209379)

[The planned Solution: 13](#_Toc121209380)

[Ideas for Potential Development 13](#_Toc121209381)

[The Design 15](#_Toc121209382)

[The Overall Design 15](#_Toc121209383)

[A modular Design: 16](#_Toc121209384)

[Success criteria 20](#_Toc121209385)

[Designing the Systems: 21](#_Toc121209386)

[*Universal Systems:* 21](#_Toc121209387)

[Colour Scheme: 22](#_Toc121209388)

[Specific Systems: 25](#_Toc121209389)

[Log in and Initial User system: 25](#_Toc121209390)

[Central dashboard 27](#_Toc121209391)

[Stock handling System 28](#_Toc121209392)

[Kit Request System 31](#_Toc121209393)

[Full User and User management System 35](#_Toc121209394)

[Event Management System 36](#_Toc121209395)

[Help system to explain how a system works 36](#_Toc121209396)

[Test Planning 36](#_Toc121209397)

[Errors I will look for and How I will look for them 36](#_Toc121209398)

[Iterative testing 37](#_Toc121209399)

[The Software Development 38](#_Toc121209400)

[References 39](#_Toc121209401)

[Common abbreviations and phrases 40](#_Toc121209402)

[Requirements 42](#_Toc121209403)

# 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.

1. **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.
2. **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.
3. **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.
4. **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 | Potential End User: Contingent commander | Would use to issues orders and post information about trips | Once a week in person, email |
| Lt Howson | Potential End 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 | Potential End 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

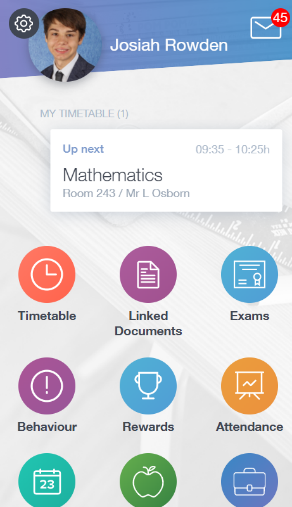
Figure 1 initial concept of Ideas for App

### Divide and conquer

I have split each section into different 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.



How 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.

Graphical user interface, application

Description automatically generatedA screenshot of a phone

Description automatically generated with medium confidenceAs 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 number 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*

## Research (review)

Part of the system will require some knowledge of the stock it will handle. This will be mostly British Army uniform, there is a system to how kit is documented in the British army, things to note are:

* Text, timeline

  Description automatically generatedNATO Stock Number (NSN): According to the NATO website (https://www.nato.int/structur/ac/135/faq/faq-e.htm#Q1):
* Sizes: for the army Personal Clothing System (The uniform used by my cadet force and the rest of the British army; abbreviated to PCS) have many different sizes and size type for an item, for instance a JACKET, COMBAT has two sizes: Height and chest whereas a TROUSERS, COMBAT, has three: inside leg, waist, and seat. All sizes are in centimetres and are just simply stated as numbers with forward slash to separated them i.e., my shirt is 170/88. Other pieces of uniform have size as words that roughly translates to equivalent to their number counterpart, i.e., a small UNDERSHIRT, COMBAT is size 160/80. Other items the system might need to store like patches will not have a size. For more information see the MOD Uniform chart in the references section.

## 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

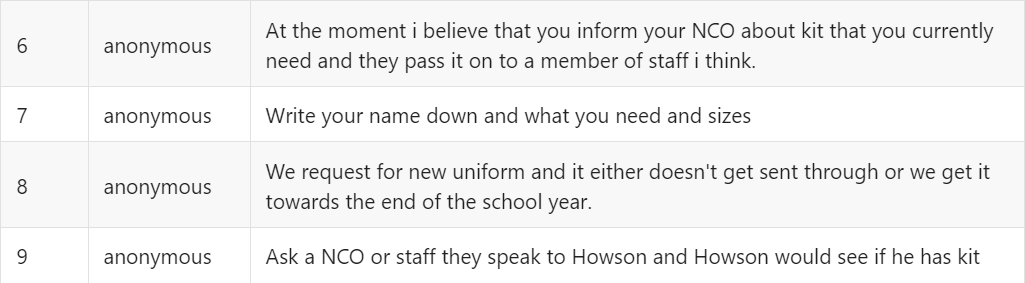
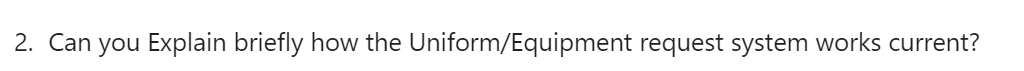
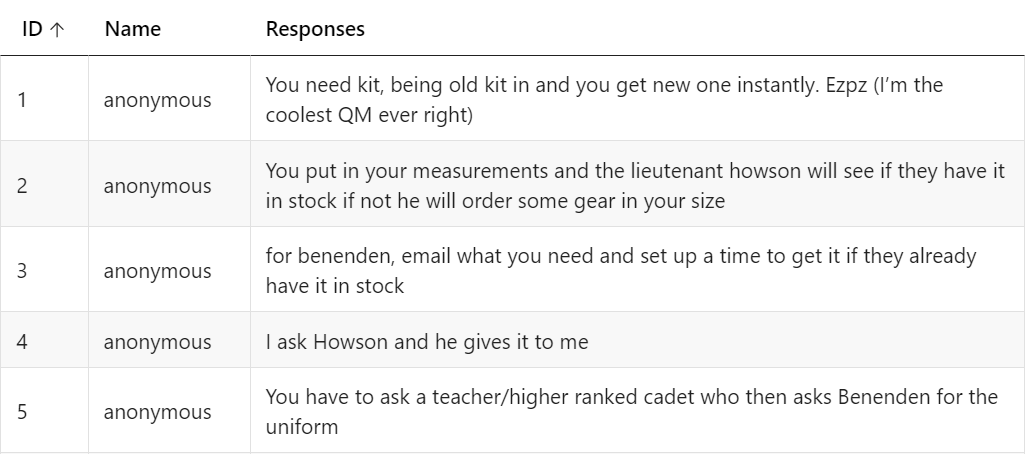
I wanted to conduct a survey of cadets (the end users of the app) to validate what I felt about the current system and gauge interest in a cadet app. I also asked them about any features they might want. I have had 10 responses at time of writing which is just under a third of the cadet force, so although 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. Certainly Possible
3. Certainly Possible
4. What you Outline here already exists, it’s called Orders, the trick will be getting the Orders to you before Thursday training Session and as for implementing it into the app it is certainly possible to add an upload able file option
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.

## Aims of the App:

Based on the analysis of the problem I have come up with 3 aims for the app; the aims are in order of importance.

1. **To improve the Kit request System for Cadets and the CF QM**
2. **TO improve and augment Stock management done by CF QM**
3. **To Improve information transfer and Communication in the CF**

**Rationale Behind each aim:**

1. **Based on the initial discussion with Stakeholders (Lt Howson and Cpt Aggrey)**
2. **Based on the initial discussion with Stakeholders (Lt Howson and Cpt Aggrey)**
3. **Based on survey of cadets (PEUs) and further discussed with stakeholder (Lt Howson)**

## The planned Solution: (Review)

I plan to solve the problems and achieve the aims stated by developing a web-based app with a back-end relational database that.

**Rationale:**

**A web-based solution will make the solution accessible to all PEUs given that they have a device that can connect to the internet, this is something all PEU have access to be it a personal or a school device.**

**A back end Relational data base will store data in a more accessible way and allow the app to manage data and make a lot of automated data changes based on a single input.**

### Ideas for Potential Development

**Here I will list all the Ideas I will consider into developing into full app**

**Key:**

* 1. **Brief description of concept**
  2. Aim Achieved as in above section
  3. Who suggested the idea

1. **A Uniform and Equipment Database**
   1. 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. The computerised database would be able to run real time comparisons and remove much of the human error inherent to a paper database or based database.
   2. Aim Achieved Nº: 2
   3. Myself, affirmed by Lt Howson and Cpt Aggrey, when I initially proposed the idea of a cadet app to them.
2. **Kit Request System**
   1. Would 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.
   2. Aim Achieved Nº: 1
   3. Myself, affirmed by Lt Howson and Cpt Aggrey, when I initially proposed the idea of a cadet app to them.
   4. I plan to develop this system as it solves the primary problem
3. **A Login and account** 
   1. Allows system to distinguish between users which will help in
   2. Aim Achieved Nº 1 (enabling)
   3. Myself, the idea to have users is central to other features affirmed by stakeholders
4. **personalised Dashboard system**
   1. 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.
   2. Aim Achieved Nº: 3
   3. Myself, affirmed by Lt Howson and Cpt Aggrey, when I initially proposed the idea of a cadet app to them.
5. **Events/ trips System**
   1. Allows CFAV’s to Add Orders to the App and create events that notify users when is happening next, in once central place.
   2. Aim Achieved Nº: 3
   3. Myself
6. **Kit Scanner** 
   1. Would be able to scan the barcode on kit and update the database
   2. Aim Achieved Nº: 1
   3. Requested by the stakeholder (Lt Howson)
7. **Talk to an NCO**
   1. Would allow users to talk to an NCO for issues such as kit and
   2. Aim Achieved Nº: 3
   3. Myself
8. **Help Page**
   1. a Page that shows users how to use the other systems
   2. Aim Achieved Nº:
   3. Myself
9. Virtual Pocket book
   1. A page where cadets can View revision material for cadets based on the lessons, they receive on Thursday training session
   2. Aim Achieved Nº: 3
   3. A cadet (anonymous)

*Some ideas in figure 1 are not written here as changes or planed changes not implemented in structure of the CCF made them irrelevant i.e., the section page.*

# 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 frameworks 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. Each section will be able to run without the other so will not compromise the effect of one system. 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:

Here I will list in order of development the systems I plan develop, giving:

1. The rationale of I am developing it and why I gave it its priority of development
2. Stages of development

Note: I will refer to systems as either Core or Enabling

1. Core – a system that directly works to achieve one or more of the stated aims
2. Enabling – a system that helps Cores System to function
3. **Log in and Initial User system (Enabling)**
   1. **Rationale:**
      1. Though this system is not a stated aim it enables a lot of the ideas that are for Aim 1. Kit requests are a lot easier if you know who is asking – this is also a requirement set by the stakeholder (Lt Howson) – and asking for the details of someone every request seems a bit redundant and annoying for the user. By having a login system, I can also keep people outside the cadet force from accessing the app and braking systems (either intentionally or unintentionally. It also makes giving more features to people who need it such as QM’s a lot easier as I can give their account special privileges. This is also similar to how Edulink does it.
      2. As it will be used by many different systems, I plan to develop this first so when developing others, I know the relevant data structures I will need to use in those.
   2. **Stages of Development:**
      1. Use of HTML and CSS to create a GUIs for both phones and computers that have relevant input fields.
      2. Back-end database work to create necessary tables to store the username and password of a user and any other necessary data
      3. Use of PHP to validate the data sent from the user and using SQL via PHP PDO check that it matches with a known username and password: use of PDO to avoid SQL injection.
      4. Use of JavaScript for initial validation and error management system to speed up validation for better user experience and better error messages.
4. **Central dashboard (Enabling)**
   1. **Rationale:**
      1. This System again though not core to the aims of the app enables other functions. As a way to get from one system to another it will help tie all the other systems together.
      2. I plan to develop this before any of the Core Systems as it will act as a bridge to all functions, and I can easily test it
   2. **Stages of Development:**
      1. Use of HTML and CSS to create a GUIs for both phones and computers.
      2. Buttons to go to each function page.
      3. Back/logout button (PHP).
5. **Stock handling System (Core aim 2)** 
   1. **Rationale:**
      1. I plan to develop this systemas this is a system that is core to the first(enabling) and second(directly) aims of the program.
      2. This is the first core system I plan to develop as it is based on the main idea of the app, it will also enable the kit request system as you can’t really request kit if there’s nowhere to request kit from. It will be helpful to know the structure of stock tables before I try and design something that requests something from it.
   2. **Stages of Development:**
      1. Back-end database work to create necessary tables and relationship between them
      2. Use of HTML and CSS to create GUIs for both phones and computers.
      3. 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
      4. Create PHP/JavaScript/AJAX functions that allow modification and deletion of the rows in the database
6. **Kit Request Systems (Core aim 1)** 
   1. **Rationale:**
      1. This system is Core to the first Aim of the project.
      2. I plan to develop this after the stock system so I know the database structure of the data I will be using to make a request, check stock etc.
   2. **Stages of Development:**
      1. Back-end database work to create necessary tables and relationship between them (SQL)
      2. Font end page that allows users to request Kit (HTML, CSS) with use of input fields.
      3. Back-end page that checks if kit in stock or not and performs appropriate action based on that and returns appropriate to user message (PHP).
      4. System that shows Users’ requests and allows them to modify and delete them (PHP, SQL, AJAX/ JavaScript?)
      5. System that shows the Requests to an Admin and allows them to modify/ respond to the request. (Intended User the CF QM) (PHP)
      6. Use of JavaScript for initial validation and error management system to speed up validation for better user experience and better error messages.
7. **Full User System (Enabling)** 
   1. **Rationale:**
      1. This system will expand on the Log in and Initial User system, giving admins the ability to manage Users accounts and create new ones. This is necessary to have a functioning user’s system and hence as discussed earlier why a user system is important to the project it is an enabling system.
      2. I plan to develop this after the two core systems as it only effects the functionality of the whole app not those individual systems who should be able to use the initial user system for anything they need. This system is also not critical to aims of the app so if necessary, however undesirable the app can function without it.
   2. Stages of Development:
      1. Use of HTML and CSS to create a GUIs for both phones and computers.
      2. A system that displays relevant information about each user that are stored from in the database. (PHP SQL)
      3. A system that allows Admins to create manage, and delete user profile (PHP, SQL, AJAX JavaScript),
8. **Event Management System (Core aim 3)** 
   1. **Rationale** 
      1. The system will achieve aim 3
      2. Aim 3 is the last aim for a reason, it is more desirable than essential for the app to function, hence why I plan to develop this system after even some enabling systems.
   2. **Stages of Development:**
      1. Creation of relevant database tables and relationships between them
      2. Use of HTML and CSS to create a GUIs for both phones and computers.
      3. A system that displays relevant information on their dashboard 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.
9. **Help system (Enabling)** 
   1. **Rationale:**
      1. This would be useful so I and CFAV don’t have to explain to new cadets/CFAV’s how the app works; it enables the functions of Core systems by telling users how to use those core systems.
      2. I plan to develop this last as it doesn’t really affect the function of any of the other systems and it purpose can be achieved by other non-computational methods
   2. **Stages of Development:**
      1. Use of HTML and CSS to create a GUIs for both phones and computers.
      2. Development of Guides for each system.
      3. A button to send you to the relevant help page

#### Ideas I do not plan to develop:

In this section I will outline:

* 1. Brief reason
  2. More detailed explanation of why I don’t plan to develop that Idea

1. **Kit Scanner** 
   1. Too Complicated
   2. For this system to work I need to develop 3 things:
      1. integration of app with hardware(camera) on multiple devices with different architectures and different operating systems.
      2. An algorithm that reads barcodes off a live video
      3. A data base of all the barcodes of the military kit and what that corresponds with

None of this I had experience with, and this is enough complexity to make it a project of its own. while I can see this system would be verry useful I do not have the experience or time to make this system a reality, on top of this project. As such I do not currently plan to develop this but if a situation arises where I have the time, I will consider adding it.

1. **Talk to an NCO**
   1. Safeguarding issues
   2. As cadets is a sanctioned youth organisation there would have to be some kind of moderation of this system by CFAV’s if this system was created and used officially by the cadet force, which is adding work do be done by CFAV’s which is the opposite of the aim of this project. Technically it is possible but other more mature systems already exist and in use by cadets unofficially such as WhatsApp, Discord, or Microsoft teams. As such I do not plan to develop this system at all.
2. **Virtual Pocket book**
   1. Relevance
   2. The cadet training guides are already on our cadet SharePoint, whereas there are some issues of getting cadets access to said documents there would be similar problems with this solution I would imagen. essentially a system already exists why try and solve an already solved problem? As such I do not plan to develop this system at all.

## Success criteria

In this section I will outline:

The stages of success I will grade my development of the app

1. Systems that I have developed
2. Rationale of the criteria
3. Other comments
4. **Stage 1 – A bare bones app** 
   1. Systems I will have developed
      1. Log in and Initial User system (Enabling)
      2. Central dashboard (Enabling)
      3. Stock handling System (Core aim 2)
   2. This stage achieves core aim 2 and has some useful functionality to the client, however dose not have the full functionality that the client has asked for.
   3. This is not a desirable end state but acts as a good lower goal if I find the systems more complex than I expect and such takes longer time to develop as its sill provides a somewhat useful system to the client and a baseline if I have time to develop later.
5. **Stage 2 – A functioning system**
   1. Systems I will have developed (in addition to previous stage)
      1. Kit Request Systems (Core aim 1)
      2. Full User System (Enabling)
   2. This stage achieves 2 core aims and gives the client a fully functioning app giving him the functionality that he requested
   3. This is a desirable end state, despite not achieving all the aims it achieves the first two which are the main
6. . **Stage 3 – A full functioning System (all aims Achieved)** 
   1. Systems I will have developed (in addition to previous stage)
      1. Event Management System (Core aim 3)
      2. Help system (Enabling)
   2. This stage archives all 3 Core aims and creates a fully functioning app with all functionalities the client requested
   3. This is a verry desirable end state, but I am aware that this project Is quite ambitious and this may not be achieved.

Other Goals

My other goals for

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 rationalee 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.

## Designing the Systems:

I have split this section into two parts,

1. Universal Systems or concepts –
   1. Systems or concepts that will affect every other system
2. Specific Systems,
   1. Based on the Plan of development in previous section, specific functions that work to achieve an aim (or enabling that) that does not affect all other systems

## Universal Systems:

In each section I will outline:

1. Goals
2. Concept – including rationale
3. How this will affect the whole app

### Colour Scheme:

##### Goals:

#### Must be readable and not uncomfortable to look at

#### Link to Stakeholder Organisation - the Combined cadet force

##### Concept:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Colour Hex code | Colour Name | Sample | Usage | Rationale (Short) |
| FFFFFF | **White** |  | Will be Used for Text on elements with Dark Green back grounds and for back grounds of Elements I want to Emphasize | Clear colour, shows up well on dark background |
| 002B17 | **Army Cadet Green** | **Shape, rectangle  Description automatically generated** | Will be used for background of generic elements | from the army cadets’ website – fits with theme of cadets |
| 000000 | **Black** | **Shape  Description automatically generated with medium confidence** | For text where White isn’t appropriate | Clear colour, shows up well on light backgrounds |
| F5EA00 | **Army Cadet Yellow** | **Shape, square  Description automatically generated** | Used on text and background some Elements where I want to highlight them, for instance when a user hovers over a button | Bright shows up an emphasises well- taken from the army cadets’ website – ties in with Cadet theme |
| 032F52 | **Benenden (Ben) Blue** | **Shape, square  Description automatically generated** | Emphasize certain elements  Text and background | **Colour of one of the schools – requested by Lt Howson** |
| AE1E30 | **John Wallis (TJWA) Red** | **Shape, rectangle  Description automatically generated** | Emphasize certain elements  Text and background | **Colour of the other school – requested by Lt Howson** |

###### Rationale (Long):

Apart from the two colours of the schools I was given free reign of what the colour scheme should be, so I decided to base my choices of the official Army Cadet website (link in references section) to tie in with the two colours that was picked for me.

After taking Army Cadet Green, I put it into colour picker: (link in references section) and saw That the Other Two colours suggested in the split complementary section were basically the two school coloursTimeline

Description automatically generated

White was chosen because using as text, it for worked on the background of the Army Cadet Green, and the black was chosen because it works on a white background.

The yellow also comes from the army cadet website as I liked how it emphasised elements

##### Usage:

I will keep a Consistent Colour scheme across all systems governed by this table, any additional colour I have to add for a function I will detail in that specific function sections.

#### User interface:

#### Goals:

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

#### Must be Adaptable for All Devices

###### Layout

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

*Graphical user interface, application

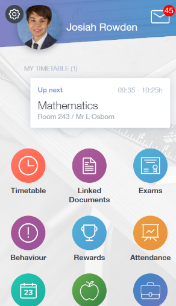
Description automatically generated*

## 

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.

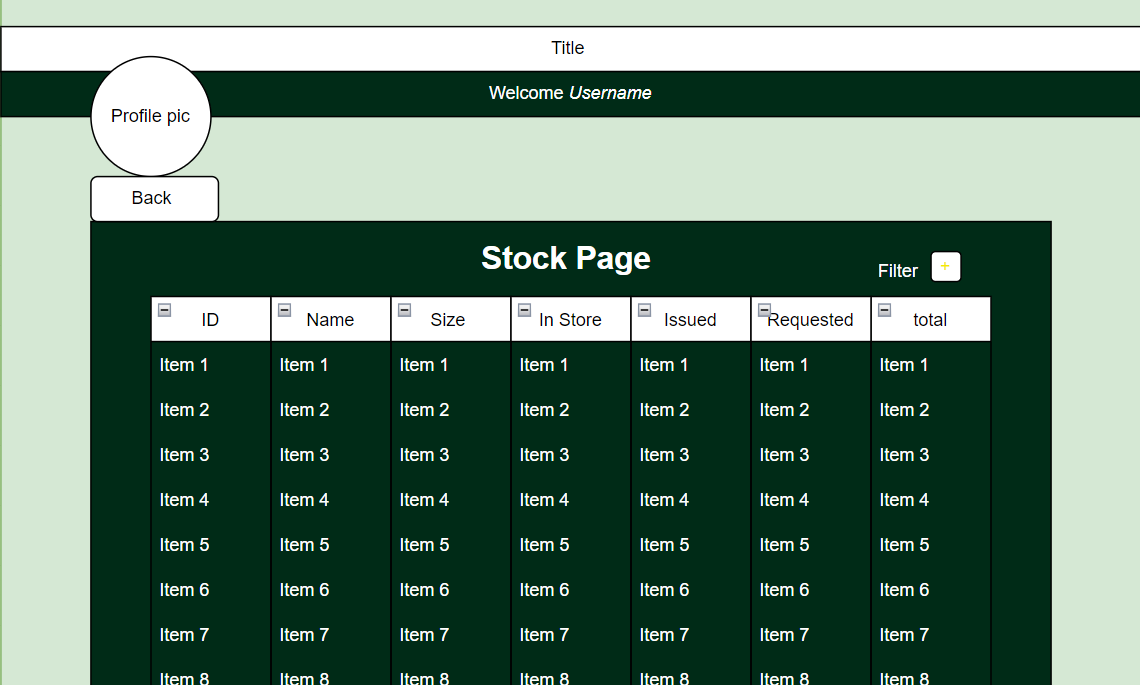
1Edulink Computer Dashboard Layout

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***

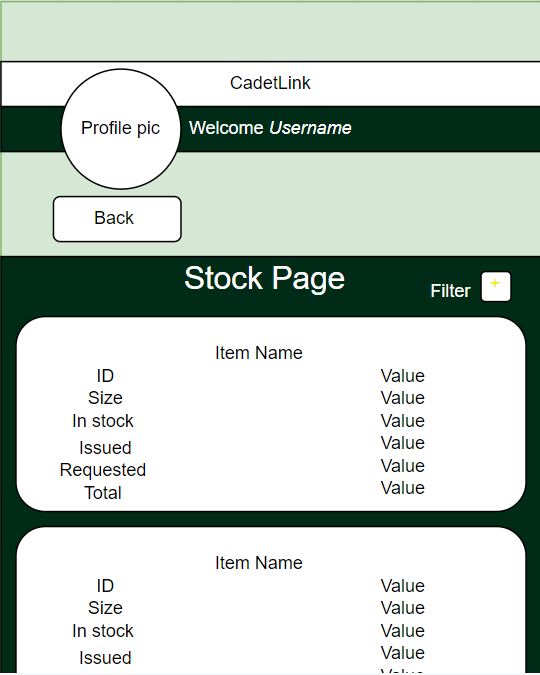
The overall Design is simple and to the point the table will be generated by an SQL query to the related table(s) in the data base

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

## Specific Systems:

In each section I will outline:

1. Concept
   1. Overview
   2. Intended users
   3. Key Variables (if appropriate)
   4. Technical explanation – how I plan it to work

### Log in and Initial User system:

#### Concept:

1. **Overview:**
   1. I plan to have a front-end page that has two input boxes that will be for the user’s cadet number and the Password, and a submit button, structured using HTML and styled using CSS
   2. This data will then be sent to a PHP process page that will check the inputted data with the data base and either direct the user to the relevant dashboard (if details match), or will relay an Error message (if details do not match)
   3. Data base table(s) that store user’s login details and any other data that is needed at this stage of development for upcoming systems such as a field that tells the system if they are a CFAV or not.
2. **Intended use** 
   1. Known users will use the details provided by their CFAV’s to log into their accounts on the App
   2. Stop any unwanted users from accessing the app
3. **Key Variables** 
   1. **Cadet number (Cnum):** instead of a simple username I plan to use a Cadet number that will be generated as when they join. This is something I recommended to the stakeholder (Lt Howson) as it allows a unique secure code that only the cadet and CFAV’s will know. He liked this idea and said that we could make it the Cadet number that is used to identify the cadets on the army’s Westminster system.
   2. **Password (Pword):** A standard Password that will be set by the CFAV when account is created but can be changed by the cadet (once full user system is developed).
4. **Technical Explanation**
   1. **Graphical user interface, application

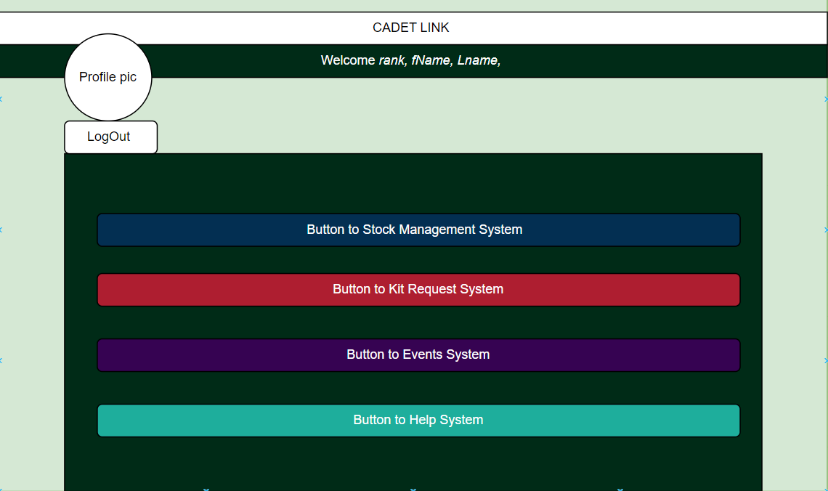
      Description automatically generatedFront end Design:** *Plan for front end (Right)* Has two input fields, which will highlight Army Cadet Yellow when Selected, and a submit button that will highlight in the same manner.
   2. **Diagram

      Description automatically generatedBack end Design:** Data will be sent to a PHP page using the POST method to ensure security of the data transmitted which will validate the data before checking it with the data base, . 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 decide upon to check the passwords match. If the details match a PHP Session will be set up inputting all the necessary data needed from the data base such as name and troop and the browser will be redirected to the relevant dashboard page using header(“address.htm”).
   3. **Data base Design:** I plan to use only one table, which I will call users that will store date related to the users of the app. Table Design with additional Comment column

| **Column** | Type | Comment |
| --- | --- | --- |
| **ID** | int(11) *Auto Increment* | Primary Key |
| **Cnum** | varchar(255) | Cadet Number |
| **Pword** | varchar(255) | Pas word |
| **fname** | varchar(255) | First name |
| **lname** | varchar(255) | Last name |
| **rank** | varchar(255) [**cdt**] | Default value is cadet to optimise for entry of new Cadets which cadet would be the starring Rank |
| **troop** | varchar(255) [**Chard**] | Default value is Chard to optimise for entry of new Cadets which Chard would be the starring Troop |
| **section** | int(11) [**1**] |  |

There is no Relational aspect at this point in development as there is only one table

## Central dashboard

1. **Overview:**
   1. Initially I plan to have a front-end page that has links to all other systems of the app if I develop the events system, I will add an upcoming events tab – that will be detailed in the events section.
   2. Depending if the User is a CFAV they will get taken to different pages from the login system, the CFAV page with more functionality such as access to stock Management system
2. **Intended use** 
   1. Users to use the Dashboard to Traverse the app, accessing the systems they need.
3. **Key Variables** 
   1. This system will not use variables being a simple HTML page**.**
4. **Technical Explanation** 
   1. **Front end:** I plan to have serval HTML buttons that will send the user to the specific functions page, this will take up the Central section of the display; each button will have a different colour so to differentiate them .There will be a header stating the rank and name of the person, this will be taken from the PHP Session Variables set up by the log in system so the user can check they are logged into the right account; a profile picture will also aid in this.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Colour Hex code | Colour Name | Sample | Usage | Rationale |
| 360352 | **Cherry Pie Purple** | **Shape, rectangle  Description automatically generated** | For a Button on the Dashboard | A Split Complement to Ben Blue Timeline  Description automatically generated |
| 1eae9c | **Mountain Meadow turquoise** | **Shape, square  Description automatically generated** | For a Button on the Dashboard | A picture containing timeline  Description automatically generated A complement TJWA red according to magecolorpicker.com |

* 1. **Colours:** I have chosen 2 new colours that I did not mention in the Colour scheme section:

## Stock handling System

1. **Overview:**
   1. I plan to have a central page that will display data from items tables in a table format,
   2. Separate specific functions will be on different pages that you get two from that main page i.e., edit row will be a sperate page to edit a specific row in the database
   3. Backend database to store Items,
2. **Intended use** 
   1. Admin User (CF QM) Can view and manage items in the database.
3. **Key Variables** 
   1. Values the will be displayed are, the item ID, NSN, Name, Size, the number in store, issued requested and the total number made up of the instore and Issued
4. **Technical Explanation:** 
   1. **Front end:** for large screens (laptops and above) I want to display the data from the database to be displayed in a table as I believe this is the simplest way to display large groups of data, it is also something the stack holder is used to (as they previously have used Microsoft Excel). For displays smaller (i.e., phones) which can’t fit all the data in on one screen there are two options: either force the phone into landscape mode or make a different display. As I am developing a web app, I believe it easier to just make a different display as such I plan to use a display based on the concept shown in the adjacent diagram with Items contained in their own sections and just listed. I also plan to have drop down filter menu that will allow users to sort by any variable displayed.

Graphical user interface

Description automatically generatedGraphical user interface, application

Description automatically generated

*Top Large Screen layout; Bottom Phone Screen Display*

* 1. **Back-end Database:** Because of how the army size their uniform (for more info see research section) there was two options for how I could store sizes: I could just have a field called sizes and just store the whole size in the table or I could split sizes and items into two tables, linking items with sizes in a one-to-many relationship. As I want to do filtering and sorting by sizes, I decided to split the tables up, this also means data will truly be atomic.

|  |
| --- |
|  |

| **Column** | Type | Comment |
| --- | --- | --- |
| **ID** | int(11) *Auto Increment* | Primary Key |
| **NSN** | varchar(255) | Secondary key, can be NULL |
| **ItemType** | int(11) |  |
| **NumIssued** | int(11) |  |
| **NumInStore** | int(11) |  |
| **NumRequested** | int(11) |  |

**Items table:**

Rationale:

I decided not to use the NSN as the Primary key as despite the fact that it is unique and is properly used as the primary key in the MOD stock database not every item stored by my CF will have an NSN ( this is also why it is set as s null field) and the format of the NSN could cause issues if I try and create an Fake NSN for nonstandard items hence why I plan to just go for a simple number incremented Primary key called ID, the NSN will be a secondary key for quick easy search though.

The ItemType is simply the name of the item, so humans understand what is represented by the record

The number fields are requested by the client and are similar to what he (Lt Howson) had in the Excel spreadsheet, adding the number of requests so the QM knows what to order in.

There is no total field as this can be calculated before being displayed on the PHP page.

**Sizes table:**

| **Column** | Type | Comment |
| --- | --- | --- |
| **ID** | int(11) *Auto Increment* | Primary Key |
| **itemID** | int(11) | Foreign key links to items table |
| **sizeType** | int(11) |  |
| **value** | int(11) |  |
| **unit** | varchar(255) *NULL* [**cm**] |  |

Rationale:

I plan to just have a separate ID to act as the primary key for the sizes table for simplicity’s sake, from my brief experience in SQL, composite keys are a pain.

The itemID field is a foreign key and links to the ID field in the items table.

The size type is what the value field actually represents i.e. height, chest, inside leg, seat etc.

I plan to convert all sizes to their number format for easy comparisons hence why the value field is an integer, I have not encountered any half as such in not using the floats variable type.

The unit field is defaulted to centimetres as this is the standard unit the British Army uses for sizes an all PCSs are using, however some, especially older kit, could use other units such as shoe size. If an item doesn’t have a size, like patches they will not have a size assigned to them

* 1. **Diagram

     Description automatically generatedBackend system page:**

The main page of the stock management system will display the data from the database tables, I plan to do this by pulling the data from the database using an SQL Statement using an inner join to get the data from both tables. Once the data has been pulled, I will format it, instance I want to display the sizes together, this will be done on the PHP page it will then be put into a HTML <table> and thus displayed to the user.

Once on the page the user will be able to edit add or delete rows through separate pages linked from the main stock page. These will simply take the inputs of the User, validate the data, and then send the data to the database, to update add or delete rows using SQL.

The filter system will take inputs and by using variable length where clauses in the SQL will filter the search by the parameters inputted by the user.

## Kit Request System

*I have split the Kit request system into three connected system as to deliver a full working that takes a request and handles it correctly, I need 3 distinct systems that work together to make one greater system:*

* *Make Request systems*
* *My Requests*
* *Request management systems*

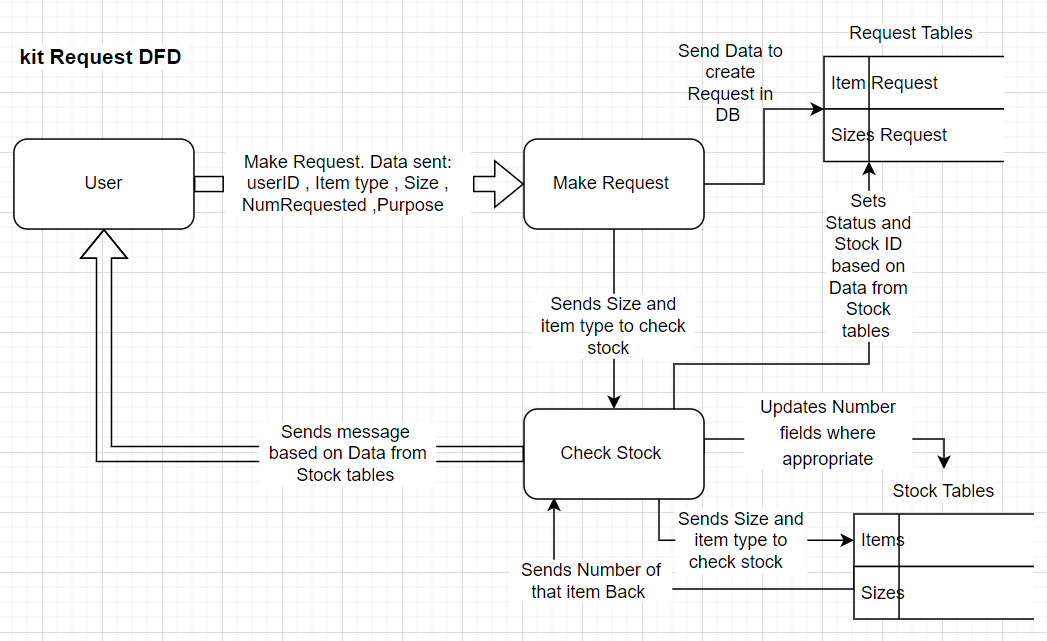
Make Request systems **Overview:**

1. **Overview**
   1. I plan to have a page with input boxes for the necessary fields needed to make a request.
   2. Backend page that validates the request, adds it to the database, checks it against the current stock.
   3. Backend database to store requests.
2. **Intended use** 
   1. Cadets to make requests of new or replacement kit they need or to request kit for lessons in aid of CFIT.
3. **Key variables** 
   1. Upon discussions with the Stakeholder (Lt Howson) the things he wants in requests are:
      1. Name of Item
      2. Size of item (if applicable)
      3. Reason why the request is being made
      4. Name of person making the request
4. **Technical Explanation:**

*Graphical user interface

Description automatically generatedGraphical user interface

Description automatically generatedFront End Mock-up designs; right Phone display; left large screen display*

* 1. **Front end:** similar to systems previously described, with 4 input fields and a submit button, and a button at the top to switch to the My Requests System; the sizes field is split up into variable number of input boxes (using java script) based on the inputted Item Type for reasons outlined in the research section and explained how I was storing sizes in the Stock System section. The layout is designed to be as simple as possible and self-explanatory – it should easily be navigable to the PEUs.
  2. **Back end:** once submitted the data will be validated on a separate PHP process page. If the data does not pass validation, it will send it back to the page with a relevant error message. If the data does pass 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 page will then add the request to the database, giving it an initial status. The system will then check the request and see if the requested item 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’s status is “to be issued”. If not in stock the one (or the number requested) is classed as
  3. being “Awaiting Ordered” and likewise the request is awaiting order of the item by the
  4. QM. The Stock ID record will also be added to Items Request table if an Item in the stock
  5. tables is matched with the requested item

Diagram

Description automatically generated

* 1. **Database:** This System will interact with the Stock tables ( items and sizes ), checking the request againgst the stock as previosly described but will aslo have 2 new tables for storing the itemRequests, requests and sizesRequest, split for the same reasons that the stock tables are. The latter serves much the same perpose as the sizes table of the stock system but for requests. The requests table holds the rest of the request data that the QM Requires for a request to be made and some ID to make comparisons easeir. The status field holds a seires of set phrases descriped in the Backend sections of this system.

itemsRequest table

| **Column** | Type | Comments |
| --- | --- | --- |
| **ID** | int(11) *Auto Increment* | Primary Key |
| **StockID** | int(11) NULL | Foreign key – links to items table |
| **UserID** | int(11) | Foreign key – links to the Users table |
| **ItemType** | int(11) |  |
| **NumRequested** | int(11) |  |
| **purpose** | varchar(255) |  |
| **status** | varchar(255) |  |

sizesRequest table

| **Column** | Type | Comment |
| --- | --- | --- |
| **ID** | int(11) *Auto Increment* | Primary Key |
| **itemID** | int(11) | Foreign key links to ItemRequest table |
| **sizeType** | int(11) |  |
| **value** | int(11) |  |
| **unit** | varchar(255) *NULL* [**cm**] |  |

My Request systems

1. **Overview**
   1. I plan to have a main page which displays all of that user’s requests.
   2. Pages that are accessed from the main page to edit and delete the requests
   3. Backend database same as described in the Make request section
2. **Intended use** 
   1. Cadets to see and edit/delete requests that they have made by the Make Request System
3. **Key Variables**
   1. The most important variables will be the Id of the request and the user ID
4. **Technical Explanation**
   1. **Front End:** the display will be much like the stock system in preemies but with Different field names of ID, StockID, ItemType, Size Number, Requested, purpose and status. These are all the fields of the Request table minus the Users ID as it will only be that users so that data is not needed to be displayed. The size will be displayed in the same way as in the stock system. The rationale behind using the same design as the stock system is to keep a familiarity between systems for ease of use and to create a sense of coherency.

## 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.

Rationale

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

Rationale

## 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:

Colour picker : <https://imagecolorpicker.com/>

MOD Uniform and Sizes: <https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/948535/s_1-6.pdf>

# Common abbreviations and phrases

* CFIT: Cadet Force Instructional Training
* CF: Cadet Force
* CFAV: Cadet Force Adult Volunteer
* NCO: Non-commissioned Officer – cadet who are in the CF for long will get promoted to an NCO and get given increasing amounts of leadership positions and responsibilities
* 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
* PEU: Potential End User
* RBS: Requested By Stakeholder

// notes and at the moment rejected paragraphs

Rationale

*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

## Requirements

|  |  |  |  |
| --- | --- | --- | --- |
| Requirement | Nº | Description | Rationale |
| 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 |
| User System | 5 | 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 | 6 | 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 | 7 | Only known user to the CF can use the app. | This is to prevent unwanted users interfering with the app either maliciously or unknowingly |

##### 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)*

##### **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.

Diagram

Description automatically generated